Instituto Universitário de Lisboa

Departamento de Ciências e Tecnologias da Informação

Desenvolvimento de aplicação móvel para submissão/revisão de despesas recorrendo a metodologias ágeis de desenvolvimento e à plataforma *low-code* OutSystems

Sara Karmali

Trabalho de projeto submetido como requisito parcial para obtenção do grau de

Mestre em Informática e Gestão

Orientador:
Doutor Carlos Coutinho, Professor Auxiliar,
ISCTE-IUL

Outubro, 2019

Esta página foi propositadamente deixada em branco.

Agradecimentos

Ao professor Carlos Coutinho, agradeço por todas as orientações e discussões que por fim me ajudaram a definir e melhorar em todas as fases deste projeto.

Um gigante obrigada aos meus colegas da IG&H, especialmente ao Bruno, à Cecília, ao Fábio, ao Délio, ao Tiago, à Cristina, ao Nuno, à Cátia, ao João Heleno, ao Constantino, entre outros, por todo o apoio, incentivo, ajuda e orientação incansável desde o início ao fim.

Aos meus colegas da Noesis, especialmente ao Tiago Fernandes e ao Idálio Rodrigues, agradeço por todo o apoio e disponibilidade para ajudar.

À minha família, pelo apoio e incentivo incondicional, muito obrigada.

Agradeço aos meus amigos, em especial à Jéssica, por todo o apoio.

Esta página foi propositadamente deixada em branco.

Resumo

Na atualidade, constata-se uma crescente necessidade de otimização de processos, que se prende com a redução de custos e de tempo. Existe, em termos gerais, uma ineficiência no processamento de despesas e faturas, tais como: a digitalização e anexo de faturas; a necessidade de internet; e a necessidade de verificar constantemente se existem despesas por rever, correndo o risco de as negligenciar por ausência de notificação para o gestor responsável.

Estes problemas podem ser traduzidos num problema geral e comum a todas as empresas, que dificultam a redução de tempo e custos.

Assim, o objetivo deste projeto é encontrar uma solução que permita automatizar e adicionar mobilidade ao processamento de despesas, agilizando o processo de forma a evitar custos desnecessários e ineficiências. Com isto, pretende-se colmatar os problemas acima levantados, pois estes processos serão consideravelmente menos custosos em termos de tempo e monetariamente quando forem automáticos e móveis.

Para desenvolver a solução proposta foi necessária uma definição clara do seu âmbito; levantamento dos requisitos; elaboração das especificações; e desenho do calendário de projeto com ciclos de desenvolvimento de software, demonstrações, testes, definição de objectivos e entregáveis.

Esta aplicação, desenvolvida numa plataforma de *low-code* (OutSystems) traz como vantagens a mobilidade, e, consequentemente, a redução do tempo de processamento de despesas, de custos, e eliminação de possíveis falhas como a perca de faturas, resolvendo as ineficiências mencionadas em cima através da opção de anexar a fatura e do funcionamento *online* e *offline*.

Palavras-chave: OutSystems, *Scrum*, *low-code*, *sprint*, despesa, submissão, revisão; aplicação móvel.

Esta página foi propositadamente deixada em branco.

Abstract

Nowadays there is a growing need for process optimization, which is related to the reduction of costs and time. There is a general lack of efficiency in expense and invoice processes, with limitations such as: scanning and attachment of invoices; the need for internet; and constantly checking for unrevised expenses, at the risk of neglecting them due to lack of notification on the managers end.

These problems can be translated into a general problem common to all companies, which makes it difficult to reduce time and costs.

Thus, the purpose of this project is to find and develop a solution that automates and adds mobility to expense processing, streamlining the process to avoid unnecessary costs and inefficiencies. This is intended to address the problems raised above since these processes will be considerably less costly in time and monetarily when they become automatic and mobile, as argued throughout this paper.

Developing the proposed solution required a clear definition of its scope; requirements gathering; elaboration of specifications; and designing the project calendar with software development cycles, demonstrations, tests, milestones and deliverables.

This application, developed on top of a low-code platform (OutSystems) will have the advantages of mobility and, consequently, the reduction of processing time, costs and elimination of the risk of losing invoices, solving these issues through the option of attaching the invoice and working online and offline.

Keywords: OutSystems, Scrum, low-code, sprint, expense, submission, review; mobile application.

Esta página foi propositadamente deixada em branco.

Índice

Índice de Figuras

Índice de Tabelas

Esta página foi propositadamente deixada em branco.

Capítulo 1 - Introdução

Neste capítulo irá ser apresentado o problema a ser investigado, a metodologia da investigação incluindo questões de investigação e hipóteses e a estrutura do documento.

1.1 Enquadramento

No mundo empresarial atual, é essencial que os processos sejam ágeis e pouco custosos em termos de tempo. Parece seguro afirmar-se que qualquer redução em termos de custos monetários e de tempo será benéfica para qualquer contexto laboral. Esta redução poderá ser alcançada através de iniciativas variadas, das quais, por exemplo, a redução da utilização de papel (nomeadamente fotocópias de faturas); de recursos físicos para arquivo de documentos; e de gasto de tempo para passar documentos por vários intervenientes de vários departamentos.

Os benefícios destas iniciativas serão comprovados através da implementação da automação do processamento de faturas no contexto laboral na empresa IG&H Consulting BV.

Através do estudo deste caso em específico, foi constatada a ineficiência de ferramentas que agilizem o processo de submissão e revisão de despesas por parte dos colaboradores e respetivos gestores, das quais:

- Digitalização de faturas e anexação das mesmas;
- Necessidade de recorrer a um computador com internet para executar os procedimentos de submissão e de revisão;
- Necessidade de verificar constantemente se há despesas por rever, correndo o risco de as deixar por rever, pois poderão ter sido submetidas em cima do prazo e o gestor responsável não ter sido consequentemente avisado.

As ineficiências acima referidas podem ser traduzidas num problema geral e comum a todas as empresas, tanto pequenas como grandes: gestão e aprovação de faturas.

A necessidade de usar o computador com internet e de entregar a fatura em mão é comum, embora não seja prático e contenha um grau de risco que pode ser colmatado. Com efeito, o indivíduo que pretende submeter/rever a fatura poderá estar longe do escritório e não conseguir entregar/receber as faturas presencialmente, bem como querer submeter/rever uma despesa e não ter internet; poderá ainda ter perdido a fatura, entre outros potenciais problemas.

O objetivo deste trabalho de projeto é encontrar uma solução que permita agilizar o processamento de despesas e evitar custos desnecessários ou ineficiências, nomeadamente através de criação de uma aplicação que reduza custos na empresa.

Tendo em conta que se pretende comprovar as potencialidades de uma aplicação móvel para o processamento de faturas, este trabalho de projeto será útil na sua contribuição para a sustentação de processos, não só do contexto laboral que é usado como exemplo, como também noutras empresas que se deparem com a mesma problemática. Se tempo e dinheiro são recursos que todas as empresas pretendem poupar, então será seguro afirmar que processos automáticos e que diminuem gastos monetários, recursos físicos e agilizam processos serão uma mais valia em qualquer contexto profissional. Quando esta redução se consegue alcançar com o mínimo de gastos de implementação de novos processos, então confere-se uma mais valia acrescida.

1.2 Metodologia de Investigação

A metodologia de investigação usada é a *Design Science Research*. Esta metodologia baseia-se em 6 passos: identificação do problema e motivação, definir os objetivos da solução proposta, criar a solução proposta, demonstrar que a solução resolve uma ou mais partes do problema, avaliar a contribuição da solução e, por fim, comunicar sobre o problema e a solução criada (Peffers, Tuunanen, Rothenberger, & Chatterjee, 2007-8).

1.2.1 Questões de Investigação

Foram levantadas as seguintes questões de investigação:

1. Como poderá a utilização de uma aplicação móvel otimizar o tempo despendido num processo de tratamento de despesas?
2. De que forma poderá a automação de um processo de entrega de faturas reduzir ineficiências?

1.2.2 Hipóteses

Serão apresentadas abaixo soluções teóricas para cada uma das perguntas de investigação:

1. O fato dos utilizadores poderem aceder a uma plataforma móvel poderá contribuir para que assim que tenham as despesas consigam submetê-las de imediato, não necessitando de esperar até estarem junto a um computador ou até terem internet;
2. O fato dos intervenientes poderem digitalizar e anexar as faturas à despesa poderá contribuir para diminuir o tempo de processamento da mesma, bem como o risco de se perder a fatura e a perda de tempo consequente;
3. O fato dos utilizadores poderem digitalizar e anexar as suas faturas mitiga a necessidade de estarem com os restantes intervenientes presencialmente para as entregar.

1.3 Estrutura do documento

Este documento começa por apresentar o problema a ser investigado e os primeiros passos dessa investigação. O capítulo 2 apresenta a revisão da literatura que contém o trabalho relacionado, as aplicações existentes no mercado, a comparação entre as metodologias de desenvolvimento de software e a comparação entre plataformas low-code. No capítulo 3, realiza-se o desenho e implementação da aplicação recorrendo ao desenho de diagramas, à análise de requisitos, à arquitetura, aos esboços da aplicação, ao calendário, ao layout final e aos testes feitos à aplicação. No capítulo 4, apresentam-se as conclusões desta investigação. Para além disso, apresenta-se o Anexo A, descrito na secção 2.2, que contém um questionário cujo objetivo é perceber se é comum a utilização de aplicações móveis para submeter e rever despesas bem como, em caso afirmativo, que funcionalidades destes softwares é que os utilizadores mais apreciam por se mostrarem úteis.

Esta página foi propositadamente deixada em branco.

Capítulo 2 - Revisão de Literatura

Este capítulo contém a revisão da literatura que inclui o trabalho relacionado, as aplicações existentes no mercado, a comparação entre as metodologias de desenvolvimento de software e a comparação entre plataformas low-code.

2.1 Trabalho Relacionado

Fez-se uma pesquisa de artigos pelo problema geral: gestão e aprovação de faturas. Como resultado desta pesquisa foram apenas encontrados dois artigos que falam sobre essa mesma problemática.

Foi analisada a forma como os sistemas de automação de processos poderão resolver algumas das problemáticas levantadas pelo processamento de documentos, nomeadamente através do desenvolvimento de um sistema de fluxo de documentos.

De entre estes problemas, pode-se referenciar os vários níveis de burocracia envolvidos, e os custos inerentes ao processamento de documentos por via física, ao invés de digital. Uma outra desvantagem seria ainda a potencial demora destes processos, pois este exige que os vários intermediários estejam fisicamente em contacto com os documentos, coisa que o processo digital, pelo contrário, não exige.

Com efeito, todo o processo de aprovação de faturas é extremamente custoso e propenso a falhas e ineficiências. De acordo com (Murphy, 2012), todo o processo que inclui receber faturas de compras nas organizações, inserir dados no sistema de contabilidade, fotocopiar as faturas, enviar as faturas em papel por toda a empresa para autorização, colocar essas faturas nas contas a pagar e enviar recibos pode ser muito dispendioso, deixando as organizações vulneráveis a ineficiências ou erros.

Uma vez que as organizações/empresas têm, em termos gerais, o mesmo objetivo: aumentar a eficiência dos processos sem, no entanto, investir em soluções complexas e caras, é natural que se procurem formas de otimizar esses processos, pois exige-se, no mundo empresarial atualmente, uma crescente agilidade e facilidade no acesso e aprovação de documentos.

O sistema de fluxo de documentos – que se pode definir como consistindo no processo de obtenção de um documento submetido e seguidamente enviado para uma aprovação passo a passo e de forma automática (Tang & Chenghui, 2012), desde a fase de rascunho á de aprovação, – parece ser um passo em direção a essa otimização. Catherine Murphy confirma

esta possibilidade quando refere que a chave para melhorar a eficiência quando se trata de contas por pagar, é automatizar o processamento de faturas usando soluções eletrónicas de gestão de documentos.

Este processo consiste na criação de documentos que seguem o caminho pré-definido e que podem ser monitorizados em tempo real. É possível verificar os documentos por aprovar e já aprovados e assinados. Esta vantagem é consideravelmente apreciada por todas as partes integrantes do processo, nomeadamente tanto o indivíduo que submete o documento, como o que o avalia.

Murphy explica em ainda mais detalhe as melhorias que a digitalização deste processo traz, nomeadamente, redução de custos e melhoria da eficiência, sem perca de faturas: a administração fica mais livre e o processo de aprovação é muito mais simplificado; elimina a possibilidade de faturas perdidas; fornece visibilidade imediata do estado de aprovação da fatura e o *audit trail* monstra quem aprovou e quando; melhora a eficiência de autorização podendo poupar-se bastante dinheiro por ano. Em complemento, usar dispositivos móveis para aprovar despesas pode também agilizar o processo de autorização de faturas.

Em conclusão, o processamento de documentos prova-se potencialmente mais dinâmico quando automatizado e móvel. Esta dinamização vem ainda com diversas vantagens, nomeadamente relacionadas com a rentabilização não só do tempo/duração do processo, mas também com os custos inerentes a esse mesmo processo.

A solução apresentada pelos autores, é viável para melhorar a eficiência e agilidade do processo. No entanto, um dos problemas que é a necessidade de ter internet para se efetuar todos os processos não é colmatado através desta solução.

2.2 Aplicações existentes no mercado

De forma a encontrar a melhor solução, foram investigadas as aplicações existentes no mercado, listadas abaixo igualmente com as funções oferecidas:

- VExpenses (VExpenses | Reembolso de Despesas Corporativas, 2019)
  A aplicação, comercializada pela empresa VExpenses tem uma aplicação móvel de registo de despesas e também um *website* para dar apoio à aplicação onde se pode ver relatórios de gastos. Para usar em ambiente empresarial o preço seria de 17€ por

utilizador por mês. Abaixo encontra-se uma lista de funcionalidades da aplicação móvel:

  - Possibilidade de tirar fotografia à fatura;
  - Registo e aprovação de despesas;
  - *Push notifications;*
  - *Online* e *offline.*

- Concur Expenses (Expense Management Made Easy - SAP Netherlands, 2019)

  A aplicação Concur Expenses é uma aplicação para registar despesas comercializada pela SAP Software Solutions. A aplicação móvel é também suportada por um *website* que fornece relatórios de gastos. Abaixo encontra-se uma lista de funcionalidades da aplicação móvel:

  - Possibilidade de tirar fotografia à fatura;
  - Registo e aprovação de despesas;
  - *Push notifications.*

- Rydoo Expense (Rydoo Expense - Rydoo, 2019)

  A aplicação Rydoo Expense é comercializada pela empresa Rydoo. Consiste numa aplicação móvel para registo de despesas com um preço de 6€ por mês por utilizador para um máximo de 50 utilizadores. Abaixo encontra-se uma lista de funcionalidades da aplicação móvel:

  - Possibilidade de tirar fotografia a fatura;
  - Registo e aprovação de despesas;
  - *Push notifications;*
  - *Online* e *offline.*

- Chrome River (Expense Management Software, 2019)

  Esta aplicação é comercializada pela empresa Chrome River e consiste numa aplicação móvel para registo de despesas, que não permite funcionamento *offline*. Abaixo encontra-se uma lista de funcionalidades:

  - Possibilidade de tirar fotografia a fatura;
  - Registo e aprovação de despesas;
  - *Push notifications.*

- Zoho Expense (Online Expense Report Software, 2019)
  Esta aplicação, comercializada pela empresa Zoho é uma aplicação móvel para registo de despesas com um preço de 2.5€ por utilizador por mês. Abaixo encontra-se uma lista de funcionalidades da aplicação:
    - Possibilidade de tirar fotografia a fatura;
    - Registo e aprovação de despesas;
    - *Push notifications;*
    - *Online* e *offline.*

Todas as aplicações listadas acima necessitam de uma licença paga para serem usadas, sendo o valor mínimo 2,5€ por mês e por utilizador. Todas elas têm também uma aplicação móvel que permite fazer registo e aprovação de despesas com possibilidade de tirar fotografia à fatura. Utilizam ainda as *push notifications* para avisar a pessoa que revê e a pessoa que submete que a despesa alterou de estado. Das 5, apenas a VExpenses, a Rydoo e a Zoho Expense têm funcionamento tanto *online* como *offline*.

Foram realizados questionários qualitativos (Anexo A) a uma amostra de 31 pessoas não pertencentes á IG&H de forma a perceber se é comum a utilização de aplicações deste género, bem como, em caso afirmativo, que funcionalidades destes *softwares* é que os utilizadores mais apreciam por se mostrarem úteis. Neste questionário percebe-se que a forma mais popular de agilizar o processo de submissão de faturas é o desenvolvimento de aplicações internas para o efeito, tal como defendido ao longo deste projeto.

De entre as aplicações mencionadas no inquérito, percebe-se que recursos de submissão e aprovação de despesas são preponderantes, bem como a existência de notificações para as partes envolvidas no processo, nomeadamente quem submete e aprova as faturas. Identifica-se ainda como sendo do agrado dos utilizadores a possibilidade de anexação de faturas; a existência da aplicação móvel e o funcionamento *online* e *offline*, como se pode ver na Figura 1:

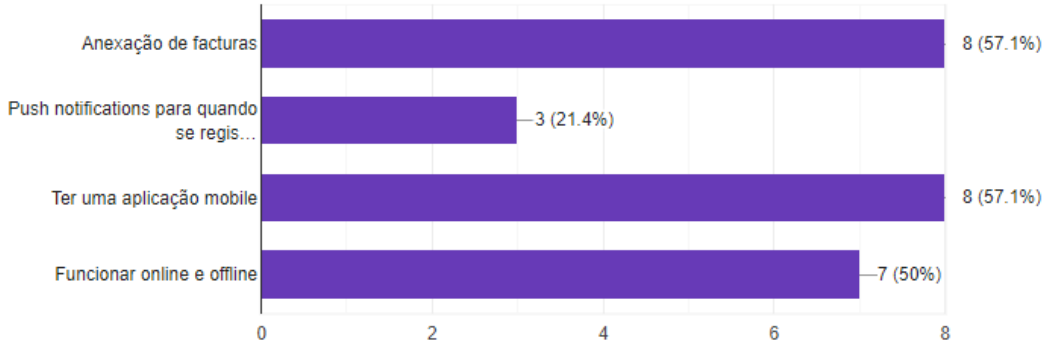

Figura 1 - Melhorias sugeridas pelos utilizadores para a aplicação usada na empresa

No entanto, a resposta mais comum foi a inexistência de aplicações para submissão de fatura, sendo que as repostas nestes casos demonstram uma abertura considerável para o uso deste tipo de aplicações caso implementadas, mencionado como fatores positivos destas agilizarem e simplificarem o processo.

2.3 Metodologias de desenvolvimento de processos de software

Segundo (Nerur, Mahapatra, & Mangalaraj, 2005), o desenvolvimento de *software* é uma atividade complexa caracterizada por tarefas e requisitos que exibem um elevado grau de variabilidade. Abaixo seguem-se os dois tipos de metodologias de desenvolvimento de *software.*

2.3.1 Metodologias tradicionais e ágeis

As metodologias tradicionais de desenvolvimento de software são centradas no processo, sendo o seu objetivo o aumento da previsibilidade e eficiência no desenvolvimento de *software*.

Nas metodologias tradicionais, a documentação e definição de requisitos é requerida no início do projeto de forma a evitar alterações e o *feedback* dos clientes e os testes são os últimos passos.

Assim, estas metodologias, das quais a mais predominante é o modelo em cascata, baseiam-se em 4 passos sequenciais: a definição dos requisitos; o desenvolvimento da solução; a fase dos testes da solução; e, por último, a implementação da mesma (Awad, 2005).

Pelo contrário, as metodologias ágeis, são essencialmente diferenciadas pela possibilidade de mudança a qualquer ponto do projeto, sendo a interação entre membros da equipa encorajada pois: as pessoas conseguem trocar ideias mais rápido falando pessoalmente do que escrevendo e lendo documentos (Cockburn & Highsmith, 2001). Da equipa não só fazem parte os programadores como também o cliente, cujo *feedback* é valioso para a implementação (Nerur, Mahapatra, & Mangalaraj, 2005) e é essencial que seja dado a qualquer momento e não só no fim do projeto.

Assim, as metodologias ágeis, de acordo com (Beck, et al., 2001) caracterizam-se por:

- Indivíduos e interações acima de processos e ferramentas;
- Software a trabalhar acima de documentação completa;
- Colaboração do cliente acima da negociação do contracto;
- Adaptação à mudança acima de seguir um plano.

Assim, as metodologias tradicionais e ágeis diferenciam-se em 5 pontos principais:

- **Papel do cliente:**

  Nas metodologias ágeis, o papel do cliente é crucial para o sucesso do projeto, pois ele faz parte da equipa. O *feedback* do cliente e os testes devem ser um exercício diário, pois quando os programadores falam com os clientes, é possível resolver dificuldades, ajustar prioridades e examinar caminhos alternativos.

  Já nas tradicionais, o papel do cliente é importante apenas durante a especificação do desenvolvimento. Nas outras atividades, a sua importância é mínima (Nerur, Mahapatra, & Mangalaraj, 2005). Tanto o *feedback* do cliente como os testes são os últimos passos do ciclo de vida do projeto.

- **Importância dos intervenientes:**

  Nas metodologias tradicionais, as pessoas são tratadas como componentes previsíveis tal como os processos. Nas metodologias ágeis, o foco está no talento do individuo e os processos são moldados para pessoas e equipas especificas.

- **Adaptação à mudança**:

  As metodologias tradicionais requerem a definição e documentação de um conjunto estável de requisitos no início do projeto (Awad, 2005), pois acreditam que assim é possível reduzir o custo eliminando a mudança. Já as metodologias ágeis, encorajam a adaptação à mudança. Porém, da mesma forma que se adapta à mudança é também importante continuar a reter qualidade.

- **Comunicação entre a equipa:**

  Nas metodologias tradicionais, é produzida uma grande quantidade de documentação que codifica o processo e o conhecimento do produto. A comunicação entre os participantes do projeto é feita por esta documentação (Nerur, Mahapatra, & Mangalaraj, 2005). No entanto, segundo (Cockburn & Highsmith, 2001), as pessoas conseguem trocar ideias mais rápido falando cara a cara em vez de escreverem e lerem documentos, tal como acontece nas metodologias ágeis.

- **Definição de sucesso:**

  Para as metodologias tradicionais, um projeto foi entregue com sucesso quando foi entregue no prazo definido com o custo definido. Já nas metodologias ágeis, o sucesso é medido questionando se o cliente tem um software que é uma mais valia para o mesmo, em comparação com o seu custo (Awad, 2005).

Quando se comparam as duas metodologias, parece seguro afirmar que a metodologia ágil é mais vantajosa. Com efeito, enquanto que nas metodologias tradicionais o produto apenas é demonstrado ao cliente no final do desenvolvimento, nas metodologias ágeis o cliente é consultado ao longo de todo o processo, nomeadamente no final de cada *sprint*. Esta característica confere ao cliente uma maior confiança no produto que adquire, bem como á equipa de consultores *feedback* construtivo ao longo do processo, o que no final irá potenciar o sucesso do produto e a sua utilidade a longo prazo.

Também a definição do grau de sucesso do produto desenvolvido demonstra a importância que as metodologias ágeis dão ao cliente como interveniente predominante ao longo do processo.

Com efeito, enquanto que nas tradicionais um produto é bem-sucedido quando entregue dentro do prazo e do *budget*, nas metodologias ágeis consideram antes a utilidade do produto para o cliente.

Tal como mencionado anteriormente, também a comunicação é diferenciada no caso das metodologias ágeis. Segundo esta, os intervenientes devem comunicar de forma presencial/direta, o que confere maior grau de entendimento mútuo e possibilidade de debate e troca de ideias, por oposição às metodologias tradicionais, onde a comunicação é feita por documentação.

Concluindo, apesar da metodologia ágil ser a escolhida para este trabalho de projeto, reconhece-se que as metodologias tradicionais têm evoluído e já existem algumas variantes que têm aprendido com as metodologias ágeis, sendo muito mais iterativas e interativas.

2.3.2 *Scrum*

Das metodologias ágeis existentes, a metodologia *Scrum* tem sido a mais popular (West, Gilpin, Grant, & Anderson, 2011), como se pode ver na Figura 2, sendo que a mesma apresenta uma percentagem de 12.3%.

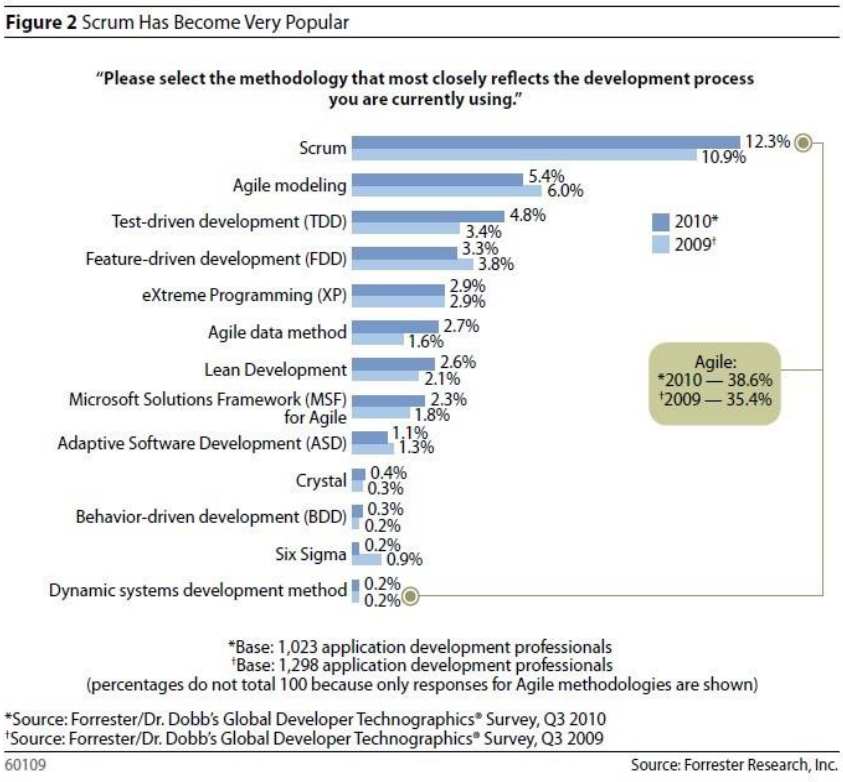

Figura 2 - Desenvolvimento da popularidade da metodologia SCRUM (West, Gilpin, Grant, & Anderson, 2011)

Segundo (Schwaber & Sutherland, 2017), *Scrum* é uma metodologia de gestão, melhoria e manutenção para um sistema ou protótipo de produção.

Na metodologia *Scrum*, desenvolve-se o projeto por sprints, que são períodos pré-definidos de duas semanas ou mais, nas quais no final é entregue um conjunto de requisitos prontos a ser utilizados. No início do projeto, é definido o *backlog* do produto onde estão inseridos todos os requisitos do projeto, que vai por sua vez servir para construir o *backlog* do *sprint* antes do seu início.

Para a construção do *backlog* do *sprint*, é necessário o planeamento do *sprint* que consiste numa reunião onde se discutem as *user stories* e se debatem as dúvidas que surgiram.

Durante o *sprint*, para além do desenvolvimento dos requisitos definidos no *backlog* do *sprint*, existem diariamente as *daily scrums* que são reuniões que duram no máximo 15 minutos em que se respondem a 3 perguntas:

- O que fiz ontem?
- O que vou fazer hoje?
- Tenho algum impedimento?

Qualquer questão a ser tratada que não seja relacionada com as 3 perguntas acima, será sempre tratada após a *daily scrum*.

No final do *sprint* existe a revisão do mesmo, onde se inspeciona o que foi feito e se adapta, se necessário, o *backlog* do produto. Posteriormente, dá-se a retrospetiva do *sprint* na qual se discute o que correu bem, o que correu mal e onde se pode melhorar.

2.3.3 Metodologia ágil da OutSystems

A OutSystems sugere uma abordagem diferente para organização e controlo de projetos, baseando-se nas metodologias ágeis. Desta forma, a necessidade de entrega rápida e constante mudança é atendida, entregando assim aplicações que realmente correspondem às necessidades de negócio.

A metodologia ágil da OutSystems assemelha-se à metodologia *Scrum*, abordada acima. Usando os principais conceitos da metodologia *Scrum*, a OutSystems adapta-os de forma a serem adequados à natureza particular dos projetos feitos na plataforma.

Nesta metodologia também existem *Sprints* em que, no final, há uma demonstração da versão atual da aplicação já desenvolvida, para que de seguida se reúna *feedback* e se negoceie os requisitos que vão ser desenvolvidos no *sprint* seguinte. Os *Sprints* têm a duração de 1 a 2 semanas e incluem as fases de análise de requisitos, desenvolvimento e testes.

O facto de os utilizadores do negócio poderem testar e trabalhar com as versões finais de cada *Sprint* aumenta a qualidade do *feedback* dos mesmos. Os requisitos que não estejam bem claros são esclarecidos assim que os utilizadores do negócio percebam o impacto que os mesmos têm em relação ao que é desejado no produto final. Assim, de seguida, os requisitos de menor impacto são puxados para baixo na lista de prioridades.

A OutSystems dá enfase á relação tempo/esforço e requisitos entregues. Limitando o período do projeto, é colocado um foco especial na organização dos recursos, garantindo a entrega no prazo definido com o orçamento definido.

Na metodologia ágil da OutSystems, o *training*, por exemplo, é dado antes da entrega final da aplicação. Através desse *training*, é possível recolher uma grande quantidade de *feedback* e realizar alterações, que podem ser cruciais, antes da entrega final (OutSystems, 2019).

2.4 Comparação entre OutSystems e outras plataformas *low-code*

2.4.1 Plataforma de desenvolvimento *low-code*

Plataformas de desenvolvimento *low-code* permitem a criação de aplicações de negócio com entrega contínua e rápida ao cliente e com o mínimo de programação, através de interfaces gráficas e componentes direcionados para as mesmas, aos quais apenas é necessário fazer *"drag and drop"*.

É importante ainda mencionar que as plataformas *low-code* são diferentes das plataformas *no-code*. Com efeito, as plataformas *no-code* são usadas por utilizadores de negócio sem conhecimento de programação, o que lhes permite resolver alguns problemas do dia-a-dia. As plataformas *low-code*, pelo contrário, já exigem conhecimentos de programação (Rayome, 2018). Em adição, produtos construídos em plataformas *low-code* permitem uma relativa personalização, que vai mais além do que o simples *drag and drop*, embora menos do que um ambiente de desenvolvimento completamente integrado, pois as mesmas não providenciam

ferramentas para a criação de produtos de raíz (Best Low-Code Development Platforms Software, 2019).

Percebe-se que apesar destas plataformas ainda serem pouco usadas, o mercado está a crescer rápido para as mesmas (Rayome, 2018), como indicado na secção 2.4.2.

2.4.2 Porque estão as plataformas *low-code* a crescer?

As plataformas *low-code* devem o seu crescimento às circunstâncias tecnológicas da contemporaneidade, em especial à progressiva valorização da experiência do utilizador face às funções disponibilizadas, tal como se pode verificar no seguinte elemento textual:

A popularidade que a programação *low-code* atingiu está relacionada com os desafios que as empresas têm de ultrapassar hoje em dia. Numa época em que indústrias que já existem a muito tempo estão a ser ultrapassadas por *startups*, a experiência digital do cliente é muito mais importante do que providenciar uma lista de *features*. É previsível um aumento da necessidade de novos tipos de experiência digital (OutSystems, Low-Code Platforms, 2019).

Enquanto que o desenvolvimento em ambiente que não seja *low-code* é consideravelmente mais lento, o desenvolvimento *low-code* fornece uma forma de as empresas superarem obstáculos nomeadamente, *backlogs* que não são reduzidos, verificando-se consequentemente um crescimento progressivo nas plataformas *low-code* enquanto mantêm uma forma estável de gestão de recursos, como se pode verificar na Figura 3.

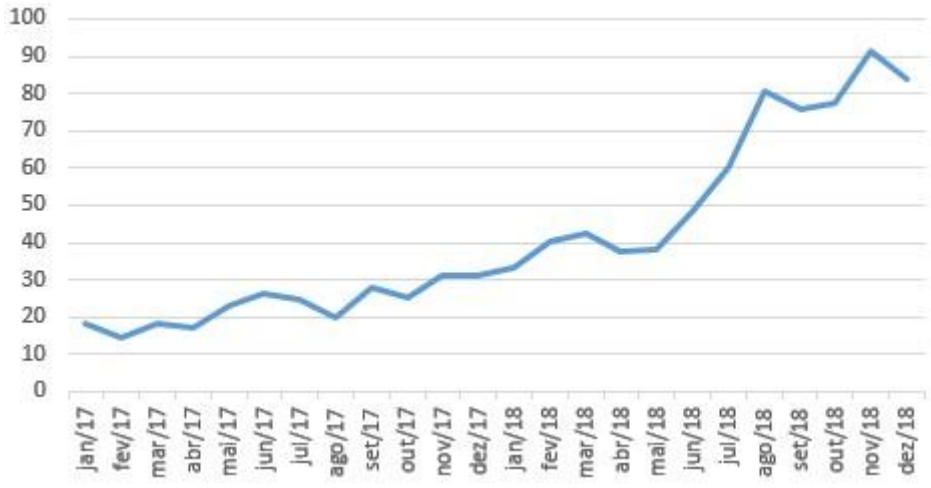

Figura 3 - Interesse no desenvolvimento de plataformas low-code, segundo a Google Trends

2.4.3 O que procurar numa plataforma *low-code*?

Identificam-se pontos centrais nas plataformas de *low-code*, dos quais: o fornecimento de modelos visuais, enquanto se permite simultaneamente uma customização; a fácil implementação e manutenção de aplicações; e a escalabilidade e segurança que as plataformas conferem (5 Best Low-Code Development Tools That Take Easy to the Next Level., 2019).

No entanto, para que se possa implementar um desenvolvimento *low-code*, um produto precisa de ter certas componentes, das quais: gerar código fonte como base para customização, permitir que os programadores personalizem a marcação HTML e o código fonte; integração com bases de dados, *web services* ou *APIs* para ligar dados (Best Low-Code Development Platforms Software, 2019)

2.4.4 Plataformas de *low-code* existentes e mais populares

São duas as plataformas exclusivamente de *low-code* mais fortes no mercado, nomeadamente a OutSystems e a Mendix. Enquanto que a Mendix tem mais presença no mercado, a OutSystems consegue ter uma maior oferta e uma estratégia mais forte. Pode-se verificar o mesmo na Figura 4:

Figura 4 - Comparação das plataformas low-code, segundo a Forrester (Rymer, Koplowitz, Mines, Sjoblom, & Turley, 2019)

Poder-se-á constatar a mesma predominância das duas plataformas no que diz respeito a aplicações móveis, tal como indicado na Figura 5.

Figura 5 - Comparação de plataformas low-code para mobile, segundo a Forrester (Mines, Hammond, Sjoblom, Allison, & Reese, 2017)

Já a Gartner faz uma comparação mais específica, nomeadamente relativamente á capacidade do produto; contracto e avaliação; integração e implementação; e serviço e suporte como se pode ver na Tabela 1:

Tabela 1 - Comparação de duas das mais populares plataformas low-code, segundo a Gartner (Comparing OutSystems, Mendix, 2019)

	OutSystems	Mendix
Geral	4.5 (429 *reviews*)	4.5 (44 *reviews*)
Capacidades do Produto	4.5	4.4
Contracto e Avaliação	4.3	4.1
Integração e Implementação	4.5	4.4
Serviço e Suporte	4.4	4.4

Contudo, tendo em conta as posições das plataformas segundo ambas as entidades, parece seguro afirmar que a OutSystems goza de um posicionamento positivo relativamente ao que nos parece ser a sua concorrente direta, não apenas em termos gerais (onde embora tenham tido ambas 4.5 pontos, a OutSystems recebe 385 *reviews* a mais do que a Mendix); como relativamente aos restantes pontos de avaliação.

Em conclusão, tal como se apresenta nas figuras e na tabela anteriores, a OutSystems é líder de mercado, tanto em termos de plataforma como no que diz respeito á sua presença no mercado (estratégia e oferta).

Capítulo 3 - Desenvolvimento da aplicação

Neste capítulo, realiza-se o desenho e implementação da aplicação recorrendo à análise de requisitos, ao desenho de diagramas, , à arquitetura, aos esboços da aplicação, ao calendário, ao layout final e aos testes feitos à aplicação.

3.1 Desenvolver ou comprar o produto?

Esta questão tem vários fatores influenciadores, nomeadamente, por exemplo, a empresa a quem se compra o produto; os elementos e recursos que se acham necessários para a aplicação, entre outras questões contextuais. Consequentemente, não existe consenso ou resposta conclusiva relativamente a esta questão.

Após uma análise de mercado verificou-se que comprar uma licença de aplicação já existente teria um custo considerável em termos monetários. Assim sendo, é necessário que se faça um levantamento do objetivo da empresa em questão.

Se o objetivo for, por exemplo, apenas um produto final sem nada a acrescentar no futuro, geralmente comprar o produto a uma empresa seria mais vantajoso em termos económicos e temporais. No entanto, seguindo o mesmo exemplo, se vier a ser, porventura, necessário estender a aplicação ou desenvolver novas funcionalidades, acrescentar-se-ia um custo enorme ao projeto, pelo que seria melhor optar por desenvolver.

No que diz respeito á aplicação a desenvolver neste projeto, para além de já existir uma aplicação web dentro da própria empresa e recursos com conhecimento sobre os processos e a aplicação web desenvolvida também em OutSystems, esta tecnologia permite também a reutilização de código o que diminui o tempo despendido no desenvolvimento. Assim, pode-se argumentar, não só será mais fácil a implementação da aplicação móvel proposta, como também mais propício em termos de custo de expansão da aplicação, se necessário, uma vez que adquirir uma licença para uma aplicação já existente poderá constituir um custo que a sua implementação irá demorar a compensar.

3.2 Análise de Requisitos

Foram investigadas as necessidades dos consultores/gestores no contexto de submissão/revisão de despesas através de questionários qualitativos com uma amostra de 47 pessoas da equipa de

Platform Services (38 consultores e 9 gestores), de forma a perceber as dificuldades das pessoas que usam a aplicação web e se gostariam ou não de ter uma aplicação móvel.

Em ambos os casos foram colocadas 2 perguntas:

1. Gostaria de ter à tua disposição a submissão/revisão de despesas através de uma aplicação móvel criada para o efeito? – Pergunta de resposta fechada em que o objetivo será apenas perceber se faz sentido a realização de uma aplicação móvel para submeter/rever despesas ou não.
2. Se respondeu sim, porquê? Quais são as dificuldades que sente ao efetuar a submissão/revisão pelo canal atual (aplicação web)? – Pergunta de resposta aberta de forma a perceber quais as maiores dificuldades reportadas e que poderão ser minoradas/resolvidas com a realização de uma aplicação móvel.

Tanto no caso dos consultores como dos gestores, a maioria das pessoas respondeu que queria ter ao seu dispor uma aplicação mobile para submeter/rever despesas, como se pode ver na Figura 6.

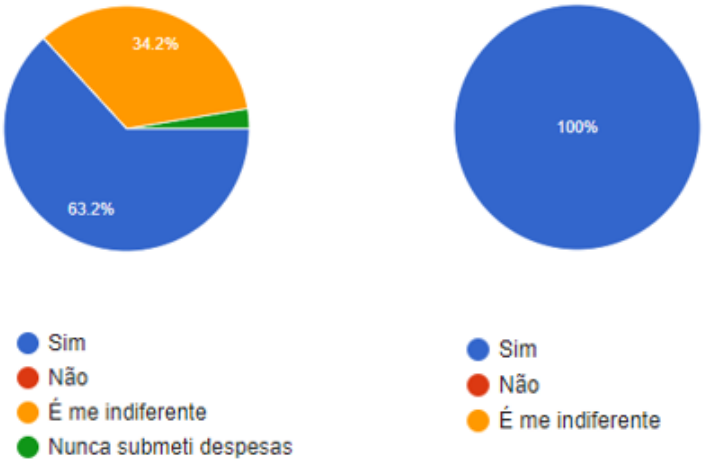

Figura 6 - Estatística da resposta à primeira pergunta por parte dos consultores (esquerda) e gestores (direita)

Concluiu-se na segunda pergunta que uma aplicação móvel seria, por exemplo, mais fácil de acesso e manuseamento, podendo assim também anexar as faturas em formato digital em vez de as entregar em papel, como se reflete nas respostas abaixo:

- “falta de mobilidade. Não poder aprovar as despesas a qualquer momento. Tendo prazos a cumprir é importante validar e aprovar assim que estão para aprovação.”
- “necessidade de aprovar despesas quando estou longe do computador.”
- “ter de enviar um email com as despesas e relacioná-lo às minhas despesas e depois aguardar uma resposta manual... dá ao processo uma ampla possibilidade de falhar. Se as despesas forem digitalizadas e associadas logo a mim e ao meu *manager* responsável por via de uma *app* e respetivamente se abrir logo um processo de aprovação, seria o ideal.”
- “Quando estou ausente do escritório, pela *app mobile* poderei submeter a qualquer momento.”
- “À medida que recebia as faturas começava logo a introduzir. Evitava de juntar tudo.”
- “Aprovações são feitas *last minute*, e no caso de as despesas não serem aprovadas resta pouco tempo para conseguir fazer a correção, submeter novamente e o *manager* aprovar. (as notificações/alertas tanto para recordar o utilizador para submeter as suas despesas antecipadamente como para o *manager* as aprovar poderão ajudar).

  - Necessidade de entregar folha de despesas e as faturas/recibos pessoalmente. Pelo menos as folhas de despesas poderiam ser enviadas automaticamente (i.e., PDF gerado, enviado e depois guardado na aplicação) já que por questões legais talvez não seja possível deixar de entregar os recibos em suporte físico.

  - Difícil saber qual o estado de aprovação das despesas atualmente. (Notificações de mudança de estado de aprovação das despesas?). *Nice to have*: Tirar fotos de cada recibo via *app* no processo de criação da despesa (ficando associado à despesa).”
- “Quero adicionar a despesa na hora com o telemóvel e tirar uma foto como comprovativo.”

Com base nos inquéritos feitos e no que foi pedido pela empresa, foram criados requisitos de forma a criar uma aplicação móvel com capacidade para trabalhar *offline* que permitisse as seguintes funcionalidades: submeter despesas com possibilidade de adicionar fotografia; rever despesas e receber notificações quando uma despesa é submetida/revista.

Ao pedido, foram adicionadas algumas funcionalidades que poderiam trazer mais valias ao utilizador, como por exemplo: aceder à aplicação por *Touch/Face Id* e ver os dados pessoais.

Assim, detalhou-se o pedido em requisitos:

- Como consultor, quero ver as despesas que eu criei/submeti de forma a controlar as mesmas;
- Como consultor, quero criar/editar uma despesa para a processar;
- Como consultor, quero poder anexar um ficheiro à despesa de forma a processá-lo;
- Como gestor, quero aceder à lista de despesas que tenho para rever de forma a processá-las;
- Como gestor, quero aceder ao detalhe de uma despesa e revê-la de forma a processá-la;
- Como gestor, quero aceder à lista de despesas que já revi para poder acompanhá-las;
- Como gestor, quero aceder ao detalhe de uma despesa que já revi para poder acompanhá-la;
- Como utilizador, quero poder ver a minha informação pessoal de forma a ter conhecimento da mesma;
- Como gestor, quero receber uma notificação quando recebo uma despesa para rever;
- Como consultor, quero receber uma notificação quando uma despesa que eu submeti é revista;
- Como utilizador, quero poder entrar na aplicação por impressão digital ou reconhecimento facial de forma a tornar o processo de autenticação mais fácil e seguro;
- Como gestor, quero ter um *dashboard* de forma a acompanhar o que está revisto e o que falta rever;
- Como consultor, quero ter um *dashboard* de forma a acompanhar quanto é que já submeti e quanto falta submeter;
- Como utilizador, quero poder aceder à aplicação quando estiver *offline* de forma a agilizar o processo.

3.3 Desenho

3.3.1 Diagrama de Casos de uso

A Figura 7 apresenta o diagrama de casos de uso da aplicação proposta.

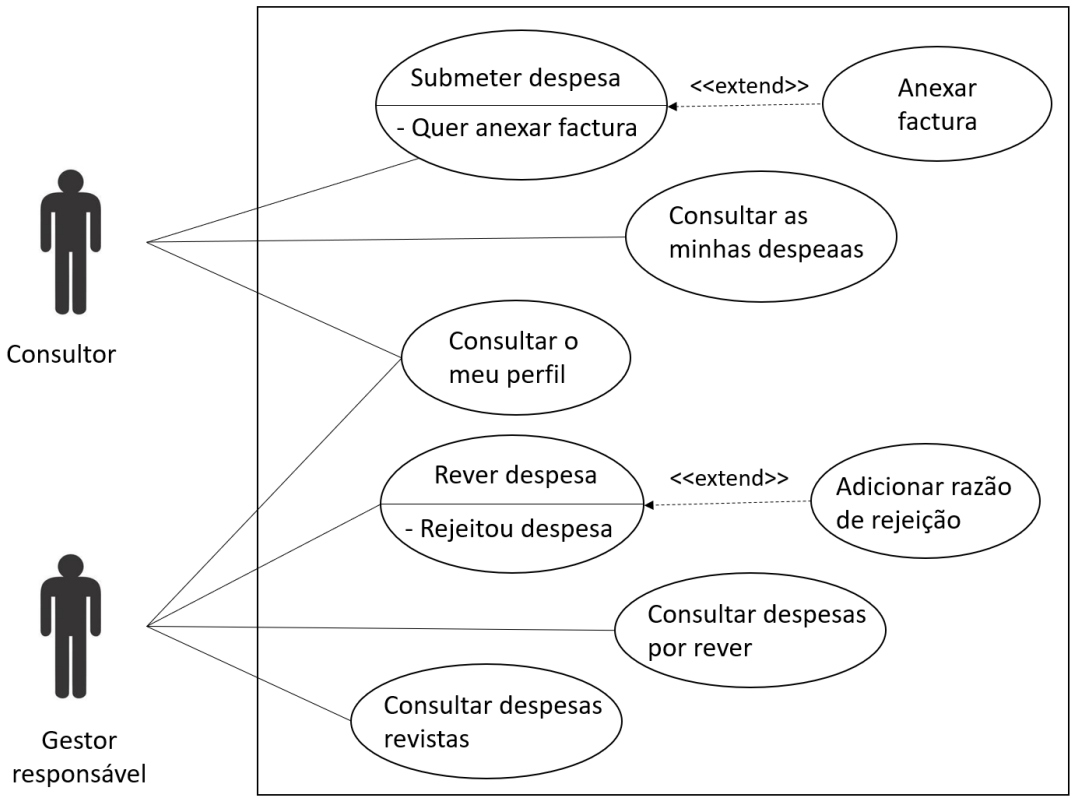

Figura 7 - Diagrama de casos de uso

Atores

Consultor – O ator que irá submeter a despesa.

Gestor – O ator que irá rever a despesa.

Casos de Uso

Submeter despesa – Na submissão de uma despesa, o consultor preenche os campos do formulário. Só após carregar no botão de submeter e os campos obrigatórios estarem preenchidos é que a despesa é submetida.

Anexar fatura – Caso o consultor queira anexar uma fatura ao submeter uma despesa, pode fazê-lo.

Consultar as minhas despesas – O consultor pode consultar a lista de todas as despesas criadas por ele

Consultar o meu perfil – Qualquer utilizador pode consultar os seus dados pessoais presentes na aplicação

Rever despesa – O gestor responsável pode rever uma despesa, aprovando ou rejeitando.

Adicionar razão de rejeição – Caso o gestor responsável tenha rejeitado a despesa, obrigatoriamente terá de adicionar uma razão de rejeição.

Consultar despesas por rever – O gestor responsável tem a possibilidade de consultar a lista de todas as despesas que tem por rever

Consultar despesas revistas – O gestor responsável tem a possibilidade de consultar a lista de todas as despesas que foram revistas por ele

3.3.2 Diagrama de Atividades

A Figura 8 apresenta o diagrama de atividades da aplicação proposta.

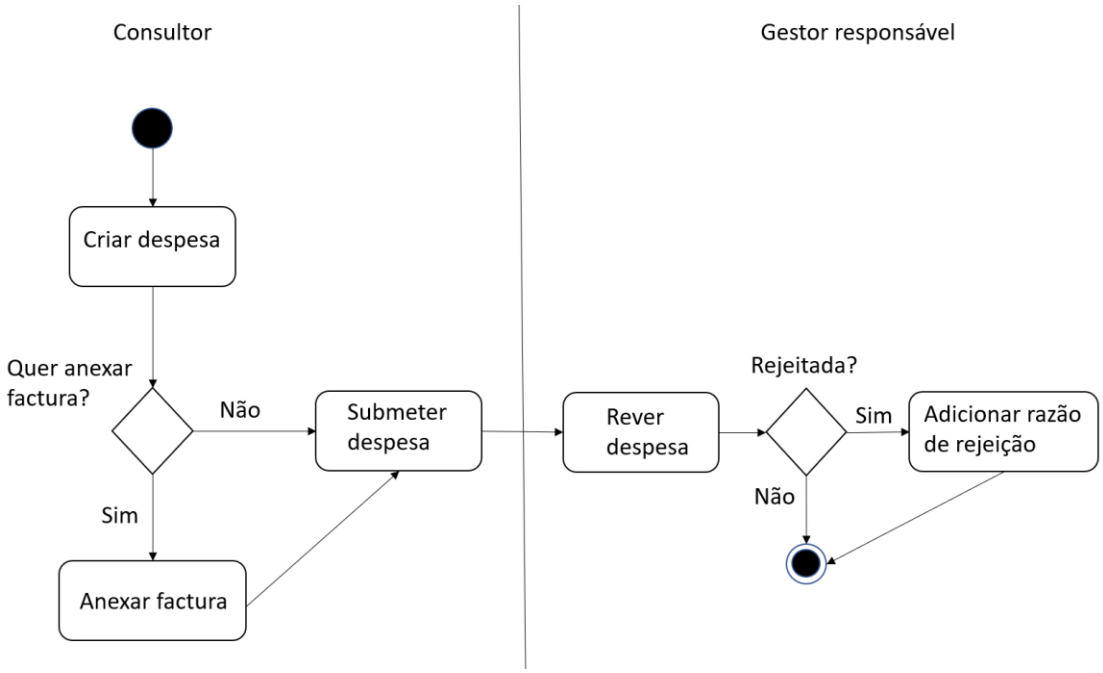

Figura 8 - Diagrama de Atividades

O processo começa pelo consultor que é responsável por criar a despesa. Pode, se quiser, anexar uma fatura e posteriormente submeter a despesa. Posteriormente, o gestor responsável revê a despesa e se a rejeitar terá de adicionar uma razão de rejeição.

Atores

Consultor – O ator que irá submeter a despesa

Gestor responsável – O ator que irá rever a despesa

Atividades

Criar despesa – O consultor cria a despesa, de acordo com os campos obrigatórios.

Anexar fatura – O consultor pode ou não anexar uma fatura, podendo tirar uma foto ou escolher dos seus ficheiros.

Submeter despesa – O consultor submete a despesa.

Rever despesa – O gestor responsável pode rever a despesa, rejeitando ou aprovando.

Adicionar razão de rejeição – No caso de a despesa ser rejeitada, é necessário adicionar uma razão para se rejeitar a despesa.

3.3.3 Diagrama de Processos

Existem 2 processos: o de submeter e o de rever uma despesa, sendo os mesmos realizados pelo consultor e pelo gestor responsável, respetivamente.

Submeter despesa

Para a submissão de uma despesa (Figura 9) é necessário criá-la preenchendo os campos necessários, bem como anexar um ficheiro (se aplicável) para, posteriormente, submeter efetivamente a despesa. É despoletado, então, o envio de uma notificação para a pessoa responsável por rever a despesa.

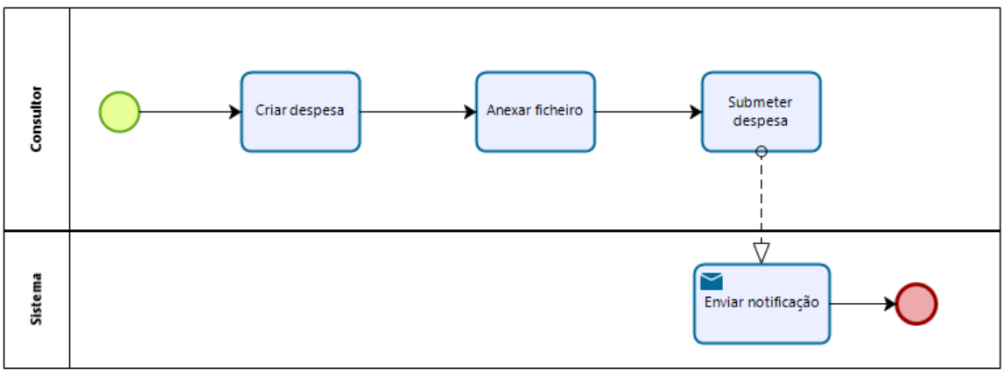

Figura 9 - Diagrama do processo de submissão de despesa

Rever despesa

No processo de revisão de despesas (Figura 10), o gestor responsável pode ver a lista de despesas por rever, entrando no detalhe ou não. Assim que a despesa é revista, uma notificação é despoletada e recebida no telemóvel da pessoa que submeteu a despesa.

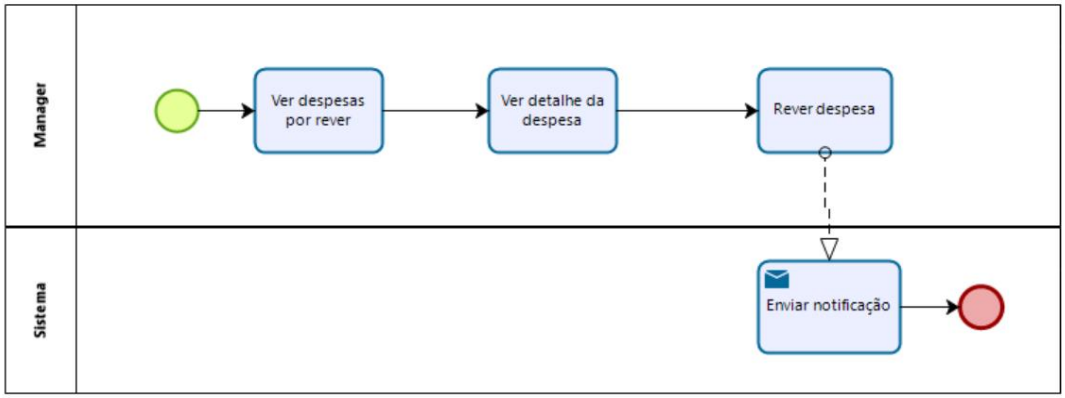

Figura 10 - Diagrama do processo de revisão de despesa

3.3.4 Arquitetura

Arquitetura da OutSystems e do projeto

Em OutSystems um projeto é desenhado com base numa arquitetura e é constituído por *eSpaces* que têm módulos. De seguida, irá ser abordada a arquitetura em OutSystems e, posteriormente, a proposta da arquitetura para este projeto.

Arquitetura OutSystems

Servidor da Plataforma

O Servidor da plataforma OutSystems é o núcleo da plataforma. É um componente de servidor que complementa uma grande quantidade de aplicações web padrão com um conjunto de serviços. Todas as aplicações dependem de arquiteturas e estruturas padrão. Isto acontece porque a OutSystems encarrega-se de todas as etapas necessárias para gerar, otimizar, compilar e implementar aplicações num servidor padrão de aplicações web usando um conjunto de serviços especializados, que irão ser explorados adiante (OutSystems, OutSystems Architecture, 2019).

Gerador de código

Esta ferramenta usa o modelo da aplicação que foi desenvolvido no editor visual e gera todos os componentes nativos da aplicação, prontos para serem implementados num servidor de aplicações (OutSystems, OutSystems Architecture, 2019).

Serviços de implementação

Implementa os componentes de aplicações geradas num servidor padrão de aplicações web, assegurando que a aplicação é consistentemente instalada em cada servidor de *front-end* do conjunto de servidores de uma organização (OutSystems, OutSystems Architecture, 2019).

Serviços Aplicacionais

Esta ferramenta gere a execução de tarefas agendadas e providencia serviços de registo assíncronos que guarda eventos como erros, *audits* e métricas de desempenho (OutSystems, OutSystems Architecture, 2019).

Proposta de arquitetura do Projeto

4 *Layer Canvas*

A OutSystems tem uma ferramenta de arquitetura de forma a simplificar o desenho de arquiteturas orientadas a serviços (SOA), promovendo a abstração correta de serviços reutilizáveis e o isolamento correto de módulos funcionais distintos, no caso em que se está a desenvolver e manter várias aplicações que reutilizem módulos comuns.

Essa ferramenta designa-se de *4 Layer Canvas* (Figura 11) em que cada camada representa uma natureza diferente da funcionalidade a ser usada num módulo.

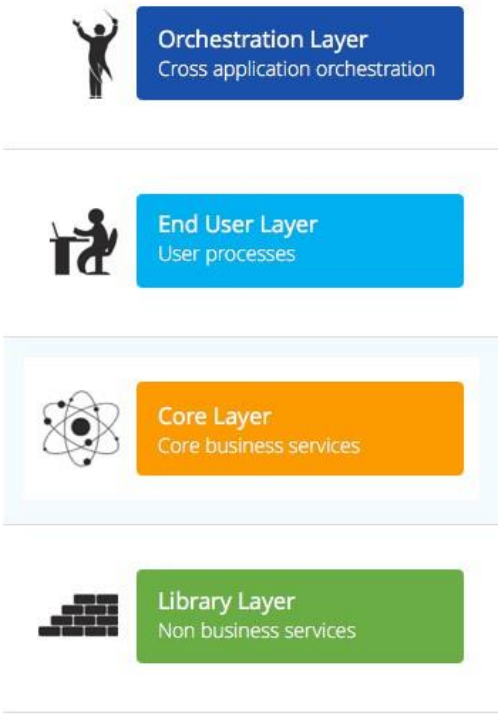

Figura 11 - 4 Layer Canvas (OutSystems, The 4 Layer Canvas, 2019)

A primeira camada – *Library Layer* – é usada para serviços independentes de negócios de forma a estender a estrutura com recursos altamente reutilizáveis, Padrões de UI, conectores a sistemas externos e integração de código nativo.

A segunda camada – *Core Layer* – é onde se constrói o modelo de dados e todas as regras de negócio que irão ser usadas na *End User Layer*.

A terceira camada- *End User Layer* – é onde se situa o UI e se pode ver as *User Stories* implementadas, reutilizando os serviços das primeira e segunda camadas.

Por fim, a quarta camada – *Orchestration Layer* – contém os processos, *dashboards* e páginas iniciais, juntando informação de diferentes aplicações de forma a fornecer ao utilizador uma experiência unificada (OutSystems, The 4 Layer Canvas, 2019).

Microserviços

Este projeto irá usar microserviços. A arquitetura do mesmo será semelhante à arquitetura *4 Layer Canvas*. No entanto, cada serviço é dissociado e tem a sua própria infraestrutura. A comunicação com e entre os serviços é apenas feito através de um mecanismo de ligação, tal como o REST API (OutSystems, Microservices Architecture in OutSystems, 2019).

Ou seja, há uma camada entre a *Core Layer e a End-User Layer*, tal como a Figura 12 indica.

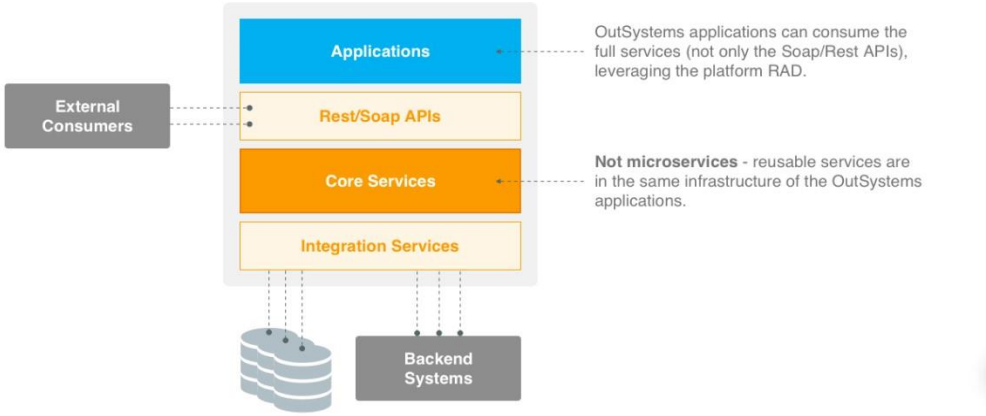

Figura 12 - Arquitetura de uma aplicação usando microserviços (OutSystems, Microservices Architecture in OutSystems, 2019)

Neste projeto, a arquitetura é constituída por 4 camadas (Figura 13):

A *Library Layer* é a primeira camada que contém serviços externos como o Azure Blob Storage (Azure Services) e a OneSignal API (Notification Services), que servem, respetivamente, para guardar ficheiros no servidor e para enviar *push notifications*.

Existe depois a *Core Layer* já mencionada em cima e depois a camada onde são expostos os microserviços a serem consumidos pelas aplicações. Por enquanto, será só a aplicação deste projeto a consumir os serviços expostos na camada do módulo Expenses Business Logic.

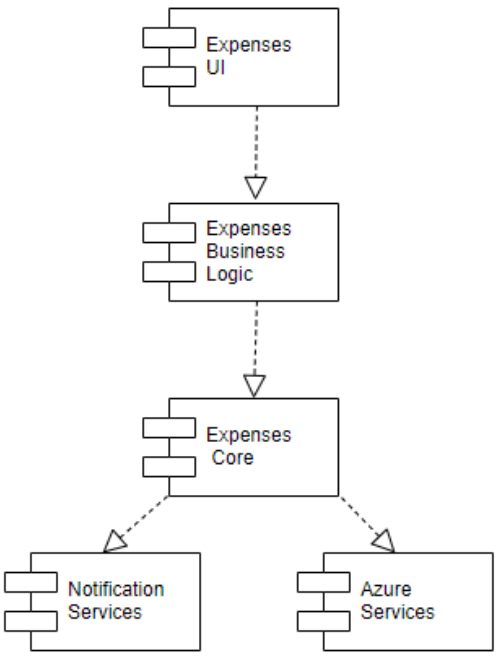

Figura 13 – Diagrama de componentes

3.3.5 Esboços da aplicação

Foram desenhados esboços, usando a aplicação Balsamiq[1], de acordo com os requisitos pedidos, de forma a fazer uma pré-visualização para se perceber como iria ficar o *layout* final.

Consultor

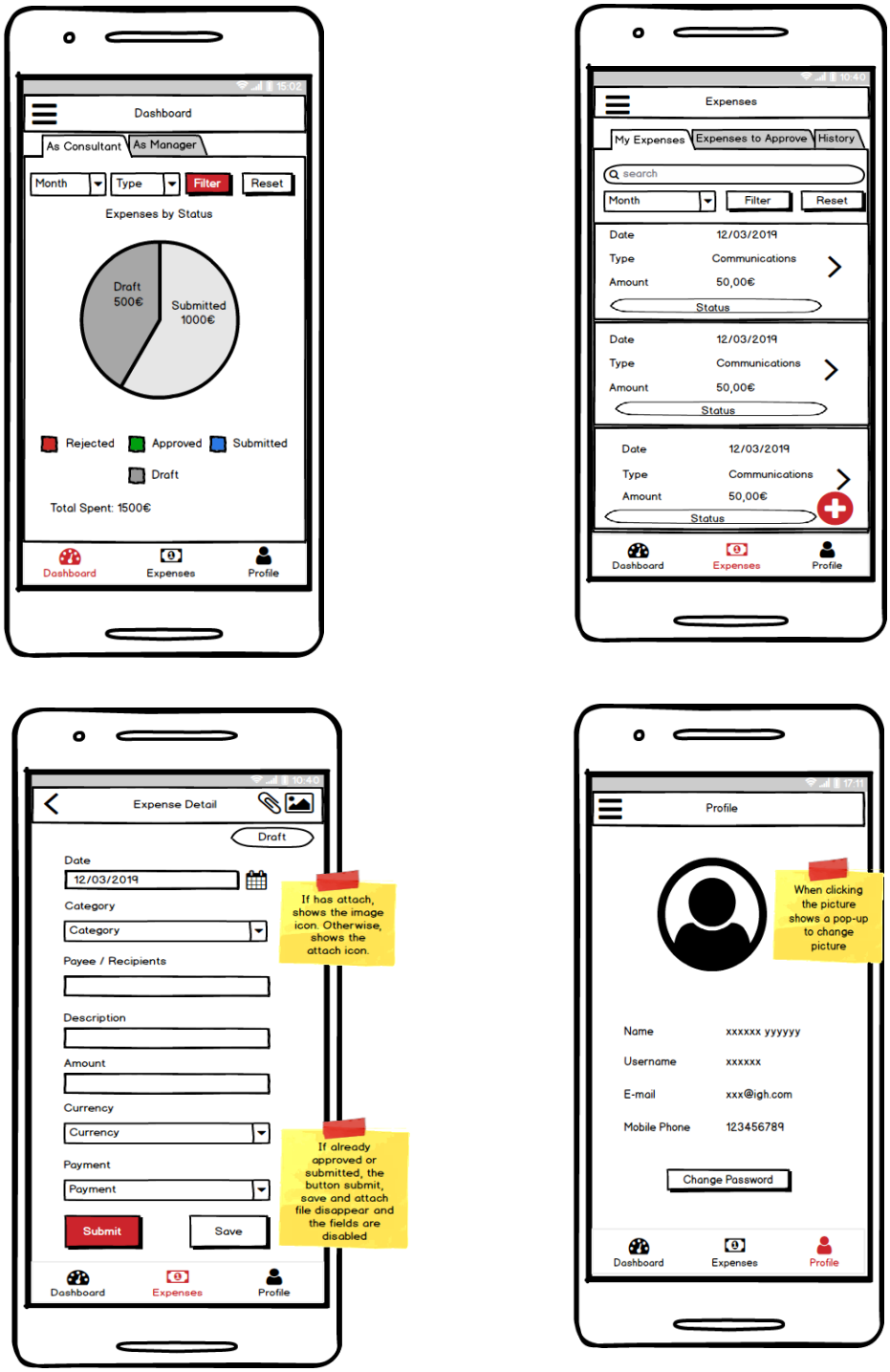

Figura 14 - Esboços da vista do consultor

[1] https://balsamiq.com/

Gestor responsável

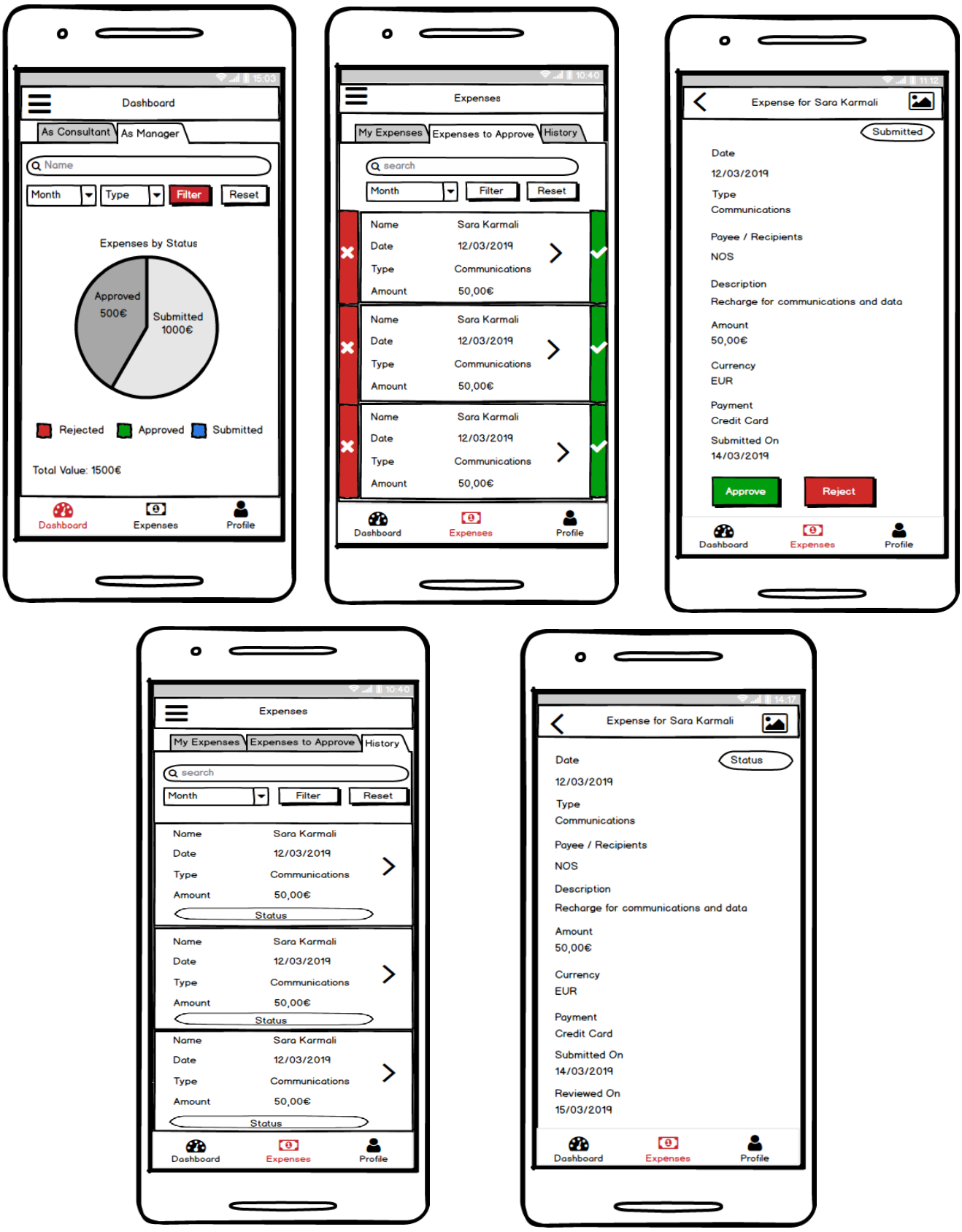

Figura 15 - Esboços da vista do gestor

3.4 Implementação

3.4.1 Calendário

Este projeto foi constituído por 3 *sprints* e um período inicial de elaboração do documento de visão, como se pode ver na Figura 16.

A definição do tempo de cada *sprint* depende do tempo do projeto e do cliente. O mais comum na metodologia *Scrum* é um *sprint* de 2 semanas, podendo haver um frequente *feedback* do cliente. Dado a quantidade de requisitos e a necessidade de constante melhoria garantindo a máxima qualidade, foram definidos *sprints* de 2 semanas excetuando no 3º *sprint* em que não era necessário tanto tempo para corrigir os erros reportados e melhorar o *layout* da aplicação.

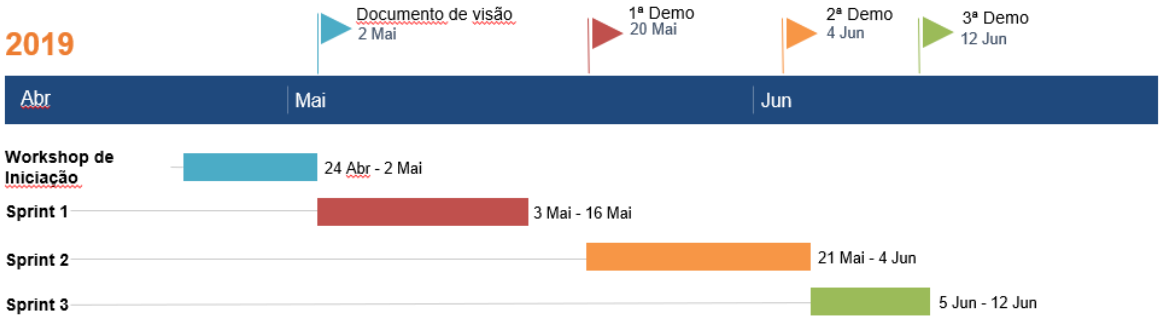

Figura 16 - Desenho do calendário do projeto

O documento de visão consiste num documento onde são descritos: o contexto do projeto, as *personas* e os processos. É também realizado o desenho dos esboços de acordo com os requisitos funcionais.

No *Sprint* 1, foi construída a aplicação sendo implementadas as funcionalidades de submissão e revisão de despesas incluindo a consulta de listas, embora funcionando exclusivamente *online*.

No *Sprint* 2, foram implementadas as funcionalidades *offline*, a funcionalidade das *push notifications*, a funcionalidade do *touch/face id* e a funcionalidade que permite consultar os dados pessoais.

No *Sprint* 3 foram corrigidos erros reportados e melhorado o *layout* da aplicação, incluindo *dashboards* para a *homepage* de forma a tornar a aplicação mais apelativa e *user-friendly*.

3.4.2 *Layout* da aplicação

Consultor

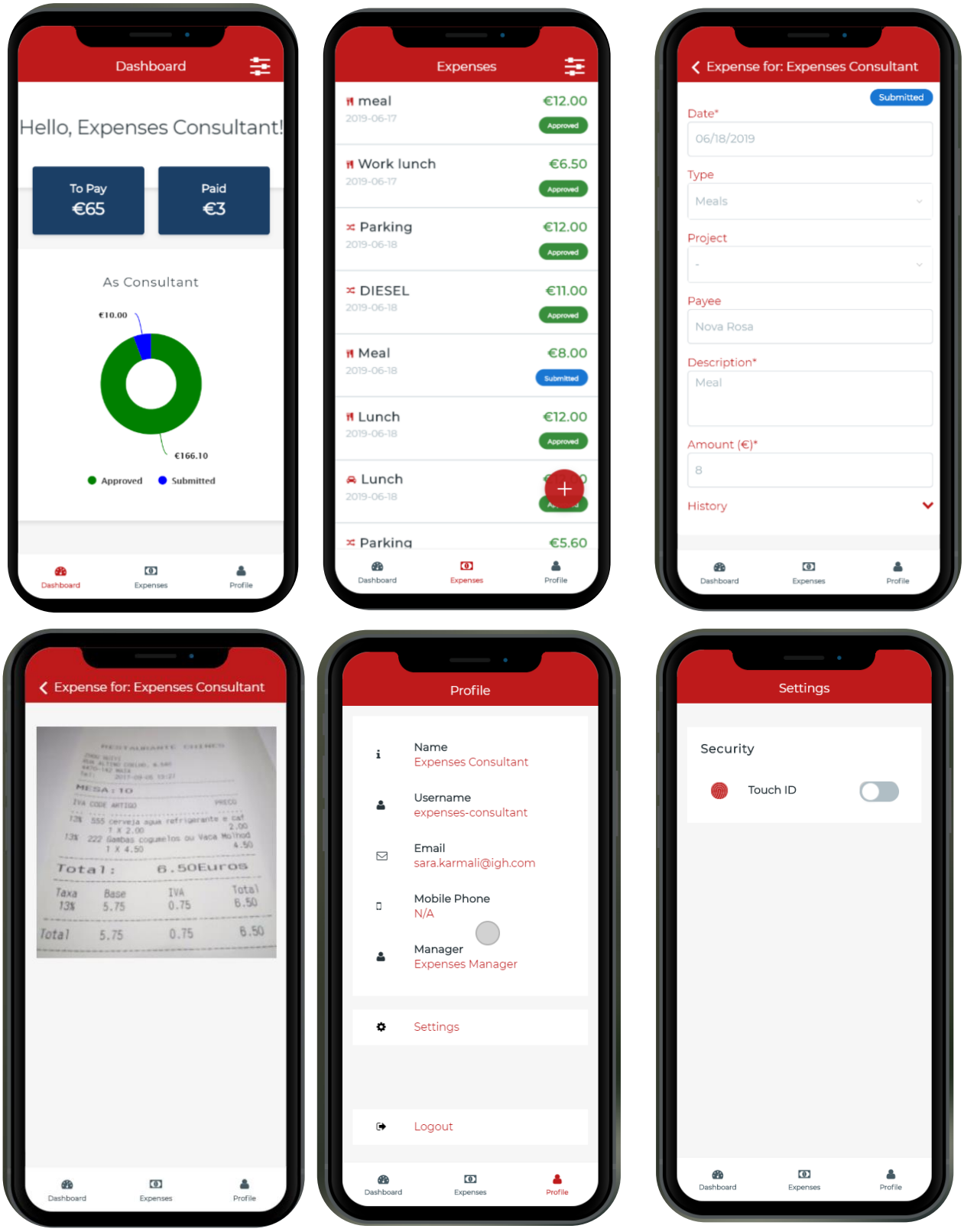

Figura 17 - Layout final pela vista do consultor

Gestor responsável

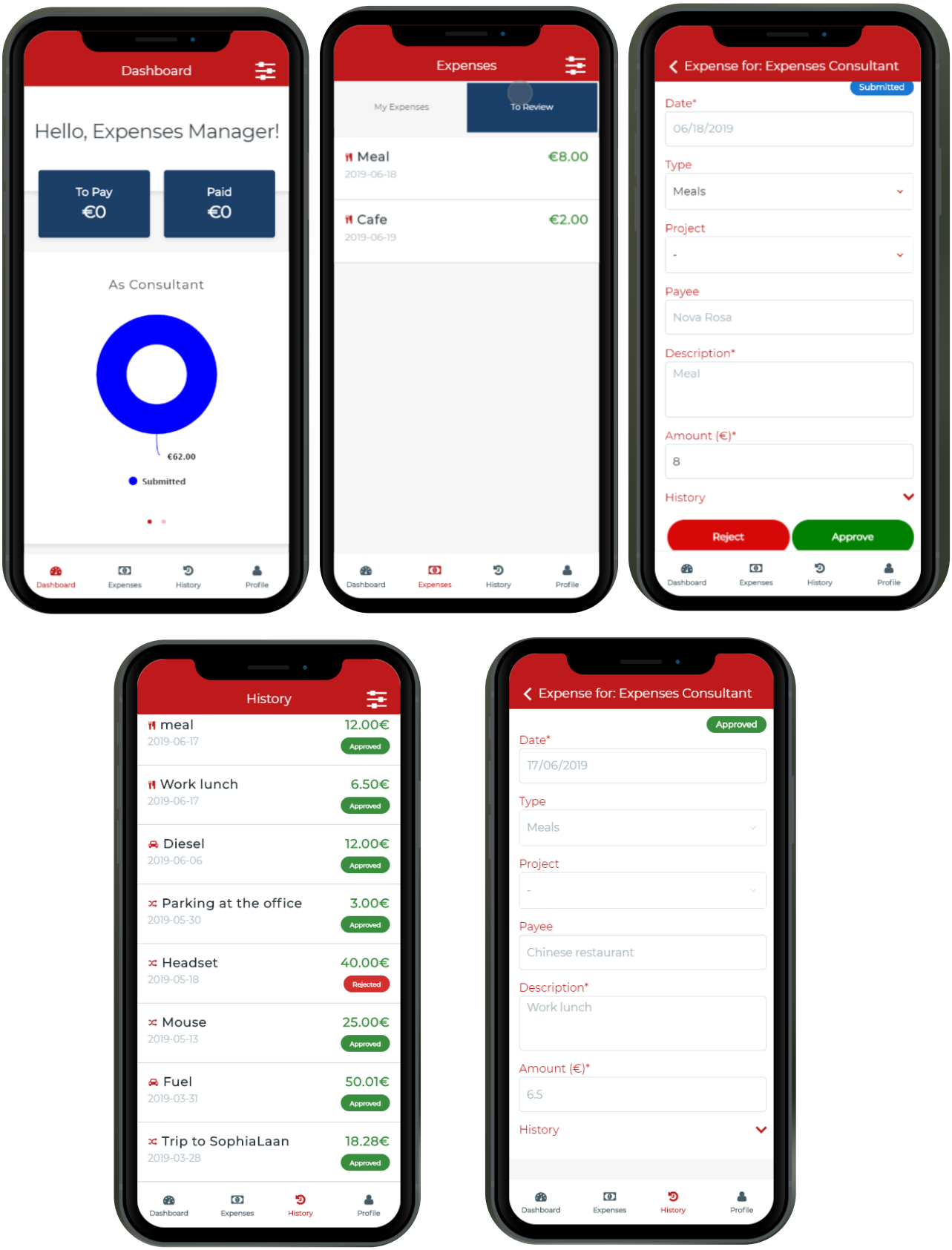

Figura 18 - Layout final pela vista de gestor

3.4.3 Boas práticas de OutSystems

Ao desenvolver a aplicação foram usadas as boas práticas de OutSystems de forma a tornar o código limpo e claro.

Abaixo alguns exemplos de boas práticas usadas:

- Usar o inglês para o código e comentários e evitar *labels* e descrições vazias (OutSystems, OutSystems Platform Best Practices, 2019)

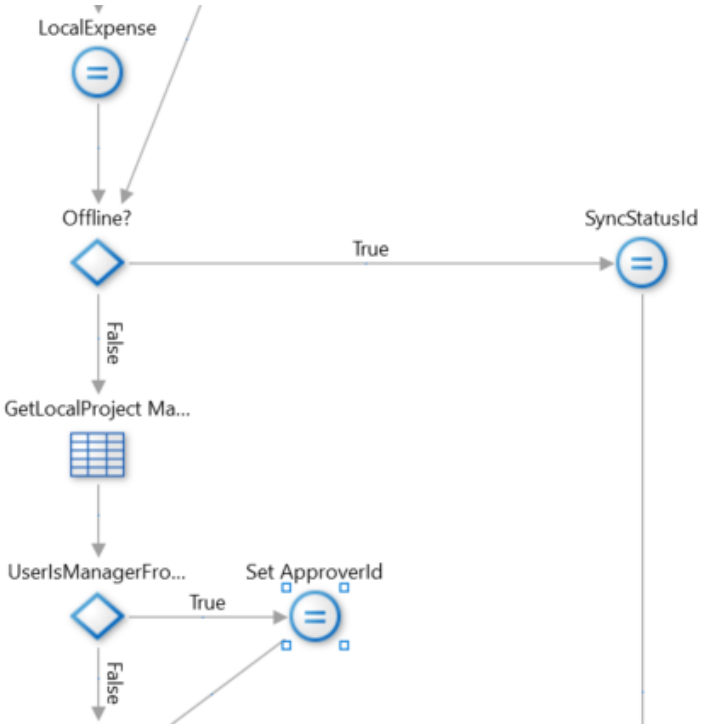

Figura 19 - Código em inglês e labels preenchidas

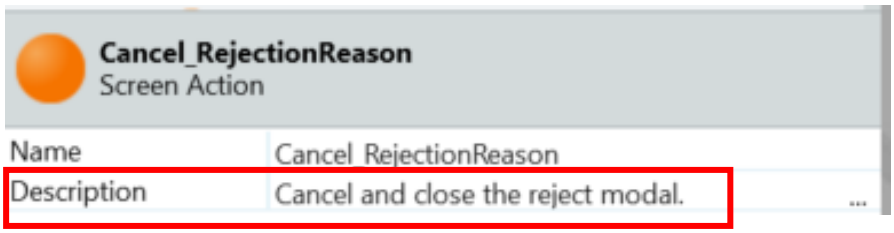

Figura 20 - Descrição preenchida

- Nomes que tenham significado (ex.: “Customer” em vez de “Cstr”)

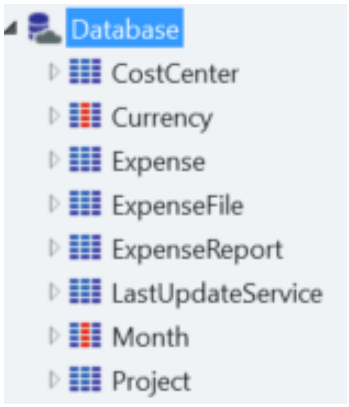

Figura 21 - Entidades com nomes que têm significado

- Adicionar o sufixo “Id” às chaves estrangeiras (ex.: “CustomerId”)

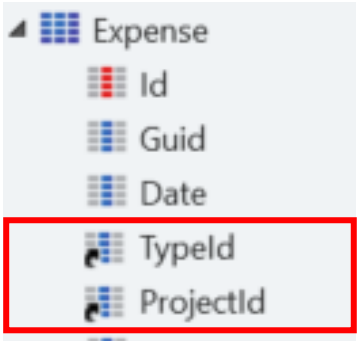

Figura 22 - Chaves estrangeiras com o sufixo "Id"

- Incluir o nome da entidade no nome do registo (ex.: “Costumer” em vez de “record”)

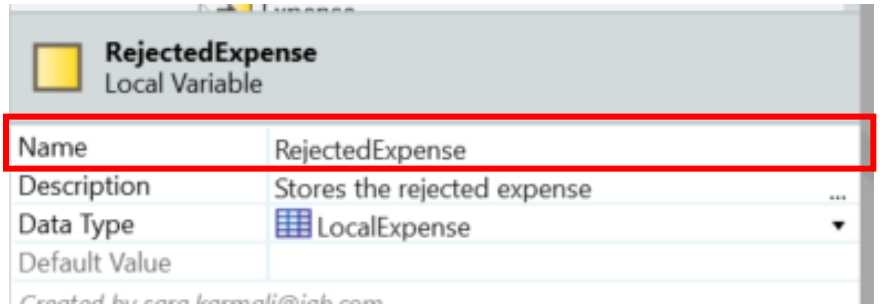

Figura 23 - Nome da entidade incluído no registo

- Ecrãs com prefixo (ex.: “Customer_Edit”, “Customer_Show”)

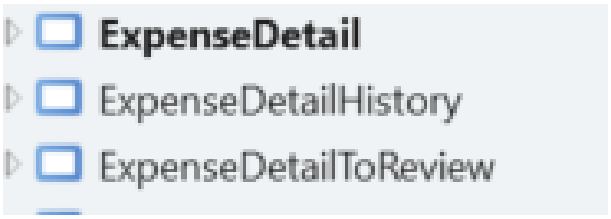

Figura 24 - Ecrã com prefixo

- Definir o nome dos ShowRecords, EditRecords e TableRecords

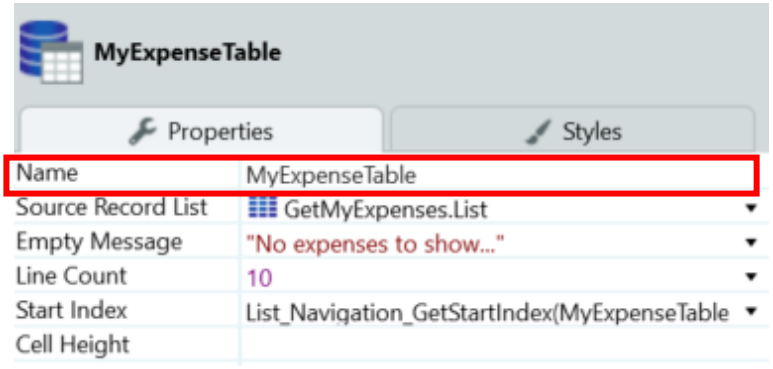

Figura 25 - Nome da TableRecord definido

- Definir o texto de exemplo das expressões

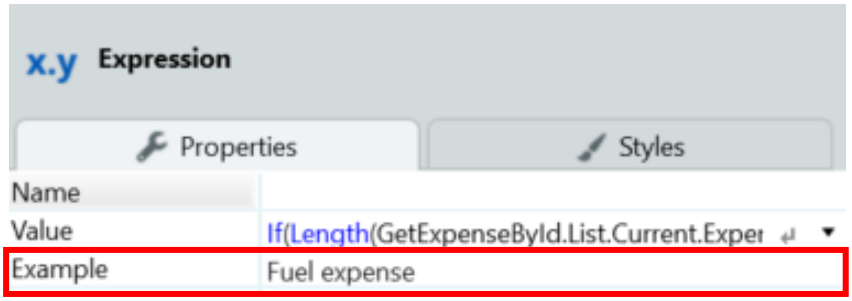

Figura 26 - Texto de exemplo da expressão definido

- Manter os fluxos de ação verticais e organizados

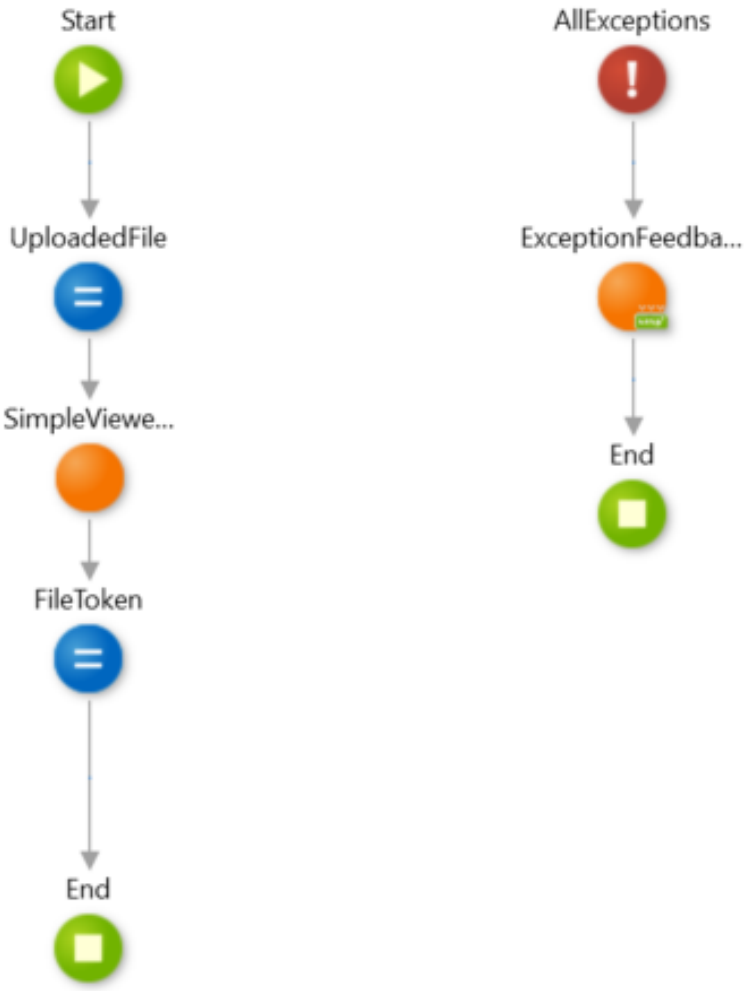

Figura 27 - Fluxo da ação vertical e claro

- Usar entidades estáticas em vez de valores *hard-coded*

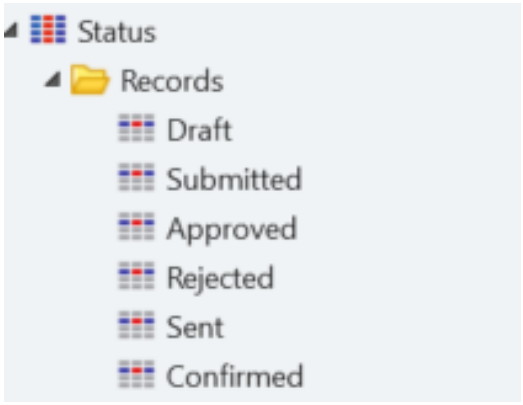

Figura 28 - Entidade estática que guarda o nome de cada estado

- Garantir que os blocos de JavaScript estão identificados e comentados

Figura 29 - Blocos de JavaScript identificados

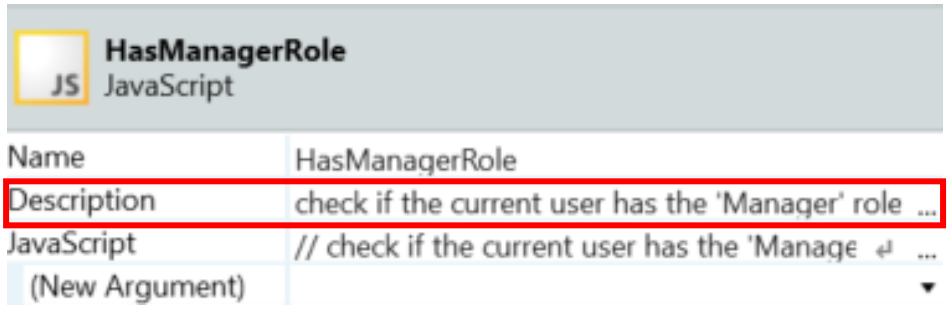

Figura 30 - Blocos de JavaScript com descrição

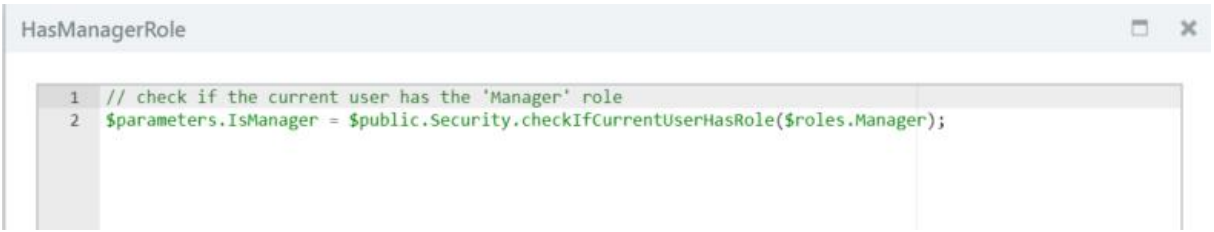

Figura 31 - Blocos de JavaScript comentados

3.5 Testes

De acordo com a metodologia escolhida, *Scrum*, foram feitos vários testes à aplicação após cada *sprint*, passando cada tarefa a *"done"* de acordo com os critérios de aceitação de forma a ter um constante *feedback* do cliente e implementar melhorias a serem apresentadas na próxima revisão de *sprint*.

Existem vários tipos de teste: unitários, integração, sistema e aceitação.

Dado o tamanho do projeto e o facto de ter sido desenvolvido numa plataforma *low-code*, não faria sentido usar a opção dos testes unitários e de integração, tendo sido feito testes de sistema e de aceitação após a entrega de cada *sprint*.

Os testes de sistema têm como objetivo testar toda a aplicação de forma a encontrar os erros, de acordo com os requisitos definidos, feitos pelos *developers*.

Os testes de aceitação têm como objetivo o teste de cada funcionalidade por parte dos utilizadores ou cliente final, garantindo que funciona como o esperado e que cumpre todos os requisitos definidos.

Exemplo de critérios de aceitação para a revisão de despesa:

- A informação da despesa deve ser visível e não editável (excetuando o campo do valor, do tipo e do projeto);
- Deve ser possível ver o anexo, se houver;
- O ecrã deve ter 2 botões: *Approve* e *Reject*;
- Se a despesa for rejeitada, é obrigatório adicionar uma razão para o efeito;
- Esta secção só deve ser visível para utilizadores com o role de gestor da aplicação.

Exemplo de critérios de aceitação para a submissão de despesa:

- A informação da despesa deve ser visível e editável;
- Todos os campos são obrigatórios exceto o *Payee*;
- Deve ser visível a opção de adicionar um anexo;
- O ecrã deve ter 2 botões: *Submit* e *Delete*.

Capítulo 4 – Conclusão

Tal como foi afirmado ao longo deste projeto, a redução de custos é uma premissa que todas as empresas visam manter. Verificou-se, no entanto, que quando feito de forma física ao invés de automática, o processamento de faturas trazia inerentemente problemas de consumo de tempo e recursos, bem como uma probabilidade considerável de se mostrar ineficiente uma vez que se poderiam perder as faturas. Verificou-se ainda que existia uma dependência da existência de internet para a submissão e revisão das faturas, quando o processo não era móvel.

Decidiu-se, então, tentar perceber de que forma se daria a resolução desta problemática. Para tal, foram colocadas como possíveis respostas a automatização e a mobilidade do processo, uma vez que se pretendia tornar o processo menos demorado, mas fácil, e diminuir os seus riscos.

Para testar estas hipóteses, foram feitos inquéritos qualitativos com uma amostra de 47 colaboradores da equipa de Platform Services, para perceber as necessidades dos consultores e gestores no contexto da empresa IG&H Consulting BV. Os resultados mostraram-se favoráveis para a comprovação das hipóteses que foram colocadas, uma vez que ambos os intervenientes do processamento de faturas (consultores e gestores*)* demonstraram interesse em ter à sua disposição uma plataforma móvel de submissão e revisão de despesas. As respostas evidenciam como fatores de melhoria a mobilidade, a rapidez do processo e a diminuição de potenciais perdas que iriam comprometer a aprovação da despesa.

Foi feito, então, o levantamento dos requisitos com os vários intervenientes, das quais ter visibilidade sobre as despesas que se submetem ou que estão pendentes de aprovação/aprovadas de forma detalhada e em *dashboard*; ser possível criar e editar as respetivas despesas; visibilidade sobre a informação pessoal do próprio; ser notificado quando existe uma nova despesa para aprovar, e quando a mesma é aprovada; poder aceder á aplicação de forma rápida, nomeadamente através de impressão digital e/ou reconhecimento facial; e, por último, ter acesso á aplicação *offline.*

Tendo comprovado a necessidade desta aplicação, e levantados os requisitos juntos dos utilizadores, tentou-se então entender qual seria o melhor método de desenvolvimento desta plataforma, sendo que se decidiu utilizar uma metodologia de desenvolvimento ágil e a plataforma *low-code* OutSytems. A aplicação foi desenvolvida usando *Scrum,* com 3 *sprints*, melhorando os requisitos de acordo com o *feedback* do cliente após cada *sprint*.

Em conclusão, percebe-se com este projeto, que ter uma aplicação móvel e automática agiliza os processos de submissão e revisão de despesas e reduz custos porque:

i. Não há necessidade do uso da internet ou de um computador, havendo a hipótese de submeter a despesa assim que a mesma é realizada;
ii. A possibilidade de anexar faturas à despesa permite também agilizar o processo, evitando a verificação física das faturas e a sua perda;
iii. Dispensa a entrega da fatura física, não sendo necessária a presença de todos os intervenientes.

Assim, tendo todas as hipóteses sido comprovadas, consideram-se as perguntas de investigação respondidas.

Embora o objetivo tenha sido direcionado à submissão e aprovação de despesas especificamente, parece seguro afirmar que mobilidade, a independência face á internet, e a diminuição de riscos de extravio podem ser entendidos como melhorias potenciais para o processamento de qualquer tipo de documentos.

O código da aplicação não está disponível devido a ser propriedade da empresa IG&H Consulting BV. No entanto, está guardado no servidor da OutSystems que a empresa usa para projetos internos.

Para trabalhos futuros resta desenvolver uma análise comparativa dos custos e tempo médio desde a submissão até a aprovação das despesas por meio físico, face aos automáticos e móveis, de forma a compilar dados mais concretos.

Está também nos objetivos futuros desta investigação escrever um artigo sobre este tema, dado que durante este trabalho de projeto tal não foi ainda possível, embora tenham havido várias tentativas de escrita do mesmo em várias conferências, tal como as conferências internacionais CEISEE e a ICE, ambas em 2019.

Anexos

Anexo A

Em que empresa trabalha? *

Your answer

A sua empresa usa algum site ou aplicação onde os colaboradores possam reportar despesas e facturas? *

- ◯ Sim
- ◯ Não

Se respondeu que sim, como se chama o site ou aplicação?

Your answer

Se respondeu que sim, acha mais útil usar um software deste género do que não usar? Porquê?

Your answer

Se respondeu que sim, seleccione as principais funcionalidades do site ou aplicação que usa?

- ☐ Anexação de facturas
- ☐ Registo e aprovação de despesas
- ☐ Notificação para a pessoa responsavel quando uma despesa é submetida
- ☐ Notificação para a pessoa que registou a despesa quando a mesma é aprovada/rejeitada
- ☐ Online e offline
- ☐ Ser mobile
- ☐ Other:

Se respondeu que sim, que melhorias sugere no site ou aplicação da sua empresa?

- ☐ Anexação de facturas
- ☐ Push notifications para quando se regista/aprova despesas
- ☐ Ter uma aplicação mobile
- ☐ Funcionar online e offline
- ☐ Other:

Se respondeu que não, acha que seria útil um software deste género? Porquê?

Bibliografia

5 Best Low-Code Development Tools That Take Easy to the Next Level. (2019). Obtido em 18 de Junho de 2018, de KISSFLOW: https://kissflow.com/low-code/lowcode-development-tools/

Awad, M. A. (2005). *A Comparison between Agile and Traditional.* Australia: The University of Western Australia.

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., . . . Thomas, D. (2001). *Manifesto for Agile Software Development*. Obtido de Manifesto for Agile Software Development: http://agilemanifesto.org/

Best Low-Code Development Platforms Software. (2019). Obtido em 5 de Janeiro de 2019, de G2crowd: https://www.g2crowd.com/categories/low-code-development-platforms

Cockburn, A., & Highsmith, J. (2001). *Agile Software Development: The business of innovation.* United States: IEEE.

Comparing OutSystems, Mendix. (2019). Obtido em 5 de Janeiro de 2019, de Gartner - Peer Insights: https://www.gartner.com/reviews/market/mobile-application-developmentplatforms/compare/OutSystems-vs-mendix

Expense Management Made Easy - SAP Netherlands. (2019). Obtido em 2 de Outubro de 2019, de SAP Concur: https://www.concur.pt/expense-management

Expense Management Software. (2019). Obtido em 2 de Outubro de 2019, de Chrome River: https://www.chromeriver.com/products/expense-management

Mines, C., Hammond, J., Sjoblom, S., Allison, V., & Reese, A. (2017). *The Forrester Wave™: Mobile Low-Code Development Platforms, Q1 2017.* Forrester.

Murphy, C. (2012). Electronic Invoice Authorization - Providing the foundation for an efficient accounts payable department. *Incorporating Asset & Risk Review*, 6.

Nerur, S., Mahapatra, R., & Mangalaraj, G. (2005). *Challenges of migrating to agile methodologies.* Texas: ACM. Obtido em Janeiro de 2019

Online Expense Report Software. (2019). Obtido em 2 de Outubro de 2019, de Zoho: https://www.zoho.com/expense/

OutSystems. (2019). *Low-Code Platforms*. Obtido em 4 de Janeiro de 2019, de OutSystems: https://www.OutSystems.com/low-code-platforms/

OutSystems. (2019). *Microservices Architecture in OutSystems*. Obtido em 4 de Janeiro de 2019, de OutSystems: https://success.OutSystems.com/Support/Enterprise_Customers/Maintenance_and_Operations/Designing_the_architecture_of_your_OutSystems_applications/Microservices_Architecture_in_OutSystems

OutSystems. (2019). *OutSystems Agile Methodology*. Obtido em 5 de Janeiro de 2019, de OutSystems: https://www.OutSystems.com/downloads/

OutSystems. (2019). *OutSystems Architecture*. Obtido em 4 de Janeiro de 2019, de OutSystems Platform: https://success.OutSystems.com/Evaluation/Architecture/2_OutSystems_Platform_architecture

OutSystems. (2019). *OutSystems Platform Best Practices*. Obtido em 10 de Dezembro de 2019, de OutSystems: https://success.OutSystems.com/Documentation/Best_Practices/OutSystems_Platform_Best_Practices

OutSystems. (2019). *The 4 Layer Canvas*. Obtido em 20 de Setembro de 2019, de OutSystems: https://success.OutSystems.com/Support/Enterprise_Customers/Maintenance_and_Operations/Designing_the_architecture_of_your_OutSystems_applications/01_The_4_Layer_Canvas

Peffers, K., Tuunanen, T., Rothenberger, M. A., & Chatterjee, S. (2007-8). A Design Science Research Methodology for Information Systems Research. *Journal of Management Information Systems*, 45-78.

Rayome, A. (16 de Maio de 2018). *Low-code platforms: A cheat sheet*. Obtido de TechRepublic: https://www.techrepublic.com/article/low-codeplatforms-a-cheat-sheet/

Rydoo Expense - Rydoo. (2019). Obtido em 2 de Outubro de 2019, de Rydoo: https://www.rydoo.com/pt/despesas/

Rymer, J., Koplowitz, R., Mines, C., Sjoblom, S., & Turley, C. (2019). *The Forrester Wave™: Low-Code Development Platforms For AD&D Professionals, Q1 2019.* Forrester Research.

Schwaber, K., & Sutherland, J. (2017). *The Scrum Guide.*

Tang, Q., & Chenghui, Z. (2012). A Simplified Document Flow System. *International Conference on Automation and Logistics* (p. 4). Zhengzhou: IEEE.

VExpenses | Reembolso de Despesas Corporativas. (2019). Obtido em 2 de Outubro de 2019, de VExpenses: https://vexpenses.com/

West, D., Gilpin, M., Grant, T., & Anderson, A. (2011). *Water-Scrum-Fall Is The Reality Of Agile For Most Organizations Today.* Forrester.