



Universidade do Minho
Escola de Engenharia
Mestrado Integrado em Engenharia Informática

Unidade Curricular de Laboratórios de Informática IV

Ano Letivo de 2020/2021

PORALI – TRANSPORT TRACKER

Ana Carneiro, Ana Peixoto, Luís Pinto, Pedro Fernandes

Março, 2021

Data de Recepção	
Responsável	
Avaliação	
Observações	



PORALI – TRANSPORT TRACKER

Ana Carneiro, Ana Peixoto, Luís Pinto, Pedro Fernandes

Março, 2021

Resumo

No âmbito da disciplina de Laboratórios de Informática IV, foi desenvolvido o projeto de monitorização de eventos aplicada à plataforma PORALI. Esta aplicação dedica-se a fornecer informação e ajuda durante as viagens em transportes públicos. Neste projeto, o *software* desenvolvido foca-se em monitorizar os autocarros da TUB, contudo a aplicação foi pensada de modo mais amplo e abrangente permitindo no futuro uma expansão para outros setores da indústria bem como diferentes empresas.

A **primeira fase** do projeto passou por contextualizar e apresentar o caso em estudo bem como definir as motivações que suportam o PORALI. Para tal, fez-se um estudo da viabilidade do projeto, da utilidade da aplicação para os utilizadores e das razões por de trás da criação da aplicação. Estabeleceram-se métricas de sucesso e catalogaram-se os recursos disponíveis que vão permitir o desenvolvimento do *software*. Finalmente, foram planeadas e estruturadas todas as tarefas das próximas etapas através do diagrama de *Gantt*, como forma de um melhor desenvolvimento e implementação da aplicação no futuro.

A **segunda fase** do projeto tem como objetivo dar forma às ideias apresentadas na fase anterior. Para isso foi realizada uma análise de requisitos e um modelo de domínio que representam o esqueleto do projeto. Numa seguinte etapa foi implementado um diagrama de classes que demonstra a estrutura arquitetural da nossa aplicação. De seguida criaram-se alguns diagramas que demonstram o fluxo entre os menus da aplicação (diagrama de atividade) e ainda uma máquina de estado que explicita a relação entre os vários estados do utilizador quando este inicia uma viagem. Além disso, também foram implementados diagramas de sequência para as principais funcionalidades do sistema. Posteriormente, foram desenhados os modelos conceptual e lógico, de forma a modelar a maneira como os dados da aplicação são armazenados. Por fim, foi apresentada, ainda que numa fase muito embrionária, diversas *mockups* representativas de alguns menus da *app*.

Por último, na **terceira fase**, ocorreu a materialização da aplicação. Por outras palavras, nesta etapa foram implementadas todas as suas componentes através da *framework* *ASP.NET*. Em adição, indicamos as ferramentas e metodologias adotadas que permitiram implementar todas as funcionalidades presentes nos requisitos, bem como o produto final e suas diversas vertentes, derivadas das / de acordo com as *mockups* apresentadas.

Área de Aplicação: Serviço de assistência em transportes públicos.

Palavras-Chave: Engenharia de *Software*, autocarros, transportes públicos, monitorização, aplicação, *WEB*, localização.

Índice

Resumo	i
Índice	iii
Índice de Figuras	vi
Índice de Tabelas	viii
1. Introdução	1
1.1. Contextualização	1
1.2. Apresentação do Caso de Estudo	2
1.3. Motivação e Objetivos	2
1.4. Estrutura do Relatório	3
2. Fundamentação	5
2.1. Justificação do Sistema	5
2.2. Utilidade do Sistema	5
2.3. Análise da viabilidade do Sistema	6
2.4. Estabelecimento da Identidade do Projeto	6
3. Planeamento	7
3.1. Identificação dos recursos disponíveis	7
3.2. Maqueta do sistema	8
3.3. <i>Mockup</i> publicitário	8
3.4. Definição de um conjunto de medidas de sucesso	10
3.5. Plano de desenvolvimento	11
4. Levantamento de Requisitos	13
4.1. Requisitos Funcionais	14
4.1.1. Registo na Aplicação	14
4.1.2. Autenticação na Aplicação	14
4.1.3. Logout na Aplicação	14
4.1.4. Configurar as informações da conta	15
4.1.5. Configurar as Notificações	15
4.1.6. Aceder às notificações	15
4.1.7. Consultar os horários dos Autocarros	16
4.1.8. Ajuda no uso da Aplicação	16
4.1.9. Recomendação a um amigo	16
4.1.10. Notificação de viagem grátis	17

4.1.11.	Iniciar Viagem	17
4.1.12.	Notificação Inicio Viagem	18
4.1.13.	Notificação Próxima Paragem	18
4.1.14.	Notificação terminar viagem	18
4.1.15.	Ativar viagens grátis	19
4.1.16.	Avaliar Aplicação	19
4.2.	Requisitos Não Funcionais	19
5.	Modelo de Domínio	20
6.	Diagramas de Use Cases	21
6.1.	Especificação de Use Cases	23
6.1.1.	Registo na Aplicação	23
6.1.2.	Autenticar na aplicação	24
6.1.3.	Terminar sessão na Aplicação	25
6.1.6.	Configurar as Notificações	27
6.1.7.	Consultar os horários dos Autocarros	28
6.1.9.	Iniciar Viagem	29
6.1.10.	Notificar início viagem	30
6.1.11.	Notificar Próxima Paragem	31
6.1.12.	Notificar terminar viagem	32
6.1.13.	Ativar viagens grátis	33
6.1.14.	Avaliar Aplicação	34
7.	Arquitetura da Aplicação	35
8.	Diagrama de Atividades	36
9.	Máquinas de Estados	37
10.	Diagrama de Componentes	38
11.	Diagrama de Classe	39
12.	Diagramas Sequência	41
12.1.	Iniciar Viagem	41
12.2.	Calcula Viagem	42
12.3.	Calcula penúltima paragem	43
12.4.	Calcula hora penúltima paragem	44
13.	Camada de Dados	45
13.1.	Modelo Conceptual	45
13.2.	Modelo Lógico	47
13.3.	Estimativa do espaço	48
14.	Views da Aplicação	51
15.	Metodologias de implementação	55
16.	Ferramentas Utilizadas	56
17.	Desenvolvimento do projeto	57
17.1.	Conexão Base de Dados	57

17.2. Obtenção dos Autocarros	58
17.3. Obtenção de Notificações	58
18. Produto Final	59
18.1. Pré-Authenticação	59
18.1.1. Barra Superior	60
18.1.2. Iniciar Sessão	60
18.1.3. Registo	61
18.2. Pós-Authenticação	62
18.2.1. Barra Superior	62
18.2.2. Iniciar Viagem	66
18.2.3. Consultar Horários	68
18.2.4. Ajuda	69
18.2.5. Configurações	69
19. Atualizações	70
19.1. Fase 1	70
19.2. Fase 2	70
20. Conclusões	71
Referências	73
Lista de Siglas e Acrónimos	75

Índice de Figuras

Figura 1: Maqueta do sistema	8
Figura 2: <i>Mockup</i> do Sistema para passageiros	9
Figura 3: <i>Mockup</i> após login do utilizador	9
Figura 4: <i>Mockup</i> do sistema para computador	10
Figura 5: Diagrama de <i>Gantt</i>	11
Figura 6: Modelo de domínio	20
Figura 7: Diagrama das funcionalidades gerais	21
Figura 8: Diagrama das funcionalidades associadas à viagem	22
Figura 9: Arquitetura da aplicação	35
Figura 10: Diagrama de atividades	36
Figura 11: Máquina de estados	37
Figura 12: Diagrama de componentes	38
Figura 13: Diagrama de classes	40
Figura 14: Diagrama de sequência - Iniciar Viagem	41
Figura 15: Diagrama de sequência de calcula viagem	42
Figura 16: Diagrama de sequência - calcula penúltima	43
Figura 17: Diagrama de sequência - calcula hora penúltima	44
Figura 18: Modelo Conceptual	46
Figura 19: Modelo Lógico	47
Figura 20: <i>Interface Login</i>	51
Figura 21: <i>Interface</i> Iniciar Viagem	52
Figura 22: <i>Interface</i> Notificação	52
Figura 23: <i>Interface</i> Conta de utilizador	53
Figura 24: <i>Interface</i> WEB do <i>Login</i>	53
Figura 25: <i>Interface</i> WEB para a consulta de funcionalidades	54
Figura 26: Menu Autenticação	59
Figura 27: Menu Autenticação - Barra Superior	60
Figura 28: Menu Iniciar Sessão	60
Figura 29: Menu Registo	61
Figura 30: Menu Principal	62
Figura 31: Menu Principal - Barra Superior	62

Figura 32: Menu Viagens Grátis - 1	63
Figura 33: Menu Viagens Grátis - 2	63
Figura 34: Menu Histórico de Viagens	64
Figura 35: Menu Avaliação Aplicação	64
Figura 36: Ver Notificações	65
Figura 37: Editar Definições	65
Figura 38: Menu Iniciar Viagem	66
Figura 39: Menu Iniciar Viagem - 2	66
Figura 40: Menu Iniciar Viagem - 3	67
Figura 41: Menu Iniciar Viagem - 4	67
Figura 42: Menu Consultar Horário - 1	68
Figura 43: Menu Consultar Horário - 2	68
Figura 44: Menu Ajuda	69
Figura 45: Menu Configurações	69

Índice de Tabelas

Tabela 1: Identidade do Projeto	6
Tabela 2: Use case - Registo na aplicação	23
Tabela 3: Use case - Autenticar na aplicação	24
Tabela 4: Use case - Terminar sessão na aplicação	25
Tabela 5: Use case - Configurar Conta	26
Tabela 6: Use case - Consultar notificações	26
Tabela 7: Use case - Configurar notificações	27
Tabela 8: Use case – Consultar horários	28
Tabela 9: Use case - Ajuda na aplicação	28
Tabela 10: Use case - Iniciar Viagem	29
Tabela 11: Use case – Notificar início de viagem	30
Tabela 12: Tamanho dos tipos de dados	48
Tabela 13: Espaço médio/máximo por cada entrada na tabela	48
Tabela 14: Tamanho da população de cada tabela	49
Tabela 15: Cálculo do espaço ocupado para anos futuros	50

1. Introdução

1.1. Contextualização

Em abril de 2019 entrou em vigor o programa de apoio à redução tarifária (PART) que tem como objetivo combater as externalidades negativas associadas à mobilidade, nomeadamente o congestionamento, a emissão de gases de efeito de estufa, a poluição atmosférica, a exclusão social, entre outros. Numa análise ao impacto criado pela redução dos tarifários nos transportes públicos entre abril e dezembro de 2019, registou-se um aumento de 15% no número de passageiros a comprar passe e uma subida de 22% na venda desses títulos, a nível nacional. Só na Grande Lisboa, o número de passes vendidos aumentou 29% neste período e os passageiros a viajar com título mensal subiu 17%. No Grande Porto, o acréscimo na venda de títulos mensais foi de 15% e no número de passageiros com passe foi de 14%.

Devido à situação pandémica do COVID-19 em Portugal e com o início do confinamento em março de 2020, houve uma diminuição do número de passageiros nos transportes urbanos provocado pelo aumento do número de pessoas em teletrabalho. Com o desconfinamento e apesar da implementação de medidas que evitam o contágio em transportes públicos como o uso de máscara e a lotação limitada a 2/3, houve um aumento do uso de transportes públicos quando comparado com os dados obtidos na altura do confinamento.

É neste ambiente que a empresa de transportes urbanos de Braga, TUB, necessita de uma aplicação dirigida aos passageiros de transportes públicas que permita a estes rastrear a circulação de transportes públicos.

1.2. Apresentação do Caso de Estudo

O PORALI – Transport Tracker é um *software* de apoio à circulação de pessoas pelos transportes públicos que permite a estas rastrear horários, rotas, paragens e meios de transporte de forma a tornar as suas viagens diárias mais simples e informadas. Com esta aplicação ambiciosa é possível tornar a circulação de passageiros pelos transportes públicos mais ágil e de fácil acesso, sem esperas ativas e atrasos que impedem os passageiros de aproveitar ao máximo a sua vida diária.

Neste caso de estudo, desenvolvemos o PORALI – *Transport Tracker* aplicado aos autocarros da TUB na cidade de Braga. Para isso, foi elaborado um conjunto completo de funcionalidades intuitivas capazes de proporcionar aos passageiros dos autocarros urbanos de Braga viagens personalizadas e descomplicadas-

1.3. Motivação e Objetivos

Hoje em dia muita gente depende dos transportes públicos, quer seja na deslocação para o local de trabalho, quer para outros espaços de lazer ou de comércio. No entanto, existem várias limitações associadas ao uso dos transportes públicos. O não cumprimento dos horários pode impedir um passageiro de chegar a horas ao seu destino, seja pelo aumento do tempo à espera na paragem ou da duração da viagem. Além disso, pode provocar *stress* ou ansiedade se este se encontrar numa cidade desconhecida e estiver dependente de transportes públicos para se deslocar. Nesse sentido, a aplicação PORALI vem mitigar ou até mesmo eliminar os efeitos destes problemas, de modo a auxiliar a empresa a fornecer uma melhor qualidade de serviço, e também proporcionar aos utilizadores uma melhor experiência nas viagens.

A aplicação tem como objetivo fornecer assistência virtual através de um mecanismo de monitorização de eventos, auxiliando os utilizadores na procura de autocarros e durante a viagem. Esta aplicação iria indicar todas as paragens do percurso – a inicial, intermédia(s) e a final. Além disso, terá acesso à informação sobre se o autocarro está atrasado ou adiantado e se este já se encontra perto do local de saída. Nestas situações serão lançadas notificações de acordo com o respetivo evento.

1.4. Estrutura do Relatório

O presente relatório está estruturado em 4 secções: Introdução, Fundamentação, Planeamento e Referências Bibliográficas.

A **Introdução** expõe uma pequena apresentação das funcionalidades da plataforma PORALI, as razões por de trás da sua criação e também o contexto no qual a aplicação está inserida.

No caso da **Fundamentação**, foi elaborada uma discussão sobre a utilidade, viabilidade e o porquê da aplicação. Neste tópico também abordamos a identidade da PORALI bem como as suas diversas características.

Em relação ao **Planeamento**, analisamos os recursos disponíveis para a criação da aplicação. Além disso, apresentamos a maqueta característica deste *software* e definimos as métricas de sucesso para manter os utilizadores e empresas na aplicação. Por fim, identificamos e planeamos as várias tarefas a realizar ao longo do projeto através de um diagrama de *Gantt*.

No **levantamento de requisitos** estão presentes todas as funcionalidades necessárias à implementação diferenciando entre requisitos funcionais e não funcionais.

Nas secções seguintes apresentamos diversos modelos e diagramas representativos da estrutura, organização e arquitetura da aplicação a implementar. Na secção **Modelo de Domínio** apresentamos a modelação das entidades e relacionamentos entre elas, já no **Diagrama de Use Cases** encontra-se não só o diagrama constituído pelas funcionalidades do sistema, mas também as especificações de cada uma delas. Além disso, nos tópicos **Arquitetura da aplicação** e **Máquina de estados** encontra-se representada a arquitetura da solução e a representação dos diversos estados que o utilizador percorre para iniciar uma viagem, respetivamente. De seguida, no tópico **Diagrama de atividades** encontra-se representado o fluxo da aplicação pelos vários menus do utilizador e através do **Diagrama de Componentes** a estrutura da camada de negócio dividida pelos múltiplos subsistemas.

No **Diagrama de classes** apresentamos a organização e estrutura da camada de negócio assim como uma descrição e caracterização das classes principais a implementar.

No tópico **Diagrama de sequência** modelamos o funcionamento e execução das funcionalidades principais do programa.

Em relação à **Camada de Dados** apresentamos o modelo conceptual e lógico do sistema de base dados do projeto, fundamental para o armazenamento de informação produzida pela aplicação.

De seguida, na secção **Views da Aplicação** apresentamos as várias *mockups* para as interfaces que utilizador terá acesso na aplicação.

Nas **metodologias de implementação** apresentamos quais os métodos e critérios que foram utilizados para a implementação do projeto assim como quais os moldes de organização tomados no desenvolvimento do projeto.

Na secção de **ferramentas a utilizar** esclarecemos quais foram as plataformas, bibliotecas e aplicações usadas para a implementação e desenvolvimento do projeto.

No capítulo de **desenvolvimento do projeto** demonstramos as características fundamentais do projeto tais como **conexão com a base de dados, obtenção dos autocarros** onde apresentamos como fizemos a interação conexão com a API externa da Google e a obtenção das notificações onde expomos as plataformas utilizadas para a implementação das várias notificações do sistema.

Em relação ao **Produto Final** apresentamos várias imagens com a interface de cada funcionalidade e página desenvolvida assim como uma breve explicação de cada uma dessas imagens.

Na secção **Atualizações** estão presentes as várias atualizações realizadas ao longo do projeto entre as várias fases, isto é, entre a fase 1 e a fase 2 e a fase 2 e a fase 3.

Finalmente no capítulo final, isto é, na **conclusão** apresentamos os pensamentos finais, melhorias e aspetos tanto positivos e negativos do nosso projeto.

2. Fundamentação

2.1. Justificação do Sistema

Embora esteja comprovado que cada vez mais pessoas têm vindo a aderir aos transportes públicos, ainda existe muita gente que prefere outras formas de transporte, como por exemplo a uber, quer seja por não conhecer bem a cidade onde se encontra ou pelo *stress* que provoca a constante atenção ao percurso para verificar o ponto de saída. Além disso, cada vez mais se verifica a consciencialização em relação ao meio ambiente e sustentabilidade. A opção por transportes públicos promete ajudar nesse sentido. Assim, a nossa proposta pretende modernizar os transportes públicos, mais propriamente os autocarros, e motivar os utilizadores a utilizá-los, através da simplificação e melhoramento da viagem para que não existam mais complicações.

2.2. Utilidade do Sistema

Após estudo do mercado e observação dos utilizadores nos transportes públicos, verificou-se que muitos relatam uma experiência desagradável devido a atrasos indevidos, horários pouco esclarecedores ou até mesmo desconhecimento da rota prestada pelo autocarro.

Nesse sentido, considerou-se pertinente elaborar um *software* que elevasse a maneira como viajamos nos transportes públicos ao próximo patamar, ou seja, através de um sistema de monitorização que notifica o utilizador com informações relevantes acerca da viagem conseguiríamos obter uma experiência mais realista, completa e agradável. Deste modo, o utilizador poderá viajar de forma mais tranquila e segura, sem haver a confusão e a inquietação constante de verificar qual a saída desejada, ou até mesmo de apanhar o autocarro atempadamente.

2.3. Análise da viabilidade do Sistema

Com o investimento na aplicação esperamos obter um melhor serviço que proporcionará uma melhor experiência aos utilizadores, que por sua vez se traduzirá num aumento da procura pelos transportes públicos gerando numa maior receita, neste caso, para a TUB e consequentemente para nós. Dado que disponibilizamos o serviço à empresa, conseguimos arrecadar dinheiro desta forma e no uso de publicidades integradas na aplicação. Concluimos assim que a receita gerada irá cobrir os custos de implementação e manutenção, tratando-se de investimento favorável com uma boa margem de lucro.

2.4. Estabelecimento da Identidade do Projeto


	
Nome	PORALI
Categoria	Assistente de transportes públicos
Idiomas	Português
Empresas Envolvidas	TUB
Faixa Etária	5+

Tabela 1: Identidade do Projeto

Aplicação Mobile que permite auxiliar dois tipos diferentes de pessoas. Serve como um assistente de viagem para os passageiros do autocarro, através do cálculo do percurso que deve seguir e respetivas paragens/mudanças de autocarro e também da notificação atempada dos locais onde se deve sair do autocarro e atrasos do mesmo.

3. Planeamento

3.1. Identificação dos recursos disponíveis

Para o desenvolvimento da aplicação PORALI são necessários diversos recursos humanos e tecnológicos de forma que todas as funcionalidades inerentes sejam atualizadas em tempo real e para que haja uma interação intuitiva com o utilizador. Além disso, é fundamental que haja conhecimento na área dos transportes públicos, neste caso no funcionamento nos autocarros da cidade de Braga de forma que o *software* proporcione aos utilizadores realismo na monitorização da sua viagem.

De forma a concretizar todas as etapas no desenvolvimento de um sistema de *software* robusto e completo que cumpra com todos os requisitos levantados, é necessário ter à disposição uma equipa de engenheiros informáticos motivados e com experiência na área de engenharia de *software*. Consequentemente, é elementar que esta equipa de profissionais seja competente e ambiciosa de modo a garantir a qualidade de trabalho ao longo de todas as etapas e que cumpram com os prazos estabelecidos.

No que toca a conhecimentos externos à área de desenvolvimento de *software*, é necessário também que haja informação e estudo de mercado, ou seja, os engenheiros a cargo do projeto devem estar a par das necessidades e desejos dos utilizadores da aplicação a desenvolver.

Além dos conhecimentos na área de *software* e nos estudos de mercado, é imperativo ter à disposição informação sobre horários, rotas e autocarros de forma a cumprir com todas os requisitos pretendidos pelos utilizadores da aplicação.

3.2. Maqueta do sistema

De forma a demonstrar a estrutura do sistema e a melhor ilustrar o programa a implementar foi elaborada uma maqueta do sistema. Esta maqueta retrata a arquitetura base do sistema seguindo uma organização típica em 3 camadas: de dados, negócio e interface, relacionando-as com o *back-end* e o *front-end*, conforme a camada. Na seguinte figura é possível observar a maqueta e as suas funcionalidades inerentes.

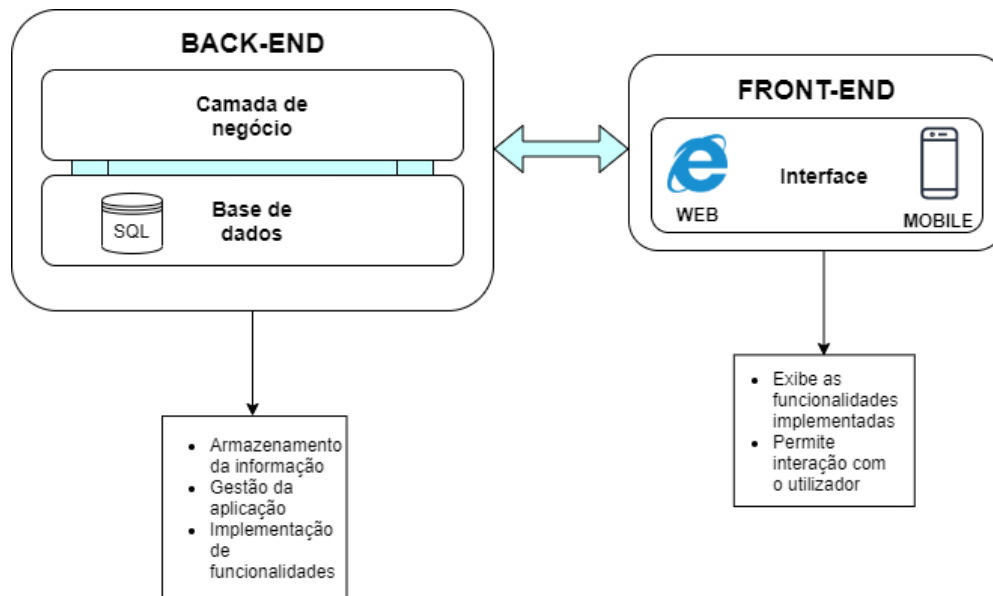


Figura 1: Maqueta do sistema

3.3. Mockup publicitário

A aplicação foi desenvolvida com o intuito de ser usada em múltiplas plataformas como o telemóvel e o computador. Para os utentes dos transportes públicos a aplicação é normalmente usada no telemóvel, contudo a versão WEB também se encontra disponível e pode ser uma mais-valia nos casos planeamento prévio de viagens.

Tal como podemos ver na figura 2, para os passageiros as funcionalidades tiram proveito da mobilidade dos telemóveis, servindo de acompanhamento nas viagens em transportes públicos. Esta vertente da aplicação tem como objetivo ser usada *on the go*, isto é, permite uma utilização rápida e direta.

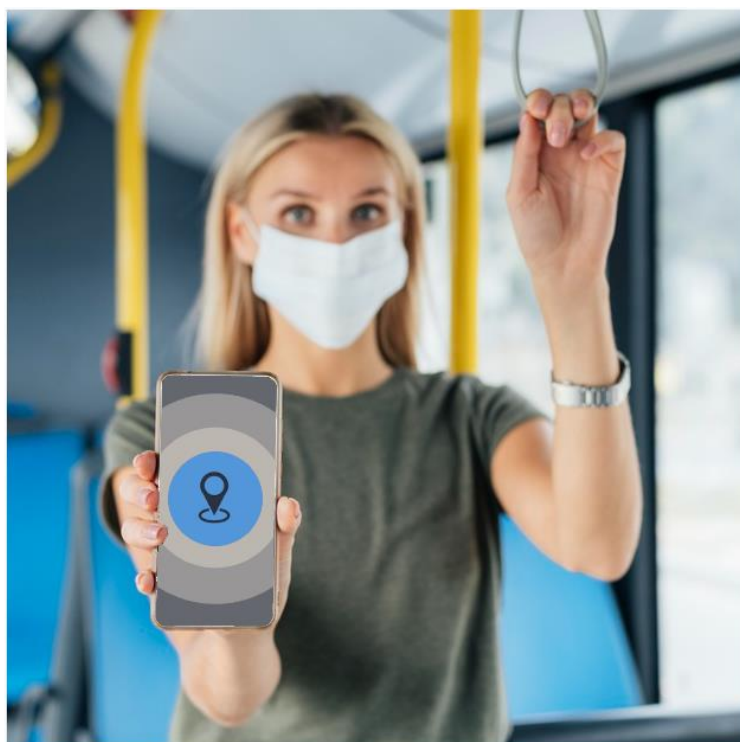


Figura 2: *Mockup* do Sistema para passageiros

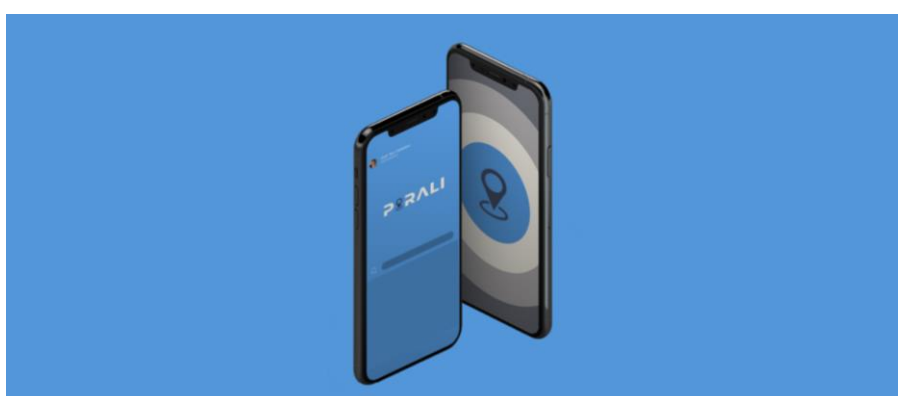


Figura 3: *Mockup* após login do utilizador

Na figura 4, apresentamos uma versão *web* publicitária da aplicação, de forma a demonstrar as funcionalidades do sistema numa outra interface e a permitir maior flexibilidade na utilização da aplicação.



Figura 4: *Mockup* do sistema para computador

3.4. Definição de um conjunto de medidas de sucesso

No PORALI, a métrica mais indicativa do seu sucesso é o número de pessoas com conta ativa que usam a aplicação diariamente. Em adição, consideramos importante avaliar o nosso sucesso no número de pessoas que começaram a usar meios de transporte ou pela maior procura na compra de passes mensais devido á aplicação.

De forma de manter o crescimento do número de utilizadores na aplicação consideramos necessário implementar um conjunto de medidas de sucesso que motive utilizadores a instalar a aplicação. A cada nova conta criada e por cada recomendação da aplicação a um amigo, o utilizador tem direito a uma viagem grátis. Em ambos casos, o passageiro mostra o passe grátis ao condutor no autocarro e este transforma o bilhete virtual em bilhete de viagem única.

A par das medidas previamente apresentadas, consideramos que o aspeto fulcral para manter os utilizadores satisfeitos será conservar uma boa prestação de serviço assente nos princípios da aplicação, isto inclui uma monitorização detalhada e atualizada dos eventos.

3.5. Plano de desenvolvimento

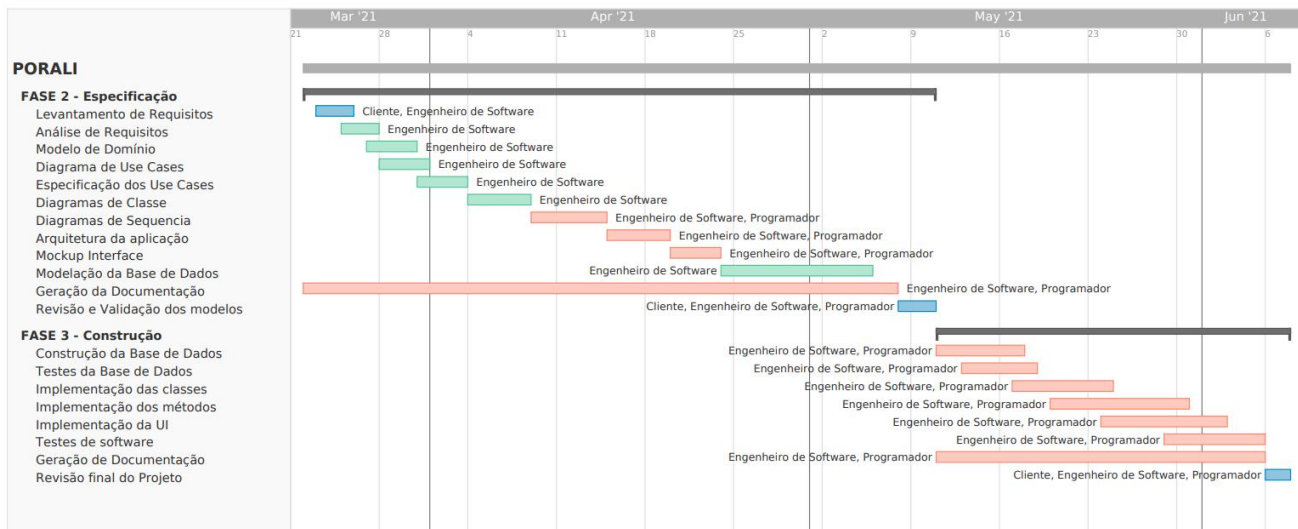


Figura 5: Diagrama de Gantt

Com a finalidade de estruturar as tarefas e desta forma organizar o processo de desenvolvimento das próximas fases do projeto, foi criado um diagrama de *Gantt* através da plataforma *WEB TeamGantt*. As tarefas foram agrupadas em 2 fases:

❖ **Especificação** que tem como objetivo analisar e especificar de forma completa todos os requisitos e use cases operacionais e funcionais. Esta fase constitui uma base fundamental para a próxima etapa através da criação de diagramas e modelos que vão servir de fundação para a implementação de *software*;

❖ **Construção** foca-se no desenvolvimento, validação, documentação e instalação do sistema de *software*. Concluída esta fase obtemos um *software* robusto e completo, isto é, o produto final da aplicação pronto a ser usado pelo público-alvo.

A fase 2 tem uma duração aproximada de 48 dias onde as tarefas são realizadas sequencialmente num movimento progressivo, fluido e constante. As tarefas têm duração semelhante. Não obstante, existe um especial foco na geração de documentação e na modelação da base de dados. Defendemos que a documentação assume um papel preponderante em qualquer projeto, necessitando de acompanhar todo o processo e ser completo na sua explicação. Além disso, achamos fundamental ter uma base de dados desenhada ao pormenor com tempo e rigor, dado que se trata de uma aplicação sujeita a um elevado número de consultas e registos na BD. Como boa prática, achamos por bem reservar 3 dias no final da segunda fase para eventuais atrasos e contingências, tentando desta forma combater a Lei de *Murphy*.

A última fase do projeto tem de duração aproximada de 28 dias onde as tarefas são desenvolvidas, tal como na fase predecessora, de modo consecutivo. As tarefas têm duração semelhante exceto a geração de documentação devido a razões supramencionadas. Além

disso, nesta fase, decidimos implementar etapas de teste à BD e ao *software*, de modo a garantir o correto funcionamento da aplicação. Como boa prática achamos por bem reservar 2 dias para revisão do trabalho efetuado tal como fizemos na fase anterior. É importante realçar que após a conclusão do projeto será necessário efetuar a sua manutenção, contudo esta etapa não faz parte do âmbito da UC.

Na globalidade, este plano de desenvolvimento assemelha-se a um modelo de desenvolvimento de *software* em cascata, isto é, análise de requisitos, projeto, implementação, testes, integração, e manutenção de *software*.

4. Levantamento de Requisitos

Nesta fase de especificação, começamos por recolher os requisitos necessários para a implementação do projeto. Neste âmbito, foram levantados requisitos de forma metódica e organizada que cumprissem com as principais funcionalidades da aplicação.

Numa primeira abordagem, reunimos com diretores e gestores da TUB para que estes nos esclareçam dúvidas quanto ao funcionamento da aplicação e da empresa, às funcionalidades que pretendem fornecer aos utilizadores e à maneira com deve ser gerida a aplicação.

Numa segunda instância, elaboramos entrevistas e questionários com passageiros de meios de transportes para tentar levantar quais as suas principais exigências de forma a torná-la o mais cómoda e funcional possível.

Durante um dia de trabalho analisamos o modo de funcionamento da empresa através da observação do modo de trabalho dos vários funcionários da empresa TUB. Assim garantimos que os vários requisitos implementados cumprem com o funcionamento da empresa e com o horário de circulação dos autocarros.

Finalmente, foi realizada uma última reunião com funcionários, gestores e diretores da empresa com intuito de validar os vários requisitos levantados nas várias tarefas, reuniões e entrevistas realizadas ao longo do tempo.

Nas secções a seguir apresentamos os diversos requisitos recolhidos, divididos entre não funcionais e funcionais, sendo estes últimos divididos por os requisitos associados ao sistema e ao utilizador.

4.1. Requisitos Funcionais

4.1.1. Registo na Aplicação

Requisitos Utilizador

1. O utilizador tem que se registar na aplicação para a poder utilizar.

Requisitos Sistema

- 1.1. O sistema requer o nome, *password* e *email*.
- 1.2. O sistema não permite mais do que um registo associado ao mesmo *email*.
- 1.3. O sistema armazena os dados.

4.1.2. Autenticação na Aplicação

Requisitos Utilizador

2. O utilizador tem que se autenticar na aplicação para a poder utilizar.

Requisitos Sistema

- 2.1. O sistema deverá pedir o *email* e a *password*.
- 2.2. O sistema deverá validar esses dados.

4.1.3. Logout na Aplicação

Requisitos Utilizador

3. O utilizador deve conseguir efetuar *logout* da conta.

Requisitos Sistema

- 3.1. O sistema deverá alertar o utilizador caso este tenha alguma tarefa pendente.
- 3.2. O sistema deverá desconectar o utilizador.

4.1.4. Configurar as informações da conta

Requisitos Utilizador

4. O utilizador deve conseguir configurar as informações da sua conta.

Requisitos Sistema

- 4.1. O sistema deverá fornecer acesso às informações da conta.
- 4.2. O sistema deverá validar as alterações feitas na conta do utilizador.
- 4.3. O sistema deverá armazenar as novas informações.

4.1.5. Configurar as Notificações

Requisitos Utilizador

5. O utilizador deve conseguir personalizar as suas notificações.

Requisitos Sistema

- 5.1. O sistema deve apresentar todas as configurações relativas às notificações do Sistema.
- 5.2. O sistema deverá permitir ao utilizador editar os campos pertinentes para o seu tipo de utilização na aplicação, ou seja, todas as notificações que pretende receber.
- 5.3. O sistema deve guardar as configurações pretendidas na base de dados.

4.1.6. Aceder às notificações

Requisitos Utilizador

6. O passageiro deve conseguir ter acesso às notificações enviadas pela aplicação durante a viagem.

Requisitos Sistema

- 6.1. O sistema deverá apresentar as notificações da viagem em questão.

4.1.7. Consultar os horários dos Autocarros

Requisitos Utilizador

7. O passageiro deve conseguir ter acesso aos horários dos autocarros.

Requisitos Sistema

- 7.1. O sistema deverá apresentar um campo onde o passageiro indica o ponto de partida e o destino.
- 7.2. O sistema deverá calcular o percurso entre a partida e o destino.
- 7.3. O sistema deverá apresentar os horários dos autocarros que realizem o percurso pretendido.

4.1.8. Ajuda no uso da Aplicação

Requisitos Utilizador

8. O utilizador deve conseguir ter acesso ao funcionamento da app, às perguntas mais frequentemente colocadas, entre outros de forma a estar informado sobre a aplicação.

Requisitos Sistema

- 8.1. O sistema deverá fornecer uma secção “Ajuda”.
- 8.2. O sistema deverá fornecer opções ao utilizador como “FAQ's”, “Funcionalidades”, “Manual de Utilização”, entre outros.
- 8.3. O sistema deverá, em cada opção, fornecer informação específica de ajuda ao utilizador.

4.1.9. Recomendação a um amigo

Requisitos Utilizador

9. O utilizador deve conseguir aplicar o código do amigo que lhe recomendou a aplicação.

Requisitos Sistema

- 9.1. O sistema deverá permitir adicionar o código de um amigo após o registo.
- 9.2. O sistema deverá validar o código de recomendação introduzido.
- 9.3. O sistema deverá atribuir um bilhete grátis tanto ao utilizador como ao amigo.

4.1.10. Notificação de viagem grátis

Requisitos Utilizador

10. O utilizador deve receber uma notificação caso receba uma viagem grátis de um amigo.

Requisitos Sistema

- 10.1. O sistema deverá apresentar ao utilizador uma notificação caso o ele receba uma nova viagem grátis de um amigo esteja atrasado ou adiantado.
- 10.2. O sistema deverá guardar a notificação apresentadas ao utilizador.

4.1.11. Iniciar Viagem

Requisitos Utilizador

11. O passageiro inicia uma viagem após escolher um autocarro de uma lista dada pelo sistema.

Requisitos Sistema

- 11.1. O sistema deverá verificar a localização do passageiro e calcular a paragem de partida mais próxima da sua localização.
- 11.2. O sistema deverá perguntar qual o destino pretendido.
- 11.3. O sistema deverá calcular os percursos entre a partida e o destino.
- 11.4. O sistema deverá determinar qual a hora de chegada dos autocarros à paragem do passageiro, através do cálculo de atrasos e adiantamentos.
- 11.5. O sistema deverá listar as viagens que realizem os percursos calculados e as respetivas horas de chegada a paragem de partida calculada.

4.1.12. Notificação Início Viagem

Requisitos Utilizador

12. O passageiro deve ser notificado quanto ao atraso e adiantamento dos autocarros.

Requisitos Sistema

- 12.1. O sistema deverá apresentar ao passageiro uma notificação quando o autocarro chega à paragem.
12.2. O sistema deverá guardar a notificação apresentada ao utilizador.

4.1.13. Notificação Próxima Paragem

Requisitos Utilizador

13. O passageiro deve receber notificações quando chega à paragem anterior à paragem de destino.

Requisitos Sistema

- 13.1. O sistema deve notificar o passageiro que chegou à paragem anterior ao destino.
13.2. O sistema deverá guardar as notificações apresentadas ao utilizador.

4.1.14. Notificação terminar viagem

Requisitos Utilizador

14. O passageiro deve receber uma notificação ao chegar ao destino.

Requisitos Sistema

- 14.1. O sistema deverá enviar uma notificação quando o passageiro chegar ao seu destino.
14.2. O sistema deverá terminar a viagem.
14.3. O sistema deverá eliminar as notificações sobre essa viagem.

4.1.15. Ativar viagens grátis

Requisitos Utilizador

15. O passageiro deve conseguir ter acesso ao conjunto de viagens grátis que tem.

Requisitos Sistema

- 15.1. O sistema deverá apresentar ao utilizador as suas viagens grátis ainda não redimidas, por ordem crescente de validade.
- 15.2. O sistema deverá ativar a viagem selecionada pelo utilizador.
- 15.3. O sistema deverá remover a viagem ativada.

4.1.16. Avaliar Aplicação

Requisitos Utilizador

16. O utilizador deve conseguir avaliar a aplicação.

Requisitos Sistema

- 16.1. O sistema deverá apresentar uma escala de classificação da aplicação entre uma a cinco estrelas, sendo cinco a avaliação máxima.
- 16.2. A base de dados deverá armazenar a classificação atribuída pelo respetivo utilizador.

4.2. Requisitos Não Funcionais

1. UI simples e de fácil manuseamento.
2. A aplicação deve estar disponível no mesmo horário da existência de autocarros.
3. Aquando dos alertas, a aplicação deve ser responsiva o suficiente para não induzir o utilizador em erro.
4. A aplicação deve ser suportada nos browsers mais usados. Assim sendo, permitirá ser acedida tanto em *android*, *IOS*, *Linux*, *Windows*, *MacOS* ou ainda *ChromeOS*.

5. Modelo de Domínio

Para a criação do modelo de domínio tivemos por base os requisitos levantados na tarefa anterior. Deste modo, foi desenvolvido um modelo representativo do *software* a implementar onde se mostra as interações entre as entidades do sistema e se modela conceitos e comportamentos fundamentais à criação da aplicação. Na figura abaixo encontra-se representado o modelo de domínio criado para o nosso projeto.

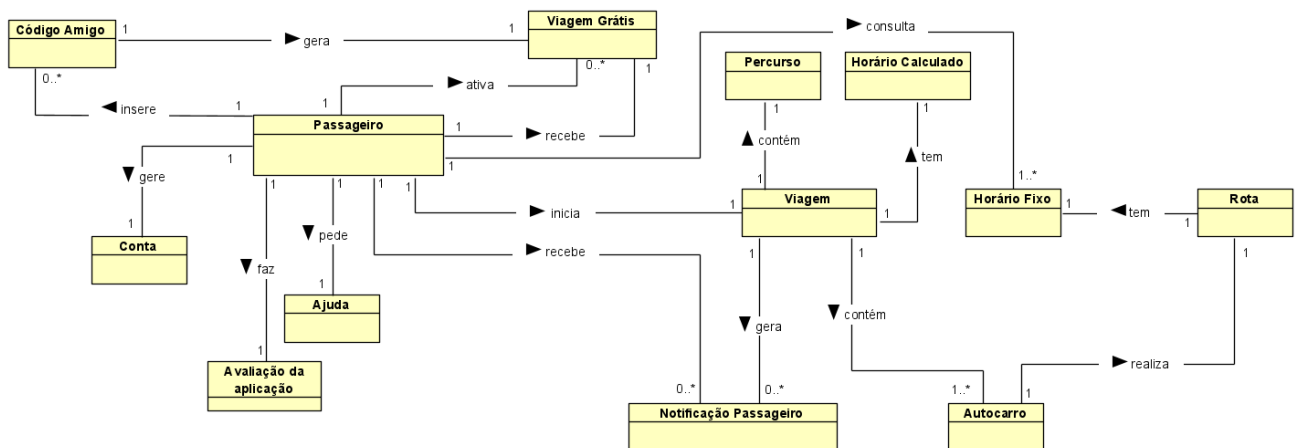


Figura 6: Modelo de domínio

Através do modelo apresentado conseguimos analisar facilmente e intuitivamente todas as funcionalidades e entidades e a maneira como elas interagem entre si. Podemos ver para o passageiro a maneira como este se relaciona com as viagens e autocarros e todos os seus conceitos inerentes. Além disso, podemos também analisar como este interage com as outras funcionalidades como a ativação de viagens grátis, a receção de notificações, consulta de ajuda, entre outros.

6. Diagramas de Use Cases

Na criação do diagrama de Use Cases foram implementados dois diagramas. O primeiro reflete todas as funcionalidades gerais ligadas a conta do utilizador. No segundo diagrama refletimos todas as funcionalidades associadas às viagens. Deste forma, tomamos a análise dos diagramas mais organizada e de fácil interpretação.

No diagrama abaixo conseguimos ver as funcionalidades relativas à gestão da conta do utilizador. É, por isso, que neste diagrama se encontra os use cases de **registar nova conta**, fazer **autenticação** e **terminar sessão** na aplicação. Além disso, é possível **configurar a conta**, **as notificações**, **avaliar a aplicação** e **pedir ajuda no funcionamento da aplicação**, **consultar horários**, **receber notificações de viagens grátis**.

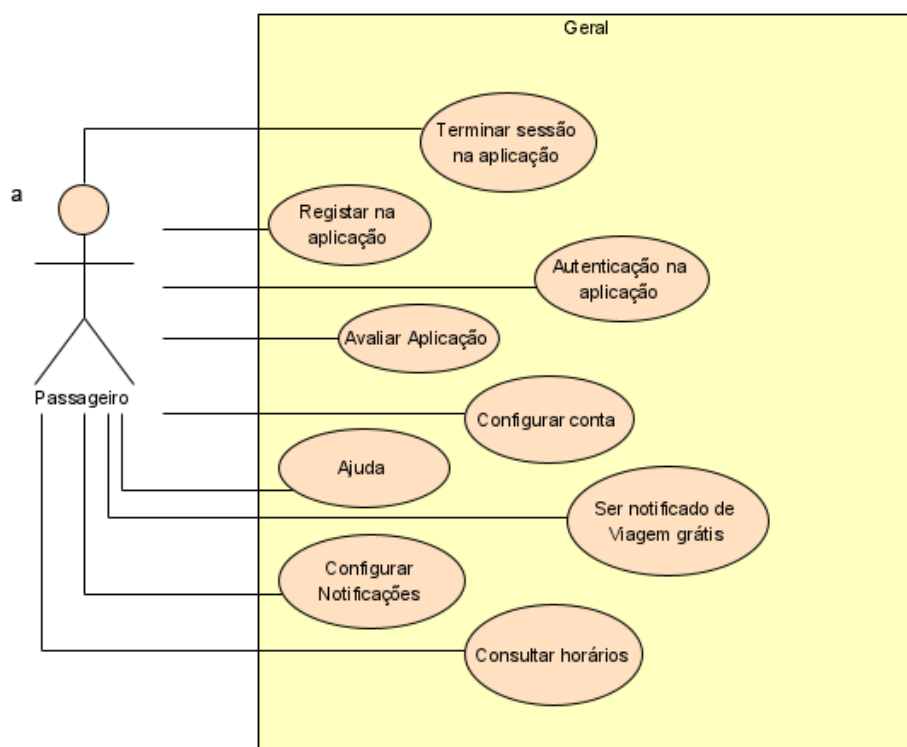


Figura 7: Diagrama das funcionalidades gerais

No próximo diagrama estão representadas todas as funcionalidades inerentes às viagens. Para o passageiro é necessário que consiga **ativar uma viagem grátis**, **iniciar uma nova viagem** na aplicação e **consultar notificações**. Além disso, é necessário que este seja notificado com **alertas de autocarros atrasados**, **alertas de fim de viagem** e de **próxima viagem**.

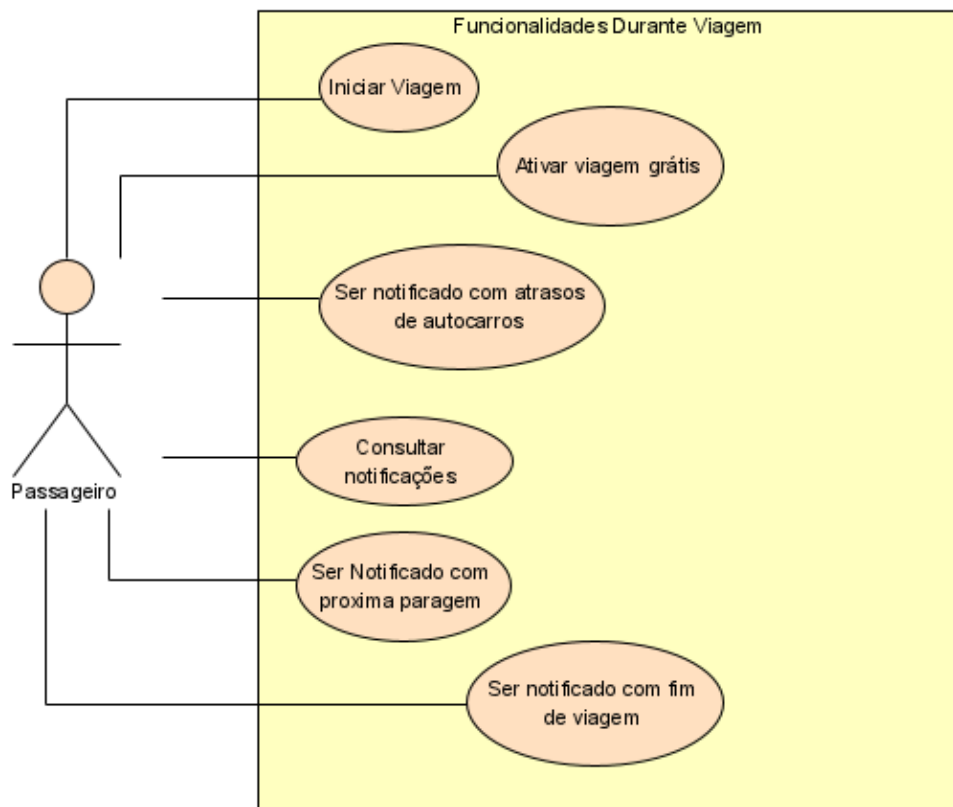


Figura 8: Diagrama das funcionalidades associadas à viagem

6.1. Especificação de Use Cases

Nas secções seguintes estão especificados os use cases que foram apresentados nos diagramas anteriores. Assim, conseguimos uma melhor análise dos passos que cada funcionalidade terá para posteriormente ser mais fácil a sua implementação.

6.1.1. Registo na Aplicação

Use Case	Registo na aplicação	
Descrição	Utilizador regista-se na aplicação	
Pré-condição	Utilizador não está registado na aplicação	
Pós-condição	Utilizador fica registado na aplicação	
	Ator	Sistema
Cenário Normal	1. Utilizador fornece email e palavra-passe	
		2. Sistema valida parâmetros fornecidos
		3. Sistema guarda um novo utilizador
		4. Sistema informa sucesso no registo
		5. Sistema atribui código de utilizador
		6. Sistema atribui viagem grátis
Cenário alternativo [Utilizador fornece código do amigo] (passo 1)	1.1. Utilizador fornece email e palavra-passe e código de recomendação	
		1.2. Sistema valida parâmetros
		1.3. Sistema atribui ao utilizador e ao amigo um bilhete grátis
		1.4. Sistema notifica o amigo de viagem grátis
		1.4. Retomar a 3
Cenário excecional 1 [Email já existe] (passo 2)		2.1 Sistema informa que o email já existe
		2.2. Sistema cancela registo
Cenário excecional 2 [Código de recomendação inválido] (passo1.2)		1.2.1 Sistema informa que o código de recomendação não existe
		1.2.2. Sistema cancela registo

Tabela 2: Use case - Registo na aplicação

6.1.2. Autenticar na aplicação

Use Case	Autenticar aplicação	
Descrição	Utilizador autentica-se na aplicação	
Pré-condição	Utilizador não está autenticado na aplicação	
Pós-condição	Utilizador fica autenticado na aplicação	
	Ator	Sistema
Cenário Normal	1. Utilizador fornece palavra-passe e email.	
		2. Sistema avalia os dados introduzidos.
		3. Sistema autentica utilizador na aplicação
Cenário Excecional [Dados incompatíveis] (passo 2)		2.1. Sistema avisa que o utilizador forneceu dados inválidos
		2.2. Sistema cancela autenticação

Tabela 3: Use case - Autenticar na aplicação

6.1.3. Terminar sessão na Aplicação

Use Case	Terminar sessão na aplicação	
Descrição	Utilizador sai da aplicação	
Pré-condição	Utilizador está autenticado na aplicação	
Pós-condição	Utilizador não fica autenticado na aplicação	
	Ator	Sistema
Cenário Normal	1. Utilizador termina sessão na aplicação	
		2. Sistema desconecta utilizador da aplicação
Cenário alternativo [Utilizador tem tarefas pendentes] (passo 2)		2.1. Sistema avisa que o utilizador tem tarefas pendentes
	2.2. Utilizador continua com terminar sessão	
		2.3. Retomar a 2
Cenário excecional 1 [Utilizador a cancela logout] (passo 2.2)		2.2.1 Sistema cancela saída da aplicação

Tabela 4: Use case - Terminar sessão na aplicação

6.1.4. Configurar as informações da conta

Use Case	Configuração de informações da conta.	
Descrição	Utilizador deve conseguir configurar definições da conta.	
Pré-condição	Utilizador tem de estar autenticado.	
Pós-condição	As definições da conta ficam atualizadas.	
	Ator	Sistema
Cenário Normal	1. Utilizador acede as configurações da conta.	
		2. Sistema apresenta configurações.
	3. Utilizador insere novos dados da conta.	
		4. Sistema valida as novas informações da conta.
		5. Sistema armazena os novos dados.
Cenário Excecional [Dados incompatíveis] (passo 4)		4.1. Sistema avisa que os dados são inválidos.
		4.2. Sistema impede a configuração dos dados da conta.

Tabela 5: Use case - Configurar Conta

6.1.5. Consultar notificações

Use Case	Consultar lista de notificações	
Descrição	Passageiro acede às notificações	
Pré-condição	Passageiro está autenticado na aplicação e a efetuar uma viagem	
Pós-condição	Passageiro tem acesso à lista de notificações	
	Ator	Sistema
Cenário Normal	1. Passageiro acede às notificações	
		2. Sistema lista as notificações do passageiro

Tabela 6: Use case - Consultar notificações

6.1.6. Configurar as Notificações

Use Case	Configurar as Notificações.	
Descrição	O utilizador deve conseguir seleccionar as notificações que pretende receber.	
Pré-condição	Utilizador tem de estar autenticado.	
Pós-condição	As preferências de notificações ficam registadas.	
	Ator	Sistema
Cenário Normal	1. Utilizador acede as notificações da conta.	
		2. Sistema apresenta as notificações adotadas.
	3. Utilizador insere preferências quanto as notificações a receber.	
		4. Sistema regista as preferências seleccionadas.

Tabela 7: Use case - Configurar notificações

6.1.7.Consultar os horários dos Autocarros

Use Case	Consultar Horários	
Descrição	Passageiro consulta horários na aplicação	
Pré-condição	Passageiro está autenticado na aplicação	
Pós-condição	Passageiro fica com acesso aos horários autocarros	
	Ator	Sistema
Cenário Normal	1. Utilizador envia informação sobre o seu ponto de partida e destino	
		2. Sistema calcula percurso entre as duas paragens apresentadas
		3. Sistema determina que autocarros fazem o percurso apresentado
		4. Sistema devolve os horários dos autocarros determinados
Cenário excecional 1 [Sistema não consegue determinar autocarros] (passo 3)		3.1. Sistema alerta passageiro que não existe autocarros para aquele percurso
		3.2. Sistema cancela operação

Tabela 8: Use case – Consultar horários

6.1.8.Ajuda no uso da Aplicação

Use Case	Ajuda no uso da Aplicação.	
Descrição	Utilizador tem acesso às perguntas mais frequentes da utilização da aplicação e outras explicações uteis adicionais.	
Pré-condição	Utilizador tem de estar autenticado.	
Pós-condição	Utilizador obteve acesso a zona da ajuda.	
	Ator	Sistema
Cenário Normal	1. Utilizador seleciona a zona Ajuda.	
		2.Sistema apresenta as informações presentes na secção de ajuda.

Tabela 9: Use case - Ajuda na aplicação

6.1.9. Iniciar Viagem

Use Case	Iniciar Viagem	
Descrição	Passageiro inicia uma viagem após selecionar autocarro da lista de autocarros dada pelo sistema.	
Pré-condição	Passageiro tem de estar autenticado	
Pós-condição	Sistema fica com mais um registo de viagem	
	Ator	Sistema
Cenário Normal	1. Passageiro indica quer iniciar viagem e insere o destino	
		2. Sistema calcula os percursos de acordo com a origem e o destino
		3. Sistema calcula autocarros que cumprem os percursos anteriormente definidos
		4. Sistema apresenta uma lista ao utilizador com os autocarros e suas horas de chegada, tanto a zona de partida como ao destino final
	5. Utilizador escolhe o autocarro de acordo com as suas preferências.	
		6. Sistema calcula penúltima paragem e respetiva hora do percurso selecionado
		7. Sistema regista escolha do utilizador.
Cenário Excecional [Não existem autocarros para aquele caminho] (passo 3)		3.1. Sistema informa utilizador que não há autocarros para o destino inserido.
		3.2. Sistema cancela início de viagem.

Tabela 10: Use case - Iniciar Viagem

6.1.10. Notificar início viagem

Use Case	Alerta sobre quando o autocarro chega à paragem de início.	
Descrição	Passageiro deve ser notificado quando o autocarro chega ao início	
Pré-condição	Utilizador deve estar autenticado	
Pós-condição	Sistema tem mais um registo de notificação.	
	Ator	Sistema
Cenário Normal		1. Sistema determina hora atual
		2. Sistema consulta horário previsto para a chegada á paragem de início
		3. Sistema compara novo hora atual com a prevista.
		4. Sistema guarda e envia notificação
Cenário Excecional horário previsto não igual ao atual] (passo 3)		3.1. Sistema não envia notificação

Tabela 11: Use case – Notificar início de viagem

6.1.11. Notificar Próxima Paragem

Use Case	Notificar de próxima paragem	
Descrição	Passageiro recebe notificação que a próxima paragem é a paragem final	
Pré-condição	Passageiro está autenticado na aplicação e a efetuar uma viagem	
Pós-condição	Passageiro fica com um novo registo de notificação	
	Ator	Sistema
Cenário Normal		1. Sistema determina hora atual.
		2. Sistema consulta horário previsto para a chegada á penúltima paragem.
		3. Sistema compara novo hora atual com a prevista.
		4. Sistema guarda e envia notificação
Cenário Excecional horário previsto não igual ao atual] (passo 3)		3.1. Sistema cancela alerta de próxima paragem

Tabela 12: Use case - Notificar de próxima paragem

6.1.12. Notificar terminar viagem

Use Case	Notificação de terminar viagem.	
Descrição	Passageiro deve receber uma notificação ao chegar ao destino.	
Pré-condição	Passageiro deve estar autenticado e durante uma viagem.	
Pós-condição	Notificações relativas à viagem em questão foram removidas.	
	Ator	Sistema
Cenário Normal		1. Sistema determina hora atual.
		2. Sistema consulta horário previsto para a chegada à última paragem.
		3. Sistema compara novo hora atual com a prevista.
	4. Utilizador termina a viagem em questão.	
		5. Sistema termina a viagem e elimina notificações lançadas durante a viagem, pois tornam-se obsoletas.
Cenário Excecional horário previsto não igual ao atual] (passo 3)		3.1. Sistema cancela alerta de terminar viagem

Tabela 13: Use case - Notificar terminar viagem

6.1.13. Ativar viagens grátis

Use Case	Ativar viagens grátis	
Descrição	Passageiro ativa viagem grátis	
Pré-condição	Passageiro está autenticado na aplicação	
Pós-condição	Passageiro fica com menos um registo de viagem grátis	
	Ator	Sistema
Cenário Normal	1. Passageiro acede às suas viagens grátis	
		2. Sistema apresenta ao utilizador a lista de viagens grátis não redimidas.
	3. Passageiro seleciona qual a viagem grátis a utilizar	
		4. Sistema ativa viagem selecionada
		5. Sistema remove registo sobre viagem selecionada.
Cenário excecional 1 [Não existem viagens não redimidas] (passo 2)		2.1. Sistema informa que não existem viagens por redimir.

Tabela 14: Use case - Ativar viagem

6.1.14. Avaliar Aplicação

Use Case	Avaliar Aplicação	
Descrição	O utilizador avalia a aplicação	
Pré-condição	Utilizador está autenticado na aplicação	
Pós-condição	Sistema fica com registo de uma nova avaliação	
	Ator	Sistema
Cenário Normal	1. Utilizador avalia a aplicação.	
		2. Sistema guarda a avaliação do utilizador

Tabela 15: Use case - Avaliar aplicação

7. Arquitetura da Aplicação

Nesta secção implementamos um modelo que nos permite demonstrar e modelar a arquitetura da solução a implementar. A arquitetura é baseada no padrão de software MVC, *Model – View – Controller*, onde o *Model* representa toda a camada de negócio associada à aplicação, o *View* é referente toda a camada de interface e o *Controller* é uma camada que recebe *inputs* do utilizador de forma a controlar e atualizar as camadas de *View* e *Model* com a informação recebida.

A aplicação será implementada em *c#*, contudo a base de dados será criada em SQL, tal como está demonstrado na figura 9. A API externa da *Google Maps*, **API Directions** e **API Distance Matrix**, terá como objetivo fornecer informação acerca dos autocarros, mapas e localizações em tempo real. Como forma de desenvolver conteúdo WEB decidimos utilizar a plataforma *Blazor* por não ser necessário utilizar várias linguagens como *JavaScript*, *Angular*, *React*, entre outras...na implementação do servidor e cliente, bastando apenas o uso do *c#*.

Na figura seguinte está representada a arquitetura da nossa aplicação onde podemos ver as interações entre as várias camadas e as interações entre a base de dados, as API's e o sistema criado.

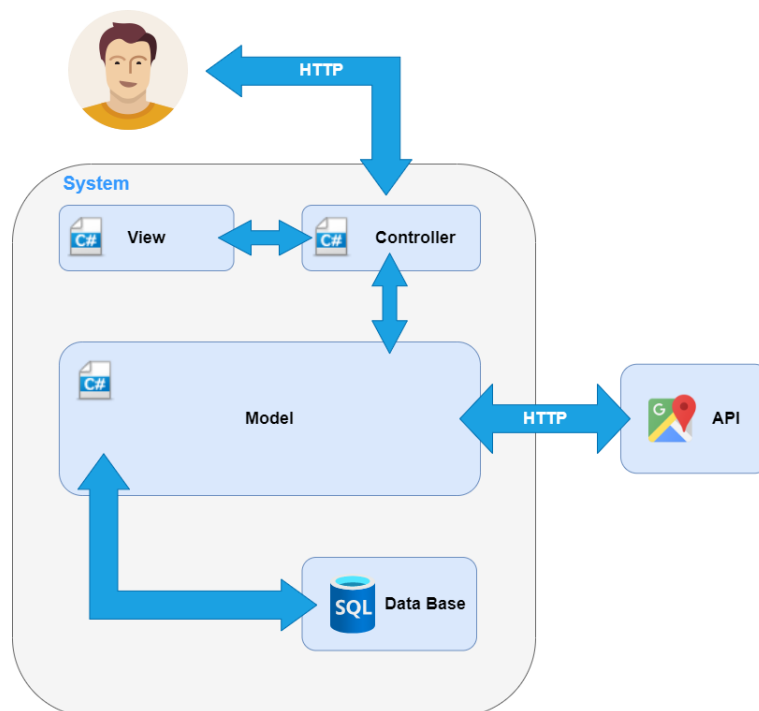


Figura 9: Arquitetura da aplicação

8. Diagrama de Atividades

Para representar o funcionamento e fluxo dos vários menus do utilizador e como estes interagem entre si decidimos implementar um **diagrama de atividades**. O diagrama desenvolvido é representativo do modo de funcionamento e organização dos vários menus associados ao passageiro. Assim, o digrama apresentado na figura abaixo tem como objetivo modelar o fluxo entre os vários menus para, conseqüentemente, ser mais fácil a sua implementação em fases e tarefas futuras.

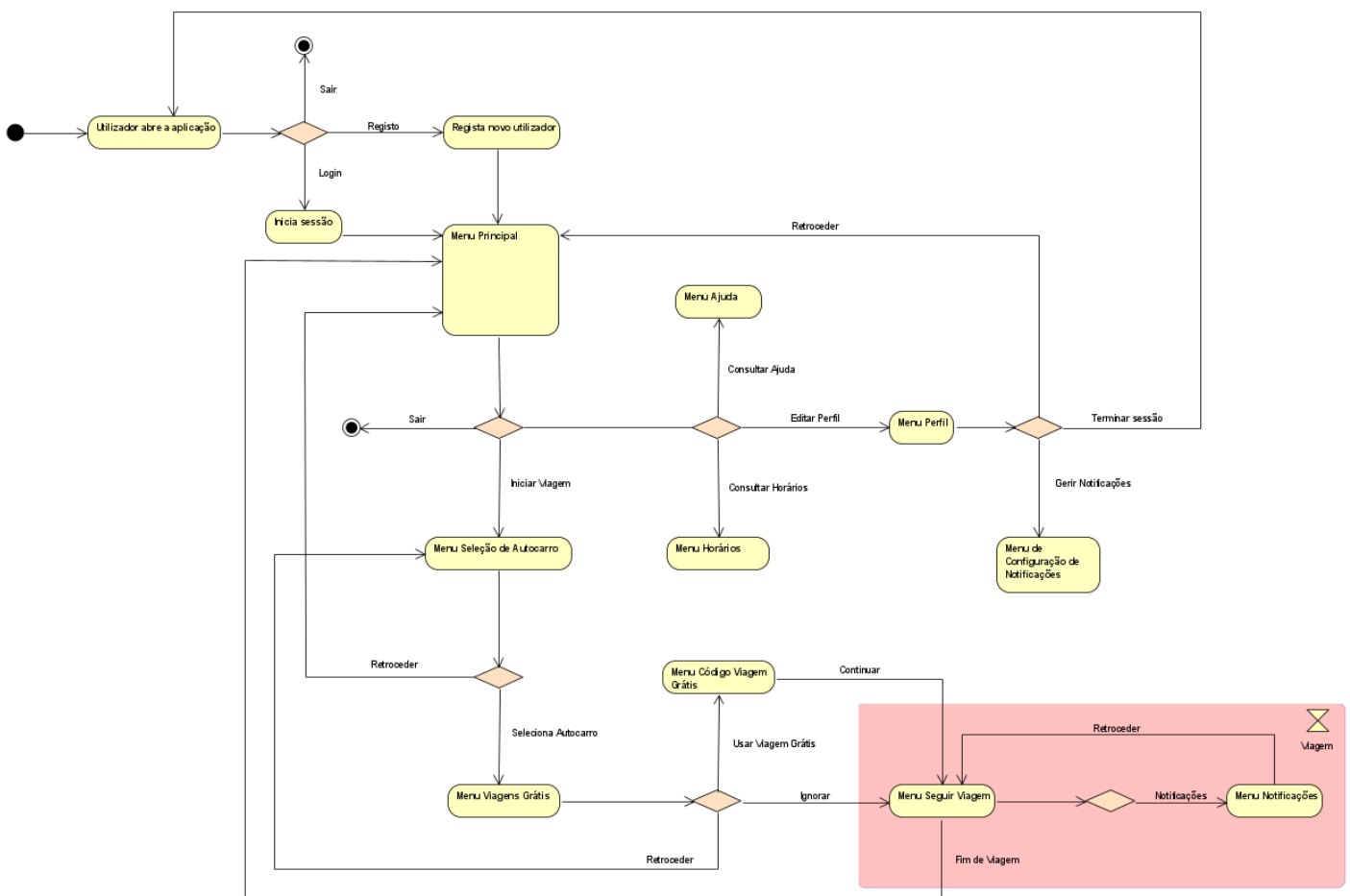


Figura 10: Diagrama de atividades

9. Máquinas de Estados

A equipa considerou essencial implementar uma máquina de estados para evidenciar quais as fases pelas quais o utilizador tem de passar para iniciar uma viagem, uma vez que é a funcionalidade principal da aplicação e que vai ser mais vezes usadas pelos nossos passageiros. O modelo apresentado abaixo os estados **Menu Destino** e **Lista de Autocarros** que são essencialmente os estados que vão determinar qual a viagem do utilizador. Existem ainda dois outros estados **Mapa de Viagem** e **Menu de Notificações** que permitem exemplificar o uso da aplicação durante a viagem.

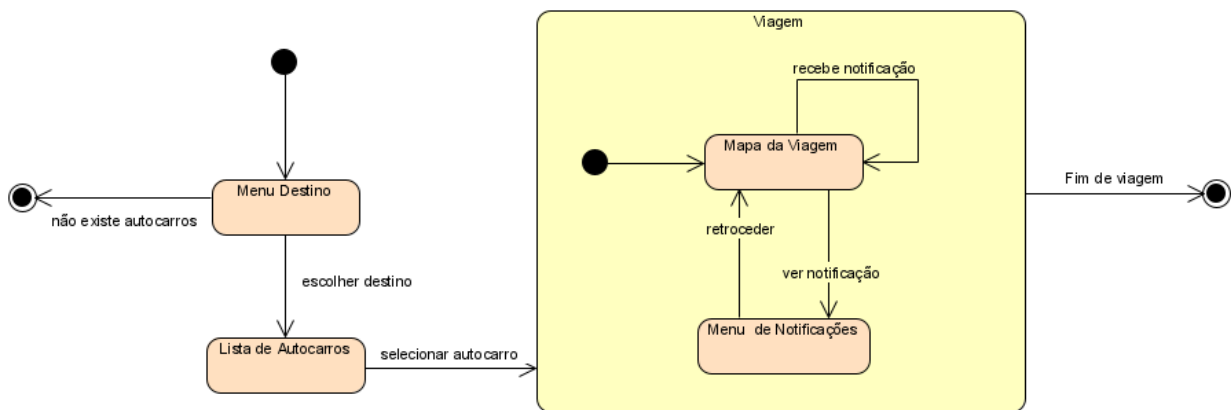


Figura 11: Máquina de estados

10. Diagrama de Componentes

A partir de um diagrama de componentes decidimos modelar a estrutura do nosso projeto como forma de melhor visualizar a disposição e organização do projeto para no futuro ser mais fácil a sua implementação. Neste diagrama estão representados os vários subsistemas e as interações entre eles, assim como a interação que o sistema tem com as API's externas e a base de dados do Porali.

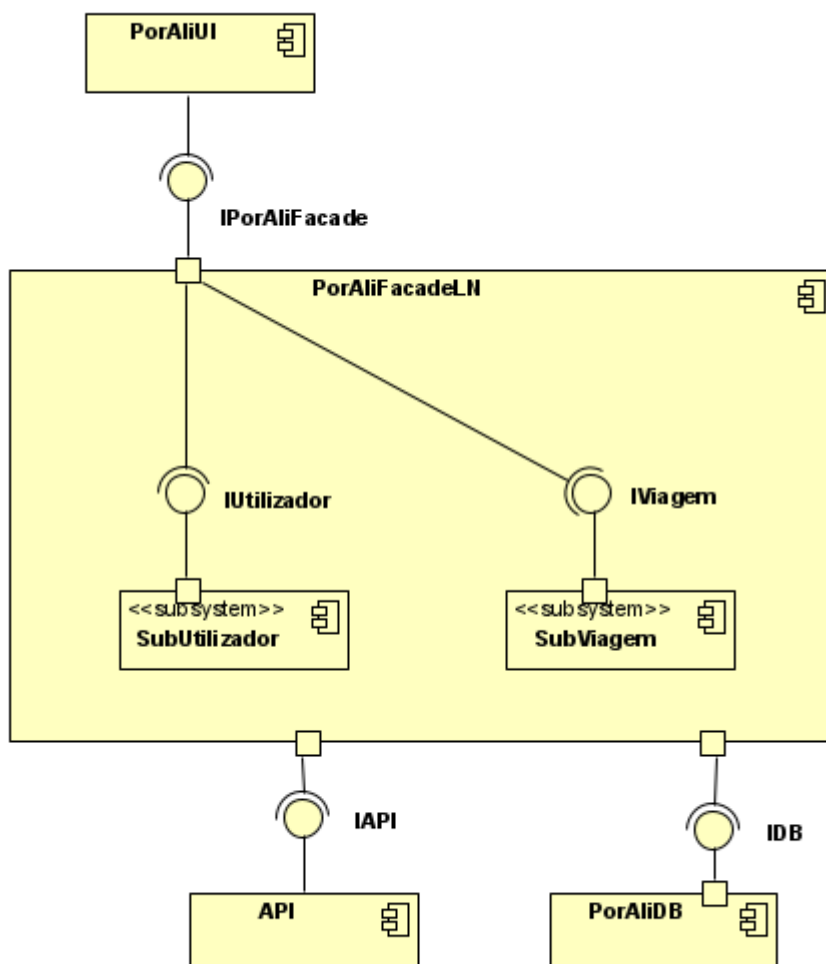


Figura 12: Diagrama de componentes

11. Diagrama de Classe

Nesta secção apresentamos a estrutura e organização da camada de negócio do projeto. Como consequência da análise dos requisitos e do modelo de domínio, implementamos dois diagramas de classes: um que representa a organização e interação entre as principais classes e outra que representa a persistência dos dados de cada classe através dos *Data Access Object* (DAO's).

Em ambas os diagramas, decidiu-se implementar um conjunto de classes e métodos que consideramos fundamentais à realização do trabalho. De seguida encontra-se a descrição das classes principais como forma de melhor caracterizar e representar os diagramas criados.

Utilizador: Corresponde a toda a informação relacionada com o passageiro como a informação sobre a conta, histórico de viagens, notificações, configurações e viagens grátis.

Viagem: classe onde esta associada toda a informação acerca da viagem, como hora de partida, chegada, percurso e autocarro associado. entre outros.

Autocarro: alberga informação sobre os autocarros como a sua rota e correspondente horário, matrícula e número de trajeto realizado.

UtilizadorFacade e **ViagemFacade:** classes onde vão estar implementadas os métodos necessários à execução das funcionalidades associadas ao utilizador e às viagens, respetivamente

PorAliFacade: classe que implementa as funcionalidades e requisitos do sistema, acedendo à informação que se encontra no utilizadorFacade e viagemFacade.

Podemos ver pelo diagrama que o utilizador terá acesso a uma lista de notificações que só serão acedidas durante a viagem, as configurações das notificações, uma lista com todas as viagens grátis associadas ao utilizador e um histórico de viagens realizadas pela aplicação. Cada viagem está associada a uma hora e local de partida e chegada e uma lista de percursos que representam as várias paragens e respetivas horas. Cada percurso é realizado por um autocarro sendo que cada autocarro realiza uma rota e cada rota é constituída por lista de paragens por onde o autocarro passa.

De modo a oferecer uma imagem mais representativa do nosso projeto e das classes que lhe oferecerão sustento, foi elaborado um **diagrama de classes** dividido em subsistemas **utilizador** e **viagem** de forma a agrupar classes.

12. Diagramas Sequência

A partir da especificação dos use cases, da arquitetura aplicacional e do diagrama de classes proposto para este sistema de *software*, os diagramas de sequência permitem representar de forma clara, simples e completa a interação entre as várias classes e a API externa, bem como, qual o fluxo de métodos e dados durante cada use case e entre cada classe.

Para representar a funcionalidade principal da aplicação, **iniciar viagem**, que a partir de uma origem e um destino inicia uma viagem escolhida pelo utilizador, decidiu-se implementar 4 diagramas de sequência que representa de forma completa os métodos utilizados pela funcionalidade de iniciar viagem.

12.1. Iniciar Viagem

Este diagrama de sequência visa representar os fluxos entre os métodos e classes de forma a representar a funcionalidade de **iniciar viagem** a partir dos métodos da interface IPorAliFacade.

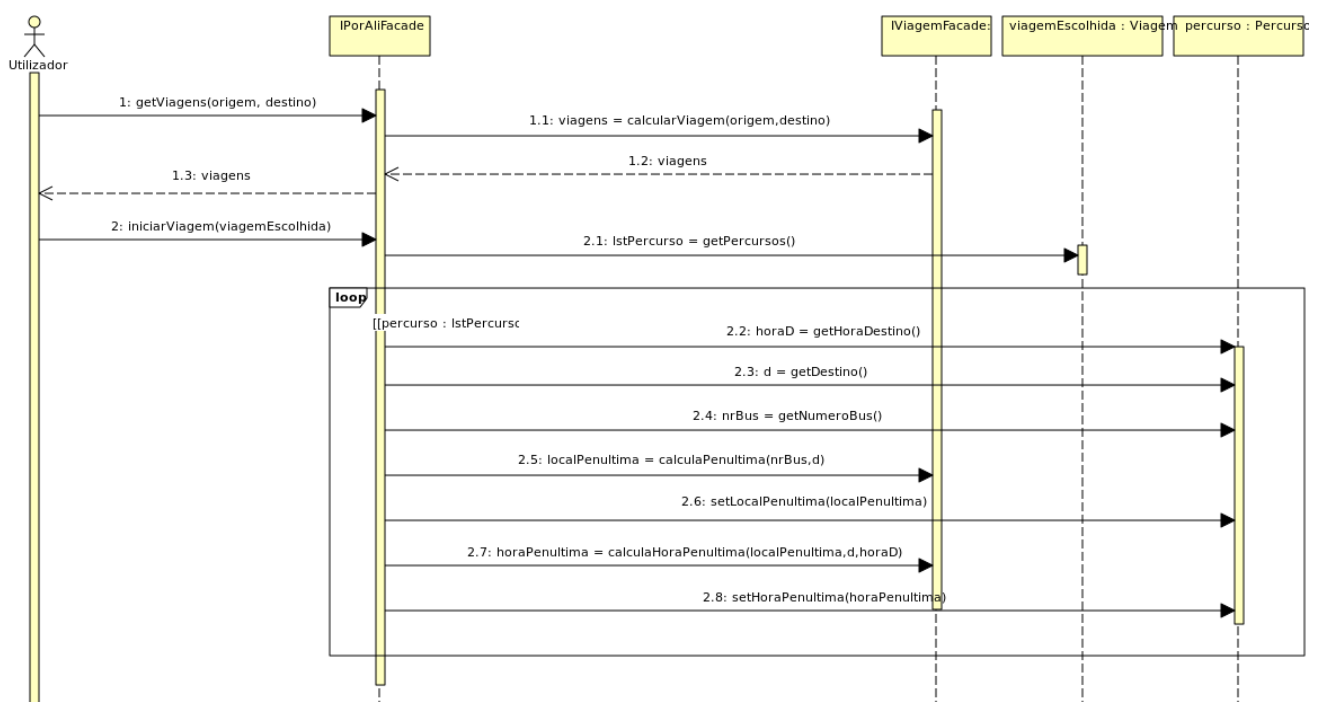


Figura 14: Diagrama de sequência - Iniciar Viagem

12.2. Calcula Viagem

O diagrama de sequência abaixo representa a funcionalidade de **calculaViagem** que a partir de uma origem e um destino acede à API Externa e devolve uma lista com as viagens fornecidas pela API do Google utilizando métodos da interface **IViagemFacade**.

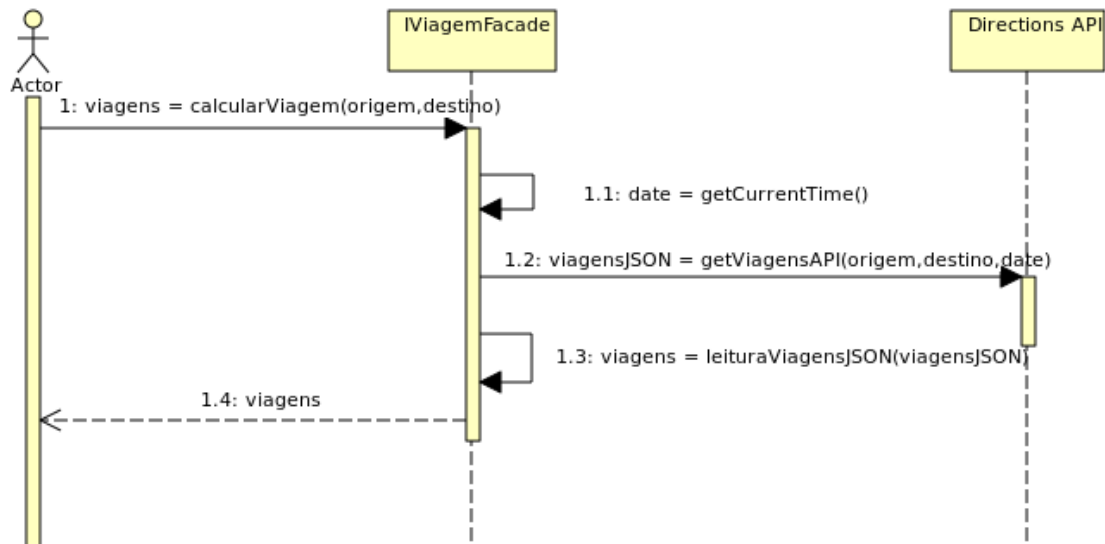


Figura 15: Diagrama de sequência de calcula viagem

12.3. Calcula penúltima paragem

Na figura abaixo está representado o diagrama de sequência que indica o fluxo de métodos de forma a representar a função **calculaPenultima** que determina a penúltima paragem de um determinado percurso.

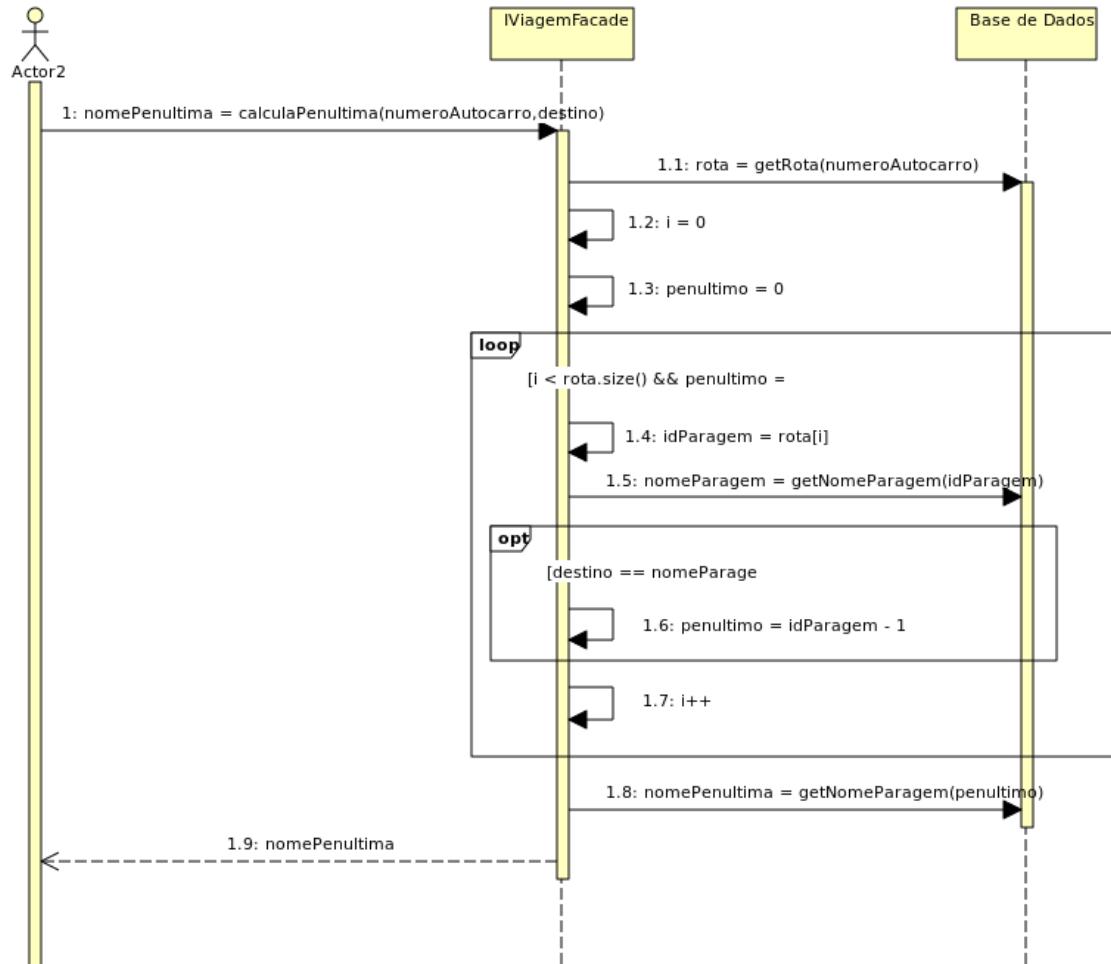


Figura 16: Diagrama de sequência - calcula penúltima

12.4. Calcula hora penúltima paragem

Abaixo está representado o diagrama de sequência que indica o fluxo de métodos de forma a representar a função **calculaHoraPenultima** que determina a hora de chegada à penúltima paragem de um determinado percurso.

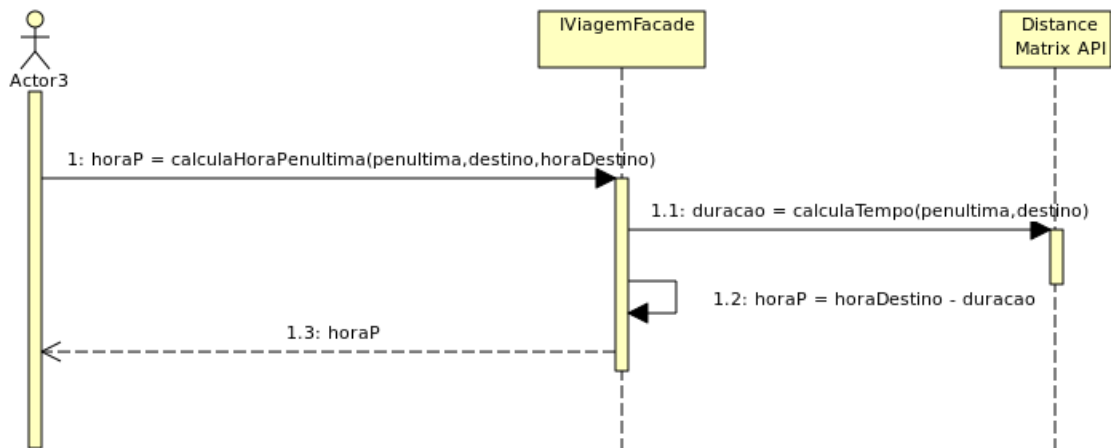


Figura 17: Diagrama de sequência - calcula hora penúltima

13. Camada de Dados

Como forma de armazenar e persistir os dados do sistema foi implementada um sistema de Base de Dados. A BD desenvolvida visa suportar a estrutura da aplicação trazendo garantidas quanto à sua integridade, consistência e segurança. Para isso, foram modeladas as várias entidades e relações entre elas fundamentais para a implementação da aplicação. Neste âmbito, foi realizado um modelo conceptual da BD seguido do seu modelo lógico.

13.1. Modelo Conceptual

O modelo conceptual surgiu como forma de modelar as entidades e relações presentes na BD. Desta forma, é possível representar as primeiras ilações sobre o sistema a implementar.

Com o auxílio da ferramenta *BRModelo* conseguimos representar esquematicamente um esboço da base de dados, enumerando tudo o que é necessário para a mesma. O esquema conceptual foi obtido através do diagrama ER (entidade-relação), onde é possível visualizar quais as entidades do sistema, os seus atributos e as relações entre as diferentes entidades. Nesta fase foi possível representar o que foi estipulado na análise de requisitos de uma forma simples e direta.

Neste modelo decidimos representar as entidades passageiro, viagem, rota, autocarro e as suas relações de forma a conseguir, posteriormente, armazenar a informação completa sobre os passageiros, viagens efetuadas e autocarros. Achamos conveniente só representar estas entidades na BD, uma vez que a informação armazenada deve persistir sempre que aplicação é encerrada. Para os restantes conceitos como notificações, achamos conveniente armazenar essa informação em memória já que as notificações só vão ser armazenadas durante a viagem. Na figura abaixo está representado o modelo conceptual implementado.

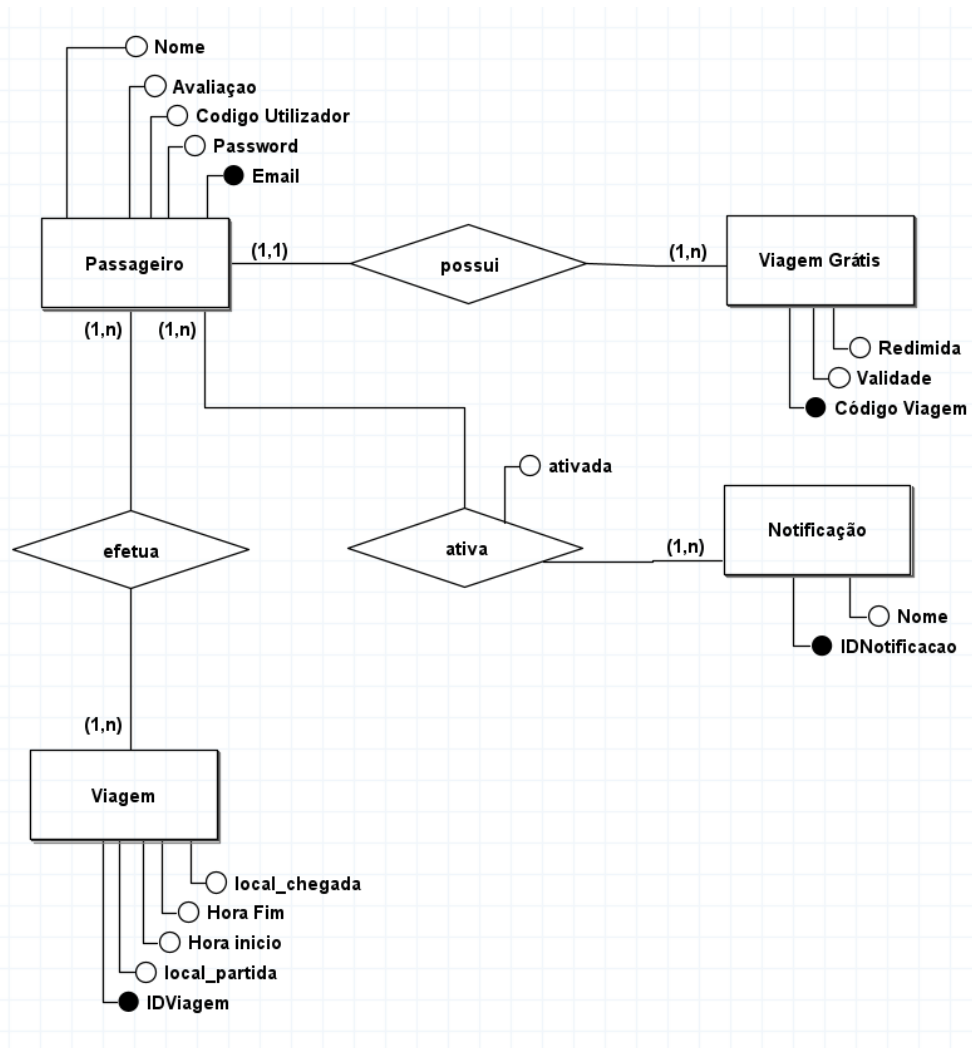


Figura 18: Modelo Conceptual

13.2. Modelo Lógico

Dada por terminada a conceptualização do nosso problema, com a criação do modelo acima explicitado, existe a necessidade de converter esse modelo para modelo lógico, usando a aplicação *SQLWorkbench*. Através da criação deste modelo vai ser possível derivar os relacionamentos entre cada uma das entidades e numa etapa futura povoar a base de dados. Na figura abaixo está representado o modelo lógico implementado segundo a conceção realizada na secção anterior.

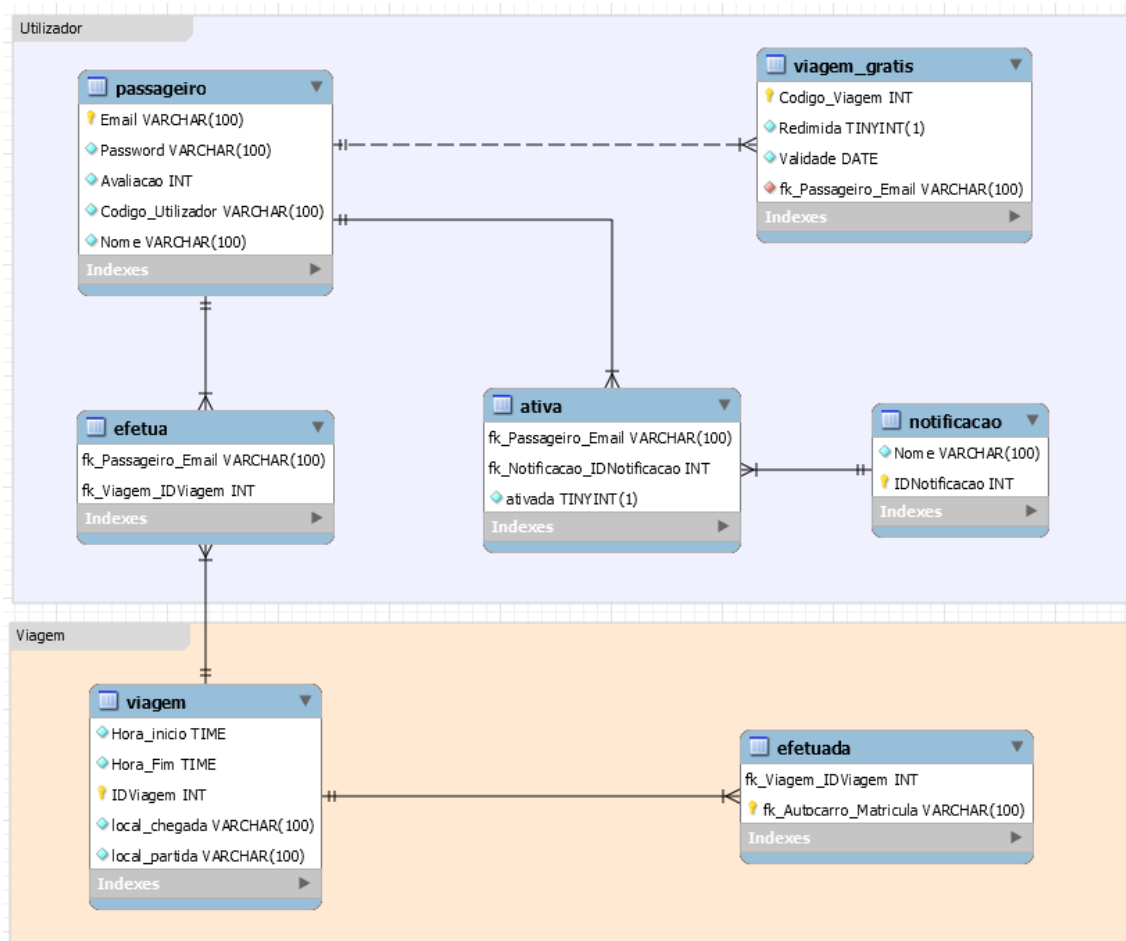


Figura 19: Modelo Lógico

13.3. Estimativa do espaço

Para o cálculo do espaço necessário para cada entrada na tabela foi utilizado o capítulo 11.7 do *MySQL 8.0 Reference Manual*, de forma a verificar o espaço necessário para cada atributo.

TIPO DE DADOS	TAMANHO (em bytes)
DOUBLE	8
VARCHAR(m)	L+1, com L o tamanho da <i>string</i>
DATE	3
TIME	3
INT	4
TINYINT	1

Tabela 12: Tamanho dos tipos de dados

É necessário ter em consideração que os tamanhos de certos atributos vão variando de acordo com os valores introduzidos, por exemplo, para o nome do passageiro, apesar de este conseguir ter um nome de 100 caracteres, o espaço armazenado neste campo (assim como todos os campos que envolvam *strings*) depende do número de caracteres introduzidos para o nome. Assim, o espaço utilizado por cada entrada vai diferindo.

De forma a melhor estimar o espaço usado, foi construída a tabela abaixo onde estão apresentados o espaço em média utilizado, isto é, para cada valor das *strings* qual é o valor máximo normalmente utilizado e o espaço máximo que pode ser utilizado:

TABELA	ESPAÇO MÉDIA DE UMA ENTRADA (em bytes)	ESPAÇO MÁXIMO DE UMA ENTRADA (em bytes)
Passageiro	≈ 74	404
Viagem_gratis	≈ 38	108
Ativa	≈ 35	105
Notificacao	≈ 19	104
Efetua	≈ 34	104
Viagem	≈ 40	210
Efetuada	≈ 10	104

Tabela 13: Espaço médio/máximo por cada entrada na tabela

Assumimos que, para um ano, o número esperado de utilizadores ativos é 20 mil, que o número de viagens de passageiros por ano são 45 e que o valor esperado de viagens grátis recebidas pelo utilizador é 3. No que toca às viagens espera-se que existam em média 2 pessoas usando a aplicação por viagem, 1 autocarro por viagem e que existem 10 rotas com 10 paragens (em média) efetuados por 10 autocarros diferentes.

Multiplicando os valores da tabela de cima pelas quantidades previstas anteriormente obteve-se o seguinte quadro para o primeiro ano:

TABELA	ESPAÇO MÉDIA DA POPULAÇÃO (em bytes)	ESPAÇO MÁXIMO DA POPULAÇÃO (em bytes)
Passageiro	≈ 1 480 000	8 080 000
Viagem_gratis	2 280 000	6 480 000
Ativa	≈ 2 800 000	8 400 000
Notificacao	≈ 79	436
Efetua	≈ 30 600 000	93 600 000
Viagem	≈ 18 000 000	94 500 000
Efetuada	4 500 000	46 800 000
+Σ	59 670 159	257 879 956

Tabela 14: Tamanho da população de cada tabela

Observando a tabela acima podemos observar que se estima que após um ano de uso da aplicação, em média, espera-se que a base de dados ocupe cerca de 60 MB e no máximo ocupe cerca de 260 MB.

Anualmente, estimamos que o volume de utilizadores aumente em cerca de 7500 passageiros. Este aumento não vai alterar o número de entradas das tabelas notificação, autocarro, rota, possui, paragem, no entanto vai gerar um aumento do espaço ocupado por todas as outras. De seguida está apresentada uma tabela com as fórmulas que demonstram o aumento das entradas dessas tabelas tendo em conta o número de anos após o ano inicial.

TABELA	AUMENTO DE ENTRADAS POR ANO (a)	ENTRADAS ADICIONADAS NO 2º ANO (a = 1)	ENTRADAS ADICIONADAS NO 3º ANO (a = 2)
Passageiro	$7500 \times a$	7 500	15 000
Viagem_gratis	$3 \times 7500 \times a$	22 500	45 000
Ativa	$4 \times 7500 \times a$	30 000	60 000
Efetua	$900000 + 337500 \times a$	1 237 500	1 575 000
Viagem	$\frac{225000 \times (3 \times a + 8)}{a + 4}$	495 000	525 000
Efetuada	$\frac{225000 \times (3 \times a + 8)}{a + 4}$	495 000	525 000

Tabela 15: Cálculo do espaço ocupado para anos futuros

Prevê-se então ano após ano que o espaço ocupado pela BD vá aumentando significativamente. Para o demonstrar, em seguida estão apresentados os cálculos que determinam o aumento com valores médios da base de dados do primeiro para o segundo ano:

$$\text{Passageiro} : 7500 \times 74 = 555000 \text{ bytes}$$

$$\text{Viagem_gratis} : 22500 \times 38 = 855000 \text{ bytes}$$

$$\text{Ativa} : 30000 \times 35 = 1050000 \text{ bytes}$$

$$\text{Efetua} : 1237500 \times 10 = 12375000 \text{ bytes}$$

$$\text{Viagem} : 495000 \times 40 = 19800000 \text{ bytes}$$

$$\text{Efetuada} : 495000 \times 10 = 4950000 \text{ bytes}$$

$$\text{Total} : 555000 + 855000 + 1050000 + 12375000 + 19800000 + 4950000 = 39485000 \text{ bytes}$$

Assim conseguimos observar que houve um aumento de cerca de 40 MB do primeiro para o segundo ano. Podemos ainda que constatar que a cada ano que passa o aumento do espaço ocupado vai sendo sucessivamente maior devido ao incremento do número de passageiros e das viagens que estes realizam.

14. Views da Aplicação

Na criação da aplicação orientada para o público em geral, neste caso para os passageiros, um dos aspetos mais relevantes para o sucesso da mesma é a sua mobilidade com o utilizador. Assim, a interface revela-se de enorme importância, uma vez que grande parte das aplicações atingem o sucesso devido à sua estética e organização de conteúdos.

Desta forma, pretendemos garantir que a Porali tenha uma interface simples, intuitiva e com aspeto agradável para que qualquer pessoa, independentemente da sua idade, consiga pegar no seu smartphone e utilizar a aplicação de uma forma simples e eficaz.

Nas figuras seguintes encontram-se representadas algumas sugestões e ideias ao modo de organização e aspeto da interface que pretendemos fornecer ao utilizador para o dispositivo móvel e para WEB.

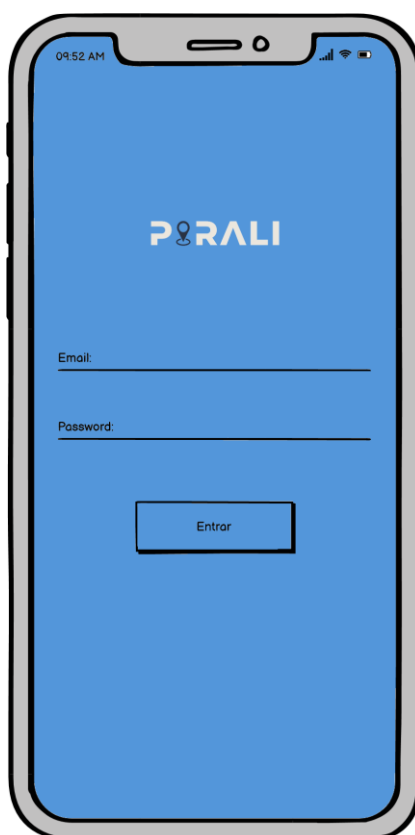


Figura 20: *Interface Login*

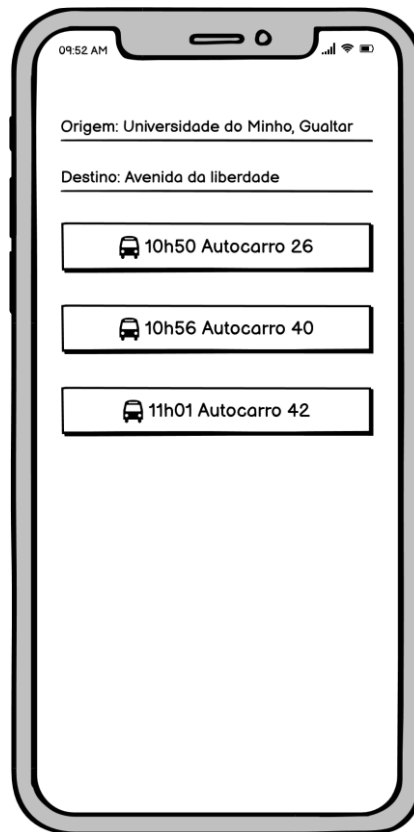


Figura 21: *Interface Iniciar Viagem*

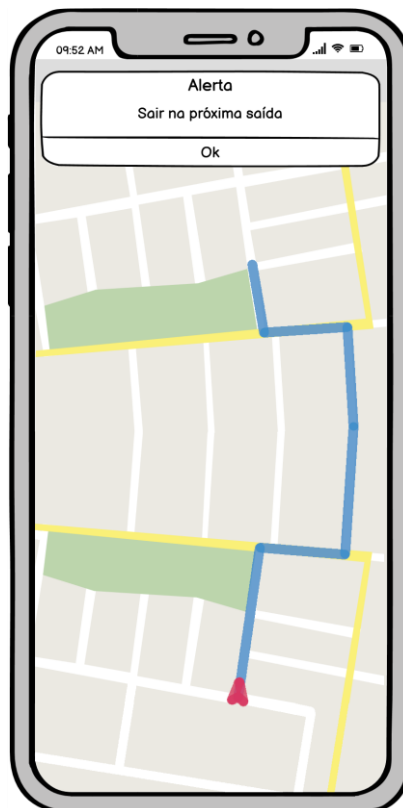


Figura 22: *Interface Notificação*



Figura 23: Interface Conta de utilizador

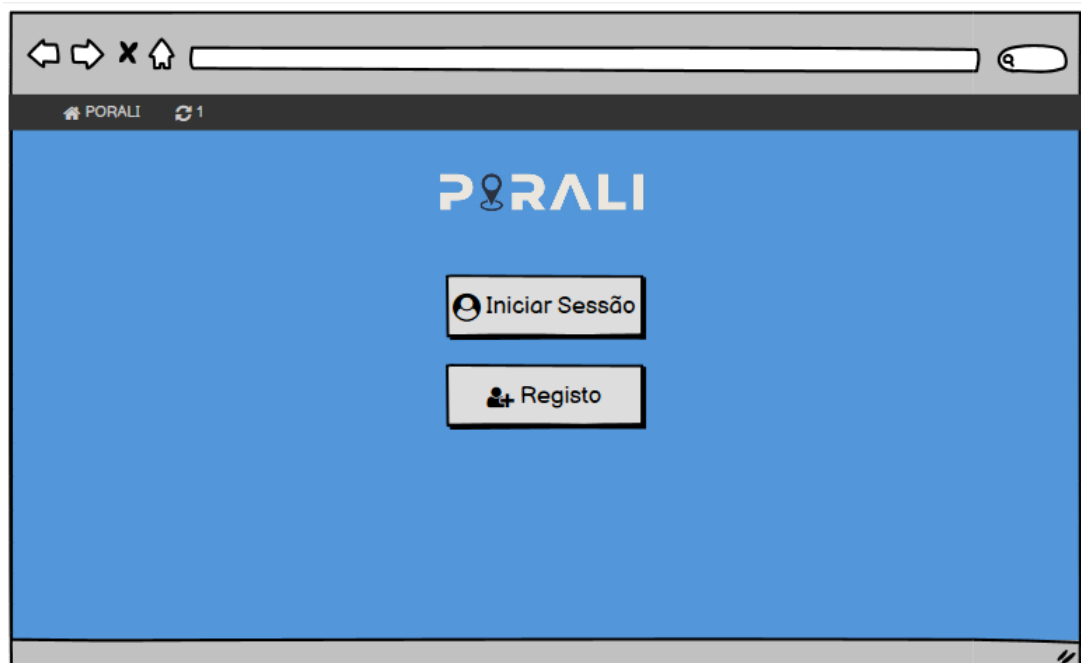


Figura 24: Interface WEB do Login



Figura 25: Interface WEB para a consulta de funcionalidades

15. Metodologias de implementação

De modo a organizar melhor o projeto e permitir um maior paralelismo no seu desenvolvimento, foi escolhido o padrão arquitetural MVC (*Model View Controller*).

Assim, a implementação de uma aplicação multiplataforma foi bastante simplificada, visto que deste modo a interface gráfica é independente de toda a camada de lógica de negócio.

Para além disso o *model* foi subdividido em duas camadas, a camada de dados e a camada de negócio. Desta forma a disponibilidade, consistência e integridade dos dados em uso é mantida independentemente da tecnologia de base de dados utilizada.

16. Ferramentas Utilizadas

Para a realização deste projeto foi necessário recorrer a diferentes ferramentas para a sua implementação. A principal ferramenta utilizada foi a *framework* de desenvolvimento *ASP.NET*. Desta utilizamos o *blazor* que nos permite realizar aplicações recorrendo a C#, *HTML* e *CSS*.

Para a implementação da base de dados foi utilizada a ferramenta *MySQL* devido ao facto de já estarmos familiarizados com a mesma.

No que toca ao envio de notificações optamos pelo o *blazor toast*, que usando *javascript* permite o envio de diversos tipos de notificações. Além disso, para mostrar a lista de notificações durante a viagem e enquanto o utilizador estiver a utilizar outras funcionalidades foi utilizada o *blazor Modal* que permite aceder a uma página em formato *popup*.

Na realização de algumas funcionalidades do sistema foram usadas duas *APIs* externas do *google maps*, a *API Directions* e *API Distance Matrix*.

De modo a unificar todas estas ferramentas numa aplicação apenas, foi utilizado o *IDE Visual Studio 2019* da *Microsoft*.

Para finalizar a aplicação no seu desenvolvimento foi sido testada em diversos *browsers*, nomeadamente o *Google Chrome*, *Microsoft Edge* e *Brave*.

17. Desenvolvimento do projeto

17.1. Conexão Base de Dados

A base de dados foi desenvolvida utilizando a plataforma *MySQL* que por sua vez foi conectada ao projeto C# utilizando uma classe de biblioteca *.NET Standard*. A separação entre estas duas componentes visa manter o bom funcionamento da aplicação e uma boa prática em desenvolvimento *WEB*.

Dentro desta classe que funcionará como conexão entre a BD e o projeto C# criamos dois tipos de métodos: um que faz o download de conteúdo da BD e outro que faça o upload do conteúdo para o sistema de base de dados. Estes dois métodos vão ser fundamentais no desenvolvimento de *queries SQL* que serão utilizadas sempre que queremos aceder à BD.

Como forma de testar o correto funcionamento da aplicação foi desenvolvido um povoamento de teste com dois utilizadores. Desta forma, conseguimos testar as funcionalidades de Login, Registo, Histórico, ect. mais facilmente, pois são essas funcionalidades que requerem mais controlo na BD do que na aplicação propriamente.

Para acedermos ao conteúdo proporcionado pela base de dados foi necessário desenvolver *queries* de *SQL* para aceder, inserir, atualizar e eliminar informação. Essas *queries* são passadas para a base de dados através do método de download e upload dependendo do tipo de *query* em questão. Desta forma, conseguimos ter um acesso à base de dados de fácil implementação, eficaz e sem ter muita preocupação de como essa informação é transformada nas classes, devido ao uso da biblioteca *Dapper* para a implementação de todas as conexões.

17.2. Obtenção dos Autocarros

Dada a natureza do nosso projeto, foi necessário recorrer a *APIs* que fornecessem informações relativas aos autocarros existentes num dado instante, de modo a fornecer opções de escolha e diferentes percursos ao utilizador. Deste modo, tal como planeado nas fases anteriores, utilizou-se *APIs* da *google*, nomeadamente a *API Directions* e a *API Distance Matrix*.

A *API Directions* foi utilizada com o propósito de aceder aos autocarros existentes entre uma dada origem e destino em tempo real. Assim, o procedimento começou por efetuar um pedido *HTTP Request* contendo os parâmetros adequados e pretendidos. A resposta proveniente deste pedido é recebida em formato *JSON* e, por isso, foi necessário efetuar a sua leitura de modo a retirar apenas as informações relevantes para este projeto, ou seja, para cada percurso encontrado, foram armazenadas apenas as informações relativas aos autocarros que o constituem, tais como o número do autocarro, número de paragem, paragens inicial e final, tempo de partida e tempo de chegada. De acordo com estes critérios é possível dar resposta a diversas funcionalidades que nos propusemos a implementar.

No que toca à *API Distance Matrix*, esta foi utilizada de modo a obter o tempo que decorre entre dois pontos. Esta informação foi necessária de modo a calcular a hora em que o autocarro chega à penúltima paragem, de forma a poder notifica-lo. Para isso, foi crucial armazenar algumas rotas de autocarros exemplificativas. Deste modo, dadas as paragens inicial e final é possível calcular qual a penúltima paragem, e de seguida efetuar o cálculo do tempo de viagem que decorre entre a penúltima e última paragens. Esta informação permitirá notificar o utilizador que chegou à penúltima paragem no momento certo.

17.3. Obtenção de Notificações

Para a existência das notificações foi necessário a instalação e implementação do *Blazor Toast*. Este, permite obter mensagens *popup* no ecrã que posteriormente serão armazenadas na lista de notificações. Desta forma, conseguimos implementar um botão que acede a uma página *popup* com as várias notificações do utilizador. Esta página de notificações foi implementada com uso do *Blazor Modal* que permite, através de um clique, abrir uma página em cima da atual, mostrando desta forma as várias notificações sem que o utilizador tenha de navegar para a próxima página.

As notificações foram implementadas com recurso a *threads* que permitem a programação de notificações no ecrã passado um determinado período de tempo. Esta funcionalidade é importante para a notificação de início de viagem, fim de viagem e de penúltima paragem. Assim, conseguimos obter uma aplicação que reage a eventos, neste caso, à passagem do tempo.

18. Produto Final

A nossa equipa passou o semestre a realizar o Porali – *Transport Tracker* até conseguir chegar à construção final da mesma. Na presente secção iremos apresentar um guia de como utilizar a aplicação recorrendo a *screenshots* da mesma.

18.1. Pré-Authenticação

Caso um utilizador aceda à aplicação e não contenha sessão iniciada vai-se deparar com um menu com diversas opções, quer na barra superior quer botões como se encontra em seguida:



Figura 26: Menu Autenticação

18.1.1. Barra Superior

Na barra superior existem três opções, uma para voltar para o menu da figura 26 (casa), uma para aceder aos horários fixos da TUB (corrente) e uma para aceder à pagina da câmara de Braga (globo).

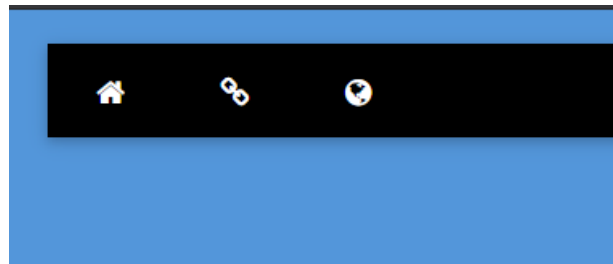


Figura 27: Menu Autenticação - Barra Superior

18.1.2. Iniciar Sessão

Caso se carregue no botão “Iniciar sessão” somos levados para uma outra página onde podemos inserir o nosso *email* e *password*, a sessão é iniciada caso haja uma correspondência guardada na base de dados.

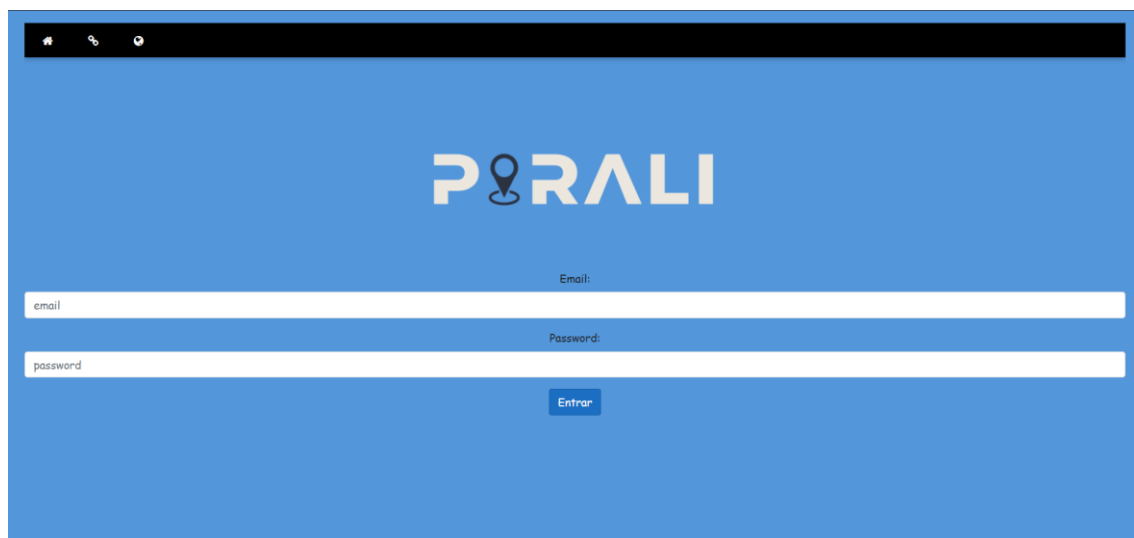
A imagem mostra a interface de login. No topo, há uma barra preta com os mesmos três ícones da barra superior. Abaixo, o fundo é azul. No centro, o logotipo "P@RALI" é exibido em branco. Abaixo do logotipo, há dois campos de entrada brancos. O primeiro campo é rotulado "Email:" e o segundo "Password:". Ambos os campos contêm o texto cinza "email" e "password" respectivamente. Abaixo dos campos, há um botão azul com o texto "Entrar" em branco.

Figura 28: Menu Iniciar Sessão

18.1.3. Registo

O botão “Registo” leva-nos para uma página que contém os campos necessários para realizar o mesmo. Tanto o email, a password e nome são campos obrigatórios, enquanto que o código de amigo pode ser ou não inserido. Após o registo somos encaminhados para o menu de iniciar sessão.


The image shows a web browser window with a blue background. At the top, there is a black navigation bar with three icons: a home icon, a magnifying glass, and a user profile icon. Below the navigation bar, the logo "PORALI" is centered, with a location pin icon integrated into the letter "O". The registration form consists of four white input fields stacked vertically. The first field is labeled "Email:" and contains the placeholder text "email". The second field is labeled "Password:" and contains the placeholder text "password". The third field is labeled "Nome:" and contains the placeholder text "Nome". The fourth field is labeled "Opcional - Código Amigo:" and contains the placeholder text "Código Amigo". Below the input fields, there is a blue button with the text "Registrar" in white.

Figura 29: Menu Registo

18.2. Pós-Autenticação

Quando é iniciada a sessão deparámo-nos com o menu da figura 30. Nele conseguimos encontrar diversas opções, quer na barra superior quer nos botões apresentados.



Figura 30: Menu Principal

18.2.1. Barra Superior

Nesta barra existem as seguintes funcionalidades:

- **Casa:** Retorna para o menu da figura 18.2;
- **Bilhete:** Mostra o menu das viagens grátis (figura 32);
- **Ciclo:** Mostra o registo das viagens efetuadas pelo utilizador (figura 34);
- **Sair:** Realiza o *logout* da aplicação.
- **Estrela:** Permite avaliar a aplicação (figura 35);
- **Sino:** Permite ver as notificações (figura 36);
- **Utilizador:** Permite configurar o nome e a password .

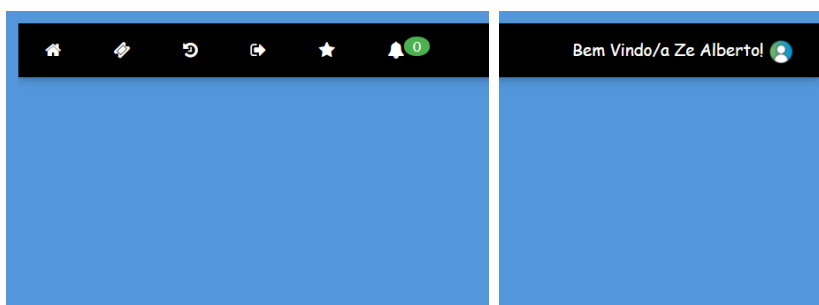


Figura 31: Menu Principal - Barra Superior

18.2.1.1. Viagens Grátis (bilhete)

Neste menu é realizada uma listagem de todas as viagens grátis que o utilizador pode usar nos transportes da TUB.



Figura 32: Menu Viagens Grátis - 1

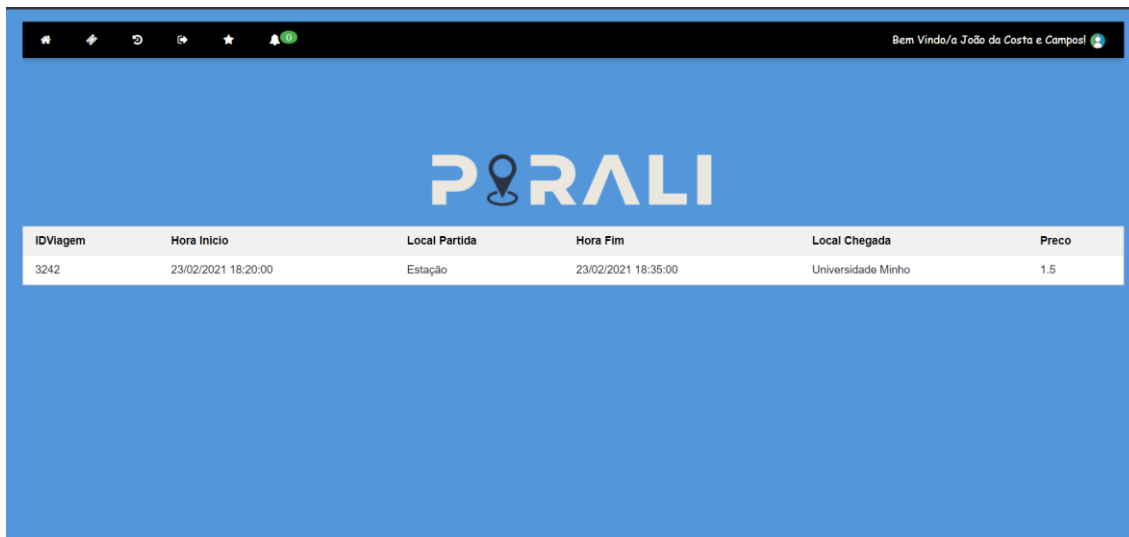
Uma viagem pode ser redimida clicando no botão “Redimir”. Após realizar esta ação o código é ativado e mostrado o seguinte menu:



Figura 33: Menu Viagens Grátis - 2

18.2.1.2. Histórico de Viagens (ciclo)

Neste menu é apresentado o histórico de todas as viagens realizadas pelo utilizador enquanto utiliza a aplicação.



IDViagem	Hora Inicio	Local Partida	Hora Fim	Local Chegada	Preço
3242	23/02/2021 18:20:00	Estação	23/02/2021 18:35:00	Universidade Minho	1.5

Figura 34: Menu Histórico de Viagens

18.2.1.3. Avaliar a aplicação (estrela)

Neste menu é possível o utilizador avaliar a aplicação de 1 a 5 estrelas consoante a sua experiência com a mesma.



Figura 35: Menu Avaliação Aplicação

18.2.1.4. Ver as notificações (sino)

Neste menu é possível obter a lista das notificações recebidas durante esta sessão de utilização. Desta forma é possível rever notificações que de outra forma teriam sido perdidas.

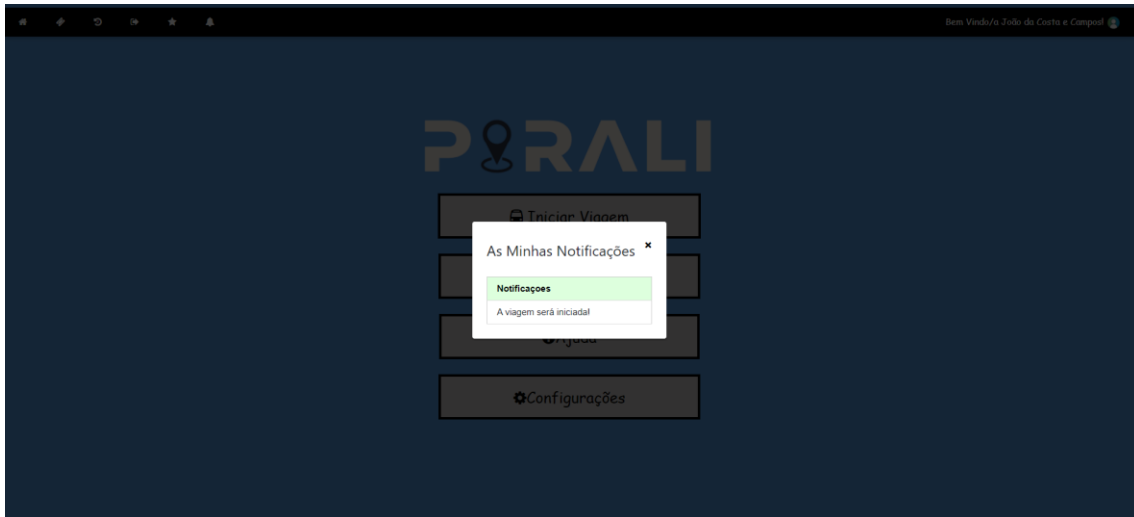


Figura 36: Ver Notificações

18.2.1.5. Editar Definições (utilizador)

Aqui é possível alterar as definições pessoais do utilizador, mais propriamente a sua *password* e o seu nome.

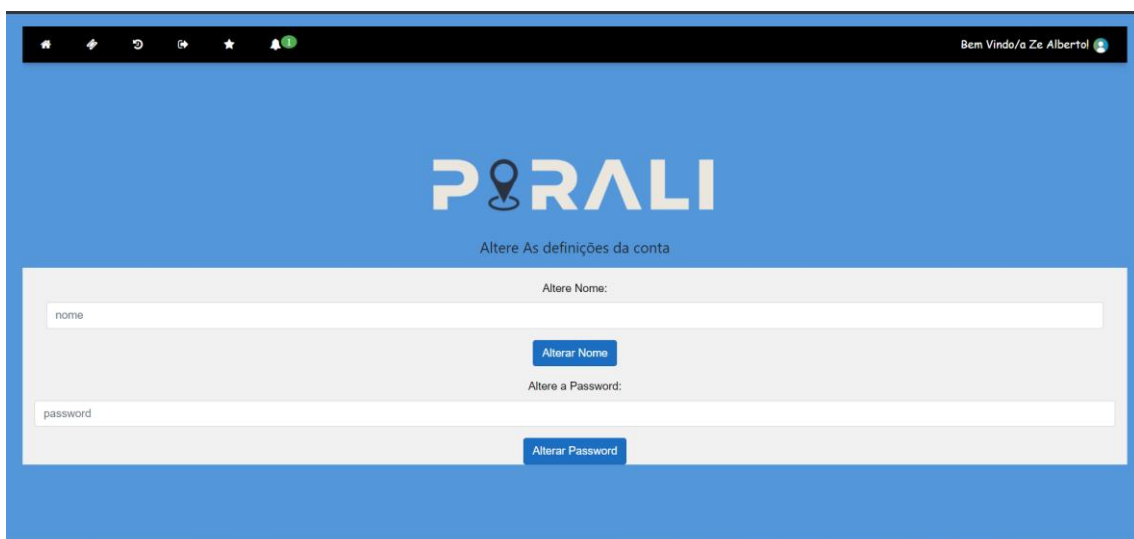


Figura 37: Editar Definições

18.2.2. Iniciar Viagem

É a partir deste menu que podemos recorrer à funcionalidade principal da nossa aplicação, ou seja, acompanhar uma viagem. Primeiramente no menu seguinte é necessário indicar a origem e o destino pretendidos.

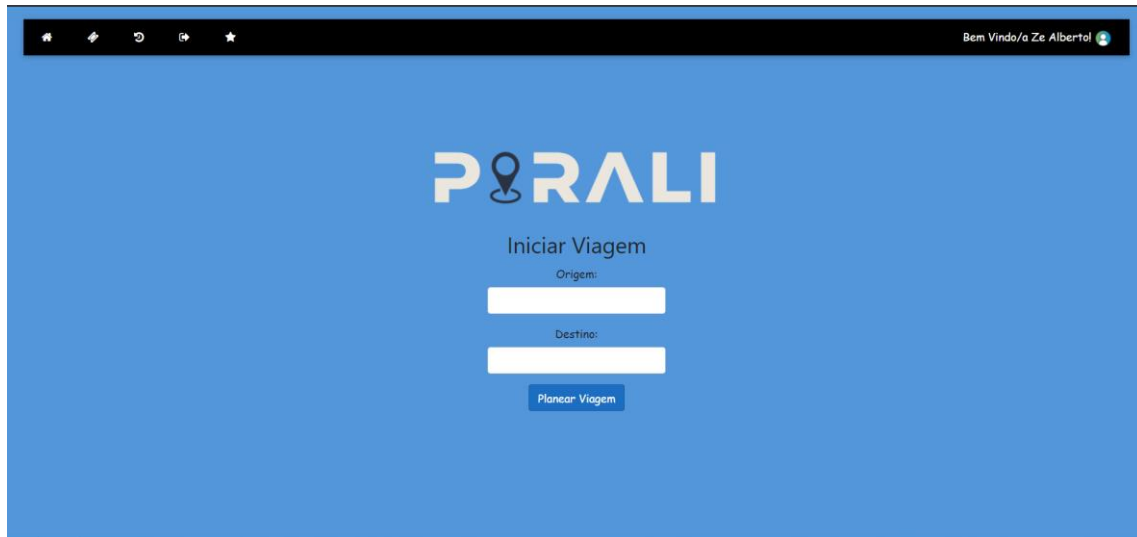


Figura 38: Menu Iniciar Viagem

Após a inserção de uma origem e de um destino o sistema mostra os autocarros disponíveis para alcançar o objetivo:

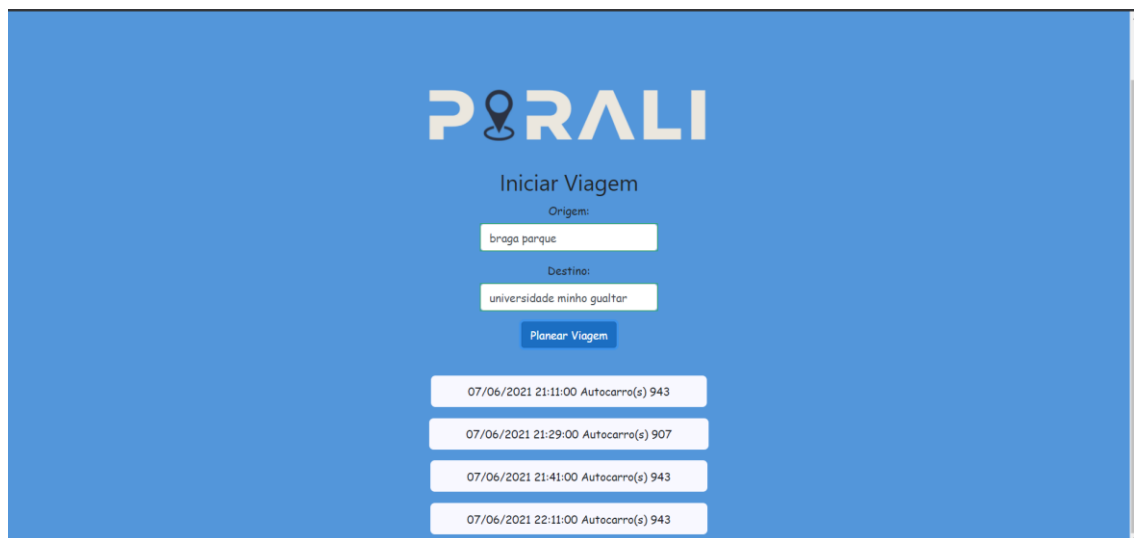


Figura 39: Menu Iniciar Viagem - 2

Ao ser selecionada uma viagem aparece o mapa com a ligação entre os dois pontos pretendidos para além de alguma informação extra como preço, número de paragens e locais e horas das paragens.

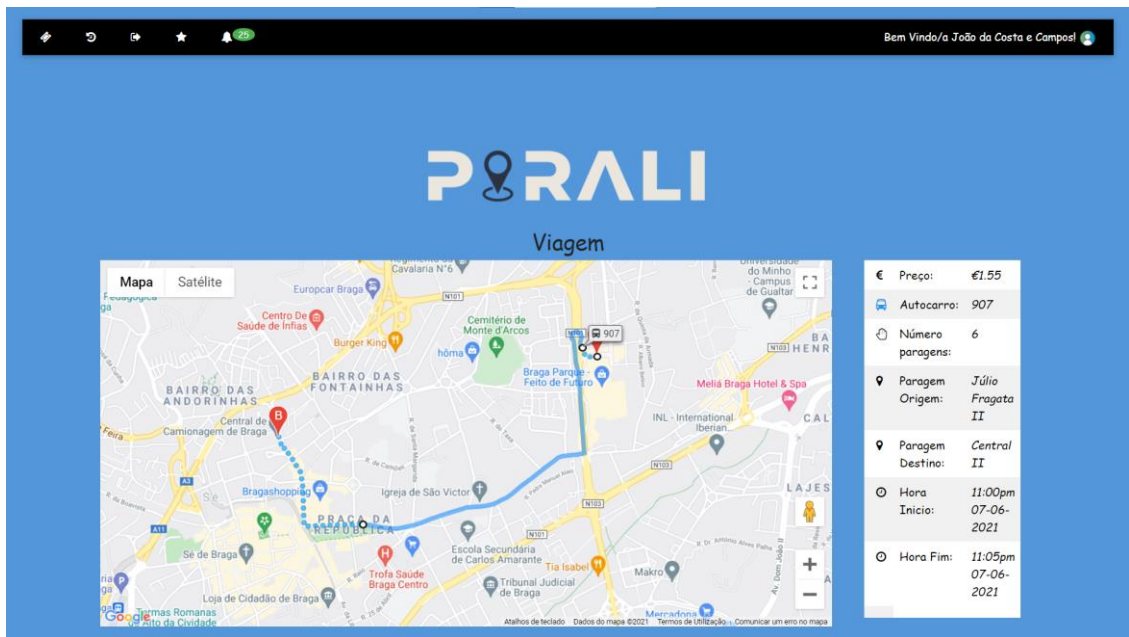


Figura 40: Menu Iniciar Viagem - 3

Ao receber alguma notificação, como por exemplo o momento em que o autocarro sai da paragem aparecem da seguinte forma:

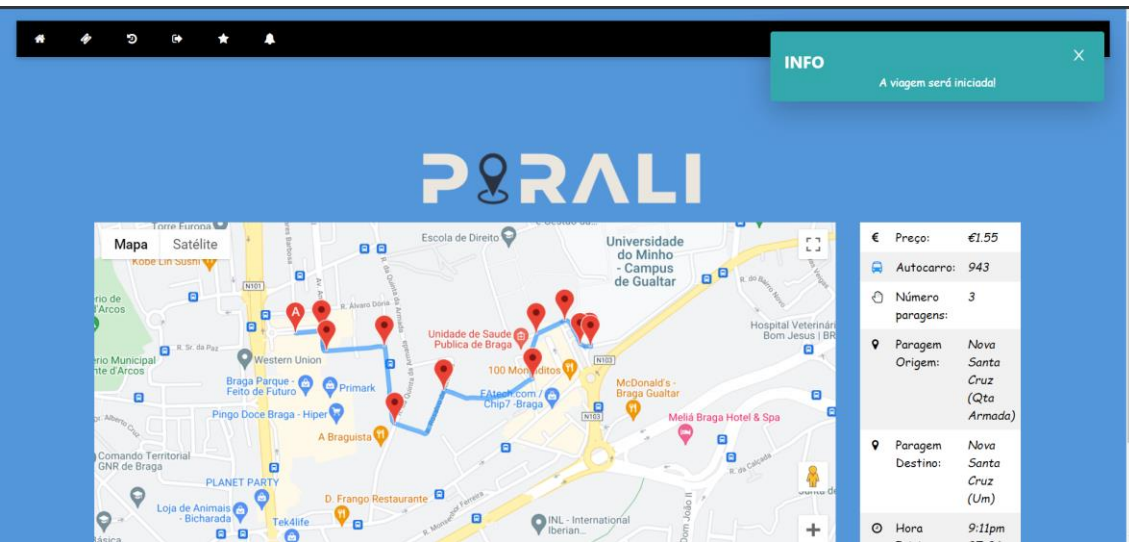
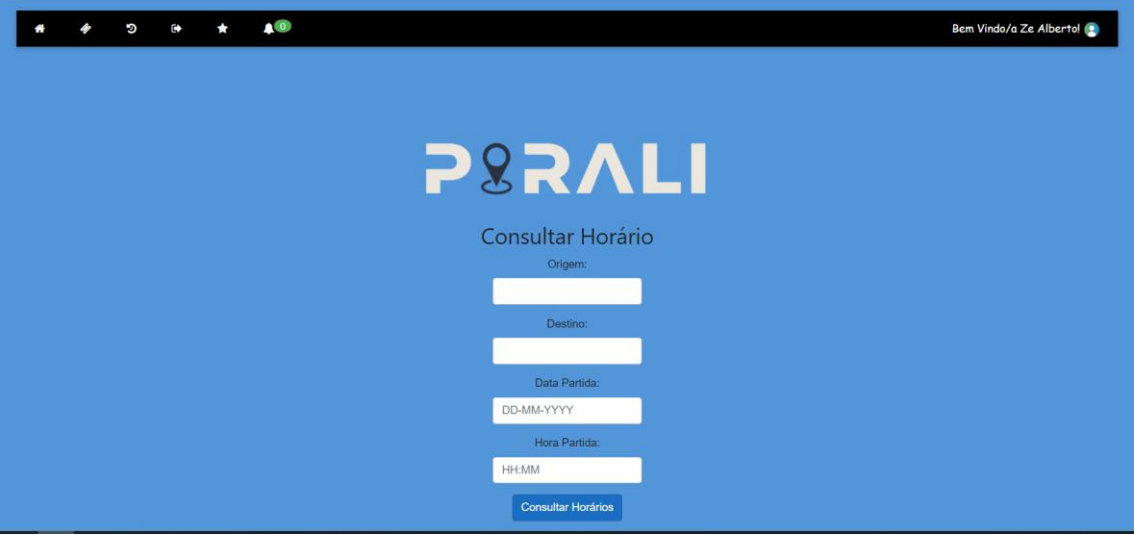


Figura 41: Menu Iniciar Viagem - 4

18.2.3. Consultar Horários


Este menu tem como função planear uma viagem futura. Para tal é necessário inserir a origem e destino pretendidos, a data de partida e a hora pretendida.



The screenshot shows the 'Consultar Horário' (Consult Schedule) form on the POrALI website. The form is centered on a blue background. It includes input fields for 'Origem:' (Origin), 'Destino:' (Destination), 'Data Partida:' (Departure Date) with a 'DD-MM-YYYY' placeholder, and 'Hora Partida:' (Departure Time) with a 'HH:MM' placeholder. A blue button labeled 'Consultar Horários' is positioned below the form fields. The top of the page features a black navigation bar with various icons and a user greeting 'Bem Vindo/a Ze Alberte!' on the right.

Figura 42: Menu Consultar Horário - 1

Após a inserção dos campos aparecem os autocarros disponíveis para realizar a viagem pretendida como se encontra em seguida:



This screenshot displays the results of the 'Consultar Horário' search. The form fields from the previous screen are now populated with specific values: 'braga parque' for Origin, 'universidade minho gualta' for Destination, '10-07-2021' for the departure date, and '11:30' for the departure time. The 'Consultar Horários' button remains visible. Below the form, a list of available buses is shown, each in a white box with a blue border. The list includes the date and time of departure followed by the number of buses available.

Data Partida	Hora Partida	Autocarro(s)
10/07/2021	11:46:00	7
10/07/2021	11:50:00	74
10/07/2021	12:20:00	74
10/07/2021	12:30:00	40

Figura 43: Menu Consultar Horário - 2

18.2.4. Ajuda

O menu de ajuda serve para dizer o que, no menu principal, corresponde a cada um dos botões principais presentes.

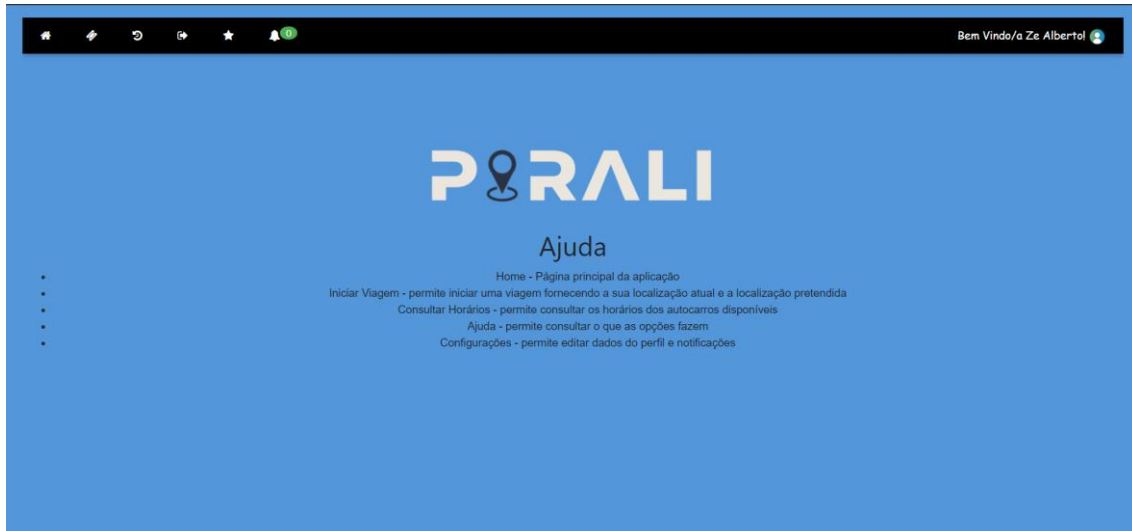


Figura 44: Menu Ajuda

18.2.5. Configurações

Este menu permite ao utilizador escolher as notificações que pretende receber aquando da realização de uma viagem.

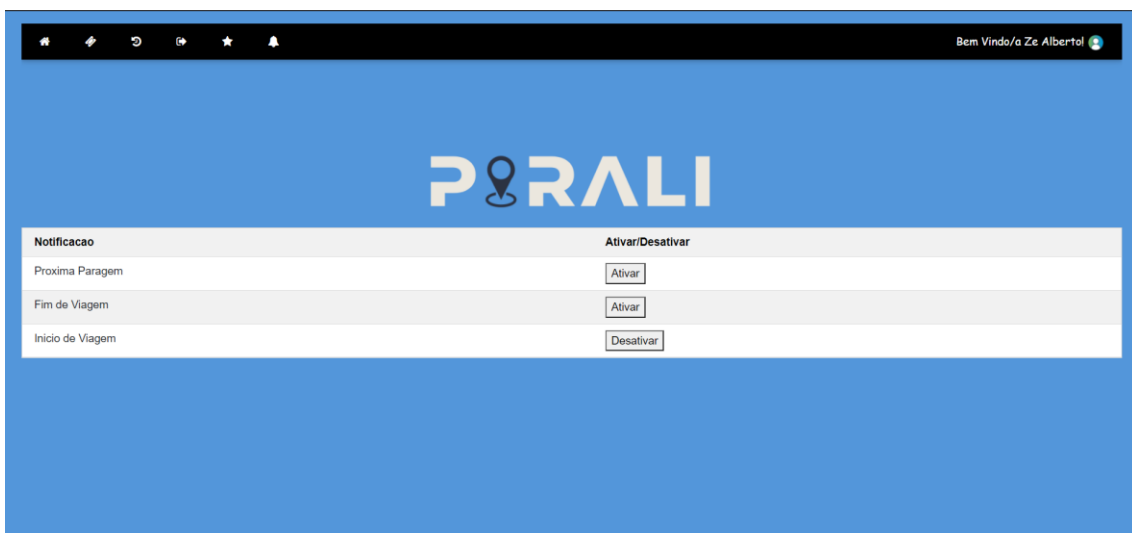


Figura 45: Menu Configurações

19. Atualizações

19.1. Fase 1

Na primeira fase do projeto, apresentamos duas vertentes da aplicação: passageiro e gestor. Contudo, nesta segunda fase, o foco incidiu sobre as funcionalidades e interface da aplicação para o passageiro, dado que se trata do público-alvo do caso de estudo a abordar. Deste modo, equacionando o tempo em mãos e as responsabilidades estipuladas do sistema, na terceira fase, a implementação passará apenas pelas funcionalidades dos passageiros. Assim, damos margem para expandir a aplicação no futuro ao implementar a interface do gestor e as suas inerentes funcionalidades.

Salientamos ainda a atualização do *mockup* publicitário e da maqueta apresentada na fase antecessora, que agora inclui um esquema mais representativo da arquitetura base do sistema.

19.2. Fase 2

Na segunda fase do projeto, apresentamos um modelo lógico e conceptual, no entanto no decorrer do trabalho apercebemo-nos que as tabelas rota, possui e paragem não eram relevantes, de modo que foram removidas dos mesmos diagramas. Além disso, também consideramos irrelevante, agora que retiramos as tabelas anteriormente mencionadas, as tabelas de autocarro e consequentemente a tabela efetuada que vazia a associação entre viagem e autocarro.

No que toca a implementação do diagrama de classes no projeto, este teve de ser atualizado de forma a conseguir aplicar no projeto planeado às mudanças realizadas na base de dados e na implementação do código. Assim, o diagrama de classes e os diagramas sequenciais são exemplificativos dos métodos, classes e fluxo que foram implementados.

20. Conclusões

A construção de uma aplicação cujo objetivo era monitorar ao minuto algum tipo de acontecimentos revelou-se em primeiro lugar um enorme desafio. No entanto, observando um pouco ao nosso redor e rapidamente percebemos que haviam certos aspetos das nossas rotinas diárias que poderiam ser melhoradas. Assim chegámos à ideia de realizar uma aplicação para a TUB, de modo a permitir uma viagem mais relaxada para os seus passageiros.

Na primeira fase do projeto, a **Fundamentação**, foi necessário fazer inúmeras pesquisas de modo a consolidar o nosso conhecimento à cerca dos transportes urbanos, mais concretamente aos autocarros da cidade de Braga. Foi necessário perceber como a TUB opera na cidade e assim perceber quais as funcionalidades mais pretendidas pelos seus passageiros. Este passo foi fulcral uma vez que foi o primeiro passo para a compreensão e modelação da aplicação.

De seguida foi desenhada uma maquete do sistema de modo a podê-lo demonstrar de uma forma mais simples. Isto foi fundamental para, numa fase futura, ser mais fácil a interpretação da comunicação entre aplicação e utilizador.

Para realizar o planeamento do projeto foi realizado um diagrama de *Gantt*. Este demonstrou-se fulcral para organizarmos o nosso tempo durante a realização de todo o projeto.

Desta forma, sentimos que no final da primeira fase, a fundamentação do nosso projeto ficou clara e ilustrativa, de modo a justificar o prosseguimento do seu desenvolvimento.

Numa segunda etapa surge a **Especificação**, onde através dos requisitos levantados conceberam-se : o modelo de domínio, as máquinas de estados, a arquitetura da aplicação, os diagrama de use cases, atividades, componentes, classes e sequência, a camada de dados e as *mockups* da aplicação. Sempre em vista as características necessárias para a implementação futura das funcionalidades.

O modelo de domínio é simples e representativo, desta forma, permite a identificação correta entre relacionamentos e entidades Indo de encontro aos objetivos pré-estabelecidos. O diagrama de use cases permitiu organizar todas as funcionalidades do sistema num único lugar, o que contribui para a organização do projeto. A máquina de estados conseguiu refletir de forma bastante precisa e no formato de diagrama o *use case* iniciar viagem, visto ser o principal aspeto da aplicação a par com as notificações. Os diagramas de componentes, que fornecem uma visualização de todos os componentes constituintes do projeto; diagramas de sequência com alguns dos métodos núcleo da nossa aplicação; diagramas de classes que

forneem uma visão geral de todas as classes necessárias na implementação; diagrama de atividades para representar o fluxo entre os diferentes menus do programa; camada de dados com estrutura para suportar as classes persistidas na aplicação. Além disso, arquitetura da aplicação permitiu demonstrar os vários componentes da aplicação. Por fim, as *Views* da Aplicação serviram para demonstrar a interface futura da aplicação.

Após toda a formulação dos objetivos e motivações que levaram a criação desta plataforma na fase da fundamentação e todos os diagramas, requisitos e modelação realizada na fase de especificação chegou-se, finalmente à fase de **implementação e construção** de todo o planeamento realizado nas fases anteriores. Nesta fase o trabalho de toda a equipa focou-se no cumprimento das metas estabelecidas nas fases precedentes tais como a implementação de todas as funcionalidades e requisitos, o desenvolvimento de uma interface de fácil leitura, acesso e manuseamento e na implementação de todos os componentes e plataformas necessários à criação da aplicação.

Desta forma consideramos pertinentes fazer uma pequena análise do trabalho realizado e do produto final obtido. É de salientar que todos os requisitos e funcionalidades foram implementadas no trabalho e se encontram funcionáveis. Além disso, seguimos o planeamento feito para a criação da interface, tal como está na secção 14, cumprindo desta forma um dos requisitos impostos na fase 2. É conveniente ainda mencionar que temos seguir os diagramas e modelos implementados através da criação de classes e sua organização tal como está nas secções 11 e 12.

Por outro lado, consideramos que existem pontos a melhorar neste projeto, tais como a existência de determinados bugs no programa, a associação de uma viagem grátis a uma viagem normal e a possibilidade de monitorizar as localizações dos autocarros em tempo real. Neste projeto não foi possível o *tracking* dos autocarros dado que era necessário outro tipo de ferramentas às quais não tivemos acesso. O programa implementado tem margem para expansão no futuro, desde a adição de novas funcionalidades até à implementação de versão mobile. Além disso, existe a possibilidade futura de implementar uma versão para o gestor e também adaptar a nossa app a várias cidades, além da cidade de Braga.

Após o longo processo de planeamento e desenvolvimento WEB o produto final é funcional satisfaz os requisitos e casos de estudo e principais características a garantir num projeto desta dimensão. Acima de tudo, é uma aplicação de fácil manuseamento e bastante interativa com o utilizador que será uma mais-valia nosso aos passageiros de transportes públicos como também às empresas como a TUB.

Referências

Albuquerque, R., 2020. *Expresso*. [Online]

Available at: <https://expresso.pt/sociedade/2020-02-03-Numero-de-passageiros-a-viajar-com-passe-nos-transportes-publicos-aumentou-15>

[Acedido em 6 março 2021].

Albuquerque, R., 3 fevereiro 2020. *Número de passageiros a viajar com passe nos transportes públicos aumentou 15%*. [Online]

Available at: <https://expresso.pt/sociedade/2020-02-03-Numero-de-passageiros-a-viajar-com-passe-nos-transportes-publicos-aumentou-15>

Google, s.d. *Google Maps Platform*. [Online]

Available at: <https://developers.google.com/maps/gmp-get-started>

[Acedido em 28 4 2021].

IMT, 2019. *IMT*. [Online]

Available at: <http://www.imt-ip.pt/sites/IMTT/Portugues/Noticias/Paginas/Programa-PART.aspx>

[Acedido em 7 março 2021].

My SQL, s.d. *MySQL 8.0 Reference Manual*. [Online]

Available at: <https://dev.mysql.com/doc/refman/8.0/en/>

[Acedido em 8 05 2021].

Nunes, D. F., 2021. *Dinheiro Vivo*. [Online]

Available at: <https://www.dinheirovivo.pt/empresas/procura-de-transportes-publicos-em-minimo-historico-em-2020-13437989.html>

[Acedido em 10 março 2021].

Nunes, D. F., 2021. *Procura de transportes públicos em mínimo histórico em 2020*. [Online]

Available at: <https://www.dinheirovivo.pt/empresas/procura-de-transportes-publicos-em-minimo-historico-em-2020-13437989.html>

[Acedido em 5 março 2021].

Silva, R. J., 2021. *Publico*. [Online]

Available at: <https://www.publico.pt/2021/03/12/local/noticia/transportes-publicos-acompanham-desconfiamento-medidas-especiais-1954227>

[Acedido em 6 março 2021].

teamgantt, 2020. *teamgantt*. [Online]

Available at: <https://www.teamgantt.com/what-is-a-gantt-chart/how-to-make-a-gantt-chart>

[Acedido em 18 março 2021].

Lista de Siglas e Acrónimos

TUB	Transportes Urbanos de Braga
BD	Base de Dados
UC	Unidade Curricular
UI	<i>User Interface</i>
IDE	<i>Integrated development environment</i>
MVC	<i>Model View Controller</i>
WEB	<i>World wide web</i>
SQL	<i>Structured Query Language</i>
API	<i>Application Programming Interface</i>
HTML	<i>HyperText Markup Language</i>
CSS	<i>Cascading Style Sheets</i>