UNIVERSITY
INSTITUTE
OF LISBON

A Multiple Criteria Route Recommendation System

Rúben André Sousa Beirão

Master in Computer Science and Business Management

Supervisor:
Doctor Fernando Brito e Abreu, Associate Professor,
Iscte

Co-supervisor:
Doctor Vítor Manuel Basto-Fernandes, Assistant Professor,
Iscte

November, 2020

[This page has been intentionally left blank]

TECHNOLOGY
AND ARCHITECTURE

A Multiple Criteria Route Recommendation System

Rúben André Sousa Beirão

Master in Computer Science and Business Management

Supervisor:
Doctor Fernando Brito e Abreu, Associate Professor,
Iscte

Co-supervisor:
Doctor Vítor Manuel Basto-Fernandes, Assistant Professor,
Iscte

November, 2020

[This page has been intentionally left blank]

A Multiple Criteria Route Recommendation System

[This page has been intentionally left blank]

To the ones that love me and support me all the time.

[This page has been intentionally left blank]

Acknowledgements

I want to thank Professor Fernando Brito e Abreu for inviting me to collaborate in the "Sustainable Tourism Crowding" research project and helping me achieve the goal of obtaining a master's degree with his mentorship.
I want to thank Professor Vítor Basto-Fernandes for always keeping me on the right track and for the constant willingness to help.
I would also like to thank Professor João Carlos Caldeira for providing us with the *LaTeX* template for writing the dissertation and for always being ready to help when it was necessary.
I thank my family (especially to my mother, sister, brother and grandmother) for all the love. Also, for always believe in my capacities and making me believe that I can achieve the goals that I set myself, whatever they may be.
Thank you, dad, for looking after me from wherever the heaven is, you're always in my heart.
I thank all my friends that support me all the time, unconditionally, and that open my eyes if I need to. A special thank to my brothers Gaviria and Nuno. In the case of the friendships that the university gave me, thanks to Figos, Elmo, Johnny, Mica, Timmy, Danicas, David, Dinocas, Hussas, Ritas, Jorjuca, Ricky, Rodri and Tigas because you made every day and night at Iscte special.
A special thank to Figos, Duarte and João, my fisherman's and partners on the research, because they dedicated so much time helping me every single time that I needed.

Lisboa, November, 2020

Rúben André Sousa Beirão

[This page has been intentionally left blank]

Abstract

The work to be developed in this dissertation is part of a larger project called Sustainable Tourism Crowding (STC), which motivation is based on two negative impacts caused by the tourism overload that happens, particularly, in the historic neighborhoods of Lisbon.

The goal of this dissertation is then to mitigate those problems: reduce the tourist burden of points of interest in a city that, in addition to the degradation of the tourist experience, causes sustainability problems in different aspects (environmental, social and local).

Within the scope of this dissertation, the implementation of one component of a recommendation system is the proposed solution. It is based on a multi-criteria algorithm for recommending pedestrian routes that minimize the passage through more crowded places and maximizes the visit to sustainable points of interest. These routes will be personalized for each user, as they consider their explicit preferences (e.g. time, budget, physical effort) and several constraints taken from other microservices that are part of the global system architecture mentioned above (e.g. weather conditions, crowding levels, points of interest, sustainability).

We conclude it is possible to develop a microservice that recommend personalized routes and communicate with other microservices that are part of the global system architecture mentioned above. The analysis of the experimental data from the recommendation system, allows us to conclude that it is possible to obtain a more balanced distribution of the tourist visit, by increasing the visit to more sustainable places of interest and avoiding crowded paths.

Keywords: tourism, sustainability, multi-criteria algorithms, route recommendation system

[This page has been intentionally left blank]

Resumo

O trabalho a desenvolver nesta dissertação insere-se num projeto de maior dimensão denominado *Sustainable Tourism Crowding* (*STC*), cuja motivação assenta, essencialmente, em dois impactos negativos provocados pela sobrecarga turística que se verifica, nomeadamente, nos bairros históricos de Lisboa.

O objetivo desta dissertação é, então, mitigar esses problemas: reduzir a sobrecarga turística dos pontos de interesse mais visitados numa cidade que, além da degradação da experiência turística, causa problemas de sustentabilidade em diversos aspetos (ambiental, social e local).

No âmbito desta dissertação, a implementação de um componente de um sistema de recomendação é a solução proposta. Baseia-se num algoritmo multicritério de recomendação de percursos pedonais que minimiza a passagem por locais mais apinhados e maximizam a visita a pontos de interesse mais sustentáveis. Essas rotas serão personalizadas para cada utilizador, pois consideram as suas preferências (por exemplo, tempo, orçamento, nível de esforço físico) e várias restrições retiradas de outros microsserviços que fazem parte da arquitetura do sistema global mencionado acima (por exemplo, condições meteorológicas, níveis de apinhamento, pontos de interesse, níveis de sustentabilidade).

Concluímos que é possível desenvolver um microsserviço que recomenda rotas personalizadas e que comunica com outros microsserviços que fazem parte da arquitetura global do sistema mencionada acima. A análise dos dados experimentais do sistema de recomendação, permite-nos concluir que é possível obter uma distribuição mais equilibrada da visita turística, aumentando a visita a pontos de interesse mais sustentáveis e evitando percursos mais apinhados.

Palavras-chave: turismo, sustentabilidade, algoritmos multi-critério, sistema de recomendação de percursos

[This page has been intentionally left blank]

Contents

List of Figures xxi

List of Tables xxiii

Listings xxv

Acronyms xxvii

1 Introduction 1

1.1 Background and motivation . . . 3
1.2 Research question and objectives . . . 5
1.3 Contributions . . . 5
1.4 Dissertation outline . . . 6

2 Related Work 7

2.1 Introduction . . . 9
2.2 Research objectives . . . 10
2.3 Rapid systematic review protocol . . . 10
2.3.1 Studies choice criteria . . . 10
2.3.2 Inclusion and exclusion criteria . . . 10
2.3.3 Search strategy . . . 11
2.3.4 Search strings . . . 11
2.3.5 Selection proceedings . . . 11
2.3.6 Articles extraction analysis . . . 12
2.3.7 Validity threats . . . 14
2.4 Proposed taxonomy for related work . . . 14
2.4.1 Introduction . . . 14
2.4.2 Recommendation approach (RA) . . . 15
2.4.3 Interface (I) . . . 16
2.4.4 User information acquisition (UIA) . . . 16
2.4.5 Connection mode (CM) . . . 16
2.4.6 Data currentness (DC) . . . 16
2.4.7 Reactive recommendation (RR) . . . 17
2.4.8 Criteria for recommendation (CR) . . . 17
2.4.9 Sustainability (S) . . . 18
2.4.10 Overcrowding (O) . . . 18

2.4.11 Functionalities (F) . . . 18
2.4.12 Validation (V) . . . 19
2.5 Review of related work . . . 19
2.5.1 Alrasheed et al., *"A Multi-Level Tourism Destination Recommender System"*, 2020 [3] . . . 19
2.5.2 Nugroho et al., *"A Context-Aware Adaptive Tourist Recommendation System"*, 2019 [73] . . . 20
2.5.3 Aragoneses et al., *"Madrid Live: a context-aware recommender system of leisure plans"*, 2018 [57] . . . 21
2.5.4 Migliorini et al., *"Adaptive Trip Recommendation System: Balancing Travelers Among Point of Interest (POI)s with MapReduce"*, 2018 [69] . . . 22
2.5.5 Makedos et al., *"PLATIS: A Personalized Location-Aware Tourist Information System"*, 2013 [62] . . . 22
2.5.6 Batet et al., *"Turist@: Agent-based personalized recommendation of tourist activities"*, 2012 [13] . . . 24
2.5.7 Gavalas et al., *"A web-based pervasive recommendation system for mobile tourist guides"*, 2011 [41] . . . 25
2.5.8 Noguera et al., *"A mobile 3D-GIS hybrid recommender system for tourism"*, 2012 [72] . . . 25
2.5.9 Umanets et al., *"GuideMe – A Tourist Guide with a Recommender System and Social Interaction"*, 2014 [98] . . . 26
2.5.10 Cao et al., *"Implementation of Personalized Scenic Spots Route Recommendation System"*, 2018 [22] . . . 27
2.5.11 Amoretti et al., *"UTravel: Smart Mobility with a Novel User Profiling and Recommendation Approach"*, 2017 [4] . . . 28
2.5.12 Baraglia et al., *"A Trajectory-Based Recommender System for Tourism"*, 2012 [11] . . . 29
2.5.13 Mehmood et al., *"Design and Development of a Real-Time Optimal Route Recommendation System Using Big Data for Tourists in Jeju Island"*, 2019 [68] . . . 30
2.6 Summary . . . 32

3 System Design **35**
3.1 Sustainable Tourism Crowding project . . . 37
3.2 STC project use case diagram . . . 39
3.3 ROUTE microservice communication design . . . 39
3.3.1 ROUTE – POINT communication . . . 40
3.3.2 ROUTE - HEAT communication . . . 41
3.3.3 ROUTE - APP communication . . . 42

4 Theory and Decisions **45**
4.1 Sustainability classification . . . 47
4.2 Crowding classification . . . 49

4.3 Weather conditions . . . 50
4.3.1 OpenWeatherMap API . . . 52
4.4 Physical effort . . . 53
4.5 Routing algorithm approach . . . 54
4.6 Graphs and paths . . . 55
4.7 Route directions API . . . 57

5 Route Recommendation System Framework and Implementation 59
5.1 Development environment . . . 61
5.2 System architecture . . . 61
5.3 Route request . . . 62
5.4 System data inputs . . . 62
5.4.1 User preferences . . . 63
5.4.2 Route constraints . . . 63
5.4.3 POIs . . . 64
5.5 Graphhopper . . . 66
5.5.1 Graphhopper server . . . 66
5.5.2 Graph . . . 67
5.5.3 Nodes . . . 67
5.5.4 Edges . . . 68
5.6 Route generator . . . 68
5.6.1 Points of interest selection . . . 68
5.6.2 Weather request . . . 70
5.6.3 All scenarios . . . 71
5.6.4 Filter scenarios . . . 71
5.7 Route recommendation . . . 75
5.7.1 The route . . . 75
5.7.2 Route response . . . 77

6 Test and Validation 79
6.1 Validation environment and dataset . . . 81
6.2 Functional validation . . . 81
6.3 Use case testing . . . 82
6.3.1 Scenario contextualizing . . . 82
6.3.2 Test scenario #1 . . . 82
6.3.3 Test scenario #2 . . . 83
6.3.4 Test scenario #3 . . . 84
6.3.5 Test scenario #4 . . . 84
6.3.6 Test scenario #5 . . . 85
6.3.7 Sustainability validation . . . 86
6.3.8 Crowding test . . . 86
6.4 Validity threats . . . 87
6.5 Summary . . . 88

7 Conclusion **89**
7.1 Conclusions 91
7.2 Future work and limitations 92

Bibliography **95**

Appendices **105**

A OpenWeatherMap API Response **105**

B Point of Interest JSON Representation **107**

C Crowding Data **109**

D PointOfInterest Data **111**

E Use Case Scenario Descriptions between the APP and the ROUTE **113**

F Set Preferences and Constraints Activity Diagram **117**

G Crowding Graphics Analysis **119**

H Test Scenarios **123**
H.1 Test Scenario #1 123
H.1.1 Scenario #1 - *RouteRequest* example 124
H.1.2 Scenario #1 - *RouteResponse* example 124
H.1.3 Scenario #1 - Result representation 125
H.2 Test Scenario #2 126
H.2.1 Scenario #2 - *RouteRequest* example 127
H.2.2 Scenario #2 - *RouteResponse* example 127
H.2.3 Scenario #2 - Result representation 128
H.3 Test Scenario #3 129
H.3.1 Scenario #3 - *RouteRequest* example 130
H.3.2 Scenario #3 - *RouteResponse* example 130
H.3.3 Scenario #3 - Result representation 131
H.4 Test Scenario #4 132
H.4.1 Scenario #4 - *RouteRequest* example 133
H.4.2 Scenario #4 - *RouteResponse* example 133
H.4.3 Scenario #4 - Result representation 134
H.5 Test Scenario #5 135
H.5.1 Scenario #5 - *RouteRequest* example 136
H.5.2 Scenario #5 - *RouteResponse* example 136
H.5.3 Scenario #5 - Result representation 137

I *ROUTE* Project Class Diagram **139**

Annexes 141

I ETIS Core Indicators 141

II Use Case Diagram of the *POINT* Web Platform 145

[This page has been intentionally left blank]

List of Figures

1.1 International tourist arrivals and tourism receipts in % change (source: UNWTO, July 2019) . . . 3

2.1 Summary of the articles research procedure . . . 12

3.1 Overview diagram of the STC project microservice architecture . . . 37
3.2 Use case diagram of Sustainable Tourism Crowding (STC) project . . . 39
3.3 *ROUTE-POINT* communication . . . 40
3.4 *ROUTE* workflow for the replication of POIs database . . . 41
3.5 *ROUTE-POINT* communication . . . 41
3.6 *ROUTE* workflow for the replication of crowding database . . . 42
3.7 *ROUTE-APP* communication . . . 42

4.1 Sustainability dimensions (adapted from [45] [93]) . . . 47
4.2 United Nations (UN) sustainable development goals . . . 48
4.3 Crowding scale (source: [86]) . . . 50
4.4 Weather-climate information for tourist decision-making (source: [84]) . . . 52
4.5 *ROUTE-OpenWeatherMap* communication . . . 52
4.6 Graph vs Network . . . 56

5.1 *ROUTE* solution framework . . . 62
5.2 POIs database class diagram . . . 64
5.3 Sequence diagram of *Graphhoper* server creation and configuration . . . 66
5.4 Representation of an edge e between node a and node b . . . 72
5.5 Representation of the horizontal and vertical components on a treadmill . . . 74

C.1 Isocrowding lines of *Santa Maria Maior* parish council . . . 110

F.1 Set preferences and constraints activity diagram . . . 118

G.1 Level of crowding comparison for scenario #1 . . . 119
G.2 Level of crowding comparison for scenario #2 . . . 120
G.3 Level of crowding comparison for scenario #3 . . . 120
G.4 Level of crowding comparison for scenario #4 . . . 121

H.1 Mobile application screenshot for scenario #1 . . . 123
H.2 Mobile application screenshot for scenario #1 result . . . 126

H.3 Mobile application screenshot for scenario #2 126
H.4 Mobile application screenshot for scenario #2 result 129
H.5 Mobile application screenshot for scenario #3 129
H.6 Mobile application screenshot for scenario #3 result 132
H.7 Mobile application screenshot for scenario #4 132
H.8 Mobile application screenshot for scenario #4 result 135
H.9 Mobile application screenshot for scenario #5 135
H.10 Mobile application screenshot for scenario #5 result 137

I.1 Package diagram of *ROUTE* implementation 139
I.2 Class diagram of *ROUTE* implementation 140

I.1 Destination management indicators 141
I.2 Economic value indicators 142
I.3 Social and cultural impact indicators 142
I.4 Environmental impact indicators 143

II.1 Use case diagram of the *POINT* web platform (adapted from: [78]) 145

List of Tables

2.1 Inclusion criteria . . . 10
2.2 Exclusion criteria . . . 11
2.3 Execution of search string 1 in the chosen databases . . . 13
2.4 Execution of search string 2 in the chosen databases . . . 13
2.5 Execution of search string 3 in the chosen databases . . . 13
2.6 Final result of the articles extraction analysis . . . 13
2.7 Taxonomy criteria . . . 15
2.8 Classification of *"A Multi-Level Tourism Destination Recommender System"* . . . 20
2.9 Classification of *"A Context-Aware Adaptive Tourist Recommendation System"* . . . 20
2.10 Classification of *"Madrid Live: a context-aware recommender system of leisure plans"* 21
2.11 Classification of *"Adaptive Trip Recommendation System: Balancing Travelers Among POIs with MapReduce"* . . . 22
2.12 Classification of *"PLATIS: A Personalized Location-Aware Tourist Information System"* 23
2.13 Classification of *"Turist@: Agent-based personalized recommendation of tourist activities"* . . . 24
2.14 Classification of *"A web-based pervasive recommendation system for mobile tourist guides"* . . . 25
2.15 Classification of *"A mobile 3D-GIS hybrid recommender system for tourism"* . . . 26
2.16 Classification of *"GuideMe – A Tourist Guide with a Recommender System and Social Interaction"* . . . 27
2.17 Classification of *"Implementation of Personalized Scenic Spots Route Recommendation System"* . . . 28
2.18 Classification of *"UTravel: Smart Mobility with a Novel User Profiling and Recommendation Approach"* . . . 29
2.19 Classification of *"A Trajectory-Based Recommender System for Tourism"* . . . 29
2.20 Classification of *"Design and Development of a Real-Time Optimal Route Recommendation System Using Big Data for Tourists in Jeju Island"* . . . 30
2.21 Classification of the related work . . . 31

4.1 Sustainability data on the database . . . 49
4.2 Facets of tourism climate, their significance and impact (source: [30]) . . . 51
4.3 *OpenWeatherMap API* call example . . . 53
4.4 Physical effort intensity classification into *METs* . . . 54

4.5 Examples of common physical activities for healthy adults by intensity of effort required in *MET* scores (adapted from: [2]) 54
4.6 Terms used for networks and graphs (source: [10]) 56

5.1 Implementation list of the proposed recommendation system 61
5.2 POIs categories 65
5.3 POI metadata 65

6.1 First scenario 83
6.2 Second scenario 83
6.3 Third scenario 84
6.4 Fourth scenario 85
6.5 Fifth scenario 85
6.6 Comparison between the results of one algorithm that considers crowding and other algorithm that do not consider crowding 87

D.1 Content of the POIs database 112

Listings

5.1 Timestamp and POI example 76
A.1 Example of OpenWeatherMap API response 105
B.1 POI representation in JSON 107
C.1 Crowding data send by *HEAT* 109
H.1 Scenario #1 *RouteRequest* query 124
H.2 Scenario #1: *RouteResponse* 124
H.3 Scenario #2: *RouteRequest* query 127
H.4 Scenario #2: *RouteResponse* 127
H.5 Scenario #3: *RouteRequest* query 130
H.6 Scenario #3: *RouteResponse* 130
H.7 Scenario #4: *RouteRequest* query 133
H.8 Scenario #4: *RouteResponse* 133
H.9 Scenario #5: *RouteRequest* query 136
H.10 Scenario #5: *RouteResponse* 136

[This page has been intentionally left blank]

Acronyms

ACSM American College of Sports Medicine.

AjiML Aji Modeling Language.

API Application Programming Interface.

ETIS European Tourism Indicators System.

GIS Geographic Information System.

HTML HyperText Markup Language.

HTTP Hypertext Transfer Protocol.

ICT Information Communication Technologies.

IDE Integrated Development Environment.

ISTAR Information Sciences and Technologies and Architecture Research Center.

JSON JavaScript Object Notation.

MET Metabolic Equivalent of Task.

MOP Mixed Orienteering Problem.

MP Multi-Period.

MPP Multi-Objective Path Problem.

MSP Multi-Objective (Multi-Criteria) Shortest Path.

MTDOPTW Mixed Time-Dependent Orienteering Problem with Time-Windows.

OP Orienteering Problem.

OSM OpenStreetMap.

PA Physical Activities.

PM Project Management.

POI Point of Interest.

REST Representational State Transfer.

RR Rapid Review.

RS Recommendation Systems.

SQL Structured Query Language.

SRTM Shuttle Radar Topography Mission.

STC Sustainable Tourism Crowding.

TD Time-Dependent.

TRS Tourism Recommendation System.

TSP Traveling Salesman Problem.

TTDP Tourist Trip Design Problem.

TW Time Window.

UML Unified Model Language.

UN United Nations.

UNWTO United Nations World Tourism Organization.

Chapter 1

Introduction

Contents

1.1 Background and motivation 3
1.2 Research question and objectives 5
1.3 Contributions 5
1.4 Dissertation outline 6

This chapter aims to present the background, motivation and problems addressed in this dissertation. In addition, the research question and objectives are also presented, as well as the organization of the text.

[This page has been intentionally left blank]

Chapter 1
Introduction

1.1 Background and motivation

The tourism sector has been showing its essential role in the world economy over the years. Recently, in 2018, achieved the 9th successive year of growth in tourist arrivals and tourism export earnings, as it is possible to see in Figure 1.1 [56].

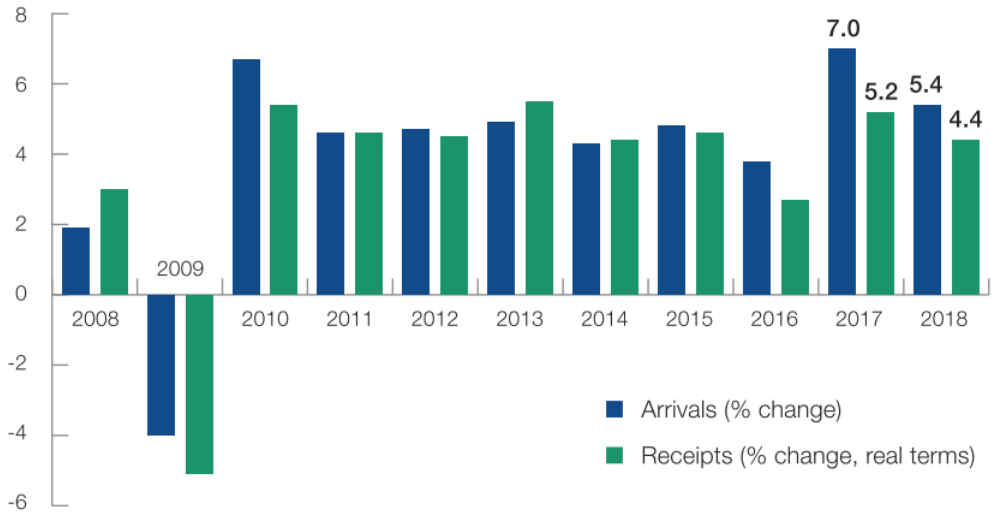

Figure 1.1: International tourist arrivals and tourism receipts in % change (source: UNWTO, July 2019)

Leisure, recreation and holidays represent more than half of the share amongst the various existing reasons for travelling, and the main channel where most persons search for tourism information turned out to be the internet [20, 52]. However, nowadays, the significant load of information available online have been seen as mostly overwhelming, because such a big list of alternatives can turn up a process of searching and analysis of information really complex [17]. Limitations as out-of-date information or the incapacity of users to find the information that suits their needs turn out into a long and frustrating process [52, 71].

With this personalization issue in mind, various Recommendation Systems (RS) have been developed and applied in different domains to help users taking their decisions, by providing them personalized content and, therefore, limiting the adverse effects of information overload when searching information over the Internet [1, 82]. Once that tourism is an activity strongly connected to personal preferences and interests [39], it was no exception to the appearance of personalized RS that assist users matching their needs with their travel plans [81]. In tourism area, RS are referred as Tourism Recommendation System (TRS) and they do not only assist tourists in a city as they also help to promote tourism in a city [96]. This is a privileged application field for RS because it leverages enormous opportunities to provide highly accurate and useful tourist recommendations that respect personal preferences, personal and environmental contextual parameters [53], typical recommendation tasks and the corresponding support

functions [43].

There's no doubt that tourism creates positive impacts that we should accentuate, but at the same time creates negative impacts that we should manage in a sustainable way to minimize its effects and prevent them from happening. These effects comprise economic, political, sociocultural, environmental and ecological areas [7]. Portugal and, mainly, Lisbon have not been an exception to this rule. Positively, the country and city have been, recently, consecutively awarded as World's Leading Destination and World's Leading City Break Destination, respectively, as it is proved in the World Travel Awards[1]. In Europe, south Mediterranean destinations led the growing tourism results with Portugal amongst the best performances [56]. Also, regular tourism growth became essential to the Portuguese economy thanks to its capacity to generate more revenue and create more jobs [33, 80].

On the other hand, tourism is bringing negative impacts on sustainability. Following the information given by the United Nations World Tourism Organization (UNWTO), it is expected an exponential growth in the number of international arrivals until 2030 worldwide. The problem is that there are several cities already suffering from the current number of tourists visiting them [29]. This phenomenon is called overtourism, and the European Parliament [77] describes it as "*the situation in which the impact of tourism, at certain times and in certain locations, exceeds physical, ecological, social, economic, psychological, and/or political capacity thresholds.*" In a simple way, it is the overcrowding caused by tourism. The overcrowding phenomenon occurs when there are not enough resources or physical infrastructures to support the increasing number of persons at a given location causing five major problems: alienated local residents, degraded tourist experiences, overloaded infrastructures, damage to nature, and threats to culture and heritage [31].

In this sense, Portugal defined a clear strategic plan for tourism, to develop until 2027 [21], which explicitly has sustainability as a guiding principle and defines ambitious objectives and goals to be achieved in three dimensions of sustainability: economic, social and environmental. This is where RS can have a crucial impact. As [8] states, RSs are a vital piece of smart tourism by assuring a sustainable development of tourism by promoting better tourist interactions, making better their quality of experience and, at the same time, the quality of life of the residents in that destination. Nevertheless, according to [13], only a few TRS focus on the sustainable aspect of Tourism. [59] declare that a system enabling routing of POIs and personal selection will help to prevent crowds because spreads tourists more uniformly across the destination region. Dynamic recommendations perform a valuable role combining the interests of tourists – who are interested in POIs that they may like according to their personal preferences – with the interests of the destination stakeholders - who are interested in increasing the visibility of the available POIs, particularly in the case of the less popular – an essential aspect of the sustainable tourism [16].

Therefore, the main problem to be dealt with in this dissertation is to mitigate the issue of local overtourism (minimizing tourist overload by causing people to spread around other places) while simultaneously promoting a more sustainable tourism. The work presented in this dissertation is integrated on a bigger project called STC, which global architecture will be fully

[1] https://www.worldtravelawards.com/

described in Section 3.1. This project will be specially deployed for the parish of *Santa Maria Maior*, Lisbon, with the goal of recommending tours that spread visitors geographically and promote more sustainable POIs instead of only the usual visited congested POI's. The expected effect is a reduction in overcrowding situations together with an improvement of quality in the touristic experiences and the sustainability of those places.

1.2 Research question and objectives

Throughout the research of this dissertation, the aim will be to answer the following question:

1. Is it possible to optimize pedestrian route recommendations, based on a multi-criteria approach, that promotes sustainability and avoids overcrowding situations?

The objectives to be accomplished for answering the research question are:

- Improve the STC research project architecture and connection between its services;
- Reformulate the objectives of the services that make up the research project and the information they exchange with each other;
- Define the JavaScript Object Notation (JSON) messages exchanged between services with a proper tool for modeling and design Application Programming Interface (API)s;
- Explain the significance of the integration of new tourist preferences, tour constraints and third APIs to the implementation of the route recommendation system;
- Implement the algorithm that combines different requirements providing a customized route that fulfills the problem constraints and user preferences;
- Elaborate hypothetical scenarios that cover the functionalities of the route recommendation system and the capacity of the algorithm to work with the existing restrictions and the user preferences;
- Obtain the route representations obtained as an answer to the scenarios represented in the mobile application.

1.3 Contributions

With the present work we aim to contribute in the following areas:

- Literature review and state of the art on RS in the field of tourism (see more details in chapter 2);
- In the STC research project, under development at Information Sciences and Technologies and Architecture Research Center (ISTAR), with the implementation of the main functionalities of *ROUTE* microservice (see chapter 5 for more details). This includes the availability of the project code, included in the STC open-source repository on *GitHub*, which contributes to future researches on the project or to other projects;

- Integration of the developed *ROUTE* prototype with other microservices within the scope of the STC architecture. This comprises the documentation and design of different APIs using a tool that permits future optimizations in an simple way (see chapter 3 for more details).
- The addition of new constraints to the route recommendation system implementation: the physical effort and the weather conditions (more details in chapter 4).

1.4 Dissertation outline

This dissertation is organized in seven chapters. In this chapter, the context, as well as the motivation and problem of this research, was presented. The research questions were defined together with the research objectives. Chapter 2 presents a discussion of what has already been produced in the literature, with the focus on the use of RSs in tourism. Chapter 3 explains the STC project, making clear the overall architecture. Chapter 4 clarifies the theory integrated into the formulation of important aspects of the implementation. Chapter 5 presents our implementation for recommending personalized routes. Chapter 6 presents the tests and validation for our solution. Chapter 7 enumerates the conclusions, and discusses the limitations and the proposed future work.

Chapter 2

Related Work

Contents

2.1 Introduction 9
2.2 Research objectives 10
2.3 Rapid systematic review protocol 10
2.4 Proposed taxonomy for related work 14
2.5 Review of related work 19
2.6 Summary 32

In this chapter, it is presented a comprehensive state of the art of TRS and what characteristics they have. In section 2.1, we have an introduction to other secondary studies in the area, its limitations and what was the protocol review used. Section 2.2 gives an overview of the goals of this review. Section 2.3 reviews the protocol used to select the studies that approach the same theme that is being studied here. Section 2.4 presents the proposed taxonomy to classify the related work. In Section 2.5, all the related work is reviewed and classified according to the taxonomy that was developed. In Section 2.6 is presented the conclusions taken from the analysis of the related work.

[This page has been intentionally left blank]

Chapter 2
Related Work

2.1 Introduction

The research process presented in this chapter was conducted during the period from April 2020 to June 2020 and aimed to identify the work that had already been done in the area of multi-criteria recommendation systems in tourism.

As a basis for this chapter, were analyzed various secondary studies about TRS [17, 43, 58, 63, 81, 96, 99]. All of these studies serve as good examples of what are the main characteristics of the RS; how they are classified; what services they offer; what criteria they take into account before a recommendation; how RS evolved and what obstacles and prospects there are to improve RS. Furthermore, those articles represent essential support for the construction of the categories included in the taxonomy explained later, especially because of the several relevant approaches of taxonomies and/or classifications of RS.

However, they all showed limitations to this particular study. [96] reviews the TRS but more with the aim of identifying the Information Communication Technologies (ICT) used in TRS development; [17] have a particular focus on TRS that exploit artificial intelligence at some point, which is not a special focus in this study; [43] are reviews specifically about mobile RS and so, the vast majority of the mobile services presented are outdated because the covered mobile phones and their technologies do not have comparison to the recent ones; [99] only overview and compare the several functionalities of systems that comprise trip plans; [63] categorize and describe multi-criteria RS, based on existing categorizations and taxonomies, but without a specific focus in tourism; [58] group the TRS in different categories under distinct criteria (e.g. personalization techniques, type of items recommended, etc.) but without making a profound analysis and with the limitation of giving outdated examples of TRS, because they are all from before 2010, so they do not consider the latest advances in the last years; finally, [81] explain in a very brief and generic way the characteristics of TRS with some examples without comparing different approaches.

In this study, a Rapid Review (RR) (also called rapid systematic review) was used. It is a secondary study dedicated to summarizing and simplifying the existing knowledge in a given research field during a limited time [97], with the aim of adopting a well-defined protocol and being systematic. So, leading a RR requires three phases (planning, performing and reporting) that contain different steps [24]. Some of the characteristics of the RR, as enumerated in [23], are: a problem restricted to a practical context; it is done during a limited period; a search strategy limited by the type of publication, year, language, amongst others; a selection procedure restricted by inclusion and exclusion criteria that can be conducted by a single person; an evaluation procedure and extraction process; and a synthesis presented in a tabular form. The used protocol of the RR is presented in the following sections.

2.2 Research objectives

The objectives of this review are to:

- Initiate a rapid systematic review of empirical research about the main characteristics, benefits and limitations of multi-criteria recommendation systems in tourism;
- Select a part of the relevant studies to do a "deep" review;
- Analyze the research methods used;
- Identify any gaps in present research in order to propose areas for further investigation;
- Provide a framework to position new research activities accordingly.

2.3 Rapid systematic review protocol

The current rapid systematic review protocol is organized and divided into seven subsections: 2.3.1 describes all the criteria used in the selection and inclusion of the studies for elaborating the literature review; 2.3.2 explains the inclusion and exclusion criteria for the studies; 2.3.3 explains the strategy for searching the studies; 2.3.4 defines the set of keywords that allow searching the most significant studies; 2.3.5 summarizes the procedure for selecting the studies; 2.3.6 describes the entire article extraction process explicitly; and 2.3.7 identifies the validity threats of the state of the art.

2.3.1 Studies choice criteria

The chosen articles to be included in this review must present empirical data or related theoretical studies that demonstrate the minimum quality according to what is described after that. Firstly, the inclusion of studies will not be restricted to any specific type of intervention, despite the fact that there is a predilection for real case scenarios and surveys in the final selection, in parallel with a preference that the participants of the studies come from tourism, computing, mathematics or engineering areas. Anyway, books, reports or regular papers are also within the set of publications that can be accepted. The outcome of interest from the literature that studies must include depends on the subject that is being explored. Though, the fundamental goals to be fulfilled correspond to the answers for the research questions.

2.3.2 Inclusion and exclusion criteria

In the selection of studies for this literature review, the inclusion and exclusion criteria shown in Tables 2.1 and 2.2 above were taken into consideration.

Table 2.1: Inclusion criteria

Criterion	Description
IC1	Only were considered studies that were published after 2000
IC2	Articles, books, papers or conference proceedings whose title and abstract are relevant or similar to the dissertation theme
IC3	Only research-based articles were considered

Table 2.2: Exclusion criteria

Criterion	Description
EC1	Unpublished articles or papers
EC2	Studies in other languages than English
EC3	Studies that are not within the area of Computer Science or Engineering or Tourism or Mathematics

2.3.3 Search strategy

As a first strategy, the following electronic databases were searched, because they were found to have the most suitable content for this dissertation theme: *Web of Science*[1], *IEEE Xplore*[2], *ACM Digital Library*[3], *ScienceDirect*[4] and *Scopus*[5]. Nevertheless, other search engines were also used, like *Google Scholar*[6], once the content of the electronic databases was not an exclusion precondition. This last referenced search engine was especially used to adopt the snowballing procedure, either forward snowballing - refers to the identification of new articles based on the works that referenced the article that was analyzed – or backward snowballing - refers to the identification of new articles based on the works that were referenced in the article that was analyzed [102]. The snowballing procedure complements the strategy because it allows minimizing the loss of some articles since the search strings are not perfect and were not searched in all existing databases [92].

2.3.4 Search strings

The search strings were defined by grouping keywords from the same domain with the logical operator "OR" and grouping the domains with the logical operator "AND". Some words were placed with the character "*" at the end so that it was possible to bring up whatever the derivation of the original word. The title, abstract and keywords of the articles, in the included electronic databases and conference proceedings, will be searched according to the following search keywords:

1. ("Tour*"OR "Travel" OR "Path" OR "Route") AND "Recommendation System";
2. ("Multi Criteria" OR "Multi Choice") AND ("Recommendation System"OR "Route Planning");
3. ("Overcrowding" OR "Sustainability") AND "Tour*";

2.3.5 Selection proceedings

The followed procedure for selecting the articles is summarized in the next steps:

1. Execute the search string;

[1] https://apps.webofknowledge.com/
[2] https://ieeexplore.ieee.org
[3] https://dl.acm.org/
[4] https://www.sciencedirect.com/
[5] https://www.scopus.com
[6] https://scholar.google.com/

2. Apply the inclusion criteria based on the title of the article;
3. Apply the inclusion criteria based on the abstract of the article;
4. Apply the inclusion criteria based on the full text of the article;

   After the selection of the first articles is done, then it was possible to execute step 5:

5. Perform the backward and forward snowballing procedure;

When executing step 5, steps 2, 3 and 4 are repeated to assure, once again, the quality of the articles. In the end, all the approved articles were stored and managed, throughout the duration of the dissertation, with the help of the *Mendeley*[7] software.

Figure 2.1 shows a summary of the research procedure that was used to find the articles that were analysed.

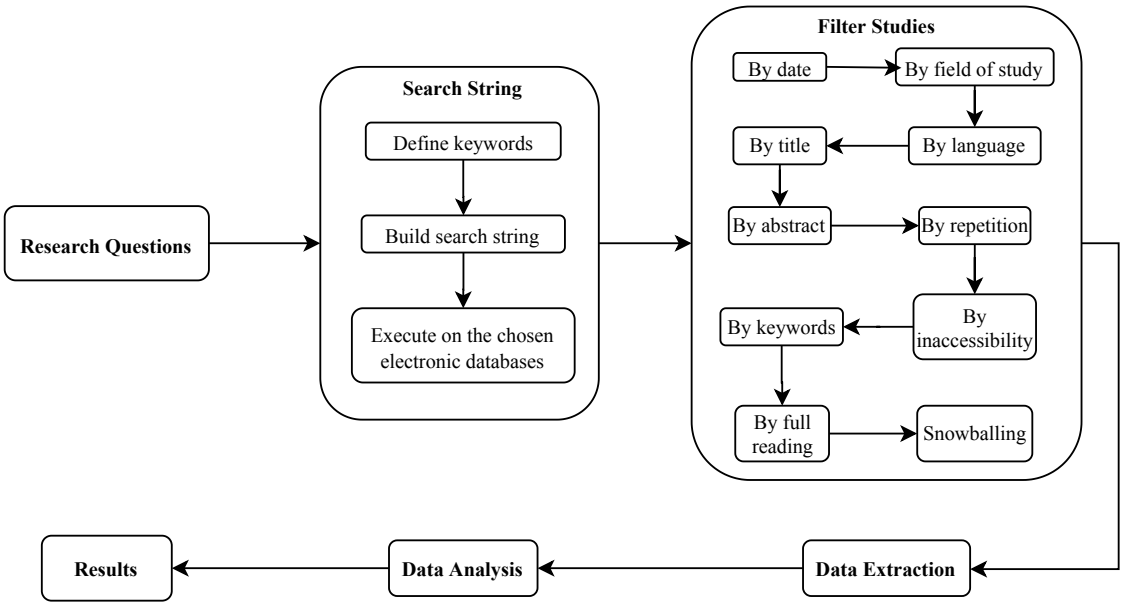

Figure 2.1: Summary of the articles research procedure

2.3.6 Articles extraction analysis

In this final step, it is possible to see the number of extracted articles in Table 2.6. A first execution of the search strings was done for each of the chosen databases, and frequently a significant amount of articles was returned. After, those articles were filtered according to the chosen criteria if the databases would allow it. Where is a *'NP'* means that it was 'not possible' to apply the criteria in that database. Where is *'NA'* means that the criterion was 'not applied' (in the one and only case that exists, happened because it was the first database to be searched and there were no repeated articles). First, the articles would be removed by date – all articles before 2000; then by field of study – any study that would not fit engineering's, computer science, mathematics or tourism area; then by language – eliminate all studies that were not written in the English language. After that, the articles were removed, first by title, then by

[7] https://www.mendeley.com

abstract – eliminate if there was no words or text that was meaningful for this study. Afterward, it was necessary to remove the repeated articles – either by comparing all the articles found in all the databases and with the different search strings or by finding articles that would explain the same TRS. Subsequently, all the articles that were inaccessible were also removed, and, finally, were eliminated after a full reading to check their meaningfulness.

Table 2.3: Execution of search string 1 in the chosen databases

	Web of Science	IEEE Xplore	ACM Digital Library	ScienceDirect	Scopus
("Tour*"OR "Travel" OR "Path" OR "Route") AND "Recommendation System"	336	245	850	1718	843
Removed by date	1	0	4	9	1
Removed by field of study	80	NP	NP	NP	215
Removed by language	0	0	0	0	25
Removed by title	202	213	833	1671	521
Removed by abstract	36	17	8	26	25
Removed by repetition	NA	9	0	2	19
Removed by inaccessibility	0	0	0	0	10
Removed by keywords	7	0	4	3	5
Removed by full reading	6	4	0	5	21
Total articles included	4	2	1	2	1

Table 2.4: Execution of search string 2 in the chosen databases

	Web of Science	IEEE Xplore	ACM Digital Library	ScienceDirect	Scopus
("Multi Criteria" OR "Multi Choice") AND ("Recommendation System"OR "Route Planning")	67	0	96	423	140
Removed by date	0	0	1	11	0
Removed by field of study	26	0	NP	NP	52
Removed by language	0	0	0	0	0
Removed by title	32	0	90	405	79
Removed by abstract	8	0	4	4	6
Removed by repetition	1	0	0	1	1
Removed by inaccessibility	0	0	0	0	0
Removed by keywords	0	0	0	0	0
Removed by full reading	0	0	0	2	2
Total articles included	0	0	1	0	0

Table 2.5: Execution of search string 3 in the chosen databases

	Web of Science	IEEE Xplore	ACM Digital Library	ScienceDirect	Scopus
("Overcrowding" OR "Sustainability") AND "Tour*"	5056	120	253	14454	6627
Removed by date	74	0	5	3818	183
Removed by field of study	4742	NP	NP	NP	6123
Removed by language	0	0	0	0	21
Removed by title	228	115	246	10608	294
Removed by abstract	9	1	2	19	3
Removed by repetition	0	2	0	3	1
Removed by inaccessibility	0	0	0	0	0
Removed by keywords	0	1	0	0	0
Removed by full reading	3	1	0	6	2
Total articles included	0	0	0	0	0

Table 2.6: Final result of the articles extraction analysis

	Web of Science	IEEE Xplore	ACM Digital Library	ScienceDirect	Scopus
Articles found	5459	365	1199	16595	7610
Removed by date	75	0	10	3838	184
Removed by field of study	4848	0	0	0	6390
Removed by language	0	0	0	0	46
Removed by title	462	328	1169	12684	894
Removed by abstract	53	18	14	49	34
Removed by repetition	1	11	0	6	21
Removed by inaccessibility	0	0	0	0	10
Removed by keywords	7	1	4	3	5
Removed by full reading	9	5	0	13	25
Total articles included	4	2	2	2	1
Total articles included through snowballing	2				
Extracted articles	13				

2.3.7 Validity threats

This rapid review undertaken in this chapter can have threats to its validity, as reported in [23]. One of the identified threats is related to the inadequate size and number of samples used for the analysis of the related work, which can lead to low reliability of the results. A total of thirteen articles extracted for analysis from all the articles found during the execution of the search strings might be low.

This leads to the second threat related to the search strings, as there are search strings that found a significant number of articles in different databases. This might be related to the lack of using more search terms that could narrow the final results or using different search terms that could have resulted in different articles.

In the process of extraction of the articles the number of articles found was reduced with the help of a set of criteria like explained before. In the process of applying that criteria, the articles that were found to be inaccessible were just excluded from the options, what constitutes another validity threat. A good practice would have been contacting the authors of the unavailable articles to obtain them before excluding.

2.4 Proposed taxonomy for related work

In this chapter, all the found related work is described. To better compare the work done, its relations and identify its gaps, is proposed one taxonomy. The taxonomy includes a set of characteristics used to identify the strengths and weaknesses of each article that serve to outline the conclusion, and through which the comparative study is guided in the form of an ordinal or nominal scale. The use of a taxonomy for this type of analysis is advantageous because it allows following a methodical approach to evaluating the proposed techniques. This approach is better when compared to an unstructured one because it allows for comparability and objectivity in the analysis. Each proposal is then schematically presented, containing the objective, the technical summary and the results of the taxonomic analysis.

- **Objective** - A brief description of the reasons for the creation of the system, framing it in state of the art;
- **Technical summary** - A non-exhaustive overview of how the system works, its virtues and limitations identified by the researchers;
- **Classification** - An assessment, presented in tabular form, according to the criteria defined in the taxonomy.

2.4.1 Introduction

In the course of researching the related work on multi-criteria recommendation systems in tourism, some specific criteria that guided the analysis of related work were identified. These criteria take into account the various possible system characteristics that a multi-criteria recommendation system in the tourism area can offer. It was also essential to refer to the context

of each study because it will guide the various techniques and implementation options proposed. There are criteria presented according to ordinal scales, while others are represented by nominal scales, which are explained below.

The criteria in our proposed taxonomy, summarized in Table 2.7, guide the classification of the related work.

Table 2.7: Taxonomy criteria

Criterion	Acronym
Recommendation Approach	RA
Interface	I
User Information Acquisition	UIA
Connection Mode	CM
Data Currentness	DC
Reactive Recommendation	RR
Criteria for Recommendation	CR
Sustainability	S
Overcrowding	O
Functionalities	F
Validation	V

2.4.2 Recommendation approach (RA)

RS are classified depending on the strategy followed to analyze the information of the user, filter the list of items and suggest recommendations.This is a criterion that is organized in the following categories on a nominal scale:

- **Content-based (CB)**: RS calculate the similarity between the user profile preferences and the items features (usually items similar to others that the user chose in previous interactions) and recommend the ones with a higher degree of similarity;
- **Collaborative filtering (CF)**: RS make recommendations by comparing a user to other users, based on groups, that have identical preferences and interests. It is necessary previous feedback from the users to know which items they have liked or disliked (e.g. which places tourists have enjoyed visiting);
- **Demographic-based (D)**: RS categorize users into pre-defined classes, according to their demographic profile characteristics (e.g. country of origin, age, level of studies, gender, etc.) and provide recommendations that match the standard preferences of the class in which they were introduced;
- **Context-awareness (CA)**: RS uses information concerning the user environment (contextual information) in order to change or predict the initial user preferences into the recommendation process;
- **Hybrid (H)**: RS that integrate two or more of the techniques mentioned before in order to avoid their limitations and compensate performance with the advantages of each other.

2.4.3 Interface (I)

This criterion refers to the interface used by the RS to interact with users and consequently, the type of platform for which it was deployed. The criterion is organized according to the following ordinal scale:

1. **Desktop application**: provides an interface from an application designed for desktops (requires downloading and installation on desktop);
2. **Web application**: provides an interface that allows easy access from any device connected to the internet, without the need for downloading or installing something;
3. **Mobile application**: the interface is provided by an application specifically designed to be used in smartphone/tablet;
4. **Hybrid**: the RS combines at least two of the previous ways of providing an interface.

2.4.4 User information acquisition (UIA)

This criterion refers to the ways in which the information about the user profile (e.g. personal information, preferences, needs, etc.) is acquired by the RS. The criterion is organized according to the following ordinal scale:

1. **Explicitly**: information may be explicitly acquired by direct interaction with the user (e.g. asking the user to fill a form, provide its locations, rate content within a given scale, etc.);
2. **Implicitly**: information may be inferred from the user's interactions with the system (e.g. mining the user activity, getting contextual information, etc.);
3. **Hybrid**: information might be acquired in both explicit and implicit ways.

2.4.5 Connection mode (CM)

This criterion is related to the ability of the RS to work in different connection modes, which means that it may require an internet connection, or not, to give recommendations. Its categories, on an ordinal scale, are as follows:

1. **Offline**: the RS only works in offline mode;
2. **Online**: it is mandatory to have internet connection in order to provide recommendations;
3. **Hybrid**: the RS works either online or offline.

2.4.6 Data currentness (DC)

This criterion is related to the different levels of data currentness that a RS might use as an input in order to suggest a personalized recommendation.

1. **None**: the RS do not use any data as an input for recommendations;

2. **Static**: the RS uses some static historical data (e.g. locations that are regularly known to be full of people, at a particular time of the day, would not be considered for routes that avoid overcrowding);

3. **Dynamic**: the RS uses dynamic data from the past (updated through the feedback given by the own system) or just present data;

4. **Combined**: the RS utilizes dynamic past and current data in a combined way.

2.4.7 Reactive recommendation (RR)

This criterion states if the RS has the capacity to generate recommendations and modify them by reacting to situational context changes (e.g. weather conditions, user location, traffic, data updates, etc.) without the need for user intervention. It is classified according to the following ordinal scale:

1. **Not reactive**: the RS is not capable of generating different recommendations by reacting to user context changes;

2. **Reactive**: the RS is capable of generating different recommendations by reacting to user context changes.

2.4.8 Criteria for recommendation (CR)

This criterion lists the criteria taken into account by the RS for deriving recommendation according to the following nominal scale:

- **Location (L)**: location is considered for the recommendation (e.g. user's current location, user departure location or user's end location);

- **Budget (B)**: the available budget that the user has to spend in attractions or POIs;

- **Distance (D)**: the distance between the user and the attractions, the distance between the attractions or the total travel distance are examples;

- **Time (T)**: the user's available time for activities (e.g. user choose a departure and arriving time to do a city tour, maximum trip duration, travel dates);

- **Transport mode (TM)**: considers different means of transport for travelling (e.g. walking, cycling, bus, tram, metro, etc.);

- **Weather conditions (WC)**: consider the weather conditions when recommending (e.g. if it is a sunny day recommends a city walk; if it is a rainy day recommends visiting museums);

- **User preferences (UP)**: the user interests or choices (e.g. POIs categories, ratings, feedback, etc.);

- **POIs info (POII)**: a different type of information about the POIs is considered (e.g. opening and closing days/hours, admission fee, type of POI/attraction, the average duration of a visit, etc.);
- **Mobility history (MH)**: analyses the user mobility history to recommend new items (e.g. avoid POIs already visited by the user);

2.4.9 Sustainability (S)

This criterion relates to the recommendations that may be suggested concerning sustainability issues, according to the following ordinal scale:

1. **Not important**: the RS is not concerned about sustainability issues when making recommendations;
2. **Indirectly**: in an indirect way, the RS makes suggestions that may promote sustainability;
3. **Important**: the RS concerns about sustainability issues and promotes it when making recommendations.

2.4.10 Overcrowding (O)

This criterion relates to the RS that may avoid overcrowding situations, according to the following ordinal scale:

1. **Not important**: the system does not try to avoid overcrowding;
2. **Indirectly**: in an indirect way the system provides recommendations that may avoid overcrowding;
3. **Important**: the system provides recommendations aiming at reducing overcrowding.

2.4.11 Functionalities (F)

This criterion lists the main recommendation tasks and the corresponding support functionalities offered by the RS, according to the following nominal scale:

- **Attractions suggestion (AS)**: recommendation of POIs (e.g. cities, museums, monuments, famous places, churches, etc.);
- **Tourist services (TS)**: recommendation of useful travel services (e.g. restaurants, accommodations, transports, events, information offices, etc.);
- **Social aspects (SA)**: includes social functionalities that enable the possibility for users to share their content (photos, videos, comments, covered routes, visited attractions, etc.) and to interact with each other (e.g. social networks connection);
- **Route planner (RP)**: the system provides an initial plan which the user can modify (add places, remove places, change the order of the places, etc.);

- **Route recommendation (RR)**: recommendation of routes that suites the user's preferences (e.g. the shortest path between two places, personalized tours for visiting various attractions, etc.) and takes into account different contextual factors (e.g. expected duration of the tour, opening and closing hours of the attractions, etc.).

2.4.12 Validation (V)

This criterion states if the system is validated, according to the following ordinal scale:

1. **Not validated**: the system was not validated;

2. **Validated**: the system was validated.

2.5 Review of related work

2.5.1 Alrasheed et al., *"A Multi-Level Tourism Destination Recommender System"*, 2020 [3]

Objective - A multi-level tourism recommender system framework is proposed to help users dealing with information overload and determine their most suitable destination.

Technical summary - The proposed system is a hybrid TRS with the aim of recommending one destination (a city) among a list of alternatives that fits the user's interests. It includes two main recommendation processes: first, providing the user with an attractive set of destinations that matches his preferences; second, ranking the list of destinations provided by the user from the most to the least recommended. Four main components compose the system. *User Profile* is a component that stores the user preference attributes (weather and attraction types) and the user constraint attributes (travel dates, accommodation budget, etc.). User profile information is acquired in a hybrid way: explicitly (e.g. requesting demographic information about the user and preferences) and implicitly (e.g. update preferences after rating a destination). *Group-based Popular Destinations* component provides a list of destinations, according to the user interests, for him to choose his preferred. *System Planner* collects information about the destinations using a set of Web Scrapers, takes the user profile as input, performs a one to one comparison between the destinations and the user profile attributes and returns a ranked list of destinations. *Web Scrapers* collects data about each destination by searching on other sources (e.g. websites) and then retrieves relevant data related to the user attributes. In the future, to complement the system, the authors want to include ratings of attractions or accommodation to the process of selecting a destination, want to implement the system and test its efficiency. In summary, the process starts by listing a few alternative destinations. Then, information about each destination is collected, and the list of destinations is ranked according to a specific set of user criteria. The destinations at the top of the list will be kept for serious consideration.

Classification

Table 2.8: Classification of *"A Multi-Level Tourism Destination Recommender System"*

Source	RA	I	UIA	CM	DC
[3]	H	-	3	-	3
RR	**CR**	**S**	**O**	**F**	**V**
1	UP; WC; B; POII	1	1	AS	1

2.5.2 Nugroho et al., *"A Context-Aware Adaptive Tourist Recommendation System"*, 2019 [73]

Objective - A tourist attraction recommendation system designed to produce recommendations based on tourist preferences, in real-time, for the tourist destination of *Yogyakarta*. It was proposed as a method to prevent tourist from wasting time gathering information for a trip that in the end, is not what was expected, and the tourist gets disappointed with the experience. Also, the goal is to develop a system that can react to contextual changes and reorganize the recommended plan in real-time, unlike other systems that already exist.

Technical summary - This is a TRS based on context awareness and two-way relationships. The process starts when a user inputs his profile contextual information into the system. That information corresponds to the following attributes: type (personal, group), demographics (gender, age, job, education), motivation (leisure, visit family and friends, business), location (distance, address and rating of each destination) and time (a day/month/year and if it was during the morning/afternoon/night when a user visited a destination). The system will process the input using an Analytical Hierarchy Process (AHP) algorithm to make a list of recommendations that match the user's preferences. The AHP algorithm weights the input criteria in order to generate recommendations. After, a user gets a list of top five filtered recommendations, a second process uses contextual information and feedback from another user to filter even more. The contextual information used is weather (season, temperature and conditions). The feedback input dimensions are rating and comment and correspond to the implementation of two-way relationships between users or between the user and the system. This way, a user will receive information about the real-time condition of the list of destinations and feedback about the destination and will be giving the chance to edit the list of recommendations according to that information, even during the trip (e.g. if he receives information that is raining, he can change the tourist attractions on the list). In the future, the researchers want to implement and test the accuracy and usability of the RS for users.

Classification

Table 2.9: Classification of *"A Context-Aware Adaptive Tourist Recommendation System"*

Source	RA	I	UIA	CM	DC
[73]	CA	-	1	-	3
RR	**CR**	**S**	**O**	**F**	**V**
2	L;T;WC;UP	1	1	AS;RP	1

2.5.3 Aragoneses et al., *"Madrid Live: a context-aware recommender system of leisure plans"*, 2018 [57]

Objective - Madrid Live is a context-aware recommender system (CARS) that suggests leisure activities in the city of Madrid. The main contributions this system wants to provide is to propose recommendations by considering additional contextual information (e.g. time, location, budget, weather or social position) because most of the RS focuses on recommending users with the most pertinent items without considering any more useful data.

Technical summary - In Madrid Live, users state their restrictions and preferences to their plans, and the system recommends the set of activities that satisfies these preferences together with the contextual knowledge. The recommendation process of the system is based on a Case-Based Reasoning (CBR) algorithm. The system stores every activity plan performed by each user and, in order to start a new recommendation, creates a query with the user preferences and contextual restrictions. User preferences are the types of restaurants, museums and activities he likes. The contextual information is the start and finish time for the plan; the location of the user obtained from the smartphone GPS; weather conditions, obtained from an external API, in the current location; the budget that the user expects to spend on the plan and an indication if the user wants or not to use public transport during the visit. After this, the query is compared to the descriptions of the stored cases. Those descriptions have the same type of information that comes in the queries. The comparison is made through similarity functions. A value for each comparison of the similarity between attributes is computed, and then the most similar cases are returned as a timetable that contains a collection of activities to be performed by each user. At this moment, Madrid Live recommends four activity types in the plans: museums, parks, points of interest and restaurants. If on the one hand containing contextual information on the recommendation process allows planning according to the user restrictions, it lowers the correspondence with the user preferences and makes it disappointing for the user. So, this system has the capability of generating explanations of how the plan is processed and selected. This means that a user understands better a recommendation and can analyze the discarded plans to decide if the context restrictions and preferences should be more flexible in order to find a more appropriate plan. To finish the subject, researchers evaluated the impact of the context features on the retrieved solution, but, in the future, also want to evaluate the quality of the explanations generated by the system.

Classification

Table 2.10: Classification of *"Madrid Live: a context-aware recommender system of leisure plans"*

Source	**RA**	**I**	**UIA**	**CM**	**DC**
[57]	H (CB+CA)	3	3	-	3
RR	**CR**	**S**	**O**	**F**	**V**
2	L;T;WC;B;TM; POII;UP;MH	1	1	AS;TS	2

2.5.4 Migliorini et al., *"Adaptive Trip Recommendation System: Balancing Travelers Among POIs with MapReduce"*, 2018 [69]

Objective - This TRS takes into account the balancing of users among different POIs in order to provide high-quality trip recommendations, yet maintaining the attractions not overcrowded. Recent TRS consider time, budget, personal interests of the user, needs and constraints, so it is all about the user viewpoint. On the other hand, the proposed TRS wants to consider also the impact that suggestions have on the status of the POIs, essentially avoiding big crowds on POIs.

Technical summary - The proposed TRS has two main components: an offline analysis of the user presence in different POIs and, working in parallel, a recommendation engine divided into two main stages. Users requiring a recommendation submit a query to the system with the coordinates where they want to start the trip, the time at which the trip will start, the trip maximum duration and the transport mode they will use. For the offline analysis, it is considered certain users' activity trips and reconstructed the set of POIs they visited. Those trips are stored in a database, and it is possible to process records of the visits at the POIs (e.g. the average number of visitors inside a POI at different times, average number of visitors), which can be consulted by the RS when there is a new user request. At the same time, in order to reply to the query, the system computes a series of values to drive the trip selection and perform multi-objective optimization of those trips. The system considers the user location, the POIs locations, the travel between each two POIs and the visit durations and assure that all of those constraints do not overcome the total duration of the trip. After that, tries to identify the best solution by introducing variations on the POIs order and considering the information of the offline analysis (visiting time depends on the POI occupancy). The values calculated offline are always updated in real-time because the estimation of the POI occupancy and the expected visiting time of each POI are updated continuously. In the end, the recommendation is sent back to the user mobile. The present TRS was evaluated using a real dataset from the city of Verona and showed consistent improvements over the paths usually followed by the tourists.

Classification

Table 2.11: Classification of *"Adaptive Trip Recommendation System: Balancing Travelers Among POIs with MapReduce"*

Source	RA	I	UIA	CM	DC
[69]	-	3	3	3	3
RR	**CR**	**S**	**O**	**F**	**V**
2	L;TM;D; T;POII	2	3	AS;RR	2

2.5.5 Makedos et al., *"PLATIS: A Personalized Location-Aware Tourist Information System"*, 2013 [62]

Objective - Personalized Location-Aware Tourist Information System (*PLATIS*) is a TRS, and its primary purpose is to provide tourists additional information about events and sights more

suitable for them, considering the knowledge of the destination place and data from a social network. Its goal is to supply end-users with a supplementary source of travel information, more personalized and better adapted to their profiles helping users to avoid information overload while planning their trip.

Technical summary - *PLATIS* is divided into two subsystems: an *Analysis System* which is about collecting data about the user and the destination place; and an *Inference System* that uses Bayesian Network and ontologies to match the user information with the events. Users start using the TRS by accessing *PLATIS* webpage and log in with their *Facebook* account, followed by request from the social network to read specific data from the profile. After the consent, *PLATIS* uses the *Facebook* profile to collect implicit information about its users (e.g. nationality, birth date, etc.) and about what users like to do or talk about (e.g. "Likes" or Wall posts which are parsed and searched by *PLATIS* for keywords). The only information explicitly extracted from users, through a questionnaire, are destination place, travel dates, trip purpose and accompanying persons. *PLATIS* uses various online webpages and *GPS* data from the user's device as sources to provide useful information to its users and agglomerates it into the profile of that user. The system extracts information that does not change over time (static information, e.g. museums, monuments, etc.); information that is more dynamic and changes over time (e.g. festivals, sports events, etc.); and environmental information (e.g. weather information). Considering that all the information was extracted, then it is sent to the *Inference System* to get enriched. *PLATIS* is then parsing the profile and categorizes every keyword into the ontology it belongs. The system has a Tourist Ontology with classes about the user information and a Travel Ontology with classes about the travel information. Once that all the information was categorized on the ontologies it is sent to the *Inference System* where the Bayesian Network connects the possibility of recommending a sight or an event to the information stored in Tourist and Travel ontologies. In the end, *PLATIS* presents all the results on a dynamic interactive map with tags, created with the help of *Google Maps API*, showing the suggested sights or events. In the case of using a mobile, the system can restrict its suggestions to a certain user-defined distance around the user's location by using *GPS*. *PLATIS* was tested and showed promising results that indicated a higher rating of its suggestion when compared to a system with random suggestions.

Classification

Table 2.12: Classification of *"PLATIS: A Personalized Location-Aware Tourist Information System"*

Source	RA	I	UIA	CM	DC
[62]	CA	2	3	2	3
RR	**CR**	**S**	**O**	**F**	**V**
2	L;T	1	1	TS	2

2.5.6 Batet et al., *"Turist@: Agent-based personalized recommendation of tourist activities"*, 2012 [13]

Objective - *Turist@* is a system whose goal is to help tourists analyze and filter the POIs and events of their interest when they arrive at a destination, rather than allowing users comparing possible tourist destinations as most RS do.

Technical summary - *Turist@* is an agent-based system that provides personalized recommendations on cultural activities according to the preferences of each user. Users start by interacting with the system (make specific queries, evaluate an activity that they attended or ask for a personalized recommendation) through the *User Agent* interface running on their phone. That user agent stores the user preferences, enabling reception of recommendations and search for activities. These preferences, stored on a centralized database, are first obtained from a questionnaire, but are consecutively refined and are automatically managed and updated through the analysis of user behaviour in the system (e.g. queries or evaluations). If the user requests a personalized recommendation, it is sent to a *Recommender Agent* that leads the personalization. This agent maintains a user profile for each tourist. Suppose the user chooses to search for activities. In that case, the information is sent to a *Broker Agent* that manages the search and mediates the communication between the *User Agents* and the *Cultural Activities Agents*. After this, the *Recommender Agent* or the *Broker Agent* communicate with the *Activity Agents*, which are entities that manage information of distinct attractions and have small local databases storing information about it, supplied by a local tourism office. Finally, the activities that better match user preferences are sent back to the *User Agent*, which shows them on the interface to the user. The *User Agent* can also provide proactive location-based recommendations (knows the position of the user in the city), warning users when they are near a cultural activity that may be interesting for them. Agent technology enables to divide the system into various small parts that can easily be executed, either on the client or server-side, and solve complex problems by cooperating proactively and independently. This way, the load of information on the smartphone can be decreased. Also, this technology gives a great degree of flexibility and scalability because it is easy to add new agents in it. The system was tested with real data (several activities information) from the city of *Tarragona* (Spain), modelled via the described agents, by users using mobile devices.

Classification

Table 2.13: Classification of *"Turist@: Agent-based personalized recommendation of tourist activities"*

Source	**RA**	**I**	**UIA**	**CM**	**DC**
[13]	H (CB+CL)	3	3	-	3
RR	**CR**	**S**	**O**	**F**	**V**
2	L;UP	1	1	AS	2

2.5.7 Gavalas et al., *"A web-based pervasive recommendation system for mobile tourist guides"*, 2011 [41]

Objective - The proposed mobile tourist platform, Mobile Tourism Recommendation System (*MTRS*), focus on assisting tourists in choosing personalized places to visit and so reducing their stressful process of selecting information by themselves. The distinction from other TRS is that before recommending, this system needs to incorporate data, attitudes, assessments or ratings of other visitors with identical interests and background information.

Technical summary - This *MTRS*, available through a web page or mobile application, records the users' interactions and uses it to feed their profiles. The information can either be inferred (e.g. items selected by the user, time spent in visiting specific content pages, etc.) or explicitly provided (the user creates an account to register personal information like age, gender, preferred leisure activities and places, etc.). Throughout the users' inferred and explicit data, *MTRS* periodically executes a k-clustering algorithm to classify users into separate groups sharing similar interests (stereotypes). Even though the server-side part of the system is identical for both types of users, mobile users are offered supplementary services as the system takes advantage of contextual information provided by mobile applications. In addition to user profiles and user reviews with common interests, various parameters such as the user's location, current time, weather conditions, and context of user mobility background are collected from third-party applications. The recommendation list for these users is constructed, taking into account the user's location and the distance to the POIs because the system prioritizes POIs located near the user´s position. Also, in this filtering process, POIs which opening times do not match the current time are eliminated from the list and weather conditions limit the choices. In the end, the available POIs are presented in the map with markers and when the user selects one to visit the route suggestion from the current location to the selected POI is graphically illustrated using *Google Maps API*. Lastly, the system was tested by a group of users in *Mytilene*, Greece, to validate its usability.

Classification

Table 2.14: Classification of *"A web-based pervasive recommendation system for mobile tourist guides"*

Source	RA	I	UIA	CM	DC
[41]	H (CF+CA)	4	3	2	3
RR	**CR**	**S**	**O**	**F**	**V**
1	L;D;POII;T; WC;MH	1	1	AS;RR;SA	2

2.5.8 Noguera et al., *"A mobile 3D-GIS hybrid recommender system for tourism"*, 2012 [72]

Objective - The proposed system, *REstaurants of Jaén* (*REJA*), was created to provide a personalized service to the tourists visiting *Jaén*, Spain, when looking for restaurants. The aim is to

have a TRS that helps tourists dealing with the massive amount of information that exists today on the internet.

Technical summary - *REJA* is a context-aware mobile recommender system (*CARS*) and is composed of three components: a *Recommender Server*, a *Geographic Information System (GIS) Server* and a *Mobile Client Application*. Through the mobile application, the users' login and their location are obtained from the mobile *GPS* receiver. The *GIS Server* stores the terrain datasets and provides 2D or 3D maps images according to the user locations. Then, on the *Recommender Server*, the users and restaurants knowledge is stored, and the recommendations are computed. The *GIS server* processes the coordinates and sends back to the user, via the open communication with the mobile device, an interactive 3D map of the surrounding area with various POIs that help the user navigation. The mobile device must download continuously, according to the user movements, a simplified 3D representation map from the remote GIS server, which means that without an internet connection, the main functionalities will be unavailable. At this point, users can then ask for a recommendation, if they provide the system with their preferences information (e.g. a distance that they are willing to travel and, according to their current locations). A request to the TRS is sent with those preferences and the exact locations. The system processes it and recommends the nearby restaurants that most probably fit the user's interests (the top-N recommendable items are computed within that distance, and the other items are ignored). Finally, with a rather positive evaluation of the application, the proposed system was implemented and tested, mentioning the ease of its utilization and the usability of the suggestions based on real-time locations.

Classification

Table 2.15: Classification of *"A mobile 3D-GIS hybrid recommender system for tourism"*

Source	RA	I	UIA	CM	DC
[72]	CA	3	1	2	3
RR	**CR**	**S**	**O**	**F**	**V**
2	L;D	1	1	TS	2

2.5.9 Umanets et al., *"GuideMe – A Tourist Guide with a Recommender System and Social Interaction"*, 2014 [98]

Objective – The proposed system aims to differentiate from other existent approaches for tourist guides that mainly focus on the recommendation of well-known touristic locations. Knowing that most of the tourists end visiting the same famous attractions of the cities around the world, this system pretends to guide people to consult interesting POIs that are not so visited or known due to the lack of public information.

Technical summary - *GuideMe* system is composed of four layers: the client application that makes the recommendation requests provides current user location (what means that is a

context-aware system) and where filtering criteria is defined (country, city, category and weather conditions); a recommender system connected to a *REST API* and a database to suggest new POIs based on the user's past actions (places that user said to be interested); an interface that connects the client app to the RS and is responsible for data security; a database layer where user data and POIs information are stored. It all starts when a user logs in (can choose the preferred social service to log in or sign up) and their current location is obtained (e.g. using *GPS* or *Wi-Fi* connections). The TRS uses that information together with the users' past visits and a list of the eligible users (those that have visited at least one location) for recommendations and processes a recommendation using the collaborative filtering algorithms. In order to find and recommend POIs, the RS consult a database regarding POIs and user data. Finally, a list of the nearby locations is outputted, sorted by decreasing preference of new interest places, according to the defined filtering criteria. Nevertheless, a significant limitation of this recommendation system is that, for performance reasons, recommendations run just once a day and it may not include all users (they are only eligible for new recommendations if they visited 5% more locations than the last time the recommendation was run). Users can then consult the locations they want to visit, consult locations they already visited or recommended and see the detailed information of each location. This system also uses the notification service of *Apple* to provide push information according to the user context (e.g. when another user is near). Finally, for most of the user features and design of the app, the usability and load tests achieved satisfactory results.

Classification

Table 2.16: Classification of *"GuideMe – A Tourist Guide with a Recommender System and Social Interaction"*

Source	RA	I	UIA	CM	DC
[98]	CF	4	2	2	3
RR	**CR**	**S**	**O**	**F**	**V**
2	L;WC	2	2	AS;SA	2

2.5.10 Cao et al., *"Implementation of Personalized Scenic Spots Route Recommendation System"*, 2018 [22]

Objective – A personalized route recommendation system based on scenic spots close to *Xiamen* University, China, is proposed in order to meet the needs of users because the authors consider that the traditional tourism industry is unable to meet the needs of users.

Technical summary – This TRS has three main functional modules: *user and system interaction module*, *recommendation system function module* and *scenic spot information background management module*. In the first module, a user can consult a scenic spot list consisting of a name, a picture, a category, an introduction and level of crowding. After this, users identify a scenic target spot they want to visit, a starting point for the route, and, optionally, some scenic spots to visit. Also, they are requested to rank the six existing categories of scenic spots: photo, shopping, eating, sports, reading, and history. The higher the classification, the higher the

priority of that category spots to visit. In the second module, some implemented functions will recommend routes according to users' options. If the user skipped the personalization section, the system makes a travel route through the most popular scenic spots. On the other hand, if the preferences were specified, the system recommends a personalized route that also takes into account the crowdedness of scenic spot by consulting the values of crowding stored on the database. The system was tested on *Xiamen* University, China, surrounding spots and returned different routes for different preferences.

Classification

Table 2.17: Classification of *"Implementation of Personalized Scenic Spots Route Recommendation System"*

Source	RA	I	UIA	CM	DC
[22]	CB	-	1	-	2
RR	**CR**	**S**	**O**	**F**	**V**
1	L;UP;POII	2	3	RR	2

2.5.11 Amoretti et al., *"UTravel: Smart Mobility with a Novel User Profiling and Recommendation Approach"*, 2017 [4]

Objective – Once there is a growing amount of information available online, the need for systems helping with personalized searches is also raising. The purpose of *UTravel* is to make tourist mobility smarter, by suggesting POIs according to user preferences, user behavior and group affinity.

Technical Summary - *UTravel* recommends POIs (e.g. shops, pubs, museums) that match user interests. Through the mobile application, users are able to see their position on a map and a set of surrounding locations with commercial or cultural importance, according to their preferences and the current context. The initial step on the process of recommendation is the creation of a user profile. When users run the mobile application for the first time, they must register an account where they provide the information that is necessary to build their demographic profile: gender, birth date and occupation. After the registration, users build their preferences profile by selecting and ranking their interest categories (e.g. cultural heritage, food services, entertainment). This system uses a multi-criteria technique to evaluate POIs taking into account: quality of services, cost, reachability, waiting time and overall rating. So, for the recommendation of POIs, the system uses the computed values and compare them with the user preferences profile. After the user logs in, a map centered in its location presents some POIs markers within a defined range (up to 10km). Those markers can be of three types: recommended POIs (generated by the system), POIs of the interest categories (they are not recommended but may be of the interest of the users according to their preferences) and services of interest (SOI). When users click on those markers, they can see detailed information about the location and description of POIs and SOIs. Also, they can rate those places. At last, this system was validated in the city of *Parma*, Italy, asking for real users to use it and showed positive results.

Classification

Table 2.18: Classification of *"UTravel: Smart Mobility with a Novel User Profiling and Recommendation Approach"*

Source	RA	I	UIA	CM	DC
[4]	H (CA+D)	3	3	2	3
RR	**CR**	**S**	**O**	**F**	**V**
2	UP;L	1	1	TS;AS	2

2.5.12 Baraglia et al., *"A Trajectory-Based Recommender System for Tourism"*, 2012 [11]

Objective - This TRS provides personalized suggestions about touristic POIs in order to assist a tourist visiting a city. The system produces recommendations depending on the current position of a tourist and formerly collected data outlining trajectories made by other tourists. The goal is offering the tourist a list of suggestions of places from which they can choose several points instead of only one place to visit.

Technical summary - This TRS consists of two modules. There is a knowledge model (a RS which works offline) that is executed when new data is available to be updated. This module uses a dataset of trajectories obtained from users *GPS* devices and a set of POIs coordinates obtained from a dataset. Once that the trajectories are sets of coordinates and timestamps, the system identifies the POIs that users visited by comparison of data and estimate the time that is needed to travel from one point to another with the help of timestamps. Then, there is a computed process (T-Pattern Tree) to find the similarity between trajectories. On the other side, from the online module (a mobile device), the current user location is obtained from GPS and sent to the offline module if it is a new position. This way, the system can build the trajectory the user is doing and compare it with the trajectories stored. For each pattern similarity found between them, a score is assigned. The highest the score, the higher the similarity with other trajectory and that way the system recommends possible next locations for the user to visit. The system was tested, and the performance results were positive for the suggestion of a list close to the user's position.

Classification

Table 2.19: Classification of *"A Trajectory-Based Recommender System for Tourism"*

Source	RA	I	UIA	CM	DC
[11]	-	3	-	3	3
RR	**CR**	**S**	**O**	**F**	**V**
2	L	1	1	AS	2

2.5.13 Mehmood et al., *"Design and Development of a Real-Time Optimal Route Recommendation System Using Big Data for Tourists in Jeju Island"*, 2019 [68]

Objective – This TRS aims to provide route recommendations based on user preference in order to maximize the quality of the touristic experiences.

Technical Summary – This system is composed of three components: a *repository layer*, a *computational layer* and an *application service layer*. The *repository layer* stores the tourists travel data: the routes they make in *Jeju Island*, collected with the help of *Wi-Fi* routers; latitude and longitude of the places that they visited and the distances between those locations, obtained from third parties APIs. The *computational layer* comprises all the management of the data on a cloud. The process starts when users access the mobile application (*application service layer*), get a list of categories (cultural heritage, ancient historical sites, art galleries, restaurants, seaside views and theme parks) and input their preferences, which are based on the cost of the location, age and gender. Then, based on the user's current position, a general recommendation is supplied, and users select the location they prefer visiting. Depending on what the user selected, the application gets the coordinates of the location. Users have then the possibility of choosing from one to three available options between time, distance and popularity of the location. If only time is chosen by the user, then the fastest route is suggested. On the other hand, if the user picks distance, then the best possible route is recommended in regard to weather and traffic condition.A third-party API is used to check the weather (*OpenWeatherMap*) and recommend an optimal route based on it. Another third-party API (*TomTom* traffic) is used to analyze traffic status and suggest alternative routes that escape rush. The popularity of each location is calculated based on the number of times it is visited in all the analyzed routes, and it is analyzed if users usually travel from one specific point to another point. The optimal route is finally recommended to the tourist. The proposed system was tested and validated, showing that the recommended optimal routes encourage tourists to visit as many popular places as possible based on user preferences, weather, and traffic.

Classification

Table 2.20: Classification of *"Design and Development of a Real-Time Optimal Route Recommendation System Using Big Data for Tourists in Jeju Island"*

Source	RA	I	UIA	CM	DC
[68]	-	3	1	2	4
RR	**CR**	**S**	**O**	**F**	**V**
1	L;D;T;B WC;UP	1	1	RR;AS	2

Table 2.21: Classification of the related work

Source	RA	I	UIA	CM	DC	RR	CR	S	O	F	V
[3]	H	-	3	-	3	1	UP; WC; B; POII	1	1	AS	1
[73]	CA	-	1	-	3	2	L; T; WC; UP	1	1	AS	1
[57]	H (CB + CA)	3	3	-	3	2	L; T; WC; B; TM; POII; UP; MH	1	1	AS; TS	2
[69]	-	3	3	3	3	2	L; TM; D; T; POII	2	3	AS; RR	2
[62]	CA	2	3	2	3	2	L; T	1	1	TS	2
[13]	H (CB+CL)	3	3	-	3	2	L; UP	1	1	AS	2
[41]	H (CF + CA)	4	3	2	3	1	L; D; POII; T; WC; MH	1	1	AS; RR; SA	2
[72]	CA	3	1	2	3	2	L; D	1	1	TS	2
[98]	CF	4	2	2	3	2	L; WC	2	2	AS; SA	2
[22]	CB	-	1	-	2	1	L; UP; POII	2	3	RR	2
[4]	H (CA + D)	3	3	2	3	2	UP; L	1	1	TS; AS	2
[11]	-	3	-	3	3	2	L	1	1	AS	2
[68]	-	3	1	2	4	1	L; D; T; B; WC; UP	1	1	RR; AS	2

2.6 Summary

With the support of the reviewed studies, it was possible to explore what are the main reasons for the creation of TRSs and to identify relevant characteristics of their implementations. The examined sample of multi-criteria tourism recommendation systems serve as a global overview, not exhaustive, of typical work in this area, since not all the existing systems were included in the analysis.

Starting with the analysis of the recommendation approach (*RA*), it is now clear that there is a tendency for hybridization of the approaches and a special focus on the context-awareness. The context-aware data is increasingly present in these systems because nowadays the number of mobile recommendation systems is growing, and they make use of different technologies present in mobile devices to collect data about the user environment. That confirms the results we get from interpreting the interfaces (*I*) used for the TRS, where it is possible to understand that the most used are mobile interfaces. Besides, it complements the conclusions taken about reactive recommendations (*RR*), because almost every system is reactive, that is, capable of generating different recommendations by reacting to user context changes.

Another topic that was studied was the TRSs connection mode (*CM*). The conclusion that it is possible to take is that all of them work or just online or in a hybrid way, where a part of the system does not require an internet connection. The general use of internet connections happens because of the different TRSs, in general, use current data from third parties APIs or have web applications or have social networks connections. Generally, before or during the trip, the TRS take some inputs from the tourist to create a user profile and calculate the recommended result, which is then sent back to the tourist. The current data (*CD*) used by these systems also matches the fact that most of them use dynamic data, which basically means they use data regularly updated.

The *UIA* results show that most of the TRS acquire information in a hybrid way and if it is not hybrid, then most of the times it is in an explicit way, that is, the data is obtained by asking to the user. Continuing, recently, some TRSs can adapt the results to the user by taking its context information and looking for the taken conclusions in Table 2.21, user location is used almost every time for the recommendations and weather conditions is the second most used context information. In this set of criteria for recommendation (*CR*), user preferences (*UP*) and time information (*T*) are usually considered. On the other hand, information about the mobility history of the users (*MH*), the available budgets (*B*) and transportation modes (*TM*) for the trips are rarely considered. All those *CR* are used to provide different functionalities (*F*). Between those functionalities, the suggestion of attractions (*AS*) leads by far. At the same time, just a few systems suggest tourists services (*TS*) or recommend routes (*RR*) even though in real travel scenarios, POI routes or itineraries are more useful for tourists compared to POI list recommendations. Simultaneously, any of those TRSs have the functionality of letting the user plan its tour (*RP*). About the validation (*V*), clearly, almost every TRS was validated to evaluate and test the proposed capabilities. Once that most of the interfaces developed for these systems are mobile applications, the most common way to validate them was by gathering up a set of users and make them use the apps in the locals for which they were developed. Also, algorithmic performance tests are common.

To finalize the topic, it is missing the overcrowding (O) and sustainability (S) analysis. In this case, it is possible to conclude that the systems almost do not cover information about supporting sustainability or overcrowding, which is used for rating the sustainability of touristic places or prevent overcrowded places. The current crowding of the POIs is only considered in [22, 69, 98].

The present dissertation aims to realize a TRS that recommends sustainable routes, combining different user preferences, sustainable POIs and avoiding overcrowding. Having been made the previous reflection, in general, the TRS that we propose in this dissertation follows the same characteristics that the majority explained before. However, it is clear that there is still space for research about sustainability and overcrowding within the area of TRS. Also, in terms of user preferences, we pretend to combine more than the usual ones.

[This page has been intentionally left blank]

Chapter 3

System Design

Contents

3.1 Sustainable Tourism Crowding project 37
3.2 STC project use case diagram 39
3.3 ROUTE microservice communication design 39

This chapter describes where the research work in this dissertation fits within the global STC project and how it was developed. Section 3.1 describes the overall project structure and its microservices architecture. Section 3.2 presents a use case that summarizes the functionalities and the relationships between the actors in this project. Section 3.3 explains the communication design between the *ROUTE* microservice and the other microservices.

[This page has been intentionally left blank]

Chapter 3
System Design

3.1 Sustainable Tourism Crowding project

As it was said in the first chapter, the STC research project intends to stimulate the visitation of more sustainable POIs and to spread tourists throughout places that are less crowded in the historical center of Lisbon, *Santa Maria Maior* parish council, which has previously been case study [78, 87].

The STC project is built around a distributed microservices architecture where the different components work separately and communicate through defined interfaces. The proposed overall microservices architecture is depicted in Figure 3.1 using Aji Modeling Language (AjiML), a graphical language to model microservice architectures [90] which focus on the services and their intercommunication structure. Each service is illustrated as a cube while interfaces are shown as adjusted circles to a service.

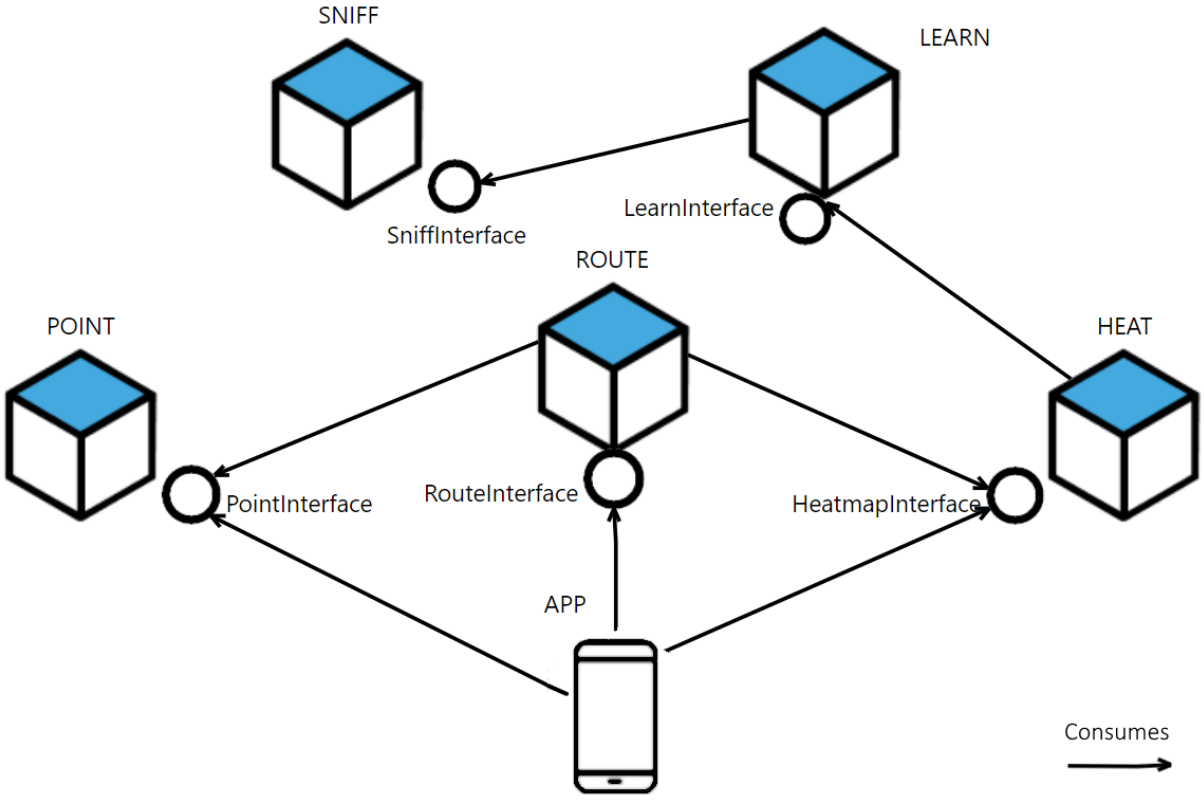

Figure 3.1: Overview diagram of the STC project microservice architecture

Microservices architectures [12] are useful for large, complex and long-lived applications [26], whenever it is required compatibility with different platforms and type of devices. This type of architecture builds upon a set of small applications capable of developing concrete tasks, that collaborate to realize a common purpose. *Netflix*, *eBay* or *Amazon* are examples of companies that invested in microservices architectures to turn out the maintenance and

scalability of their products easier and faster [40]. When opting for a microservices architecture, there is the freedom to choose the technology to be used in each of the small services that form an application, so it is possible to select the one that best suits the current needs. Another benefit of adopting a microservice platform is that each service can fail and heal independently with a likely reduced impact on the global platform's functionalities [26]. All these benefits are significant for our project, because we have a mobile app (*APP*) that depends on the other services, and those services must be deployed in isolation, so that if one of them fails, it does not affect the others in the short term (a few hours), ending up ruining the tourist experience. It is also an essential type of architecture, because each of the services was deployed using varied technologies.

An overview of each service and corresponding interactions, as represented in Figure 3.1, will now be provided. A more detailed explanation of the relevant parts for this dissertation will appear in Section 3.3.

- **SNIFF** – this edge computing device (smart sensor) detects the number of mobile devices active in its proximity, by tracking the usage of Bluetooth, Wi-Fi, and the Mobile Network, what is considered to be a good surrogate for the local crowding level;
- **LEARN** – this microservice stores all the data from the *SNIFF* sensors and uses it to produce future forecast patterns of crowding of the different areas of the covered territory to be passed on to the *HEAT* microservice;
- **HEAT** – this microservice is responsible for composing crowding contour lines, based upon the data from past and forecast occupancy received from *LEARN*, as well as the current data from continuous location pings from mobile devices (*APP* instances);
- **POINT** – this microservice consists of a web-platform interface that allows for user input of geo-located POIs and that stores this data in a local database;
- **ROUTE** – is a core microservice that concurrently serves multiple requests from APP instances (users' smartphone apps) in order to suggest route recommendations that avoid crowded areas and simultaneously meet sustainable POIs. For creating these routes, it consumes the data from the *HEAT* microservice that gives the actual and forecast crowded data and the POIs from the *POINT* microservice. Once the calculation of the ideal route is done, it is supplied to the *APP* interface;
- **APP** – is the interface between the system and the tourist. It is a mobile application that listens to tourists' preferences and constraints (more details in Section 3.3.3) and provides them to the *ROUTE* system which generates the route recommendation that is shown to the user in the interface. Also functions as a monitoring tool that feeds useful information (e.g. user's location) to the system.

We can say that all these six microservices have different individual goals, elaborating their tasks. Still, they are all connected to materialize a common purpose: to recommend the most personalized route attending to the existing restrictions. In this architecture, the solution developed in this dissertation is represented by the microservice *ROUTE*, so this dissertation's main work concerns its development.

3.2 STC project use case diagram

In Figure 3.2 there is a Unified Model Language (UML) use case diagram that summarizes the relationships between the actors and systems requirements of the communication between the *ROUTE*, the *APP*, the *POINT* and the *HEAT* microservices. The diagram is essentially built on a perspective of the functionalities that are important for the *ROUTE* microservice. In blue colour, we have the main functionalities provided by the *APP*, where the users also participate, which are: *Set constraints and preferences*, *Manage route recommendation* and *Request route recommendation*. In orange colour, we have the main functionality of the *POINT*: *Manage POIs*. In red colour, we can observe the functionality offered by the *HEAT* microservice: *Manage crowding data*. Finally, in green colour, we can see the functionalities of the *ROUTE* microservice: *Provide route recommendation*, *Update crowding data* and *Update POI database*. All the use cases are explained ahead during the dissertation.

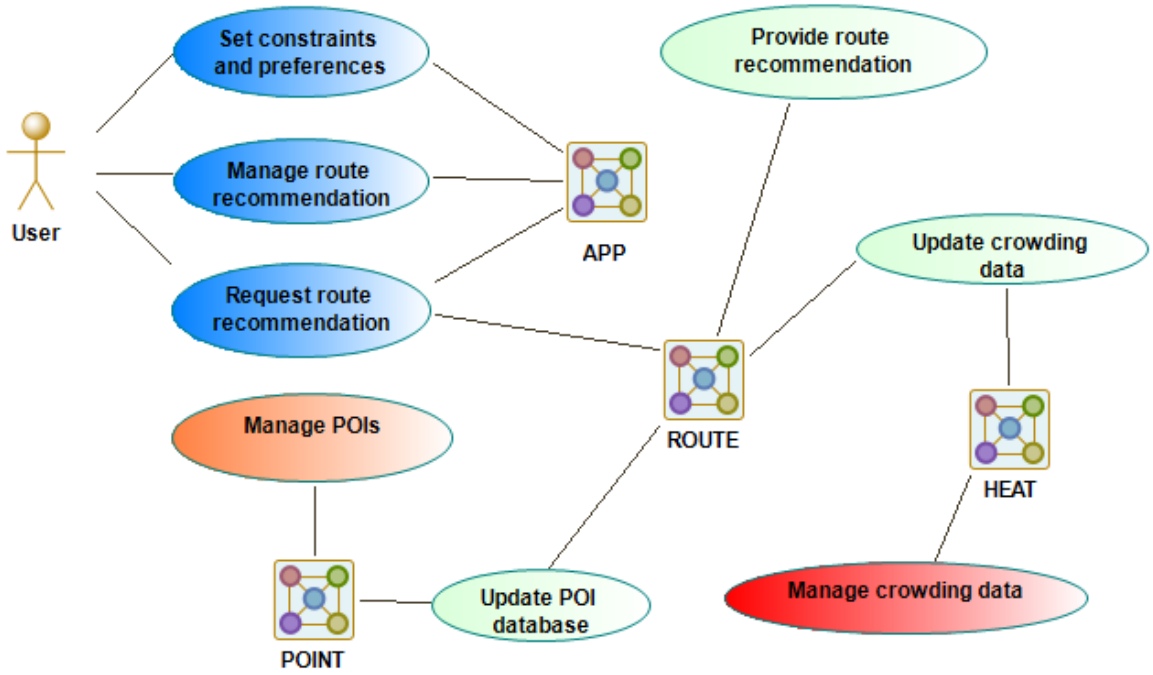

Figure 3.2: Use case diagram of STC project

3.3 ROUTE microservice communication design

We used the *Stoplight*[1] application for API creation, design, modelling, and technical writing. An API is an interface which defines a set of definitions and interactions between multiple services without having to know how they are implemented (e.g. determines the requests that may be made, how to make them, the conventions to follow, the data formatting that must be used, etc.). By using *Stoplight*, we were able to define the intended use of our APIs, that is, describe the services and data we would interface and establish how services would interact with each other. All this process also helped in having proper documentation, which makes the

[1] https://stoplight.io/

development easier, taking less of effort than a fewer structured approach. All the interactions between the *ROUTE* and the *POINT* (Subsection 3.3.1), the *HEAT* (Subsection 3.3.2) and the *APP* (Subsection 3.3.3) were developed and represented ahead.

3.3.1 ROUTE – POINT communication

The *ROUTE – POINT* connection is crucial so that the *ROUTE* obtains all the information about the POIs that it needs to suggest personalized routes. On the *POINT* side, a web platform interface was already implemented, in HyperText Markup Language (HTML) and *Javascript*[2], with the purpose of creating a user-friendly interface so that the local government body (in this case, *Santa Maria Maior* parish council workers) could manage the information about the POIs in the database that saves it. A Representational State Transfer (REST) API, which is a web architecture to manage information over the internet [64], was designed to connect the web platform to the linked database. The platform sends Hypertext Transfer Protocol (HTTP) requests (GET, POST and DELETE), the API execute Structured Query Language (SQL) queries in the database, and the result return information is obtained in the JSON[3] format. On the other side, the *ROUTE* microservice periodically replicates the POIs database. A representation of these communications between the microservices and the database can be observed in Figure 3.3.

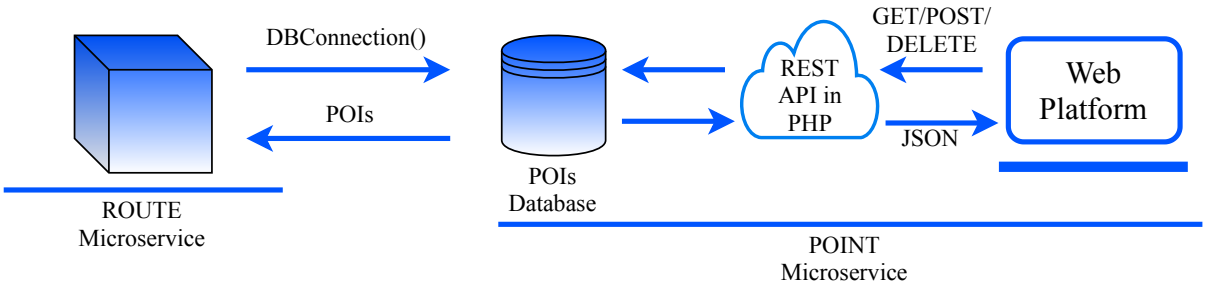

Figure 3.3: *ROUTE-POINT* communication

Next, in Figure 3.4, it is possible to observe an activity diagram correspondent to the use case *Update POI database* in Figure 3.2. It is suitable to describe how the activities needed to accomplish the replication of the POIs data on the *ROUTE* microservice side are coordinated. Considering that all data has been filled in by the responsible persons and is updated, *ROUTE* replicates the POIs from the *POINT* microservice database every week. Then, the first step is to start a connection with the POIs database. If the connection is unsuccessful, then the service must receive a warning that something went wrong. Otherwise, the process can continue with the processing of an SQL command with the request for the strictly necessary information. The retrieved data is replicated in Java, and then the connection to the database is closed. Finally, the microservice should keep waiting for another signal of an update to repeat the process.

[2]https://www.javascript.com/
[3]https://json.org/

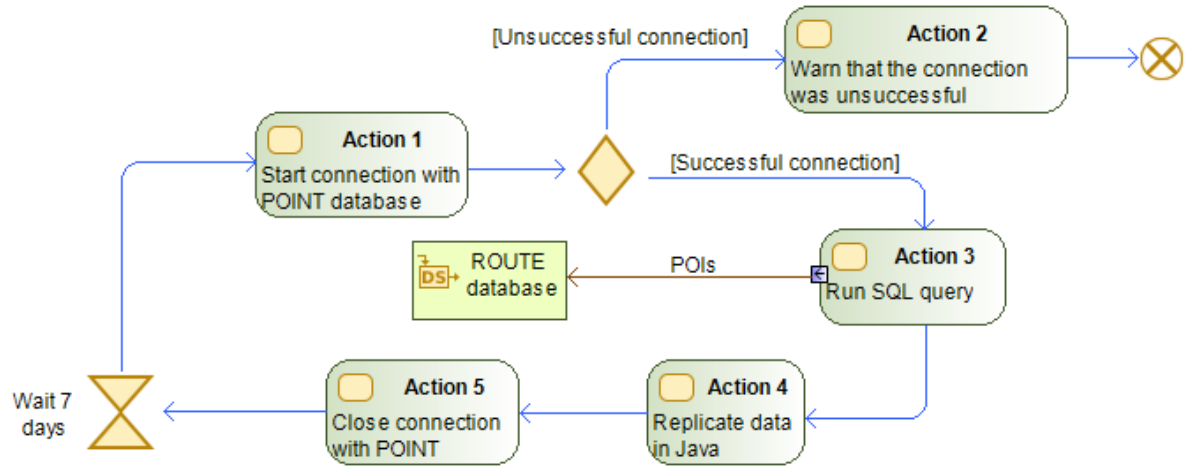

Figure 3.4: *ROUTE* workflow for the replication of POIs database

3.3.2 ROUTE - HEAT communication

We can observe in Figure 3.5 the communication between the *ROUTE* and the *HEAT* microservice. As explained earlier, the *HEAT* microservice is recurrently receiving past and current crowding data from the *LEARN* and *APP*, respectively. Then, *HEAT* stores that data on a database and uses it to calculate and predict the crowding level of the different given areas (more details in Section 4.2) which are also stored.

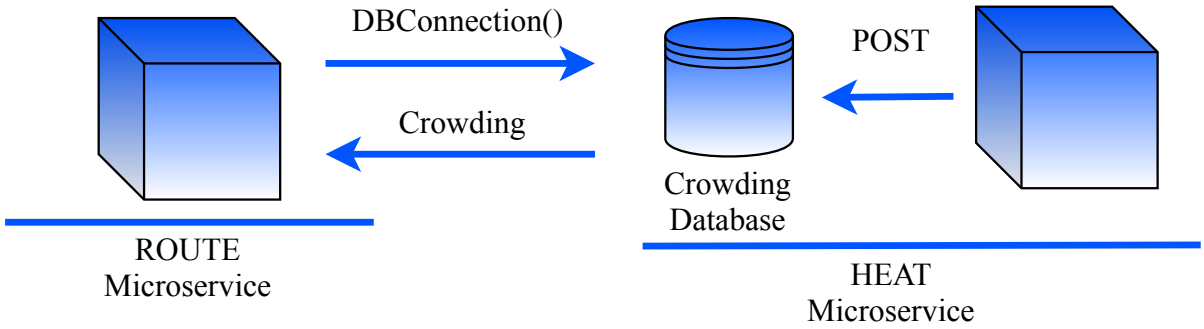

Figure 3.5: *ROUTE-POINT* communication

The *HEAT* microservice regularly processes those data inputs to improve the predictions of crowding for the next hours. This way, predictions change, and *ROUTE* must also be as updated as possible, because the closer to the current date the predictions are asked for, the more accurate the recommendations will be. So, every hour the *ROUTE* microservice replicates the crowding data from the *HEAT* database. The data chosen are all those whose hour is equal to or greater 12 hours than that moment. This process is visible in the activity diagram represented in Figure 3.6.

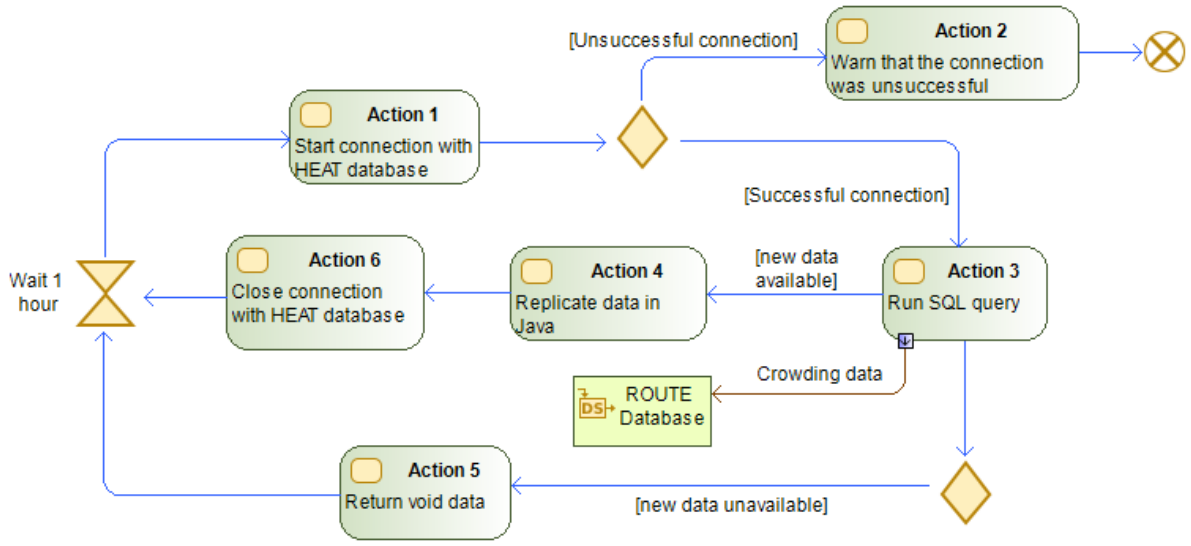

Figure 3.6: *ROUTE* workflow for the replication of crowding database

3.3.3 ROUTE - APP communication

For the purpose of getting information from users and retrieve route suggestions, it was also designed the communication between the *ROUTE* microservice and the *APP*, as we can see in Figure 3.7. The connections are made between the *APP* and the *ROUTE* using an API. Once there is no standard API message response format for use in RESTful applications, developers can adjust response data to the exact specifications and shape they need when building the API. In our case, we chose the JSON format.

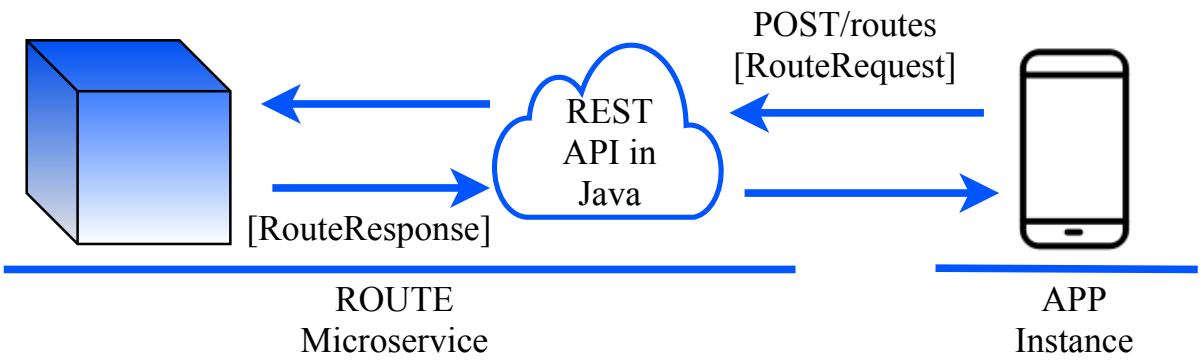

Figure 3.7: *ROUTE-APP* communication

The communication process starts when the *APP* instances send HTTP requests (POST/*RouteRequest*). This corresponds to the mobile app sending the users' preferences and constraints to the *ROUTE* system. The message body consists of a *RouteRequest* object which can be seen in the condensed examples in JSON in annex H (e.g. Listing H.1). More details about the route requests can be seen in 5.3.

Then, the *ROUTE* system executes *Java* methods with the result being returned to the mobile app, in the JSON format, through other HTTP response (POST/*RouteResponse*). The

response's body is a *RouteResponse* object, which can be seen in the condensed examples in JSON in annex H (e.g. Listing H.2). More details about route responses can be seen in 5.7.2.

All the fields exemplified in the listings are explained in more detail in chapters 4 and 5.

[This page has been intentionally left blank]

Chapter 4

Theory and Decisions

Contents

4.1 Sustainability classification 47
4.2 Crowding classification 49
4.3 Weather conditions 50
4.4 Physical effort 53
4.5 Routing algorithm approach 54
4.6 Graphs and paths 55
4.7 Route directions API 57

This section intends to clarify the theory integrated into the formulation of the tourist preferences, the tour constraints, and the implementation explained in chapter 5. Subsection 4.1 describes the principles of sustainability, how to measure it and with what indicators since it is necessary to classify the level of sustainability of each recommended POI. Subsection 4.2 intends to clarify the definition of crowding and how this work classifies the crowding of different areas of the map. Subsection 4.3 explains what the influence of the weather conditions on the tourist experience is and how to obtain the weather condition data to make decisions. Subsection 4.4 describes how to measure the physical effort of walking during a tourist trip. Subsection 4.5 describes the tourist trip design problem, its requirements and the intention of solving it. Section 4.6 explains the relation between graphs and route recommendation. Section 4.7 explains the choice of the API used in this project to generate routes.

[This page has been intentionally left blank]

Chapter 4
Theory and Decisions

4.1 Sustainability classification

The diversity and breadth in the definition of sustainability and related terms have been growing as the concern on this topic increases around the world. A commented glossary of sustainability terms can be found in [45], where the author concludes that it encompasses principles aligned within three fundamental dimensions: environmental, economic and social. Sustainable development must encompass concerns across all those dimensions.

As mentioned earlier, this project focuses on tourism and aims to promote the local, social and environmental sustainability dimensions in Lisbon neighborhoods most affected by tourism overcrowding. Our aim is to support a policy at the heart (intersection) of the aforementioned sustainability dimensions (see Figure 4.1). It is claimed in [93] that for a territory to be considered sustainable it must combine interactions between the dimensions to reach a development classified as: *equitable* (synergy between the economic and social dimensions for impartial treatment of all citizens), *livable* (interaction between environment and social needs for an agreement between the environment and social needs concerning a better quality of life) and *viable* (economic and environmental cooperation, e.g. avoid the use of non-renewable resources).

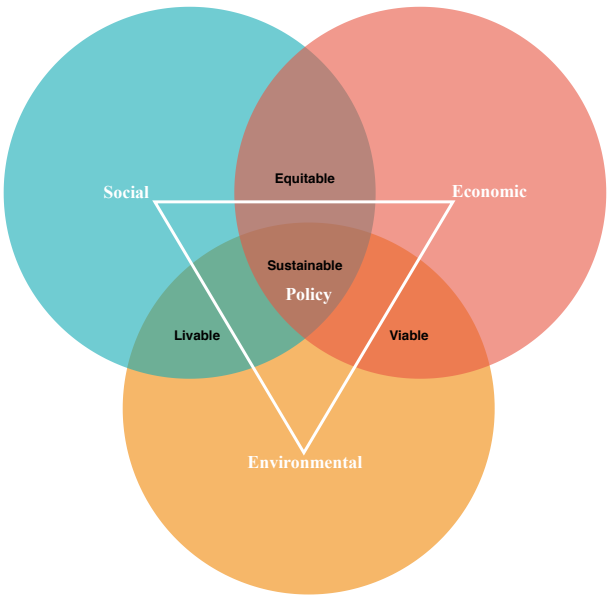

Figure 4.1: Sustainability dimensions (adapted from [45] [93])

A sustainability policy is a *"set of ideas or a plan of what to do in particular situations that have been agreed officially by a group of people, a business organization, a government or a political party, about environmental, economic and social issues"* [45]. In other words, the term policy is strictly related to environmental legislation that can be locally, nationally or internationally determined and can address issues, such as sustainable development, climate change, etc. Planning policies

for the sustainable use of resources drives to what is called sustainable development, that is described in the literature as *"development that meets the needs of the present without compromising the ability of future generations to meet their own needs"* [19].

The UN Sustainable Development Goals[1] address the several global challenges we face (see Figure 4.2).

Figure 4.2: UN sustainable development goals

The current work is directly aligned with two of those UN goals:

- **goal 8 (Decent work and economic growth)** - since among the sustainable POIs that are recommended to tourists there are different local stores or restaurants that usually are less visited, which promotes local employment and local economic growth;

- **goal 11 (Sustainable cities and communities)** – because the suggestion of more sustainable POIs while avoiding crowded places, disperses tourists, reduces the crowds, prevents social problems (e.g. citizens' frustration with tourist agglomerates), and helps to increase tourists' quality experience.

One major concern about sustainability is how to measure it. Sustainability indicators for tourism development are useful instruments for managing and controlling tourism activity in a region. Their selection and periodic monitoring are fundamental activities while planning and guaranteeing the deployment of a sustainable tourism policy [49]. Numerous institutions have also formulated a list of indicators, called Indicator Systems, to be applied in several countries and to help measure and manage sustainable development. An example of a system like that is the European Tourism Indicators System (ETIS) [74]. ETIS is a system of indicators suitable for all tourist destinations, intending to help destinations to monitor and measure their sustainable tourism performance, by using a common comparable approach. In [67], the author reports the application of ETIS in County Donegal, Ireland, in order to deal with the challenge of tourism growth in the region.

In our specific case, the task of classification the POIs sustainability is assigned to one of the project partners, namely *GEOTA*[2] and the sustainability specialists of *Santa Maria Maior* parish council. To do such work, they should use both the ETIS website[3] and toolkit [74] that contain a few tools to collect and compare information on tourist destinations. The research, collection

[1] https://www.un.org/sustainabledevelopment/sustainable-development-goals/
[2] https://www.geota.pt/
[3] https://ec.europa.eu/growth/sectors/tourism/offer/sustainable/indicators_en

of data and calculation of the results of the indicators is all their responsibility. They should collect the necessary data on the indicators contained within the ETIS (Annex I) that they find suitable to measure the sustainability dimensions of this work. After all the indicators have their results calculated, each POI must be classified according to a scale that varies between 0% and 100%, where 0% indicates that the POI is not sustainable at all and 100% is completely sustainable. To classify the POIs, the sustainability specialists must access the web interface of the *POINT* microservice, authenticate and fill that information like it is shown in Annex II. Table 4.1 shows a short example of how sustainability looks like in the POIs database.

Table 4.1: Sustainability data on the database

Point_ID	Latitude	Longitude	Sustainability
1	38.7139092	-9.1334762	50
2	38.7119051	-9.1406362	25
3	38.7121290	-9.1394235	75

4.2 Crowding classification

Part of this project is motivated by the observation of the severity of the crowding effects understood by the populations of the historic neighbourhoods in Lisbon. Through people who live and work in these communities, by associations of residents and heritage and by parish councils, it was revealed that, although the communities in historic neighbourhoods are used to the tourist presence, the discomfort associated with crowding nowadays emerges mainly from the accumulation of the presence of passing tourists with the many who remain in local accommodations.

Crowding is described as a negative evaluation of a certain density [83]. While density can be equitably determined by counting the number of people and measuring the space that they occupy in a certain area, perceived crowding is a psychological dimension that exists in the minds of individuals, and that involves a value judgment requiring information about the environment, the desired activity, and the person making the evaluation [86]. In order to distinguish both terms, we can give a simple example. Let us assume that there are 200 persons in an event one day and 500 the next. Certainly, density is higher on the second day, but maybe it is not more crowded. Suppose the event is being hosted in a big square, neither 200 nor 500 people are enough to consider that the event is crowded on both days. On the other hand, if it is an event on a small street, it might be crowded both times. The definition of crowding raises some problems about how to measure it, but the undertaking project intends to make the detection, monitoring, management and classification of crowding in real-time in the historic districts of Lisbon, namely in the parish council of *Santa Maria Maior*. Crowding is a frequently studied phenomenon in the outdoor recreation, and diverse sources of crowding data have been studied, such as online social networks [32], mobile operators [75], image and sound capturing [6, 36, 61]. Thus, as a way of monitoring the number of individuals in a given area and collecting crowding data, as has already been explained through the microservices that make up the project, two crowding detection and monitoring solutions are planned: *SNIFF* and *APP* (see Section 3.1).

In order to classify the levels of crowding of the POIs, microservice *HEAT* is going to periodically update the crowding data in its database (more details in Section 3.3.2) using a scale developed by Heberlein and Vaske [51], as we can observe in Figure 4.3. In this item, two of the nine scale points label the situation as uncrowded. The remaining seven points mark it as crowded to some degree.

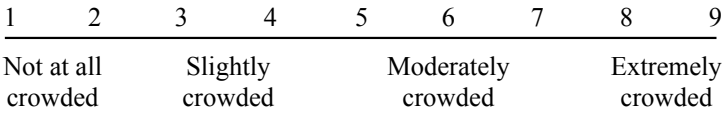

Figure 4.3: Crowding scale (source: [86])

Originally, the purpose of the scale was to ask people to indicate how crowded the area was at the time of their visit. However, for the sake of better user experience when using the mobile app, it will not be asked to users how crowded each place they visited was, so it will be calculated in a closer way to what is the definition of population density. As it was already explained, with the help of the *SNIFF* and *APP* it is possible to detect and measure the number of devices (which can be seen as the number of people) in the vicinity of those gadgets by using different technologies like *Wi-Fi* and *Bluetooth*. With all that current information and the historical data that the *HEAT* microservice can access, it will periodically calculate what is the percentage of people that is present in each place compared to what is the normal and maximum acceptable number of persons in that place. After that, the data is inserted in the crowding database of the *HEAT* microservice and the *ROUTE* microservice can replicate that data. This way, *ROUTE* can calculate routes based on the data provided. In Appendix C (Listing C.1) is exemplified the type of data inserted into the database relative to the crowding level calculation. It is used a *MongoDB*[4] database, which is a document database and that is why it allows storing data JSON-like documents as it is exemplified in the listing.

As we can observe in the abbreviated example, there is a timestamp associated, "*timestamp*", for the date and hour that the crowding level, "*crowding*", concerns. Then there are the list of coordinates, *"coordinates"*, that the *HEAT* found to have or predict to have the specific level of crowding at the specified date and hour. The example is abbreviated because it would continue with more objects for different dates, different crowding levels and coordinates. This way, the *ROUTE* microservice can search the levels of crowding of the coordinates that the route may pass through at a specific day and hour.

4.3 Weather conditions

Searching in the Cambridge Dictionary[5] , we can find the definitions of climate and weather. For the first one, it says that is "*the general weather conditions usually found in a particular place*", and for the second we know that are "*the conditions in the air above the earth such as wind, rain, or temperature, especially at a particular time over a particular area*". This means that beyond that

[4]https://www.mongodb.com/
[5]https://dictionary.cambridge.org/dictionary/english/

there's no doubt that climate invokes the concept of weather, tourists will experience and be influenced by the current weather in their destination and not exactly by the climate conditions.

The impact of weather conditions in route planning is an area that was identified for research some years ago [42]. However, what criteria represent the most suitable or ideal conditions that establish importance for personal experiences of tourists? To answer that question, this chapter required some review of literature related to the impact of weather/climate in tourists' satisfaction and their participation in activities. Back in 1973, [5] demonstrated that between a lot of different factors that would affect tourism demand, the climate was one of the principal. More recently, [30] said that the relation between weather conditions and the attractiveness of outdoor activities is a function of thermal, physical and aesthetic aspects that have their own significance and impact, as it is possible to see in Table 4.2. The aesthetic facet is about a psychological aspect of the climatic conditions that the tourist might like or not, e.g. having a sky full of clouds (high cloudiness); the physical facet is about the conditions that disturb the satisfaction of a tourist (except in a thermal sense) when participating in some activity, e.g. the rain duration preventing some activity from occurring for a long time or high wind blowing sand against people on the beach; the thermal facet is about the level of comfort that a tourist feels with the temperature, wind, etc.

Table 4.2: Facets of tourism climate, their significance and impact (source: [30])

Facet of climate	**Significance**	**Impact**
Aesthetic		
Sunshine/cloudiness	Quality of experience	Enjoyment, attractiveness of site
Visibility	Quality of experience	Enjoyment, attractiveness of site
Day length	Convenience	Hours of daylight available
Physical		
Wind	Annoyance	Blown belongings, sand, dust...
Rain	Annoyance, charm	Wetting, reduced visibility and enjoyment
Snow	Winter sports/activities	Participation in sports/activities
Ice	Danger	Personal injury, damage to property
Severe Weather	Annoyance, danger	All of the above
Air quality	Annoyance, danger	Health, physical wellbeing, allergies
Ultraviolet radiation	Danger, attraction	Health, suntan, sunburn
Thermal		
Integrated effects of air temperature, wind, solar radiation, humidity, longwave radiation, metabolic rate	Thermal comfort, therapeutic, restorative	Environmental stress, Physiological strain, Hypothermia Hyperthermia, Potential for recuperation

From what we can understand, weather acts either as an impediment or a stimulus to the participation in different outdoor activities while impacts the degree of satisfaction and enjoyability of those activities [54]. This relationship has been studied and published in different studies, showing that temperature, wind, snow, radiation, humidity, precipitation, etc. are key factors in specific recreational activities, but also in general tourism [46, 88, 95]. Understanding that climatic information is essential for the tourists' travel experience and to make decisions, there is the need to explore different sources where its information is available: television, internet, radio, travel agents, tourism organizations, newspapers, and hand-held devices. However, Scott and Lemieux [84] report that there are different types of information required at different stages of the trip planning process, but the most common tourism marketing information and

web sites provide limited climate information to potential travellers (usually provide only average monthly temperatures). Looking for Figure 4.4, it is possible to understand that climatic information (i.e. expected conditions) is advantageous in advance of the trip while weather forecasts are more important than climatic averages at the destinations.

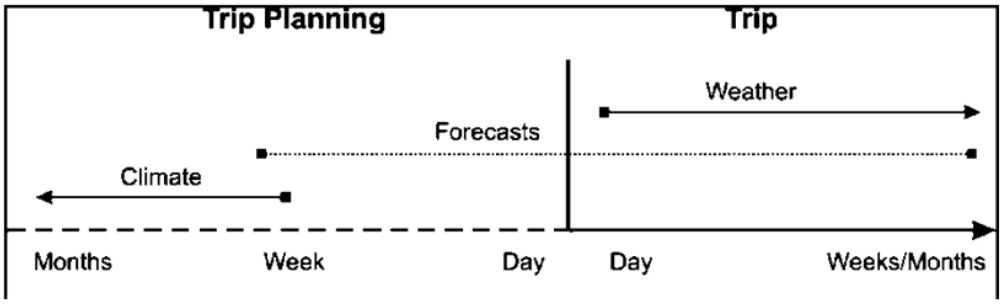

Figure 4.4: Weather-climate information for tourist decision-making (source: [84])

4.3.1 OpenWeatherMap API

Since our tour recommendations are prepared to be calculated in real-time, we chose to have a weather API as data source provider, but before selecting which one was better, it was important to know the service features they would offer that suits better our needs. A first point to consider was that the API would be free or not because one of the requirements of this project is to use free technology. Based on this, it was possible to filter some weather providers because some features may not be free. Furthermore, there are weather providers that limit the number of calls to their services if you select a free plan, and it was of our interest to be the less limited possible. Another important aspect was the possibility to gather weather conditions for a selected location based on the specific geographic coordinates (latitude and longitude) instead of just trust on standard meteorological data that could just be representative of a general location or not representative of some areas at all [14]. Finally, having comprehensive API documentation that includes many examples of requests and the responses returned, would facilitate integration with the rest of the project.

Thus, in this project, we chose to use the *"5-days/3hour weather forecast" OpenWeatherMap* API[6] (Figure 4.5).

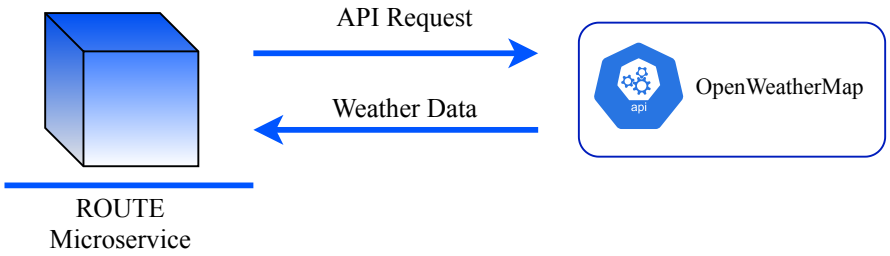

Figure 4.5: *ROUTE-OpenWeatherMap* communication

[6] https://openweathermap.org/

This API offers a forecast of every 3 hours after the moment of the request for the next five days. With this information, we can recommend tours that take into account the development of weather conditions not just in the moment but also in the future. This forecast is accessible for any location or city by providing geographic coordinates and is available in JSON format. Next, I present an example of an API call (free users can make up to 60 API calls per minute):

Table 4.3: *OpenWeatherMap API* call example

OpenWeatherMap API call example		
*api.openweathermap.org/data/2.5/forecast?lat=**{lat}**&lon=**{lon}**&appid=**{your api key}***		
lat - latitude	lon - longitude	appid - API key to connect the project to the API and authenticate the request

This call returns a response that is exemplified in annex A in Listing A.1. As we can see in the response, the *OpenWeatherMap API* currently provides a wide variety of weather data, including the current temperature, minimum temperature, maximum temperature, air pressure, sea level, humidity, level of cloudiness, wind speed and direction, raining probability and all this information for every three hours within the next five days. However, even though we have all this information, not all of them is relevant for us for two main reasons. Firstly, because from the multitude of meteorological and climatological parameters measured, the most relevant for tourism are air temperature, air humidity, wind speed, wind direction, cloud cover, sunshine duration or radiation fluxes, rain and precipitation, snow cover and water temperature. However, in most cases, knowing the air temperature and precipitation provide very helpful information [65]. Secondly, because tourists in Lisbon revealed the intention of engaging in a diverse bundle of activities [60] which, according to [85], turns it much easier for tourists to avoid unsuitable weather. For those reasons, we chose to obtain from the JSON data just the rain probability. This will be useful for our recommendations because in our POI database, each attraction is related to a category, and each category will be suitable to visit depending on being indoor or outdoor. What we do is recommend just the indoor attractions (local stores, religious spots, restaurants, *tasquinhas* and museums) if the rain probability is equal or higher than 50%. This substitution from outdoor activities for less weather-dependent activities (e.g. indoor activities) when facing unpleasant meteorological conditions is stated by [66]. The implementation of this decision is explained in Section 5.6.2.

4.4 Physical effort

A criterion that is used to customize a recommended tour is the physical effort made by each user which allows to adequate the route to the level of effort they prefer and feel comfortable. The effort can be measured in Metabolic Equivalent of Task (MET), which is a way to calculate the rate of energy expenditure of bodies. It shows the ratio difference between the working metabolic rate and resting metabolic rate. A unit of MET ($1.0MET = (4.184kJ) \times kg^{-1}h^{-1}$) is considered a resting metabolic rate obtained during quiet sitting [2] and can be simply defined as the amount of oxygen consumed at rest (approximately $3,5ml$ of $O_2/kg/min$). So, if an activity requires 5 METs, it means a person is exerting five times the energy of resting. From sleeping

(0.9 METs) to running at 10.9*mph* (18 METs), our bodies are always burning energy, no matter the activity we are practicing [50].

Based on the model proposed by Pate [76], there are three types of intensity for classifying the MET intensity of Physical Activities (PA), as it is possible to see in Table 4.4. *Light* intensity is for values of MET smaller than three, *moderate* is for levels between three and six, and *vigorous* values of METs is for higher than six.

Table 4.4: Physical effort intensity classification into *MET*s

Level	Intensity	METs
1	Light	<3
2	Moderate	[3 ... 6]
3	Vigorous	>6

To put different activities in perspective, we chose some examples to compare (Table 4.5) from *"The Compendium of Physical Activities"* [2], a widespread study accepted among PA specialists, that identify 605 specific examples of PA measured in MET intensities.

Table 4.5: Examples of common physical activities for healthy adults by intensity of effort required in *MET* scores (adapted from: [2])

Physical Activity	**MET**	**Intensity**
Sitting quietly	1.0	Light Intensity
Listening to music	1.0	
Reading	1.0	
Sitting at a sporting event (spectator)	1.5	
Walking, less than 2.0 mph (strolling)	2.0	
Walking for pleasure	2.5	
Walking the dog	3.0	Moderate Intensity
Water aerobics	4.0	
Tennis (doubles play)	5.0	
Walking (5.0 mph)	8.0	Vigorous Intensity
Rollerblading	12.5	

For this project, it was important to study the MET of walking because it is the activity that will be practised by our users. Walking is the only mode of transportation considered in *ROUTE* microservice, and so the outcome of *ROUTE* is an optimal walking path for each route request. In order to calculate the effort of walking down the streets, we used the formula of effort that is used in treadmills with the calculation and corresponding implementation being explained ahead in Section 5.6.4.2.

4.5 Routing algorithm approach

Most tourists visit a region or city with the main purpose of visiting multiple tourist attractions [35]. However, because they usually do it for a limited period (one or more days), it becomes clearly impossible to visit all the POIs in that region. So, they always need to restrict themselves and select those POIs they consider to be the most interesting or valuable for them [35, 91].

They start selecting the POIs based on different sources (websites, guidebooks, etc.) and once the selection is made, they decide on the times to visit each attraction and trace a route between them. What happens is that tourists face several difficulties planning their tour because it involves a high number of constraints that may be considered such as the user's available time, POI's visiting days/hours, the visiting time required for each POI, the travelling distance among POIs, the user's available budget, etc. [48, 100].

In the extant literature, this mentioned challenging problem is designated as the Tourist Trip Design Problem (TTDP) [100]. The TTDP is actually one of the applications of the Orienteering Problem (OP) which in turn is also a variation of the Traveling Salesman Problem (TSP). The TSP, [55], is predetermined to minimize the distance between a set of POIs. In the OP, the goal is to maximize a trip score by determining the shortest path between a set of POIs to be visited in a specific order, limited in time [48]. This score relates to the domain of the problem and its constraints and typically comes from a value connected to the edges that join the POIs. Nevertheless, certain domains have to take into account values directly associated with the POIs. A Mixed Orienteering Problem (MOP) [101] is referred to when that condition arises.

A few different variants of the TTDP are studied and defined in the literature by considering different constraints and parameters of the generic problem [42]. Because time is an indicator of the quality of experience that is of major importance from the viewpoint of the users [48, 100], there are different approaches of the TTDP that incorporate time into the tour, namely Time Window (TW), Time-Dependent (TD) and Multi-Period (MP). The first one, TW, aims to suggest only POIs that are open at the time when the user arrives there (consider for each POI an opening and closing hour). The TD focuses on offering tours that do not exceed the user's available time for the tour (similar to the budget limit) by taking into account the calculation of the time needed to travel between POIs and the estimated time spent on each POI by the tourist. A tour that can be taken for more than one day is suggested by the MP.

The routing algorithm presented in this paper is based on the variations of the TTDP. Tour recommendations shall obey the requirements of the tourists and schedule a visit to multiple POIs over space and time with the knowledge provided by them. The issue is to choose the best POIs to suggest in order to obtain an enjoyable tour and to decide in which order they should be visited.

Therefore, analyze the definitions and variants of problems related with the set of constraints explained before is fundamental, because in this project it should also be selected an approach which solution is the closer as possible to the real problem. In this project, we consider the TW and TD concepts along with the MOP, which gives us an approach called Mixed Time-Dependent Orienteering Problem with Time-Windows (MTDOPTW).

4.6 Graphs and paths

Leonhard Euler was the first to discuss and prove that mathematical problems and other issues can be explained and solved by utilizing graph theory [34]. Graph theory is a branch of discrete mathematics which has had as a main area of expertise the study of networks as the following explanations based on [10, 15, 28] demonstrate.

A network often refers to real systems while graphs deliberate the mathematical representation of those networks. A network is a set of components often called nodes that are connected between them through links. This way, the number of components is the same as the number of nodes, where N_i represent each node, $i = 1, 2, \ldots, N$ and N is the size of the network. At the same time, the number of interactions between the nodes is equal to the number of links, designated by L. Even though the systems being studied differ in scope or aspect, a network representation proposes a common language to study them. This is where the connection between the networks and graphs happens because the same graph can be a simple structural model representation of different networks with divergent domains and concepts as it is pictured by Figure 4.6 (c).

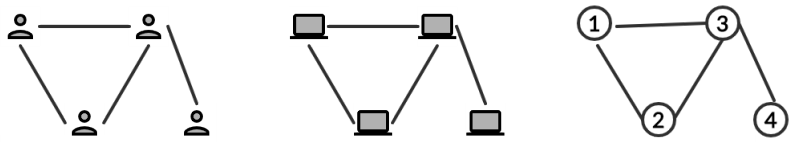

Figure 4.6: Graph vs Network

In Figure 4.6 (a) we have an example of a network where people are connected which may represent either actors that act together in the same series or an example of a social network where two people are connected if they *"follow"* each other. In Figure 4.6 (b) we have a representation of a subspace of the internet where a group of routers (computers) are linked to each other. Then, looking for Figure 4.6 (c) it is possible to see a graph that has the same representation for the two different networks consisting of $N = 4$ nodes and $L = 4$ links, although the nature of the nodes and the links contrasts. Usually, networks are also weighted, which means that their nodes and links have values associated, and consequently, its graph representation has costs associated with the vertexes and edges. The weights constitute conditions, constraints or data stored in the graph structure. For example, the weight of a link between two person in Figure 4.6 (a) can be the number of movies in common made by two actors or the number of friends in common that two social users have.

Table 4.6: Terms used for networks and graphs (source: [10])

Network Science	**Graph Theory**
Network	Graph
Node	Vertex
Link	Edge

Graphs are mathematical structures used to model pair relations between objects which are composed of vertices (also called nodes or points) that are connected by edges (also called links or lines). In the purpose of this dissertation, it is intended to use graphs and with this structure represent the paths for tourists. So, a graph will be expressed by $G = (V, E)$, where $V = \{1, \ldots, n\}$ is the set of vertices and $E = \{1, \ldots, m\}$ the set of edges. Consequently, a path, p, in a graph is an ordered list of edges within two specific nodes, n_0 and n_l, which length, l, coincide with the

number of connections since is the number of edges in the path. The path p is represented as $p = \{(n_0, n_1), (n_1, n_2), \ldots, (n_{l-1}, n_l)\}$.

In this dissertation we deal with a route recommender where the calculated paths aim to correspond to the goal of this dissertation which is to find the best path that treats all the restrictions: minimizes the distance between POIs, minimizes the crowding values, maximizes the sustainability values of the POIs, respects the user preferences and constraints (physical effort, trip budget, user available time, POIs open and close hours and weather conditions). Once we are facing a complex problem that is finding the best path in various aspects (more than two objectives) we are undeniably in front of a Multi-Objective (Multi-Criteria) Shortest Path (MSP) problem, that is addressing the Multi-Objective Path Problem (MPP). A state of the art of MPP is presented in [94]. Let S be the set of all viable solutions and f^k be a function that designates a real value to each solution in S, $k = 1, 2, \ldots, r$. The MPP proposes to find a solution that is optimal for the set of its objective functions $f^1, \ldots, f^r$, where $r \geq 1$ is the absolute number of objectives. This way, for $x \in S$, the MPP can be formulated as: $minf(x) = (f^1(x), f^2(x), \ldots, f^r(x))$. Nevertheless, there is usually not a singular solution for the problem, i.e. a path, that is optimal for all the objective functions. In these cases, the concept of optimality is replenished with the concept of efficiency. A feasible solution $x \in S$ is efficient if there does not exist any $y \in S$ with $f(y) \leq f(x)$ and $f(x) \neq f(y)$ [27].

Therefore, for the search of efficient solutions to be suggested by our route generator algorithm in this project we will be using MPP algorithms, which formulation and implementation can be analyzed in chapter 5.

4.7 Route directions API

A directions API is a service that calculates directions between locations. To be able to generate routes for suggesting, there was a need to choose a route directions API adequate to our project. Among the various options that exist, we made a comparison that was based on a set of crucial established criteria and features. First of all, it should be a free API in order to keep the requirement of developing a solution without costs. Particularly, it is preferable that the API is completely free and not limited free (e.g. free for a limited period or a limited number of calls per day) so that it is a scalable solution. Then, it is essential that it is open source. The code should be available on a source code hosting platform (e.g. *GitHub*) and written in *Java* to enable customization of the API to our needs. Another important aspect is the capability of providing pedestrian paths. Supporting directions for several modes of transportation, including transit, driving, or cycling is worthwhile, but walking is the most important because our suggestion is for visiting POIs on foot. The integration with OpenStreetMap (OSM)[7] is also important because it was the map chosen for representation on the mobile app. Two more aspects took into consideration were the graphic areas served (coverage) by the services, which must include the area of Lisbon, and considering changes in elevation along the route (*elevation profile*) so that it is possible to calculate the effort (more details in Section 5.6.4.2).

[7] https://www.openstreetmap.org/

Looking for what was explained before, research for APIs that fulfil the requirements was done, and consequently, the decision on which solution to use was made. The option fell on the open-source routing server *GraphHopper*[8] *Java* library.

[8]https://www.graphhopper.com/

Chapter 5

Route Recommendation System Framework and Implementation

Contents

5.1 Development environment 61
5.2 System architecture 61
5.3 Route request 62
5.4 System data inputs 62
5.5 Graphhopper 66
5.6 Route generator 68
5.7 Route recommendation 75

This chapter discusses the architecture, technical description and interesting algorithms for the implementation of our solution. Section 5.1 describes the development environment in which the project was built. Section 5.2 explains the overall *ROUTE* system architecture. Section 5.3 presents the structure of the requests made by the users. Section 5.4 describes the essential data that is inputted to the *ROUTE* system to achieve a solution. Section 5.5 presents the *Graphhopper* library and its configuration. Section 5.6 explains the different algorithms and calculations done to reach the solution. Section 5.7 presents the structure of a route and the response to the user.

[This page has been intentionally left blank]

Chapter 5
Route Recommendation System Framework and Implementation

5.1 Development environment

This system is built on the environment described in Table 5.1. It was developed on *Eclipse Java Oxygen*, an Integrated Development Environment (IDE) for *Java* development, on an *Asus* laptop with 16 GB of *RAM*, an *Intel Core i7* and *Windows 10 Home* operating system.

Other software tools used were the cross-platform web server *XAMPP*, which allows using the *MySQL* database management system; the *Maven* build automation tool which facilitates the use of multiple libraries; and the *MongoDB* database program.

Junit library was used for software testing; the JSON library to convert java objects to JSON and the reverse; the *MySQL* connector to connect the *Java* client application to the database; the *Graphhopper* library to access the routing engine and the *OpenWeatherMap* API to request weather data.

Table 5.1: Implementation list of the proposed recommendation system

	Component	**Description**
Hardware	Computer	ASUS X541UV laptop
	Operating System	Windows 10 Home
	RAM	16 GB
	CPU	Intel ® Core ™ i7-6500U CPU @ 2.50GHz 2.59 GHz
Software Tools	Eclipse Java Oxygen	Eclipse Java Oxygen 2018-04 for Java development
	XAMPP	XAMPP Control Panel v.3.2.2
	Maven	Apache Maven 3.6.3
	MongoDB	MongoDB 3.7.0
Programming	Java	Java version 8 update 231
	SQL	Structured Query Language
Libraries and API's	Weather API	OpenWeatherMap API
	Directions	Graphhopper 0.13 library
	Junit	Junit 4.13 library
	JSON	org.json library
	MySQL	MySQL connector java

The developed implementation was coded using *Java* programming language and it is available on the *GitHub* repository of the *STC* project. All the classes referenced during the explanation of the implementation can be seen in the class diagram in appendix I.

5.2 System architecture

Figure 5.1 shows an overview of the *ROUTE* microservice. It is composed of three major modules: *System Data Inputs*, *Route Generator Algorithm* and *Route Recommendation*. The *System Data Input* receives data inputs from the users and the other microservices: user preferences,

tour constraints and POIs. Thereafter, the *Routing Generator Algorithm* module processes the input data into an algorithm to generate a feasible route recommendation. Finally, the *Route Recommendation* module sends the recommendation back to the user through the mobile application.

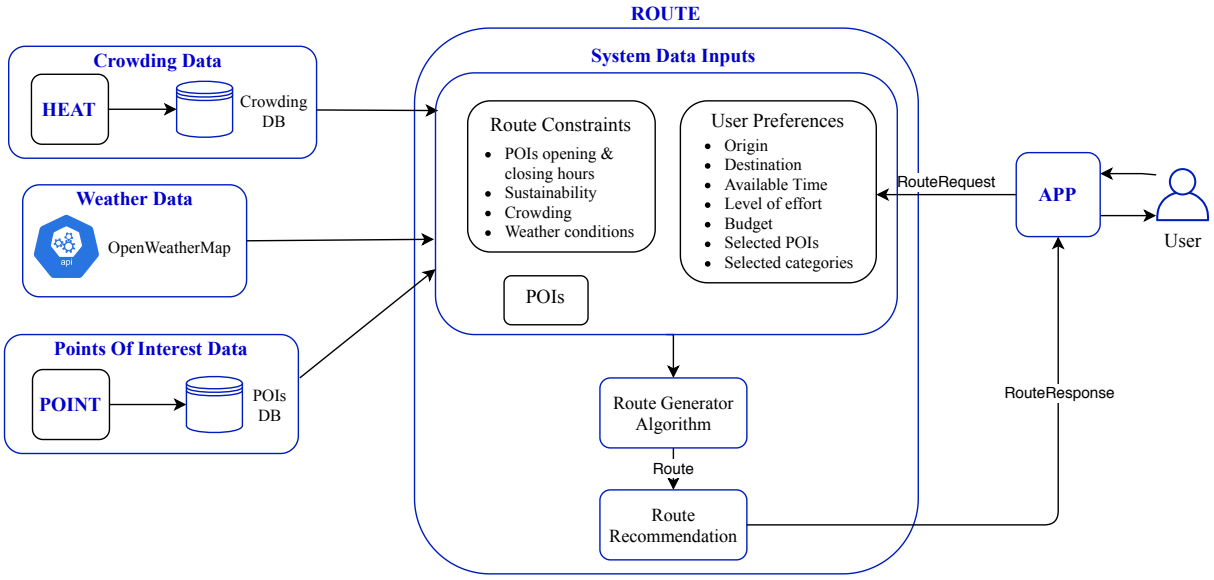

Figure 5.1: *ROUTE* solution framework

5.3 Route request

The first step is to handle all the user preferences and some constraints that are sent from the *APP* to the *ROUTE* in the form of a *RouteRequest* (structure example in Listing H.3). From the *APP* side the process of selection of these preferences is explained in the activity diagram in annex F. To handle this data from the *APP*, a class (*ConvertRouteRequest*) is dedicated to parse the JSON data into *RouteRequest* objects. Each *RouteRequest* object is then composed by the following attributes, which are explained in subsections 5.4.1 and 5.4.2: a *GHPoint* **origin**, a *GHPoint* **destination**, a *Timestamp* **departureDate**, an *int* **visitationTime**, an *int* **effortLevel**, an *int* **budget**, a *LinkedList<Integer>* of **selectedPoints**, a *List<Integer>* of **selectedCategories** and a *boolean* **checkWeather**. During the creation of these objects, an object *Calendar* is also created and initialized with the same date and hour as the initial time chosen by the user as the start time for the route. It is used for several comparison of dates and registration of timestamps during the process of searching a solution.

5.4 System data inputs

A collection of appropriate data is crucial for any successful projects. In this section we analyze the data that is collected by the *ROUTE* system to run properly and achieve a solution. Subsection 5.4.1 explains the user preferences. Subsection 5.4.2 explains the route constraints and their purpose. Subsection 5.4.3 shows the structure of the POIs and what data is obtained

from them. The user preferences, the route constraints and the POIs are then used to feed the algorithms that provide the route recommendation.

5.4.1 User preferences

In TRS, there are many applications of RS, most of them, learn the user preferences to recommend places or attractions according to the user interests [47]. In this system, users' preferences are specified and collected from the mobile application (*APP*) through *RouteRequest* messages.

The aspects on which users must make decisions are:

- **Origin** and **destination** - to begin, users must decide the origin of the route and the destination. The local departure can either be the current location of the users or a different location chosen by them, as well as the arrival place;
- **Departure date** - users can decide the departure date of the route, so that it starts at the moment chosen by them. It can either be the current date or a different date chosen by them;
- **Available time** - users determine the available time for the route, so that the total time of the recommended route do not exceed their limited time for visitation;
- **Level of effort** - users must decide the maximum level of effort of the route. In order to keep the route adequate in terms of physical effort and maintain it comfortable for each user, they can choose between three levels of effort (more details in Section 4.4, Table 4.4);
- **Budget** - users should state the amount of money they have available for the route so that the route suggests POIs which sum of the visiting prices does not exceed that amount;
- **Selected POIs** - users are allowed to choose a set of POIs through which the route must pass. They can also leave the set of POIs empty;
- **Categories of POIs** - users can define, choosing from a list of options (more details in Section 5.4.3, Table 5.2), which are the categories of POIs that they prefer so that it enables the suggestion of routes that meet their interests;
- **Check the weather** - one more option is the preference to check the weather. If the user selects that option, there will be a concern about rain when the recommendations are done. If not, then it does a *'normal'* recommendation (more details in Section 4.3).

5.4.2 Route constraints

In the existing literature, different constraints are chosen amidst what the authors consider pertinent to their problems and their study's objectives. Route constraints are aimed at solving difficulties with the destination domain that affect the tourist and the route. As already stated, some of these route constraints are defined by the user's preferences and choices (see Section 5.4.1).

However, some constraints condition the suggestion of routes and do not depend on the user's preferences:

- **Sustainability level** - the sustainability level of the POI is evaluated and influences the priority in the order of recommendation of the POIs (more details in Section 4.1);
- **Crowding level** - the crowding data about the different areas of the city permits to avoid the most congested areas in the suggestion of routes (more details in Section 4.2);
- **Opening** and **closing hours** - the recommended route should be practicable with all the opening and closing hours (schedule) of every recommended POI;
- **Weather data** - the weather condition data is useful to filter the POIs that a user should visit according to the appropriateness of the weather (more details in Section 4.3).

5.4.3 POIs

The POI database referred in Section 3.3.1 was implemented in *MySQL*[1] on a *PHPMyAdmin*[2] (a free software tool to handle the administration of *MySQL* databases) and stores information about 39 POIs of the region of *Santa Maria Maior* in Lisbon, Portugal. Every POI is characterized following the structure of the class diagram illustrated in Figure 5.2.

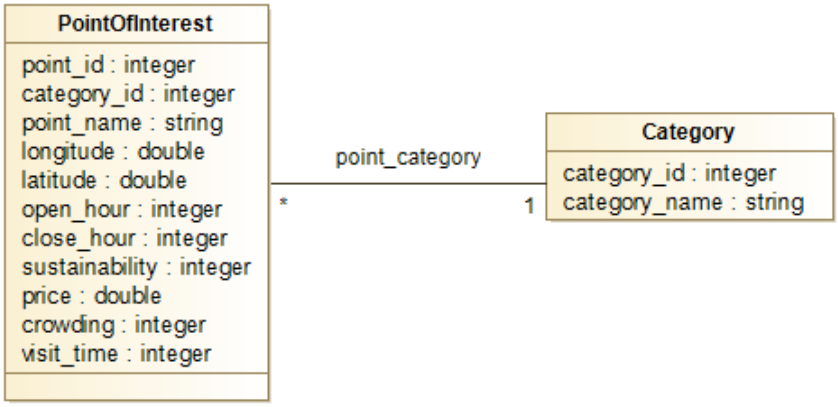

Figure 5.2: POIs database class diagram

From the database, the *ROUTE* gets the needed attributes to suggest a POI in a route. As a unique identifier of each POI stores the *point_id*; to describe the name of the POI to the tourist stores the *point_name*; to position the POI in space gets the geographic coordinates (*longitude*, *latitude*); to know what is a normal price to pay in each POI and to do an estimate of the budget needed for the route gets the *price*; to promote sustainable POIs and guarantee more sustainable routes, the *sustainability* level associated to each POI is also stored (more details in Section 4.1). To know how much crowded each POI is, gets the crowding level (more details in Section 4.2).

Moreover, as we can see in the class diagram (Figure 5.2), it is mandatory that each POI only have precisely one category associated. On the other hand, a category can be associated with many POIs. The *category_id* attribute corresponds to a set of predefined categories as they are catalogued in Table 5.2 that is used to adjust the tourist preferences so that they visit POIs

[1] https://www.mysql.com/
[2] https://www.phpmyadmin.net/

of their interest. Maintaining the set of categories fixed gives the possibility to filter the POIs suggested by category.

Table 5.2: POIs categories

Category_ID	Category
0	NA
1	Local Store
2	Religious Spot
3	Viewpoint
4	Restaurant
5	Tasquinha
6	Museum
7	Monument
8	Square

Another needed value is the attribute *visit_time*. This is an average of the real-time (in minutes) that a tourist spends visiting a POI and was obtained by searching in the official websites of the POIs. Having the expected visiting duration is useful to compute the total time duration of the route to be suggested. Finally, for a POI to be visited, it must be open, which means that the opening and closing hours are also important metadata to be stored. There must be a table schedule for each POI that contains *open_hours* and *close_hours* for each *weekday* (day of the week). However, to simplify the problem, it was only considered one hour to open and another hour to close, for each POI, that is equal for every day of the week.

In Table 5.3, we have an example of the data for a POI stored in the database. It has the identification number 13, is located in the coordinates (38.7139092, -9.1334762) and has the name *"Castelo de São Jorge"*. Also, it has a sustainability value of 84 and fits into category number 7, which is a monument (consult Table 5.2). The POI opens at 9 a.m. and closes at 6 p.m., a typical price for a ticket to visit the place is 10€ with an average visiting duration of 70 minutes. At the moment that the data was obtained the level of crowding was 4.

Table 5.3: POI metadata

Attributes	Values
point_id	13
category_id	7
point_name	*Castelo de São Jorge*
longitude	-9.1334762
latitude	38.7139092
open_hour	9
closes_hour	18
sustainability	84
price	10
crowding	4
visit_time	70

5.5 Graphhopper

As explained in Section 4.7, we use the *GraphHopper* java library to create our routes. Then, in Subsection 5.5.1, we explain how to create and configure a *GraphHopper* server that provides the necessary tools to obtain our solution routes. In Subsection 5.5.2 is explained the constitution of the graph provided by the *GraphHopper* server. In Subsection 5.5.3, it is described what the nodes of the graph are and what are its weights. In Subsection 5.5.4, it is explained what the edges of the graph are and what are its weights.

5.5.1 Graphhopper server

After receiving a *RouteRequest*, it is necessary to create a *Graphhopper* instance, if it is the first time that a request is made, or to load a *Graphhopper* instance if it was already created before. The goal of creating this instance is to produce a graph from the map area that is provided in a specific file. Therefore, in this step, a *Graphhopper* instance has its settings configured to be used in the following manner.

A folder location to store the *Graphhopper* data is set, and an OSM xml or pbf file is provided. This is the file that is parsed and allows that a graph is created from it. The vehicles that can be read by this *Graphhopper* instance are specified, which in this case is only one – foot, and an encoding manager defines how data from this type of vehicle is written and read into the edges of the graph. The functionality of storing elevation data is enabled, and the elevation provider is set. The elevation data is provided by a file containing high-resolution topographic data generated from NASA's Shuttle Radar Topography Mission (SRTM)[3]. Finally, the provided data in the configuration is loaded from the file system and creates a graph. Depending on the settings, the resulting graph will be stored to file system, so on a second call, this method will only load the graph from the file system, which is usually a lot faster. In Figure 5.3, it is possible to observe the sequence diagram of the creation and configuration of the *Graphhopper* server.

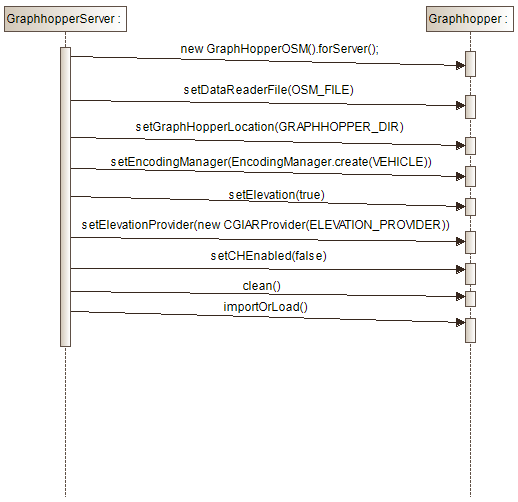

Figure 5.3: Sequence diagram of *Graphhoper* server creation and configuration

[3]https://www2.jpl.nasa.gov/srtm/

5.5.2 Graph

As said before, the *Graphhopper* instance creates a graph by parsing an OSM pbf file. In this case, the graph that is created reproduces a network of the *Santa Maria Maior* parish, because the content of the OSM file is the metadata of a real map of that area. In this process, the *Graphhopper* stores specific relation attributes required for routing. It reads nodes from the OSM file and stores latitude and longitude information into the intermediate data structure, and it also reads ways from the OSM file and creates edges. Ways are ordered lists of nodes (lines) that can represent roads, parks, traffic passes, etc. This results in a graph containing 10729 edges (representing the streets) and 7286 nodes (representing streets' intersections).

5.5.3 Nodes

Initially, the graph nodes created only represent street intersections. In this case, the weights stored in those nodes are the spherical coordinates (latitude, longitude and elevation). However, a graph node can also represent a POI. In this situation, more values are stored in addition to the ones stored in a street intersection.

Succinctly, the weights included in the POIs nodes (comprising the weights of the street's intersections) are:

- **Spherical coordinates**: x_i-latitude, y_i-longitude, z_i-elevation (degrees);
- **Crowding**: cv_i-crowding value (see Section 4.2);
- **Sustainability**: s_i-sustainability value (see Section 4.1);
- **Category**: c_i-category (see Table 5.2);
- **Visiting time**: vt_i-visiting time (minutes);
- **Opening hour**: o_i-opening hour ($[0,\ldots,23]$);
- **Closing hour**: f_i-closing hour ($[0,\ldots,23]$);
- **Price**: p_i-price (euros)

where $i = 1,\ldots,N_i$, and N_i is the number of nodes.

Most of these weights were already explained in sections 5.4.1 and 5.4.2. About the elevation, it is useful to calculate the slope associated with edges and then the physical effort of walking that edge (street).

Obtaining the POIs as nodes is possible due to the connection established with the database described in Subsection 5.4.3. For the connection with the database, it is used a *MySQL jdbc driver*, and the request of the POIs is made through the execution of an SQL query (query 1). This query returns a list of all POIs existing in the database which can be introduced in the graph with the corresponding weights.

Then, to represent a POI as a node in the graph, it is used specific methods of *Graphhopper*, that given a specific latitude and longitude they return the closest edge, that is, the nearest place on the graph structure. This allows the change made to the graph to be minimized and the streets to be used to walk directly into and from the POIs.

mySQL Query 1 Replicating POIs

```
SELECT point_id, point_name, longitude, latitude, sustainability,
opens_hours, closes_hours, category_id, price, crowding, visit_time
FROM point_of_interest;
```

5.5.4 Edges

The edges of the graph represent a street or a section of a street that exists on the map. Each edge connects to two end-nodes: a node n_i (*"initial"* node) and a node n_{i+1} (*"final"* node), and is also associated with some weights, namely:

- **Crowding**: cv_j-crowding value (see Section 4.2);
- **Distance**: d_j-distance (meters);
- **Time**: t_j-time (milliseconds);

where $j = 1,\ldots,M$, and M denotes the total number of edges.

The crowding values permit to know the level of congestion of the streets and suggest routes that avoid streets (edges) with high values of crowding. The time and distance are estimated by the *Graphhopper* and are used to calculate the physical effort (see Section 5.6.4.2). The time of each edge is also important to estimate the route total time, because it must not exceed the user's established visiting time.

5.6 Route generator

When a query (*RouteRequest*) from a user is received, the process that results in a *RouteResponse* starts. The Route Generator is mainly composed of the steps explained in the following subsections, where it is described the iterations taking place in the implementation that lead to the terminal route solution. In Subsection 5.6.1, the algorithm used to verify and recommend POIs is explained. Subsection 5.6.2 explains how weather conditions influence the recommendation of POIs. Subsection 5.6.3 describes the creation of different possibilities of routes that can be suggested to the user. Subsection 5.6.4 explains the verification of those route possibilities to understand if a user can really travel them and how the best route is filtered and selected.

5.6.1 Points of interest selection

The process starts with an algorithm that is a selector of feasible POIs for the route. There is a need to check three characteristics of the *RouteRequest*:

- the number of POIs chosen by the user - $Unumber_{pois}$;
- the amount of time that a user has left after visiting the POIs chosen by him - $Utime_{left}$;
- the users' budget - U_{budget}.

Checking for those constraints will allow obtaining an adequate result list of POIs to be suggested in the route ($result_{list}$). The intention is to understand if it makes sense to include more POIs than the ones chosen by the users.

First of all, the number of POIs chosen by the user must be, at the maximum, equal to the number of POIs established as the maximum for a recommendation in the system (max_{pois}), which was defined as being five (axiom 5.1).

Axiom 5.1 $Unumber_{pois} \leq max_{pois} \rightarrow Unumber_{pois} \leq 5$

If the number of POIs chosen by the user is equal to five, then there is no change to that list. However, if it is less than five, it makes sense to check the other constraints (axiom 5.2).

Axiom 5.2 $Unumber_{pois} < 5$

Then, the following step is a verification to see if the time that a user has available for visitation ($Uvisit_{time}$) is longer than the estimation of visit time for the POIs s/he chose ($EstimationTime_{pois}$). The estimation of the time for visiting the POIs chosen by the user is done by adding the visiting times for all POIs (vt), that is:

$$EstimationTime_{pois} = \sum_{i=1}^{N} vt_i \tag{5.1}$$

where $i = 1, \ldots, N$ and N is the number of POIs (nodes).

When a user chooses its POIs, the *APP* assures that the minimum value of a user available time for visitations is equal to the sum of the times of visit of each selected POI.

Axiom 5.3 $min(Uvisit_{time}) = EstimationTime_{pois}$

So, if the estimation of visit time for the POIs is equal to the time that a user has available for visitation, then it means that the user has no time left for visit more POIs and it is not possible to recommend more POIs. On the other hand, if it is smaller, then it means that the user has still time to visit more POIs than the ones s/he chose ($Utime_{left}$).

Axiom 5.4 $EstimationTime_{pois} < Uvisit_{time} \rightarrow Utime_{left} > 0$

After checking the last constraints, it is necessary to compare the budget of the users with the cost of visiting the POIs chosen by them ($TotalCost_{pois}$). Here, again, the *APP* assures that the minimum value of a user's budget is equal to the sum of the cost of visiting the POIs chosen by them.

Axiom 5.5 $min(U_{budget}) = TotalCost_{pois}$

So, if the users' budget is greater than the sum of the cost of visiting the POIs, then the user still has money left ($Ubudget_{left}$) to visit non-free POIs. But, if they are equal it can only be suggested free POIs.

If axioms 5.2 and 5.4 are verified, then it is possible to suggest more POIs to the user. To enable the customization of the route POIs, it is necessary to look for the set of favourite

categories, C, of the user. What is intended is that the suggested POIs must respect at least one category in the set:

$$\exists_{i=1}^{N} c_i \in C \tag{5.2}$$

Then, the POIs contained in the set of favorite categories are sorted in order to get the ones that maximize the values of sustainability (s_i), according to the following algorithm:

Algorithm 1: Suggest Point Of Interest Algorithm

1. Find the P POIs existing in the database, that have not been chosen by the user, belonging to C;
2. Sort P by the value of sustainability (s_i);
3. **While** $result_{list}$.size() $< max_{pois}$ **and** $Utime_{left} > 0$
   For each poi k in P:
   if $p(k) < Ubudget_{left}$ **and** $vt(k) < Utime_{left}$ **do**
   $result_{list}$.add(k);
   $Ubudget_{left}$ - $p(k)$;
   $Utime_{left}$ - $vt(k)$;

5.6.2 Weather request

As previously explained in Section 4.3.1, a request for obtaining weather data, particularly the rain probability feature, can be obtained by sending a request to the *OpenWeather* API. However, this process is only initialized if the user states its will, in the *APP*, in which the weather data must be checked, that is, when the *checkWeather* boolean from the *RouteRequest* is true.

The structure created for these requests to the *OpenWeather* API is an *OWMRequest* object which then can transform the JSON data (Listing A) it gets into *WeatherData* objects. Each object of *WeatherData* is composed of a value of rain probability of precipitation (rp), that varies between 0 and 1 (axiom 5.6), and a date (r_{date}).

$$WeatherData = (rp, r_{date})$$

Axiom 5.6 $0 \leq rp \leq 1$

Each user establishes a date to initiate a route, $departure_{date}$ and an interval of time within s/he is available to visit POIs ($Uvisit_{time}$). A date format is *'DD-MM-YYYY HH:MM:SS'*, so it is possible to split that information and get days, months, years, hours, minutes, seconds and more. This means that the route has an expected final time, FT, which is equal to the sum of the time of the departure date with time the user has available for visiting POIs (equation 5.3).

$$FT = departure_{date} + Uvisit_{time} \tag{5.3}$$

The weather data is filtered, only being considered the rain probabilities that have a date that is between the date that the user established to initiate the route until the expected final time of the route.

Axiom 5.7 $departure_{date} \leq r_{date} \leq FT$

If there is no rain forecasted to that period, then the filtering results in empty weather data objects, and it does not affect the recommendation of POIs. However, if it is not returned empty, then the process is the same as explained in Section 5.6.1 with little differences.

In this case, there is a verification if there is any raining probability (rp) greater than or equal to 50%: $rp \geq 0.5$.

If that is true, then there is a set of categories pre-defined as suitable for rainy weather (C_{rain}). If there are POIs in the user-selected POIs that do not fit into those categories, they are removed from the list. Then if there is space to add POIs to the result list, they are just added between the POIs that belong to the set of POIs suitable for rain.

5.6.3 All scenarios

After getting the definitive set of POIs to be suggested in the route ($result_{list}$), they are ordered in all different ways possible, without repetition, to have all the possibilities of visiting those POIs in a different order. Basically, we have a class (see class poiCombination) that permits to do permutations of a set of POIs in the same way we do permutations in mathematics. The following theory explanation is based in [25].

Permutations are isolated cases of simple arrangements (equation 5.4). Having any ordered sequence in hand with a number n of distinct elements, any other sequence formed by the same n reordered elements is called a permutation.

$$A_{n,k} = \frac{n!}{(n-k)!} \tag{5.4}$$

For it to be a simple permutation, the number of elements of the resulting new sets, k, must be equal to the number n of distinct elements of A, resulting in the equation 5.5.

$$A_{n,n} = n! \tag{5.5}$$

Let's imagine we have a set of POIs, $Result_{list} = I, L, S$, then the list with all the simple arrangements of the three elements of $Result_{list}$ taken 3 to 3 is as follows:

$$ILS, ISL, LIS, LSI, SIL, SLI.$$

This way, we can say that if A is a permutation of B, then A and B are made up of the same elements but ordered differently. So, for example, when we have a set of five POIs to be suggested, we have a result of 120 different possibilities of visiting them (equation 5.6), which we will call *scenarios*.

$$A_{5,5} = 5! = 120 \tag{5.6}$$

5.6.4 Filter scenarios

After getting the set of resulting scenarios obtained from the process explained in 5.6.3, they are iterated with the goal of finding the best path p that respects the following criteria:

- the departure date established by the user, the opening and closing hours of the POIs, the visit times of the POIs, the user arrival times at each POI and the expected final time of the route must be feasible with each other (see Subsection 5.6.4.1);
- the level of effort of the route must be smaller or equal than the level of effort chosen by the user (see Subsection 5.6.4.2);
- the relation between the distance that is covered and the crowding that a path crosses must be the minimum possible (see Subsection 5.6.4.3).

When iterating each scenario, it is necessary to look to the data that we have in each of the recommended POIs (nodes) and look at the data about the edges between those nodes. Lets look for the example in Figure 5.4, where we have a node a, a node b and an edge e between those two nodes.

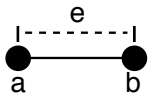

Figure 5.4: Representation of an edge e between node a and node b

Graphhopper is capable of returning useful information about the edge e in the form of *PathWrappers*, which are objects that hold the data points like the distance between the nodes, instructions (how to go from one node to the other), time of travelling between nodes, etc. This is the data that will be used to reduce the scenario options to the best one.

The desired result is the best oriented path, p, composed by nodes (street intersections and POIs) and edges (streets) that connect the nodes that solves equation 5.7.

$$p = (n_0, n_1), (n_1, n_2), \ldots, (n_{k-1}, n_k)$$

where n_i, $i = 0, 1, \ldots, k$, represent the nodes in the path and (n_j, n_{j+1}) each of the path edges.

$$\begin{aligned} min \quad & \left(\sum_{j=1}^{M} cv_j, \sum_{j=1}^{M} d_j \right) \\ max \quad & \left(\sum_{i=1}^{N_p} s_i \right) \\ s.t. \quad & c_i \in C,\ i = 1, \ldots, N_p \\ & o_i \leqslant departure_{date} + tt_i \leqslant f_i,\ i = 1, \ldots, N_p \\ & e_j \leqslant Effort, j = 1, \ldots, M \end{aligned} \tag{5.7}$$

5.6.4.1 User and POIs time

For each scenario, the visit times of the POIs (vt_i) are stored along with the open hour (o_i) and closing hours (f_i) to be analyzed against the user arrival times at each POI.

The departure date of a user, $departure_{date}$, must be feasible with the opening hours of every POI in the scenario. At the same time, the expected final time, FT, must be feasible with

the closing hours. However not only that needs to happen, but also the time when a user arrives at each POI needs to be smaller than its closing hour.

Let tt_i be the time of travelling from the first node in the scenario until the POI i:

$$tt_i = \sum_{j=1}^{M_i} t_j + \sum_{f=1}^{N_i} vt_f \tag{5.8}$$

where M is the number of edges, t_j is the time of travelling the edge, N_i is the number of POIs in the scenario and vt_f is the visitation time of each POI.

For a scenario to be accepted, for each POI i with an o_i and a f_i, o_i must be smaller than $departure_{date} + tt_i$ and f_i higher than $departure_{date} + tt_i$, that is:

$$o_i \leq departure_{date} + tt_i \leq f_i$$

5.6.4.2 Effort

A user can decide the maximum level of physical effort for the route, $Effort$. As a result, each edge e of the scenarios, must have a level of effort that is equal or lower than the level set by the user (equation 5.9).

$$e_j \leqslant Effort, j = 1, \ldots, M \tag{5.9}$$

To calculate the effort of walking down the streets, we used the formula of effort, e_j, in METs, that is used in treadmills[4] (equation 5.15), where $j = 1, \ldots, M$, and M stands for the total number of edges in a route. This equation allows estimating the METs and the maximum rate of oxygen consumption measured during incremental exercise (VO_2) given the treadmill speed and the grade. In this case, it allows estimating the METs given the walking speed in miles per hour (mph), W_{speed}, (equation 5.10) and the slope of the streets in percentage (%), s_j, (equation 5.11).

$$W_{speed} = \frac{d_j}{t_j} \times 2.236936 \tag{5.10}$$

where d_j is the walking path distance in meters, t_j are estimated duration of walking in seconds and 2.236936 is a constant to transform speed from meter per second (m/s) to mph.

$$s_j = \frac{z_{i+1} - z_i}{d_j} \tag{5.11}$$

where z_i is an *"initial"* node altitude and z_{i+1} is a *"final"* node altitude, both in meters, of the two associated end-nodes of each edge. The altitude of each node is obtained from the SRTM file that is provided to the *Graphhopper* instance. The slope is equivalent to the treadmill grade, which is a measure of the height distance for every 100-horizontal distance, e.g. a rise of 15 meters for every 100 meters is a 15 % grade.

With the previous information calculated it is then possible to calculate the VO_2 rate (equation 5.14) that consecutively needs the calculation of the horizontal component (HC) and

[4] http://www.csecho.ca/wp-content/themes/twentyeleven-csecho/cardiomath/?eqnHD=stress&eqnDisp=mvo2tm

the vertical component (VC), in equations 5.12 and 5.13, respectively. The representation of those components is possible to observe in Figure 5.5.

$$HC = W_{speed} \times 26.8 \times 0.1 \tag{5.12}$$

$$VC = W_{speed} \times 26.8 \times 1.8 \times s_j \tag{5.13}$$

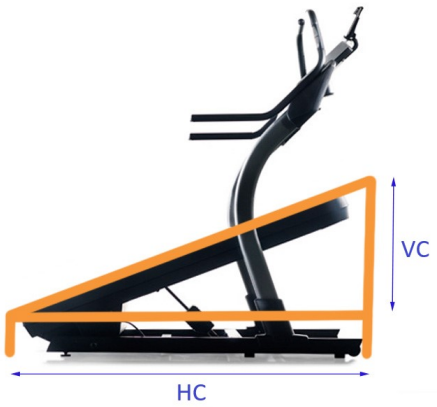

Figure 5.5: Representation of the horizontal and vertical components on a treadmill

$$VO_2 = HC + VC + rest \tag{5.14}$$

where *rest* is, as said before, the amount of oxygen consumed at rest ($3,5ml$ of $O_2/kg/min$).

Finally, it is possible to calculate the effort equation e_j:

$$e_j = \frac{VO_2}{3.5} \tag{5.15}$$

Basically, the effort equation uses the slope value of the respective end-nodes (can be calculated using the altitude), the walking distance between the nodes and the estimated travel time between nodes. In the end, the maximum level of effort of the edge that gets more effort to travel is compared to the maximum level that the user is apt to do. If it is higher than the effort level chosen by the user, then it is necessary to find an alternative route.

5.6.4.3 Crowding and distance

The key goal is to propose routes that escape already overcrowded areas and try to prevent more congestion. Alternatively stated, it is supposed to minimize the total crowding value of the route. On the other hand, there must continue to be a balance with the distance travelled during the route.

To reach this goal, it is proposed to find the best *scenario* that minimizes both criteria: distance and crowding (equation 5.16). The desired result is the path p that minimizes each criteria.

$$min\left(\sum_{j=1}^{M} cv_j, \sum_{j=1}^{M} d_j\right) \tag{5.16}$$

In order to compute this, there is an iteration for each scenario where the distances of walking between the *origin*, the POIs and the *destination* are added. At the same time, the maximum value of crowding found in the path between the *origin*, the POIs and the *destination* are also added.

$$crowding = \sum_{j=1}^{M} Max(cv_j) \tag{5.17}$$

$$distance = \sum_{j=1}^{M} d_j \tag{5.18}$$

As already explained before, the values of distance are obtained from the *pathwrappers*, but the values of crowding are provided by the *HEAT* microservice. Once that the *HEAT* was not concluded on time, it didn't calculate the values of crowding but what it does is substitute the crowding value with the elevation of each coordinate. The elevation values are based on what is shown in the contour map of *Santa Maria Maior* parish council C.1. On this map we have contour lines, which are lines that joins points of equal elevation above the sea level, and contour intervals that shows the variance in elevation between the lines. This way, instead of avoiding the most crowded paths, it is supposed that the algorithm avoids the paths that contains the coordinates with higher elevation.

Subsequently, the way that was found to balance distance and crowding was by creating a variable named $ratio$. In equation 5.19, it is possible to see how it is calculated. We divide one by the result of the multiplication between the crowding and the distance, which allows the value returned by the equation to always have an upper and lower limit.

$$ratio = \frac{1}{distance \times crowding} \tag{5.19}$$

The goal is to search for the *scenario* that returns the highest value of $ratio$, because we want the lower value of $distance \times crowding$. In the end, it is expected that the algorithm returns a route which distance covered is not so big and that is attracted to less high places.

5.7 Route recommendation

The final stage is then the route recommendation. In Subsection 5.7.1 it is explained the structure of a route. In Subsection 5.7.2, it is described the construction of a *RouteResponse*, which is the object representation of the information sent to the user. These objects are then included in the *RouteResponse*, explained in Section 5.7.2.

5.7.1 The route

In the end we have the best route. A route is an object containing a polyline (list of coordinates of the best path); the list of POIs associated with a timestamp; the route total travel time; the route total travel distance; the average sustainability of the recommended POIs; the price of visitation of the POIs; the coordinates of the local of departure and of the destination; the start time and end time of the route; and the number of calories burned during the route.

5.7.1.1 Best path

The polyline of the route is represented by an oriented path, p, composed by nodes (street intersections and POIs) and edges (streets) that connect the nodes.

$$p = (n_0, n_1), (n_1, n_2), \ldots, (n_{k-1}, n_k)$$

where n_i, $i = 0, 1, \ldots, k$, represent the nodes in the path and (n_j, n_{j+1}) each of the path edges.

5.7.1.2 Timed POIs

The route object contains a list of all the POIs included. Each of those POIs is associated with a timestamp that represents the estimation of the moment that a user arrives there. If we have the example of Listing 5.1, it means that it is expected that the user arrives at the *Arco da Rua Augusta* on the 15 of October of 2020 at 23:31:39.

Listing 5.1: Timestamp and POI example

```
{
        "timestamp":1602804699000,
        "poi": {
            "name":"Arco da Rua Augusta",
            ..., // Omitted for brevity
        }
}
```

5.7.1.3 Total route time

By adding the visiting times for all nodes that are POIs and the walking time at each edge, the total route time is determined (equation 5.8).

5.7.1.4 POIs average sustainability

The average sustainability of the suggested POIs is calculated by adding the value of sustainability of each POI and dividing by the total number of POIs suggested.

$$\frac{\sum_{i=1}^{N_p} s_i}{N_p} \tag{5.20}$$

5.7.1.5 POIs price

The total price of the suggested route is calculated by adding the price of visiting each POI.

$$\sum_{i=1}^{N_p} p_i \tag{5.21}$$

5.7.1.6 Walking calories estimation

Seizing the opportunity of using some of the calculations applied in the effort estimation, it is provided to the users the estimate of how much calories they will expend when taking the recommended route.

The equation used to express the walking energy expenditure is based on the American College of Sports Medicine (ACSM) investigation [79], and is stated as:

$$Cal = (0.1 \times S + 1.8 \times S \times s_j + 3.5) \times BM \times t \times 0.005 \tag{5.22}$$

where Cal is the walking energy expenditure in kilocalories, S is the walking speed in meters per minute, s_j is the slope in decimal form, BM is the tourist's body weight in kilograms and t is the walking time in minutes.

To use equation 5.22, it is necessary to make some assumptions. For S, we should assume that the walking speed of the tourist is regular during the walking time t of the route, and, according to [44], must be in the interval between 1.9 and 3.7 *mph* so that the values of Cal are considered precise. By doing some tests to *Graphhopper*, we understood that the engine considers the tourist to travel around 3.0 *mph*. Another assumption is about body weight. It would be more accurate if the body weight were asked to the tourist, but the calculation is being made with the belief that the user weights 65 kg. The other assumption is about the slope s_j, which should be homogeneous along the way. To ensure slope homogeneity, a walking path may be split into n walking segments where each segment has homogeneous slope.

This will result in the total energy expenditure (equation 5.23) for all the walking segments as:

$$Cal_{total} = \sum_{i=1}^{n} Cal_i \tag{5.23}$$

where Cal_i is the energy expenditure of the ith segment, estimated by equation 5.22, n is the number of edges.

5.7.2 Route response

The route is then introduced in a *RouteResponse* object (*RR*), which is also composed by a *code* and a *justification*.

$$RR = (Route, code, justification)$$

The *code* that is attached is based on the HTTP status codes. Based on [37], HTTP status codes are defined codes released by servers in response to client's requests. Those are usually composed of three digits where the first digit defines a class of response. Looking for those existing classes, the one that best fits our case is class number two, which is related to successful requests. The *APP* assures that the requests can always be successfully received and understood by the *ROUTE*, however the response might contain content or not. Therefore, we chose to respond with the code *"200"* when we have a recommendation that fulfils the user preferences and the constraints, and with code *"204"* when the recommendation is unsuccessful for some reason and is not returning any route.

To explain to the user the reason why a request is successful or unsuccessful, we have then the justification field, which contains standard phrases accompanying the codes. With a *code* = 200 we have a message for the user just informing that process went well, e.g. *"The request got a successful recommendation"*. On the other hand, if it is a *code* = 204 we inform the user of what went wrong, e.g. *"There is not any possible recommendation regarding the combination of the chosen POIs, their schedule and the user available time"*.

In the end, the *RouteResponse* objects are converted to JSON so that they are able to be sent to the *APP* again, with a format example shown in Listing H.8 (the code and the justification are omitted).

Chapter 6

Test and Validation

Contents

6.1 Validation environment and dataset 81
6.2 Functional validation 81
6.3 Use case testing 82
6.4 Validity threats 87
6.5 Summary 88

In this chapter, we present the results gathered from all the tests done to our route generation algorithm to validate our solution. Section 6.1, illustrates the validation environment and data used for the tests. Section 6.2, explains the purpose of validating and what type of validation is done. Section 6.3 defines the different scenarios of validation and its results. Section 6.4 identifies the validity threats detected in the validation tests. Finally, Section 6.5 summarizes the conclusions.

[This page has been intentionally left blank]

Chapter 6
Test and Validation

6.1 Validation environment and dataset

The following experiment and its tests were reached using the same environment as the one illustrated in Table 5.1. A *Java* environment (*Eclipse Java Oxygen 2018-04*) running in an *ASUS X541UV* laptop, with an *Intel ® Core ™ i7-6500U CPU @ 2.50GHz 2.59 GHz*, 16 GB of RAM and a *Windows 10 Home* operative system. Also, the internet connection was of *100 Mbps* for download (*ethernet*).

This study uses a real dataset of 7286 nodes and 10729 edges from the *Santa Maria Maior* perish in Lisbon, Portugal. We also use a dataset of 39 POIs based on real and fictional data which has the following statistics:

- Includes POIs from eight different categories: five *'Local Stores'*, six *'Religious Spots'*, four *'Viewpoints'*, six *'Restaurants'*, two *'Tasquinhas'*, seven *'Museums'*, six *'Monuments'*, and three *'Squares'*;
- The average sustainability level of the POIs is 74.2%;
- Half of the POIs are free to visit, and the average price of visitation for the other half is 9.25€;
- The average visitation time of each POI is 39 minutes;
- The opening and closing hours vary in the range between $[0, 24]$ hours.

The crowding values were generated for the POIs by the *HEAT* and are based on the altitudes of each coordinate, that is, if a point has an elevation of 60, then its value of crowding is also 60.

6.2 Functional validation

Validation is an attempt to ensure that the software accomplishes its specific intended purpose [18]. For functional validation to happen, it is necessary to run tests that meet our goals, this is, to carry out tests in view of specific objectives. To do our validation, we opted for acceptance testing in which we use test cases that cover the typical scenarios under which we expect the software to be used. Test cases can be designed to check that the functional specifications are correctly implemented [70]. Acceptance testing concludes whether a system satisfies its acceptance criteria, generally by checking the desired system behaviours against the requirements that were planned [89].

Thus, to better describe the functional requirements of the *ROUTE* microservice and to demonstrate the main use cases of the *APP*, we have created 5 scenarios that show the way we envision how a person would use the mobile app while justifying the reasons behind some of the main features. These scenarios represent some of the broad possible deployment options in

which we aim to apply our solution. By testing our solution in this way, we aim to envisage the behaviour of the solution in the development phase and adapt the system prototype in order to have the best behaviour possible.

The following test scenarios intend to experiment and cover various scenarios of a user requesting route recommendations. At the same time, there is the aim to show the results and validate the system. Build data with concrete points and concrete budgets and other concrete values and jointly test what comes from the application with the *ROUTE* microservice.

6.3 Use case testing

This section is divided into seven subsections. Subsection 6.3.1 is a contextualization for the scenarios explained ahead. From subsection 6.3.2 to subsection 6.3.6, different scenarios are tested to analyze the outputs. A test in these circumstances is a simulation of a request made by a user. It is followed by a call to the algorithm to provide a route according to the specific preferences and constraints and, lastly, the creation of a response to the user. The solutions presented are screenshots of the recommended routes taken from the real mobile application that was developed in this project. Subsection 6.3.7 analyzes the competence of the algorithm to maximize the sustainability levels. Subsection 6.3.8 shows the influence of considering the crowding levels in the output of the scenarios.

6.3.1 Scenario contextualizing

Let's assume Dinis, a tourist, is staying in a hostel at the *Santa Maria Maior* perish for a few days. While walking around Lisbon, he encounters an advertisement, which is part of a promotional campaign for our *APP*. Dinis becomes interested and decides to download the *APP* by scanning the *QR* code in the ad, which redirects him to the app store page. After downloading and launching the *APP*, he is welcomed with a map-based interface of the *Santa Maria Maior* perish. From this initial state, the *APP* uses Dinis' location to center him in the map and also displays a list of the city's POIs as markers on the map. Dinis can search and drag the map freely to explore the city as he pleases, and he can tap a marker to get more information on a POI or tap on any street to discover the street name. Additionally, Dinis can use the search panel to manually search for locations (e.g. where is the hotel he is staying) by typing an address, which the *APP* will geocode and then display its location on the map.

The system will use constraints and preferences to create a trip that maximizes sustainability and reduces crowding by recommending the right POIs. This trip is displayed to Dinis both in the map and in the form of a timeline, which he can use to confirm or reject it. After the trip starts, Dinis can consult the *APP* for guidance and information about the route and POIs he is visiting.

6.3.2 Test scenario #1

In this first scenario, Dinis decides to spend his afternoon visiting the city. Through the mobile application, he plans a route to depart immediately from his current location, the train station, and conclude at the hostel he is staying at. To do so, he has approximately 5 hours to spend

on visits and 50 euros available to spend. Additionally, he specifies he is interested in visiting churches and monuments, while not being sure of a specific attraction. Optionally, Dinis can change the effort level of his trip, and the system will adjust the physical effort of the route accordingly. All this information is represented in Table 6.1 and a representation of its *RouteRequest* is in Listing H.1.

Table 6.1: First scenario

Parameter	**Description**
Departure Time	15:30
Visitation Time	5 hours
Origin	Train Station – Rua 1º Dezembro
Destination	Hostel – Rua dos Douradores
Budget	50€
Selected POIs	-
Selected Categories	Religious Spots, Monuments
Effort Level	Vigorous
Check Weather	No

This scenario permits to demonstrate that the algorithm is capable of recommending a route when the user selects the preferred categories and do not select POIs. Looking for the *RouteResponse* in Listing H.2 and for the demonstration of the result in the mobile application (Figure H.2), it is possible to observe that it respects all the preferences and constraints. It suggested five POIs, the most sustainable within the chosen categories (religious spots or monuments); allows to start and end the route within the available time established and in the right origin and destination; the money needed for the visit is under the budget and the effort don't affect the choices because the level chosen by the user is *'high'* and that is the maximum level possible for a path.

6.3.3 Test scenario #2

In the second scenario, it is midday and Dinis decides that during the next five hours he is going to do a gastronomic tour. He specifically chooses five places to visit and eat (*Nicola Café*, *Casa Portuguesa do Pastel de Bacalhau*, *Martinho da Arcada*, *As Bifanas do Afonso* e *A Brasileira*) where he does not expect to spend more than 70 euros. All this information is represented in Table 6.2 and a representation of its *RouteRequest* is in Listing H.3.

Table 6.2: Second scenario

Parameter	**Description**
Departure Time	12:00
Visitation Time	5 hours
Origin	Current Location - Campo das Cebolas
Destination	Hostel – Rua dos Douradores
Budget	70€
Selected POIs	*Nicola Café*, *Casa Portuguesa do Pastel de Bacalhau*, *Martinho da Arcada*, *As Bifanas do Afonso*, *A Brasileira*
Selected Categories	All
Effort Level	Moderate
Check Weather	No

Once that the user specifically asked a route with maximum number of POIs, the goal is to show that, in this situations, the recommended route is the best path between the origin, the POIs and the destination. Looking for the result in Listing H.4 and Figure H.4, we can see that the route passes in the chosen points, is under the budget and within time. Regardless that all the categories are chosen as preferred, the expected result do not change and the effort level *moderate* permits to obtain a route solution as well.

6.3.4 Test scenario #3

In the third scenario, it is 10 in the morning, it is raining a lot in the city, but Dinis still wants to enjoy the holidays and visit new interesting places. So, he activates the weather conditions check button on the mobile app. He has heard of an exciting exhibition taking place at the *"Museu Nacional do Desporto"*, so he chooses to visit the museum and add this category to the preferences. Also, once the holidays are finishing, and Dinis intends to take some souvenirs to his family and friends, he intends to visit some local stores where he expects to spend around 50 euros. If it stops raining within the interval he has available for visitations, he also wants to visit some viewpoint. Dinis intends to leave from his hostel and want the route to end at the same place within the next three hours. This information is represented in Table 6.3 and a representation of its *RouteRequest* is in Listing H.5.

Table 6.3: Third scenario

Parameter	Description
Departure Time	10:00
Visitation Time	3 hours
Origin	Hostel – Rua dos Douradores
Destination	Hostel – Rua dos Douradores
Budget	50€
Selected POIs	*Museu Nacional do Desporto*
Selected Categories	Local Stores, Museums, Viewpoints
Effort Level	Vigorous
Check Weather	Yes

In this scenario, we test a round-trip route, the check weather functionality when it is raining, and a low number of POIs chosen by the user. Looking for the result in Listing H.6 and Figure H.6, we observe that the route effectively starts and terminates at the same place. The route suggests the selected POI and add others of the *"local stores"* and *"museums"* category, but do not add any *viewpoint*. This happens, as expected, because *"viewpoints"* is not an adequate category to recommend during rain. The route cost is under the budget, but in this case surpasses the available time of the user in 12 minutes (the duration is 3 hours and 12 minutes).

6.3.5 Test scenario #4

In the fourth scenario, Dinis decides to visit the city by night, because he wants a vision of Lisbon and its lights. He wants to departure right from the restaurant he just had dinner and does not want to spend more money. The idea is to go for a walk passing through some viewpoints

and squares, during the next three hours, until he arrives at the hostel. This information is represented in Table 6.4 and a representation of its *RouteRequest* is in Listing H.7.

Table 6.4: Fourth scenario

Parameter	Description
Departure Time	23:00
Visitation Time	3 hours
Origin	Restaurant – Largo do Picadeiro
Destination	Hostel – Rua dos Douradores
Budget	0€
Selected POIs	-
Selected Categories	Viewpoints, Squares
Effort Level	Moderate
Check Weather	No

In this scenario, we can show that the algorithm is capable of recommending a route when it needs to pass from one day to the next and the user has no budget. Looking for the result in Listing H.8 and Figure H.8, we observe that the route passes through free POIs, so the budget is accomplished, and starts at 11pm of one day and ends at 12:58 am. The budget, the available time, the effort, the origin and destination and the selected categories are all respected.

6.3.6 Test scenario #5

In the fifth scenario, Dinis has an appointment soon. However, he still wants to enjoy the next hour and a half, so he planned to visit the *"Igreja de Santo António"* quickly, and the *"Miradouro do Recolhimento"*. He pretends to leave from his current location and arrive at the local of the appointment.

Table 6.5: Fifth scenario

Parameter	Description
Departure Time	18:30
Visitation Time	1:30 hours
Origin	Current Location – Rua Garrett
Destination	Appointment Local – Rua Vítor Cordon
Budget	10€
Selected POIs	*Igreja de Santo António*, *Miradouro do Recolhimento*
Selected Categories	-
Effort Level	Vigorous
Check Weather	No

This scenario was made to show that the *ROUTE* system can effectively deal with the conjugation of departure time, the users' available time, the time of visitations of the recommended POIs and its opening and closing hours. Both the *Igreja de Santo António* and *Miradouro do Recolhimento* closes at 7pm, so starting a route at 6:30pm at *Rua Garrett*, walk to one of those POIs, visit one of them and walk to the other to arrive before it closes its impossible. That is the reason why the result in Listing H.10 and Figure H.10, shows that the configuration for the route is not feasible.

6.3.7 Sustainability validation

This section was created to analyze the competence of the algorithm to maximize the sustainability of the recommended POIs. To do this, we can observe Table D.1. In this table, we have a representation of the POIs available on the database to be included in the recommendations, along with its different attributes and values. Those POIs are ordered by category to facilitate the analysis and conclusions.

For this analysis, it is interesting to look to the *category_id* and to the *sustainability* value. Also, the last four columns are essential because they were created for each scenario explained before. In the columns about the scenarios, there is a symbol *"X"* that means that POI belongs to a category that the user chose and can be recommended in the solution. If it is just blank, then it can't be recommended. In the case that it has an *"S"*, then it means that the user chose that POI to be mandatorily included in the solution. Finally, the cells may be filled in green, which means that the POIs were included in the solution.

If we look at the green cells of each scenario, we can conclude that all POIs with an *"S"* were effectively included in the solution. Further, we can state that only POIs with an *"X"* were recommended, and they were all the ones that had higher values of sustainability.

6.3.8 Crowding test

This section was created to the extent of showing the influence of considering crowding when choosing the paths for a route. To do this, it was made a comparison, for each scenario explained before, between the algorithm described in chapter 5 and other algorithm that just searches for the shortest path between two points and do not consider crowding.

In Table 6.6 we can see the analysis that was made. Both algorithms were run for each scenario in order to calculate three variables: *total distance*, *total time* and *maximum crowding*. The *total distance* variable is the total distance of the route in meters. The *total time* is the total time of the route in minutes. The *maximum crowding* is obtained by adding the maximum value of crowding found in the path nodes between the origin and the first visited POI, between the other visited POIs and between the last POI and the destination (as it is explained in 5.6.4.3).

If the crowding avoidance is working well, it is expected that the values of crowding in the recommended routes that consider crowding are lower than the values of the recommended routes that were recommended based on the shortest path. In a simple way, it is expected that the algorithm that considers crowding will be attracted to places of interest that are low altitude places because the crowding is being simulated with elevation values and it is supposed to avoid high crowded points, which in this cases are high elevation points. On the other hand, it is expected that the algorithm that does not consider crowding recommends routes with a shorter distance and time, because it searches for the shortest paths.

Table 6.6: Comparison between the results of one algorithm that considers crowding and other algorithm that do not consider crowding

		Scenario 1	Scenario 2	Scenario 3	Scenario 4
Crowding considered	Total distance	4869	2547	2329	3606
	Total time	238	225	192	118
	Maximum crowding	310	230	160	330
Crowding not considered	Total distance	4599	2488	2329	3606
	Total time	235	224	192	118
	Maximum crowding	340	250	160	330

As we can observe in the table, for the first and second scenarios, the obtained result was the expected. The algorithm that considers crowding returned a lower value of the maximum crowding, while the algorithm that does not consider crowding returned routes solutions that are faster and shorter. However, for the third and fourth scenarios, the results were precisely the same because the route solutions were equal. To complement the information on the table, it is useful to consult Appendix G, which contains graphics that compare the two different algorithms. In those graphics, we have for each scenario the distance covered by the route and the crowding profile of the route along with the distance.

6.4 Validity threats

This section identifies the validity threats detected in the previously performed scenarios test and validation, based on [103].

Construct validity concerns about the theory and the relation between the expected results and the obtained results. In this case, there is a validity threat related with the experimenter expectations. Once that it was the same person developing the study and doing the experimentation, either consciously or inconsciously, can adjust the results to its expectations.

Internal validity threats are concerned with the impacts that the tests can have on the outcomes. In this context, we face a validity threat related to the uncertainty of the different scenarios causing a significant difference in the crowding avoidance. As it was possible to observe in crowding tests, only the first and second scenarios presented differences in the outcomes between the algorithm that considered crowding and the one that did not consider.

External validity considers about the generalization of the results obtained from the tests. For this validity, we have a threat related to the generalization of the conclusions taken from the five developed scenarios. Probably, it would be better to have a more significant number of scenarios that explored more combinations of the criteria and the map more effectively. Another threat is the fact of using the elevation of the points as if it were the crowding. This limits the results because even if the paths are different to go from one point to another, the height of the paths will not vary that much. On the other hand, crowding around a street can vary a lot to another street right next door. This might then have caused the limitation in the crowding results that, as we could observe, in two scenarios, considering or not considering crowding had the same route solution output.

6.5 Summary

Analyzing the results, we can conclude that the first four scenarios respected the *departure date*, the *origin* and *destination*, the *budget*, the *selected categories* and the *selected POIs*. Regarding the *visitation time*, three out of four scenarios were capable of complying with the users' available time. Nevertheless, the scenario that failed (scenario three) surpassed the users' available time in just twelve minutes, which represents a tiny margin. The check weather also proved to work well, by not changing the result when was *'false'* and recommending only POIs feasible with rain conditions when was *'true'*.

About the effort level, it was not an obstacle to find routes, whether the level was *'vigorous'* or *'moderate'*. However, the a limitation comes with the *'easy'* level, because all the scenarios would fail with such a low level. This is related to the proper activity of walking because considering that the *Graphhopper* assumes that a tourist walks at a speed of 3.0*mph*, it is always, at least, a *'moderate'* activity (consult Table 4.5).

In terms of the system functionalities, it was possible to show some capabilities of the system in dealing with the user preferences and constraints. We could demonstrate that the algorithm is capable of recommending a route when the user does not select POIs but selects the preferred categories. Every time there is space, budget (the budget can be zero) and time for recommending more POIs, the algorithm selects the most sustainable ones belonging to the preferred categories. Also, when a user selects the maximum number of POIs it is only returned the best route between the origin, the POIs and the destination.

In the particular case of the scenario five, we could also demonstrate that the algorithm can deal with the conjugation of the different preferences and constraints to the point of informing the user that it can be impossible to make a recommendation.

Chapter 7

Conclusion

Contents

7.1 Conclusions 91
7.2 Future work and limitations 92

This chapter summarizes all the work developed in this research. In Section 7.1, it is stated the conclusions that are taken from the developed work. Section 7.2 identifies limitations of the work and elaborates research opportunities for the given route generator algorithm systematized.

[This page has been intentionally left blank]

Chapter 7
Conclusion

7.1 Conclusions

The research question of this dissertation was if it was possible to optimize pedestrian route recommendations, based on a multi-criteria approach, that promotes sustainability and avoids overcrowding situations. Regarding this question, we concluded that it is possible to optimize the recommendations by recommending personalized routes that comprise different preferences and constraints.

To reach the answer of the research question it was necessary to accomplish the previously defined objectives. Looking for the developed work during the dissertation, we conclude that the communication between three of the components of the project was improved, namely between the *ROUTE*, the *HEAT* and the mobile application (*APP*). The microservices architecture of the project was modeled, and the communication between them was defined along with the interfaces that state what each interface should provide to be consumed. The focus was mainly on the three components evidenced before, where, with the help of UML diagrams, it was possible to outline the requirements, functionalities and participants on the connection between them. The main development was done with the proper software tools where the messages exchanged between the parts, in JSON, was established and guaranteed that carried all the important data.

Regarding the recommendations, the multi-criteria approach was dependent on several criteria that we proposed to include and taking into account the recommendation of the pedestrian routes. To do this, we started by justifying the utilization of some of the criteria: sustainability levels, crowding levels and weather conditions. Also, we researched the theory behind tourism trip problems to know where our work is positioned amongst others and what we could obtain as knowledge from the other problem solutions. That way, we based our solution in routes and graphs. The next step comprised the implementation of the solution. So, the architecture and the algorithms of the system developed with the aim of providing a solution to our problem were described.

Finally, it was possible to validate the work. The elaboration of different scenarios that show the way we envision how a tourist would use the mobile app allowed to show the features of the developed system. It was possible to understand that the user preferences (origin, destination, available time, level of effort, budget, selected POIs, selected categories) and constraints (POIs opening and closing hours, sustainability, crowding, weather conditions) are respected. Especially regarding the crowding and sustainability, which have a major focus for the project, it was possible to show that the sustainability level of the suggested POIs is maximized and the highest levels of crowding are avoided.

7.2 Future work and limitations

This section identifies limitations of the work and elaborates research opportunities for the given route generator algorithm systematized in the next three topics: 1) Evolution of the algorithm approach and future tests; 2) Evolution of the functionalities offered by the system; 3) Make the system more adaptive and intelligent.

1) Evolution of the algorithm approach and future tests

Performance comparison of algorithms

Compare the performance of the currently used algorithm with the performance of other known algorithms in terms of response time, suggested routes, etc.

Maximum number of POIs to suggest in each route

The maximum number of POIs suitable to suggest in each recommendation was established as five ($Max_{pois} = 5$) because the several iterations done along the algorithm would consume an exponential amount of time with the increase of that number. So, an improvement to the quality of the algorithm in terms of the consumed time must be done to increase the number of POIs that can be included in a solution. This is particularly important because, right now, the user may still have free time for more visits, but the maximum number of POIs will not let him do it. Moreover, the use of a "Greedy algorithm", commonly used in combinatorial optimization and in TSP, may accomplish a substantially better (not optimal) solution but with a reasonable amount of time [9].

Explore *ELECTRE* methods

Explore a different approach to handle the aggregation of the multiple criteria used in decisions, which can be by investigating *ELECTRE* methods [38].

Multiple Time Windows

The assumption of POIs having a fixed periodic time window corresponds to a problem simplification. So, the modelling of TTDP shall consider multiple time windows for a closer approximation to reality, once that POIs typically operate at specific days weekly, probably with variable opening and closing hours.

Crowding consideration

Right now the crowding is only being considered to choose the paths between the locations. However, each POI can also have a crowding level associated that may vary along the day and should be taken into account when choosing the POIs and their order of visitation. The consideration of crowding should be done by obtaining the current values of crowding and by taking into account the forecasting estimation of those values, either for POIs and streets. Furthermore, there must be a change on the crowding values used by the algorithm because, at the moment, they are based on contour lines but should be real estimations of how many people are in each place.

2) Evolution of the functionalities offered by the system

Evaluate the system

Create a questionnaire to ask real users their opinion about the system functionalities and produce a quantitative and qualitative analysis.

Route recommendations for groups

Usually, TRSs are developed to make suggestions to individual users, however tourist activities are usually done in a group, and the preferences and goals of the different group members can be very different and conflict. In the future, it can be good to implement an approach that allows recommending personalized routes for a heterogeneous group of tourists.

Plan holiday routes

At the moment it is only possible to create routes that last a few hours. Develop the functionality of creating and manage personalized routes for more than the actual day (e.g. for the entire holiday period) would represent a significant improvement. This could also turn to an excellent opportunity to integrate accommodation, where for the end of each route that lasts more than one day, the end location could be chosen from a set of accommodation locations.

Complete connection with other STC research project components

Although it is already known that the *ROUTE* service is capable of communicating with the *APP*, the *POINT* and the *HEAT* service, these components are not fully connected yet to collaborate automatically with each other. Practically, to make their integration, the connections between the services must be implemented (web client to consume other services, webserver to serve the app). Then, this component must be deployed in a docker container on virtual machines. Also, the periodic connection to the databases is not implemented yet. For this, a web client should be implemented, which would periodically seek to update the crowding and POI data. The *POINT* and *HEAT* web services would serve the updates.

Dynamic rescheduling

Even if it is a single day route or a multiple-day route planning, there is always a chance that users deviate from the plan or that an unexpected event occurs. These can make the actual selection of POIs or route invalid, so the system should be improved so that it presents a new route schedule in real-time. In this process, the starting and ending location, as well as the POIs, already visited, must be excluded.

Public transports

At the moment, all the recommended solutions are pedestrian routes. Nevertheless, make available the possibility of planning tours making use of several modes of public transport (bus, metro, train, tram, etc.) and providing information about stations, connection points and times is relevant. That is called multi-modal transportation and is important because when travelling between POIs, especially over large distances, these alternatives to walking can save substantial amounts of time that could be spent on POI visits. It can also avoid difficulties in the use of public transport when users are from foreign countries or facilitate the recommendation of

routes that generally have a higher level of physical effort associated.

Consider break times
At this point, the recommended routes only comprehend the visits to POIs and the time of travelling between them. However, it is not very realistic to expect that a tourist makes the visits all in a row and, as soon as they finish one visit they set out on their way to another. To turn the solutions more realistic and more likely to be followed by the tourists, it is essential to consider break times when they can have a meal or relax.

3) Make the system more adaptive and intelligent

Avoid repeated POIs recommendations
Record the places visited by the user, so that if the user uses the application several times during consecutive days (or even more than once on the same day), the places that he has already visited are not suggested. It is necessary to find a mechanism that understands if the user does not want repetitions to occur because the user may even wish to repeat the visit to certain places and not others.

Improve the average visitation time of POIs and build profiles of its visitors
Right now, for each POI, there is a fixed visit time that was estimated based on tourism websites data. This numbers can later be improved using statistical analysis, by monitoring the visiting times of the POIs by tourists. It also permits to learn and build a model for distributing the visiting time of users in the POIs and classifying the profiles of the people who visit them.

Calculate the level of acceptance of the suggested routes
Comparison of the recommended routes with the routes taken by users to understand what the tourists' adherence is to what is suggested. In order to make this comparison, the system must be fully implemented to obtain the actual data on the use of the application.

Bibliography

[1] G. Adomavicius and A. Tuzhilin. *Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions*. June 2005. DOI: 10.1109/TKDE.2005.99.

[2] B. E. Ainsworth, W. L. Haskell, S. D. Herrmann, N. Meckes, D. R. Bassett, C. Tudor-Locke, J. L. Greer, J. Vezina, M. C. Whitt-Glover, and A. S. Leon. *Compendium of Physical Activities: A second update of codes and MET values*. Aug. 2011. DOI: 10.1249/MSS.0b013e31821ece12.

[3] H. Alrasheed, A. Alzeer, A. Alhowimel, N. Shameri, and A. Althyabi. "A Multi-Level Tourism Destination Recommender System." In: *Procedia Computer Science* 170 (Jan. 2020), pp. 333–340. ISSN: 18770509. DOI: 10.1016/j.procs.2020.03.047.

[4] M. Amoretti, L. Belli, and F. Zanichelli. "UTravel: Smart Mobility with a Novel User Profiling and Recommendation Approach." In: *Pervasive and Mobile Computing* 38 (July 2017), pp. 474–489. ISSN: 15741192. DOI: 10.1016/j.pmcj.2016.08.008.

[5] P Anderssen and R. Colberg. *Multivariate analysis in travel research: a tool for travel package design and market segmentation*. University of Washington, 1973.

[6] M. Andersson, S. Ntalampiras, T. Ganchev, J. Rydell, J. Ahlberg, and N. Fakotakis. "Fusion of acoustic and optical sensor data for automatic fight detection in urban environments." In: *13th Conference on Information Fusion, Fusion 2010*. IEEE Computer Society, 2010. ISBN: 9780982443811. DOI: 10.1109/icif.2010.5712105.

[7] B. Archer, C. Cooper, and L. Ruhanen. "The positive and negative impacts of tourism." In: *Global tourism* 3 (2005), pp. 79–102.

[8] A. de Avila. "Smart destinations: XXI century tourism." In: *ENTER2015 conference on information and communication technologies in tourism, Lugano, Switzerland*. 2015, pp. 4–6.

[9] J. Bang-Jensen, G. Gutin, and A. Yeo. "When the greedy algorithm fails." In: *Discrete Optimization* 1.2 (2004), pp. 121 –127. ISSN: 1572-5286. DOI: https://doi.org/10.1016/j.disopt.2004.03.007.

[10] A.-L. Barabási and M. Pósfai. *Network Science*. Cambridge University Press, 2016. ISBN: 9781107076266 1107076269.

[11] R. Baraglia, C. Frattari, C. I. Muntean, F. M. Nardini, and F. Silvestri. "A trajectory-based recommender system for tourism." In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Vol. 7669 LNCS. Springer, Berlin, Heidelberg, 2012, pp. 196–205. ISBN: 9783642352355. DOI: 10.1007/978-3-642-35236-2_20.

[12] L. Baresi, M. Garriga, and A. De Renzis. "Microservices identification through interface analysis." In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Vol. 10465 LNCS. Springer Verlag, 2017, pp. 19–33. ISBN: 9783319672618. DOI: 10.1007/978-3-319-67262-5_2.

[13] M. Batet, A. Moreno, D. Sánchez, D. Isern, and A. Valls. "Turist@: Agent-based personalised recommendation of tourist activities." In: *Expert Systems with Applications* 39.8 (June 2012), pp. 7319–7329. ISSN: 09574174. DOI: 10.1016/j.eswa.2012.01.086.

[14] S. Becken. "The importance of climate and weather for tourism: literature review." In: *Land Environment and People (LEaP) background paper, Lincoln University* (2010).

[15] S. Boccaletti, V. Latora, Y. Moreno, M. Chavez, and D. U. Hwang. *Complex networks: Structure and dynamics*. Feb. 2006. DOI: 10.1016/j.physrep.2005.10.009.

[16] J. Borràs, J. de la Flor, Y. Pérez, A. Moreno, A. Valls, D. Isern, A. Orellana, A. Russo, and S. Anton-Clavé. "SigTur/E-Destination: A System for the Management of Complex Tourist Regions." In: *Information and Communication Technologies in Tourism 2011*. Springer Vienna, 2011, pp. 39–50. DOI: 10.1007/978-3-7091-0503-0_4.

[17] J. Borràs, A. Moreno, and A. Valls. "Intelligent tourism recommender systems: A survey." In: *Expert Systems with Applications* 41.16 (2014), pp. 7370–7389. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2014.06.007.

[18] P. Bourque, R. E. Fairley, et al. *Guide to the Software Engineering Body of Knowledge SWEBOK ® A Project of the IEEE Computer Society*. Ed. by P. É. de technologie supérieure) Bourque, R. E. S. Fairley, and S. E. Associates). 3.0. IEEE Computer Society, 2014, p. 335. ISBN: 10: 0-7695-5166-1.

[19] G. H. Brundtland, M Khalid, S Agnelli, S Al-Athel, and B. Chidzero. "Our common future." In: *New York* 8 (1987).

[20] D. Buhalis and R. Law. "Progress in information technology and tourism management: 20 years on and 10 years after the Internet-The state of eTourism research." In: *Tourism Management* 29.4 (Aug. 2008), pp. 609–623. ISSN: 02615177. DOI: 10.1016/j.tourman.2008.01.005.

[21] M. C. Cabral. *ESTRATÉGIA TURISMO 2027 LIDERAR O TURISMO DO FUTURO*. 2018.

[22] L. Cao, J. Tao, and B. Chen. "Implementation of personalized scenic spots route recommendation system." In: *13th International Conference on Computer Science and Education, ICCSE 2018*. Institute of Electrical and Electronics Engineers Inc., Sept. 2018, pp. 533–538. ISBN: 9781538654958. DOI: 10.1109/ICCSE.2018.8468845.

[23] B. Cartaxo, G. Pinto, and S. Soares. "The Role of Rapid Reviews in Supporting Decision-Making in Software Engineering Practice." In: *Proceedings of the 22nd International Conference on Evaluation and Assessment in Software Engineering 2018*. EASE'18. Christchurch, New Zealand: Association for Computing Machinery, 2018, 24–34. ISBN: 9781450364034. DOI: 10.1145/3210459.3210462.

[24] B. Cartaxo, G. Pinto, and S. Soares. "Rapid Reviews in Software Engineering." In: *Contemporary Empirical Methods in Software Engineering*. Ed. by M. Felderer and G. H. Travassos. Cham: Springer International Publishing, 2020, pp. 357–384. ISBN: 978-3-030-32489-6. DOI: 10.1007/978-3-030-32489-6_13.

[25] C. A. Charalambides. *Enumerative Combinatorics*. Discrete Mathematics and Its Applications. CRC Press, 2018. ISBN: 9781482296310.

[26] M. Ciavotta, M. Alge, S. Menato, D. Rovere, and P. Pedrazzoli. "A Microservice-based Middleware for the Digital Factory." In: *Procedia Manufacturing* 11 (Jan. 2017), pp. 931–938. ISSN: 23519789. DOI: 10.1016/j.promfg.2017.07.197.

[27] J. C. Clímaco and M. M. Pascoal. *Multicriteria path and tree problems: Discussion on exact algorithms and applications*. Jan. 2012. DOI: 10.1111/j.1475-3995.2011.00815.x.

[28] A. M. Cohen, H. Cohen, D. Eisenbud, M. F. Singer, and B. S. Volume. *Algorithms and Computation in Mathematics* •. ISBN: 9783540727798.

[29] C. Colomb and J. Novy. *Protest and resistance in the tourist city*. Routledge, 2017. ISBN: 9781138342248.

[30] C. R. De Freitas. *Tourism climatology: Evaluating environmental information for decision making and business planning in the recreation and tourism sector*. Sept. 2003. DOI: 10.1007/s00484-003-0177-z.

[31] A Dichter and G Guevara Manzo. "Coping with success: Managing overcrowding in tourism destinations." In: *World Travel and Tourism Council, London* (2017).

[32] D. R. Domínguez, R. P. Díaz Redondo, A. F. Vilas, and M. B. Khalifa. "Sensing the city with Instagram: Clustering geolocated data for outlier detection." In: *Expert Systems with Applications* 78 (July 2017), pp. 319–333. ISSN: 09574174. DOI: 10.1016/j.eswa.2017.02.018.

[33] *Estatísticas do Turismo : 2018*. Lisboa, 2019.

[34] L. Euler. "Solutio problematis ad geometriam situs pertinentis." In: *Commentarii academiae scientiarum Petropolitanae* (1741), pp. 128–140.

[35] A. Expósito, S. Mancini, J. Brito, and J. A. Moreno. "Solving a fuzzy tourist trip design problem with clustered points of interest." In: *Studies in Fuzziness and Soft Computing*. Vol. 377. Springer Verlag, 2019, pp. 31–47. DOI: 10.1007/978-3-030-10463-4_2.

[36] J. C. Farrés. "Barcelona noise monitoring network." In: *Proceedings of the EuroNoise*. 2015.

[37] R. Fielding and J. Reschke. *Hypertext transfer protocol (http/1.1): Semantics and content*. 2014.

[38] J. R. Figueira, V. Mousseau, and B. Roy. "ELECTRE Methods." In: *Multiple Criteria Decision Analysis: State of the Art Surveys*. Ed. by S. Greco, M. Ehrgott, and J. R. Figueira. New York, NY: Springer New York, 2016, pp. 155–185. ISBN: 978-1-4939-3094-4. DOI: 10.1007/978-1-4939-3094-4_5.

[39] I. Garcia, L. Sebastia, and E. Onaindia. "On the design of individual and group recommender systems for tourism." In: *Expert Systems with Applications* 38.6 (June 2011), pp. 7683–7692. ISSN: 09574174. DOI: 10.1016/j.eswa.2010.12.143.

[40] L. M. Garcia, S. Aciar, R. Mendoza, and J. J. Puello. "Smart tourism platform based on microservice architecture and recommender services." In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Vol. 10995 LNCS. Springer Verlag, 2018, pp. 167–180. ISBN: 9783319971629. DOI: 10.1007/978-3-319-97163-6_14.

[41] D. Gavalas and M. Kenteris. "A web-based pervasive recommendation system for mobile tourist guides." In: *Personal and Ubiquitous Computing* 15.7 (Oct. 2011), pp. 759–770. ISSN: 16174909. DOI: 10.1007/s00779-011-0389-x.

[42] D. Gavalas, C. Konstantopoulos, K. Mastakas, and G. Pantziou. "A survey on algorithmic approaches for solving tourist trip design problems." In: *Journal of Heuristics* 20.3 (June 2014), pp. 291–328. ISSN: 1572-9397. DOI: 10.1007/s10732-014-9242-5.

[43] D. Gavalas, C. Konstantopoulos, K. Mastakas, and G. Pantziou. *Mobile recommender systems in tourism*. Mar. 2014. DOI: 10.1016/j.jnca.2013.04.006.

[44] S. Glass, G. B. Dwyer, A. C. of Sports Medicine, et al. *ACSM'S metabolic calculations handbook*. Lippincott Williams & Wilkins, 2007.

[45] P. Glavič and R. Lukman. "Review of sustainability terms and their definitions." In: *Journal of Cleaner Production* 15.18 (Dec. 2007), pp. 1875–1885. ISSN: 09596526. DOI: 10.1016/j.jclepro.2006.12.006.

[46] M. Gómez-Martín, X. Armesto-López, and E. Martínez-Ibarra. "Tourists, Weather and Climate. Official Tourism Promotion Websites as a Source of Information." In: *Atmosphere* 8.12 (Dec. 2017), p. 255. ISSN: 2073-4433. DOI: 10.3390/atmos8120255.

[47] U. Gretzel, M. Sigala, Z. Xiang, and C. Koo. "Smart tourism: foundations and developments." In: *Electronic Markets* 25.3 (Sept. 2015), pp. 179–188. ISSN: 14228890. DOI: 10.1007/s12525-015-0196-8.

[48] A. Gunawan, H. C. Lau, and P. Vansteenwegen. *Orienteering Problem: A survey of recent variants, solution approaches and applications*. Dec. 2016. DOI: 10.1016/j.ejor.2016.04.059.

[49] F. Y. Hanai, E. Luiz, and G. Espíndola. "INDICATORS OF SUSTAINABILITY: CONCEPTS, TYPOLOGIES AND THEIR APPLICATION IN LOCAL TOURISM DEVELOPMENT." In: *Environmental & Social Management Journal/Revista de Gestão Social e Ambiental* 5.3 (2011), pp. 135–149. ISSN: 1981-982X.

[50] W. L. Haskell, I. M. Lee, R. R. Pate, K. E. Powell, S. N. Blair, B. A. Franklin, C. A. Macera, G. W. Heath, P. D. Thompson, and A. Bauman. "Physical Activity and Public Health." In: *Circulation* 116.9 (Aug. 2007), pp. 1081–1093. ISSN: 0009-7322. DOI: 10.1161/CIRCULATIONAHA.107.185649.

[51] T. A. Heberlein and J. Vaske. "Crowding and visitor conflict on the Bois Brule River." In: (1977).

[52] C. I. Ho, M. H. Lin, and H. M. Chen. "Web users' behavioural patterns of tourism information search: From online to offline." In: *Tourism Management* 33.6 (Dec. 2012), pp. 1468–1482. ISSN: 02615177. DOI: 10.1016/j.tourman.2012.01.016.

[53] W. Höpken, H. Ravensburg-Weingarten, and M. Fuchs. "Context-based adaptation of ubiquitous web applications in tourism." In: *Information and Communication Technologies in Tourism* 12 (2008), pp. 175–195. DOI: 10.3727/109830510X12887971002783.

[54] A. Hübner and S. Gössling. "Tourist perceptions of extreme weather events in Martinique." In: *Journal of Destination Marketing and Management* 1.1-2 (Nov. 2012), pp. 47–55. ISSN: 2212571X. DOI: 10.1016/j.jdmm.2012.09.003.

[55] K. Ilavarasi and K. S. Joseph. "Variants of travelling salesman problem: A survey." In: *2014 International Conference on Information Communication and Embedded Systems, ICICES 2014*. Institute of Electrical and Electronics Engineers Inc., Feb. 2015. ISBN: 9781479938346. DOI: 10.1109/ICICES.2014.7033850.

[56] *International Tourism Highlights, 2019 Edition*. Aug. 2019. DOI: 10.18111/9789284421152.

[57] J. L. Jorro-Aragoneses, M. B. Diaz Agudo, and J. A. Recio Garcia. "Madrid live: A context-aware recomendar system of leisure plans." In: *Proceedings - International Conference on Tools with Artificial Intelligence, ICTAI*. Vol. 2017-Novem. IEEE Computer Society, June 2018, pp. 796–801. ISBN: 9781538638767. DOI: 10.1109/ICTAI.2017.00125.

[58] K. Kabassi. *Personalizing recommendations for tourists*. Feb. 2010. DOI: 10.1016/j.tele.2009.05.003.

[59] R. Kramer, M. Modsching, and K. ten Hagen. "A City Guide Agent Creating and Adapting Individual Sightseeing Tours Based on Field Trial Results." In: *International Journal of Computational Intelligence Research* 2.2 (2006), pp. 191–206.

[60] O. d. T. de Lisboa. "Inquérito de Satisfação e Imagem." In: *Região de Lisboa* (2011).

[61] H. L. Liu, J. H. Li, and J. Peng. "A novel recommendation system for the personalized smart tourism route: Design and implementation." In: *Proceedings of 2015 IEEE 14th International Conference on Cognitive Informatics and Cognitive Computing, ICCI*CC 2015*. Institute of Electrical and Electronics Engineers Inc., Sept. 2015, pp. 291–296. ISBN: 9781467372893. DOI: 10.1109/ICCI-CC.2015.7259400.

[62] K. Makedos and N. Tryfona. "PLATIS: A Personalized Location-Aware Tourist Information System." In: *Proceedings of the Second ACM SIGSPATIAL International Workshop on Mobile Geographic Information Systems*. MobiGIS '13. New York, NY, USA: Association for Computing Machinery, 2013, pp. 59–66. ISBN: 9781450325318. DOI: 10.1145/2534190.2534195.

[63] N. Manouselis and C. Costopoulou. "Analysis and classification of multi-criteria recommender systems." In: *World Wide Web* 10.4 (Dec. 2007), pp. 415–441. ISSN: 1386145X. DOI: 10.1007/s11280-007-0019-8.

[64] M. Masse. *REST API Design Rulebook: Designing Consistent RESTful Web Service Interfaces*. "O'Reilly Media, Inc.", 2011.

[65] A. Matzarakis. "Weather-and climate-related information for tourism." In: *Tourism and Hospitality, Planning and Development* 3.2 (2006), pp. 99–115. ISSN: 1479053X. DOI: 10.1080/14790530600938279.

[66] B. McKercher, N. Shoval, E. Park, and A. Kahani. "The [Limited] Impact of Weather on Tourist Behavior in an Urban Destination." In: *Journal of Travel Research* 54.4 (July 2015), pp. 442–455. ISSN: 0047-2875. DOI: 10.1177/0047287514522880.

[67] E. McLoughlin, J. Hanrahan, A.-m. Duddy, and S. Duffy. "European Tourism Indicator System for Sustainable Destination Management in County Donegal Ireland." In: *European Journal of Tourism Research* 20 (2018).

[68] F. Mehmood, S. Ahmad, and D. Kim. "Design and Development of a Real-Time Optimal Route Recommendation System Using Big Data for Tourists in Jeju Island." In: *Electronics* 8.5 (May 2019), p. 506. ISSN: 2079-9292. DOI: 10.3390/electronics8050506.

[69] S. Migliorini, D. Carra, and A. Belussi. "Adaptive Trip Recommendation System: Balancing Travelers among POIs with MapReduce." In: *Proceedings - 2018 IEEE International Congress on Big Data, BigData Congress 2018 - Part of the 2018 IEEE World Congress on Services*. Institute of Electrical and Electronics Engineers Inc., Sept. 2018, pp. 255–259. ISBN: 9781538672327. DOI: 10.1109/BigDataCongress.2018.00045.

[70] K. Naik and P. Tripathy. *Software testing and quality assurance: theory and practice*. John Wiley & Sons, 2011. ISBN: 978-1-118-21163-2.

[71] M. Nazim. "Information searching behavior in the Internet age: A users' study of Aligarh Muslim University." In: *International Information & Library Review* 40.1 (Mar. 2008), pp. 73–81. ISSN: 1057-2317. DOI: 10.1080/10572317.2008.10762764.

[72] J. M. Noguera, M. J. Barranco, R. J. Segura, and L. Martínez. "A mobile 3D-GIS hybrid recommender system for tourism." In: *Information Sciences* 215 (Dec. 2012), pp. 37–52. ISSN: 00200255. DOI: 10.1016/j.ins.2012.05.010.

[73] L. E. Nugroho, R. Y. Saputra, V. M. Putri, and Y. Efindo. "A Context-Aware Adaptive Tourist Recommendation System." In: *ACM International Conference Proceeding Series*. New York, NY, USA: Association for Computing Machinery, Dec. 2019, pp. 453–457. ISBN: 9781450371797. DOI: 10.1145/3366030.3366088.

[74] P. Office. *The European Tourism Indicator System ETIS toolkit for sustainable destination management*. Mar. 2016. DOI: 10.2873/982144.

[75] S. Park, M. Bourqui, and E. Frias-Martinez. "MobInsight: Understanding Urban Mobility with Crowd-Powered Neighborhood Characterizations." In: *IEEE International Conference on Data Mining Workshops, ICDMW*. Vol. 0. IEEE Computer Society, July 2016, pp. 1312–1315. ISBN: 9781509054725. DOI: 10.1109/ICDMW.2016.0192.

[76] R. R. Pate, C. A. Macera, M. Pratt, G. W. Heath, S. N. Blair, C. Bouchard, W. L. Haskell, A. C. King, D. Buchner, W. Ettinger, A. Kriska, A. S. Leon, B. H. Marcus, J. Morris, R. S. Paffenbarger, K. Patrick, M. L. Pollock, J. M. Rippe, J. Sallis, and J. H. Wilmore. "Physical Activity and Public Health: A Recommendation From the Centers for Disease Control and Prevention and the American College of Sports Medicine." In: *JAMA: The Journal of the American Medical Association* 273.5 (Feb. 1995), pp. 402–407. ISSN: 15383598. DOI: 10.1001/jama.1995.03520290054029.

[77] P. Peeters, S. Gössling, J. Klijs, C. Milano, M. Novelli, C. Dijkmans, E. Eijgelaar, S. Hartman, J. Heslinga, R. Isaac, O. Mitas, S. Moretti, J. Nawijn, B. Papp, and A. Postma. "Research for TRAN Committee-Overtourism: impact and possible policy responses." In: *Research for TRAN Committee-Overtourism: impact and possible policy responses* October (2018), pp. 1–255. DOI: 10.2861/919195.

[78] A. R. H. Peixoto. "A graph-based approach for sustainable walking tour recommendations: the case of Lisbon overcrowding." Master's thesis. Iscte – Instituto Universitário de Lisboa, 2019. DOI: http://hdl.handle.net/10071/20234.

[79] J. A. Peterson and S. J. Tharrett. *ACSM's health/fitness facility standards and guidelines*. ERIC, 1997.

[80] T. de Portugal. *Relatório de Sustentabilidade | 2017*. 2018.

[81] F. Ricci. "Travel Recommender Systems." In: *IEEE Intelligent Systems* 17.6 (2002), pp. 55–57.

[82] F. Ricci. "Mobile Recommender Systems." In: *Information Technology & Tourism* 12.3 (May 2011), pp. 205–231. ISSN: 10983058. DOI: 10.3727/109830511x12978702284390.

[83] D. E. Schmidt and J. P. Keating. "Human crowding and personal control: An integration of the research." In: *Psychological Bulletin* 86.4 (July 1979), pp. 680–700. ISSN: 00332909. DOI: 10.1037/0033-2909.86.4.680.

[84] D. Scott and C. Lemieux. "Weather and climate information for tourism." In: *Procedia Environmental Sciences*. Vol. 1. 1. Elsevier B.V., 2010, pp. 146–183. DOI: 10.1016/j.proenv.2010.09.011.

[85] D Scott, S Gössling, and C. de Freitas. "Preferred climates for tourism: case studies from Canada, New Zealand and Sweden." In: *Climate Research* 45.1 (Nov. 2008), pp. 61–73. ISSN: 0936-577X. DOI: 10.3354/cr00774.

[86] B. Shelby and J. J. Vaske. "Comparative analysis of crowding in multiple locations: Results from fifteen years of research." In: *Leisure Sciences* 11.4 (Jan. 1989), pp. 269–291. ISSN: 15210588. DOI: 10.1080/01490408909512227.

[87] R. D. da Silva. "A tourism overcrowding sensor using multiple radio techniques detection." Master's thesis. Iscte – Instituto Universitário de Lisboa, 2019. DOI: http://hdl.handle.net/10071/20212.

[88] K. Smith. "The influence of weather and climate on recreation and tourism." In: *Weather* 48.12 (Dec. 1993), pp. 398–404. ISSN: 00431656. DOI: 10.1002/j.1477-8696.1993.tb05828.x.

[89] I Sommerville. *Software Engineering 9th ed.* 2011.

[90] J. Sorgalla. "AjiL: A Graphical Modeling Language for the Development of Microservice Architectures." In: *Extended Abstracts of the Microservices 2017 Conference*. 2017.

[91] W. Souffriau, P. Vansteenwegen, J. Vertommen, G. V. Berghe, and D. V. Oudheusden. "A PERSONALIZED TOURIST TRIP DESIGN ALGORITHM FOR MOBILE TOURIST GUIDES." In: *Applied Artificial Intelligence* 22.10 (2008), pp. 964–985. DOI: 10.1080/08839510802379626.

[92] R. Streeton, M. Cooke, and J. Campbell. *Researching the researchers: using a snowballing technique*. 2004. DOI: 10.7748/nr2004.07.12.1.35.c5929.

[93] G. A. Tanguay, J. Rajaonson, J.-F. Lefebvre, and P. Lanoie. "Measuring the sustainability of cities: An analysis of the use of local indicators." In: *Ecological Indicators* 10.2 (2010), pp. 407–418. DOI: https://doi.org/10.1016/j.ecolind.2009.07.013.

[94] Z. Tarapata. "Selected multicriteria shortest path problems: An analysis of complexity, models and adaptation of standard algorithms." In: *International Journal of Applied Mathematics and Computer Science* 17.2 (June 2007), pp. 269–287. ISSN: 1641876X. DOI: 10.2478/v10006-007-0023-2.

[95] K. Tervo. "The Operational and Regional Vulnerability of Winter Tourism to Climate Variability and Change: The Case of the Finnish Nature-Based Tourism Entrepreneurs." In: *Scandinavian Journal of Hospitality and Tourism* 8.4 (Dec. 2008), pp. 317–332. ISSN: 1502-2250. DOI: 10.1080/15022250802553696.

[96] P. Thiengburanathum, S. Cang, and H. Yu. "Overview of personalized Travel Recommendation Systems." In: *2016 22nd International Conference on Automation and Computing, ICAC 2016: Tackling the New Challenges in Automation and Computing*. Institute of Electrical and Electronics Engineers Inc., Oct. 2016, pp. 415–422. ISBN: 9781862181311. DOI: 10.1109/IConAC.2016.7604955.

[97] A. C. Tricco, J. Antony, W. Zarin, L. Strifler, M. Ghassemi, J. Ivory, L. Perrier, B. Hutton, D. Moher, and S. E. Straus. "A scoping review of rapid review methods." In: *BMC medicine* 13.1 (2015), p. 224. DOI: https://doi.org/10.1186/s12916-015-0465-6.

[98] A. Umanets, A. Ferreira, and N. Leite. "GuideMe – A Tourist Guide with a Recommender System and Social Interaction." In: *Procedia Technology* 17 (2014), pp. 407–414. ISSN: 22120173. DOI: 10.1016/j.protcy.2014.10.248.

[99] P. Vansteenwegen and W. Souffriau. "Trip planning functionalities: state of the art and future." In: *Information Technology & Tourism* 12.4 (2010), pp. 305–315. DOI: 10.1007/978-3-642-16985-4_46.

[100] P. Vansteenwegen and D. Van Oudheusden. "The Mobile Tourist Guide: An OR Opportunity." In: *OR Insight* 20.3 (July 2007), pp. 21–27. ISSN: 1759-0477. DOI: 10.1057/ori.2007.17.

[101] P. Vansteenwegen, W. Souffriau, and D. V. Oudheusden. *The orienteering problem: A survey*. Feb. 2011. DOI: 10.1016/j.ejor.2010.03.045.

[102] C. Wohlin. "Guidelines for snowballing in systematic literature studies and a replication in software engineering." In: *ACM International Conference Proceeding Series*. Association for Computing Machinery, 2014. ISBN: 9781450324762. DOI: 10.1145/2601248.2601268.

[103] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén. *Experimentation in software engineering*. Springer Science & Business Media, 2012, 104–116. DOI: 10.1007/978-3-642-29044-2.

[This page has been intentionally left blank]

Appendix A

OpenWeatherMap API Response

In this annex, it is represented an example of an *OpenWeatherMap* API response in mode *5 Day / 3 Hour Forecast"*. To clarify the meaning of each field consult the API documentation [1].

Listing A.1: Example of OpenWeatherMap API response

```
{
  "list": [
    {
      "dt": 1603119600,
      "main": {
        "temp": 18.5,
        "feels_like": 13.37,
        "temp_min": 18.5,
        "temp_max": 18.5,
        "humidity": 83,
        ..., //Omitted for brevity
      },
      "weather": [
        {
          "id": 500,
          "main": "Rain",
          "description": "light rain",
        }
      ],
      "clouds": {
        "all": 100
      },
      "wind": {
        "speed": 9.93,
        "deg": 178
      },
      "visibility": 10000,
```

[1] https://openweathermap.org/forecast5

```
      "pop": 0.71,
      "rain": {
        "3h": 0.93
      },
      "dt_txt": "2020-10-19 15:00:00"
    },
    ..., //Omitted for brevity
    "city": {
    "id": 6930126,
    "name": "Lisbon",
    "coord": {
      "lat": 38.71,
      "lon": -9.14
    },
    "country": "PT",
    "timezone": 3600,
    "sunrise": 1602830814,
    "sunset": 1602871010
  }
}
```

Appendix B.

Point of Interest JSON Representation

As it is possible to observe, each POI has a specific structure as it is exemplified in listing B.1. It is composed of a unique identifier, a name, a set of coordinates associated, an address, a price of visitation, a level of sustainability and a timetable.

Listing B.1: POI representation in JSON

```
"poi": {
        "name":"Martinho da Arcada",
        "pointID":36,
        "visitTime":60,
        "openHour":7,
        "closeHour":17,
        "price":35,
        "coordinates":{
            "latitude":38.708675,
            "longitude":-9.135841
        },
        "sustainability":50,
        "categoryID":4
      }
```

[This page has been intentionally left blank]

Appendix C.

Crowding Data

Listing C.1 is an example of the data calculated and stored by the *HEAT* to supply the *ROUTE* microservice. The *"timestamp"* identifies the moment in time (can be the actual moment or a moment in the future) when the *"crowding"* level associated is felt. The *"coordinates"* identify the areas on the map where that specific level of crowding is felt and on that specific moment in time.

Listing C.1: Crowding data send by *HEAT*

```
{
    "timestamp":"2020-10-15 16:04:44",
    "crowding":"7",
    "geometry":{
        "type":"LineString",
        "coordinates":[
            [-9.137159090909092,38.71541666666667],
            [-9.137083333333333,38.71546568627451],
            [-9.136875,38.71541666666667],
            [-9.137083333333333,38.715],
            [-9.137159090909092,38.71541666666667]
        ]
    },
       ... // Omitted for brevity
}
```

Image C.1 depicts the area of the *Santa Maria Maior* with lines joining points of equal altitude named contour lines.

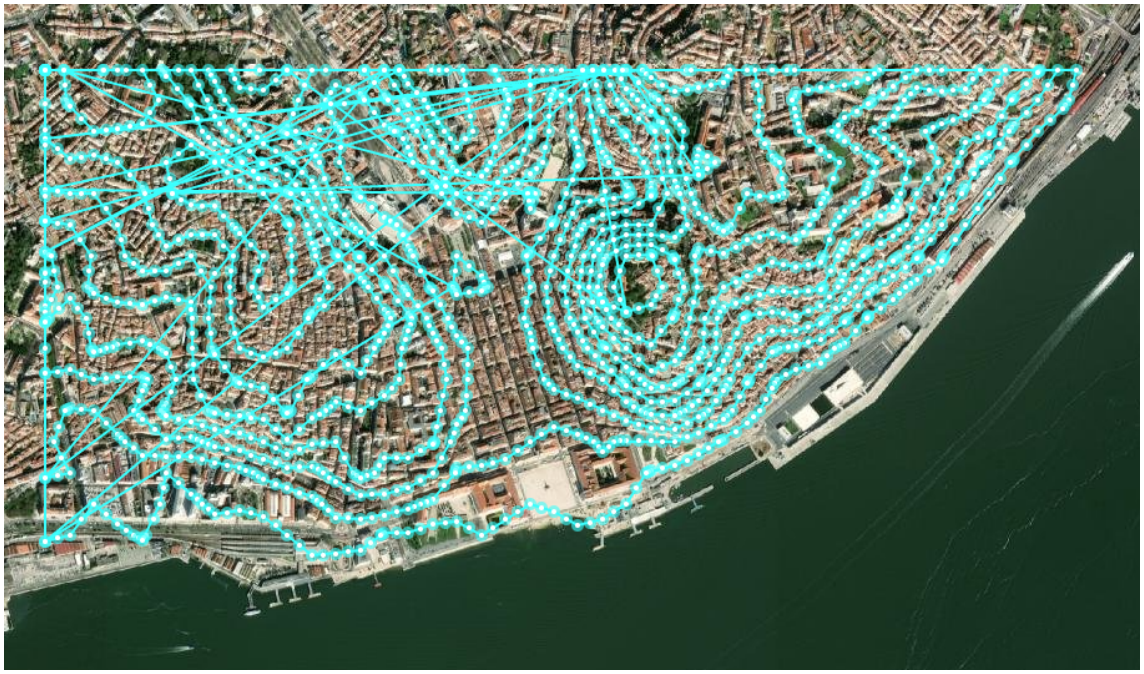

Figure C.1: Isocrowding lines of *Santa Maria Maior* parish council

Appendix D.

PointOfInterest Data

Table D.1 is a representation of the data present on the database where the POIs are stored. The last four columns, about the scenarios, were added to help with the validation.

Table D.1: Content of the POIs database

point_id	category_id	point_name	longitude	latitude	open_hour	close_hour	sustainability	price	visit_time	Scenario1	Scenario2	Scenario3	Scenario4
5	1	Pérola do Rossio	-9.13862817	38.71356062	10	19	60	0	20		X	X	
13	1	Nicolau Café	-9.1364516	38.7105554	9	20	80	10	20		X	X	
22	1	Nicola Café	-9.13967177	38.71321458	8	24	58	10	20		**S**	X	
28	1	Hospital de Bonecas	-9.13776636	38.71417458	10	19	90	0	30		X	X	
29	1	Manteigaria Silva	-9.13876112	38.71406446	9	20	94	10	25		X	**X**	
2	2	Capela Nossa Senhora da Saúde	-9.13569194	38.71586278	0	24	75	0	30	X	X		
12	2	Igreja de São Nicolau	-9.13669238	38.71093597	8	23	83	0	30	**X**	X		
19	2	Igreja da Nossa Senhora da Conceição Velha	-9.13423873	38.70896476	10	20	80	0	30	X	X		
24	2	Igreja da Madalena	-9.13476212	38.7101015	9	19	70	0	30	X	X		
25	2	Igreja de Santo António de Lisboa	-9.13401066	38.70999347	8	19	87	0	30	**X**	X		
27	2	Igreja de São Domingos	-9.13875324	38.71470839	8	19	89	0	30	**X**	X		
7	3	Miradouro do Recolhimento	-9.13171991	38.71267494	10	19	89	0	15		X	X	**X**
8	3	Miradouro de Santa Luzia	-9.13034395	38.71159619	0	24	90	0	15		X	X	**X**
15	3	Cais das Colunas	-9.1360868	38.70661524	0	24	65	0	15		X	X	X
18	3	Arco da Rua Augusta	-9.13682345	38.70839262	0	24	88	0	15		X	X	**X**
33	4	Casa Portuguesa do Pastel de Bacalhau	-9.137441	38.710253	10	22	70	6.5	40		**S**		
35	4	Café da Garagem	-9.132872	38.714858	15	23	80	10	50		X		
36	4	Martinho da Arcada	-9.135841	38.708675	7	17	50	35	60		**S**		
37	4	Aura Lisboa	-9.137792	38.707931	11	23	72	20	60		X		
38	4	A Brasileira	-9.142077	38.710671	8	22	85	10	35		**S**		
39	4	Solar do Duque	-9.141825	38.713401	12	23	55	14	60		X		
3	5	Tasquinha A Vaidosa	-9.13441912	38.71695138	9	22	45	7.5	45		X		
34	5	As Bifanas do Afonso	-9.135668	38.711555	7	19	81	5	40		**S**		
6	6	Casa dos Bicos - Fundação José Saramago	-9.1326586	38.709058	10	18	87	3	90		X	X	
16	6	Museu de Lisboa - Torreão Poente	-9.13726697	38.7067785	10	18	73	3	70		X	X	
17	6	Museu do Design e da Moda	-9.1369361	38.7090582	10	18	81	0	70		X	X	
21	6	O Mundo Fantástico da Sardinha Portuguesa	-9.1393499	38.7144703	10	20	69	0	20		X	X	
30	6	Museu Arqueológico do Carmo	-9.14063627	38.71190513	10	19	50	5	70		X	X	
31	6	Museu Nacional de Arte Contemporânea do Chiado	-9.14102261	38.70968009	10	18	97	4.5	70		X	**X**	
32	6	Museu Nacional do Desporto	-9.143269	38.71562	10	17	65	9	70		X	**S**	
1	7	Castelo de São Jorge	-9.1334762	38.7139092	9	18	84	10	70	**X**	X		
4	7	Palácio da Rosa	-9.1346331	38.714776	0	24	49	0	70	X	X		
9	7	Elevador da Bica	-9.14669643	38.70858166	7	21	81	3.7	20	**X**	X		
10	7	Elevador de Santa Justa	-9.1394235	38.71212908	8	21	72	5.15	20	X	X		
11	7	Elevador da Glória	-9.14335103	38.71539594	7	24	76	3.7	20	X	X		
26	7	Sé de Lisboa	-9.13340813	38.70980306	10	17	77	0	40	X	X		
14	8	Praça do Comércio	-9.1364489	38.707532	0	24	70	0	15		X		**X**
20	8	Praça do Rossio	-9.139444	38.713889	0	24	65	0	15		X		**X**
23	8	Praça da Figueira	-9.13786591	38.71372417	0	24	63	0	15		X		X

Appendix E.

Use Case Scenario Descriptions between the APP and the ROUTE

Next, there is a scenario description of the flow of events that happen since a user starts using the mobile app to request a route, until he receives the response recommended by the *ROUTE* system. The sequence of events is structured in natural language in the specific steps.

SET PREFERENCES AND CONSTRAINTS
PRECONDITIONS:

- -

POSTCONDITIONS:

- User has filled the preferences and constraints;

MAIN SCENARIO:

1. APP presents a form with origin and destination and departure time;
2. User fills the form and the APP uses the departure time to filter out POIs according to their schedule;
3. APP presents the list of filtered POIs and categories which the user then selects to visit in his trip;
4. APP defines the minimum values for the budget and arrival time constraints according to the selection the user made;
5. APP requests the user for the final constraints: budget, arrival time, effort and if he the weather to be checked;
6. **[REQUEST ROUTE RECOMMENDATION]**;

REQUEST ROUTE RECOMMENDATION

PRECONDITIONS:

- User has internet connection;
- User has filled the preferences and constraints;

POSTCONDITIONS:

- A route recommendation is presented to the user;

MAIN SCENARIO:

1. User asks the route recommendation;
2. APP sends the information to the ROUTE service;
3. *{User has no internet connection}*[**No Internet connection**];
4. [**PROVIDE ROUTE RECOMMENDATION**];
5. APP shows the route to the user on a map-based interface; which the user can accept or decline;
6. User accepts or declines the trip recommendation;

ALTERNATIVE SCENARIO [No Internet connection]

i. APP displays an error message informing that it requires Internet connection;
ii. Suggests the user to try again when Internet connection is available;
iii. Goes to step 1;

PROVIDE ROUTE RECOMMENDATION
PRECONDITIONS:

- *ROUTE* receives a route request;

POSTCONDITIONS:

- A route is created and sent to the mobile app;

MAIN SCENARIO:

1. *{The user selected the maximum number of POIs possible}* Go to step 11;
2. *{The user requests that the weather need to be checked}***[Wants weather to be checked]**;
3. ROUTE filters out POIs according to the user's categories of interest;
4. ROUTE checks the available time and budget;
5. *{No more time available left after visiting mandatory POIs}* Do not add new POIs to the route and goes to step 9;
6. *{Budget is totally spent on visiting the mandatory POIs}* **[Maximum budget is reached]**;
7. Route filters out POIs that are not open during the available time period;
8. *{No POIs left to suggest}* Do not add new POIs to the route and goes to step 9;
9. Order POIs by sustainability level;
10. Add as many POIs to the trip as the constraints allow, from the top of the list;
11. ROUTE service creates a route that complies the requirements and sends it to APP;
12. Sends route to the mobile app;

ALTERNATIVE SCENARIO [Wants weather to be checked]

i. ROUTE checks the probability of raining during the available duration for the trip;

ii. *{Rain probability > 50%}* ROUTE filters out POIs not suitable to visit when it is raining;

ALTERNATIVE SCENARIO [Maximum budget is reached]

i. Filter out non-free POIs;

ii. *{number of suitable POIs = 0}* Goes to step 9;

iii. *{number of suitable POIs > 0}* Goes to step 11;

[This page has been intentionally left blank]

Appendix F.

Set Preferences and Constraints Activity Diagram

The activity diagram represent in Figure F.1 shows the sequence of activities that occurs when a user is choosing its preferences and constraints for the route.

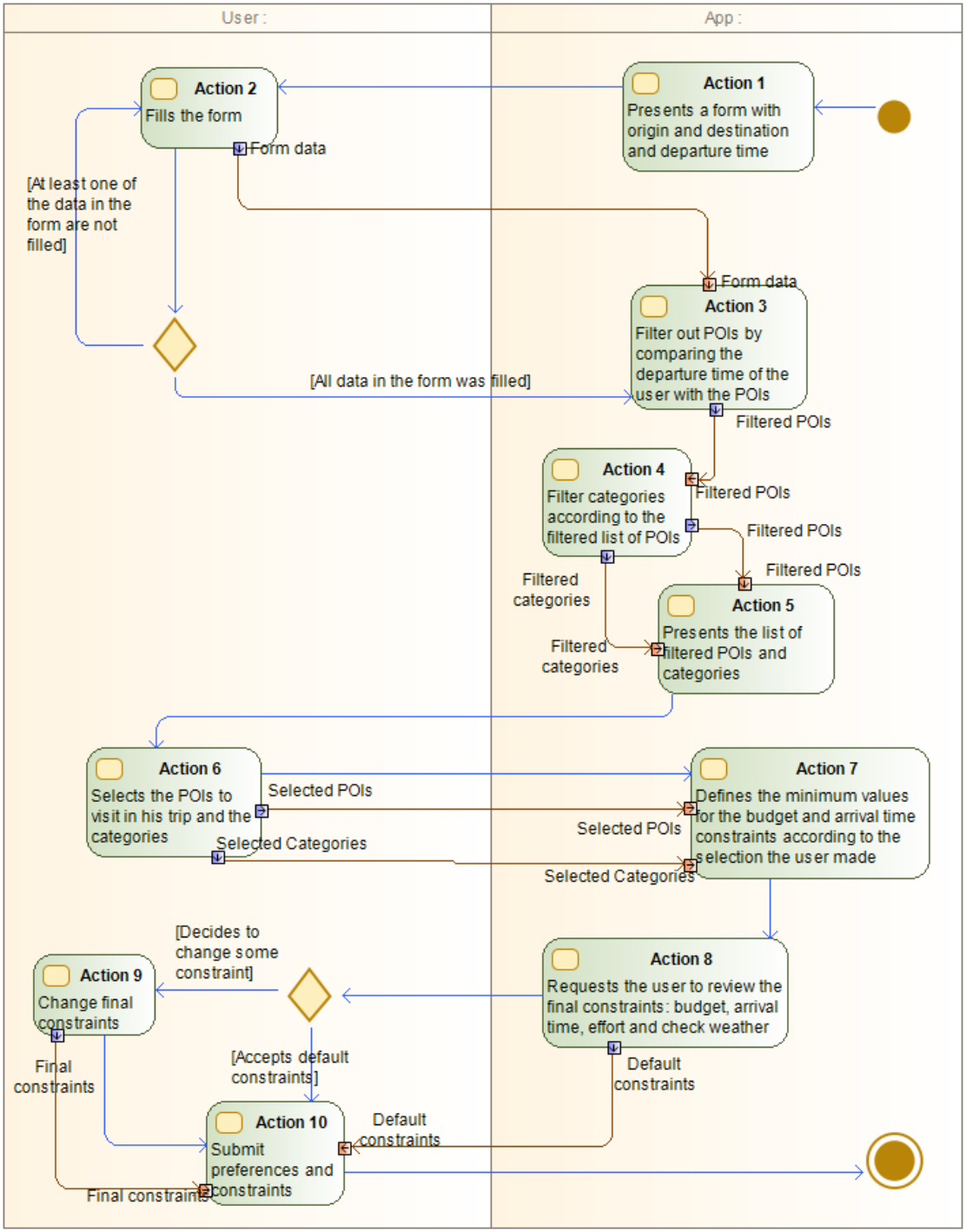

Figure F.1: Set preferences and constraints activity diagram

Appendix G

Crowding Graphics Analysis

This appendix contains graphics that compares two different algorithms and the levels of crowding that its recommended routes cross, as it is discussed in section 6.3.8.

Figure G.1 shows the comparison of the two algorithms for the scenario one explained in 6.3.2.

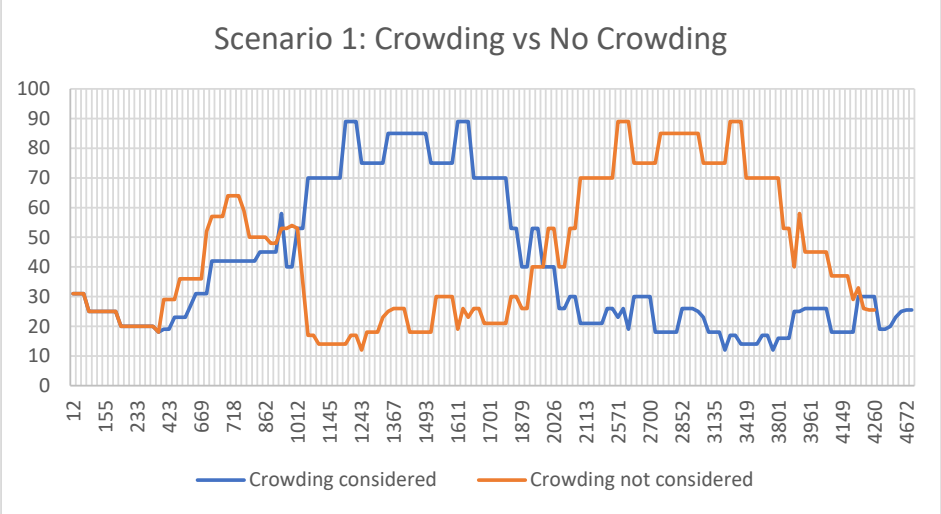

Figure G.1: Level of crowding comparison for scenario #1

Figure G.2 shows the comparison of the two algorithms for the scenario one explained in 6.3.3.

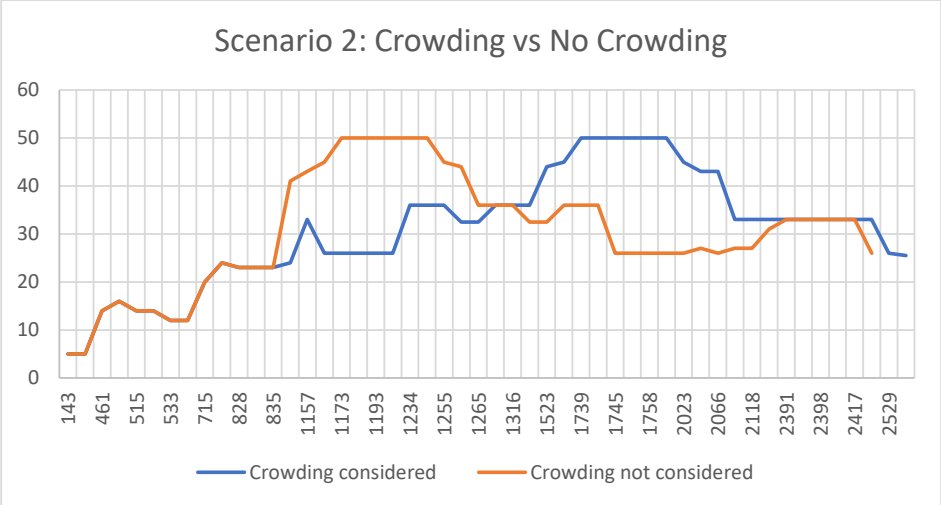

Figure G.2: Level of crowding comparison for scenario #2

Figure G.3 shows the comparison of the two algorithms for the scenario one explained in 6.3.4.

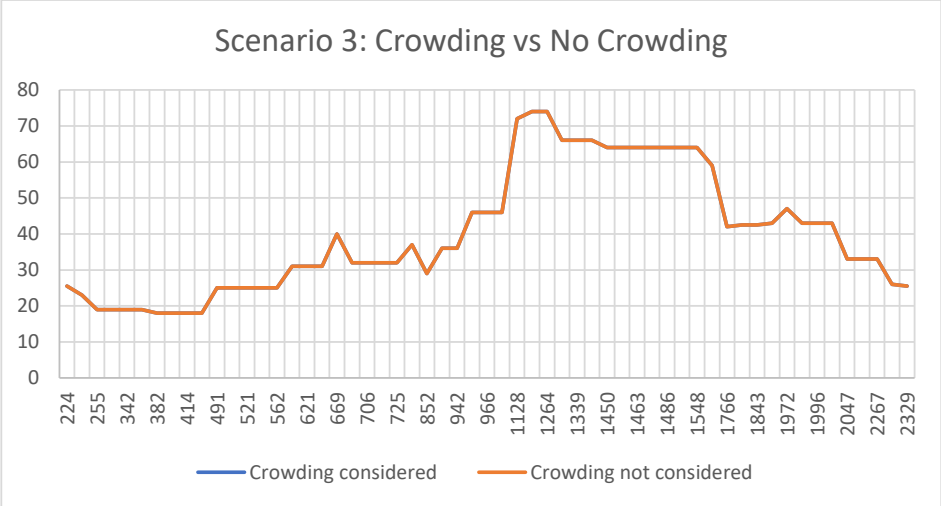

Figure G.3: Level of crowding comparison for scenario #3

Figure G.4 shows the comparison of the two algorithms for the scenario one explained in 6.3.5.

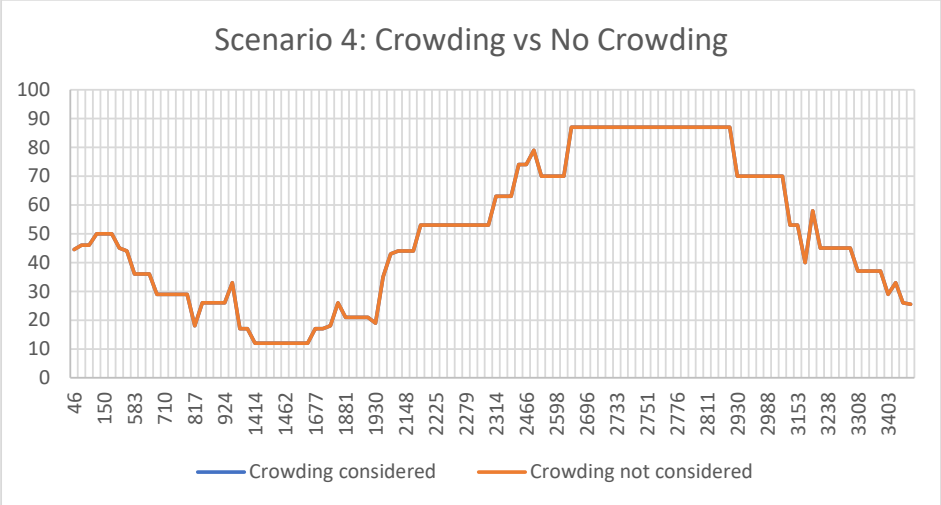

Figure G.4: Level of crowding comparison for scenario #4

[This page has been intentionally left blank]

Appendix H

Test Scenarios

In this annex we show the screenshots of the different stages on the mobile application when a user is setting the constraints and preferences correspondent to each test scenario. We also show the *JSON* messages that are traded between the *ROUTE* microservice and the *APP*.

H.1 Test Scenario #1

Figure H.1 represents the choices of the user in the mobile application for the first scenario.

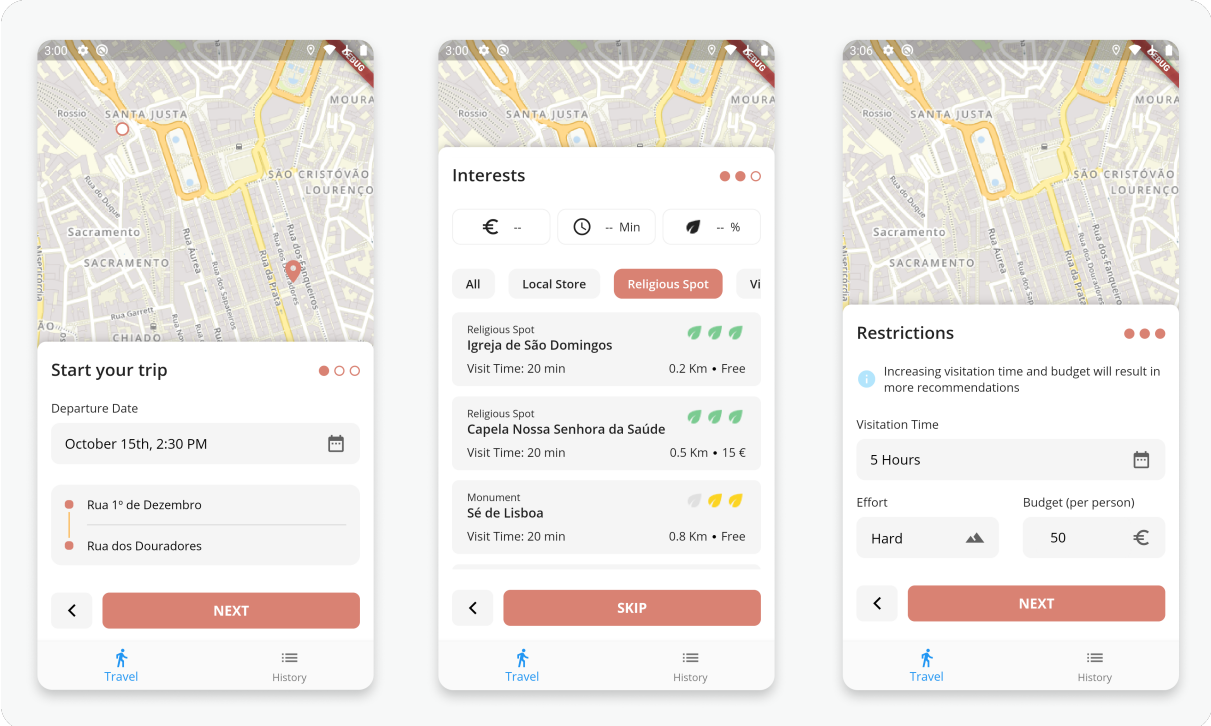

Figure H.1: Mobile application screenshot for scenario #1

H.1.1 Scenario #1 - *RouteRequest* example

Listing H.1 is the JSON example of the *RouteRequest* for the first scenario. It has the information of the figure H.1, the same as in the table 6.1.

Listing H.1: Scenario #1 *RouteRequest* query

```
"origin":{
    "latitude":38.7144118,
    "longitude":-9.1408772
},
"destination":{
      "latitude":38.7115605,
      "longitude":-9.1367243
},
"departureDate":1602772200000,
"visitationTime":300,
"budget":50,
"effortLevel":3,
"selectedPoints":[],
"selectedCategories":[2,7],
"checkWeather": false
```

H.1.2 Scenario #1 - *RouteResponse* example

Listing H.2 is the JSON example of the response send by the *ROUTE* for the first scenario request.

Listing H.2: Scenario #1: *RouteResponse*

```
"origin":{
      "latitude":38.7144118,
      "longitude":-9.1408772
},
 "destination":{
      "latitude":38.7115605,
      "longitude":-9.1367243
},
"time":{
      "startTime":1602772200000,
      "endTime":1602786480000
},
"durationTime":14305687,
"distance":4869.105330647069,
"price":13,
"calories":229,
"sustainability":84,
"line":[
    {
        "latitude":38.71449467646691,
        "longitude":-9.14075525419612
```

```
    },
    ..., //Omitted for brevity
    {
        "latitude":38.7115624504062,
        "longitude":-9.136716261610735
    }
],
"pois":[
    {
        "timestamp":1602772367000,
        "poi": {
            "name":"Igreja de Sao Domingos",
            ..., //Omitted for brevity
        }
    },
    {
        "timestamp":1602775033000,
        "poi": {
            "name":"Castelo de Sao Jorge",
            ..., //Omitted for brevity
        }
    },
    {
        "timestamp":1602779775000,
        "poi": {
            "name":"Igreja de Santo Antonio de Lisboa",
            ..., //Omitted for brevity
        }
    }
    ... //Omitted for brevity
]
```

H.1.3 Scenario #1 - Result representation

Figure H.2 is what the mobile application shows to the user after getting the route response to the first scenario.

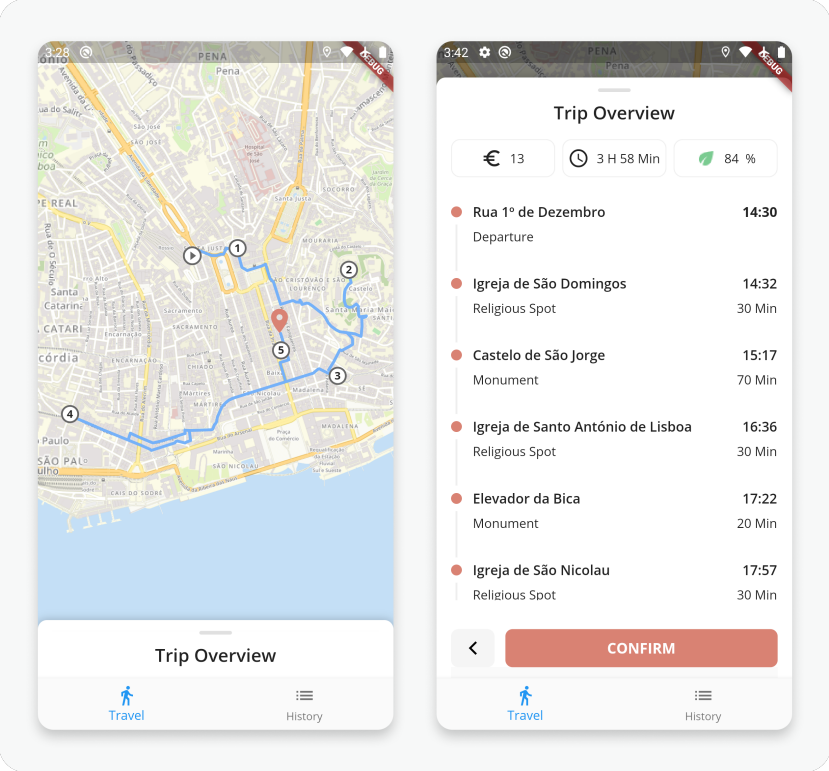

Figure H.2: Mobile application screenshot for scenario #1 result

H.2 Test Scenario #2

Figure H.3 represents the choices of the user in the mobile application for the second scenario.

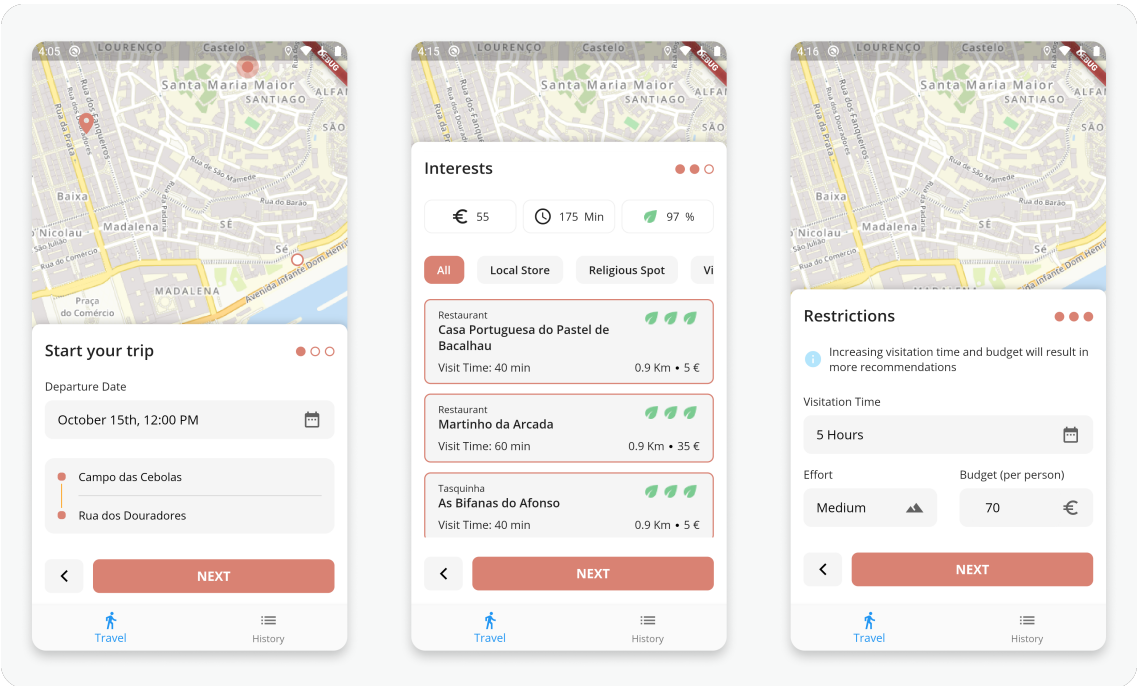

Figure H.3: Mobile application screenshot for scenario #2

H.2.1 Scenario #2 - *RouteRequest* example

Listing H.3 is the JSON example of the *RouteRequest* for the first scenario. It has the information of the figure H.3, the same as in the table 6.2.

Listing H.3: Scenario #2: *RouteRequest* query

```
"origin":{
  "latitude":38.7087856,
  "longitude":-9.1309565
},
"destination":{
  "latitude":38.7115605,
  "longitude":-9.1367243
},
"departureDate":1602763200000,
"visitationTime":300,
"budget":70,
"effortLevel":2,
"selectedPoints":[22,38,34,36,33],
"selectedCategories":[1,2,3,4,5,6,7,8],
"checkWeather": false
```

H.2.2 Scenario #2 - *RouteResponse* example

Listing H.4 is the JSON example of the response send by the *ROUTE* for the first scenario request.

Listing H.4: Scenario #2: *RouteResponse*

```
"origin":{
      "latitude":38.7087856,
      "longitude":-9.1309565
},
 "destination":{
      "latitude":38.7115605,
      "longitude":-9.1367243
},
"time":{
      "startTime":1602763200000,
      "endTime":1602776720000
},
"durationTime":13533855,
"distance":2547.0722801598895,
"price":66,
"calories":125,
"sustainability":68,
"line":[
    {
        "latitude":38.70860329900734,
        "longitude":-9.130819564102051
    },
```

```
    ..., //Omitted for brevity
    {
        "latitude":38.7115624504062,
        "longitude":-9.136716261610735
    }
],
"pois":[
    {
        "timestamp":1602763570000,
        "poi": {
            "name":"Martinho da Arcada",
            ..., //Omitted for brevity
        }
    },
    {
        "timestamp":1602767394000,
        "poi": {
            "name":"Casa Portuguesa do Pastel de Bacalhau",
            ..., //Omitted for brevity
        }
    },
    {
        "timestamp":1602770103000,
        "poi": {
            "name":"Nicola Café",
            ..., //Omitted for brevity
        }
    }
    ... //Omitted for brevity
]
```

H.2.3 Scenario #2 - Result representation

Figure H.4 is what the mobile application shows to the user after getting the route response to the second scenario.

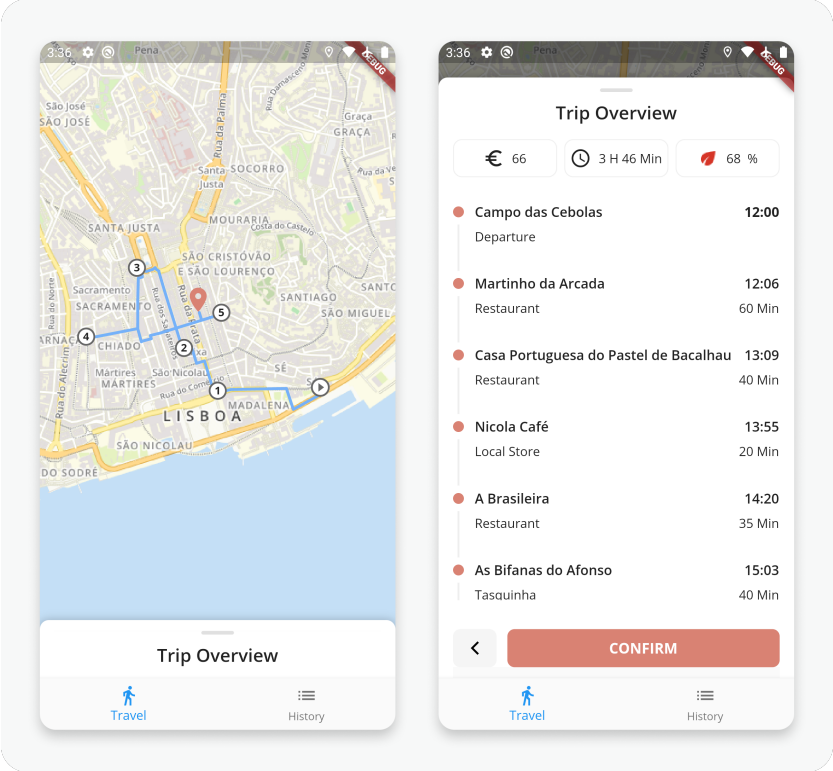

Figure H.4: Mobile application screenshot for scenario #2 result

H.3 Test Scenario #3

Figure H.5 represents the choices of the user in the mobile application for the third scenario.

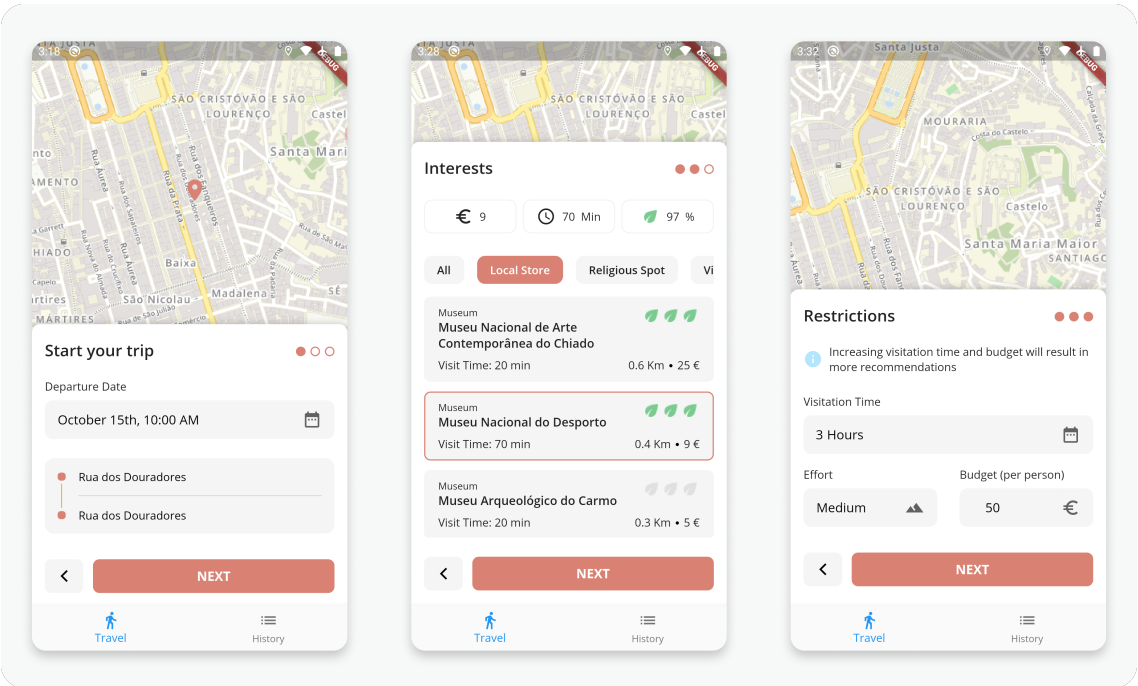

Figure H.5: Mobile application screenshot for scenario #3

H.3.1 Scenario #3 - *RouteRequest* example

Listing H.5 is the JSON example of the *RouteRequest* for the first scenario. It has the information of the figure H.5, the same as in the table 6.3.

Listing H.5: Scenario #3: *RouteRequest* query

```
"origin":{
  "latitude":38.7115605,
  "longitude":-9.1367243
},
"destination":{
  "latitude":38.7115605,
  "longitude":-9.1367243
},
"departureDate":1602756000000,
"visitationTime":180,
"budget":50,
"effortLevel":2,
"selectedPoints":[32],
"selectedCategories":[1,3,6],
"checkWeather": true
```

H.3.2 Scenario #3 - *RouteResponse* example

Listing H.6 is the JSON example of the response send by the *ROUTE* for the first scenario request.

Listing H.6: Scenario #3: *RouteResponse*

```
"origin":{
      "latitude":38.7115605,
      "longitude":-9.1367243
},
 "destination":{
      "latitude":38.7115605,
      "longitude":-9.1367243
},
"time":{
      "startTime":1602756000000,
      "endTime":1602767565000
},
"durationTime":11577370,
"distance":2329.7214642941553,
"price":23,
"calories":117,
"sustainability":85,
"line":[
    {
        "latitude":38.7115624504062,
        "longitude":-9.136716261610735
    },
```

```
    ..., // Omitted for brevity
    {
        "latitude":38.7115624504062,
        "longitude":-9.136716261610735
    }
],
"pois":[
    {
        "timestamp":1602756273000,
        "poi": {
            "name":"Manteigaria Silva",
            ..., // Omitted for brevity
        }
    },
    {
        "timestamp":1602758185000,
        "poi": {
            "name":"Museu Nacional do Desporto",
            ..., // Omitted for brevity
        }
    },
    {
        "timestamp":1602763107000,
        "poi": {
            "name":"Museu Nacional de Arte Contemporânea do Chiado",
            ..., // Omitted for brevity
        }
    }
]
```

H.3.3 Scenario #3 - Result representation

Figure H.6 is what the mobile application shows to the user after getting the route response to the third scenario.

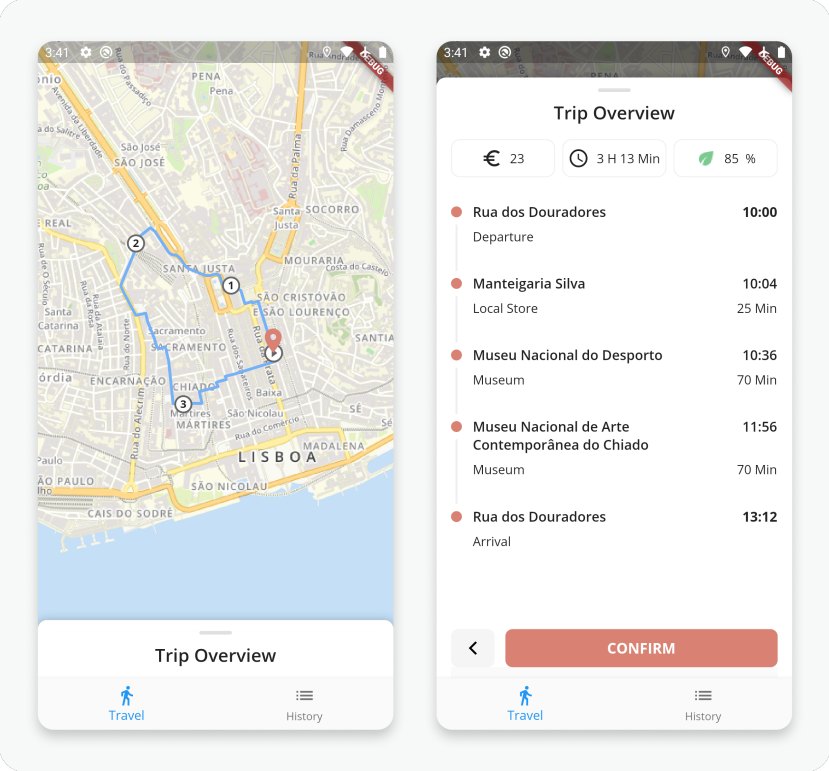

Figure H.6: Mobile application screenshot for scenario #3 result

H.4 Test Scenario #4

Figure H.7 represents the choices of the user in the mobile application for the fourth scenario.

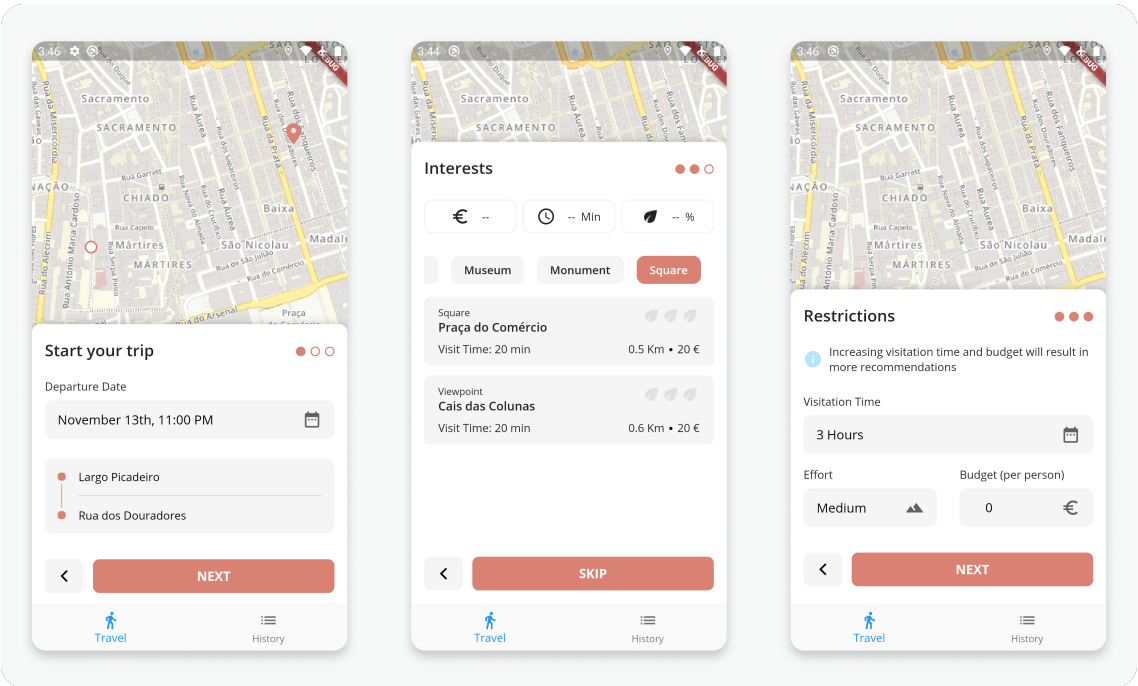

Figure H.7: Mobile application screenshot for scenario #4

H.4.1 Scenario #4 - *RouteRequest* example

Listing H.7 is the JSON example of the *RouteRequest* for the first scenario. It has the information of the figure H.7, the same as in the table 6.4.

Listing H.7: Scenario #4: *RouteRequest* query

```
"origin":{
  "latitude":38.7093008,
  "longitude":-9.141986
},
"destination":{
  "latitude":38.7115605,
  "longitude":-9.1367243
},
"departureDate":1602802800000,
"visitationTime":180,
"budget":0,
"effortLevel":2,
"selectedPoints":[],
"selectedCategories":[3,8],
"checkWeather": false
```

H.4.2 Scenario #4 - *RouteResponse* example

Listing H.8 is the JSON example of the response send by the *ROUTE* for the first scenario request.

Listing H.8: Scenario #4: *RouteResponse*

```
"origin":{
      "latitude":38.7093008,
      "longitude":-9.141986
},
 "destination":{
      "latitude":38.7115605,
      "longitude":-9.1367243
},
"time":{
      "startTime":1602802800000,
      "endTime":1602809871000
},
"durationTime":7096643,
"distance":3606.5402985991464,
"price":0,
"calories":170,
"sustainability":80,
"line":[
    {
        "latitude":38.709302325206714,
        "longitude":-9.141963953758477
```

```
    },
    ..., // Omitted for brevity
    {
        "latitude":38.7115624504062,
        "longitude":-9.136716261610735
    }
],
"pois":[
    {
        "timestamp":1602803309000,
        "poi": {
            "name":"Praça do Rossio",
            ..., // Omitted for brevity
        }
    },
    {
        "timestamp":1602804699000,
        "poi": {
            "name":"Arco da Rua Augusta",
            ..., // Omitted for brevity
        }
    },
    {
        "timestamp":1602805621000,
        "poi": {
            "name":"Praça do Comércio",
            ..., // Omitted for brevity
        }
    }
    ... // Omitted for brevity
]
```

H.4.3 Scenario #4 - Result representation

Figure H.8 is what the mobile application shows to the user after getting the route response to the fourth scenario.

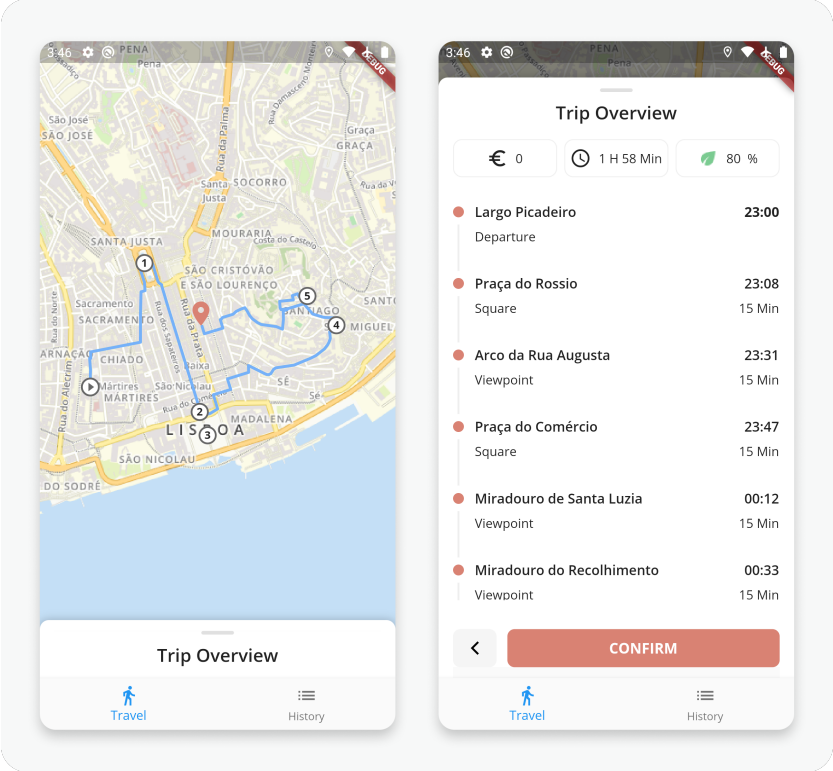

Figure H.8: Mobile application screenshot for scenario #4 result

H.5 Test Scenario #5

Figure H.9 represents the choices of the user in the mobile application for the fifth scenario.

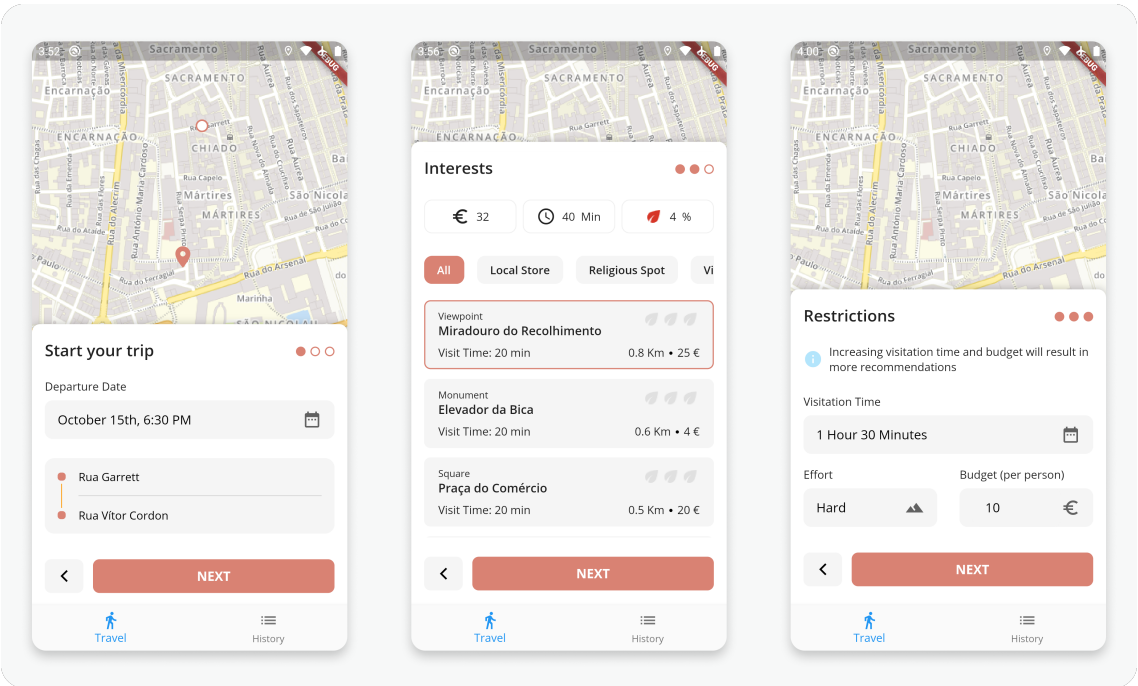

Figure H.9: Mobile application screenshot for scenario #5

H.5.1 Scenario #5 - *RouteRequest* example

Listing H.9 is the JSON example of the *RouteRequest* for the first scenario. It has the information of the figure H.9, the same as in the table 6.5.

Listing H.9: Scenario #5: *RouteRequest* query

```
"origin":{
  "latitude":38.710793100000004,
  "longitude":-9.140844399999999
},
"destination":{
  "latitude":38.7079863,
  "longitude":-9.141485
},
"departureDate":1602786600000,
"visitationTime":90,
"budget":10,
"effortLevel":3,
"selectedPoints":[19,38],
"selectedCategories":[1,2,3,4,5,6,7,8],
"checkWeather": false
```

H.5.2 Scenario #5 - *RouteResponse* example

Listing H.10 is the JSON example of the response send by the *ROUTE* for the first scenario request.

Listing H.10: Scenario #5: *RouteResponse*

```
"code": 204,
"codeJustiffication": "There is any possible recommendation regarding the chosen
POIs, their schedule and the user available time"
```

H.5.3 Scenario #5 - Result representation

Figure H.10 is what the mobile application shows to the user after getting the route response to the fifth scenario.

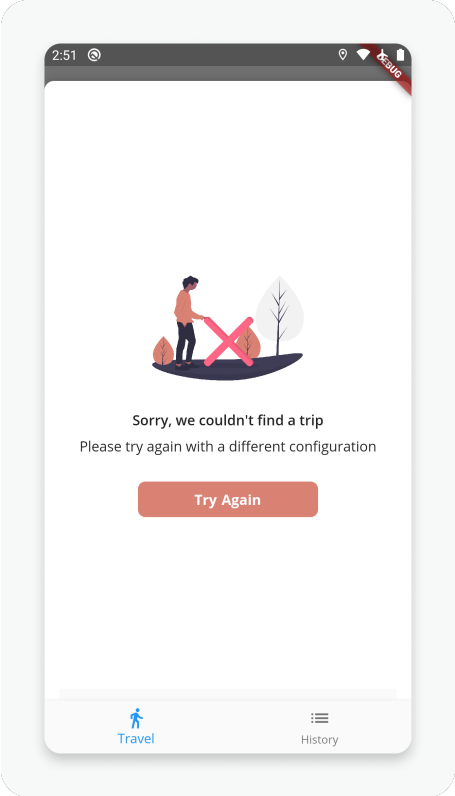

Figure H.10: Mobile application screenshot for scenario #5 result

[This page has been intentionally left blank]

Appendix I.

ROUTE Project Class Diagram

Figure I.1 shows the package diagram of the *ROUTE* system implementation in UML. We can see that the system is divided into six packages that have dependencies between them.

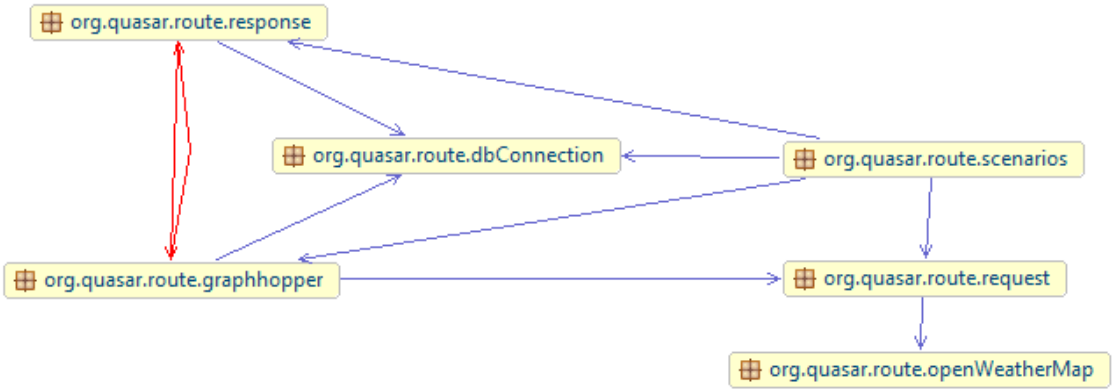

Figure I.1: Package diagram of *ROUTE* implementation

Figure I.2 exhibits the structure of the *ROUTE* system through an UML class diagram. Those classes are logically grouped in the packages illustrated in Figure I.1.

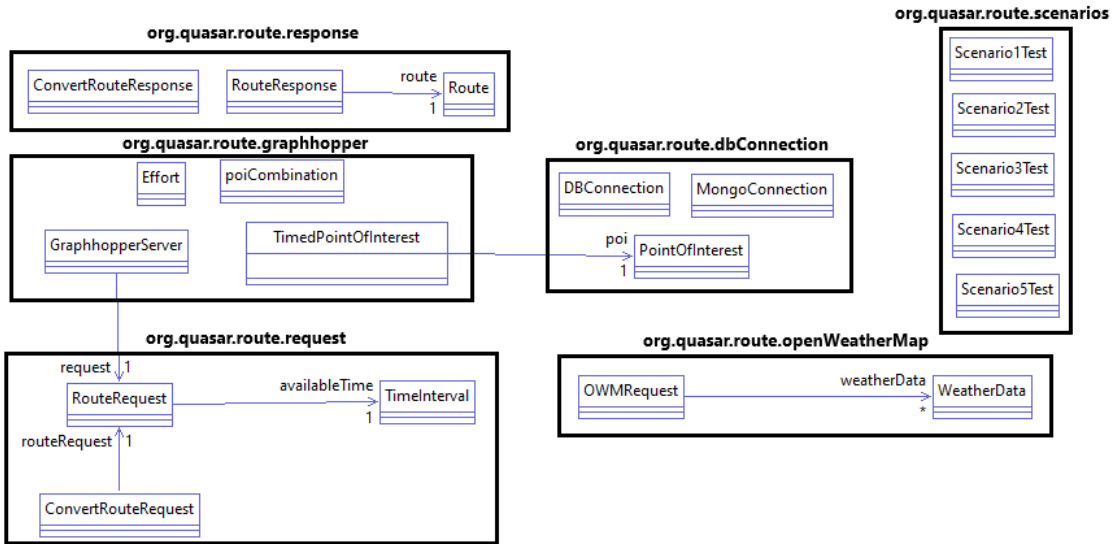

Figure I.2: Class diagram of *ROUTE* implementation

ANNEX I.

ETIS Core Indicators

In the next images are represented the core indicators of sustainability monitoring defined by ETIS [74].

Section A: Destination management		
Criteria	**Indicator reference#**	**ETIS core indicators**
A.1 Sustainable tourism public policy	**A.1.1**	Percentage of tourism enterprises/establishments in the destination using a voluntary certification/labelling for environmental /quality/sustainability and/or Corporate Social Responsibility
A.2 Customer satisfaction	**A.2.1**	Percentage of tourists and same-day visitors that are satisfied with their overall experience in the destination
	A.2.2	Percentage of repeat/return visitors (within 5 years)

Figure I.1: Destination management indicators

Section B: Economic value		
Criteria	**Indicator reference#**	**ETIS core indicators**
B.1 Tourism flow (volume and value) at destination	B.1.1	Number of tourist nights per month
	B.1.2	Number of same-day visitors per month
	B.1.3	Relative contribution of tourism to the destination's economy (% GDP)
	B.1.4	Daily spending per overnight tourist
	B.1.5	Daily spending per same-day visitors
B.2 Tourism enterprise(s) performance	B.2.1	Average length of stay of tourists (nights)
	B.2.2	Occupancy rate in commercial accommodation per month and average for the year
B.3 Quantity and quality of employment	B.3.1	Direct tourism employment as percentage of total employment in the destination
	B.3.2	Percentage of jobs in tourism that are seasonal
B.4 Tourism supply chain	B.4.1	Percentage of locally produced food, drinks, goods and services sourced by the destination's tourism enterprises

Figure I.2: Economic value indicators

Section C: Social and cultural impact		
Criteria	**Indicator reference#**	**ETIS core indicators**
C.1 Community/social impact	C.1.1	Number of tourists/visitors per 100 residents
	C.1.2	Percentage of residents who are satisfied with tourism in the destination (per month/season)
	C.1.3	Number of beds available in commercial accommodation establishments per 100 residents
	C.1.4	Number of second homes per 100 homes
C.2 Health and safety	C.2.1	Percentage of tourists who register a complaint with the police
C.3 Gender equality	C.3.1	Percentage of men and women employed in the tourism sector
	C.3.2	Percentage of tourism enterprises where the general manager position is held by a woman
C.4 Inclusion/accessibility	C.4.1	Percentage of rooms in commercial accommodation establishments accessible for people with disabilities
	C.4.2	Percentage of commercial accommodation establishments participating in recognised accessibility information schemes
	C.4.3	Percentage of public transport that is accessible to people with disabilities and specific access requirements
	C.4.4	Percentage of tourist attractions that are accessible to people with disabilities and/or participating in recognised accessibility information schemes
C.5 Protecting and enhancing cultural heritage, local identity and assets	C.5.1	Percentage of residents that are satisfied with the impacts of tourism on the destination's identity
	C.5.2	Percentage of the destination's events that are focused on traditional/local culture and heritage

Figure I.3: Social and cultural impact indicators

Section D: Environmental impact		
Criteria	**Indicator reference#**	**ETIS core indicators**
D.1 Reducing transport impact	D.1.1	Percentage of tourists and same-day visitors using different modes of transport to arrive at the destination
	D.1.2	Percentage of tourists and same-day visitors using local/soft mobility/public transport services to get around the destination
	D.1.3	Average travel (km) by tourists and same-day visitors from home to the destination
	D.1.4	Average carbon footprint of tourists and same-day visitors travelling from home to the destination
D.2 Climate change	D.2.1	Percentage of tourism enterprises involved in climate change mitigation schemes — such as: CO_2 offset, low energy systems, etc.— and 'adaptation' responses and actions
	D.2.2	Percentage of tourism accommodation and attraction infrastructure located in 'vulnerable zones'
D.3 Solid waste management	D.3.1	Waste production per tourist night compared to general population waste production per person (kg)
	D.3.2	Percentage of tourism enterprises separating different types of waste
	D.3.3	Percentage of total waste recycled per tourist compared to total waste recycled per resident per year
D.4 Sewage treatment	D.4.1	Percentage of sewage from the destination treated to at least secondary level prior to discharge
D.5 Water management	D.5.1	Water consumption per tourist night compared to general population water consumption per resident night
	D.5.2	Percentage of tourism enterprises taking actions to reduce water consumption
	D.5.3	Percentage of tourism enterprises using recycled water
D.6 Energy usage	D.6.1	Energy consumption per tourist night compared to general population energy consumption per resident night
	D.6.2	Percentage of tourism enterprises that take actions to reduce energy consumption
	D.6.3	Percentage of annual amount of energy consumed from renewable sources (Mwh) compared to overall energy consumption at destination level per year
D.7 Landscape and biodiversity protection	D.7.1	Percentage of local enterprises in the tourism sector actively supporting protection, conservation and management of local biodiversity and landscapes

Figure I.4: Environmental impact indicators

[This page has been intentionally left blank]

ANNEX II.

USE CASE DIAGRAM OF THE *POINT* WEB PLATFORM

This annex shows the use case diagram of the web platform of the *POINT* microservice. Here we can demonstrate how parish workers access the database, add POIs and add its sustainability values.

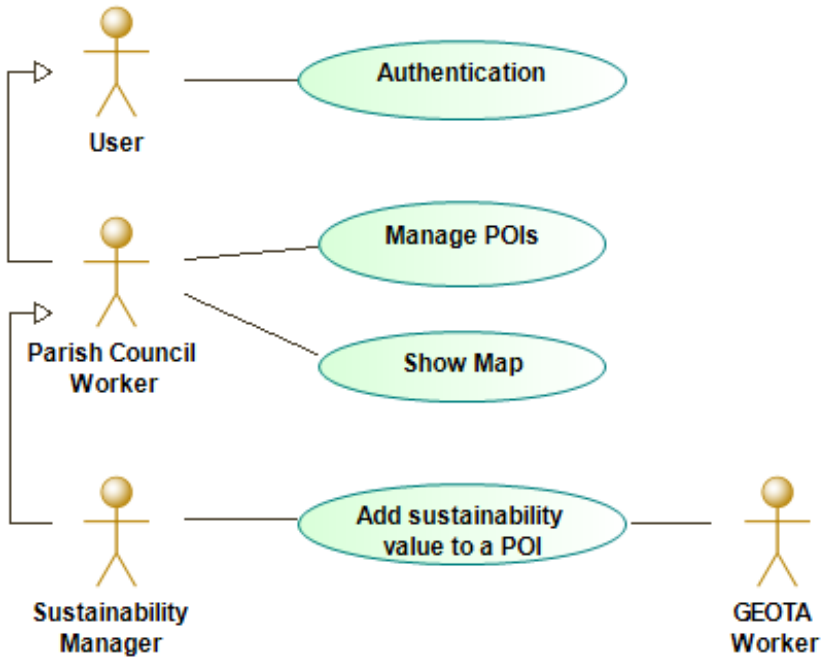

Figure II.1: Use case diagram of the *POINT* web platform (adapted from: [78])

As participants (actors) in the diagram we have: a *User*, a *Parish Council Worker*, a *Sustainability Manager* and an *GEOTA Worker*.

As use cases of the diagram we have: *Authentication*, *Manage POIs*, *Show Map* and *Add Sustainability Value to a POI*.

A *User* is an actor that accesses the web platform and executes the use case *Authentication*. If this authentication (username and password) is successful, then it means the user is a *Parish Council Worker*. These actors have permissions to explore the map (*Show map*) and insert, update or delete data about the POIs (*Manage POIs*). Some parish council workers have some special permissions because of being sustainability specialists (*Sustainability Manager*) and can add sustainability values to POIs (*Add sustainability value to a POI*) by working together with external workers that help in this process (*GEOTA Workers*).