UMinho

Mestrado Engenharia Informática

Requisitos e Arquiteturas de Software (2023/24)

PROBUM v.1

Grupo C, PL1 / Entrega 2

Ana Filipa da Cunha Rebelo (pg53624)
Diogo Filipe da Costa Marques (pg52678)
Eduardo Francisco Longras Figueiredo (pg52679)
Eduardo José Azevedo Ferreira Araújo (a86012)
Gonçalo Campos Pereira (pg53834)
João Pedro Campos Rodrigues (pg52687)
Pedro Marcelo Bogas Oliveira (pg54144)
Ricardo Lopes Lucena (pg54187)
Robert Beniamin Szabo (pg54194)
Santiago Vicente Ferreira Fernandim Domingues (pg54225)
Telmo José Pereira Maciel (pg54246)

Braga, 28 de janeiro de 2024

Índice

Capítulo 1

Prefácio

No decorrer da realização deste documento surgiram algumas dificuldades nomeadamente no desenvolvimento de alguns diagramas tais como o Diagrama de *Deployment* dado que não tínhamos tido contacto ainda e com alguns diagramas de sequência devido à sua complexidade.

Por fim,consideramos importante analisar o contributo de cada elemento do grupo para o desenvolvimento do trabalho. Deste modo, cada elemento é classificado com + para alunos que contribuíram significativamente acima da média, 0 para alunos cujo contributo foi próximo da média e - para alunos que contribuíram significativamente abaixo da média:

- Ana Filipa da Cunha Rebelo (pg53624):0
- Diogo Filipe da Costa Marques (pg52678):0
- Eduardo Francisco Longras Figueiredo (pg52679):0
- Eduardo José Azevedo Ferreira Araújo (a86012):0
- Gonçalo Campos Pereira (pg53834):0
- João Pedro Campos Rodrigues (pg52687):0
- Pedro Marcelo Bogas Oliveira (pg54144):0
- Ricardo Lopes Lucena (pg54187):0
- Robert Beniamin Szabo (pg54194):0
- Santiago Vicente Ferreira Fernandim Domingues (pg54225):0
- Telmo José Pereira Maciel (pg54246):0

Capítulo 2

Introdução e Objetivos

O presente documento visa apresentar a arquitetura definida para o projeto ***Probum***, assim como demonstrar as justificações para a escolha da mesma. Sendo este documento a continuação do documento de requisitos, irá ser feita apenas uma breve contextualização da aplicação em questão, sendo que a sua apresentação e explicação foi feita no documento anterior.

Nos últimos anos, tem-se assistido a uma crescente integração da tecnologia no ambiente educativo e esse fenómeno não passa despercebido no ensino superior. Com o objetivo de otimizar processos e melhorar a experiência de Alunos e Docentes, muitas organizações têm explorado a digitalização de várias atividades, incluindo a realização de provas de avaliação. Propomos então o desenvolvimento de um produto, o *Probum*, que tenha algumas características diferenciadoras e até inovadoras, que aborda todos estes problemas, relacionados com a realização de provas de avaliação, de forma integrada. O *Probum* permitirá:

- Criação de provas de avaliação digitais;
- Calendarização inteligente das provas;
- Realização das provas;
- Consulta das provas;
- Correção das provas.

Desta forma, está intrinsecamente associada à aplicação a contribuição para o aumento da sustentabilidade, um ponto de extrema importância nos tempos em que vivemos. Uma vez mais, todo o contexto que envolve a aplicação pode ser lido de forma bastante mais aprofundada no documento de requisitos.

2.1 Objetivos

O *Probum* tem como principal objetivo proporcionar uma solução abrangente para otimizar os processos relacionados à administração de avaliações em instituições educacionais. Os objetivos fundamentais desta aplicação são meticulosamente delineados para atender às necessidades específicas dos docentes, alunos e à eficácia geral da administração académica. A seguir, destacam-se esses objetivos de forma estruturada:

- Docentes capazes de criar,editar e gerenciar avaliações de modo eficiente, proporcionando flexibilidade na definição de parâmetros, como questões, datas e critérios de avaliação.
- Ter uma plataforma acessível e intuitiva para os alunos realizarem avaliações online, aprimorando a conveniência e a eficácia do processo avaliativo, poupando assim recursos naturais.
- Disponibilizar aos alunos a capacidade de consultar facilmente os resultados de suas avaliações, promovendo transparência e fornecendo feedback rápido.
- Facilitar a Interatividade entre Docentes e Alunos fortalecendo o acompanhamento do progresso académico.
- Implementar um sistema eficiente de reserva de salas, permitindo que os docentes agendem espaços específicos para a realização de testes de forma organizada.
- Contribuir para a otimização da logística relacionada à realização de testes, garantindo agendamentos eficientes de salas, datas e horários.
- Reduzir a carga administrativa associada à gestão de avaliações e reservas, promovendo uma administração académica mais eficiente.
- Implementar medidas robustas de segurança para garantir a integridade e confidencialidade das avaliações, prevenindo práticas fraudulentas.

Ao atender a esses objetivos, a aplicação visa transformar positivamente a dinâmica académica, proporcionando uma experiência mais eficaz e colaborativa para docentes e alunos, além de contribuir para a eficiência na gestão académica.

2.2 Metas de qualidade

No âmbito deste trabalho, tomamos a decisão de destacar de maneira proeminente os seguintes objetivos de qualidade, levando em consideração os requisitos não funcionais que se apresentam, visando assim assegurar um desempenho otimizado e atender às expectativas estabelecidas.

- O sistema tem grande disponibilidade durante horário letivo
- O Sistema informa se o utilizador já estiver registado
- O Sistema restringe o acesso a funcionalidades de acordo com o utilizador
- Para a realização de uma prova de avaliação, o computador disponibilizado a cada Aluno apenas deve permitir acesso ao *Probum*
- O sistema deve correr em computadores de gama baixa

Esses requisitos são essenciais para garantir a eficácia e a adequação do programa em diversos cenários. A grande disponibilidade durante o horário letivo assegura que o sistema esteja prontamente acessível quando for mais necessário, facilitando o seu uso contínuo para atividades letivas. A capacidade do sistema de informar se um utilizador já está registado contribui para uma experiência mais fluida e segura, promovendo uma gestão eficiente das contas.

A restrição de acesso a funcionalidades com base no perfil do utilizador é fundamental para garantir a segurança e a personalização da interação, adaptando o sistema às necessidades específicas de cada utilizador. No caso da realização de provas de avaliação, a restrição do acesso do computador do aluno apenas ao *Probum* é crucial para evitar interferências externas e garantir a integridade do processo de avaliação.

A compatibilidade do sistema com computadores de gama baixa amplia sua acessibilidade, permitindo que seja utilizado em uma variedade de dispositivos, inclusive aqueles com recursos mais limitados. Isso é especialmente relevante em ambientes letivos, onde a diversidade de hardware pode ser significativa. Assim, todos esses requisitos desempenham papéis fundamentais na otimização do desempenho, na segurança e na acessibilidade do *Probum*.

Capítulo 3

Restrições

As restrições denotam elementos essenciais capazes de moldar a natureza e o desenvolvimento de um projeto, atuando como diretrizes que influenciam cada fase deste processo. Representam condições, parâmetros e limites que se demonstram pontos críticos para a definição, realização e sucesso do projeto. Portanto, conhecer à partida estas restrições é indispensável para o desenvolvimento deste projeto, pois evidenciam características substanciais da sua arquitetura, funcionalidade e viabilidade.

3.1 Restrições à Solução

3.1.1 Infraestrutura

Uma restrição referente à infraestrutura informática do projeto passa por executar a aplicação na infraestrutura atual da respetiva IES. Deste modo torna-se desnecessário o investimento em novos equipamentos, sendo um ponto positivo na gestão da economia da própria IES.

Para tal os componentes de software necessários para o funcionamento da aplicação devem estar instalados em máquinas da IES e todas as funcionalidades da plataforma para os Alunos devem executar em pleno nas máquinas que forem disponibilizadas para as provas de avaliação

3.1.2 Isolamento da aplicação

Todas as máquinas disponibilizadas aos alunos, no momento de realização de qualquer prova de avaliação, devem permitir exclusivamente o acesso ao *Probum*, não permitindo que um Aluno saia do ambiente da aplicação.

Esta restrição visa evitar que um Aluno recorra a outras aplicações, como *WhatsApp*, *Skype*, navegadores *web* durante a realização da sua prova de avaliação.

3.2 Restrições Temporais

- **Descrição:** O documento presente terá de ser entregue, numa fase inicial, até dia 1 de dezembro de 2023.

  Justificação: De forma a poder ser avaliado o estado do projeto nesta fase , é necessário que seja feita uma entrega que contenha a segunda fase deste projeto, que abarca planear e implementar a arquitetura da aplicação a ser desenvolvida.

3.3 Restrições Orçamentais

- **Descrição:** O orçamento total para o desenvolvimento do projeto é de 20 000€ (vinte mil euros), durante um período de 4 meses.

  Justificação: A equipa responsável pelo desenvolvimento do projeto é constituída por onze engenheiros de software. Para além de ter em conta os salários dos elementos, é preciso também a compra de um domínio, bem como de um computador para hospedar todos os dados da aplicação.

Capítulo 4

Requisitos de qualidade

Os requisitos de qualidade foram identificados e delineados no relatório da Fase 1, especificamente no capítulo dedicado aos Requisitos Não Funcionais, distribuídos ao longo de diversas secções para proporcionar uma abordagem abrangente e organizada. Com isto, definimos as nossas metas de qualidade, expostas no ponto 1.2 do presente relatório.

4.0.1 Árvore de Qualidade

A árvore de qualidade apresentada tem como objetivo ilustrar como os distintos requisitos de qualidade são abordados, utilizando exemplos práticos que evidenciam a aplicabilidade dos diferentes parâmetros de qualidade. Para identificar as referências específicas na Fase 1, cada parâmetro foi associado ao número correspondente do requisito não funcional equivalente, valorizando preferencialmente os requisitos abordados nas metas de qualidade.

Figura 1: Árvore de qualidade

Com os requisitos de qualidade expostos, foram construídos cenários de qualidade que descrevem o comportamento esperado dos mesmos.

- **Grande disponibilidade:** Durante o horário letivo, o sistema deve permanecer operacional e acessível em mais de 96% do tempo, garantindo assim que os utilizadores possam utilizar as funcionalidades sem interrupções significativas.
- **Performance em gama baixa:** O sistema deve ser capaz de operar eficientemente em computadores de gama baixa, garantindo uma experiência fluida e acessível para utilizadores com diferentes configurações de hardware.
- **Tempo de resposta:** Qualquer interação entre o utilizador e o sistema, como clicar em botões ou realizar consultas, deve ter um tempo de resposta inferior a 2 segundos, proporcionando uma experiência responsiva e eficiente.
- **Escalabilidade:** O sistema deve ser escalável para lidar com um aumento significativo no número de utilizadores ou volume de dados, garantindo um desempenho consistente mesmo em cenários de crescimento.
- **Verificação Utilizador:** Ao tentar realizar o *login*, o sistema deve fornecer uma notificação imediata indicando se o utilizador já está registado, assegurando uma experiência rápida e esclarecedora.
- **Restrição de Acesso:** O sistema deve permitir que diferentes utilizadores tenham acesso apenas às funcionalidades relevantes às suas permissões, assegurando uma gestão eficaz dos privilégios e uma experiência personalizada.
- **Segurança do Acesso:** O sistema deve permitir que diferentes utilizadores tenham acesso apenas às funcionalidades relevantes às suas permissões, assegurando uma gestão eficaz dos privilégios e uma experiência personalizada.
- **Restrição da Aplicação:** Durante a realização de uma prova de avaliação, o sistema deve garantir que o computador disponibilizado a cada aluno restrinja o acesso exclusivamente à aplicação *Probum*, evitando distrações e garantindo a integridade do processo de avaliação.
- **Ordem notificações:** As notificações geradas pelo sistema devem ser apresentadas aos utilizadores de forma organizada, seguindo uma ordem cronológica, para garantir uma compreensão clara e lógica das informações importantes.

Capítulo 5

Contexto e Âmbito

Neste capítulo, será explorado o âmbito e o contexto do sistema, que permitirá estabelecer limites claros entre o sistema em questão e todas as entidades com as quais ele se comunica. Com esse propósito a equipa decidiu desenvolver um diagrama de contexto, o qual está representado abaixo.

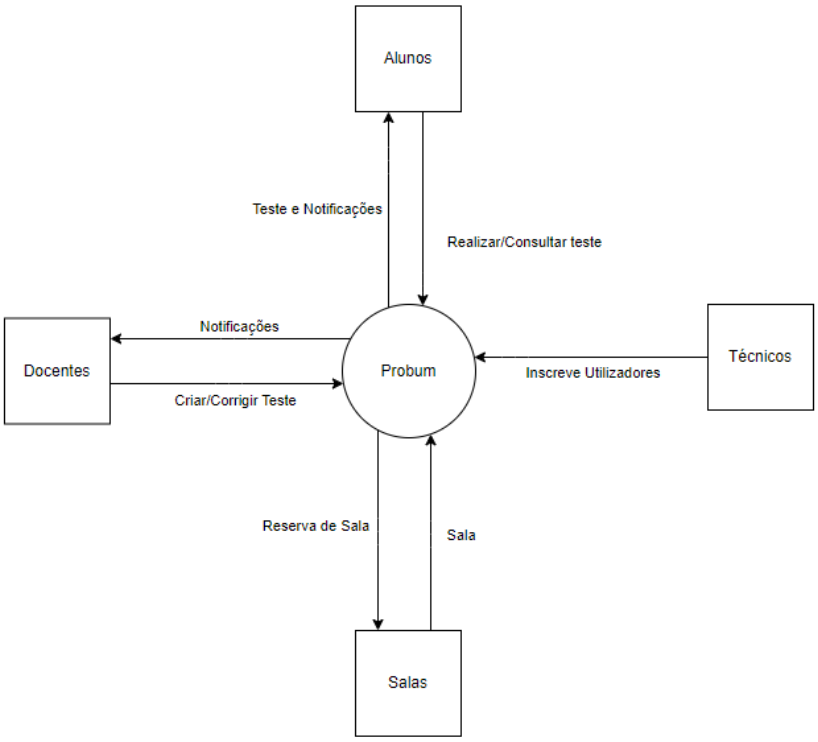

Figura 2: Diagrama de Contexto

Como já foi referido anteriormente no relatório e evidenciado pelo diagrama, os docentes podem criar e gerir testes online, estabelecendo parâmetros como questões, os tipos das mesmas e os respetivos critérios de avaliação. Os alunos, por sua vez, têm a capacidade de realizar esses testes através da plataforma e consultar os resultados correspondentes.

Adicionalmente, a aplicação inclui um sistema de reserva de salas na instituição para a realização dos testes. Os docentes podem agendar salas, especificar datas e horários para a aplicação dos testes, caso os mesmos estejam disponíveis.

Esta abordagem proporciona uma gestão eficaz dos processos de avaliação na instituição educacional, promovendo a interação entre docentes e alunos, além de facilitar a organização logística por meio do sistema de reservas de salas.

Capítulo 6

Estratégia da Solução

No decorrer do desenvolvimento desta fase do projeto,algumas das principais decisões envolveram a escolha das tecnologias a ser utilizadas e a definição da arquitetura do sistema. Considerando todas as metas de qualidade desejadas para o produto final, foi crucial identificar as áreas em que a aplicação precisaria melhorar para alcançar essas metas e garantir a melhor experiência possível para os utilizadores.

Deste modo, as secções a seguir destacam as soluções propostas durante o desenvolvimento da aplicação para lidar com os problemas identificados em relação à qualidade do produto.

6.1 Objetivos de Qualidade

6.1.1 Acessibilidade

- Cenário:Fácil utilização,intuitiva e acessível
- Abordagens de solução:Design centrado nas necessidades do utilizador,acessível para diferentes browsers e em diferentes línguas

6.1.2 Segurança

- Cenário:O utilizador deverá ter os dados guardados de forma segura
- Abordagens de solução:Desenvolver uma estratégia de encriptação de dados do utilizador

6.1.3 Desempenho

- Cenário:O produto deve ser capaz de responder de forma rápida e eficaz
- Abordagens de solução:Desenvolver estratégias para prevenir o *delay* provocado pelas APIs

6.1.4 Manutenção

- Cenário:A aplicação deverá ser capaz de armazenar novos dados e recursos sem causar impacto negativo
- Abordagens de solução:Desenvolver a aplicação com boas práticas de codificação e modularidade do código de modo a permitir a sua manutenção e alteração.

6.2 Estrutura do programa *Probum*

O sistema *Probum* utiliza um diagrama de classes que representa os elementos essenciais do programa a ser desenvolvido. Nesse diagrama, estão representados de forma simples e clara os componentes fundamentais, sendo os mais importantes: as salas, notificações , provas e o utilizador. Para a implementação da aplicação optamos por utilizar a linguagem python/flask para o Backend e Angular para o *Frontend*.
Optamos por utilizar python/flask devido à:

- Facilidade de Aprendizagem e uso
- Flexibilidade e Modularidade
- Escalabilidade
- comunidade Ativa e com muita documentação

Quanto ao Angular para o *Frontend* a sua escolha foi devido a :

- Arquitetura MVC o que ajuda na organização do código e facilita a manutenção e testes do sistema
- Data Binding Bidirecional permitindo a sincronização automática entre os modelos de dados e a interface do utilizador.Qualquer alteração no modelo de dados é refletida instantaneamente na interface e vice-versa.
- Injeção de Dependência:Facilita a criação, manutenção e teste de aplicações, permitindo uma injeção fácil de dependências em componentes, reduzindo a complexidade do código e aumentando a modularidade.
- Suporte para Aplicações de Grande Escala

Para persistir os dados iremos utilizar o MySQL devido a:

- Desempenho:rápido e eficiente,sendo capaz de lidar com grande quantidade de dados e processamento de consultas de forma eficaz.
- Facilidade de Uso
- Integridade de dados mesmo em situações de falha ou erro
- Compatibilidade e Portabilidade
- Segurança
- Escalabilidade

De forma a encapsular cada componente no seu próprio ambiente utilizaremos o Docker pois este oferece:

- Isolamento: O docker permite isolar as dependências e configurações de cada componente da aplicação, evitando conflitos entre diferentes projetos ou ^{oo}versões do software.
- Escalabilidade: Facilita a implementação e o dimensionamento da aplicação, permitindo rápida replicação de *containers*.
- Implementação Simplificada: Simplifica o processo de implementação e testagem, pois cada *containers* contem tudo o que é necessário para executar cada componente.
- Gestão de Recursos: Oferece ferramentas para gerir eficientemente os recursos utilizados pelos *containers*

Capítulo 7

Diagramas

Diagramas são representações visuais que ajudam a modelar e descrever um sistema, os seus componentes e como eles se relacionam. São uma maneira eficaz de comunicar e entender a arquitetura e os aspetos funcionais de um sistema de forma mais clara e acessível.
Os diagramas elaborados para esta segunda entrega foram:

- **Diagrama de Blocos**: O grupo fez uma divisão dos requisitos funcionais da forma a criar subdivisões. Este diagrama permitiu ao grupo decidir qual a arquitetura a utilizar no projeto.
- **Diagrama de Componentes**: A elaboração de um diagrama de componentes permite uma visualização clara das partes constituintes do sistema, identificando os componentes principais e as suas interações. Esta representação gráfica é essencial para compreender a estrutura modular do sistema e como os diferentes elementos colaboram entre si.
- **Diagrama de Sequência**: Realização de um diagrama de sequência por cada *use case*, para se conseguir compreender melhor as interações entre o ator e os vários componentes do sistema.
- **Diagrama de Classes**: O diagrama de classes é uma representação visual das classes e das suas relações no sistema, destacando as classes, atributos e métodos, bem como as associações entre elas.
- **Diagrama de *Deployment***: O diagrama de *deployment* oferece uma visão da distribuição física dos componentes do sistema, mostrando como eles são implementados. Este tipo de diagrama é crucial para entender a infraestrutura necessária e a disposição dos componentes no ambiente de implementação, proporcionando-nos *insights* valiosos para a configuração do nosso sistema.

7.1 Diagrama de Blocos

Este diagrama representa um sistema ou processo usando caixas ou blocos para representar funções, operações ou etapas. Esses blocos são conectados por linhas que mostram a relação entre eles, ajudando a ilustrar a estrutura geral de um sistema e como suas partes se relacionam.

Para a definição dos blocos foram revistos os requisitos funcionais do documento de requisitos, agrupando-os pelas respetivas funcionalidades. Estes grupos deram origem a quatro blocos principais: Gestão de Provas, Gestão de Contas, Gestão de Sistemas e Gestão de Salas.

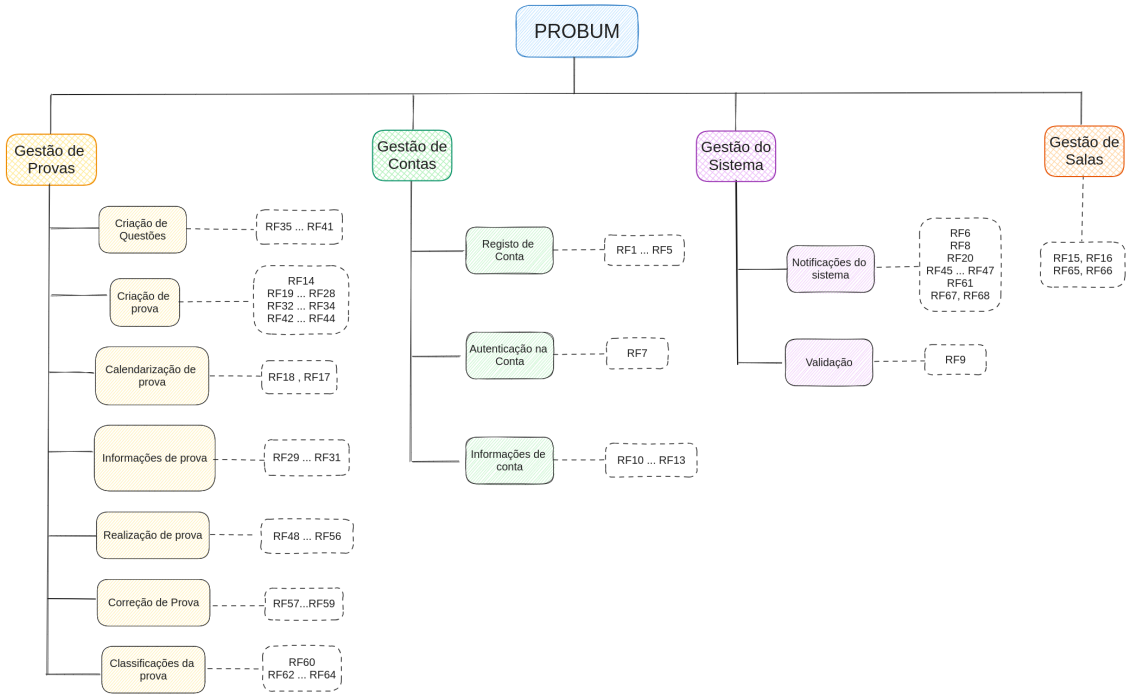

Figura 3: Diagrama de Blocos

7.2 Diagrama de Componentes

Uma vez escolhida a arquitetura do projeto, a equipa de trabalho elaborou um Diagrama de Componentes, dado que este permite identificar os principais componentes do sistema e as suas interações. Aqui é possível observar claramente cada microsserviço, e a forma como cada um comunica com os restantes, estando também cada um deles, obviamente, a comunicar com a *Frontend*, através das suas interfaces. Consideramos relevante dividir o sistema em 4 componentes/microsserviços: Provas, Utilizadores, Salas e Notificações de modo a suportar as diferentes funcionalidades do sistema. Esta decisão foi tomada tendo em conta as funcionalidades definidas pelo Diagrama de Blocos, foram tidas em consideração as potenciais transações entre estes componentes, de forma a maximizar a coesão de funcionalidades.

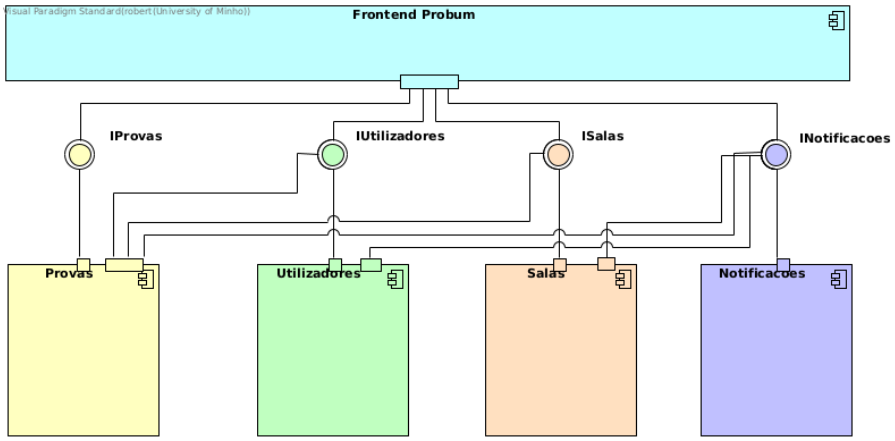

Figura 4: Diagrama de componentes

7.3 Diagramas de Sequência

Este diagrama permite mostrar a interação entre diferentes partes de um sistema, especialmente a ordem e o fluxo de mensagens, eventos ou ações entre objetos ao longo do tempo. Ele descreve como objetos em um sistema interagem sequencialmente em um cenário específico, ajudando a entender a lógica por trás das interações.

7.3.1 Registar Aluno

Neste diagrama é apresentada a interação entre o utilizador (o Técnico), o *Frontend* e cada microsserviço de forma a registar alunos. O técnico solicita ao *Frontend* o formulário de registo de alunos, podendo escolher entre registo individual, ou então registo de vários ao mesmo tempo.

Caso o técnico selecione o registo individual, é lhe enviado um formulário de registo, em que tem de preencher com os dados do aluno que quer registar. Caso os dados sejam válidos, é registado um novo aluno, através do microsserviço utilizadores, e também enviado um email ao aluno registado com as informações da sua conta através do microsserviço notificações.

Ao selecionar o registo de vários alunos, o técnico envia um ficheiro com todos os dados de cada aluno, sendo estes validados. Após serem validados é efetuado o registo dos novos alunos através do microsserviço utilizadores, enviando também um email a cada aluno com as informações das suas contas através do microsserviço notificações.

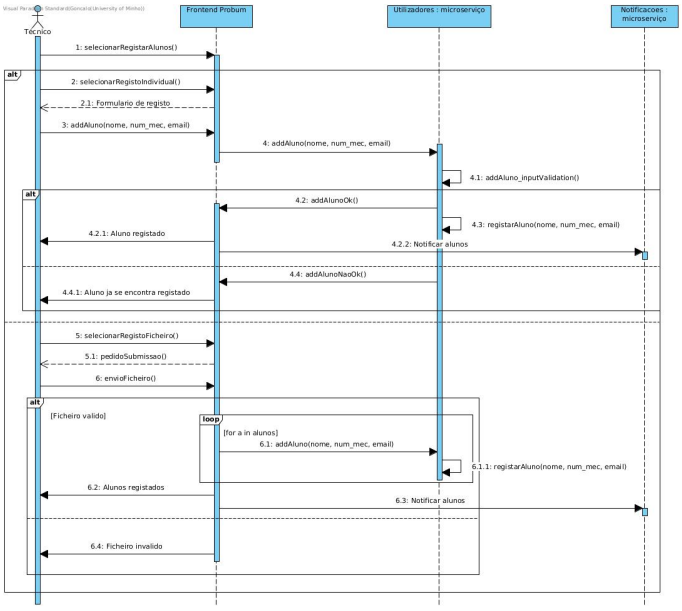

Figura 5: Registar aluno

7.3.2 Registar Docente

Neste diagrama é apresentada a interação entre o utilizador (o Técnico), o *Frontend* e cada microsserviço de forma a registar docentes. O técnico solicita ao *Frontend* o formulário de registo de docentes, podendo escolher entre registo individual, ou então registo de vários docentes ao mesmo tempo.

Caso o técnico selecione o registo individual, é lhe enviado um formulário de registo, em que tem de preencher com os dados do docente que quer registar. Caso os dados sejam válidos, é registado um novo docente, através do microsserviço utilizadores, e também enviado um email ao docente registado com as informações da sua conta através do microsserviço notificações.

Ao selecionar o registo de vários docentes, o técnico envia um ficheiro com todos os dados de cada docente, sendo estes validados. Após serem validados é efetuado o registo dos novos docentes através do microsserviço utilizadores, enviando também um email a cada docente com as informações das suas contas através do microsserviço notificações.

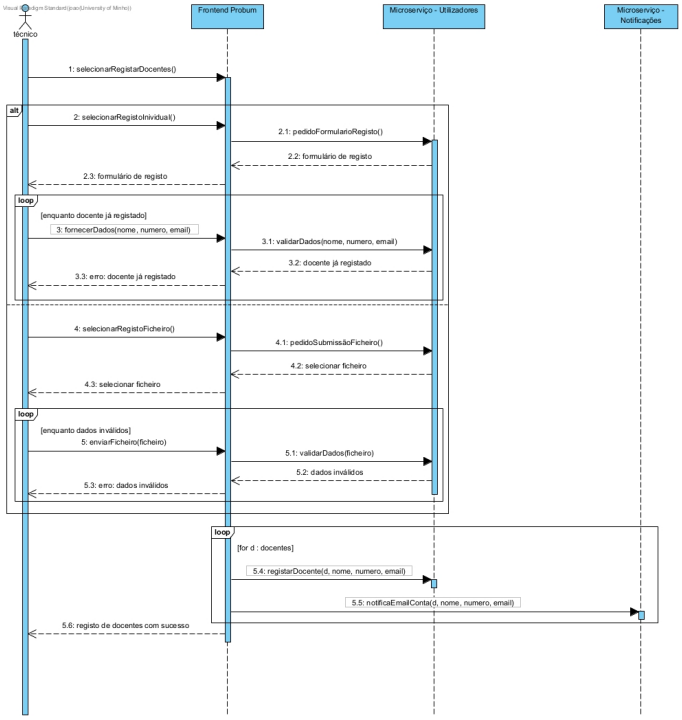

Figura 6: Registar docente

7.3.3 Editar Perfil

No diagrama representado abaixo é possível observar a interação entre o Utilizador, *Frontend* e o microsserviço "Utilizador". Em primeiro lugar, o Utilizador seleciona a opção de aceder ao Perfil e o *Frontend* através do microsserviço Utilizador obtém as informações do perfil e apresenta ao Utilizador.

Depois de ter a sua informação disponível, o Utilizador pode escolher entre alterar o Email ou a Password.

Depois da alteração ser efetuada, o *Frontend* verifica se os dados alterados são válidos. No caso de serem, as alterações são guardadas no perfil do Utilizador, caso contrário informa o mesmo que as novas informações não podem ser aceites (email já existente/password insegura), redirecionando-o à página onde pode editar um dos dois campos de novo.

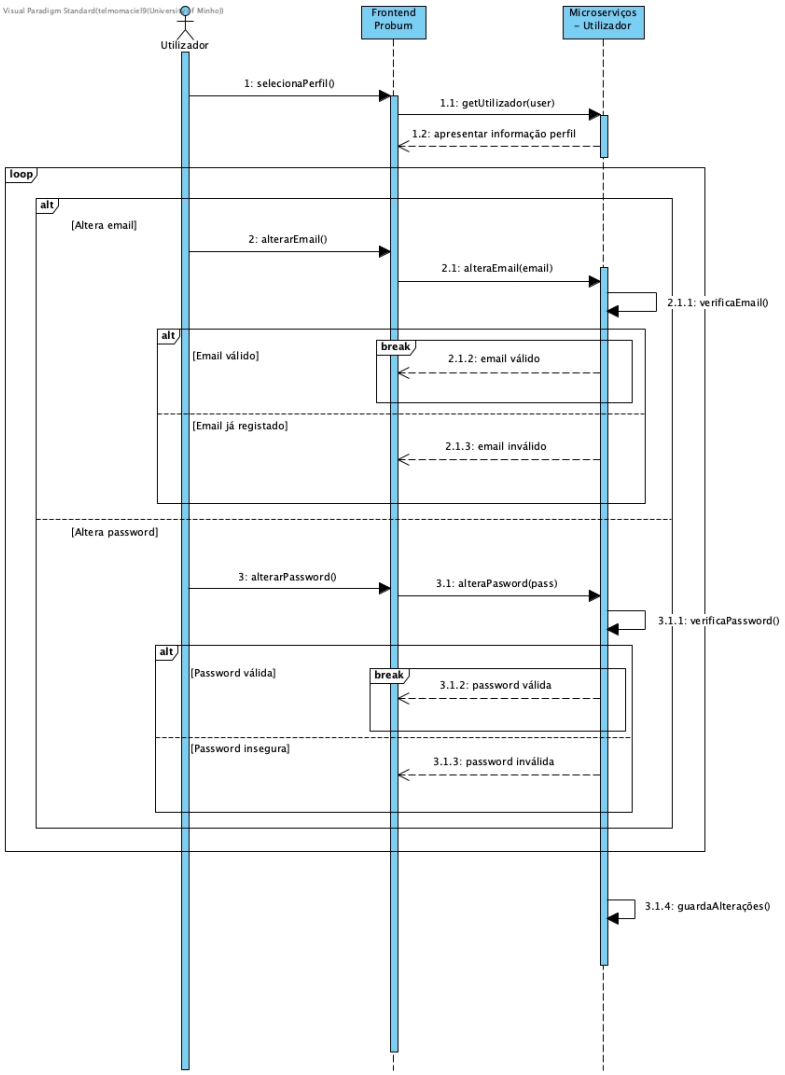

Figura 7: Editar perfil

7.3.4 Autenticação

Neste diagrama é apresentada a interação entre o utilizador (o Técnico, Docente e Aluno), o *Frontend* e cada microsserviço de forma a se autenticar. O utilizador solicita ao *Frontend* o formulário de autenticação, em que esse o retorna, iniciando um ciclo em que o utilizador preenche o formulário e solicita ao *Frontend* a validação dos dados. O *Frontend*, solicita assim ao microsserviço 'Utilizadores' a validação dos dados, recebendo uma resposta indicando assim o sucesso ou o insucesso. Se a autenticação falhar, o *Frontend* informa o utilizador dos dados incorretos e permite-lhe alterar os dados e tentar validar os dados novamente. Este ciclo acaba quando os dados forem validados com sucesso.

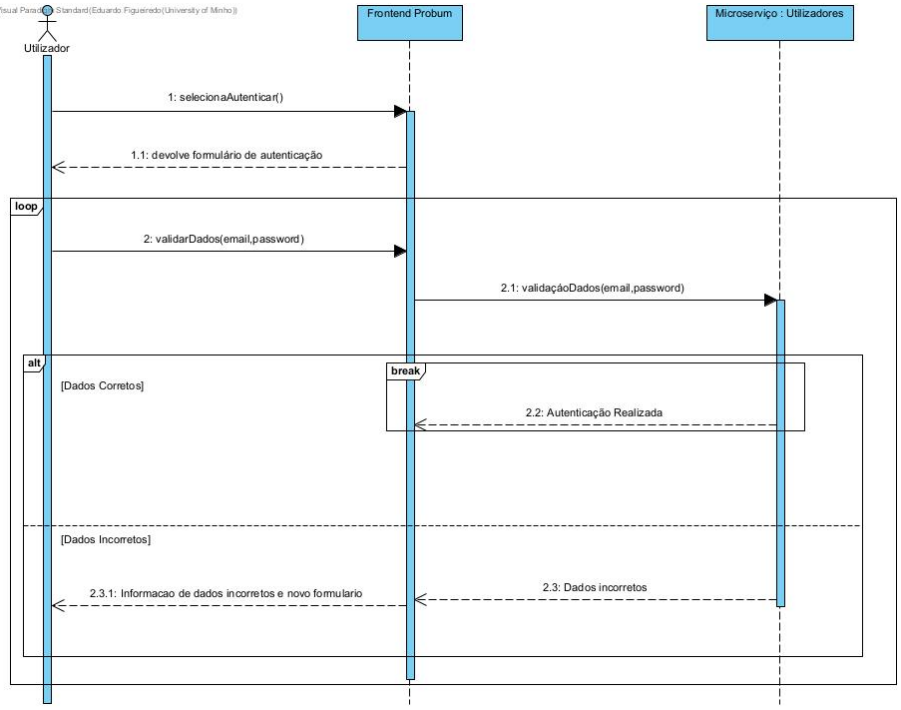

Figura 8: Autenticação

7.3.5 Criar Prova

Neste diagrama apresentamos a interação entre o utilizador (o Docente), o *Frontend* e cada microsserviço de forma a conseguir criar uma nova prova. O utilizador insere o nome da prova e o ficheiro, que é validado pelos microserviços Prova e Utilizadores, pois o ficheiro e/ou os alunos podem ser inválidos. Se ambas as condições não se verificarem, o Docente inicia o processo de calendarização automática do *Probum*. Caso o utilizador rejeite a primeira proposta, o microsserviço associado às salas repete o processo de calendarização automática até que o utilizador aceite uma. Com isto, a prova é criada e as versões são atribuídas, dependendo do número de salas. Com as versões atribuídas e criadas, inicia-se o processo de criar questões para as provas. Com todas as versões com questões, o utilizador procede ao término da criação da prova, o que leva a que o microsserviço das notificações envie um e-mail aos alunos inscritos na prova.

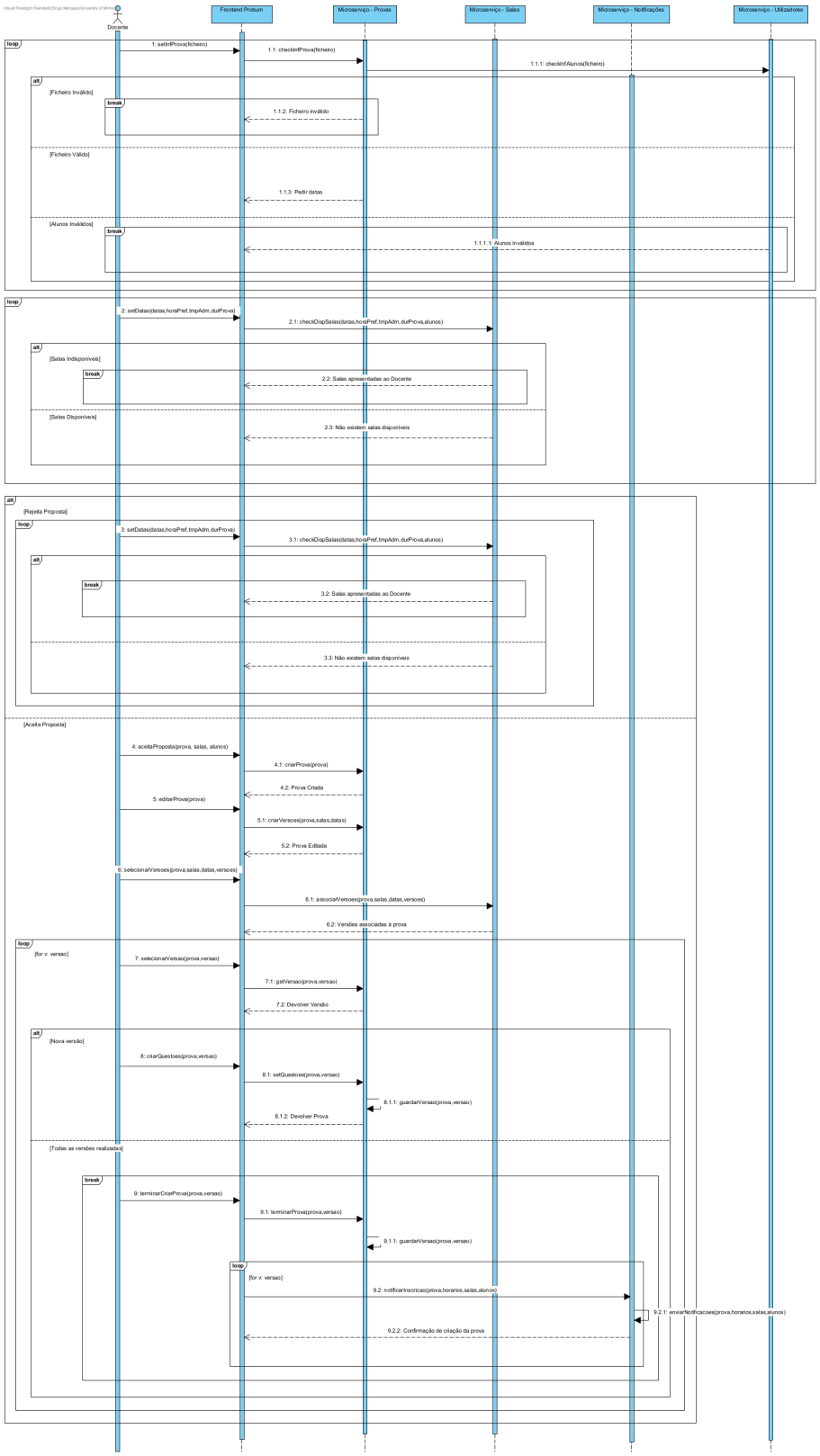

Figura 9: Criar prova

7.3.6 Editar Prova

Neste diagrama é apresentada a interação do Docente com o *Frontend* e os microsserviços Provas e Salas, de forma a editar uma prova. O Docente seleciona a secção de Editar prova e o *Frontend* através do microsserviço Prova obtém a lista de provas e apresenta-a ao Docente. Este seleciona a prova que pretende editar e o *Frontend* através do microsserviço Provas obtém detalhes sobre a prova (nome,duração,datas,sala e horários) e disponibiliza ao Docente.

Em primeiro lugar, caso o Docente queira alterar os detalhes básicos da prova, estes terão de ser validados pelo microsserviço das Salas, alertando caso a data/hora selecionada não estejam disponíveis. Posteriormente, o docente pode editar a versão da prova, e até mesmo criar/editar/-remover questões, necessitando todas estas alterações de entrar em contacto com o microsserviço das Provas.

No final, o Docente termina a edição e o *Frontend* através do microsserviço das Provas guarda as alterações efetuadas.

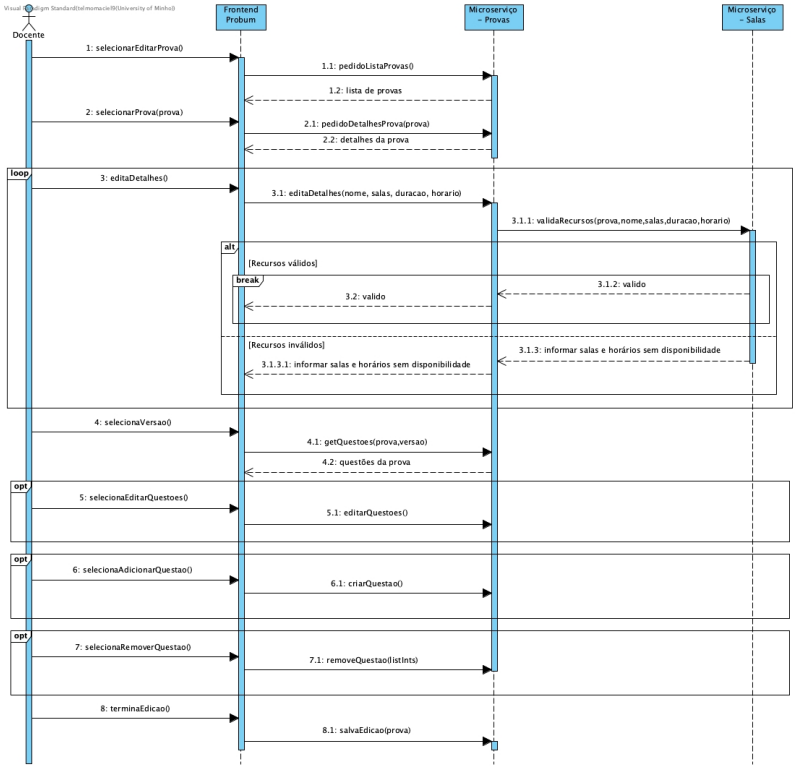

Figura 10: Editar prova

7.3.7 Realizar Prova

Neste diagrama é apresentada a interação do Aluno com o Sistema aquando da realização de uma prova. Inicialmente o Aluno seleciona consultar a listagem de provas, para responder a este pedido o *Frontend* necessita de requisitar esta ao microsserviço provas. Desta listagem o aluno selecionará a prova que pretende realizar, consultando os seus detalhes antes de começar a responder à mesma. No entanto se a prova não estiver disponível ou o seu tempo de admissão tiver esgotado é mostrada uma mensagem de erro e cancelada a operação. Se a prova puder ser realizada é apresentada a primeira questão da mesma. O utilizador vai então iterativamente respondendo a questões e o sistema registando as respostas a cada passo. Em certas provas o aluno não poderá retroceder nas perguntas, se este for o caso o aluno recebe uma mensagem de erro com essa indicação. Se o aluno escolher finalizar a prova é apresentado um menu de confirmação, se for confirmado ou o tempo de admissão da prova esgotar o sistema guarda a submissão da prova e a prova dá-se como terminada.

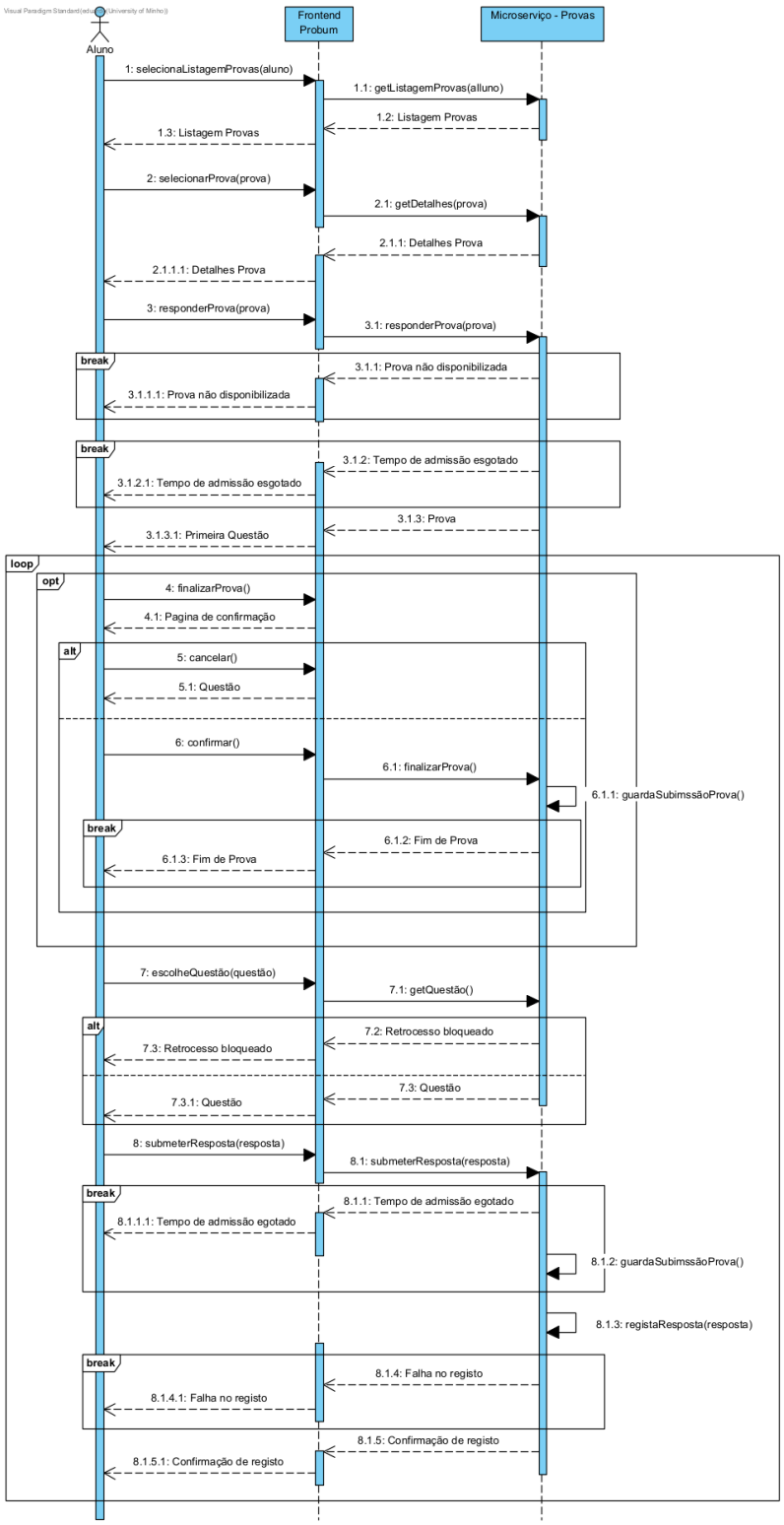

Figura 11: Realizar prova

7.3.8 Consultar Detalhes da Prova

Neste diagrama é apresentada a interação entre o utilizador (o Aluno), o *Frontend* e o microsserviço de provas. O aluno solicita ao *Frontend* a lista de provas realizadas que as obtém através do microsserviço Prova e apresenta a lista ao Aluno, podendo depois selecionar a prova que pretende consultar e o *Frontend* obtém os detalhes da Prova através do microsserviço Prova e disponibiliza ao Aluno.

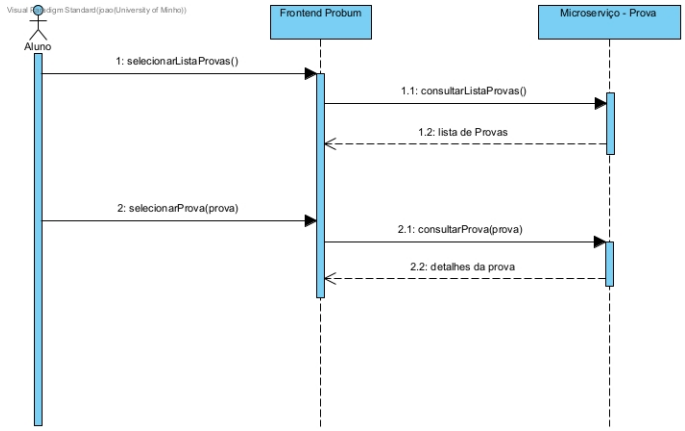

Figura 12: Consultar detalhes da prova

7.3.9 Consultar Prova Corrigida

No diagrama de sequência representado em baixo podemos observar a interação entre o aluno, o *Frontend* da nossa aplicação, e do microsserviço das Provas quando um aluno pretende consultar uma prova corrigida anteriormente realizada pelo mesmo.

O aluno começa por selecionar a opção de visualizar todas as classificações relativas aos testes que o mesmo realizou. Após o pedido ter sido realizado para a *Frontend*, a mesma pede ao microsserviço das Provas pelas provas realizadas pelo respetivo aluno. Após as mesmas serem "entregues"ao aluno e visualizadas, o aluno seleciona a que deseja consultar. Depois do microsserviço das provas responder à *Frontend* com a prova em questão o aluno consulta os aspetos gerais da nota (data de realização, nota). Posteriormente o aluno seleciona a opção de consulta dos conteúdos da prova, como as respostas e as respetivas questões, sendo assim efetuado um pedido ao microsserviço das provas, sendo que este devolve toda a informação desejada pelo utilizador.

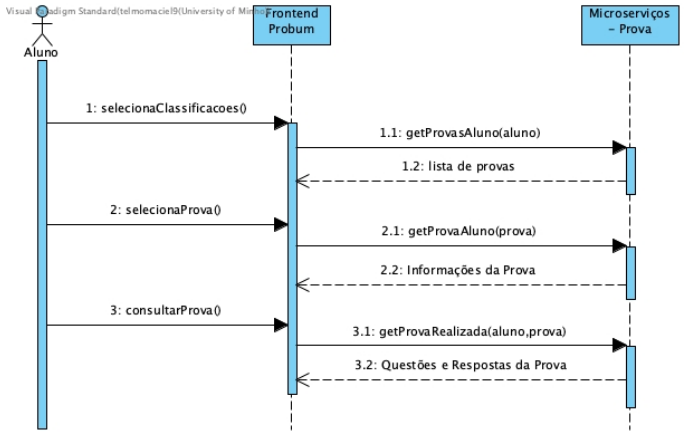

Figura 13: Consultar prova corrigida

7.3.10 Partilhar Prova

Neste diagrama é especificada a interação do Docente com o sistema quando este pretende partilhar uma prova. Inicialmente quando a opção partilhar prova é selecionada é realizado um pedido ao microsserviço provas de forma a obter uma listagem das provas que o docente pode partilhar, de seguida o docente seleciona a prova e os docentes com os quais quer partilhar a mesma. De forma a verificar se os docentes selecionados são validos é efetuado um pedido ao microsserviço provas que efetua tal verificação e responde com erro caso algum docente não seja válido. Se todos os docentes fornecidos forem validos o sistema notifica iterativamente todos os docente, tanto por email como pela própria plataforma.

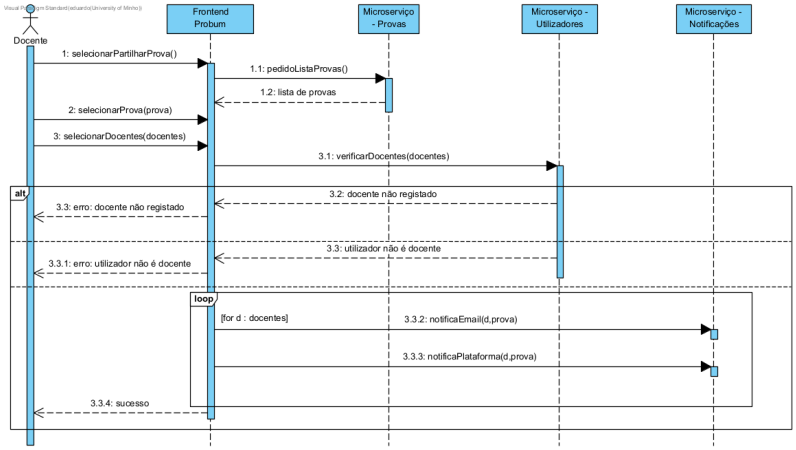

Figura 14: Partilhar prova

7.3.11 Classificar Respostas

Neste diagrama é apresentada a interação entre o utilizador (o Docente), o *Frontend* e cada microsserviço de forma a classificar respostas. Inicialmente o Docente, por meio do *Frontend*, solicita ao microsserviço 'Provas' a lista de provas disponíveis para correção. Depois da escolha da prova a ser classificada, o *Frontend* solicita ao microsserviço 'Provas' a lista de alunos daquela prova. O docente pode optar pela correção automática, pedindo ao microsserviço 'Provas' para a executar. Inicia-se um ciclo, em que o Docente pode escolher individualmente a prova de um aluno a classificar,onde após classificar a prova manualmente envia para o *Frontend*, onde armazena no microsserviço 'Provas'.Esse ciclo termina quando o docente não desejar continuar a classificar provas.

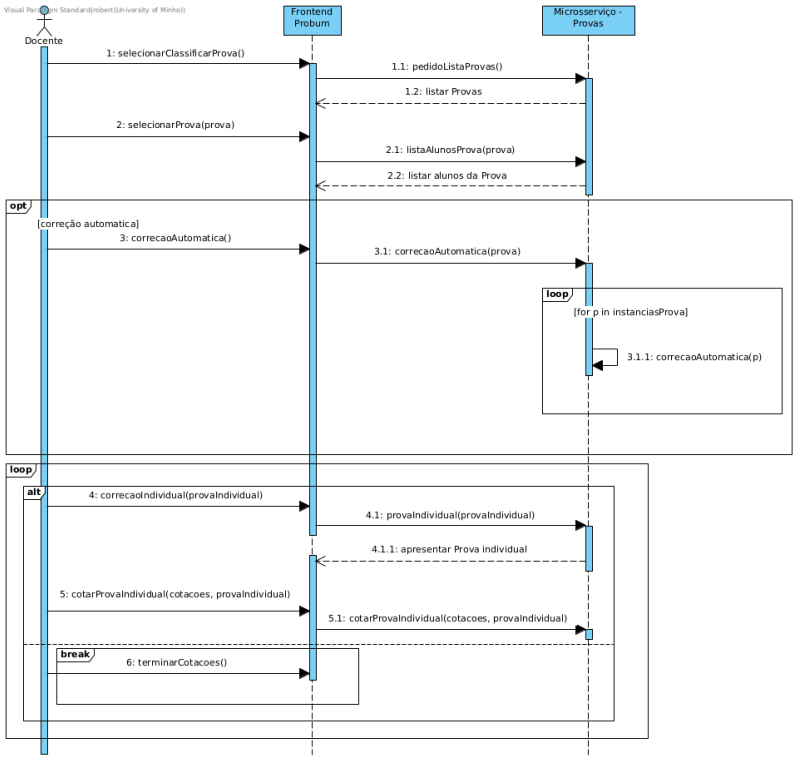

Figura 15: Classificar respostas da prova

7.3.12 Publicar Classificações da Prova

O diagrama seguinte apresenta a interação que existe entre o utilizador (o Docente), a *Frontend* e os microsserviços "Prova"e "Notificações", de forma a publicar as classificações de um prova. A fim de publicar as classificações de uma determinada prova, o docente deve aceder à lista de provas e selecionar a prova pretendida. Depois, o docente seleciona a opção de tornar as classificações públicas. Esta ação pode não ser possível se o sistema verificar que nem todos os alunos têm as suas provas classificadas. Neste caso, o sistema emite um aviso, dizendo que para publicar as classificações da prova é preciso classificar por completo as provas de todos os alunos. Uma vez públicas as classificações, o sistema notifica os alunos tanto pela plataforma como por email

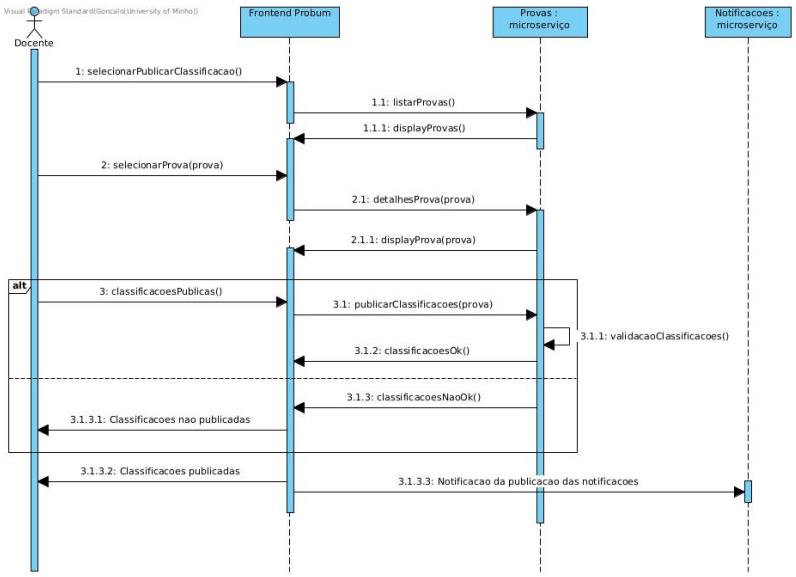

Figura 16: Publicar classificações da prova

7.3.13 Criar Questões

Neste diagrama é apresentada a interação entre o utilizador (o Docente), a *Frontend* e cada microsserviço de forma a criar questões. Em primeiro lugar, o docente acede à lista das provas e seleciona a prova à qual pertence a questão que pretende criar. É então apresentada a lista de questões e o docente seleciona a opção de criar questão. Uma vez selecionada esta opção, o docente insere a descrição/enunciado podendo de seguida inserir uma ou mais imagens, sendo esta ação opcional. De seguida, o docente seleciona o tipo de questão que quer criar, dentro dos 4 tipos possíveis: escolha múltipla, verdadeiro e falso, desenvolvimento e completar espaços. Uma vez selecionado o tipo, o docente define a cotação ou o critério de avaliação, dependendo do tipo escolhido, ficando assim criada a questão.

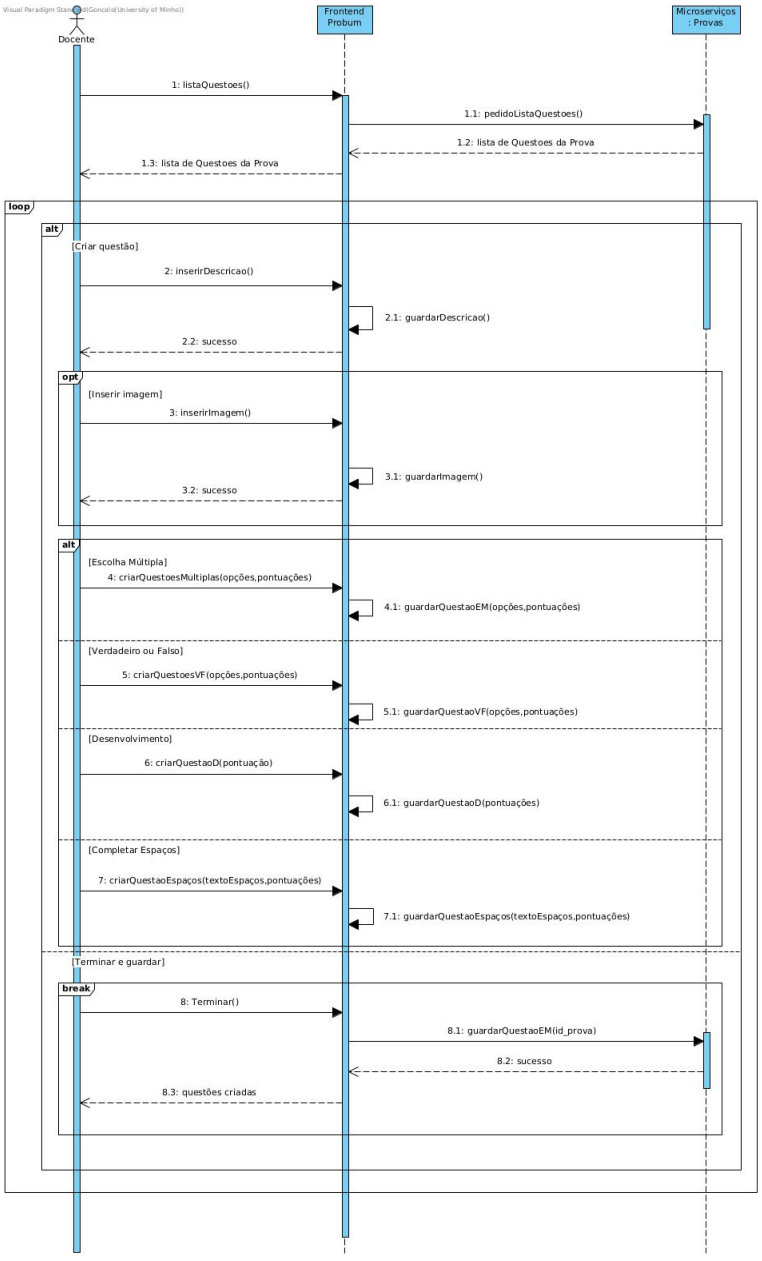

Figura 17: Criar questões

7.3.14 Editar Questões

Neste diagrama é apresentada a interação entre o utilizador (o Docente), a *Frontend* e cada microsserviço de forma a editar questões.O docente solicita ao *Frontend* a lista de questões para uma determinada prova. O *Frontend*, por meio do microsserviço 'Provas', exibe uma lista de questões ao docente e inicia um ciclo de edição. Depois o docente escolhe uma questão, pedindo assim os detalhes ao *Frontend*, obtidos pelo microsserviço 'Provas'.Durante a edição, altera a descrição e a imagem (opcionalmente), e as respostas. No final o docente termina e o *Frontend* guarda as alterações no microsserviço 'Provas', onde o sistema regressa ao ciclo de escolher novamente uma questão, onde termina quando o docente estiver satisfeito com as questões da prova.

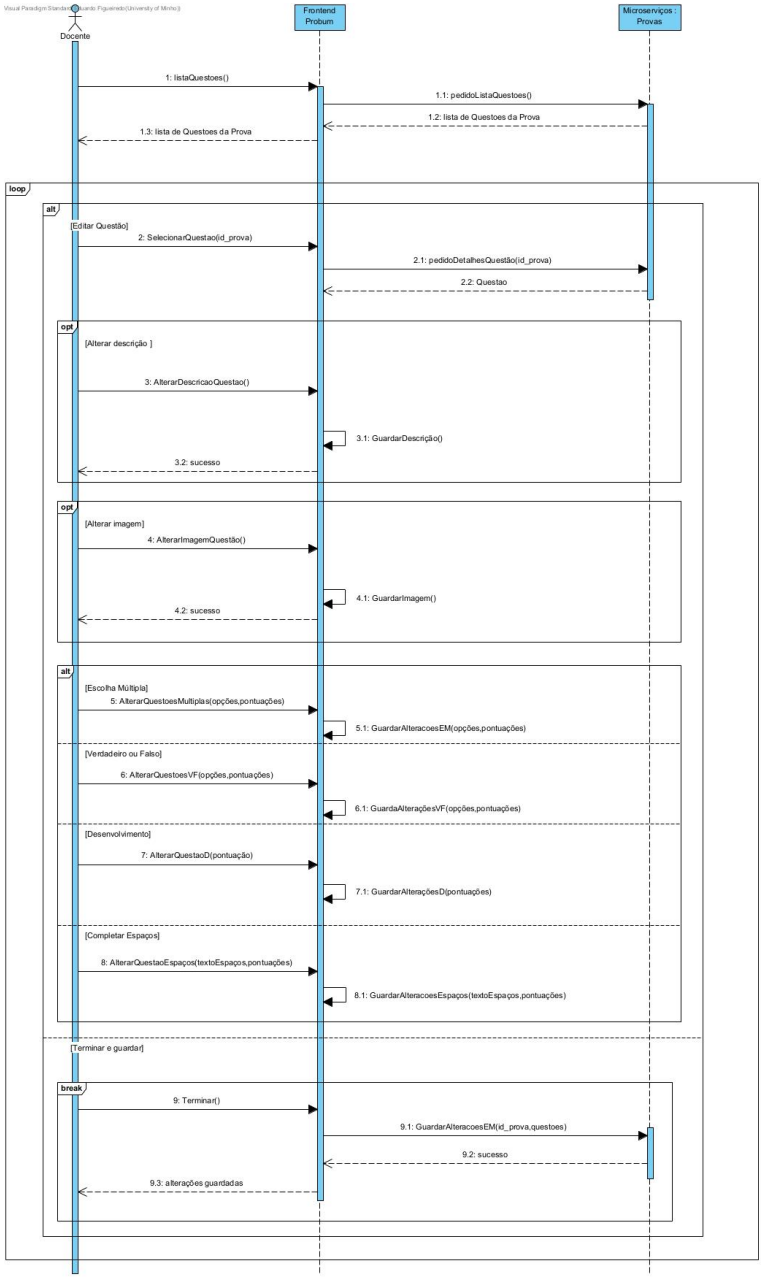

Figura 18: Editar questões

7.3.15 Aceder a Notificações

Neste diagrama é apresentada a interação entre o utilizador (o Docente ou Aluno), a *Frontend* e o microsserviço utilizadores de forma a aceder às notificações. O *Frontend* informa o utilizador da existência de notificações por ler e este seleciona a secção das notificações.O *Frontend*,por meio do microsserviço Utilizadores,vai buscar a lista de notificações e exibe a lista.
Em alternativa o utilizador pode aceder às notificações mesmo não tendo notificações por ler selecionando para o efeito a secção correspondente e o *Frontend* por meio do microsserviço utilizadores vai buscar a lista de notificações e por fim exibe-a.

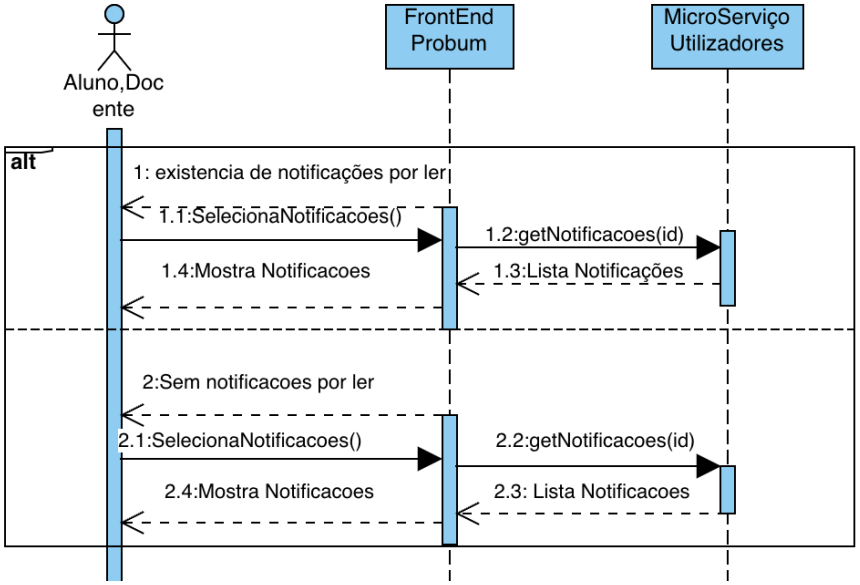

Figura 19: Aceder a notificações

7.3.16 Gerir Salas

Neste diagrama é apresentada a interação entre o utilizador(Técnico),a *Frontend* e os microsserviços salas,provas e notificações. O técnico seleciona a funcionalidade de adicionar salas e o *Frontend* solicita a submissão de um ficheiro com informações relativas à sala que pretende adicionar,O Técnico submete o ficheiro e o *Frontend* verifica se o ficheiro é válido.
No caso dos dados inseridos não serem corretos o *Frontend* avisa o Técnico,quando os dados forem corretamente inseridos o *Frontend* através do microsserviço salas adiciona a sala ao sistema.
Em Alternativa o Técnico pode remover salas selecionando a funcionalidade correspondente e o *Frontend* através do microsserviço salas obtém a lista de salas disponíveis e mostra ao técnico. O Técnico seleciona as que pretende remover e o *Frontend* através do microsserviço Provas verifica se as salas selecionadas estão alocadas a alguma prova e notifica os docentes da indisponibilidade da sala através do microsserviço Notificações e por fim remove as salas.

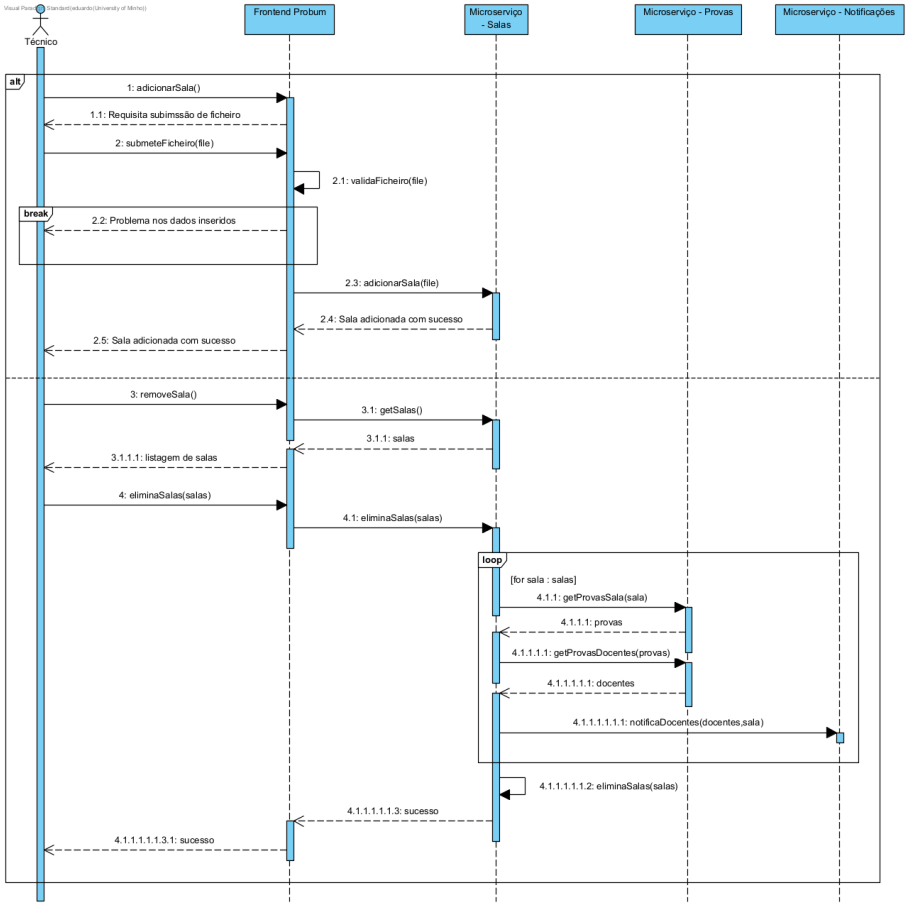

Figura 20: Gerir salas

7.4 Diagrama de classes

O diagrama de classes é uma representação estática da estrutura e relacionamentos entre classes num sistema orientado a objetos. Ele exibe classes, os seus atributos, métodos e as associações entre as classes, ajudando a visualizar a estrutura de um sistema e as suas interações.

No sistema *Probum*, a utilização consistente de Interfaces e *Facades* emerge como uma prática arquitetural fundamental. As Interfaces, representadas por contratos definidos, estabelecem um conjunto de operações padronizadas, promovendo a modularidade e facilitando a substituição de implementações subjacentes. Essas interfaces definem a API do sistema, proporcionando uma clara separação entre a especificação funcional e sua realização concreta e serão fulcrais para as transações necessárias entre os microsserviços.

Por outro lado, as *Facades* desempenham o papel crucial de simplificar a interação com os diferentes subsistemas. Ao encapsular a lógica de acesso e manipulação dos componentes, as *Facades* proporcionam uma camada de abstração, facilitando o uso consistente desses subsistemas em todo o sistema *Probum*. A relação de realização entre as *Facades* e as Interfaces reflete a adesão à especificação definida pelas interfaces, garantindo uma consistência na interação com diferentes partes do sistema.

7.4.1 Utilizadores

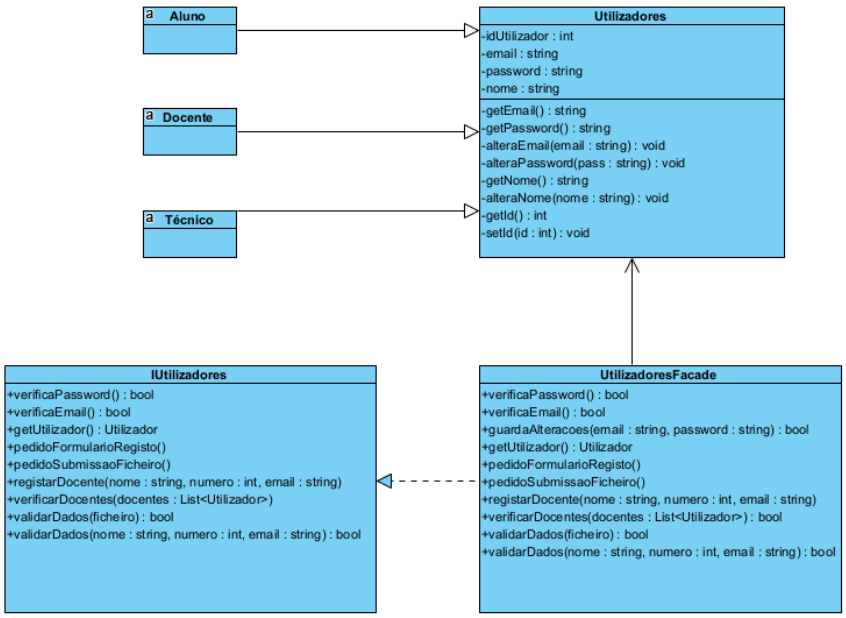

Figura 21: Utilizadores

7.4.2 Salas

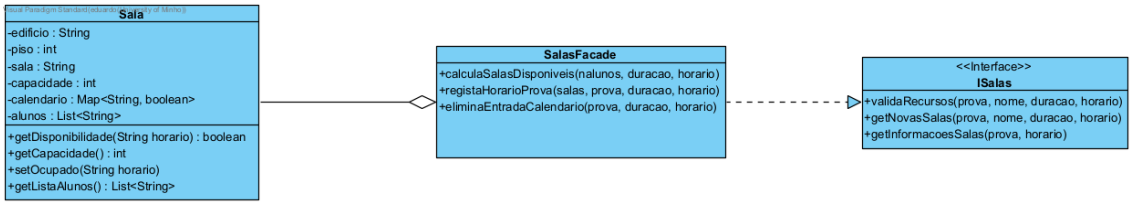

Figura 22: Salas

7.4.3 Provas

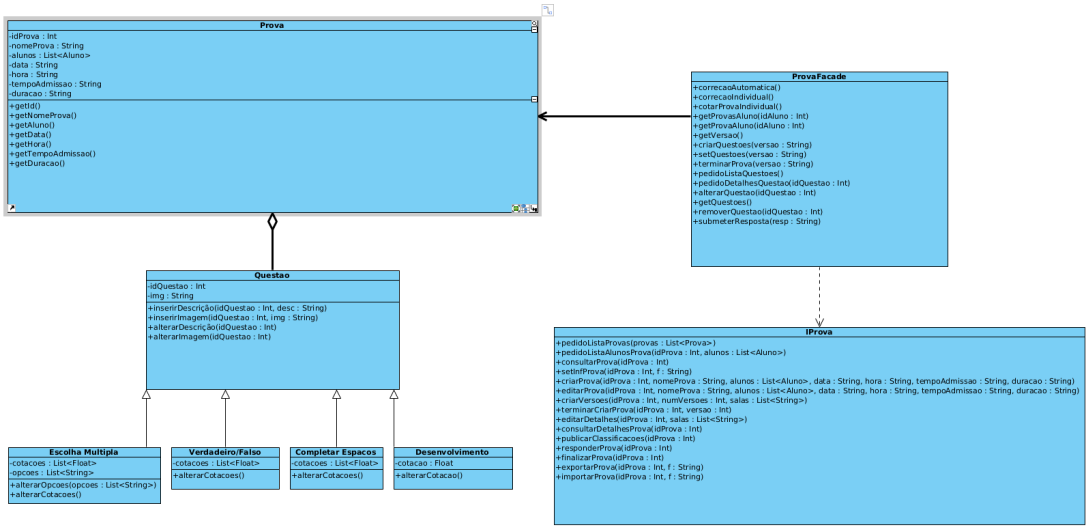

Figura 23: Provas

7.4.4 Notificações

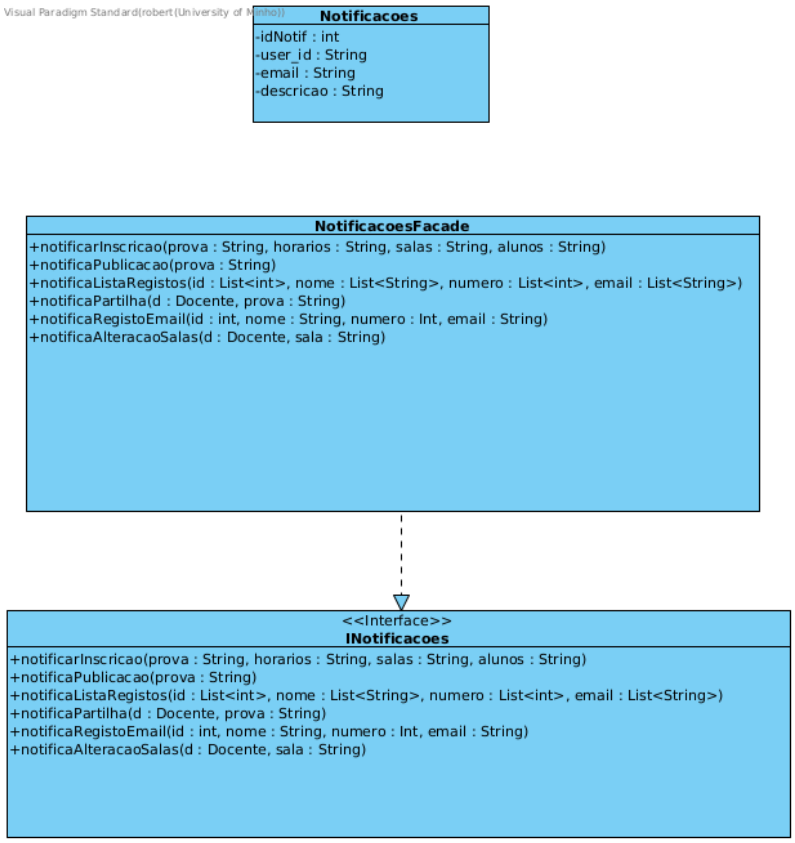

Figura 24: Notificações

7.5 Diagrama de *Deployment*

Este diagrama é usado para visualizar a distribuição de artefactos de software em hardware físico. Ele mostra como os diferentes componentes, artefactos de software e nós de processamento são implantados em servidores, máquinas ou dispositivos físicos, mostrando a configuração e a disposição física do sistema.

O nó cliente, representando a interface do utilizador (*Browser*), comunica-se com um *Frontend* que está encapsulado num *Docket container* dentro do nó PC. Esta abordagem facilita a portabilidade e a implantação consistente do *Frontend*.

Dentro do nó PC, há quatro *VMs* (máquinas virtuais) dedicadas a cada microsserviço: Provas, Utilizadores, Salas e Notificações. Cada VM contém dois *Docker containers*, um para o serviço em si e outro para o motor da base de dados, exceto para o microsserviço Notificações, que possui apenas um *container* para o serviço.

Esta arquitetura modularizada promove a escalabilidade e a manutenção independentes de cada microsserviço. A utilização de *Docker containers* facilita a gestão de dependências e ambientes de execução consistentes. A presença de máquinas virtuais para cada microsserviço sugere um isolamento mais robusto entre eles, proporcionando benefícios como segurança, independência de recursos e mitigação de falhas, garantindo que problemas num microsserviço não afetem diretamente outros componentes do sistema.

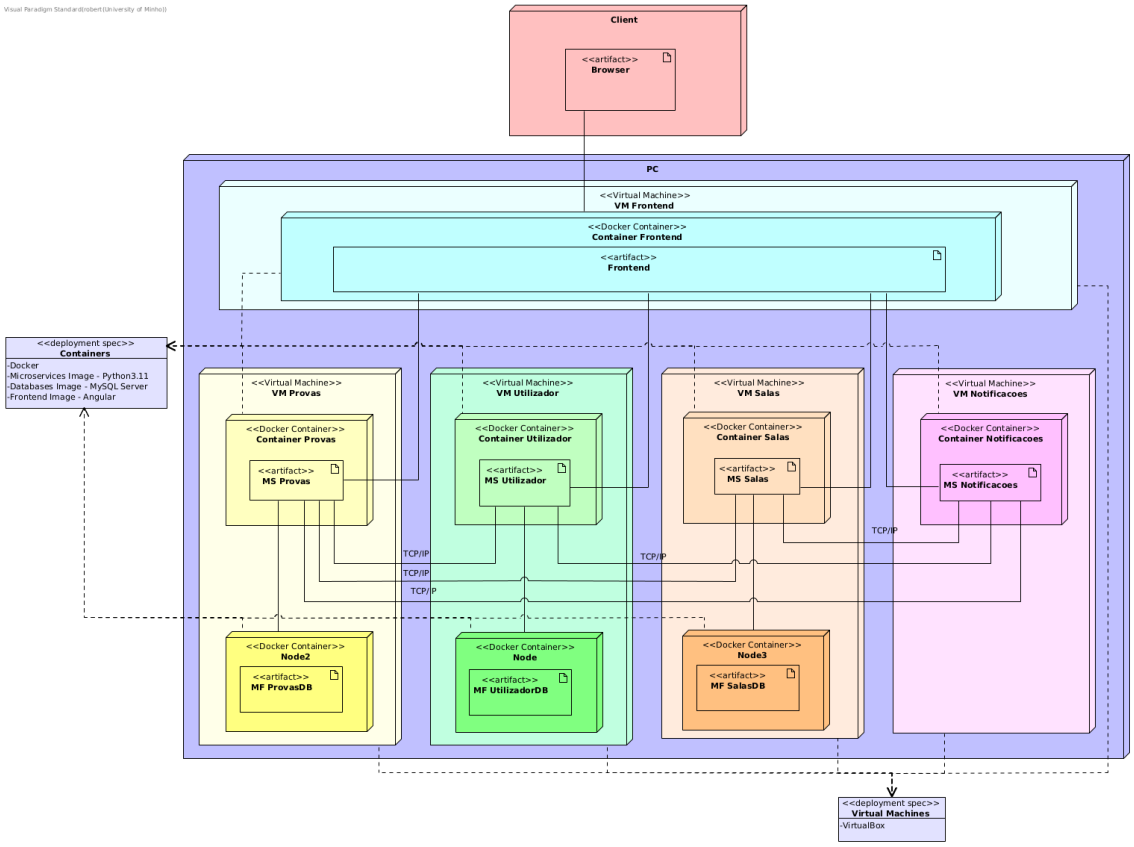

Figura 25: Diagrama de Deployment

Capítulo 8

Conclusão

Durante a realização desta fase do projeto conseguimos planear e implementar a arquitetura da aplicação a ser desenvolvida, assim como definir sua estrutura e metas. Além disso,foi crucial estabelecer estratégias para garantir uma experiência segura e ágil para o utilizador,visando melhorar consideravelmente a qualidade da aplicação.

Dada por concluída a segunda fase do projeto consideramos importante realizar uma análise crítica, e ainda, realçar os pontos positivos e negativos.

No lado positivo, é importante ressalvar que este documento de arquitetura está alinhado com o *template* fornecido. Através deste documento, é possível obter uma visão abrangente e precisa dos aspetos arquiteturais do sistema, utilizando os diagramas mais apropriados. Isso é feito seguindo uma progressão de abstração, começando com uma visão mais ampla e indo gradualmente para detalhes mais específicos, proporcionando uma compreensão completa do sistema em diferentes níveis de detalhe.

Por outro lado,em termos de aspetos negativos,consideramos que caso este projeto fosse desenvolvido para o mundo real seria necessário adicionar novas funcionalidades para uma melhor experiência do utilizador.

Para concluir, consideramos que houve um balanço positivo do trabalho realizado dado que apesar das dificuldades sentidas estas foram superadas e foram cumpridos os requisitos propostos. Deste modo, acreditamos que através deste documento o projeto tem uma base sólida para a próxima fase e última fase de desenvolvimento e implementação.