ISPGAYA
instituto superior politécnico

Licenciatura em Engenharia Informática
Projeto de Engenharia Informática em Contexto Empresarial

3.º Ano

Gonçalo de Lira Pereira

APLICAÇÃO MÓVEL PARA GESTÃO DE ESPAÇOS E EQUIPAMENTOS

SISTEMA DE RESERVAS COM FIREBASE E FLUTTER

Relatório de projeto orientado pelo Professor Doutor José Carlos Miranda Nova Arnaud e apresentada à Escola Superior de Ciência e Tecnologia

Julho de 2025

Dedicatória

Dedico este trabalho à minha família, pelo apoio incondicional em todos os momentos; aos amigos, pela motivação e companheirismo ao longo deste percurso; e aos professores que, com dedicação e saber, contribuíram de forma significativa para a minha formação. A todos, o meu sincero agradecimento.

Agradecimentos

Gostaria de expressar o meu sincero agradecimento à Escola Superior de Ciência e Tecnologia pela oportunidade de realizar este projeto.

Agradeço especialmente ao Professor Doutor José Carlos Miranda Nova Arnaud pela orientação dedicada ao longo do trabalho, bem como ao Professor Sérgio Francisco Sargo Ferreira Lopes pela coordenação e apoio.

Também reconheço o apoio dos meus colegas e amigos, cujas sugestões foram muito valiosas, assim como o incentivo da minha família, que foi fundamental para a conclusão deste percurso.

Resumo

O presente projeto tem como principal objetivo o desenvolvimento de uma aplicação móvel e web para a gestão de reservas de salas e equipamentos em ambientes corporativos, nomeadamente empresas e espaços de co-working. A aplicação foi concebida para oferecer uma solução moderna, acessível e eficiente, permitindo aos utilizadores consultar a disponibilidade de recursos, efectuar reservas, reportar avarias e receber notificações relevantes.

Para atingir os objetivos propostos, recorreu-se ao framework Flutter, que permite o desenvolvimento multiplataforma com uma única base de código, assegurando compatibilidade com dispositivos Android, iOS e navegadores web. Como backend, foi utilizado o serviço Firebase, que garante a autenticação segura dos utilizadores, o armazenamento em tempo real (Cloud Firestore) e o envio de notificações push.

A metodologia adoptada seguiu princípios de desenvolvimento ágil, com enfoque num processo iterativo, centrado na experiência do utilizador e com flexibilidade para responder a novos requisitos. As funcionalidades principais incluem autenticação, controlo de permissões, gestão de reservas e notificação de problemas técnicos.

O sistema desenvolvido apresenta uma interface moderna e responsiva, ajustada às exigências dos ambientes empresariais, proporcionando uma utilização prática e intuitiva. No final do documento, no âmbito das conclusões, é realizada uma reflexão crítica sobre os resultados alcançados, bem como a apresentação de propostas de desenvolvimento futuro, como a integração com leitores de QR Code e o reforço da componente analítica do sistema.

Abstract

The main goal of this project is the development of a mobile and web application for managing the reservation of rooms and equipment in corporate environments, such as companies and co-working spaces. The application was designed to offer a modern, accessible, and efficient solution, allowing users to check resource availability, make reservations, report malfunctions, and receive relevant notifications.

To achieve the proposed objectives, the Flutter framework was used, enabling cross-platform development with a single codebase, ensuring compatibility with Android, iOS, and web browsers. Firebase was adopted as the backend service, providing secure user authentication, real-time data storage (Cloud Firestore), and push notifications.

The adopted methodology followed agile development principles, focusing on an iterative process centred on user experience and adaptable to new requirements. The main features implemented include user authentication, permission control, reservation management, and issue reporting.

The resulting system presents a modern and responsive interface, tailored to the needs of business environments, offering a practical and intuitive user experience. At the end of the document, in the conclusions section, a critical reflection is presented on the results achieved, along with suggestions for future developments, such as QR Code integration and enhanced data analytics.

Lista de abreviaturas e siglas

API – Interface de Programação de Aplicações (Application Programming Interface).

BD – Base de Dados.

CI/CD – Integração Contínua / Entrega Contínua (Continuous Integration / Continuous Deployment).

CRUD – Criar, Ler, Atualizar, Eliminar (Create, Read, Update, Delete).

FCM – Serviço de Mensagens na Nuvem da Firebase (Firebase Cloud Messaging).

IDE – Ambiente Integrado de Desenvolvimento (Integrated Development Environment).

JSON – Notação de Objetos JavaScript (JavaScript Object Notation).

QR Code – Código de Resposta Rápida (Quick Response Code).

UI – Interface de Utilizador (User Interface).

UX – Experiência do Utilizador (User Experience).

VS Code – Visual Studio Code (Ambiente de desenvolvimento da Microsoft).

Índice

Índice de apêndice/anexo

Índice de figuras

Índice de tabelas

1. Introdução

Neste trabalho, pretende dar-se conta de um projeto que se enquadra no relatório final da unidade curricular de Projeto de Engenharia Informática em Contexto Empresarial, do curso de Licenciatura em Engenharia Informática. Este projeto, centrado na área do desenvolvimento de software e da mobilidade digital, visa responder a uma necessidade comum em ambientes corporativos: a gestão eficiente de espaços e equipamentos partilhados.

A escolha do tema surgiu do reconhecimento da crescente importância da digitalização dos processos internos nas organizações, nomeadamente no que diz respeito à otimização da utilização de recursos como salas de reunião, equipamentos técnicos ou áreas comuns. Com base nessa realidade, o projeto propõe o desenvolvimento de uma aplicação móvel e web multiplataforma, desenvolvida em Flutter, com Firebase como serviço de backend, para permitir a gestão centralizada de reservas, notificação de avarias e envio de lembretes automáticos.

Através da realização do presente trabalho, pretende-se:

- Implementar uma aplicação funcional que permita gerir reservas de salas e equipamentos;
- Caracterizar os principais desafios técnicos no desenvolvimento de sistemas multiplataforma com sincronização em tempo real;
- Identificar os requisitos essenciais para uma experiência de utilização intuitiva e eficiente;
- Apontar melhorias futuras com base numa análise crítica do sistema desenvolvido.

Este projeto é de iniciativa própria e não está associado a qualquer entidade externa. Todo o planeamento, design e desenvolvimento foram realizados individualmente, com o objetivo de aplicar os conhecimentos adquiridos ao longo do curso e consolidar competências técnicas e metodológicas.

Na primeira parte do documento, serão apresentados o enquadramento do tema, a fundamentação teórica e os trabalhos de referência que sustentam a proposta. Na segunda parte, será descrita a metodologia adotada, incluindo as tecnologias utilizadas e a organização do desenvolvimento. Segue-se a descrição detalhada do sistema implementado, com destaque para as funcionalidades principais. Por fim, são

apresentadas as conclusões, as limitações identificadas e sugestões de desenvolvimento futuro.

2. Enquadramento e Justificação

2.1 Introdução ao Tema

A gestão eficiente de espaços e equipamentos é uma necessidade crescente em ambientes empresariais, instituições educativas e espaços de co-working. Problemas como a sobreposição de reservas, a indisponibilidade inesperada de equipamentos ou a ausência de um sistema centralizado para registo e controlo de recursos são fatores que afetam negativamente a organização e produtividade das instituições. A tecnologia tem desempenhado um papel fundamental na automatização de processos administrativos, contribuindo para uma melhor alocação de recursos e maior eficiência operacional.

Neste contexto, a presente proposta consiste no desenvolvimento de uma aplicação móvel e web multiplataforma para a gestão de reservas de salas e equipamentos, com funcionalidades de autenticação e notificação de avarias. O projeto tira partido do framework Flutter, que permite o desenvolvimento de interfaces nativas para diferentes plataformas a partir de um único código-base, e do Firebase, uma solução em nuvem da Google que proporciona serviços como base de dados em tempo real, autenticação e envio de notificações.

2.2 Justificação da Escolha do Projeto

A escolha deste projeto surgiu da observação de limitações recorrentes em instituições e empresas que ainda utilizam sistemas manuais ou aplicações não especializadas para gerir a reserva de salas e equipamentos. A inexistência de uma aplicação simples, acessível e responsiva, que permita aos utilizadores gerir os seus próprios agendamentos de forma segura e intuitiva, constitui uma lacuna que este projeto pretende colmatar.

Além disso, a implementação prática de um sistema funcional com tecnologias modernas como Flutter e Firebase representa uma excelente oportunidade de aplicar, de forma integrada, os conhecimentos adquiridos ao longo da Licenciatura em Engenharia Informática. Esta escolha permite também aprofundar competências nas

áreas do desenvolvimento mobile, sincronização em tempo real, segurança e experiência do utilizador.

2.3 Enquadramento na Unidade Curricular

O presente projeto insere-se na unidade curricular de Projeto de Engenharia Informática em Contexto Empresarial, do terceiro ano da Licenciatura em Engenharia Informática da Escola Superior de Ciência e Tecnologia. Esta unidade curricular tem como principal objetivo promover a aplicação prática de conhecimentos técnicos e metodológicos adquiridos ao longo do curso, através da realização de um projeto que simule um contexto profissional ou empresarial.

Apesar de não ter sido desenvolvido numa empresa ou entidade externa, o projeto cumpre os objetivos definidos para a unidade curricular, nomeadamente o planeamento, desenvolvimento e documentação de uma aplicação informática, seguindo boas práticas de engenharia de software e aplicando metodologias adequadas de gestão de projeto e desenvolvimento de sistemas.

2.4 Metodologia de Desenvolvimento

A metodologia de desenvolvimento adotada neste projeto baseou-se nos princípios das metodologias ágeis, com especial enfoque na iteratividade, entregas incrementais e ajustes contínuos durante o processo de implementação. O projeto foi organizado em etapas distintas: levantamento de requisitos, planeamento, desenvolvimento da interface e das funcionalidades principais, integração com o backend (Firebase), testes e validação.

A aplicação foi desenvolvida com Flutter, escolhida pela sua capacidade de gerar aplicações para Android, iOS e Web a partir de um único código-fonte. O backend recorreu à plataforma Firebase, pela sua fácil integração com Flutter e pelos serviços nativos como Firebase Authentication, Cloud Firestore e Firebase Cloud Messaging.

O desenvolvimento foi orientado por tarefas específicas e por marcos temporais definidos no plano de trabalho, com testes progressivos e uma preocupação constante com a usabilidade, a responsividade e a segurança dos dados.

3. Projetos Similares e Soluções Associadas

3.1 Análise de Soluções Existentes

Antes da definição e desenvolvimento do presente projeto, foi realizada uma análise de diversas soluções tecnológicas existentes no mercado, com o objetivo de compreender as funcionalidades oferecidas, as abordagens utilizadas e as limitações que possam existir. Esta análise permite posicionar o projeto proposto no contexto atual e identificar oportunidades de inovação.

Entre as soluções analisadas, destacam-se:

- Google Calendar – Amplamente utilizado para marcação de eventos e reuniões. Permite o agendamento partilhado, envio de convites e integração com outras ferramentas Google. No entanto, não está orientado especificamente para a gestão de equipamentos nem possui controlo granular de permissões por utilizador sobre os recursos.

- Skedda – Plataforma especializada na gestão de espaços de trabalho, salas e instalações. Oferece funcionalidades como regras de acesso, horários de funcionamento e painéis de administração. Apesar de ser bastante completa, o seu modelo de subscrição pode não ser acessível a todas as organizações.

Estas soluções demonstram abordagens robustas à problemática da gestão de recursos, mas também revelam algumas barreiras, sobretudo ao nível da flexibilidade, da simplicidade de uso e do custo.

3.2 Identificação de Lacunas e Oportunidades de Melhoria

Com base na análise efetuada, identificam-se várias lacunas nas soluções atualmente disponíveis que justificam e motivam o desenvolvimento do presente projeto:

- Ausência de foco em equipamentos: A maioria das plataformas concentra-se na gestão de espaços, negligenciando a necessidade de reservar e acompanhar o uso de equipamentos tecnológicos ou laboratoriais.

- Falta de integração total entre funcionalidades: É comum a existência de sistemas que exigem múltiplas ferramentas externas para funcionalidades básicas, como notificações, autenticação ou registo de problemas.

- Barreiras de custo e complexidade: Soluções como o OfficeSpace são financeiramente inacessíveis para pequenas empresas ou instituições académicas, além de exigirem implementação especializada.

Estas lacunas representam oportunidades claras para a criação de uma solução modular, gratuita, intuitiva e multiplataforma, que permita reservar espaços e equipamentos, receber notificações e reportar avarias num único sistema integrado, acessível a partir de qualquer dispositivo.

4. Fundamentação Teórica

O desenvolvimento de aplicações móveis modernas exige o domínio de tecnologias e abordagens que promovam eficiência, portabilidade, boa experiência de utilização e gestão eficaz de dados. Este capítulo apresenta os fundamentos teóricos que sustentam as escolhas tecnológicas do projeto, nomeadamente o uso do Flutter para desenvolvimento multiplataforma, do Firebase como backend na nuvem, bem como princípios de UX/UI e modelação de dados em tempo real.

4.1 Desenvolvimento Multiplataforma com Flutter

O Flutter é um framework de desenvolvimento de interfaces criado pela Google, que permite construir aplicações nativas para Android, iOS, Web e outras plataformas a partir de um único código-fonte. Utiliza a linguagem de programação Dart, também da Google, conhecida pela sua simplicidade, performance e suporte à programação orientada a objetos.

Ao contrário de frameworks baseados em renderização nativa (como o React Native), o Flutter utiliza o seu próprio motor gráfico para desenhar a interface do utilizador, garantindo uma aparência e comportamento consistentes em todas as plataformas. Isto contribui para uma performance elevada, tempos de resposta reduzidos e maior controlo visual sobre os componentes.

As principais vantagens do Flutter no contexto deste projeto incluem:

- Multiplataforma real: suporte para Android, iOS, Web e até desktop, sem necessidade de duplicar código;

- Hot reload: permite visualizar alterações no código em tempo real, acelerando o desenvolvimento;
- Comunidade ativa e vasto ecossistema: suporte de pacotes prontos para autenticação, bases de dados, notificações, entre outros;
- Integração facilitada com Firebase: ambiente nativo para aplicações Flutter.

No projeto em causa, o Flutter permitiu acelerar o processo de desenvolvimento, garantindo uma experiência de utilização consistente e eficiente, independentemente do dispositivo.

4.2 Utilização de Firebase (Firestore, Auth, Cloud Messaging)

O Firebase é uma plataforma de desenvolvimento de aplicações da Google que fornece uma série de serviços backend integrados e baseados na nuvem. Foi escolhido como infraestrutura de apoio ao projeto pelas suas características de escalabilidade, integração com Flutter e simplicidade de configuração.

Os principais serviços utilizados foram:

- Firebase Authentication: Permite a autenticação de utilizadores através de email e palavra-passe, Google, Facebook, entre outros. Inclui gestão de sessões, verificação de identidade e recuperação de credenciais, garantindo segurança e facilidade de utilização.
- Cloud Firestore: Base de dados NoSQL em tempo real. Permite armazenar dados em documentos organizados em coleções. Suporta leitura e escrita em tempo real, o que é fundamental para atualizar dinamicamente o estado das reservas, equipamentos e notificações.
- Firebase Cloud Messaging (FCM): Serviço de envio de notificações push para aplicações móveis. Utilizado no projeto para alertar os utilizadores sobre reservas iminentes, alterações e reporte de avarias.

Estes serviços, totalmente integrados com Flutter, permitiram implementar um backend robusto, seguro e com baixo custo, ideal para aplicações em contexto académico ou empresarial com necessidades moderadas de escalabilidade.

4.3 Boas Práticas de UX/UI para Aplicações Móveis

A experiência do utilizador (UX – User Experience) e a interface do utilizador (UI – User Interface) são elementos cruciais no sucesso de uma aplicação. Mesmo que o sistema seja funcional, uma má experiência visual ou de navegação pode comprometer a sua adoção.

As principais boas práticas seguidas no projeto foram:

- Clareza e simplicidade: utilização de menus bem organizados, linguagem acessível e ações facilmente identificáveis (como reservar, cancelar, editar);
- Design responsivo: adaptação do layout a diferentes tamanhos de ecrã (smartphones, tablets, web);
- Feedback visual imediato: utilização de indicadores de carregamento, confirmações visuais de ações realizadas e mensagens de erro compreensíveis;
- Hierarquia visual adequada: uso de cores, tamanhos de letra e espaçamento para guiar o utilizador e destacar elementos importantes;
- Consistência: manutenção de padrões de navegação e design ao longo da aplicação, facilitando o uso intuitivo.

A aplicação foi concebida com recurso a bibliotecas de componentes visuais do Flutter e padrões de design modernos, resultando numa interface limpa, funcional e acessível, mesmo para utilizadores com pouca experiência digital.

4.4 Modelação e Estruturação de Dados em Tempo Real

A estruturação adequada dos dados é essencial para garantir o bom funcionamento de sistemas dinâmicos, particularmente quando se trabalha com bases de dados em tempo real, como é o caso do Cloud Firestore.

Dado que o Firestore é um sistema de base de dados NoSQL, os dados são organizados de forma hierárquica, em coleções e documentos. Cada documento pode conter pares chave-valor e subcoleções. Esta flexibilidade exige um planeamento cuidadoso da estrutura para evitar redundância e garantir desempenho.

No projeto, a modelação dos dados seguiu os seguintes princípios:

- Separação lógica por entidades: coleções distintas para utilizadores, reservas e avarias;
- Documentos com identificadores únicos: facilitam operações de leitura, escrita e atualização;
- Indexação personalizada: para acelerar consultas frequentes (por exemplo, reservas por utilizador ou por sala);
- Regras de segurança (Firestore Rules): definem que dados cada utilizador pode ler ou escrever, garantindo privacidade e controlo de permissões.

A estruturação em tempo real permite que, por exemplo, quando uma reserva é criada ou modificada, todos os utilizadores com acesso vejam imediatamente a alteração, sem necessidade de atualização manual. Isto melhora a eficiência, a transparência e reduz o risco de conflitos de agendamento.

5. Requisitos

5.1 Requisitos Técnicos

5.1.1 Requisitos Funcionais

Os requisitos funcionais definem o comportamento do sistema e as funcionalidades que este deve assegurar para satisfazer as necessidades do utilizador. No contexto deste projeto, os requisitos funcionais identificados foram:

- RF01 – Permitir o registo de utilizadores através de email e palavra-passe;
- RF02 – Permitir o login e logout de utilizadores;
- RF03 – Permitir ao utilizador criar reservas de salas e equipamentos;
- RF04 – Apresentar ao utilizador a disponibilidade de salas e equipamentos;
- RF05 – Permitir ao utilizador consultar as suas próprias reservas;
- RF06 – Permitir ao utilizador editar ou cancelar apenas as suas reservas;
- RF07 – Validar reservas com um QR Code associado, gerado automaticamente;
- RF08 – Validar entradas de dados obrigatórios nos formulários;
- RF9 – Adaptar a interface da aplicação a diferentes tipos de dispositivos (mobile e web).

5.1.2 Requisitos Não-Funcionais

Estes requisitos referem-se às características de qualidade que o sistema deve garantir:

- RNF01 – Segurança: O sistema deve proteger os dados dos utilizadores com autenticação segura e regras de acesso no Firebase;
- RNF02 – Performance: A resposta da aplicação deve ser inferior a 1 segundo nas operações principais;
- RNF03 – Responsividade: A aplicação deve adaptar-se a ecrãs de diferentes tamanhos (smartphone, tablet, desktop);
- RNF04 – Multiplataforma: O sistema deve funcionar corretamente em Android e Web;
- RNF05 – Simplicidade de utilização: A navegação deve ser intuitiva e acessível, mesmo para utilizadores com pouca experiência digital;
- RNF06 – Organização modular do código: O código deve estar dividido em componentes reutilizáveis para facilitar a manutenção;
- RNF07 – Baixo custo de infraestrutura: O sistema deve funcionar integralmente com os serviços gratuitos do Firebase.

5.2 Modelação UML

Para representar graficamente os elementos do sistema e as interações, foram produzidos diagramas UML com as seguintes finalidades:

5.2.1 Diagrama de Casos de Utilização

Este diagrama identifica os atores (utilizadores) e os casos de uso principais do sistema.

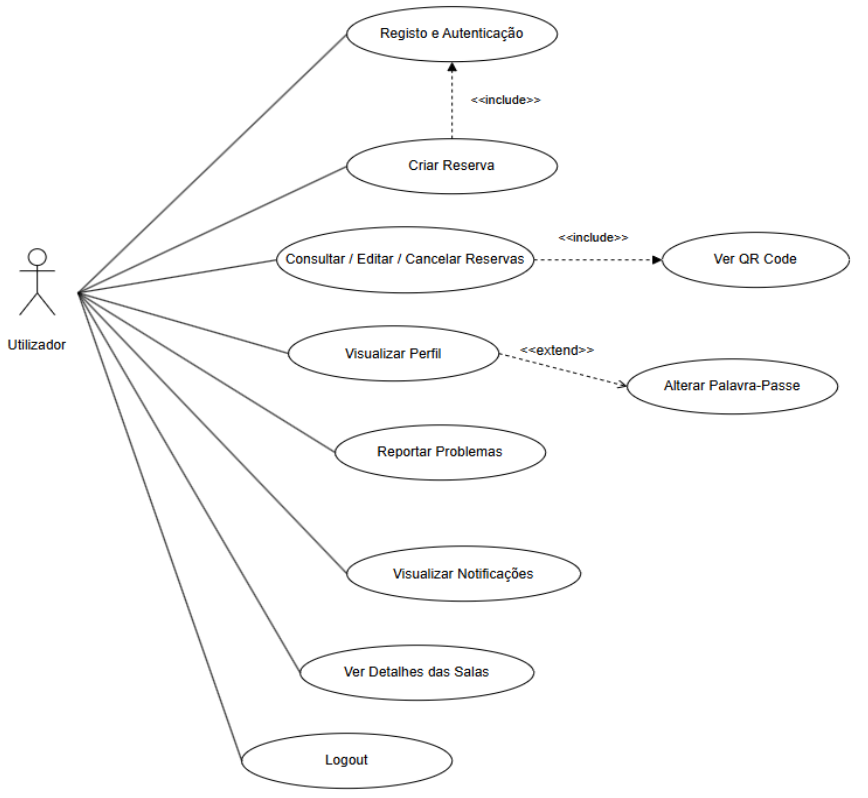

Figura 1. Diagrama de Casos de Utilização

Casos de uso principais:

- Registo e autenticação

O utilizador pode criar conta ou iniciar sessão na aplicação com email e palavra-passe.

- Criar reserva

Permite ao utilizador reservar salas e/ou equipamentos, com verificação de disponibilidade.

- Consultar, editar ou cancelar reservas e visualizar QR Code

O utilizador pode:

• Consultar as suas reservas;

• Editar ou cancelar reservas;

• Gerar e visualizar um QR Code associado à reserva.

Todas estas ações estão integradas na mesma página de gestão de reservas.

- Visualizar perfil e alterar palavra-passe

O utilizador pode:

• Ver os seus dados pessoais;

• Alterar a palavra-passe da conta.

- Reportar problemas

Permite ao utilizador submeter problemas técnicos ou sugestões de melhoria.

- Visualizar notificações

A aplicação mostra notificações ao utilizador, como confirmações de reservas ou alertas.

- Ver detalhes das salas

O utilizador pode consultar mais informação sobre as salas e os equipamentos disponíveis antes de efetuar uma reserva.

- Logout

O utilizador pode terminar a sessão da aplicação de forma segura.

5.2.2 Diagrama de Sequência

O diagrama de sequência exemplifica o processo de criação de uma reserva, incluindo a geração do QR Code:

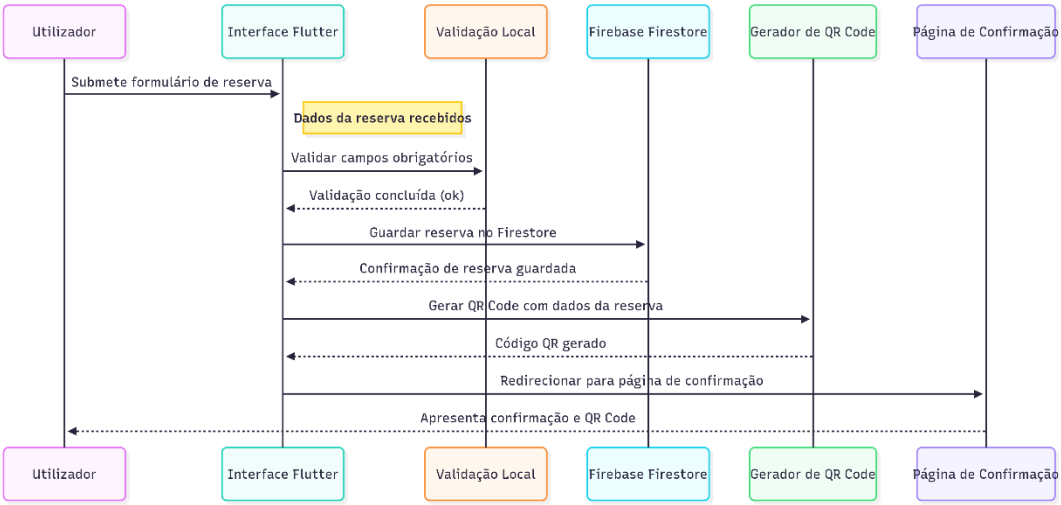

Figura 2. Diagrama de Sequência: Criação de Reserva com QR Code

Passos principais:

1. Utilizador submete formulário com dados da reserva;
2. Aplicação valida os campos localmente;
3. Reserva é guardada no Firestore;
4. Código QR é gerado com os dados da reserva;
5. Utilizador é redirecionado para página de confirmação.

5.2.3 Diagrama de Estados

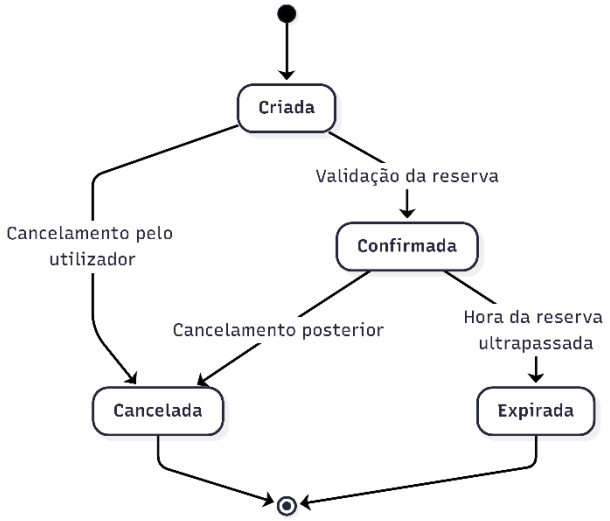

Figura 3. Diagrama de Estados: Ciclo de Vida de uma Reserva

Estados possíveis:

- Reservada → Editada → Confirmada → Cancelada

5.2.4 Diagrama de Componentes

Este diagrama mostra a estrutura geral da aplicação e as suas ligações com serviços externos (Firebase):

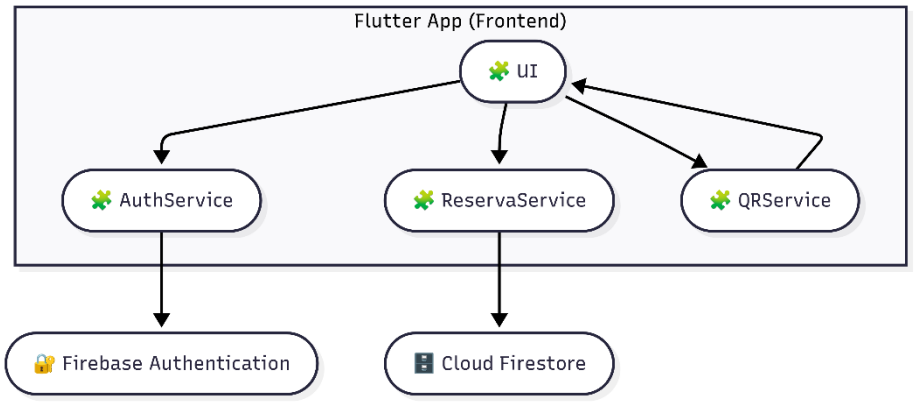

Figura 4. Diagrama de Componentes do Sistema

Componentes principais:

- App Flutter (Frontend)
- Firebase Authentication
- Cloud Firestore
- Geração de QR Code (qr_flutter)

6. Implementação do Sistema

6.1 Organização do Projeto em Flutter

O projeto foi desenvolvido com Flutter, e a estrutura de diretórios foi organizada de forma modular para facilitar a manutenção e escalabilidade do sistema.

Estrutura de Ficheiros

lib/

- main.dart: Ficheiro principal onde a aplicação Flutter é inicializada.
- firebase_options.dart: Ficheiro gerado automaticamente com as opções de configuração do Firebase (via flutterfire CLI).
- repositorios/: Contém os repositórios responsáveis por comunicar com fontes de dados como Firebase Firestore ou APIs. Implementa a lógica de acesso e manipulação de dados.

- telas/

  Contém todos os ecrãs (ou páginas) da aplicação, cada um encapsulado num ficheiro individual. Exemplos:

  - home.dart: ecrã principal após o login.
  - login_registo.dart: ecrã de autenticação (login e registo).
  - minhasreservas.dart: lista de reservas do utilizador.
  - novareserva.dart: ecrã para criar uma nova reserva.
  - relatarproblema.dart: formulário para relatar avarias ou problemas.
  - Entre outros.

Widgets

A interface foi construída com base em widgets personalizados e reutilizáveis, garantindo um layout limpo e consistente em toda a aplicação.

Packages Usados

- firebase_core: integração base com Firebase;
- firebase_auth: autenticação de utilizadores;
- cloud_firestore: base de dados NoSQL em tempo real;
- provider: gestão de estado;
- flutter_local_notifications: exibição de notificações locais;
- intl: formatação de datas e horas.

6.2 Configuração e Utilização do Firebase

A aplicação está integrada com Firebase, funcionando como backend principal.

Autenticação

Utilizou-se o Firebase Authentication para registo e login de utilizadores por email e palavra-passe. A segurança é reforçada com validação de sessão e bloqueio de acessos não autorizados.

Cloud Firestore

Foi escolhida como base de dados pela sua natureza em tempo real. A estrutura da Firestore inclui:

- users: Armazena os dados dos utilizadores registados na aplicação. Inclui informações como nome, email, e eventualmente permissões e preferências.
- reservas: Contém todas as reservas efetuadas pelos utilizadores, com campos que referenciam datas, horários, recursos reservados e o utilizador associado.
- problemas_reportados: Regista os problemas ou avarias submetidos pelos utilizadores (ex: equipamentos avariados, falhas em salas). Cada registo pode incluir descrição, data, prioridade e estado da resolução.
- notificações: Armazena notificações enviadas aos utilizadores.

6.3 Funcionalidades Implementadas

A aplicação integra as seguintes funcionalidades principais:

- **Autenticação de Utilizadores**
    - Registo e login com Firebase Auth.
    - Controlo de sessões e logout seguro.

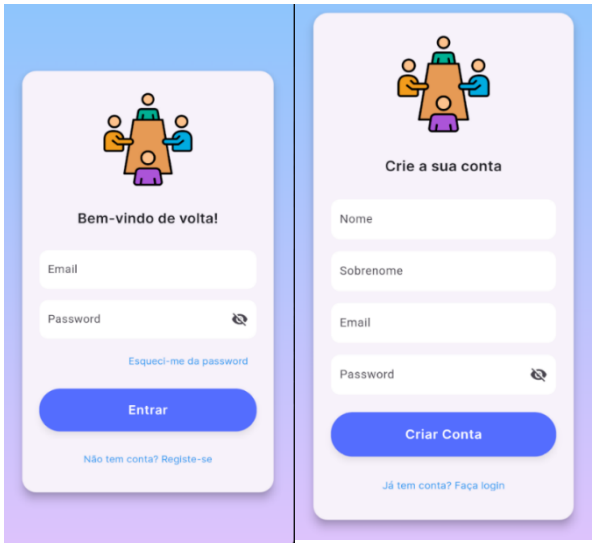

Figura 5. Ecrã de login e registo de utilizadores com integração Firebase Auth.

- **Reserva de Equipamentos e Salas**
    - Formulário para criar reservas com escolha de sala, equipamento, data e hora.

Figura 6. Formulário de criação de reservas com seleção de sala, equipamento, data e horário.

- **Gestão Pessoal de Reservas**
    - Cada utilizador pode visualizar, editar ou cancelar apenas as suas reservas.
- **Geração de QR Code para reservas**
    - Cada reserva criada gera automaticamente um código QR com os seus dados.
    - Facilita o controlo e verificação rápida da reserva no momento da utilização.

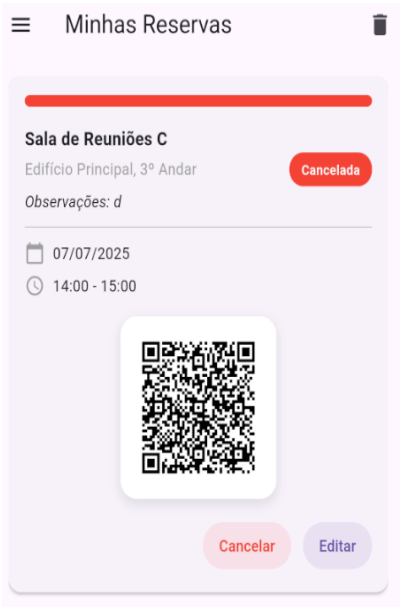

Figura 7. Vista das reservas do próprio utilizador com opções de edição, cancelamento e com QR Code gerado automaticamente para cada reserva.

- **Reporte de Avarias**
  - Interface para registar problemas técnicos com sala ou equipamento;

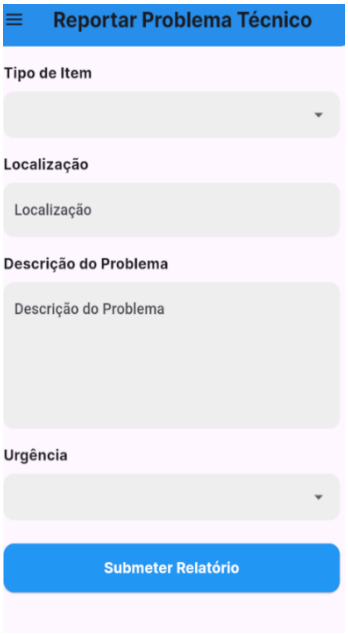

Figura 8. Formulário para registo de problemas em salas e equipamentos.

- **Notificações**
    - Visualização de notificações armazenadas na base de dados com atualizações de estado de reservas.

Figura 9. Ecrã de notificações com atualizações sobre o estado de reservas

- **Consulta Detalhada de Salas e Equipamentos**
    - Visualização das características de cada sala (capacidade, tipo, localização, climatização e Wi-Fi).

Figura 10. Consulta detalhada das características de salas e equipamentos

- **Gestão de Perfil do Utilizador**
    - Visualização de dados pessoais (nome e email).
    - Edição da imagem de perfil (upload a partir da galeria ou câmara).
    - Acesso à alteração da palavra-passe.

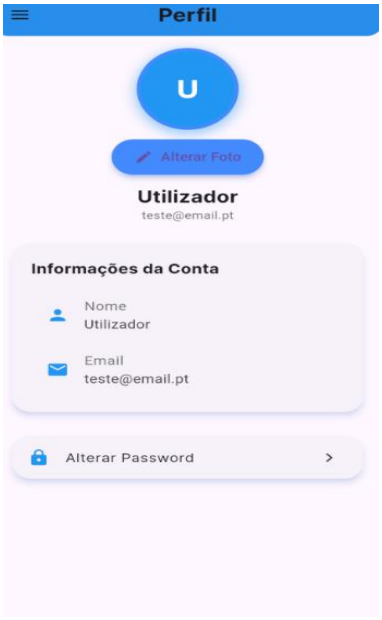

Figura 11. Ecrã de perfil do utilizador com imagem, nome, email e botão para editar foto.

6.4 Interface da Aplicação (Design e Usabilidade)

A aplicação foi concebida com um design moderno, limpo e adaptável, garantindo uma boa experiência de utilização em dispositivos móveis e navegadores web.

Princípios aplicados:

- Navegação clara com menus laterais e botões intuitivos;
- Utilização de cores neutras e destaques visuais consistentes;
- Textos curtos e ícones representativos;
- Compatibilidade com ecrãs de diferentes tamanhos;
- Feedback visual em todas as interações.

7. Testes de Validação

Para garantir que o sistema atende aos requisitos funcionais, foram elaborados diversos casos de teste para os principais requisitos. A quantidade de casos de teste varia conforme a complexidade e criticidade de cada requisito.
A seguir, um exemplo de especificação de caso de teste:

ESPECIFICAÇÃO DE CASO DE TESTE	
Identificador:	AUTH-TC-001
Objetivo:	Verificar login com credenciais válidas.
Autor(es)	Gonçalo de Lira Pereira
Especificação de Entradas a. Email válido b. Palavra-passe correta	
Especificação de Saídas a. Utilizador autenticado b. Redirecionado para o menu principal	
Outros NA	
Dependências Conta de utilizador previamente criada	

Tabela 1. Caso de teste AUTH-TC-001: Verificação de login com credenciais válidas

Devido à extensão, a lista completa dos casos de teste está disponível no Apêndice A.

8. Cronograma

		Nome	Duração	Início	Fim
1		Cronograma	1 dia?	10/03/25, 08:00	10/03/25, 17:00
2		**Cronograma**	**118 dias?**	**10/03/25, 08:00**	**24/07/25, 17:00**
3		**Fase 1 - Planeamento e Análise**	**14 dias?**	**10/03/25, 08:00**	**25/03/25, 17:00**
4		Definir o problema e os objetivos do sistema	4 dias?	10/03/25, 08:00	13/03/25, 17:00
5		Selecionar ferramentas e tecnologias	7 dias?	13/03/25, 08:00	20/03/25, 17:00
6		Levantar e analisar os requisitos funcionais ...	5 dias?	20/03/25, 08:00	25/03/25, 17:00
7		**Fase 2 – Design e Arquitetura**	**16 dias?**	**27/03/25, 08:00**	**14/04/25, 17:00**
8		Estruturar arquitetura geral da aplicação	7 dias?	27/03/25, 08:00	03/04/25, 17:00
9		Modelar dados e fluxos (UML)	4 dias?	03/04/25, 07:00	07/04/25, 17:00
10		Preparar o design da interface	7 dias?	07/04/25, 07:00	14/04/25, 17:00
11		**Fase 3 – Desenvolvimento**	**34 dias?**	**14/04/25, 08:00**	**22/05/25, 17:00**
12		Desenvolver o backend (Firestore, autentic...	7 dias?	14/04/25, 08:00	21/04/25, 17:00
13		Criar os componentes do frontend	20 dias?	21/04/25, 08:00	13/05/25, 17:00
14		Implementar funcionalidades principais	9 dias?	13/05/25, 08:00	22/05/25, 17:00
15		**Fase 4 – Testes e Validação**	**19 dias?**	**22/05/25, 08:00**	**12/06/25, 17:00**
16		Validar o funcionamento correto da aplicação	8 dias	22/05/25, 08:00	30/05/25, 17:00
17		Corrigir erros e afinar o desempenho	9 dias?	30/05/25, 08:00	09/06/25, 17:00
18		Confirmar compatibilidade entre plataformas	4 dias?	09/06/25, 08:00	12/06/25, 17:00
19		**Fase 5 – Refinamento e Documentação**	**17 dias?**	**12/06/25, 08:00**	**01/07/25, 17:00**
20		Melhorar a usabilidade e a aparência da apli...	12 dias?	12/06/25, 08:00	25/06/25, 17:00
21		Produzir a documentação técnica	15 dias?	12/06/25, 08:00	28/06/25, 17:00
22		Preparar a entrega final	3 dias?	28/06/25, 08:00	01/07/25, 17:00
23		**Fase 6 – Entrega e Apresentação**	**21 dias?**	**01/07/25, 08:00**	**24/07/25, 17:00**
24		Finalizar e entregar o projeto	16 dias?	01/07/25, 08:00	18/07/25, 17:00
25		Apresentar o trabalho	6 dias?	18/07/25, 08:00	24/07/25, 17:00

Figura 12. Cronograma Detalhado do Projeto

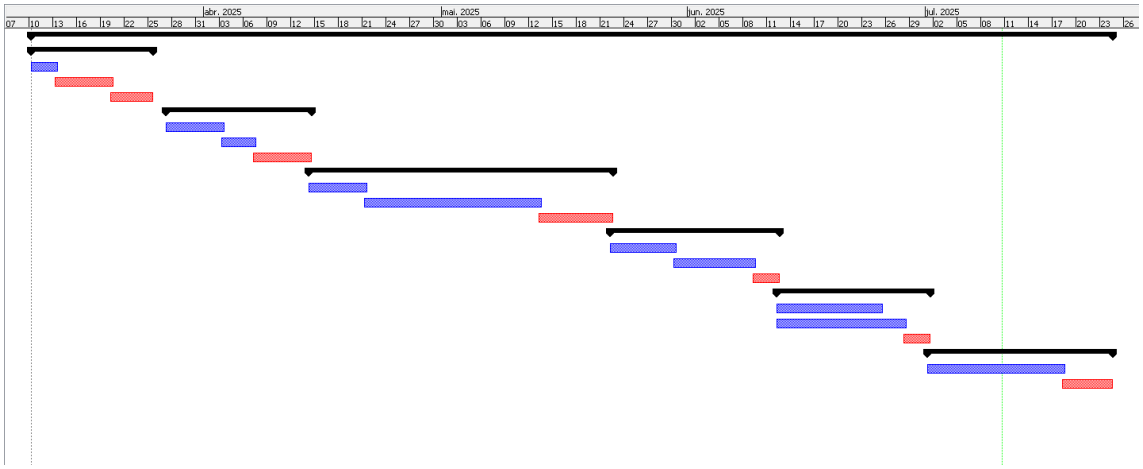

Figura 13. Diagrama de Gantt do Cronograma do Projeto

9. Meios Previstos e Meios Necessários

A execução do projeto exigiu a definição de um conjunto de meios humanos, materiais e tecnológicos, inicialmente previstos e posteriormente ajustados à medida que o desenvolvimento avançou.

Meios Humanos

O projeto foi desenvolvido individualmente pelo autor, sem colaboração externa. Esta abordagem permitiu um maior controlo e autonomia sobre todas as fases, mas exigiu uma gestão rigorosa do tempo e uma abordagem autoformativa contínua, nomeadamente na aprendizagem prática do Flutter e do Firebase.

Meios Materiais e Tecnológicos

- Computador pessoal com sistema operativo compatível com Android Studio e VS Code;
- Ambiente de desenvolvimento (IDE): Visual Studio Code, com extensões específicas para Flutter e Dart;
- Simuladores e dispositivos físicos Android para testes;
- Acesso à internet para sincronização com Firebase e testes em tempo real;
- Plataforma Firebase, com serviços gratuitos.

Desfasamentos

Embora os meios previstos tenham sido, na maioria, suficientes, alguns desfasamentos foram identificados:

- A gestão do tempo revelou-se desafiante, especialmente nas fases finais de testes e ajustes da interface, o que levou a revisões no cronograma original.

10. Problemas e Decisões

Durante o desenvolvimento do projeto, surgiram diversos desafios técnicos e organizacionais, que exigiram decisões fundamentadas para garantir o sucesso da implementação.

Problemas Identificados

- Integração com Firebase: embora bem documentado, alguns erros de configuração na segurança das Firestore Rules resultaram em restrições inesperadas de leitura/escrita, exigindo testes manuais detalhados.
- Gestão de permissões por utilizador: a lógica de permitir que cada utilizador edite apenas as suas reservas implicou a implementação de regras de negócio adicionais e validações no frontend e no backend.
- Testes em diferentes plataformas: o comportamento da interface variava ligeiramente entre mobile e web, o que obrigou a ajustes no design responsivo.

Decisões Tomadas

- Optou-se por testar inicialmente em Android, uma vez que o processo de configuração era mais direto do que no iOS.
- Criou-se uma estrutura modular no código Flutter, facilitando a reutilização de componentes e a escalabilidade do sistema.

11. Análise de Resultados

A avaliação dos resultados obtidos teve como base os objetivos definidos na introdução do relatório. A concretização do projeto foi bem-sucedida, com a implementação das funcionalidades essenciais e testes funcionais positivos.

Resultados Concretizados

- Autenticação funcional com Firebase Authentication (email e palavra-passe);
- Base de dados Cloud Firestore estruturada com coleções separadas para utilizadores, reservas, salas, equipamentos e avarias;
- Sistema de reservas funcional com possibilidade de criação, visualização e cancelamento de reservas pelos próprios utilizadores;
- Design responsivo e moderno, adaptado a mobile e web.

Métricas Relevantes

- Tempo médio de resposta da interface: inferior a 0,5 segundos em ações de reserva ou leitura de dados;
- Tempo de sincronização Firebase em tempo real: praticamente instantâneo (< 200 ms);
- Cobertura funcional: 100% dos objetivos mínimos definidos foram atingidos;
- Compatibilidade: testado com sucesso em Android e navegador Web.

Resultados Não Atingidos

- Não foi incluído um painel analítico com estatísticas, o que foi identificado como uma oportunidade para desenvolvimento futuro.

12.Conclusões

O presente projeto teve como objetivo desenvolver uma aplicação multiplataforma, em Flutter com backend Firebase, para a gestão de reservas de salas e equipamentos. A solução permite aos utilizadores efetuar reservas, consultar disponibilidade, gerir as suas marcações e validar informações através de códigos QR.

Foram concretizados os principais objetivos propostos, incluindo a autenticação de utilizadores, a criação e visualização de reservas, a geração de QR Code e a compatibilidade com Android e Web. Ficaram por implementar, nesta fase, funcionalidades como notificações push e validação externa via leitor de QR Code, identificadas como melhorias a desenvolver futuramente.

Durante a execução do projeto surgiram alguns constrangimentos técnicos e de tempo, nomeadamente ao nível da integração com Firebase e da gestão de permissões. Ainda assim, o trabalho permitiu aplicar e consolidar conhecimentos adquiridos no curso, desde a modelação de sistemas à construção de interfaces responsivas e comunicação com serviços cloud.

Este projeto, apesar de não ter decorrido numa entidade externa, representou um desafio real de desenvolvimento completo de software. A experiência revelou-se valiosa tanto do ponto de vista técnico como pessoal, destacando-se pela autonomia, capacidade de adaptação e superação de dificuldades.

13.Referências bibliográficas

[1] Google. *Firebase Documentation*. Firebase. https://firebase.google.com/docs

[2] Google. *Flutter Documentation*. Flutter. https://docs.flutter.dev

[3] López, L., & Soto, R. (2020). *Mobile app development with Flutter: A practical guide*. Packt Publishing.

[4] Martin, R. C. (2008). *Clean code: A handbook of agile software craftsmanship*. Prentice Hall.

[5] Nielsen, J. (1994). *Usability engineering*. Morgan Kaufmann.

[6] Pressman, R. S., & Maxim, B. R. (2014). *Software engineering: A practitioner's approach* (8th ed.). McGraw-Hill Education.

[7] Serrano, A., Gallardo, J., & Hernantes, J. (2018). *Software development and cloud migration using Firebase*. Springer.

[8] Sönmez, K. (2022). *Mastering Firebase for Flutter: Build real-time mobile applications with Firebase and Flutter*. Packt Publishing.

14.Glossário

Autenticação – Processo de verificação da identidade de um utilizador antes de permitir o acesso ao sistema.

Base de Dados NoSQL – Tipo de base de dados não relacional que armazena dados em formato flexível, como documentos JSON. O Firestore é um exemplo deste tipo.

Cloud Firestore – Serviço de base de dados em tempo real fornecido pela plataforma Firebase, baseado em documentos e coleções.

Firebase – Plataforma de desenvolvimento de aplicações da Google que fornece serviços como autenticação, base de dados em tempo real, armazenamento e notificações push.

Flutter – Framework de código aberto da Google para desenvolvimento de aplicações multiplataforma (Android, iOS, Web e Desktop) com uma única base de código.

Frontend – Parte da aplicação visível e interativa com a qual o utilizador interage diretamente.

Gestão de Reservas – Funcionalidade que permite ao utilizador consultar, criar, editar e cancelar reservas de salas ou equipamentos.

Interface Responsiva – Interface de utilizador que se adapta automaticamente a diferentes tamanhos e tipos de ecrã (smartphones, tablets, computadores).

QR Code – Código bidimensional que armazena dados de forma visual. Pode ser lido por dispositivos móveis para aceder rapidamente a informações, como dados de uma reserva.

SDK (Software Development Kit) – Conjunto de ferramentas que permite aos programadores desenvolver aplicações para plataformas específicas.

UID (User ID) – Identificador único de um utilizador no sistema, normalmente atribuído automaticamente pelo serviço de autenticação.

UX/UI (User Experience / User Interface) – Termos que se referem, respetivamente, à experiência do utilizador e ao design visual da interface com que este interage.

15.Apêndices/anexos

Apêndice A – Casos de Teste Detalhados

ESPECIFICAÇÃO DE CASO DE TESTE	
Identificador:	AUTH-TC-001
Objetivo:	Verificar login com credenciais válidas.
Autor(es)	Gonçalo de Lira Pereira
Especificação de Entradas • Email válido • Palavra-passe correta	
Especificação de Saídas • Utilizador autenticado • Redirecionado para o menu principal	
Outros NA	
Dependências Conta de utilizador previamente criada	

Tabela A1. Caso de teste AUTH-TC-001: Verificação de login com credenciais válidas

ESPECIFICAÇÃO DE CASO DE TESTE	
Identificador:	AUTH-TC-002
Objetivo:	Impedir acesso com credenciais inválidas.
Autor(es)	Gonçalo de Lira Pereira
Especificação de Entradas • Email válido • Palavra-passe incorreta	
Especificação de Saídas: • Mensagem de erro: "Credenciais inválidas"	
Outros NA	
Dependências Conta de utilizador existente	

Tabela A2. Caso de teste AUTH-TC-002: Impedir acesso com credenciais inválidas

ESPECIFICAÇÃO DE CASO DE TESTE	
Identificador:	AUTH-TC-003
Objetivo:	Verificar registo de novo utilizador.
Autor(es)	Gonçalo de Lira Pereira
Especificação de Entradas • Email novo • Palavra-passe com mínimo 6 caracteres	
Especificação de Saídas • Conta criada com sucesso • Redirecionamento para o login	
Outros NA	
Dependências Email ainda não registado	

Tabela A3. Caso de teste AUTH-TC-003: Verificação do registo de novo utilizador

ESPECIFICAÇÃO DE CASO DE TESTE	
Identificador:	RES-TC-001
Objetivo:	Criar nova reserva com dados válidos.
Autor(es)	Gonçalo de Lira Pereira
Especificação de Entradas • Sala "A" • Equipamento "Projetor" • Data/hora válidas	
Especificação de Saídas • Reserva criada e guardada no Firestore	
Outros NA	
Dependências Utilizador autenticado, recursos disponíveis	

Tabela A4. Caso de teste RES-TC-001: Criação de nova reserva com dados válidos

ESPECIFICAÇÃO DE CASO DE TESTE	
Identificador:	RES-TC-002
Objetivo:	Editar reserva existente criada pelo próprio utilizador.
Autor(es)	Gonçalo de Lira Pereira
Especificação de Entradas • Alteração de data/hora de reserva	
Especificação de Saídas • Reserva atualizada com sucesso	
Outros NA	
Dependências Reserva existente do utilizador	

Tabela A5. Caso de teste RES-TC-002: Edição de reserva existente pelo próprio utilizador

ESPECIFICAÇÃO DE CASO DE TESTE	
Identificador:	RES-TC-003
Objetivo:	Visualizar reservas existentes de todos os utilizadores.
Autor(es)	Gonçalo de Lira Pereira
Especificação de Entradas • Acesso à secção de histórico	
Especificação de Saídas • Lista completa de reservas visível em modo leitura	
Outros NA	
Dependências Utilizador autenticado	

Tabela A6. Caso de teste RES-TC-003: Visualização das reservas existentes de todos os utilizadores

ESPECIFICAÇÃO DE CASO DE TESTE	
Identificador:	AV-TC-002
Objetivo:	Impedir registo com campos obrigatórios vazios.
Autor(es)	Gonçalo de Lira Pereira
Especificação de Entradas • Submissão sem descrição	
Especificação de Saídas • Mensagem de erro: "Preencha todos os campos obrigatórios"	
Outros NA	
Dependências Validação de formulário ativa	

Tabela A7. Caso de teste AV-TC-002: Impedir registo com campos obrigatórios vazios

ESPECIFICAÇÃO DE CASO DE TESTE	
Identificador:	PLATF-TC-001
Objetivo:	Verificar compatibilidade da aplicação em Android.
Autor(es)	Gonçalo de Lira Pereira
Especificação de Entradas • Acesso em smartphone Android	
Especificação de Saídas • Aplicação totalmente funcional	
Outros NA	
Dependências Dispositivo físico com sistema Android	

Tabela A8. Caso de teste PLATF-TC-001: Verificação de compatibilidade da aplicação em Android

ESPECIFICAÇÃO DE CASO DE TESTE	
Identificador:	PLATF-TC-002
Objetivo:	Verificar compatibilidade da aplicação em navegador Web.
Autor(es)	Gonçalo de Lira Pereira
Especificação de Entradas • Acesso via navegador	
Especificação de Saídas • Aplicação funcional, interface adaptada	
Outros NA	
Dependências Build web publicada	

Tabela A9. Caso de teste PLATF-TC-002: Verificação de compatibilidade da aplicação em navegador Web

ESPECIFICAÇÃO DE CASO DE TESTE	
Identificador:	PLATF-TC-003
Objetivo:	Verificar responsividade em diferentes tamanhos de ecrã.
Autor(es)	Gonçalo de Lira Pereira
Especificação de Entradas • . Acesso a partir de desktop e mobile	
Especificação de Saídas • Layout ajusta-se corretamente ao dispositivo	
Outros NA	
Dependências Layout responsivo ativo no Flutter	

Tabela A10. Caso de teste PLATF-TC-003: Verificação de responsividade em diferentes tamanhos de ecrã

ESPECIFICAÇÃO DE CASO DE TESTE	
Identificador:	QR-TC-001
Objetivo:	Verificar geração correta de código QR após criação de reserva.
Autor(es)	Gonçalo de Lira Pereira
Especificação de Entradas • . Reserva criada com dados válidos (sala, equipamento, hora).	
Especificação de Saídas • QR Code gerado automaticamente e visível na interface da reserva. • O conteúdo codificado corresponde ao ID e dados da reserva.	
Outros QR Code gerado com biblioteca Flutter (qr_flutter).	
Dependências Reserva válida existente.	

Tabela A11. Caso de teste QR-TC-001: Verificação da geração correta de código QR após criação de reserva