

PROBUM

Requisitos e Arquiteturas de Software-2^a Fase

Grupo 8-B

2023/2024

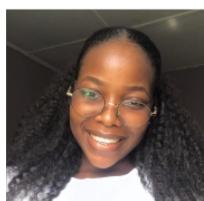
26 de janeiro de 2024

Elementos do Grupo e Contribuição

Número de Aluno	Nome
PG53951	João Pedro Vilas Boas Braga
PG51636	Nsimba Teresa Armando
PG53791	Duarte Gonçalves Parente
PG50006	Juliana Ngoma Cesar Silvério
PG52672	André Lucena Ribas Ferreira
A94942	Miguel Velho Raposo
A91775	José Pedro Batista Fonte
PG52675	Carlos Eduardo da Silva Machado
PG52693	Maria Francisca Mendes Lemos
A96075	João Bernardo Teixeira Escudeiro
PG54162	Rafael Picão Ferreira Correia
PG54093	Miguel Ângelo Silva Senra



PG53951



PG51636



PG53791



PG50006



PG52672



A94942



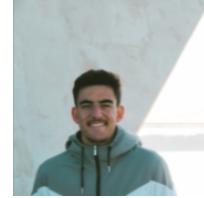
A91775



PG52675



PG52693



A96075



PG54162



PG54093

Aluno	Cont.	Aluno	Cont.	Aluno	Cont.
PG53951	+0.2	PG52672	+0.2	PG52693	+0.2
PG51636	-1	A94942	-1	A96075	+0.2
PG53791	+0.2	A91775	+0.2	PG54162	+0.2
PG50006	+0.2	PG52675	+0.2	PG54093	+0.2

Tabela 1: Contribuição por Aluno

Índice

Elementos do Grupo e Contribuição	1
Introdução e Objetivos	4
Visão geral dos requisitos	4
Stakeholders	5
Restrições Arquiteturais	6
Requisitos de Qualidade	9
Ârvore de Qualidade	10
Extensão e Contexto do Sistema	11
Estratégia da Solução	12
1. Autenticação	12
2. Perfil	12
3. Gestão de Provas	12
4. Agendamento de Provas	13
5. Realizar Prova	13
6. Correção	13
7. Consulta	13
Visualização dos Blocos de Construção	14
Diagrama de Componentes	15
Microserviços	17
Microsserviço Gestão de Utilizadores	17
Microsserviço Gestão de Provas(Criação)	17
Microsserviço Gestão Logística	18
Microsserviço Gestão de Provas(Realização)	19
Microservice Gestão de Provas(Correção/Consulta)	20
Visualização em Tempo de Execução	21
Registrar Docentes - <i>Use Case 1</i>	21
Registrar Alunos - <i>Use Case 2</i>	23
Autenticar - <i>Use Case 3</i>	25
Editar Perfil - <i>Use Case 4</i>	27
Criar Prova - <i>Use Case 5</i>	28
Criar Questões - <i>Use Case 6</i>	30
Editar Prova - <i>Use Case 7</i>	32
Editar Questões - <i>Use Case 8</i>	34
Consultar Detalhes da Prova - <i>Use Case 9</i>	37
Partilhar Prova - <i>Use Case 10</i>	38
Responder Prova - <i>Use Case 11</i>	41
Classificar Respostas - <i>Use Case 12</i>	45
Publicar Classificações da Prova - <i>Use Case 13</i>	47
Consultar Prova Corrigida - <i>Use Case 14</i>	48
Gerir Salas - <i>Use Case 15</i>	50
Aceder às Notificações - <i>Use Case 16</i>	51
Visualização do Deployment	53
Modelos Lógicos de Bases de Dados	55
Documentação API	57

Introdução e Objetivos

Este documento representa o resultado da segunda fase do projeto da Unidade Curricular de Requisitos e Arquitetura de Software. O desafio apresentado foi a conceção da arquitetura da solução do sistema ProbUM. Durante este processo, seguimos os princípios fundamentais discutidos nas aulas teóricas, aplicando metodologias de análise de requisitos.

A proposta visa aprimorar a experiência tanto dos alunos quanto dos docentes, simplificando os processos de realização, criação e correção de avaliações. Além disso, o ProbUM tem o intuito de otimizar os recursos da universidade, promovendo a transição do formato em papel para o ambiente digital. Este capítulo fornecerá uma visão geral das metas e objetivos da arquitetura proposta, destacando a sua relevância no contexto do aprimoramento da experiência educacional e na eficiência operacional da instituição.

Visão geral dos requisitos

O objetivo principal do ProbUM é facilitar aos alunos a realização das suas provas académicas de forma adequada, garantindo a integridade do processo e fornecendo ferramentas necessárias para a avaliação eficaz dos conhecimentos adquiridos. Tendo em vista a conceção de uma solução para o problema que temos em mãos, começámos por identificar uma visão geral dos requisitos essenciais para o desenvolvimento e funcionamento eficaz do sistema ProbUM.

- **Autenticação e Autorização:** Os alunos devem poder autenticar-se de forma segura no sistema usando credenciais fornecidas pela universidade. Os professores devem ter acesso a um painel de controle para autorizar e monitorar as provas em andamento. Esta é uma característica muito importante pois envolve muitas questões de segurança que é um aspecto crítico no sistema.
- **Criação de Provas:** Os professores devem poder criar conjuntos de questões e gerar provas aleatórias a partir desses. A plataforma deve suportar diferentes tipos de perguntas (múltipla escolha, dissertativa, etc.).
- **Realização de Provas:** Alunos devem poder acessar a prova no horário designado. O sistema deve garantir a integridade da prova e a identificação do aluno, prevenindo batotas e plágio.
- **Submissão de Respostas:** Alunos devem poder enviar suas respostas de forma segura. O sistema deve registar a hora de início e término da prova para fins de avaliação.
- **Avaliação Automática:** O sistema deve permitir a avaliação automática de respostas múltipla escolha. Professores devem poder revisar e avaliar manualmente respostas dissertativas.

A criação do ProbUM foi impulsionada por uma série de necessidades identificadas no contexto da realização de provas académicas para alunos de uma unidade curricular em um curso universitário. A motivação para desenvolver este sistema baseia-se principalmente nas seguintes considerações:

- Redução de Carga Administrativa
- Tecnologia e Educação
- Relatórios Detalhados
- Redução de Consumo de Papel

Estando numa fase muito precoce da conceção necessitámos de identificar os aspectos da especificação que são mais importantes e que permitem, desde logo, o funcionamento da plataforma. O sistema deverá ser construído, portanto, baseando-se nos requisitos funcionais e não funcionais mais importantes pelo agrupámos e sintetizámos aqueles que são a base de funcionamento do

ProbUM. Para uma melhor percepção das funcionalidades, apresentamos em seguida os use cases que destacam mais o nosso sistema.

Use Cases	Descrição
Autenticação e Autorização	Alunos, professores e administradores devem autenticar-se no sistema para acessar funcionalidades específicas.
Registrar Docentes e Alunos	Consiste no registo individual de docentes e alunos com os seus respetivos níveis de permissões.
Gerir Questões	Os Docentes têm a capacidade de criar, editar e eliminar questões, incluindo os tipos de questões e o seu critério de avaliação.
Gerir Provas	Os Docentes têm a capacidade de criar, editar e eliminar provas, incluindo a definição de parâmetros como tipo de questão, tempo e pontuação.
Gerir Salas	Consiste no registo de salas ao Sistema, para facilitar a calendarização de provas.
Realizar Provas	Alunos podem acessar as provas no horário e sala designados, respondendo às questões de acordo com as instruções fornecidas.
Classificar Respostas	A classificação é atribuição de uma pontuação para cada resposta, de forma automática (feita pelo sistema) ou manual (feita pelos Docentes).
Consultar Provas	Consiste na verificação de detalhes relativos a uma Prova já realizada.

Stakeholders

Esta secção fornece uma visão detalhada dos *stakeholders* envolvidos no sistema PROBUM, delineando os seus papéis e respetivas expectativas em relação à arquitetura do mesmo. Esta análise revela-se fundamental para uma compreensão clara da interação das partes interessadas e envolvidas no projeto proposto, contribuindo para uma melhor eficácia na gestão de expectativas e tomadas de decisão ao longo do desenvolvimento do sistema.

Papel	Expectativas/Responsabilidades
Cliente	<ul style="list-style-type: none"> - Requer um sistema que automatize os processos de criação, calendarização, realização e correção de provas de avaliação. - Procura melhorar a experiência dos corpos envolvidos no seu departamento pedagógico de IESs. - Participa ativamente no planeamento e validação da solução.
Consumidor	<ul style="list-style-type: none"> - Responsável por contactar com os alunos e controlar todas as implicações associadas a uma prova de avaliação. - Procura uma ferramenta que facilite a criação, distribuição e correção de provas de avaliação. - Participa ativamente na definição e validação da solução.
Aluno	<ul style="list-style-type: none"> - Responsável pela realização da prova de avaliação a que se apresentar. - Espera um produto simples, eficiente, flexível e robusto.
Técnico	<ul style="list-style-type: none"> - Responsável pela instalação da plataforma e respetiva manutenção. - Pretende um processo de instalação e configuração simplificado do sistema. - Requer métricas e <i>logs</i> para uma antecipação e eventual resolução eficaz de problemas.

Restrições Arquiteturais

Projetos de grande nível estão sempre limitados seja em termos de questões temporais como de questões monetárias. Neste sentido, e considerando que a nossa solução dependerá destas restrições, fizemos um estudo e identificámos-las em seguida.

Requisitos Técnicos

- Limitações de Hardware
Gama baixa - O sistema deve correr em computadores de gama baixa, para garantir a mesma experiência de utilização em diferentes computadores de diversas gamas.
- Compatibilidade de Software
Acesso exclusivo ao Probum durante uma prova - Para a realização de uma prova de avaliação, o computador disponibilizado apenas deve permitir acesso ao Probum, para evitar que o aluno recorra a outras aplicações (navegadores web, WhatsApp, Discord, etc.) durante a realização da sua prova.

Segurança e restrições de privacidade

- Restrições de proteção de dados
Dados protegidos - Os dados de cada utilizador não devem ser partilhados com terceiros sem autorização para garantir a privacidade do utilizador.
- Autenticação e autorização
Acesso não autorizado - O sistema restringe o acesso a funcionalidades de acordo com o utilizador para impedir acessos não autorizados a informação sensível.

Restrições relacionadas com a experiência dos utilizadores

- Restrições de Acessibilidade:
Deficiência visual - O produto está acessível a pessoas com algum tipo de deficiência

visual e estes devem conseguir utilizar o produto com facilidade.

- Usabilidade:

Utilizadores do ensino superior - O produto deve ser fácil de utilizar por pessoas que frequentam o ensino superior, assim estes conseguem usar o produto sem grande dificuldade. Para permitir uma melhor experiência aos utilizadores, as Provas são apresentadas por ordem descendente à data de calendarização, e as Notificações são apresentadas por ordem cronológica.

Restrições Temporais

- Deadlines: O documento terá de ser entregue até dia 1 de dezembro de 2023.
O projeto terá de ser entregue até dia 22 de dezembro de 2023.
- Tempo para teste: o Docente indica a duração de cada Prova no momento em que a cria.

Restrições a nível de recursos

- Financeiros: O orçamento total do projeto é de 20000€. A equipa é constituída por 15 engenheiros de software, e temos de ter em conta os salários destes, a compra de um domínio, bem como um computador para hospedar todos os dados da aplicação.

Restrições de escalabilidade

- O sistema deve ser escalável para permitir o aumento ou diminuição de recursos conforme o número de utilizadores.

Restrições Culturais e Sociais

- **Suporte de Linguagem:** O produto deve suportar vários idiomas, para permitir que os utilizadores possam utilizar a língua que mais dominam.

Aliado às restrições acima mencionadas surgem os requisitos arquiteturais :

Restrições de Escalabilidade

Restrição: O sistema deve lidar com um potencial grande número de utilizadores em simultâneo durante os períodos de pico, como os de exames. Impacto: As decisões arquiteturais devem centrar-se na escalabilidade, possivelmente envolvendo o uso de soluções escaláveis em nuvem, balanceamento de carga e estruturas de banco de dados otimizadas.

Segurança arquitetural

Restrição: A segurança é primordial devido à sensibilidade dos dados de estudantes e docentes. O sistema deve proteger contra acessos não autorizados, violações de dados e outras ameaças de segurança. Impacto: A implementação de mecanismos robustos de autenticação e auditorias de segurança regulares torna-se crucial. A conformidade com padrões e melhores práticas de segurança é essencial.

Compatibilidade com dispositivos e Navegadores

Restrição: Estudantes e Docentes podem acessar o sistema a partir de vários dispositivos e navegadores. Impacto: A arquitetura deve ser responsiva e compatível com uma variedade de dispositivos e navegadores para garantir uma experiência do usuário sem problemas.

Armazenamento e recuperação de dados

Restrição: O armazenamento e a recuperação eficientes de dados de exames, resultados e informações relacionadas são cruciais para o desempenho do sistema. Impacto: Deve-se empregar estratégias de design e otimização de banco de dados para garantir acesso rápido aos dados relevantes, mantendo a integridade dos dados.

Estas restrições são muito importantes para o correto e bom funcionamento do sistema. Alguns destes pontos serão considerados até críticos, pelo que uma simples falha poderá descredibilizar

o mesmo. Neste sentido, estas restrição acompanhar-nos-ão durante todo o processo de conceção, desenvolvimento e manutenção.

Requisitos de Qualidade

No processo de planeamento da arquitetura de software, a análise dos requisitos não funcionais desempenha um papel crucial, moldando as decisões que sustentam a escolha da arquitetura escolhida para o desenvolvimento do sistema final.

Ao delinear e priorizar esses requisitos, é possível notar características comuns relativas às necessidades técnicas dos utilizadores e do ambiente operacional. Neste capítulo será efetuada essa análise, com a apresentação de uma lista dos requisitos finais que se apresentam com um papel essencial para a escolha da arquitetura a desenvolver.

O Sistema restringe o acesso a funcionalidades de acordo com o utilizador [RNF21]

Numa arquitetura baseada em microsserviços é expectável que diferentes serviços compreendam responsabilidades e funcionalidades distintas, sendo necessário garantir a segurança no acesso a estes serviços. A garantia de acessos personalizados a funcionalidades e informações sensíveis, consoante o perfil de cada utilizador, emerge como uma estratégia essencial para esse controlo de segurança.

Este controlo de acesso, não reforça apenas a segurança do Sistema, como também se alinha com os princípios de modularidade e independência dos microsserviços. Cada um poderá ser projetado para oferecer funcionalidades específicas, e a implementação de restrições ao seu acesso contribui para o seu isolamento eficaz.

Para além disso, a natureza flexível e escalável de uma arquitetura baseada em microsserviços contribui para que à medida que novos serviços sejam implementados ou atualizados, o controlo de acesso possa ser facilmente adaptável, contribuindo para a evolução do sistema sem comprometer a segurança do mesmo.

O Sistema guarda e valida palavras-passe através de técnicas criptográficas [RNF22]

O requisito não funcional 22, que enfatiza a necessidade de o sistema armazenar e validar passwords por meio de técnicas criptográficas, desempenha um papel crucial na definição da arquitetura do sistema. A exigência de encriptar as senhas torna a escolha de uma arquitetura em camadas menos viável, pois a criptografia e descriptografia dentro de um único processo podem ter um impacto significativo no desempenho.

Por outro lado, ao optar por uma arquitetura baseada em microsserviços, é imperativo que a informação seja encriptada para evitar interseções durante a comunicação entre a base de dados e a aplicação.

O sistema opera em modo local caso seja perdida a conexão com o servidor [RNF10]

A capacidade do Sistema permanecer em modo de funcionamento local na eventualidade de uma perda de conexão com o servidor, emerge como uma característica crucial para uma arquitetura baseada em microsserviços. Este aspeto é particularmente significativo no que diz respeito à tolerância a falhas e continuidade dos serviços, uma vez que o Sistema como um todo deverá ser resiliente o suficiente para continuar a suportar as funcionalidades essenciais, ainda que não exista uma conexão ao servidor.

Para além disso, esta característica alinha-se com os princípios fundamentais de independência e autonomia dos microsserviços de uma arquitetura deste tipo, contribuindo também para a resiliência do Sistema como um todo. Cada serviço poderá operar de forma independente e adaptando-se dinamicamente às condições do ambiente tais como eventuais falhas na rede.

O sistema deve ser escalável [RNF13]

Este requisito assume uma relevância extraordinária no contexto da arquitetura de microsserviços. Trata-se de um componente essencial na construção de sistemas capazes de evoluir e de se adaptar continuamente às transformações no ambiente operacional, nomeadamente através do

aumento da carga de trabalho, o crescimento dos dados e a evolução constante das necessidades do utilizador ao longo do tempo.

Numa arquitetura baseada em microserviços, o sistema é composto por vários serviços independentes. A escalabilidade permite distribuir as diferentes cargas de trabalho de forma eficiente entre esses serviços, contribuindo para diminuição do sobrecarregamento de um serviço ou eventuais *bottlenecks* de performance. Para além disso é ainda possível lidar com picos de tráfego sem comprometer o desempenho geral do sistema. Um aumento repentino no número de acessos de utilizadores ou na procura por determinados serviços pode ser devidamente atendido através da alocação dinâmica de recursos.

Árvore de Qualidade

Uma *Quality Tree* é uma técnica utilizada na área de controlo de qualidade para representar e organizar as diferentes dimensões ou aspectos da qualidade de um produto ou processo. Essa abordagem ajuda a identificar e visualizar de forma hierárquica os diversos elementos que contribuem para a qualidade global. Neste sentido, criámos a seguinte árvore de qualidade.

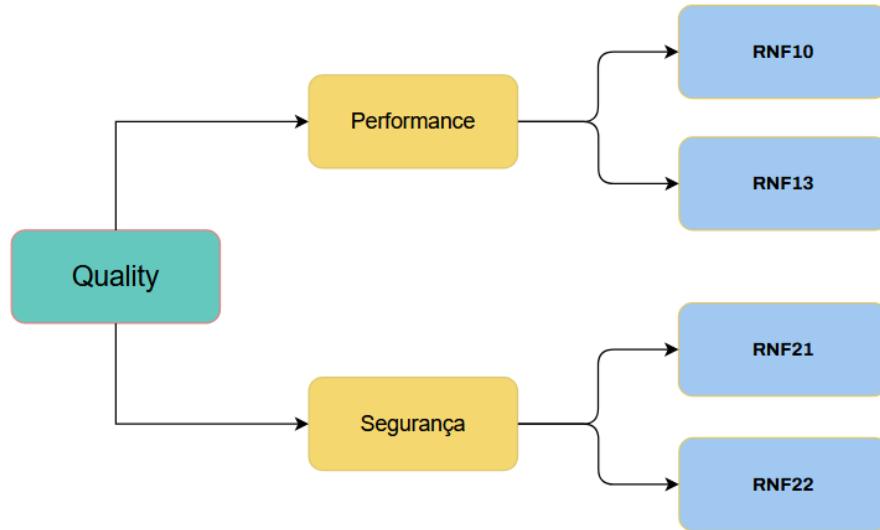


Figura 1: Quality Tree

A figura acima denota claramente os pilares para avaliação e garantia de qualidade do sistema, nomeadamente: Performance e Segurança. Os requisitos não funcionais que procuram assegurar fatores como a integridade, eficiência operacional e escalabilidade do sistema revelam-se como fundamentais para a arquitetura do sistema.

Extensão e Contexto do Sistema

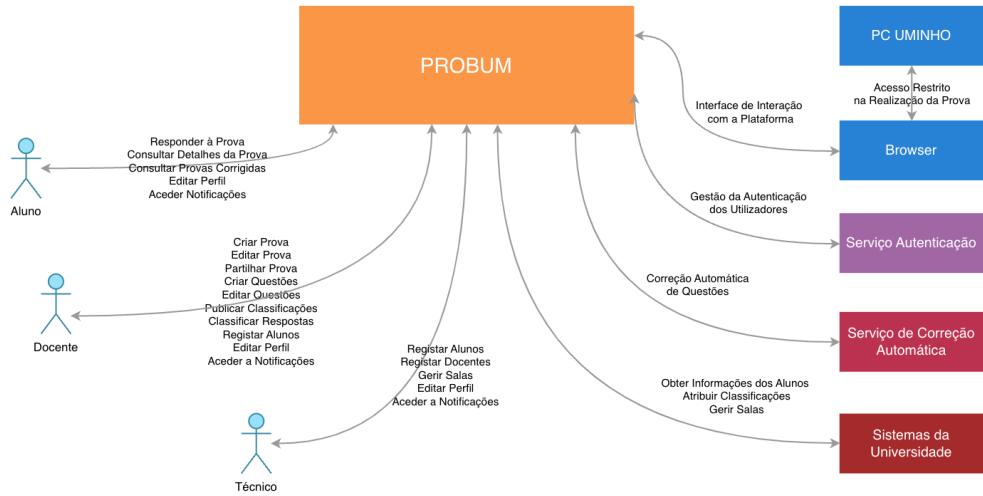


Figura 2: Diagrama de Contexto de Negócio

O diagrama de contexto de negócio ajuda a compreender as interações entre o sistema e o ambiente externo, destacando as principais entidades envolvidas, utilizadores, administradores e outras partes interessadas. Define claramente as fronteiras do sistema, mostrando quais elementos estão dentro do sistema e quais estão fora, assim como as suas interações com serviços externos. Desta forma, é considerado crucial para evitar confusões sobre a extensão do sistema a ser construído.

No caso do PROBUM, o diagrama apresenta todos atores que interagem com o sistema, assim como quais são as suas interações. Quanto à integração com sistemas externos, o PROBUM integra cinco sistemas diferentes:

- Browser - PC UMINHO : Definem a forma de interação que os utilizadores tem com a plataforma, no momento da realização das provas.
- Serviço de Autenticação : Este sistema pode ser considerado opcional, visto que já existem imensas soluções *standard* da indústria que facilitam a componente de autenticação. Dessa forma, de forma a reduzir tempo de desenvolvimento, a equipa pode optar por fazer a integração de um serviço externo de autenticação, como por exemplo, o OAuth 2.0.
- Serviço de Correção Automática : Este(s) sistema(s) seria(m) utilizado(s) em certo tipos de questões, como perguntas de código, diagramas, pautas música, entre outros de forma a agilizar a correção.
- Sistemas da Universidade : Nesta integração espera-se que o ProbUM se conecte com os sistemas já pre-existentes na Universidade, como a base de Dados com as informações de alunos (para registrar alunos e guardar informações do percurso académico) e o sub-sistema de gestão de espaços

Estratégia da Solução

Após examinarmos minuciosamente os requisitos e casos de uso delineados, procedemos à decomposição funcional do sistema a ser desenvolvido. Nesse processo, identificamos e agrupamos as principais atividades do sistema em subsistemas mais pequenos. A ideia subjacente é que esses subsistemas possam, numa fase subsequente, transformar-se em microsserviços que constituem a solução desejada.

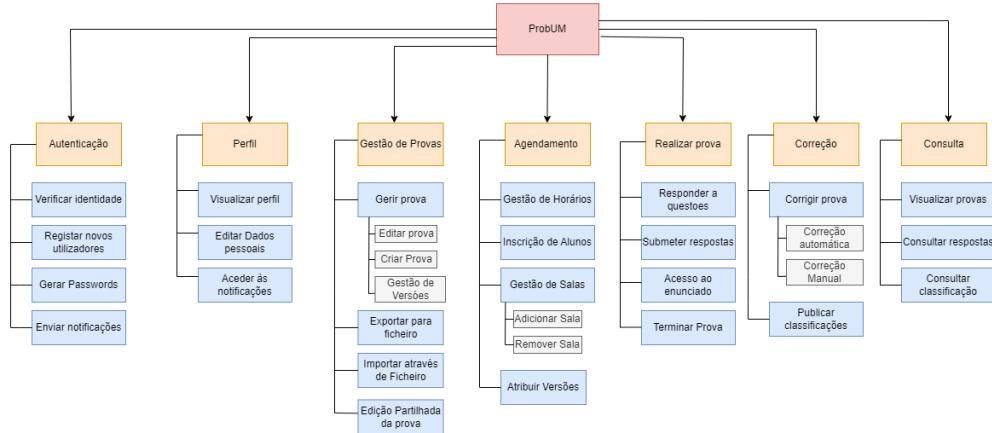


Figura 3: Diagrama de Decomposição Funcional

1. Autenticação

A autenticação é um pilar essencial para o sistema ProbUM. Surge da necessidade de controlar o acesso dos utilizadores à plataforma de forma a garantir a segurança e devida utilização do sistema. Será possível registar utilizadores, verificar a identidade de utilizadores, tal como gerar passwords aleatórias e enviar notificações a utilizadores.

2. Perfil

O perfil surge da necessidade de guardar e apresentar a informação de utilizadores, sejam docentes ou alunos, sendo possível visualizar o perfil, editar os dados pessoais e aceder às notificações recebidas.

3. Gestão de Provas

Baseado num dos maiores objetivos do projeto ProbUM que é a criação e realização de provas, a **Gestão de Provas** surge na medida em que todas as ações de gestão relacionadas com uma prova estão contidas neste grupo de funcionalidades. Com base nos requisitos existentes, as seguintes funcionalidades estão contidas neste grupo funcional

- **Gestão de Prova** No subsistema de gestão de provas, estão incorporadas as seguintes funcionalidades:

– Edição de prova

A Edição da prova é o grupo de funcionalidades mais complexo. Aqui estão contempladas todas as funcionalidades de edição como criar questões para uma prova que podem ser escolha múltipla, resposta restrita, Verdadeiro e Falso ou resposta Extensa. Também é possível editar perguntas de uma prova já existentes. O docente pode atribuir pontuações bem como as respostas esperadas para cada questão.

- **Criar prova**
O Docente pode criar provas, adicionando as várias questões da prova e as respetivas cotações.
- **Gestão de Versões**
O docente deve ser capaz de criar novas versões facilmente para uma prova e para posteriormente serem associadas a um momento de avaliação.
- **Exportar para ficheiro**
Deve ser possível exportar para um ficheiro uma prova que já exista, para que no futuro possa ser importada de novo.
- **Importar através de ficheiro**
Deve ser possível importar através de um ficheiro uma possível prova que já foi outrora exportado para o mesmo ficheiro.
- **Edição partilhada da prova**
A edição partilhada da prova permite que os Docentes partilhem uma prova com outros docentes via e-mail, o que permitirá que quem tiver acesso consiga realizar alterações e verificar eventuais erros na elaboração das questões.

4. Agendamento de Provas

- **Gestão de Horários**
Neste serviço o sistema é capaz de prever quais os horários disponíveis, bem como gerir horários de uma forma eficiente .
- **Inscrição de Alunos**
O docente associa uma lista de alunos admitidos a uma prova.
- **Gestão de Salas**
Neste serviço o sistema possui um mecanismo que permite facilmente verificar quais as disponibilidades de cada sala.
- **Atribuir Versões**
A quando do agendamento de uma prova o docente possui uma opção em que é possível adicionar diferentes versões a diferentes salas.

5. Realizar Prova

Neste microsserviço, os alunos têm a oportunidade de realizar as provas. Inicialmente, o Aluno tem acesso à prova que pretende realizar. Após a seleção, o Sistema apresenta o enunciado de cada questão de maneira sequencial. O aluno responde a cada questão, no respetivo espaço, e, em seguida, submete as suas respostas. Estes passos são repetidos até que o Aluno conclua a prova ou até que o sistema encerre automaticamente a Prova quando o tempo designado se esgotar.

6. Correção

Após a conclusão de todas as provas, é realizada a correção das mesmas. O Docente seleciona a prova que deseja corrigir e especifica quais perguntas devem ser corrigidas automaticamente pelo Sistema. As restantes perguntas são corrigidas manualmente pelo Docente. Uma vez concluída a correção de todas as provas, o docente publica as respetivas classificações.

7. Consulta

Este grupo funcional lida com as informações relacionadas as consultas, onde permite que os alunos visualizem provas corrigidas, consultem informações das respostas e classificações, sendo estas publicadas anteriormente por um Docente. Tendo como funcionalidade da consulta:

- Visualizar provas
O aluno pode visualizar as questões da Prova
- Consultar Respostas
O aluno pode visualizar as respostas que deu a cada uma das questões da prova
- Consultar Classificação
O aluno pode consultar a classificação da sua prova

Visualização dos Blocos de Construção

Após realizarmos a decomposição funcional, decidimos construir os diagramas de sequência (um por Use Case mencionado no documento de requisitos), para obter uma primeira iteração e identificar as interações entre os grupos funcionais decididos na decomposição funcional. Desta forma seria mais fácil identificar/ definir quais seriam os microsserviços do produto final.

Posto isto, tentamos identificar quais seriam os grupos funcionais que poderiam ser unificados para evitar comunicações desnecessárias entre microsserviços, bem como evitar a duplicação de dados, em dois microsserviços distintos. Esta última restrição é ainda mais importante devido ao facto de quebrar o conceito de microsserviço, que deve ser independente de todos os outros, evitando possíveis redundâncias e inconsistências nos mesmos. Percebemos facilmente que os grupos funcionais mencionados na figura (decomposição funcional inicial) Consulta e correção poderiam ser unificados de forma a partilharem a mesma base de dados na qual é armazenada a resposta do aluno a cada questão, bem como a correção da resposta que poderia ser automática ou manual. A autenticação e o perfil também poderiam partilhar a mesma base de dados, visto que realiza operações sobre utilizadores/perfil/notificações/registo/autenticação, e poderia tudo ser comprimido num microsserviço mais complexo, mas que poupará diversos pedidos , caso estes fossem microsserviços diferentes.

Obtemos então os seguintes microsserviços com a descrição do que está incluído em cada um deles na figura abaixo:

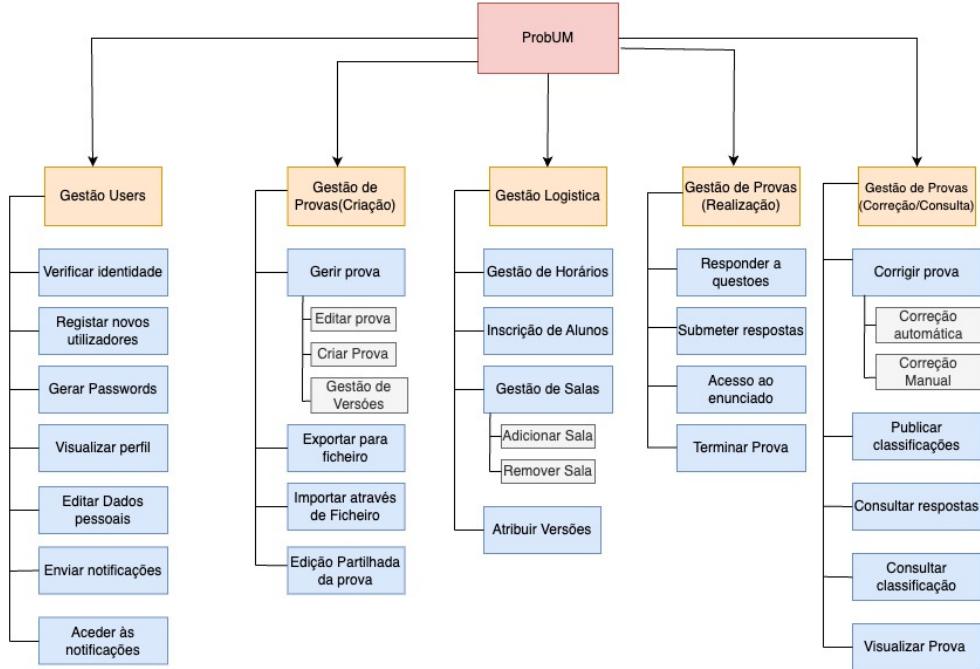


Figura 4: Diagrama com os microsserviços finais

Através do diagrama da figura acima conseguimos obter um conjunto de grupos funcionais que serão convertidos em microsserviços no produto final. É esperado que cada um destes microsserviços possua uma base de dados independente, e que disponibilize uma API que possa ser accedida pelos outros microsserviços e recolhida numa **API GATEWAY** que realiza toda a interação entre o Front-end e os Pedidos para o Back-end e os respetivos microsserviços.

A reestruturação dos microsserviços face à decomposição funcional é fundamentada na necessidade de melhorar a eficiência e modularidade do sistema. A abordagem consiste em criar três fases distintas para uma prova - sua criação, realização e correção/consulta - cada uma operando como um microsserviço independente. Além disso, procura-se agrupar operações logísticas relacionadas a horários, salas e alunos, visando separar a gestão operacional. Paralelamente, operações ligadas ao perfil, autenticação e notificações serão consolidadas num microsserviço único, simplificando a administração e manutenção dessas funcionalidades. Essa reorganização visa não apenas a eficácia técnica, mas também a facilidade de compreensão e manutenção do sistema como um todo.

Ainda dentro da Visão por Blocos criamos um diagrama de componentes que fornece uma visão de alto nível da arquitetura de software, mostrando como os diferentes componentes do sistema estão organizados e como eles se relacionarão entre si. A finalidade principal do mesmo é ajudar na compreensão da estrutura do sistema, identificando os principais componentes e mostrando como eles comunicam entre si.

Diagrama de Componentes

Definidos os microsserviços, não existe a noção de como estes irão comunicar de forma a disponibilizar todas as tarefas aos vários tipos de utilizadores. Neste sentido, surgiu a necessidade de apresentar um modelo que se foque neste problema. O diagrama de componentes apresentado abaixo mostra-nos a interação entre

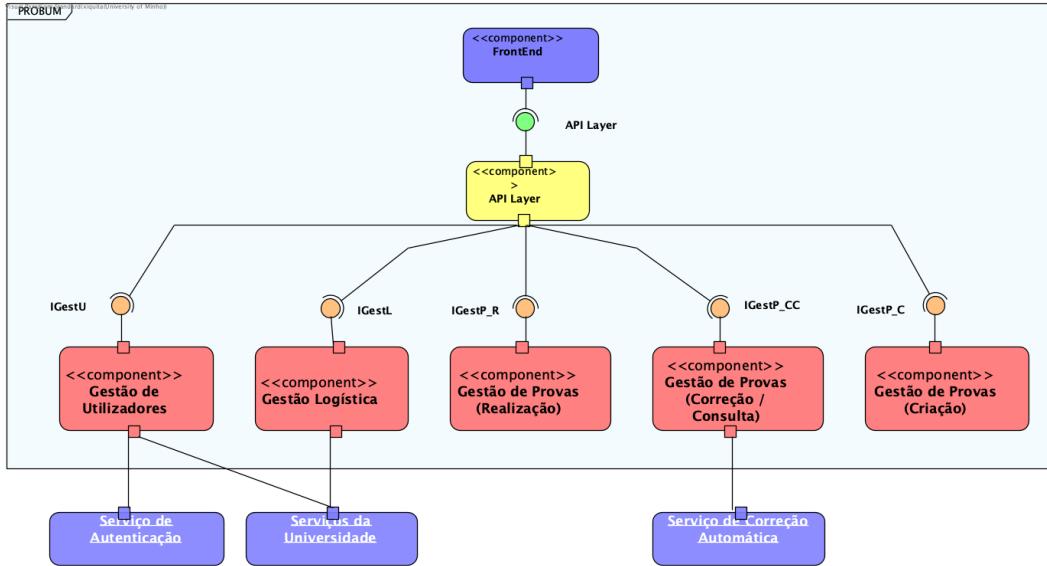


Figura 5: Diagrama de componentes

O Ponto de acesso ao sistema ProbUM é o FrontEnd. Este é o componente responsável por permitir aos utilizadores realizarem as várias tarefas encapsulando todo o processo que está a acontecer por detrás. Este componente é abastecido pelo APILayer que é responsável por contactar os vários microsserviços de forma a obter dados ou a fazer atualizações nos mesmos. A APILayer é o controlador que gere todo o sistema realizando chamadas às API's de cada um dos microsserviços. Os microsserviços também estabelecem conexão com alguns serviços externos como foi explicado no Diagrama de Contexto de Negócio.

Microsserviços

Dado como concluídos o diagrama de componentes, decidimos, para cada microsserviço, construir um diagrama de classes, que, de uma forma mais específica representa a estrutura estática de um sistema orientado a objetos, bem como fornece uma visão visual das classes no sistema, juntamente com seus atributos, métodos e os relações entre elas. De seguida estão apresentadas as classes, separadas por microsserviço.

Microsserviço Gestão de Utilizadores

Decidimos desenvolver uma classe principal denominada Utilizador que contém atributos essenciais para cada utilizador, tais como email, password, nome, número e um indicador do tipo de utilizador. Se um utilizador é designado como gestor, ele possui a capacidade de criar alunos. No caso de um Técnico, a funcionalidade permitida é a criação de docentes. Para um Docente, entretanto, o acesso é exclusivamente restrito às provas associadas a ele. Todos os utilizadores podem realizar as seguintes ações: alterar email, alterar password ou até mesmo gerar uma password aleatória. A API serve como ponto de passagem para os outros microsserviços conseguirem obter informações do microsserviço de Gestão de utilizadores, esta possui ações possíveis como obter dados pessoais, verificar autenticação, criar utilizador, ou até mesmo apagar um utilizador.

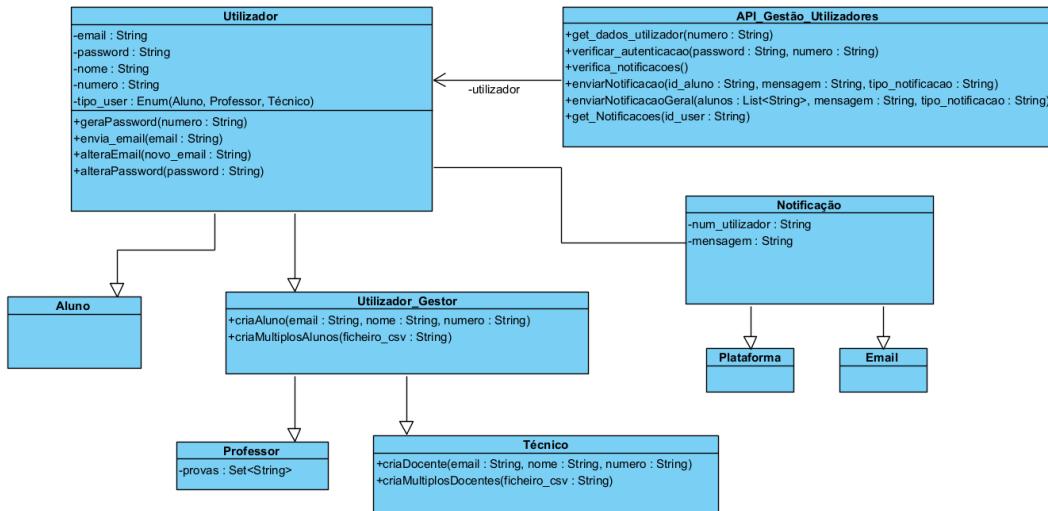


Figura 6: Diagrama de Classes para Gestão de Utilizadores.

Microsserviço Gestão de Provas(Criação)

O microsserviço Gestão de Provas é responsável por armazenar todas as informações relacionadas às provas e os seus detalhes. Este sistema é projetado de forma a garantir que apenas docentes tenham a capacidade de criar novas provas. Cada prova contém informações cruciais, incluindo a identificação do docente responsável pela criação. Além disso, contém uma lista de utilizadores que têm acesso à prova, a versão correspondente, a data e hora preferenciais para a realização, o tempo total de duração da prova e o tempo permitido para a admissão. Para além destes atributos, cada prova terá associado um conjunto de questões. Uma Questão tem associado um id, uma cotação total, um enunciado e pode permitir uma imagem. Além disso uma Questão apresenta vários tipos: Verdadeiro e Falso, Escolha Múltipla em ambos são armazenadas as opções e as cotações, Desenvolvimento onde é armazenada a cotação e a resposta e Espaços que contêm os espaços, o texto e as cotações. Existem ainda todos os métodos relativos a edição,

criação ou até mesmo remoção de questões de uma prova. Na API deste microsserviço são disponibilizados métodos como obtenção da prova, consultar uma prova em específico, importar ou exportar prova, e obtenção da versão.

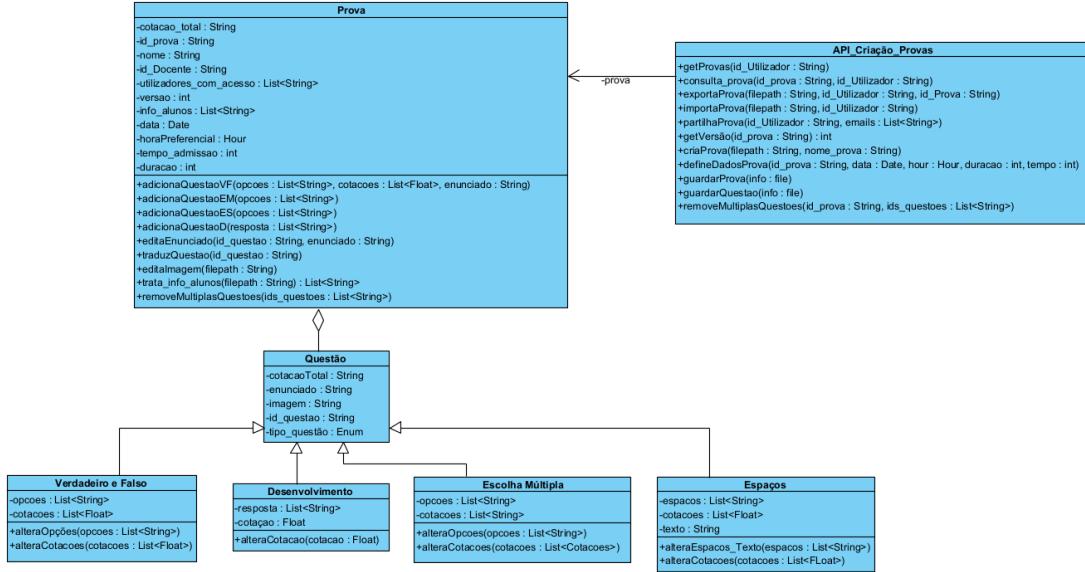


Figura 7: Diagrama de Classes para a Criação de Prova.

Microsserviço Gestão Logística

A Gestão Logística assenta no envolvimento de tudo o que esteja relacionado com horários, inscrição de alunos e gestão de salas. Cada sala é identificada por um ID único, o ID do edifício ao qual pertence e sua capacidade. Associada a esta classe, apresentamos a classe Horário, que inclui informações como hora, disponibilidade, ID da prova associada, ID do docente responsável pela criação da mesma e a lista dos alunos agendados para realizá-la. O Agendamento, tendo em conta os detalhes introduzidos pelo docente, calcula a disponibilidade de salas e horários e propõe-nos ao docente, através de métodos essenciais, como a validação do horário, onde permitimos que o docente aceite esse agendamento proposto. Caso a sua proposta não seja aceite, é possível editá-la ou aceitar uma diferente. Um método adicional permite solicitar a disponibilidade, retornando uma lista das salas disponíveis para a realização da prova. Também é possível associar alunos à prova para que o sistema tenha acesso à quantidade de alunos envolvidos.

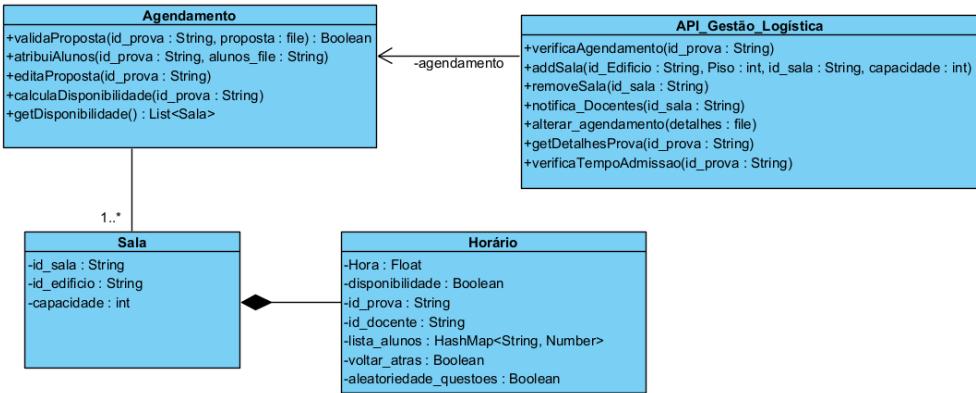


Figura 8: Diagrama de Classes para a Gestão Logística.

Microsserviço Gestão de Provas(Realização)

Como seria de esperar, no microsserviço dedicado à realização de provas, o utilizador mais frequente é o Aluno. Dessa forma, o grupo optou por armazenar, para cada realização, o ID da prova para identificar qual é a prova, o ID do aluno que a realiza, a lista de respostas desse aluno e, para cada resposta, um identificador correspondente a cada questão, juntamente com a respectiva cotação. Foram incorporados métodos como "submit answers", "show question" e "show prova", que serão úteis na parte da interação entre os microsserviços, principalmente nas interações entre o Front-end e o microsserviço de realização de provas.

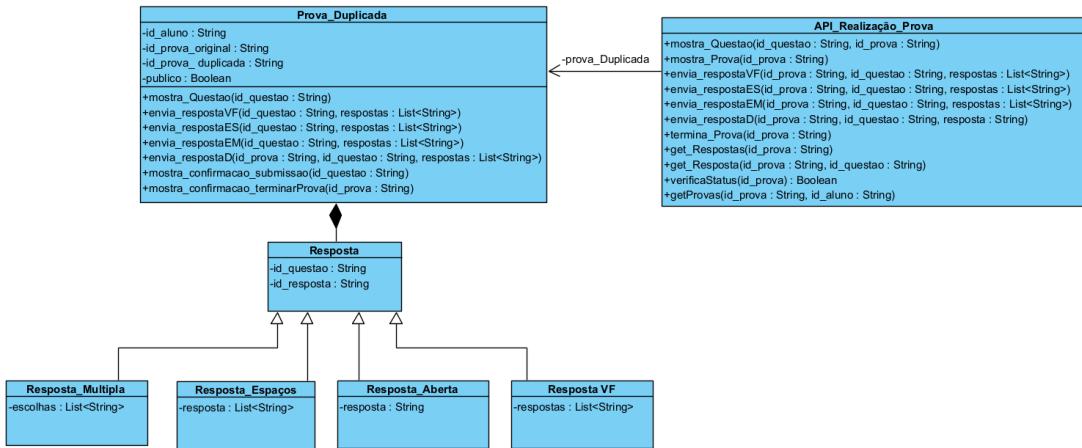


Figura 9: Diagrama de Classes para a Realização de prova.

Microservice Gestão de Provas(Correção/Consulta)

Neste microserviço, como o próprio nome indica contém todas as informações relativas à correção da prova que posteriormente poderá ser consultada. Desta forma a prova realizada por um aluno com a lista de questões ficara aramazenada neste microserviço, sendo que para cada resposta dada existe um campo cotação que inicialmente é nulo e após a correção é-lhe atribuido um valor.

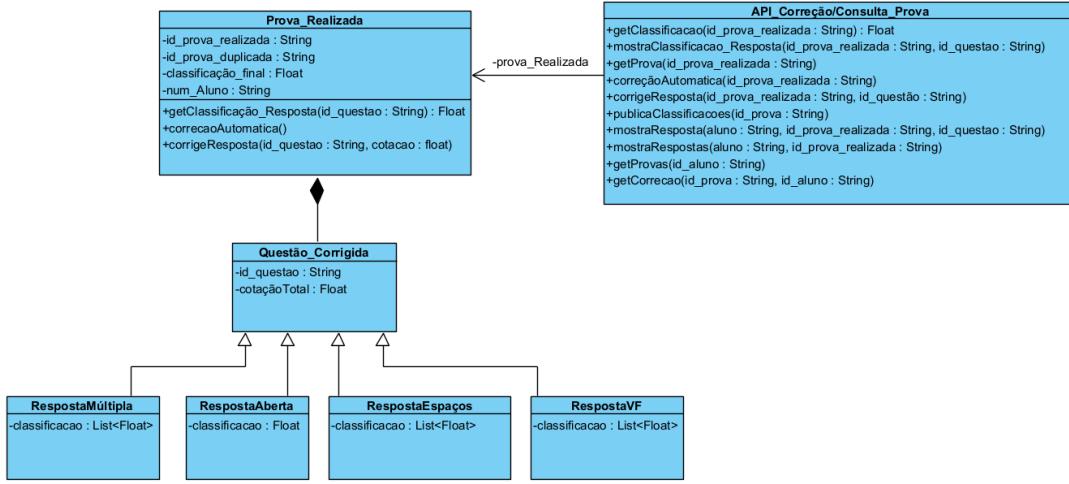


Figura 10: Diagrama de Classes para a Realização de prova.

Visualização em Tempo de Execução

A **Runtime View** destaca o comportamento dinâmico do Sistema, proporcionando uma compreensão abrangente de como os componentes interagem para atender às necessidades operacionais do PROBUM. Estes diagramas de sequência abordam **Use Cases** e recursos críticos, ilustrando as interações entre os microsserviços. Cada diagrama captura uma narrativa específica, revelando a dinâmica precisa entre os vários componentes internos do sistema.

O desenvolvimento dos Diagramas de Sequência ocorreu em duas fases distintas. Na primeira etapa, adotou-se uma abordagem baseada na decomposição funcional, considerando os diagramas como uma primeira iteração, equivalente a um rascunho inicial. Após analisar a primeira iteração de todos os diagramas, procedeu-se à reestruturação da arquitetura de microsserviços, resultando na Figura 5 do Capítulo Building Block View. Na segunda fase, a equipa concentrou-se na criação de diagramas com base na nova arquitetura, bem como nos diagramas de classes de cada microsserviço, alcançando assim uma segunda iteração que se aproximou consideravelmente da solução final de implementação.

Em seguida, são apresentados todos os Diagramas de Sequência com breves descrições, a sua 1^a e a 2^a iteração.

Registrar Docentes - *Use Case 1*

Este diagrama de sequência mostra a interação de um utilizador com o sistema de forma a representar a funcionalidade de registrar docentes no sistema. O registo dos docentes pode ser realizado seja através de um formulário individual como através da submissão de um ficheiro. O processo é muito simples segundo qualquer das hipóteses. No entanto o diagrama de sequência torna-se extenso devido ao tratamento de erros que o sistema terá que verificar e que já tentámos especificar aqueles que serão mais comuns. Quanto às alterações da primeira para a segunda versão apenas houve a alteração do nome do microsserviço (Autenticação → Gestão Users)

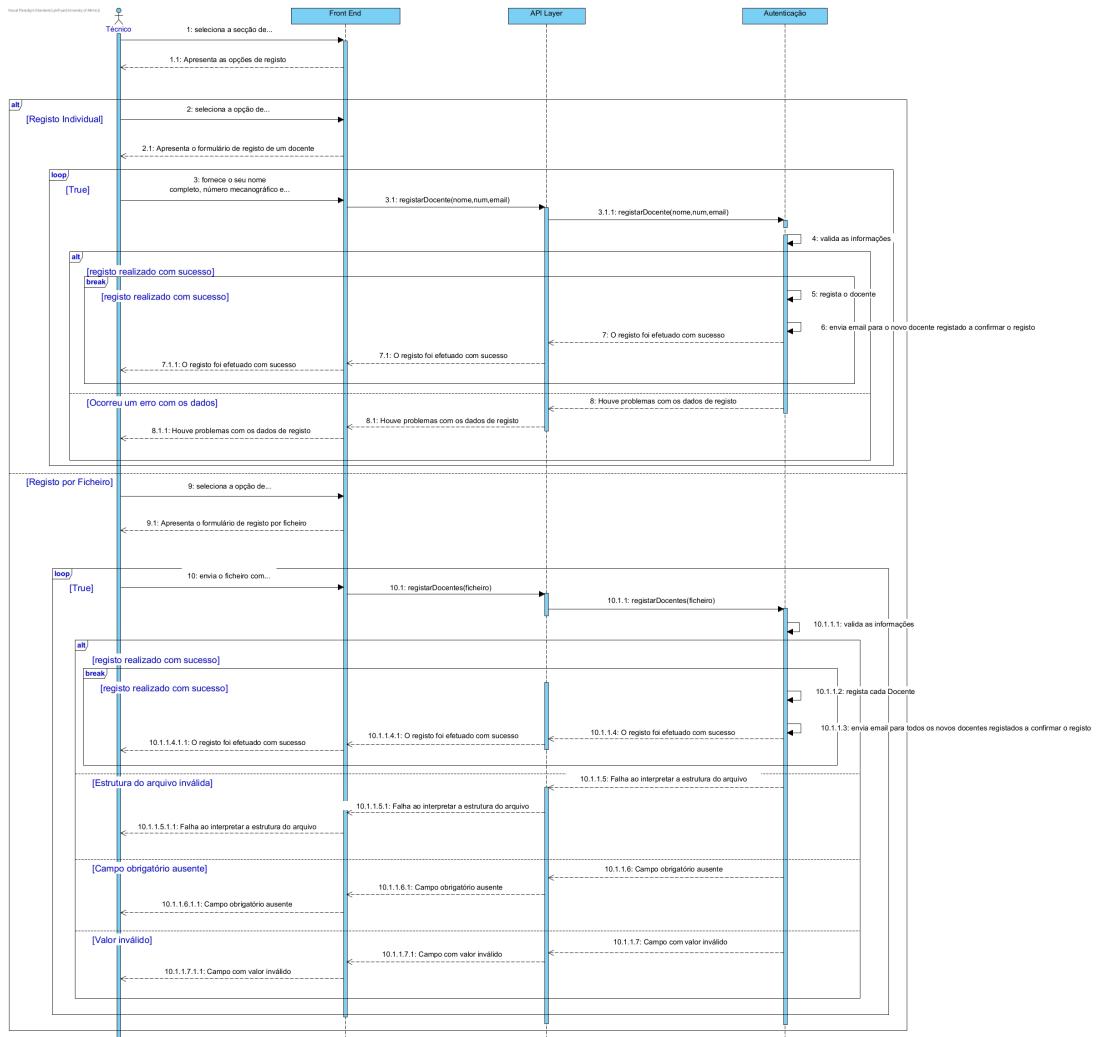


Figura 11: 1^a iteração do Diagrama de Sequência para Registar Docentes.

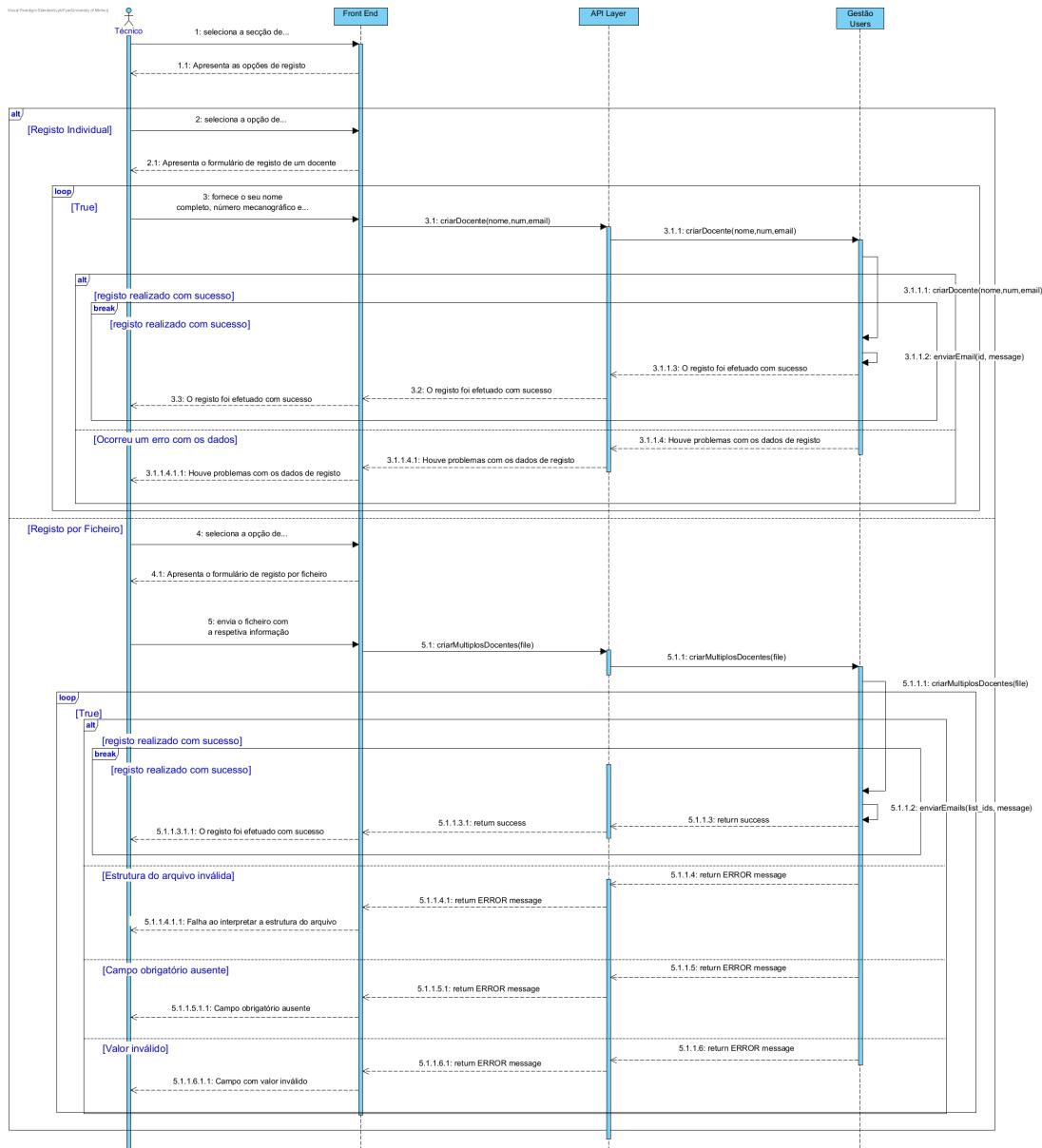


Figura 12: 2^a iteração do Diagrama de Sequência para Registar Docentes.

Registrar Alunos - *Use Case 2*

Este diagrama é muito semelhante ao diagrama que permite o registo de docentes sendo a estrutura igual mas alterando o nome dos métodos.

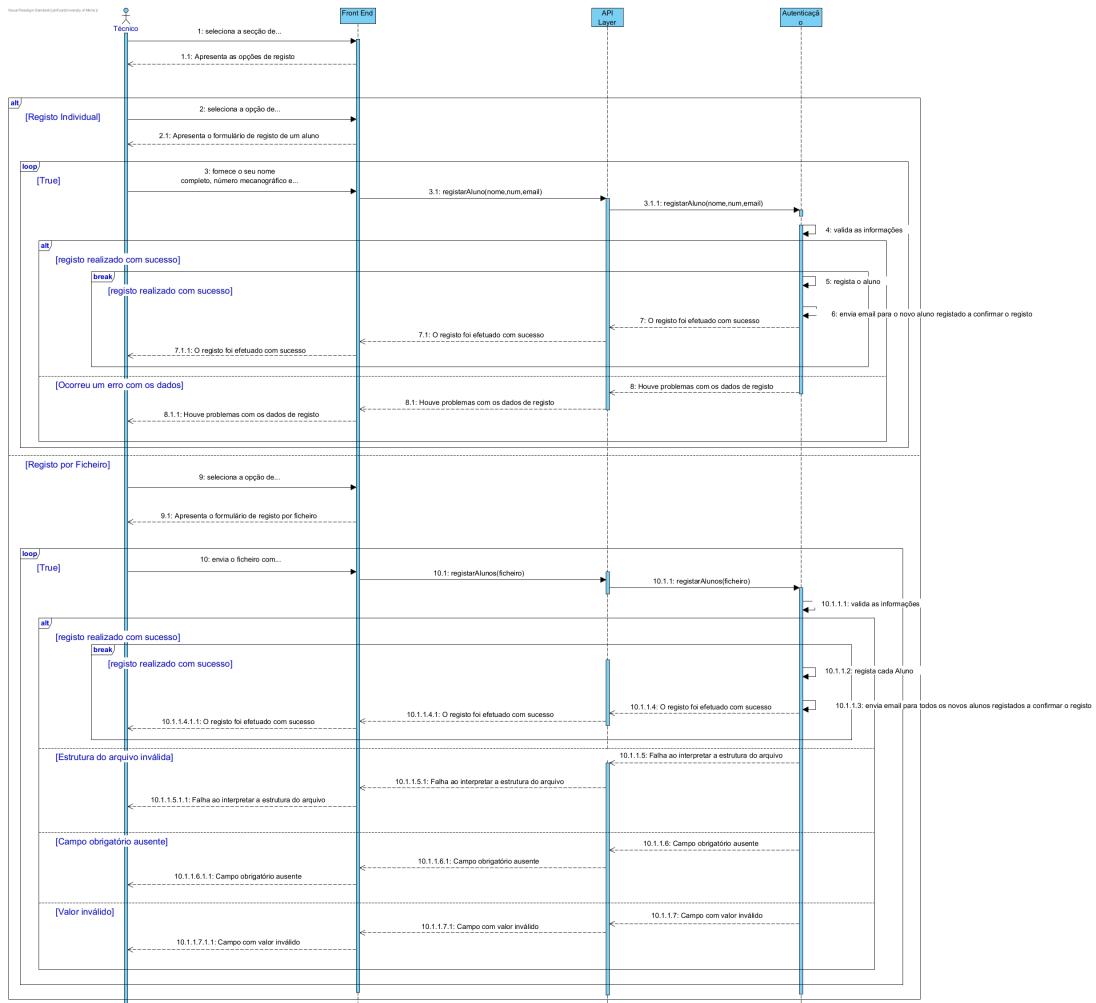


Figura 13: 1^a iteração do Diagrama de Sequência para Registar Alunos.

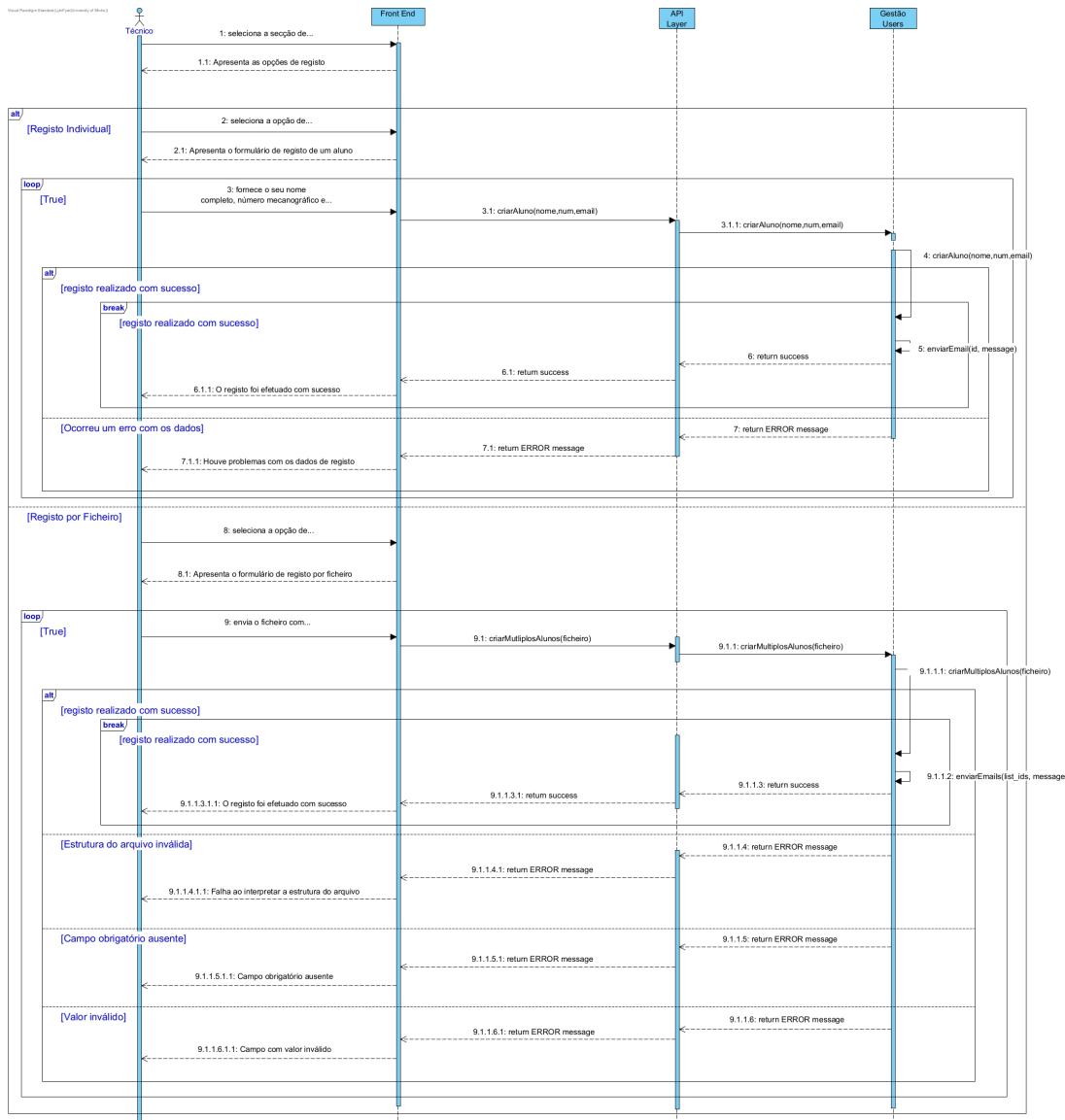


Figura 14: 2^a iteração do Diagrama de Sequência para Registar Alunos.

Autenticar - *Use Case 3*

Este é o diagrama de sequência que permite autenticar o utilizador. O utilizador começa por aceder à página de *login*, inserir as suas credenciais, e depois o sistema trata da correção de eventuais erros que possam acontecer, nomeadamente o utilizador não estar registado ou a password estar incorreta. As alterações da primeira fase para a segunda foram apenas no nome do microsserviço.

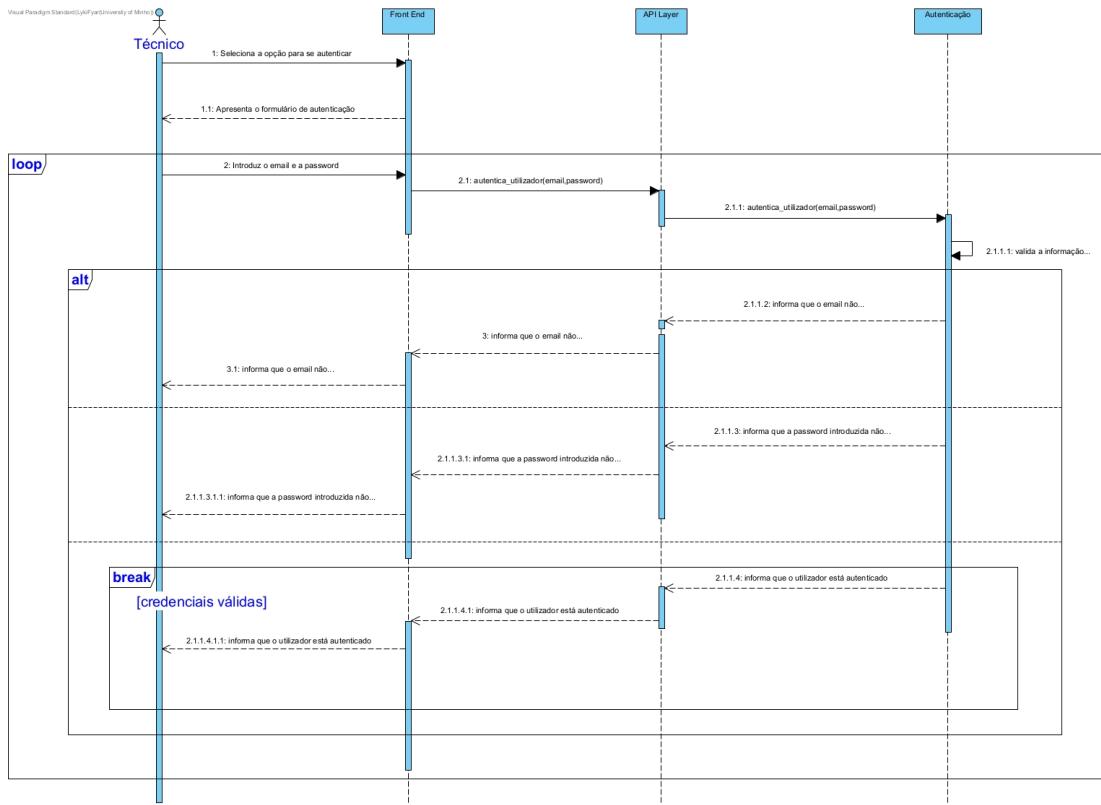


Figura 15: 1^a iteração do Diagrama de Sequência para Autenticar.

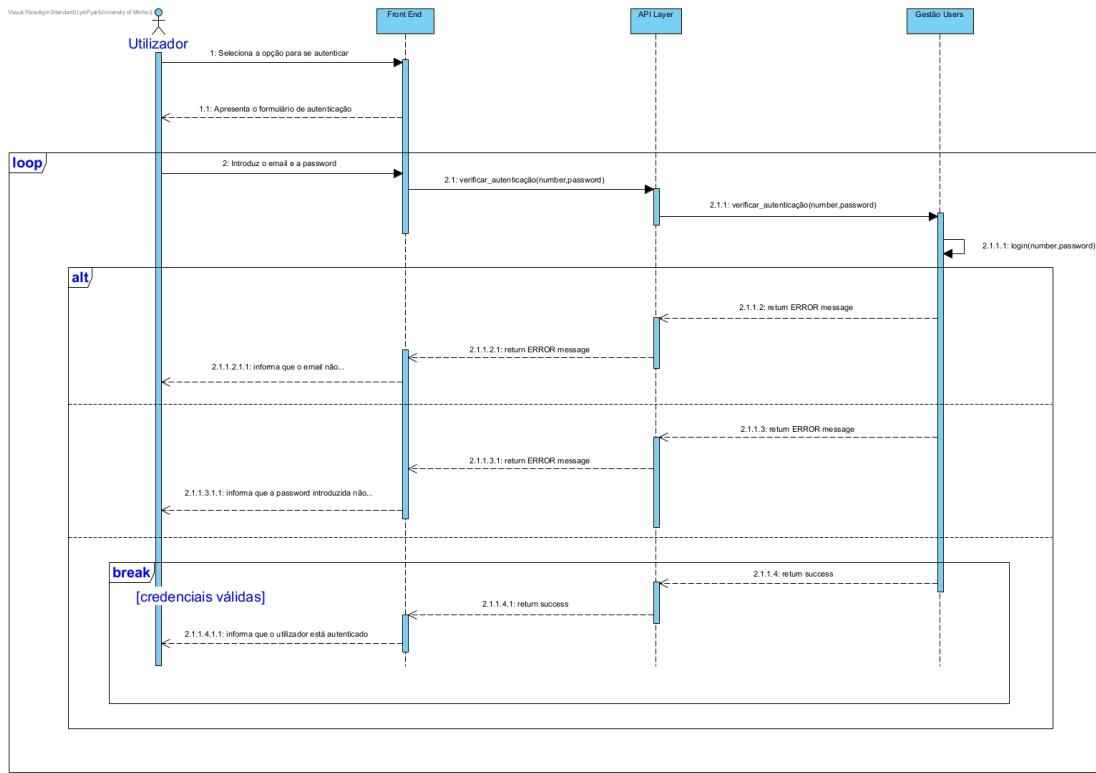


Figura 16: 2^a iteração do Diagrama de Sequência para Autenticar.

Editar Perfil - *Use Case 4*

Neste diagrama, considerou-se que um utilizador só consegue aceder às informações do seu perfil e que o mesmo está autenticado. O utilizador começa por selecionar a opção de editar perfil, o *Front end* irá comunicar com a *api layer* que por sua vez irá pedir as informações do perfil ao microserviço de Gestão de Users. Após isso, o utilizador pode alterar o seu email ou password, para ambos os casos temos validação do input por parte do microserviço. A diferença entre a primeira iteração e a segunda evidencia-se apenas no nome do microserviço que passa de Perfil para Gestão de Users.

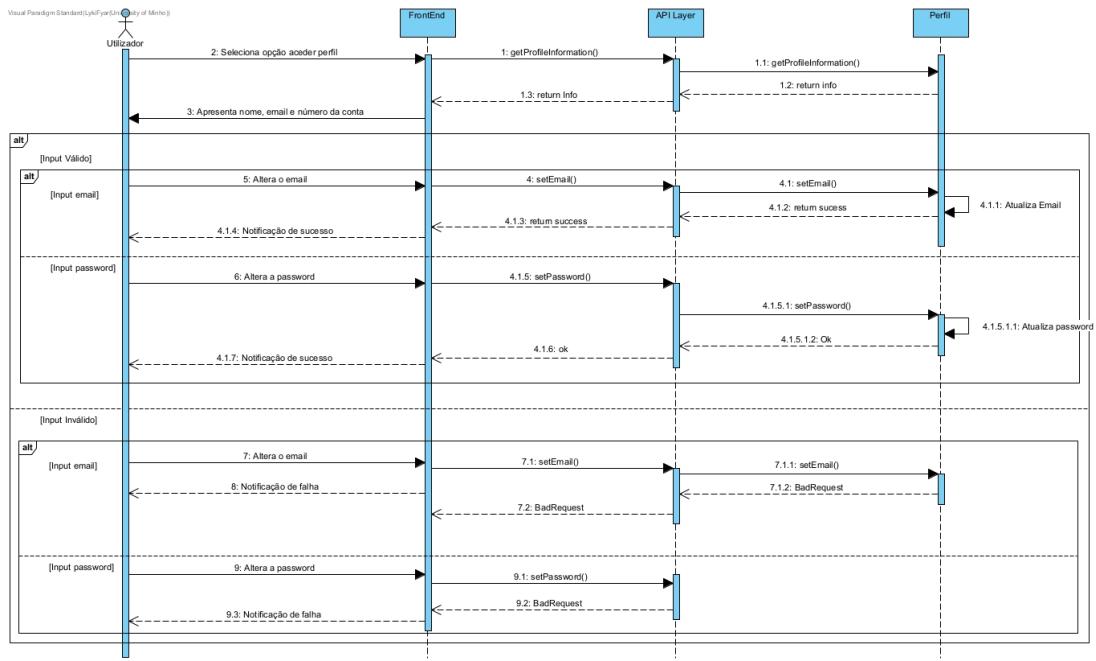


Figura 17: 1^a iteração do Diagrama de Sequência para Editar Perfil.

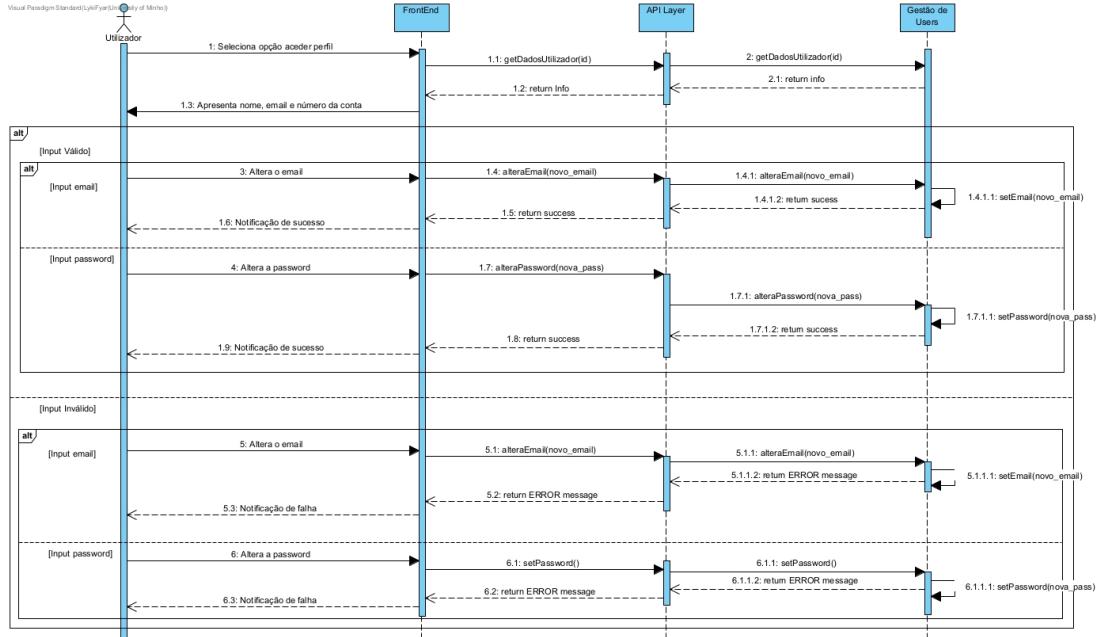


Figura 18: 2^a iteração do Diagrama de Sequência para Editar Perfil.

Criar Prova - Use Case 5

Neste diagrama, explicamos a lógica de criar e agendar a prova seguindo a lógica do use case do documento de requisitos, portanto, apresentamos as etapas mencionadas no documento de requisitos.

A semântica das operações é semelhante entre as duas iterações, mudando o nome dos microsser-

viços envolvidos.

Começamos por explicar o processo de criação do nome da prova e inscrição dos alunos na prova, esse processo poderá ser repetido até o utilizador indicar um ficheiro válido dos alunos. O Microsserviço Gestão de Provas(criação) pede ao microsserviço de Gestão User para validar o ficheiro de alunos.

Seguidamente temos outro processo iterativo na introdução do cronograma da prova que se irá repetir até o cronograma oferecer um conjunto de propostas para as salas, dentro desse conjunto o docente poderá iterar sobre as diferentes as propostas e escolhe uma. O microsserviço Gestão de Provas(criação) constrói o pedido de disponibilidade e pergunta ao microsserviço Gestão Logística para calcular a disponibilidade e se a disponibilidade for válida envia a informação do cronograma para o docente até o mesmo aceitar como referido anteriormente.

Após isso, o docente cria diferentes versões, e para cada versão cria diferentes questões, a criação de cada uma das questões é referenciada no Use case 6, se o Docente pretender guardar a versão, envia toda a informação criada relativamente a essa versão para o microsserviço Gestão de Prova(criação) através da *api layer*.

Terminada a criação da prova, os dados são guardados no microsserviço Gestão de Prova(criação) e notifica os alunos inscritos na prova através do microsserviço Gestão de Users.

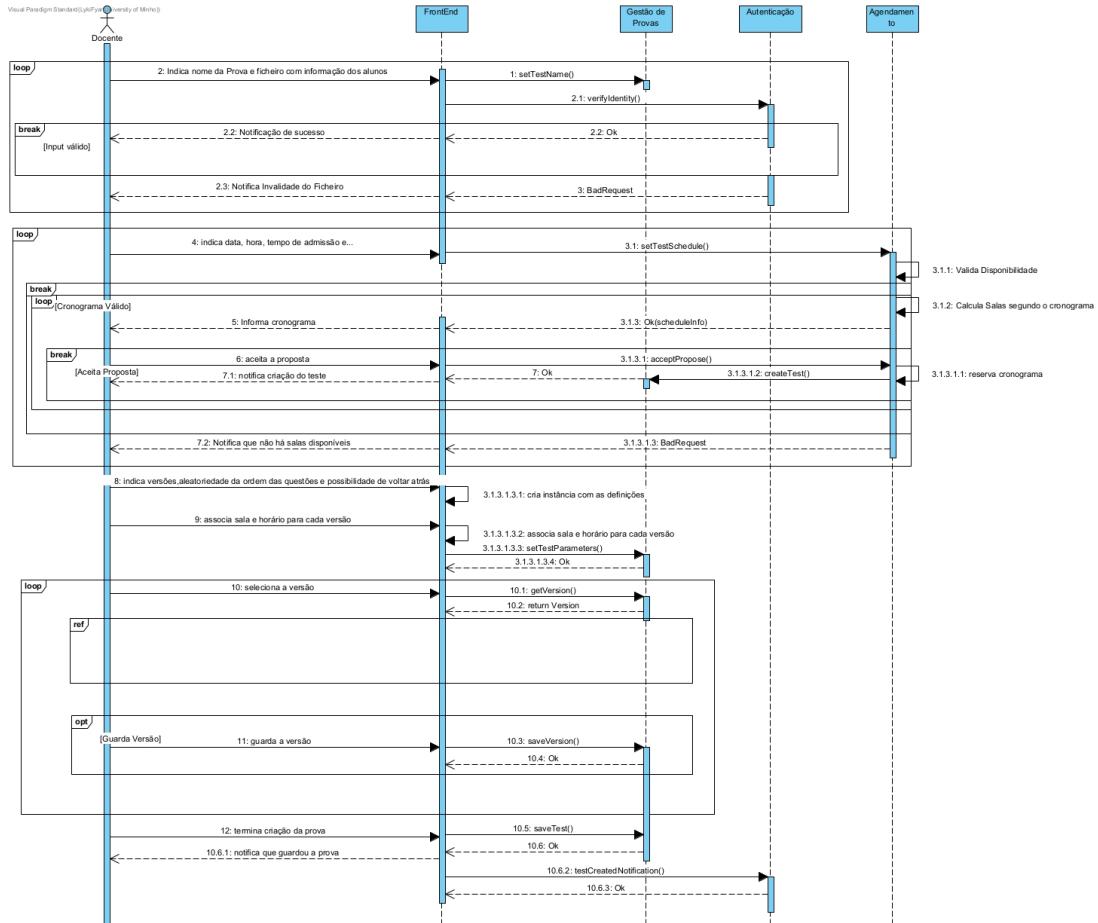


Figura 19: 1^a iteração do Diagrama de Sequência para Criar Prova.

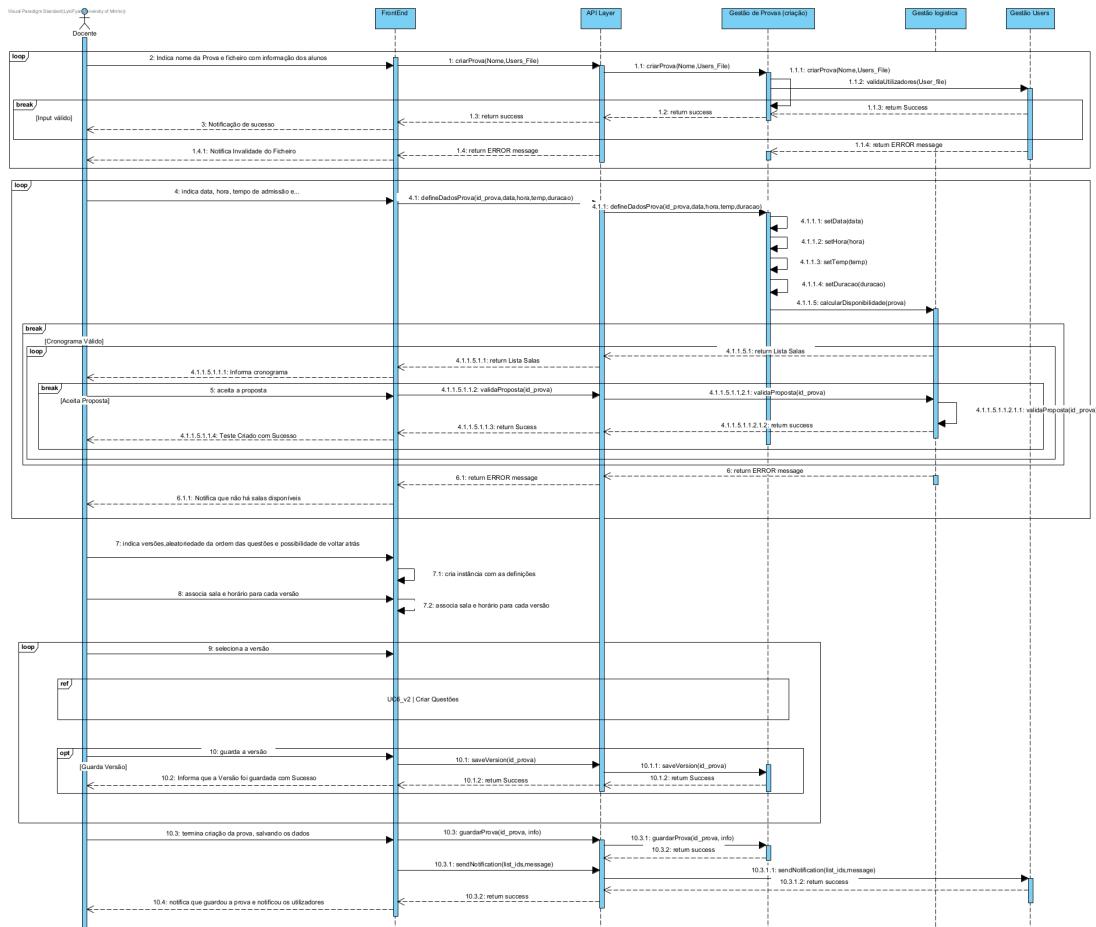


Figura 20: 2^a iteração do Diagrama de Sequência para Criar Prova.

Criar Questões - *Use Case 6*

O criar Questões está inserido tanto no diagrama de criar prova tanto no de editar prova. O único microsserviço que é afetado por este use case é o microsserviço de Gestão de Provas que passa a se chamar Gestão de Provas(criação) na segunda iteração, e por consequência, ambas as iterações diferem apenas no nome do microsserviço em questão.

O processo de criação da questão é feito através da inserção de diferentes inputs por parte do utilizador como inserir imagem ou não e depois inserir o tipo da questão e os seus parâmetros. O processo de construção da pergunta é feito todo no *Front End* e no final a Questão é enviada para o microsserviço, passando pela *api layer*. Deste modo, evitamos comunicações excessivas entre o *Front End* e o microsserviço.

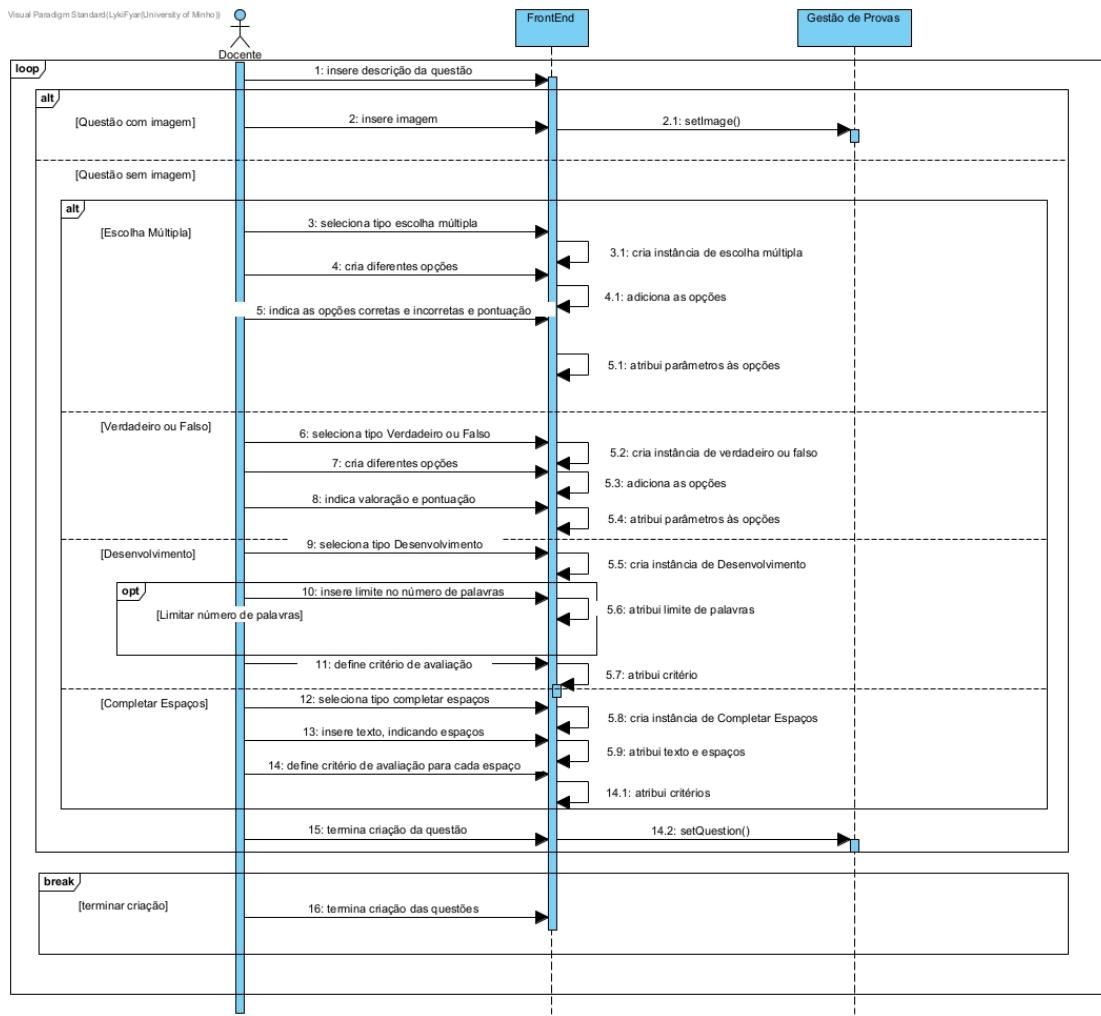


Figura 21: 1^a iteração do Diagrama de Sequência para Criar Questões.

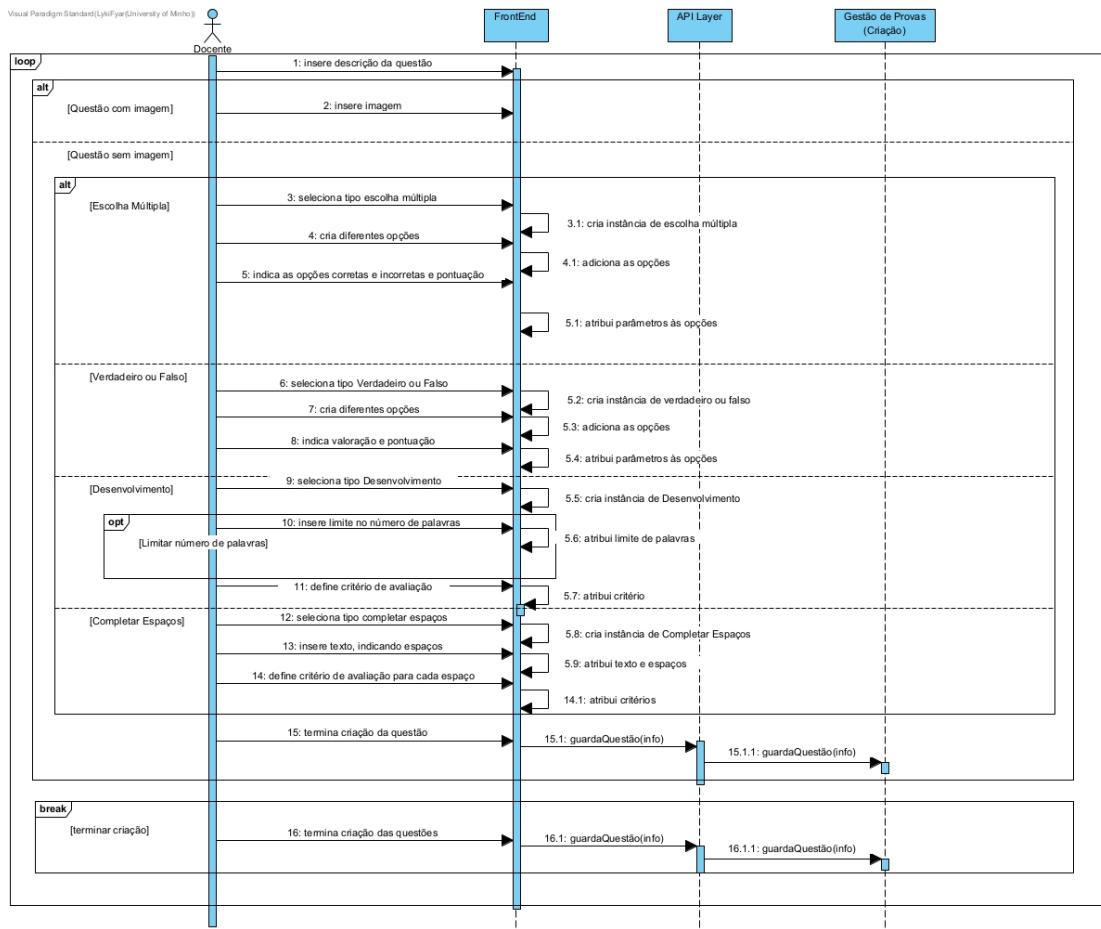


Figura 22: 2^a iteração do Diagrama de Sequência para Criar Questões.

Editar Prova - *Use Case 7*

Neste diagrama, utilizou-se uma versão análoga da edição e validação das salas, horários e durações que a usada em Criar Prova. No entanto, o *Use Case* em que se baseia não identificava este problema expandido o suficiente, então foi considerado o mais detalhado presente no *Use Case* de Criar Prova.

A única alteração entre as versões do diagrama foram os nomes dados aos Microsserviços.

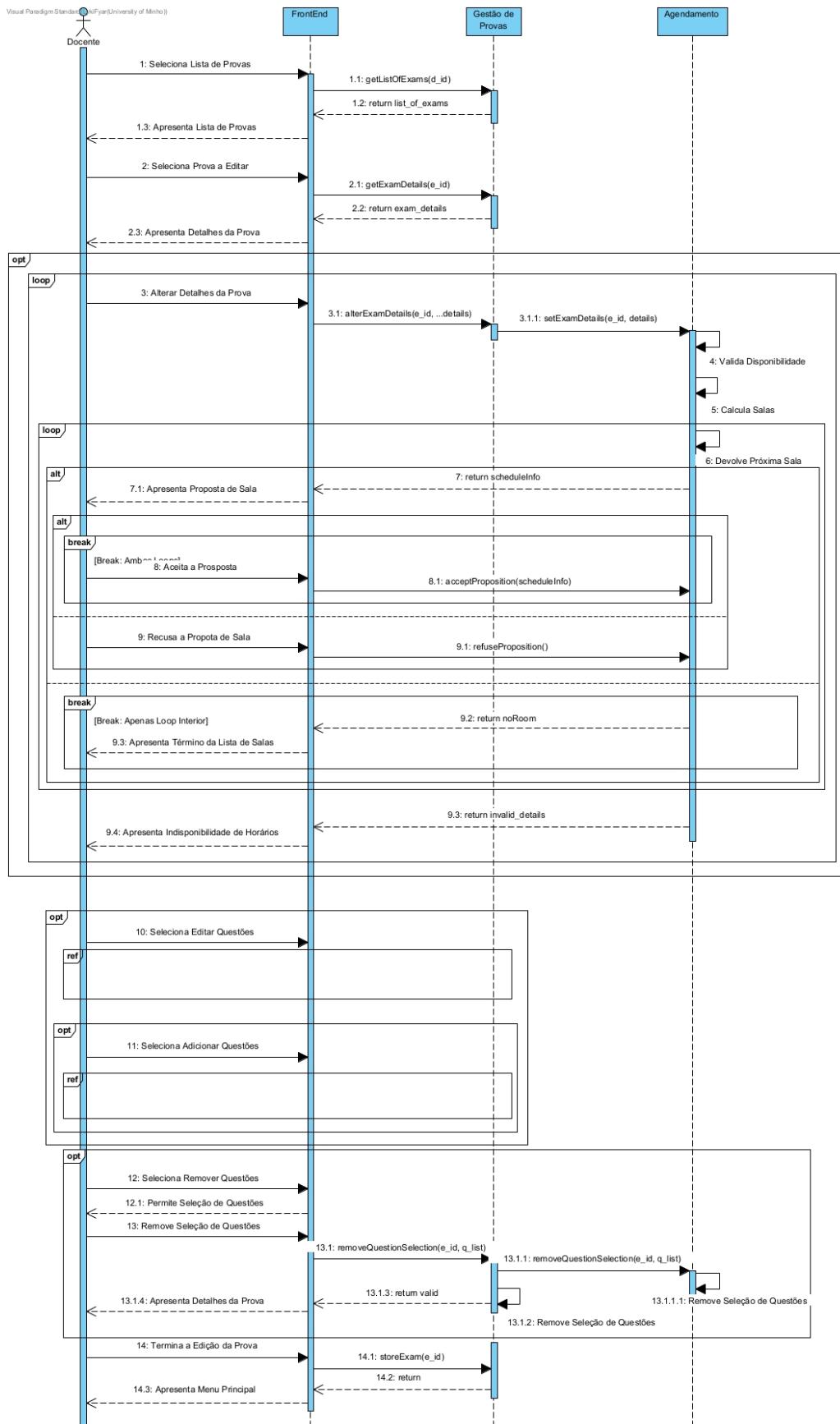


Figura 23: 1^a iteração do Diagrama de Sequência para Editar Prova.

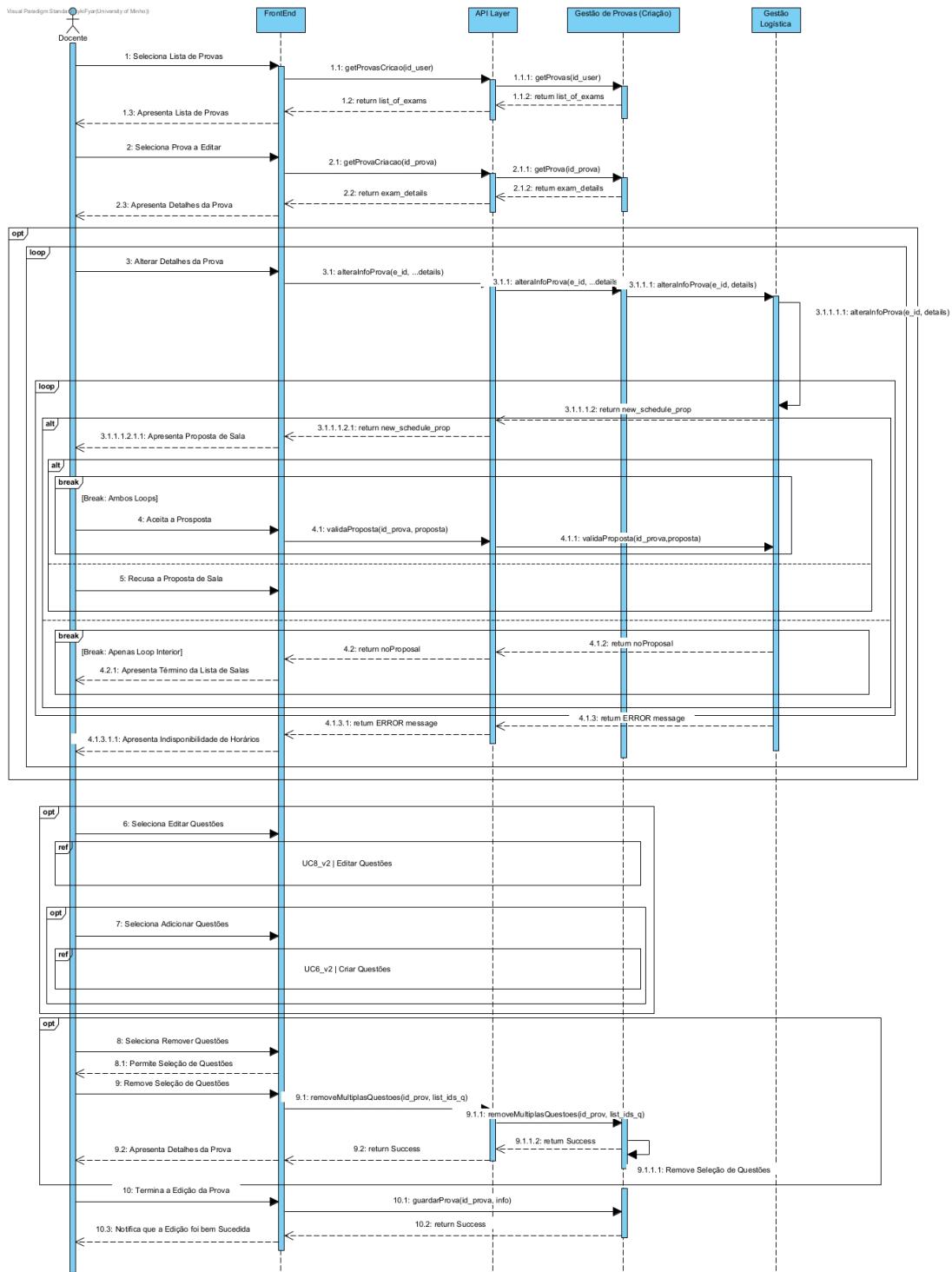


Figura 24: 2^a iteração do Diagrama de Sequência para Editar Prova.

Editar Questões - *Use Case 8*

O Diagrama de Sequência de Editar Questões determina todos os tipos de questão possíveis: Escolha Múltipla, Verdadeiro ou Falso, Desenvolvimento e Completar Espaços.

A única alteração entre as versões do diagrama foram os nomes dados aos Microsserviços.

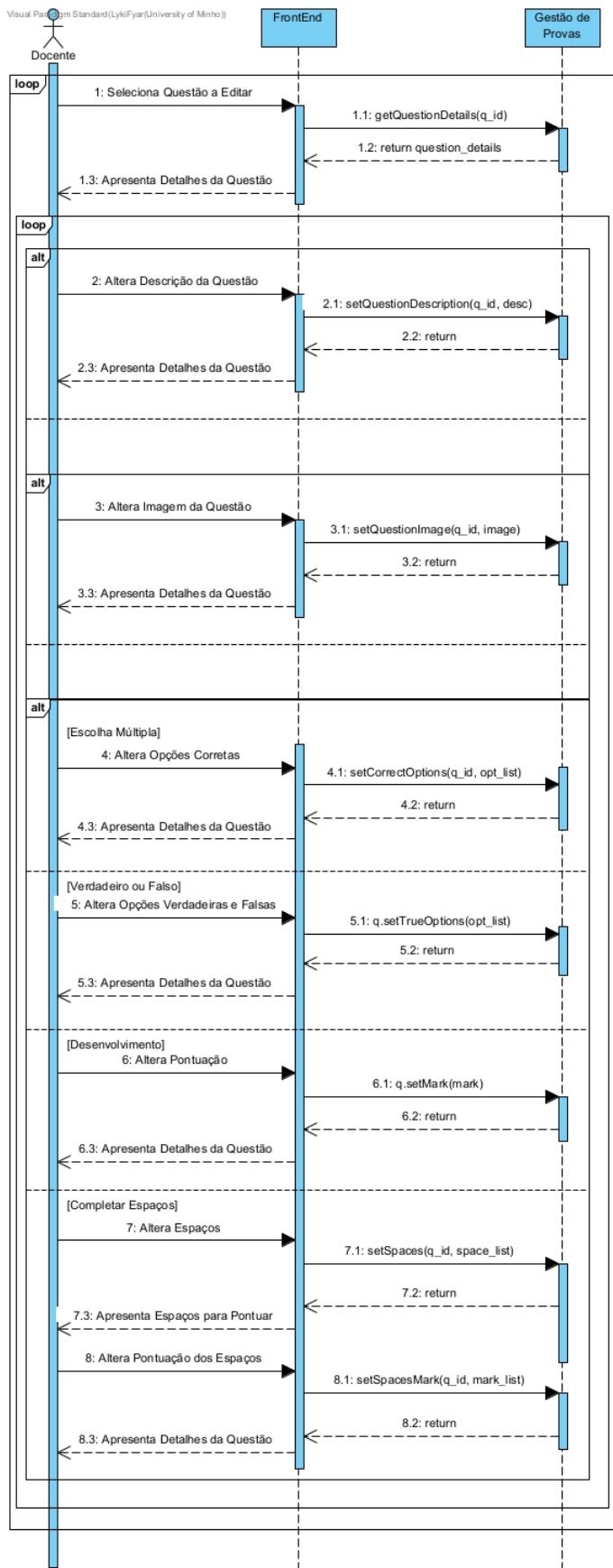


Figura 25: 1^a iteração do Diagrama de Sequência para Editar Questões.

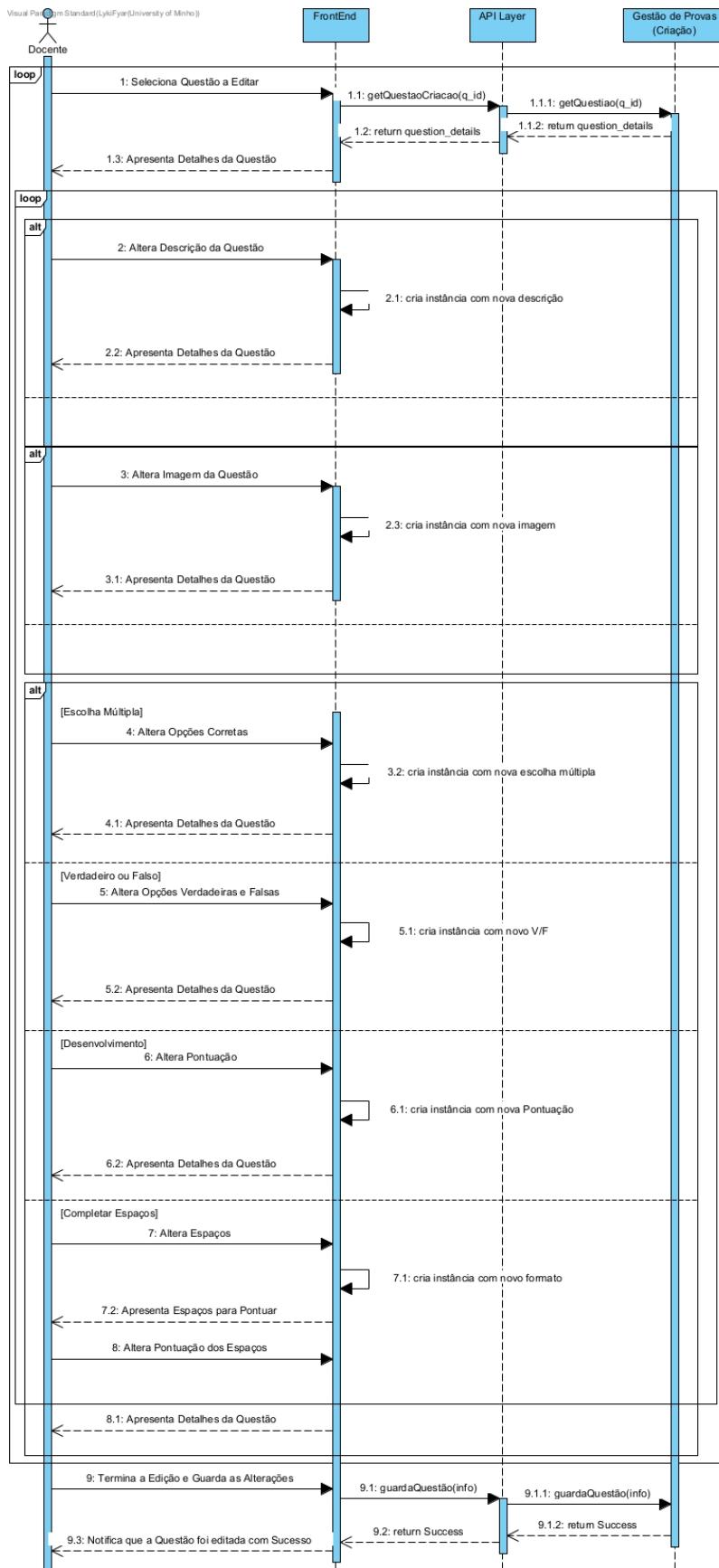


Figura 26: 2^a iteração do Diagrama de Sequência para Editar Questões.

Consultar Detalhes da Prova - Use Case 9

Este Use Case representa a consulta dos detalhes de uma prova que não estão relacionados com a consulta da sua correção, mas sim daqueles necessários para um aluno conhecer antecipadamente a sala, horário e duração da prova. Considerou-se irrelevante o acesso de um aluno aos outros alunos que estão presentes no mesmo teste, então esse detalhe do use case foi ignorado. O FrontEnd presente neste diagrama poderia vir a ser incorporado com o FrontEnd presente na Consulta de Prova (dos seus resultados).

A única alteração entre as versões do diagrama foram os nomes dados aos microsserviços.

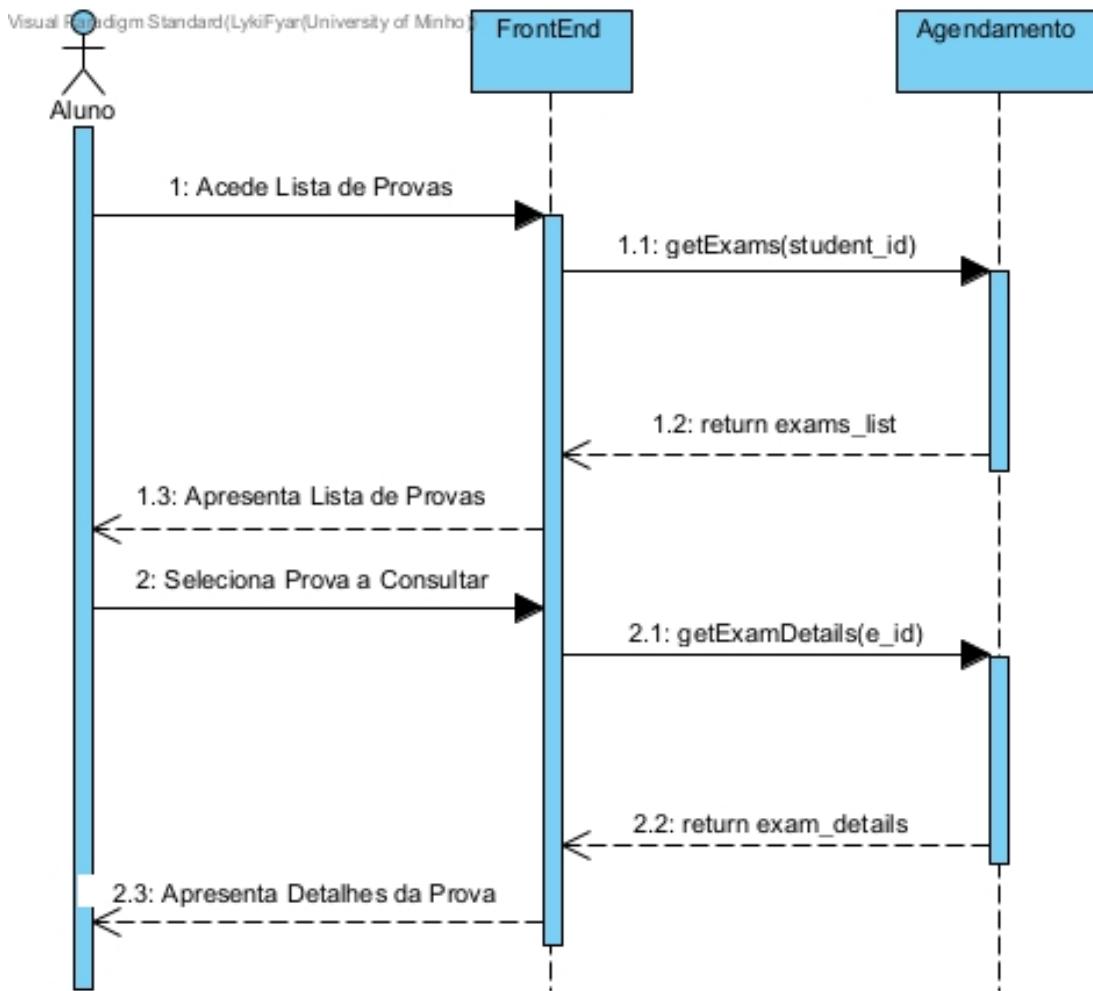


Figura 27: 1^a iteração do Diagrama de Sequência para Consultar Detalhes da Prova.

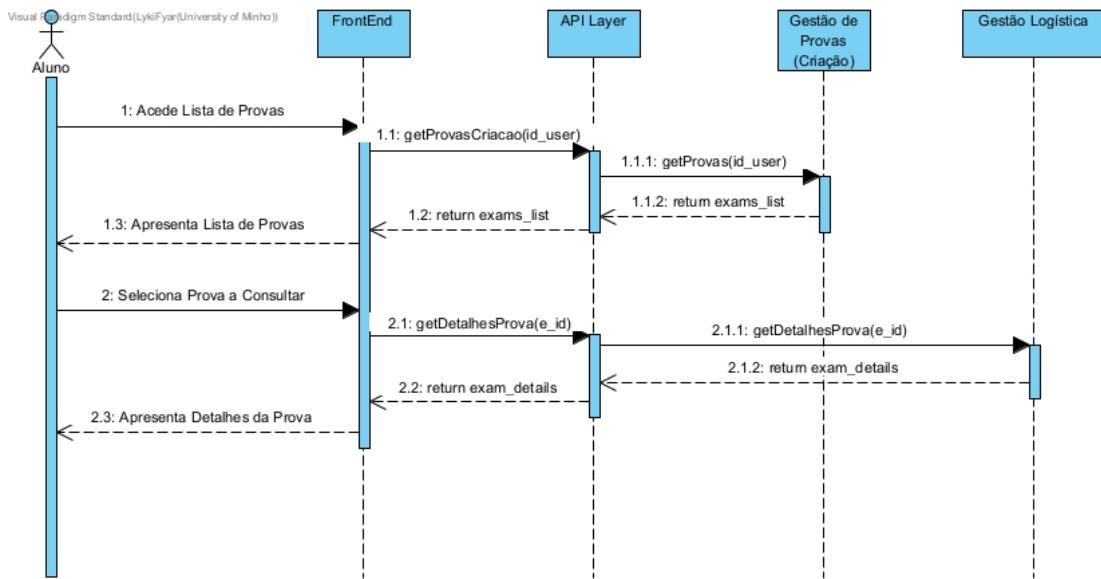


Figura 28: 2^a iteração do Diagrama de Sequência para Consultar Detalhes da Prova.

Partilhar Prova - *Use Case 10*

Este diagrama de sequência determina a interação de um docente quando pretende partilhar acesso a uma prova que criou com outro docente. Isto é feito através dos emails dos docentes específicos de cada um desses docentes.

As diferenças existentes entre as iterações baseiam-se apenas na mudança dos nomes dos microsserviços.

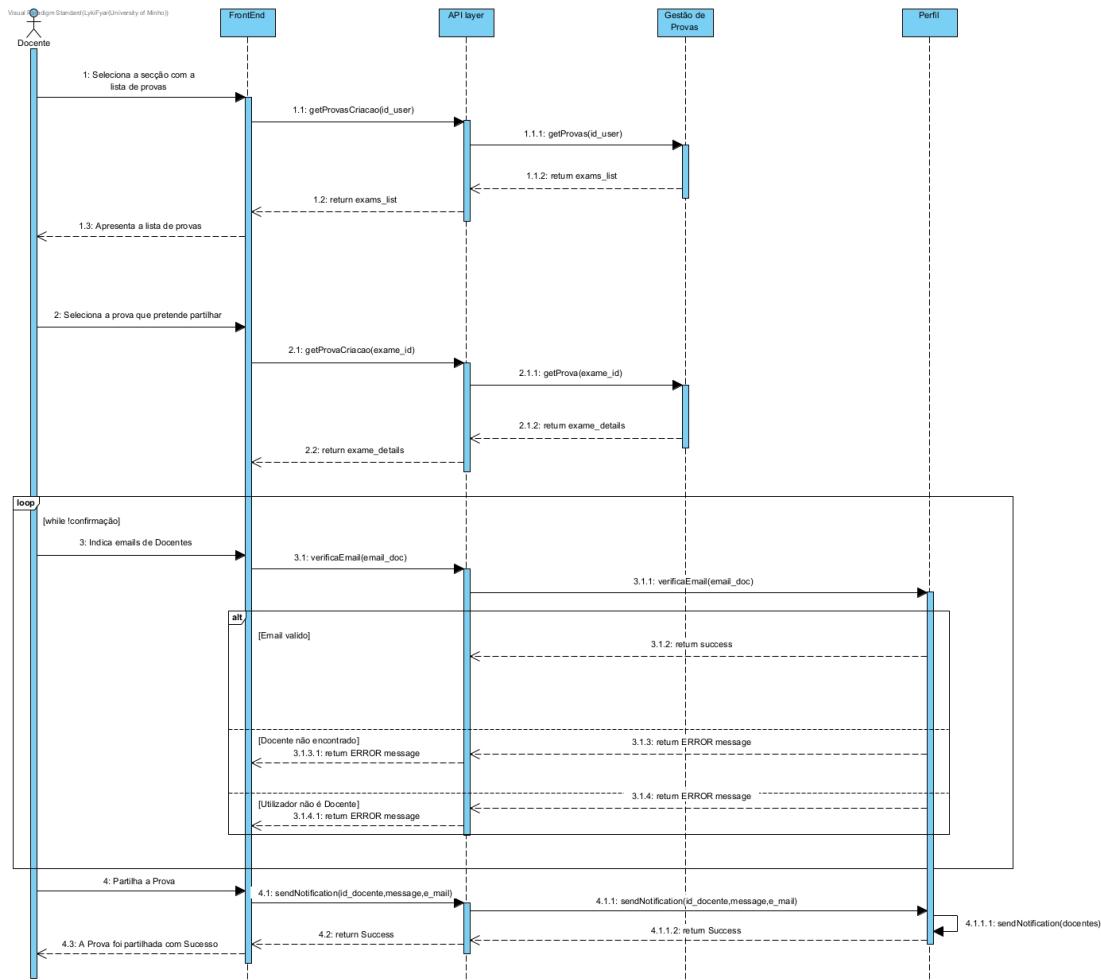


Figura 29: 1^a iteração do Diagrama de Sequência para Partilhar Prova.

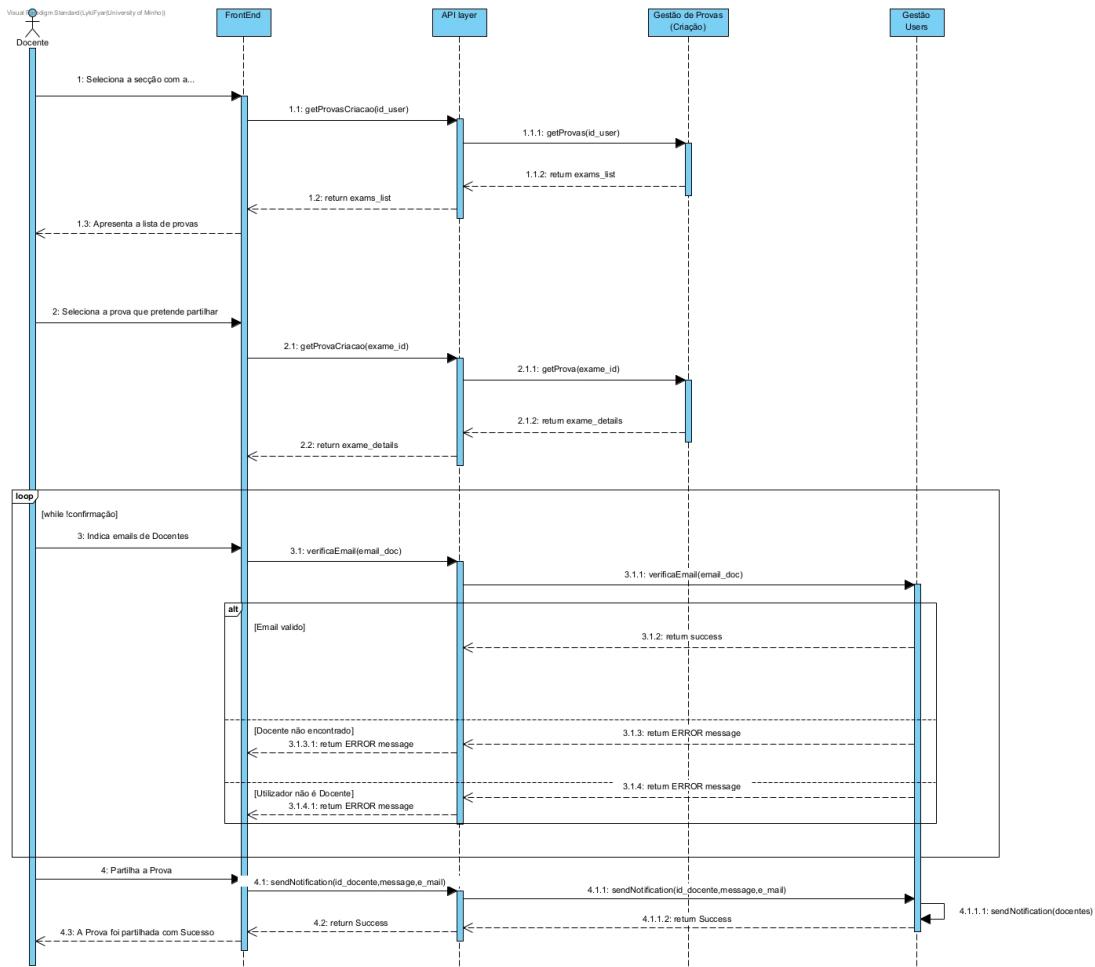


Figura 30: 2^a iteração do Diagrama de Sequência para a Partilhar Prova.

Responder Prova - Use Case 11

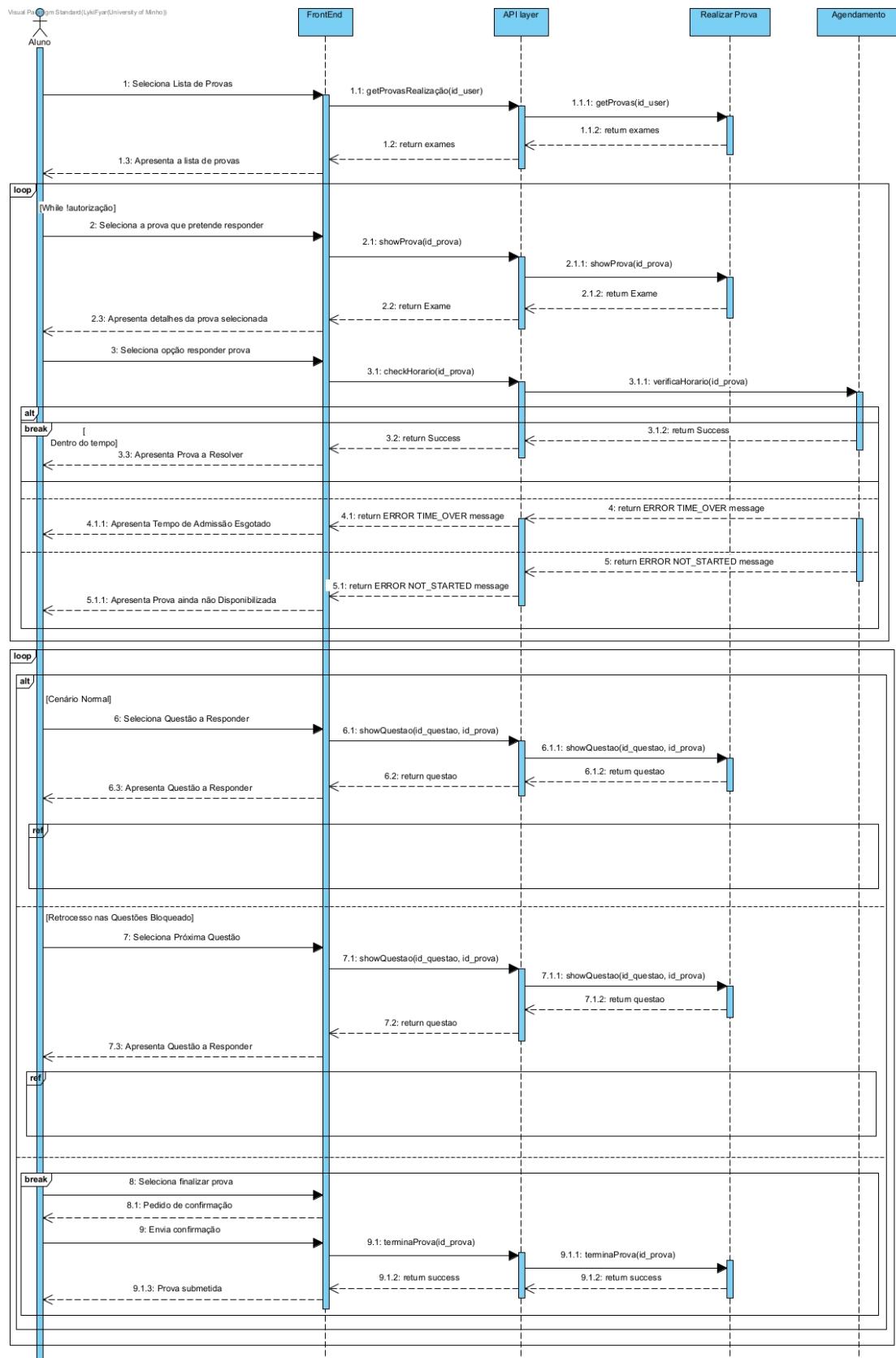


Figura 31: 1^a iteração do Diagrama de Sequência para Responder Prova.

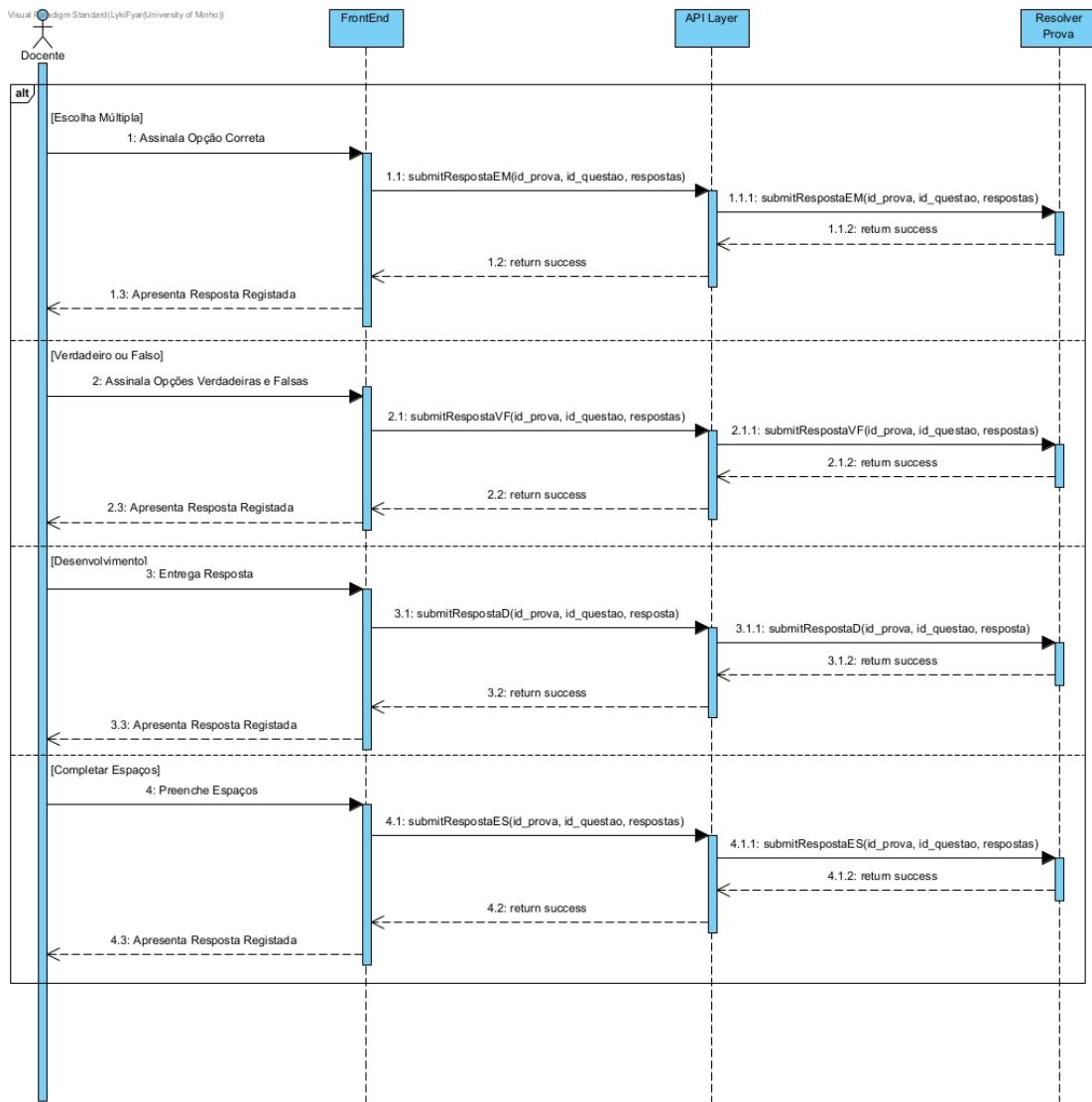


Figura 32: 1^a iteração do Diagrama de Sequência para Responder Prova - continuação.

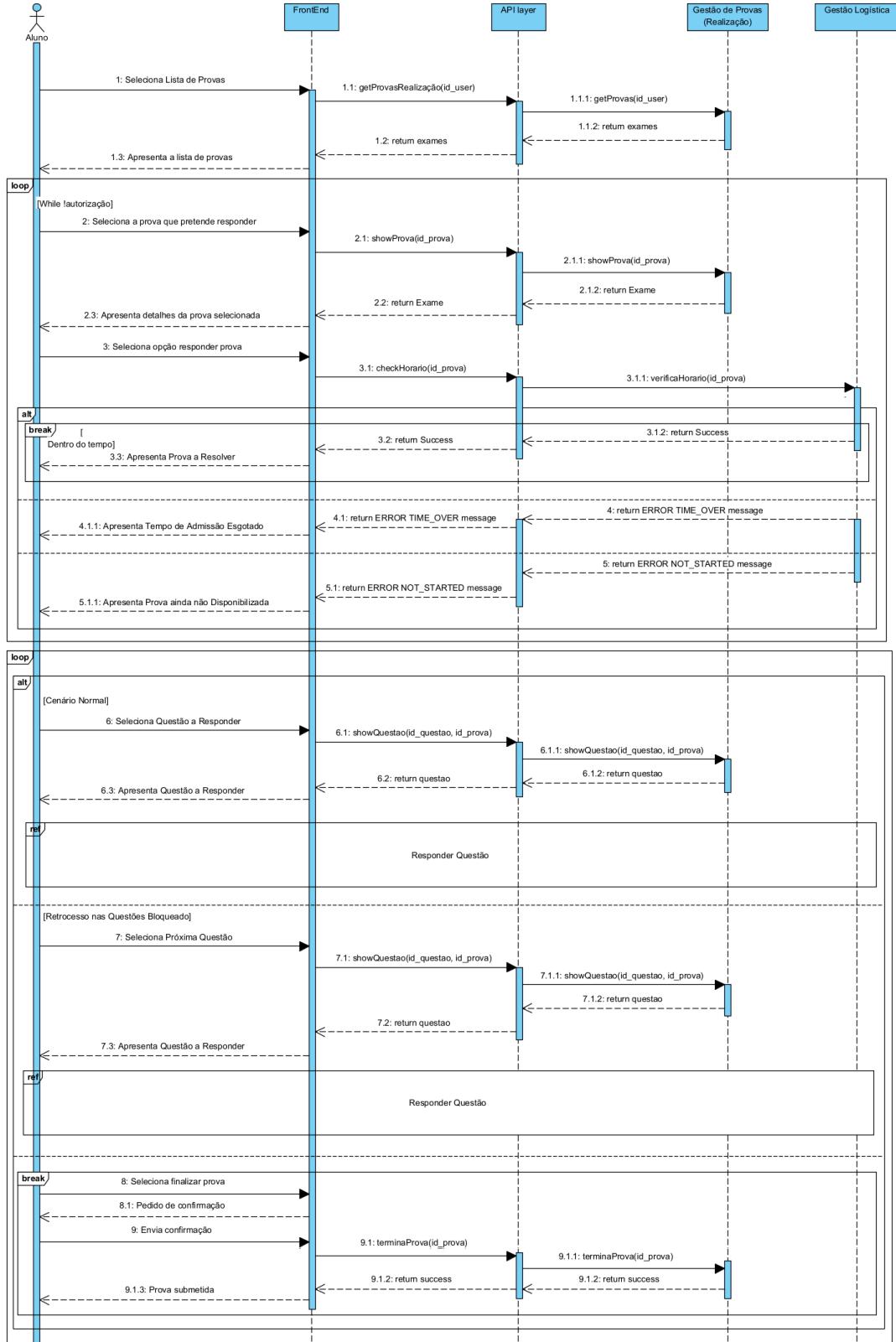


Figura 33: 2^a iteração do Diagrama de Sequência para a Responder Prova.

Este diagrama de sequência determina a interação de um aluno com o sistema quando responde a provas, nomeadamente quando tem ou não a possibilidade de escolher a questão a que vai responder. A sua resposta fica registada no microsserviço de Resolução, necessitando de comunicar com a Gestão Logística para saber se a Prova está a decorrer ou não.

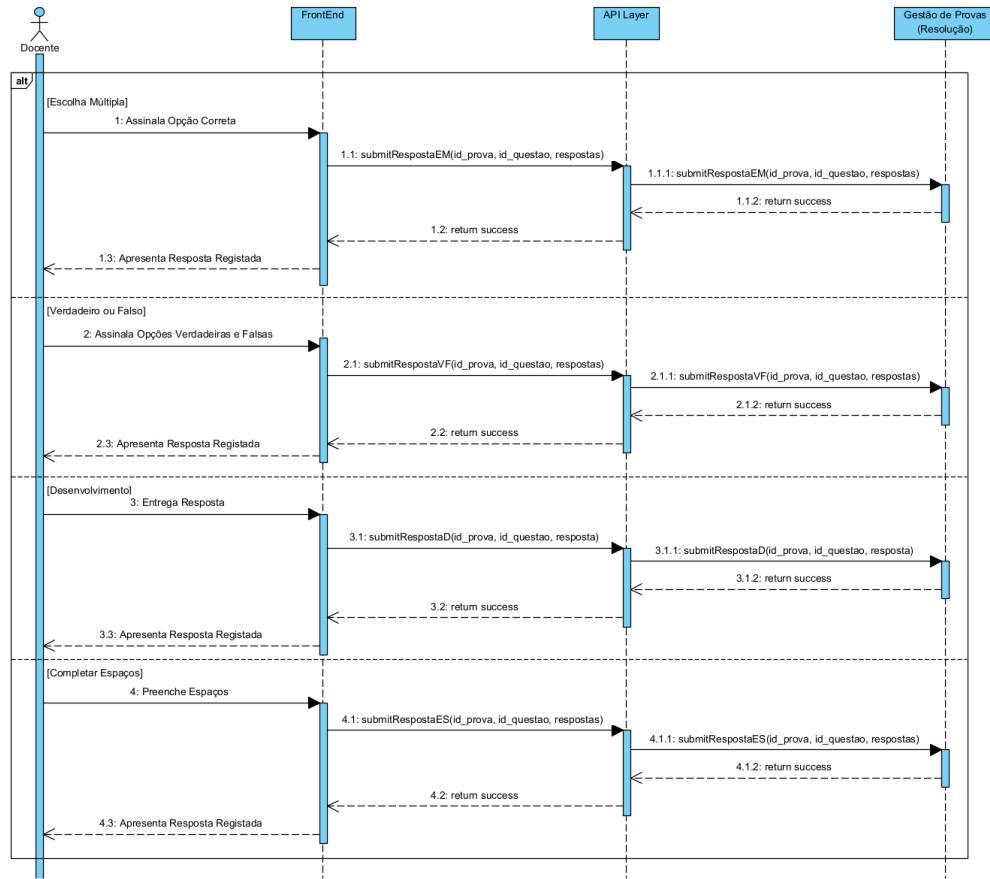


Figura 34: Diagrama de Sequência para Responder a Questão.

Classificar Respostas - Use Case 12

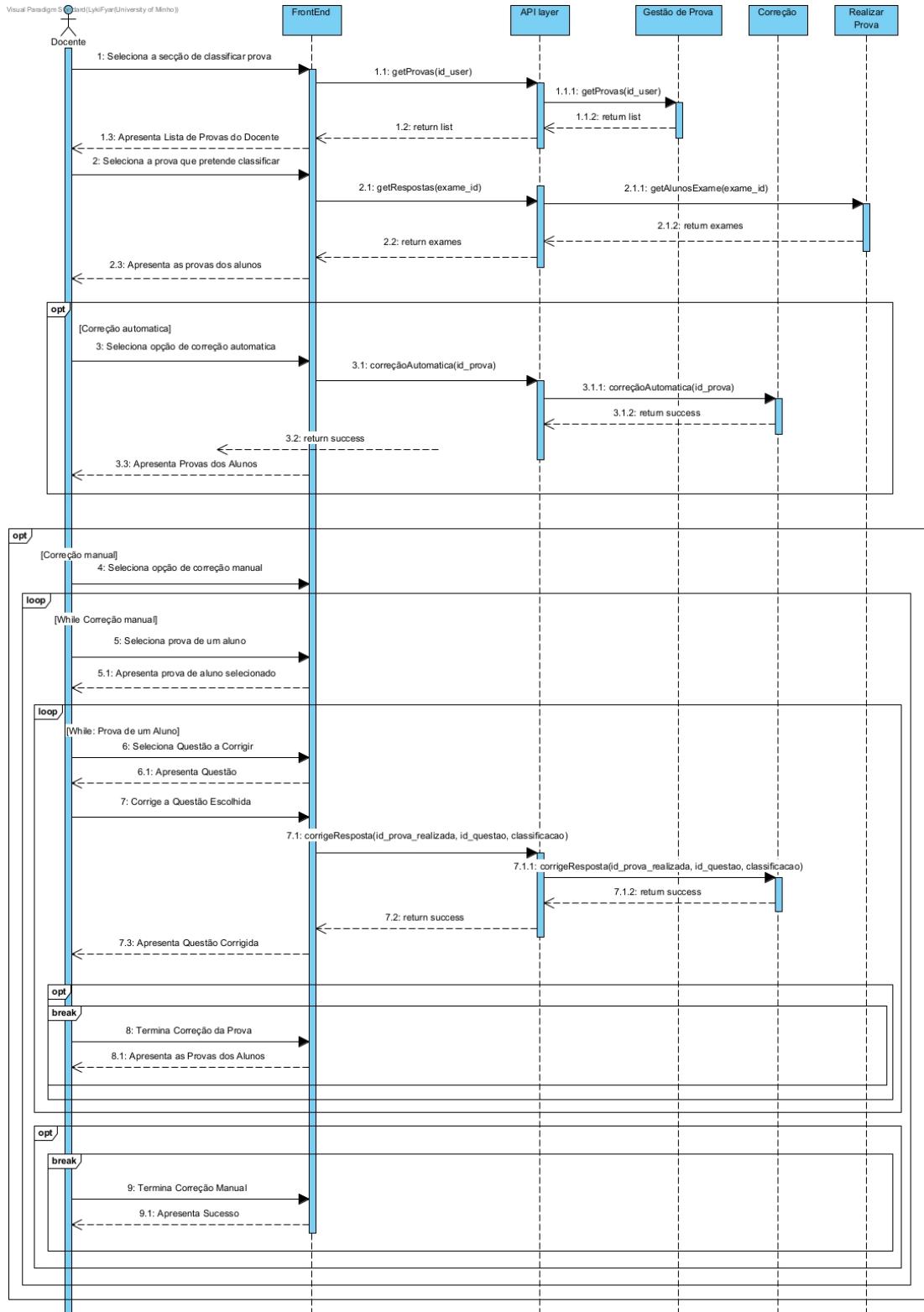


Figura 35: 1^a iteração do Diagrama de Sequência para Classificar Respostas.

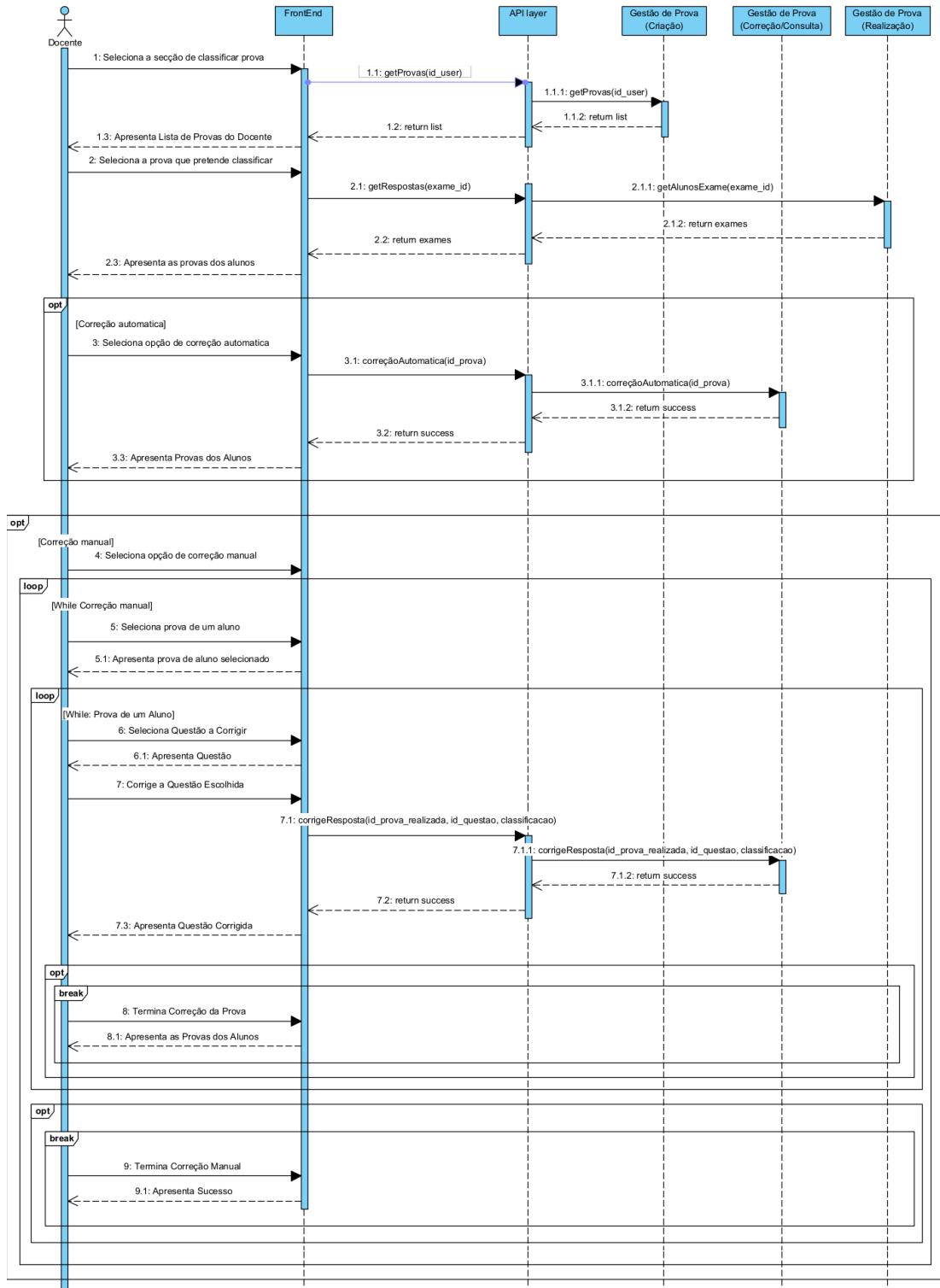


Figura 36: 2^a iteração do Diagrama de Sequência para Classificar Respostas.

Este diagrama é aquele que mais comunicação gera, ao precisar de registar todas as Provas de um dado Docente da Criação de Provas, de recolher as Respostas dos alunos a essa Prova, da Realização de Provas, e de registrar a sua correção na Correção/Consulta. Note-se a utilização

junto do utilizador de todas as respostas, depois destas serem recuperadas da base de dados desse microsserviço, para maximizar a eficiência no uso da rede.

As diferenças entre a primeira e a segunda iteração residem na reestruturação dos microsserviços, onde os microserviços "Correção" e "Consulta" foram consolidados no "Gestão de Provas (Correção/Consulta)", enquanto que os outros foram renomeados.

Publicar Classificações da Prova - Use Case 13

Este diagrama representa o caso de uso em que um docente disponibiliza as classificações para os alunos associados a uma prova específica. Essa ação envolve o docente acessar uma lista de provas, selecionando uma prova específica e tornando-a pública, o que significa comunicar com o microserviço "Gestão de Provas (Correção/Consulta)". Em seguida, o sistema notifica todos os alunos, estabelecendo comunicação com o microserviço "Gestão de Usuários".

As diferenças entre a primeira e a segunda iteração residem na reestruturação dos microsserviços, onde os microserviços "Correção" e "Consulta" foram consolidados no "Gestão de Provas (Correção/Consulta)" e o "Autenticação" foi integrado no "Gestão de Users".

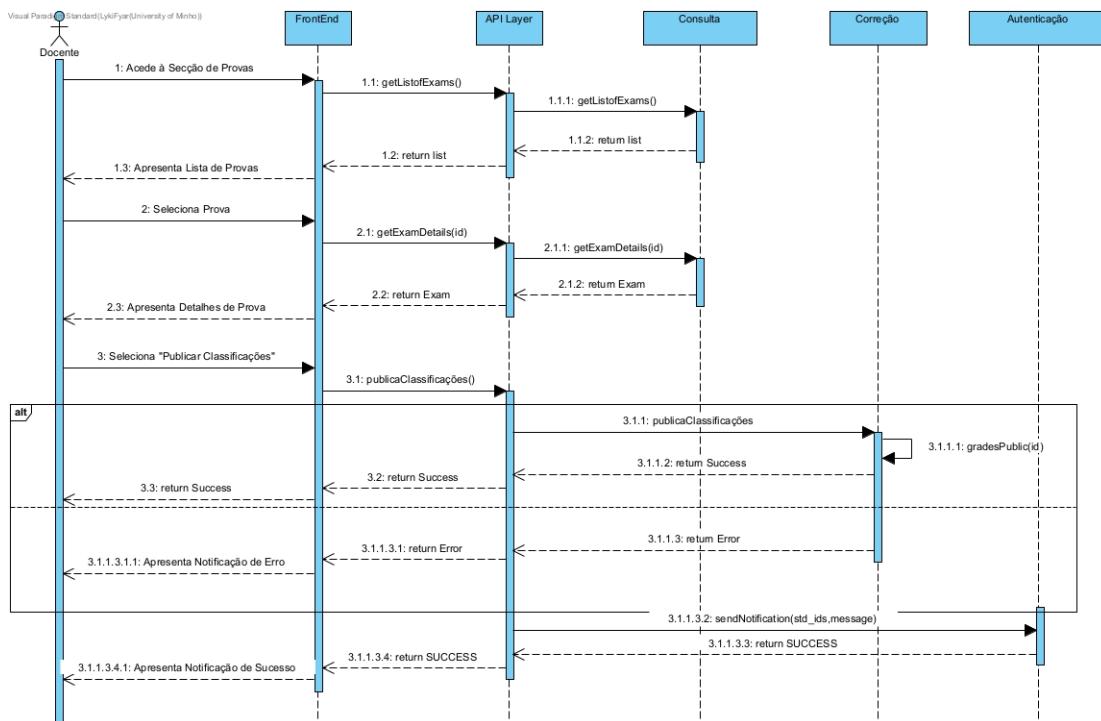


Figura 37: 1^a iteração do Diagrama de Sequência para Publicar Classificações da Prova.

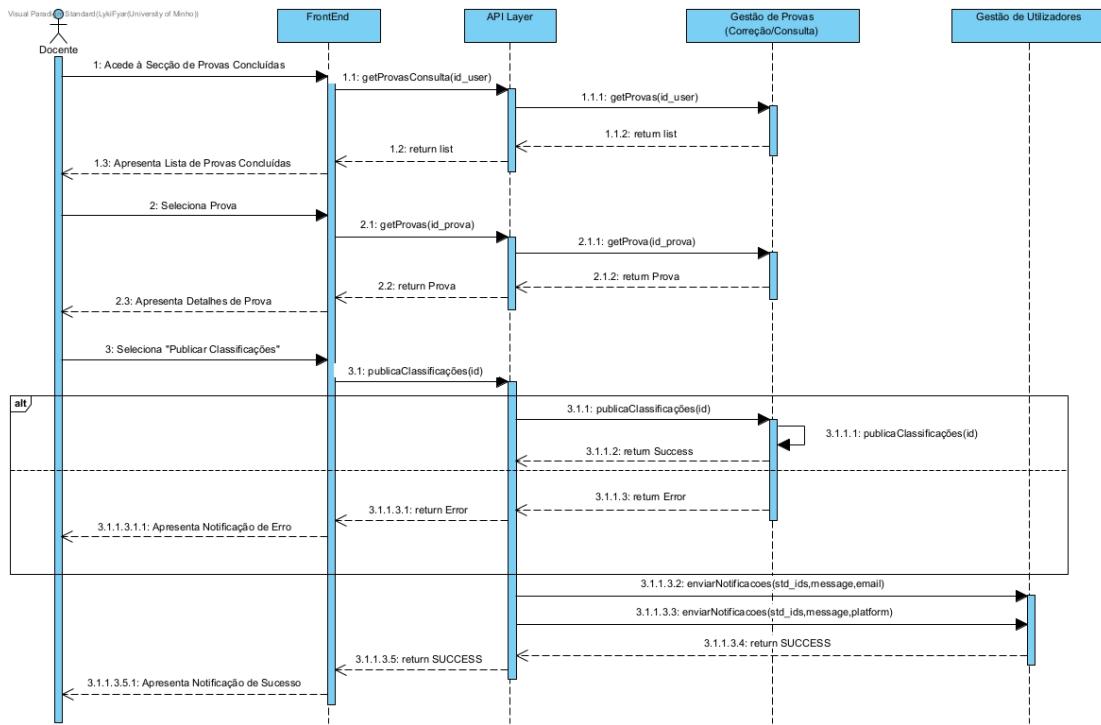


Figura 38: 2^a iteração do Diagrama de Sequência para Publicar Classificações da Prova.

Consultar Prova Corrigida - *Use Case 14*

Este diagrama representa o use case de um aluno consultar a correção de uma prova. Esta ação implica o aluno aceder às suas provas já concluídas e publicadas, e consultar os detalhes de uma prova - que lhe fornece toda a correção com as questões, a sua resposta e a classificação, deste modo, apenas é necessária interação com o microsserviço "Gestão de Provas (Consulta/Correção)"

As diferenças entre a primeira e a segunda iteração residem na reestruturação dos microsserviços, onde o microsserviço "Consulta" foi consolidado no "Gestão de Provas (Correção/Consulta)".

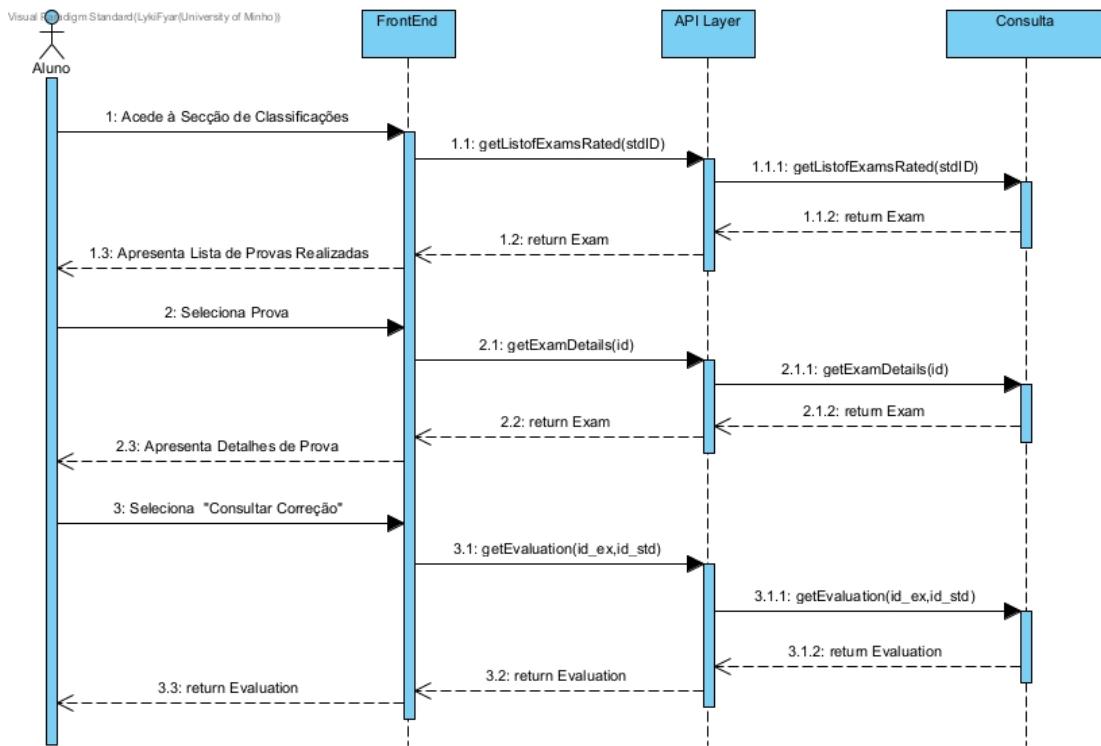


Figura 39: 1^a iteração do Diagrama de Sequência para Consultar Prova Corrigida.

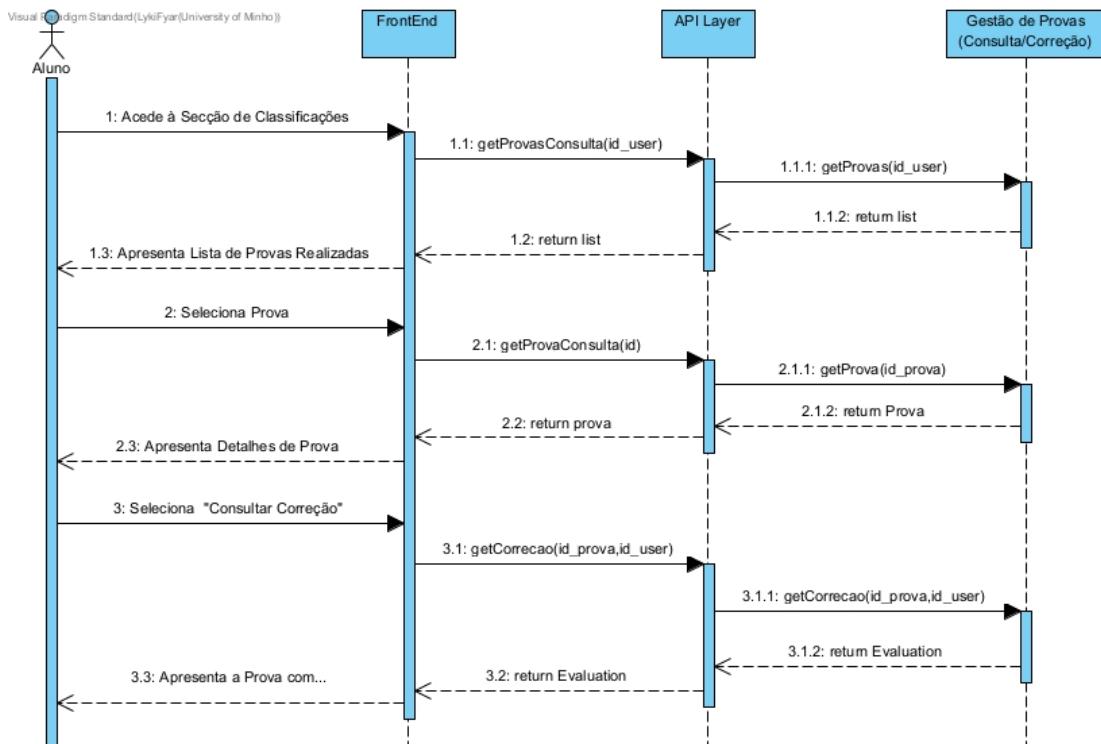


Figura 40: 2^a iteração do Diagrama de Sequência para Consultar Prova Corrigida

Gerir Salas - Use Case 15

Este diagrama ilustra o caso de uso em que um técnico realiza a gestão de salas, tendo a capacidade de adicionar novas salas ou eliminar conjuntos específicos. No caso da adição, o técnico submete um arquivo, e dentro do microsserviço "Gestão Logística", a validade do formato e conteúdo é confirmada. Em caso de remoção, o técnico acessa uma lista de salas, seleciona aquelas que deseja remover e confirma a ação, utilizando novamente o microsserviço "Gestão Logística". Se houver salas associadas a testes que serão removidos, o sistema envia notificações a todos os docentes associados, utilizando o microsserviço "Gestão de Users".

As diferenças entre a primeira e a segunda iteração residem na reestruturação dos microsserviços, onde os microsserviços Agendamento e Autenticação foram integrados nos microsserviços "Gestão Logística" e "Gestão de Users", respectivamente.

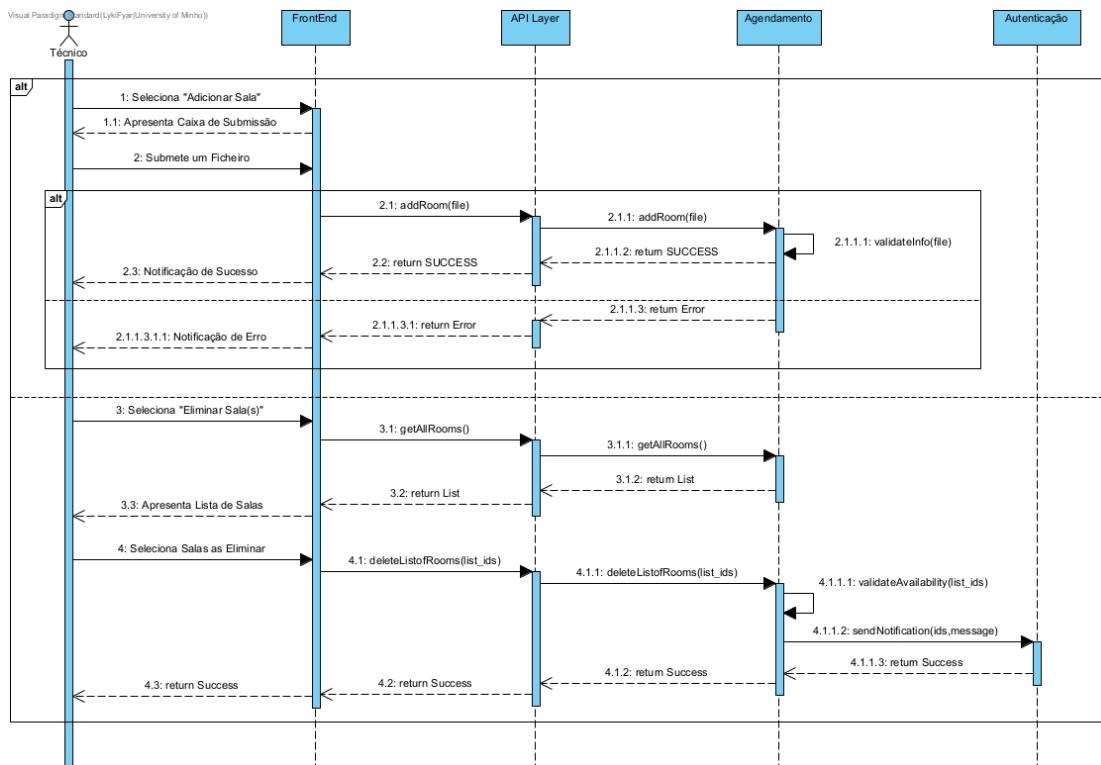


Figura 41: 1^a iteração do Diagrama de Sequência para Gerir Salas.

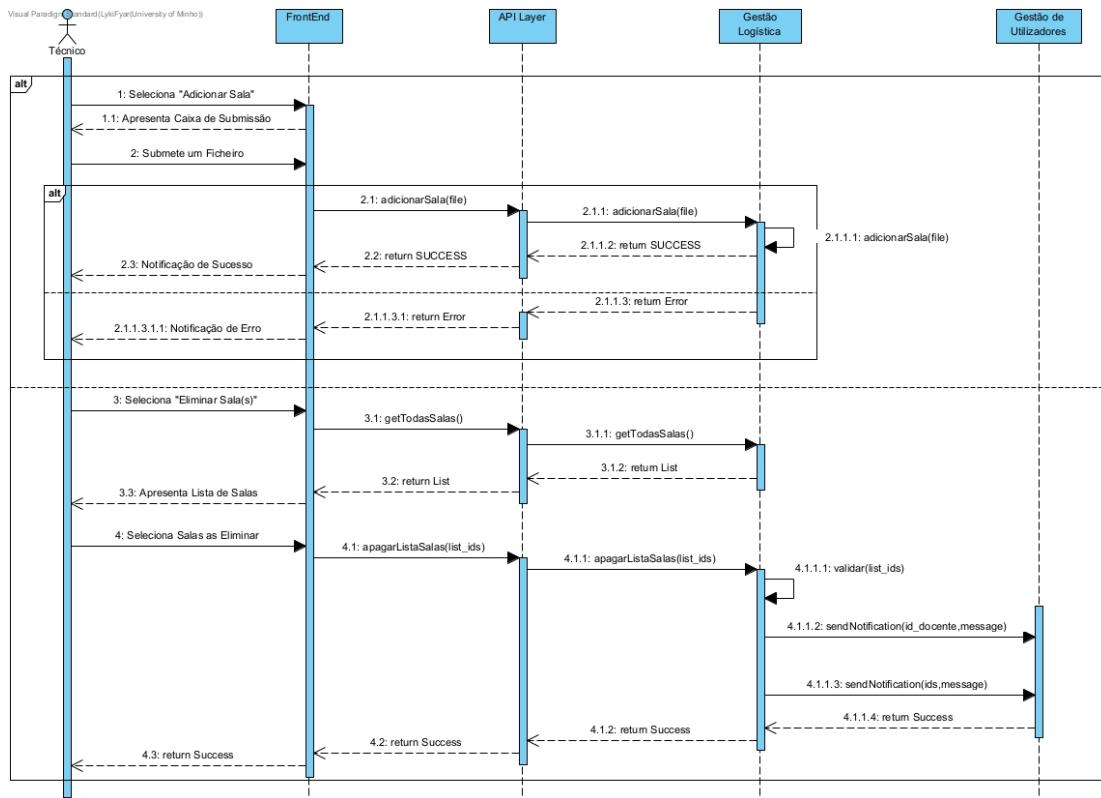


Figura 42: 2^a iteração do Diagrama de Sequência para Gerir Salas.

Aceder às Notificações - *Use Case 16*

Este é o diagrama de sequência que representa a apresentação e a leitura de notificações por parte dos Alunos e dos Docentes. O diagrama é muito simples, visto que o sistema indica ao utilizador que o mesmo possui notificações por visualizar, e depois apresenta o processo de mensagens para obter a lista de notificações de um determinado utilizador. As alterações entre a primeira e a segunda versão é apenas na alteração do nome do microsserviço a aceder (Perfil → Gestão de Users)

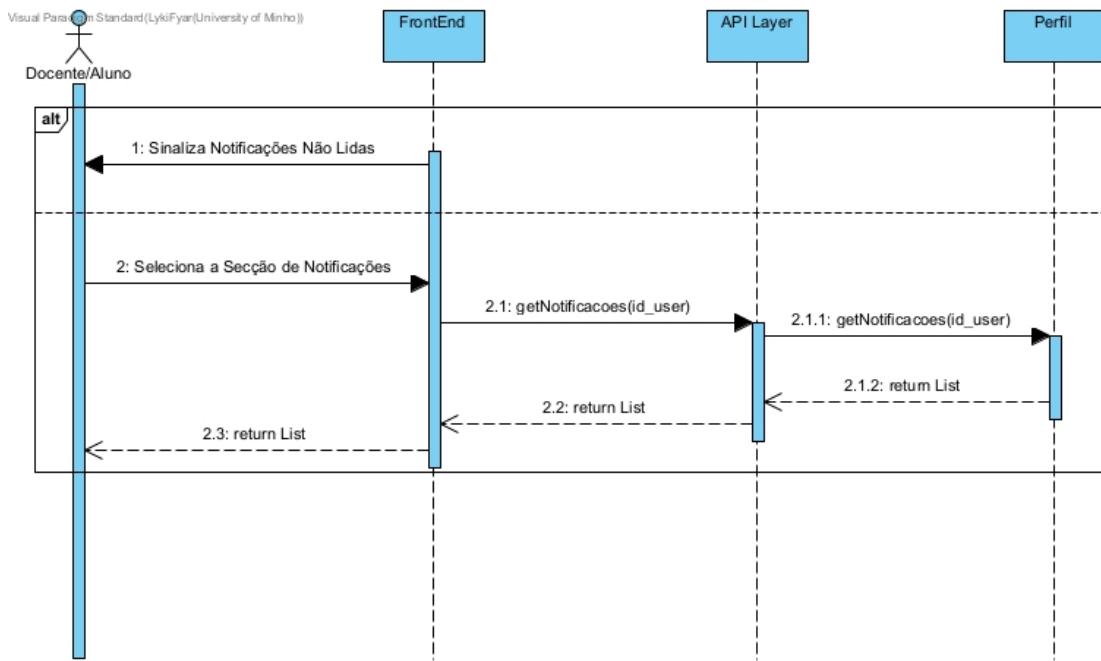


Figura 43: 1^a iteração do Diagrama de Sequência para Aceder às Notificações.

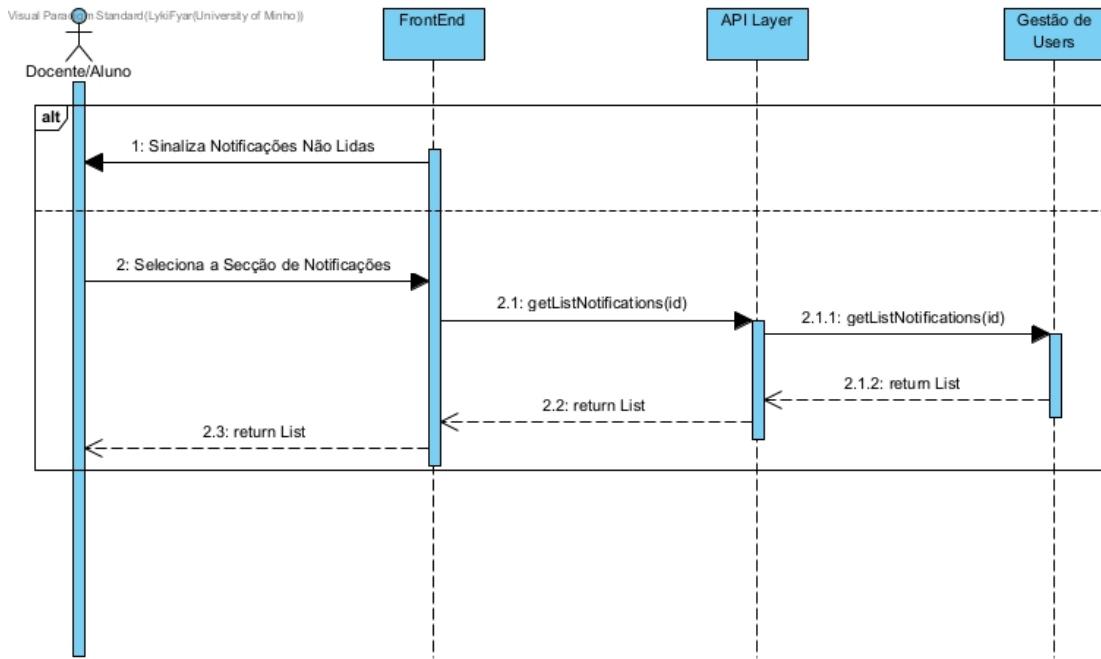


Figura 44: 2^a iteração do Diagrama de Sequência para Aceder às Notificações.

Visualização do Deployment

A **Deployment View** serve como ponte entre os blocos de software evidenciados na *Building block View* e a infraestrutura que o nosso sistema irá correr. Esta visualização permite entender o impacto da infraestrutura na nosso sistema e perceber as limitações que a infraestrutura impõe sobre o nosso sistema.

Como não temos informação detalhada sobre a infraestrutura da Instituição de Ensino Superior que o **PROBUM** irá atuar, decidimos abstrair conceitos específicos de Hardware como PU's ou RAM, assumindo sempre que o nosso sistema deverá ser suportado por computadores de gama baixa segundo o RNF12 (requisito não funcional 12) do documento de Requisitos.

Decidimos apresentar dois cenários: Um de Teste e outro Real.

O cenário de teste serve para ilustrar o funcionamento básico do **PROBUM** num computador, dessa forma, todos os componentes são descritos em containers separados e comunicam pela rede docker, também serve como cenário de desenvolvimento.

O cenário real descreve o cenário que o **PROBUM** deverá atuar após o seu lançamento.

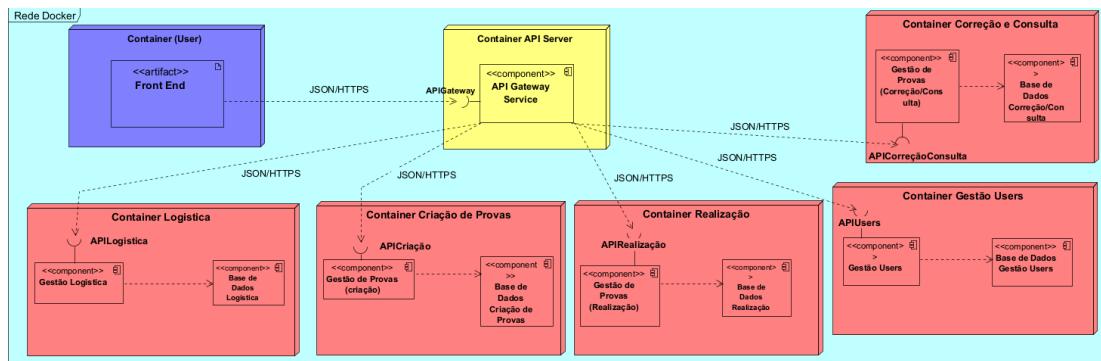


Figura 45: Diagrama de deployment no Cenário de Teste

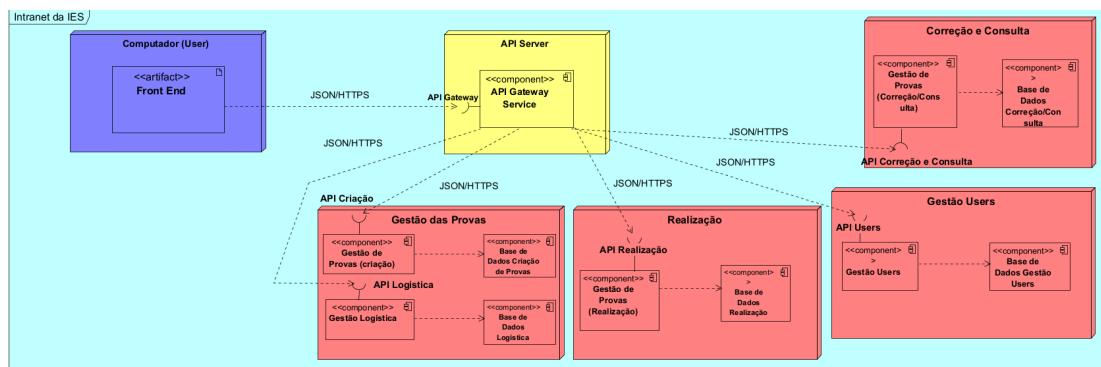


Figura 46: Diagrama de deployment no Cenário Real

Como não temos informações sobre a infraestrutura da IES que traz como consequência não sabermos o número de nodos e como os microsserviços Gestão de Provas(criação) e Gestão Logística comunicam em contextos temporais próximos, para evitar congestionamento na intranet da IES, colocamos ambos no mesmo Nodo.

Sobre o deployment, ainda queríamos acrescer que uma vez não se trata do fim do desenvolvimento da aplicação, analisar futuramente mudanças na infraestrutura da universidade e/ou problemas com *Load Balancing* em microsserviços, o número de replicas e a divisão por Nodos poderão ser decisões mutáveis.

Modelos Lógicos de Bases de Dados

Nesta fase da definição da arquitetura do sistema, surgiu a necessidade de caracterização das diferentes entidades de cada microserviço assim como o aprofundamento das suas relações. Este capítulo revela os elementos fundamentais não apenas para uma boa estruturação dos dados, mas também para uma melhor eficiência no armazenamento dos mesmos.

1. Gestão de Utilizadores:

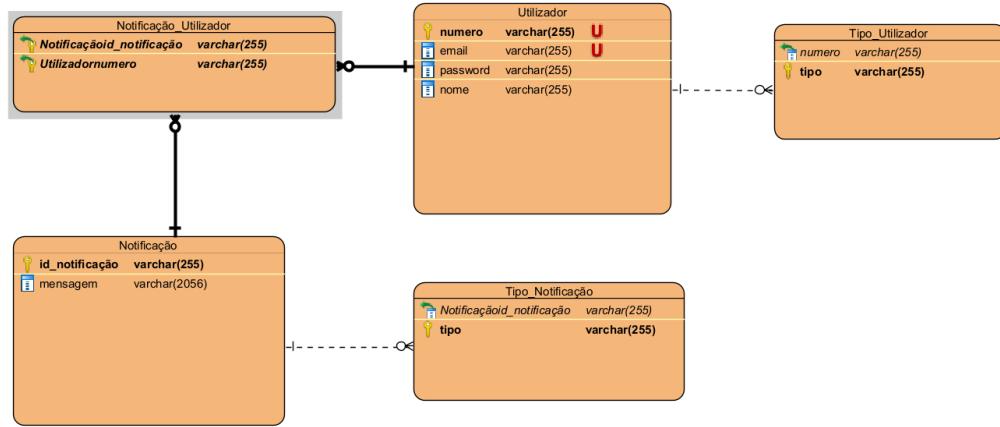


Figura 47: Modelo Lógico da Base de Dados da Gestão de Utilizadores.

2. Gestão de Prova (Criação):

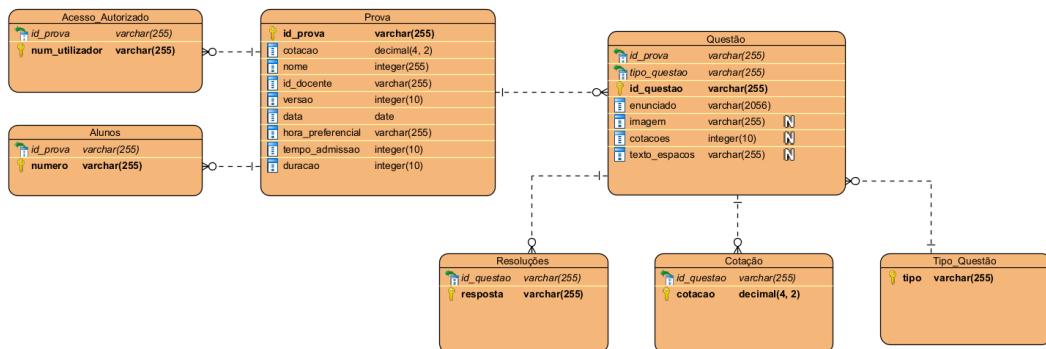


Figura 48: Modelo Lógico da Base de Dados da Gestão de Provas (Criação).

3. Gestão Logística:

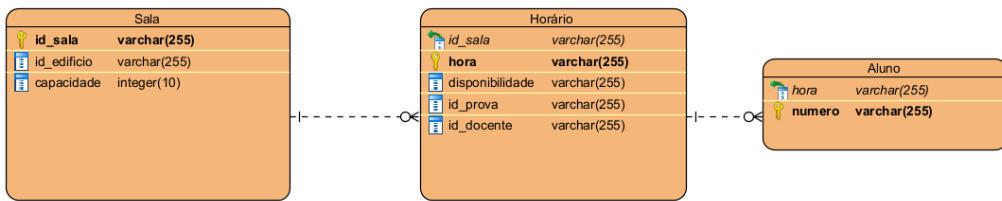


Figura 49: Modelo Lógico da Base de Dados da Gestão Logística.

4. Gestão de Provas (Realização):

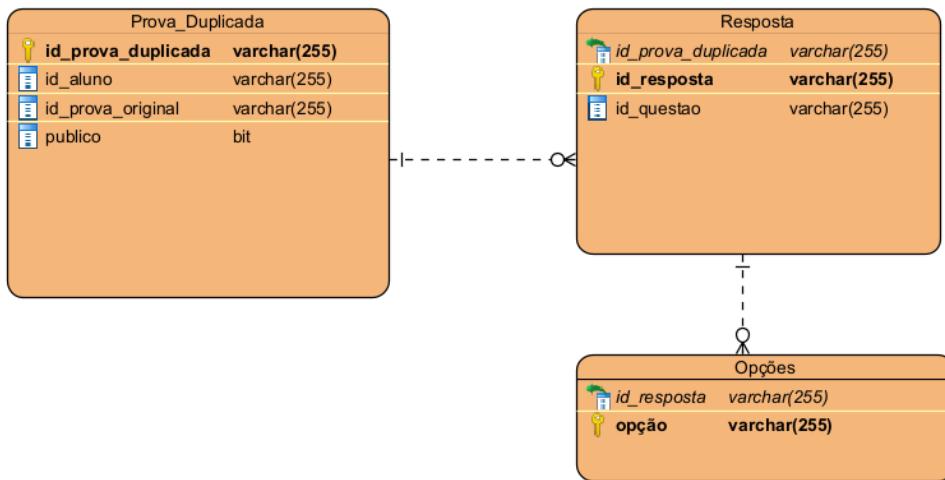


Figura 50: Modelo Lógico da Base de Dados da Gestão de Provas (Realização).

5. Gestão de Provas (Correção e Consulta):

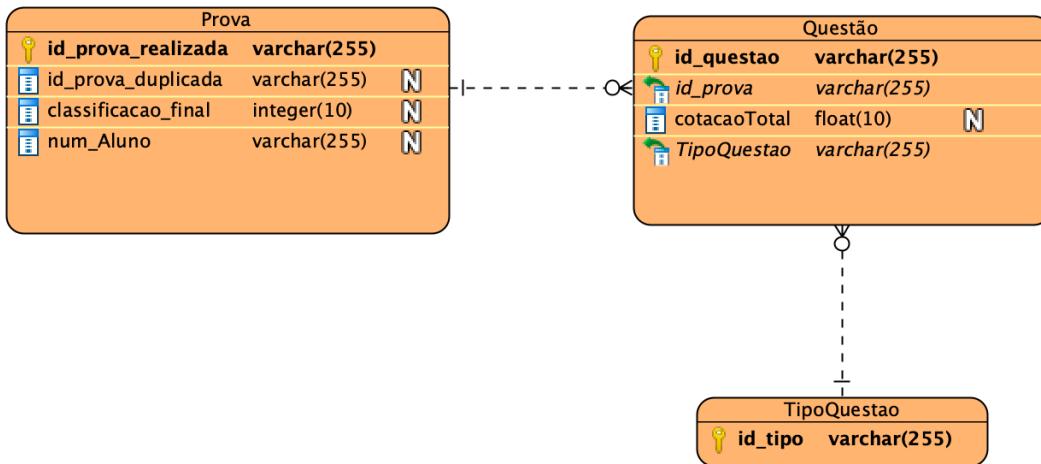


Figura 51: Modelo Lógico da Base de Dados da Gestão de Provas (Realização).

Documentação API

Concluindo esta etapa e entrando na esfera da implementação, decidimos elaborar documentações minuciosas para a API de cada microsserviço. Isto proporcionará um ponto de partida claro e fundamentado para os desenvolvedores, estabelecendo as operações HTTP essenciais de POST, GET, PUT e DELETE que cada microsserviço deve incorporar. Tendo em conta os microsserviços delineados anteriormente **Gestão de utilizadores**, **Gestão de Provas (Criação)**, **Gestão de Provas (Agendamento)**, **Gestão de Provas(Realização)** e **Gestão de Provas(Correção e Consulta)**, documentamos as possíveis ações a realizar em cada um deles:

1. Gestão de utilizadores:

Método	Rota	Descrição
GET	/api/usr/profile/<IDU>	Perfil de um utilizador do Sistema
GET	/api/usr/notificação/<IDU>	Notificações de um utilizador
GET	/api/usr/login	Página de login
POST	/api/usr/login	POST do formulário de Login
PUT	/api/usr/change_pass/<IDU>	Altera a palavra passe de um user
GET	/api/usr/register	Página de registo de utilizador
POST	/api/usr/register	POST do formulário de Registo
POST	/api/usr/logout	Encerra a sessão de um utilizador
GET	/api/usr/profile/edit/<IDU>	Página de edição de perfil
PUT	/api/usr/profile/edit/<IDU>	Atualiza as informações de um utilizador

2. Gestão de Prova(Criação):

Método	Rota	Descrição
GET	/api/gestao/mainpage/<IDU>	Dá a página principal deste microserviço
GET	/api/gestao/getprovas/<IDU>	Provas que o docente tem acesso
GET	/api/gestao/getprovas/<IDU>/prova?id=<IDP>	GET de uma prova específica
GET	/api/gestao/create/prova?id=<IDP>	Página de criação de prova
POST	/api/gestao/create/prova?id=<IDP>	Cria uma prova
PUT	/api/gestao/create/prova?id=<IDP>	Guarda a prova que ainda está a desenvolver na BD
DELETE	/api/gestao/prova?id=<IDP>	Elimina uma prova
POST	/api/gestao/agendaprova?id=<IDP>	Prova duplicada e enviada para o microserviço da realização 4

3. Gestão Logística:

Método	Rota	Descrição
GET	/api/mark/mainpage	Página principal deste microserviço
GET	/api/mark/rooms	Dá as salas disponíveis e as informações das mesmas
GET	/api/mark/room?id=<ID/timetable>	Dá os horários disponíveis para uma sala
GET	/api/mark/createROOM	Formulário de criação de uma sala (técnico)
POST	/api/mark/createROOM	Criação de uma sala para a base de dados (técnico)
DELETE	/api/mark/room?id=<ID>	Elimina uma sala (técnico)

4. Gestão de Provas (Realização):

Método	Rota	Descrição
GET	/api/evaluation/mainpage	Dá a página principal deste microserviço
GET	/api/evaluation/prova?id=<ID>	Página de realização de uma prova
PUT	/api/evaluation/prova?id=<ID>	Guarda as alterações ao longo da prova
POST	/api/evaluation/correct?id=<ID>	Manda a prova para o microserviço de correção e consulta prova
POST	/api/evaluation/save/<PROVA>	Guarda uma prova na sua base de dados

5. Gestão de Provas (Correção e Consulta):

Método	Rota	Descrição
GET	/api/correct/mainpage/<IDU>	Dá a página principal deste microserviço
GET	/api/correct/listaprova/<IDU>	Recebe uma lista de todos os IDs de prova
PUT	/api/correct/corrAUToprovas?id=<IDP>,<IDP>,...	Corrigir automaticamente uma prova
PUT	/api/correct/corriMANUALprovas?id=<IDP>	Correção de uma prova manualmente
PUT	/api/correct/prova?id=<IDP>	Update da prova com a correção de um professor
GET	/api/see/mainpage/<IDU>	Página principal deste microserviço
GET	/api/see/listprovas/<IDU>/ready	Lista de provas que estão corrigidas
GET	/api/see/prova?id=<IDP>	Consulta de uma prova específica
POST	/api/see/prova?id=<IDP>/recorrect	Aluno solicita nova correção

Visto que a comunicação entre páginas e microsserviços ainda está em fase de definição, especialmente em relação à transmissão segura de informações do utilizador (IDU) e identificação de provas (IDP). Neste momento, não conseguimos chegar a uma conclusão sobre a abordagem ideal para esta comunicação. Estamos a considerar diferentes opções, como a possibilidade de transmitir informações de forma encriptada através do URL REQUEST, o uso de cookies ou a implementação de tokens que seriam transmitidos tanto pela API quanto pelas páginas. O nosso objetivo é garantir que o utilizador tenha acesso apenas a determinadas páginas e locais de forma segura.

Conclusão

Nesta fase, empenhamo-nos na decomposição funcional do sistema, optando por uma abordagem centrada em microsserviços para garantir flexibilidade e escalabilidade.

A criação de diagramas de sequência proporcionou uma visão detalhada das interações entre os diferentes componentes da arquitetura, garantindo uma compreensão clara do fluxo de dados e das operações executadas em cada fase do processo. A decisão de implementar microsserviços não favorece apenas a modularidade e a manutenção, mas também possibilita a escalabilidade eficientes.

A elaboração de APIs robustas desempenha um papel crucial na integração e na comunicação eficiente entre os diversos microsserviços. Essas interfaces foram desenhadas com precisão para garantir a interoperabilidade dos componentes, promovendo uma arquitetura coesa e orientada a serviços.

Os diagramas de componentes e de contexto, bem como os de classes por microsserviço, oferecem uma visão holística da arquitetura, destacando a interconexão entre os diversos módulos e sua relação com o ambiente externo. Essas representações visuais não só facilitam a compreensão global do sistema, mas também servem como guias valiosos durante o desenvolvimento e a manutenção da plataforma.

Em suma, a solução arquitetural proposta para o ProbUM reflete um planeamento cuidadoso e

uma abordagem sólida, através da planificação da arquitetura, com o objetivo de garantir uma plataforma de realização de testes eficiente, escalável e facilmente mantida. A combinação de decomposição funcional, microserviços, diagramas detalhados e APIs bem definidas estabelece as bases para o sucesso do projeto, proporcionando uma estrutura robusta e adaptável para enfrentar os desafios presentes e futuros.