

UNIVERSIDADE DE  
COIMBRA



## Programação Avançada em Java

Sistema de gestão de projetos Innovation Lab  
Management.

Tiago Sousa Caniceiro  
Vasco Azevedo Castro

Julho 2024

# Índice

---

Introdução.....	4
Descrição do problema e Análise de requisitos.....	5
Objetivo Geral: .....	7
Objetivos Específicos:.....	8
Glossário .....	9
Modelagem do sistema.....	10
Diagrama ER.....	10
Descrição de Entidades.....	10
Arquitetura do Sistema.....	15
Visão Geral.....	15
Componentes Principais.....	15
Fluxo de Dados.....	16
Tecnologias Utilizadas.....	17
Frontend.....	17
Backend.....	17
Base de dados.....	18
Implementação.....	19
Estrutura do Código: .....	19
Funcionalidades Implementadas: .....	20
Classes Associadas ao Utilizador.....	20
Funcionalidades de Projetos.....	21
Gestão de Tarefas no Projeto.....	24
Controlo de acesso: .....	26
Controlo de Acesso Baseado em Níveis de Acesso (RBAC).....	28
Implementação do RBAC no Sistema.....	28
Estrutura do RBAC.....	28
Benefícios do RBAC.....	28
Implementação e funcionalidade Gráfica.....	30
Testes e Validação.....	46
Testes com Postman.....	46
Testes com Mockito.....	46
Testes com Jest.....	47
Benefícios e Desafios.....	47
Desafios e soluções.....	48

Principais Desafios Encontrados.....	48
Conclusão.....	50

## Introdução

---

O objetivo deste relatório é documentar de forma detalhada e sistemática o desenvolvimento do projeto "Innovation Lab Management", realizado no âmbito da 11ª edição do Programa Acertar o Rumo da Universidade de Coimbra. O projeto curricular em questão é promovido pela Critical Software. O relatório visa não apenas descrever o processo de conceção e implementação de uma aplicação web, mas também analisar os desafios enfrentados e as soluções encontradas ao longo do projeto.

O presente relatório fornece um registo completo das atividades realizadas, tecnologias e arquitetura implementadas no projeto, documentando as decisões tomadas, as metodologias utilizadas e os resultados obtidos.

Através da documentação detalhada dos requisitos, da arquitetura, da implementação e dos testes, o relatório permite auxiliar a análise ao projeto, descrevendo os objetivos estabelecidos e o cumprimento dos requisitos técnicos e funcionais enunciados inicialmente. Ao documentar a arquitetura e as funcionalidades implementadas, é permitido que de uma forma mais detalhada todas as funções principais do sistema sejam descritas.

## Descrição do problema e Análise de requisitos

---

Este projeto tem como principal finalidade criar uma aplicação web robusta e intuitiva que facilite a gestão de um laboratório de Inovação. A aplicação visa proporcionar um ambiente centralizado para a organização, coordenação e acompanhamento de projetos inovadores, promovendo a colaboração entre os membros da equipa e otimizando a utilização dos recursos disponíveis.

A aplicação "Innovation Lab Management" tem como objetivo principal fornecer uma plataforma que permita gerir eficazmente todas as fases de um projeto inovador, desde a sua conceção até à sua conclusão. A plataforma foi concebida para suportar a interação entre os utilizadores, a partilha de conhecimentos e a administração eficiente dos recursos do laboratório, contribuindo para um ambiente de trabalho colaborativo e produtivo.

O âmbito do projeto abrange o desenvolvimento de diversas funcionalidades essenciais para a gestão eficiente de vários laboratórios de inovação. Estas funcionalidades estão organizadas em várias áreas-chave:

### Gestão de Utilizadores:

A gestão de utilizadores na aplicação inclui funcionalidades para o registo e autenticação, permitindo a criação e gestão de perfis personalizados. Cada utilizador pode registar as suas competências e interesses, o que facilita a formação de equipas com habilidades complementares. A aplicação distingue entre diferentes perfis de utilizador, como administradores, utilizadores padrão e utilizadores não autenticados, atribuindo permissões específicas a cada perfil. O administrador da plataforma tem permissões para gerir utilizadores, projetos e recursos, enquanto o utilizador padrão pode gerir a sua página pessoal, participar em projetos e colaborar na execução dos mesmos. Já o utilizador não autenticado tem permissão para registar-se no sistema e recuperar a palavra-passe.

A gestão de conta permite que utilizadores autenticados editem informações pessoais como nome, local de trabalho, fotografia, biografia e visibilidade da página pessoal. A página pessoal também permite que utilizadores vejam e editem as suas competências e interesses, além de listar os projetos em que participam.

### Gestão de Projetos:

A gestão de projetos na plataforma abrange a criação, edição e acompanhamento de projetos. Os utilizadores podem associar-se a projetos como gestores ou participantes, definir o estado dos projetos e gerir os recursos necessários para a sua execução. Esta funcionalidade permite uma documentação detalhada de cada projeto, facilitando a monitorização do progresso e a colaboração entre os membros da equipa.

A criação e edição de projetos permite aos utilizadores criar projetos, editar detalhes existentes e associar utilizadores, laboratórios, competências e recursos. A plataforma também possibilita a gestão do estado dos projetos, permitindo que estes passem por diferentes fases como "Planeamento", "Pronto", "Em Progresso", "Concluído" e "Cancelado". Além disso, há a funcionalidade de aprovação, onde os projetos podem ser aprovados ou rejeitados, garantindo uma gestão eficiente dos recursos e objetivos da organização.

Os utilizadores podem ver a lista de projetos a que estão associados e obter detalhes de cada projeto. Os gestores de projeto têm a capacidade de adicionar ou remover membros, enquanto os utilizadores podem solicitar participar em projetos. A gestão de projetos também inclui a definição de um plano de execução, especificando tarefas, responsáveis e estados das tarefas. Para facilitar a comunicação, cada projeto possui um chat interno acessível apenas aos seus membros. Além disso, é mantido um registo de atividades do projeto, incluindo alterações de estado e movimentações de membros, garantindo transparência e rastreabilidade.

### Gestão de Componentes/Recursos:

A aplicação inclui funcionalidades para a criação e gestão simplificada de componentes e recursos utilizados nos projetos. Os utilizadores podem adicionar novos componentes, editar informações existentes e pesquisar recursos disponíveis no laboratório. Esta gestão dos recursos garante que todos os projetos tenham acesso aos materiais e ferramentas necessários para o seu desenvolvimento.

### Comunicação e Colaboração:

Para promover a colaboração, a plataforma permite o envio e receção de mensagens pessoais entre utilizadores e inclui uma área de chat interno para cada projeto. Esta funcionalidade facilita a comunicação constante entre os membros da equipa, garantindo que todos estejam alinhados e possam colaborar efetivamente.

### Segurança e Qualidade:

O projeto implementa diversas medidas de segurança para proteger os dados dos utilizadores e garantir a integridade da aplicação. Entre essas medidas, destacam-se a validação rigorosa de inputs, a criação de logs detalhados das atividades, o armazenamento seguro de palavras-passe e a utilização de comunicação segura via HTTPS. Além disso, a aplicação foi desenvolvida para ser responsiva e suportar internacionalização, proporcionando uma experiência de uso satisfatória em diferentes dispositivos e idiomas.

No que diz respeito ao acesso e autenticação, a plataforma permite que novos utilizadores se registem através de e-mail e palavra-passe, com validação por e-mail para garantir a veracidade dos dados. Utilizadores registados podem fazer login utilizando as suas credenciais, criando uma sessão identificável. Há também a funcionalidade de terminar a sessão a qualquer

momento, garantindo que os utilizadores possam encerrar suas atividades com segurança. A recuperação e alteração de palavras-passe são facilitadas através do envio de um link por e-mail, assegurando que os utilizadores possam manter as suas contas seguras.

Para reforçar a segurança, foi implementado um mecanismo de término de sessão após um período de inatividade, configurável pelo administrador do sistema. As atividades no sistema são registadas em logs, incluindo operações de administradores, com data, hora, autor e IP de origem. As palavras-passe devem atender a bons padrões de segurança e são armazenadas de forma segura. O controlo de acesso é garantido apenas por informações armazenadas no servidor e associadas à sessão, assegurando que não haja dependência de informações provenientes do cliente. Toda a comunicação com o servidor é realizada via HTTPS ou WSS, protegendo os dados em trânsito.

Em termos de qualidade, a aplicação suporta pelo menos dois idiomas, permitindo a sua utilização por utilizadores de diferentes regiões. A responsividade da aplicação é garantida, funcionando adequadamente em dispositivos móveis, tablets e computadores de secretária. Sendo que algumas funcionalidades estão limitadas a dispositivos que não utilizam touchscreen. Implementou-se a funcionalidade de "soft delete" para tarefas, marcando-os como inativos ou excluídos sem removê-los fisicamente, facilitando a recuperação de dados, se necessário.

## Documentação e Testes:

Uma documentação completa do código e das funcionalidades é mantida, facilitando a manutenção e a evolução futura da aplicação. A aplicação é extensivamente testada através de testes unitários, de integração e de frontend para garantir a sua qualidade e funcionalidade.

## Enquadramento:

Para apoiar a gestão eficiente deste Laboratório de Inovação, surgiu a necessidade de uma aplicação web que permita organizar e acompanhar os diversos projetos, desde a sua conceção até à sua conclusão. Esta necessidade é impulsionada pela crescente complexidade dos projetos, pelo número de colaboradores envolvidos e pela variedade de recursos necessários. A aplicação visa facilitar a gestão dos projetos, a alocação de recursos, a comunicação entre os membros da equipe e o acompanhamento do progresso de cada iniciativa.

## Objetivo Geral:

O objetivo principal do projeto "Innovation Lab Management" é o design e a implementação de uma aplicação web que atenda às necessidades de organização e gestão dos projetos desenvolvidos no Laboratório de Inovação da Critical Software.

## Objetivos Específicos:

O sistema desenvolvido integra uma série de funcionalidades fundamentais para a gestão eficiente e segura de utilizadores, projetos e recursos. A gestão de utilizadores permite o registo, autenticação e administração de perfis, onde cada colaborador pode registar as suas competências e interesses, facilitando a atribuição de tarefas conforme suas habilidades. Na gestão de projetos, o sistema oferece ferramentas para criar e editar projetos, associando-os a utilizadores, laboratórios e recursos, e acompanhando o estado dos projetos ao longo de todo o ciclo de vida. A gestão de recursos é igualmente abrangente, permitindo a criação, edição e alocação de componentes e materiais necessários para os projetos, assegurando que o laboratório mantém uma gestão eficiente dos seus recursos.

Além disso, o sistema fomenta a comunicação e colaboração entre os membros da equipe, com ferramentas como envio de mensagens pessoais e chat interno específico para cada projeto, promovendo uma interação constante e eficaz. A segurança e a qualidade são prioridades, com medidas implementadas para proteger os dados dos utilizadores e garantir a integridade e a confidencialidade da aplicação. Finalmente, a manutenção de uma documentação completa e a realização de testes extensivos asseguram que a aplicação funciona corretamente e atende aos padrões de qualidade exigidos, proporcionando uma base sólida para o desenvolvimento contínuo e a melhoria do sistema.



## Glossário

---

**Skill:** Conhecimento, experiência ou habilidade específica que o utilizador possui, como programação em Java, design gráfico, entre outros. As skills podem ser categorizadas em várias áreas, como conhecimento (KNOWLEDGE), software (SOFTWARE), hardware (HARDWARE) e ferramentas (TOOLS).

**Interesse:** Temas, causas ou áreas de conhecimento pelas quais o utilizador tem interesse, como inteligência artificial, desenvolvimento sustentável, cyber-segurança, etc. Os interesses são classificados em tópicos (TOPICS), causas (CAUSES) e áreas de conhecimento (KNOWLEDGE\_AREAS).

**Ativos:** Itens físicos ou digitais que podem ser utilizados nos projetos, como sensores, controladores, licenças de software, entre outros. Os ativos são do tipo de recurso (RESOURCE) para itens não físicos e componentes para itens físicos (COMPONENT).

**Projeto:** Iniciativa desenvolvida no Laboratório de Inovação, envolvendo um ou mais utilizadores, e que pode passar por diferentes estados, como planeamento (PLANNING), pronto (READY), em progresso (IN\_PROGRESS), concluído (FINISHED) ou cancelado (CANCELLED).

**Laboratório:** Espaço físico onde os projetos são desenvolvidos, equipado com os recursos e componentes necessários.

**Gestor de Projeto (Project Manager):** Utilizador com permissões especiais para gerir um projeto, incluindo a capacidade de alterar o estado do projeto e alocar recursos. O papel de gestor de projeto é distinto de outros papéis, como utilizador normal.

**Participante:** Utilizador que faz parte de um projeto e contribui para a sua execução, mas sem as permissões de gestão atribuídas ao gestor de projeto.

**Administrador:** Utilizador com permissões elevadas que pode gerir a plataforma como um todo, incluindo a atribuição de permissões e a configuração de parâmetros de segurança. Os administradores têm um papel distinto em comparação com utilizadores padrão (STANDARD\_USER).

**Estado da Tarefa:** As tarefas dentro dos projetos podem estar em diferentes estados, como planeada (PLANNED), em progresso (IN\_PROGRESS) e concluída (FINISHED).



#### Configuration Entity:

Armazena pares chave-valor para configurações de sistema, sendo vital para a flexibilidade e eficiência na gestão de configurações críticas para o desempenho da aplicação e gerenciamento de utilizadores. No momento do arranque da aplicação, são criadas por default as configurações pedidas em enunciado: número máximo de participantes por projeto e o tempo máximo de sessão. Esta entidade garante que as configurações do sistema possam ser geridas e atualizadas de maneira centralizada e eficaz.

#### Message Entity:

É uma classe abstrata que serve como base para mensagens individuais e em grupo. Ela armazena informações comuns a todos os tipos de mensagens, como conteúdo, remetente, tempo de envio e se foi visualizada. A herança desta entidade permite a reutilização de atributos e métodos comuns, promovendo uma estrutura de código limpa e eficiente.

#### Group Message Entity:

Representa mensagens enviadas em grupo dentro de um projeto, facilitando a comunicação entre membros do projeto. Cada mensagem está associada a um projeto específico e possui informações sobre o remetente, conteúdo, tempo de envio e se foi visualizada. As relações com Notification Entity e User Entity permitem notificar utilizadores e rastrear quem leu a mensagem, garantindo uma comunicação clara e rastreável.

#### Individual Message Entity:

Representa mensagens enviadas entre utilizadores individuais, possibilitando uma comunicação direta e privada. Cada mensagem possui um remetente, destinatário, conteúdo, assunto, tempo de envio e se foi visualizada. A relação com Notification Entity permite a notificação do destinatário sobre novas mensagens, melhorando a interação e comunicação dentro do sistema.

#### Interest Entity:

Armazena interesses de utilizadores, categorizados como causas, áreas de conhecimento ou tópicos, ajudando a personalizar a experiência do utilizador e conectar pessoas com interesses semelhantes. A relação com User Entity facilita a associação de interesses a utilizadores específicos, promovendo a colaboração e engajamento baseado em interesses comuns.

#### Keyword Entity:

Armazena palavras-chave utilizadas para categorizar projetos, facilitando a busca e organização dos mesmos. A relação com Project Entity permite associar palavras-chave a projetos, melhorando a filtragem e a descoberta de projetos relevantes para os utilizadores.

#### Laboratory Entity:

Representa os laboratórios onde os projetos são desenvolvidos, categorizados por localização. Esta entidade é essencial para organizar os projetos por local e facilitar a alocação de recursos físicos e humanos. As relações com User Entity e Project Entity permitem associar utilizadores e projetos a laboratórios específicos, promovendo uma gestão eficiente e localizada.

#### Method Entity:

Define os métodos ou ações que podem ser executados dentro do sistema, categorizados por nome e descrição. Esta entidade é fundamental para implementar e controlar a lógica de negócios do sistema, garantindo que cada ação esteja devidamente registada e gerenciada. A relação com Role Entity permite associar métodos a diferentes papéis de utilizadores, controlando assim o acesso e as permissões.

#### Notification Entity:

Gere as notificações enviadas aos utilizadores, que podem ser relacionadas a mensagens individuais, mensagens em grupo, projetos ou tarefas. Cada notificação contém informações sobre o tipo, conteúdo, data e se foi lida. Esta entidade é crucial para manter os utilizadores informados sobre eventos importantes, garantindo que eles não percam atualizações relevantes.

#### Project Asset Entity:

Corresponde à alocação de ativos para projetos específicos, permitindo rastrear a quantidade de cada ativo utilizado num projeto. Esta entidade facilita a gestão eficiente de recursos, garantindo que os projetos tenham os materiais necessários para seu progresso. A relação com Asset Entity e Project Entity assegura que cada ativo esteja corretamente associado ao projeto relevante.

#### Project Entity:

É central para o sistema, representando projetos. Ela armazena informações como nome, descrição, motivação, estado, datas relevantes e se foi aprovado. As diversas relações com outras entidades, como User Entity, Task Entity, Skill Entity, Keyword Entity e Project Asset Entity, permitem uma gestão completa e integrada dos projetos, garantindo que todos os aspetos sejam considerados e geridos adequadamente.

#### Project Log Entity:

Regista logs de atividades e mudanças ocorridas nos projetos, armazenando informações sobre a data de criação, tipo de log, conteúdo, utilizador responsável e projeto associado. Esta entidade é essencial para manter um histórico detalhado de todas as ações realizadas dentro dos projetos, facilitando auditorias e a rastreabilidade das atividades.

#### Project Membership Entity:

Representa a associação de utilizadores a projetos, incluindo informações sobre o papel do utilizador no projeto, se a associação foi aceite e um token de aceitação. Esta entidade é crucial para controlar a participação dos utilizadores nos projetos, garantindo que somente membros autorizados tenham acesso e possam contribuir para os projetos.

#### Role Entity:

Define os diferentes papéis que os utilizadores podem ter dentro do sistema, como administrador ou utilizador padrão. Cada papel possui um conjunto de métodos associados que determinam as ações que o utilizador pode executar. Esta entidade é fundamental para implementar o controle de acesso baseado em papéis, garantindo que os utilizadores tenham permissões apropriadas para suas funções.

#### SessionEntity:

Representa as sessões de utilizador, incluindo tokens de autenticação, tokens de sessão, data de expiração e estado da sessão. Esta entidade é essencial para a segurança e gestão de sessões, garantindo que os utilizadores autenticados mantenham as suas sessões de maneira segura e controlada.

#### Skill Entity:

Armazena as habilidades dos utilizadores e projetos, categorizadas por tipo, como hardware, conhecimento, software e ferramentas. Esta entidade é importante para mapear as competências dos utilizadores e requisitos dos projetos, facilitando a alocação de recursos humanos com as habilidades necessárias para o sucesso dos projetos.

#### Task Entity:

Representa as tarefas dentro dos projetos, incluindo título, descrição, datas de criação e conclusão, estado e relações com utilizadores e projetos. Esta entidade é vital para o planeamento e execução dos projetos, garantindo que todas as tarefas sejam rastreadas e geridas de forma eficiente. As relações com User Entity e Project Entity permitem associar tarefas a responsáveis e projetos específicos, promovendo uma gestão detalhada e organizada das atividades.

#### User Entity:

Representa os utilizadores do sistema, armazenando informações como email, palavra-passe, nome de utilizador, foto, biografia e privacidade. Esta entidade é central para o sistema, pois define os utilizadores que interagem com todas as outras entidades e funcionalidades. As relações com Laboratory Entity, Role Entity, Project Membership Entity, Session Entity, Task Entity, Skill Entity, Interest Entity, Notification Entity, Message Entity e Project Log Entity permitem uma integração completa e detalhada dos utilizadores em todos os aspetos do sistema, promovendo uma experiência de utilizador rica.

# Arquitetura do Sistema

A arquitetura do sistema é projetada para ser robusta, eficiente e escalável, garantindo uma comunicação eficaz entre o frontend, o backend e a base de dados.

## Visão Geral

O sistema é dividido em três camadas principais: o frontend, o backend e a base de dados. A comunicação entre essas camadas é realizada através de requisições e respostas Json Https, garantindo segurança e integridade dos dados.

## Componentes Principais

- Frontend: Implementado com HTML, CSS, JavaScript e React.js, o frontend é responsável pela interface do utilizador, fornecendo uma experiência interativa e responsiva. Ele envia requisições JSON ao backend e recebe respostas JSON para renderizar a interface dinâmica.
- Backend: Construído com Jakarta EE, JAX-RS, e EJBs, o backend é dividido em Camada de Negócios (Business Layer) e Camada de Persistência (Persistence Layer). A camada de negócios processa a lógica de aplicação, enquanto a camada de persistência é responsável pela interação com a base de dados.
- Base de dados: Um base de dados relacional que armazena entidades persistidas, como utilizadores, projetos, mensagens e notificações. A interação com a base de dados é feita através dos DAOs na camada de persistência.

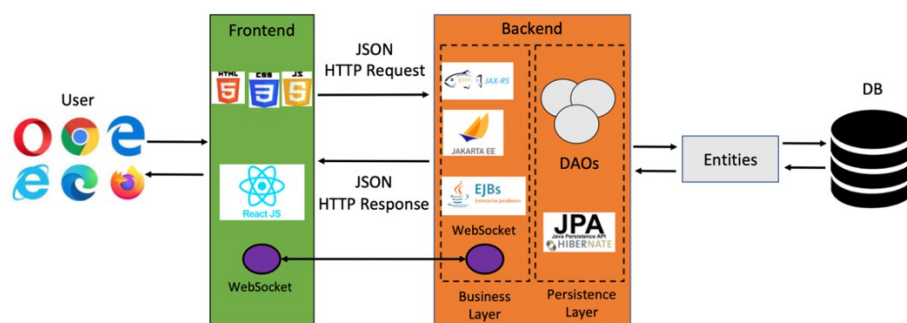


Fig.2 - Arquitetura da aplicação.

## Fluxo de Dados

O fluxo de dados no sistema começa com o utilizador interagindo com a interface no frontend. Ao realizar uma ação, como o login ou a criação de um novo projeto, o frontend transforma essas interações em requisições JSON, que são então enviadas para o backend através de uma conexão HTTPS. Esta conexão segura garante que os dados transmitidos entre o cliente e o servidor são protegidos contra interceções maliciosas.

Quando a requisição JSON chega ao backend, ela é recebida pela camada de negócios, onde é validada e processada de acordo com a lógica de aplicação. Por exemplo, uma requisição de login envolverá a verificação das credenciais do utilizador com os dados armazenados na base de dados. Se as credenciais forem válidas, a camada de negócios gerará um token JWT para autenticação e o enviará de volta ao frontend. Caso contrário, uma mensagem de erro apropriada será retornada.

Para operações que envolvem manipulação de dados persistentes, a camada de negócios interage com a camada de persistência. Aqui, os DAOs (Data Access Objects) desempenham um papel crucial, executando consultas SQL para recuperar ou modificar dados na base de dados relacional. Por exemplo, ao criar um projeto, o backend salvará os detalhes do projeto na base de dados e retornará uma confirmação ao frontend, incluindo quaisquer dados adicionais necessários para a interface do utilizador.

Depois de processar a requisição e realizar as operações necessárias na base de dados, a camada de negócios do backend prepara uma resposta JSON que é enviada de volta ao frontend. Esta resposta contém todos os dados necessários para atualizar a interface do utilizador, refletindo as mudanças feitas ou fornecendo os resultados das ações solicitadas. O frontend então utiliza esses dados para renderizar a interface de forma dinâmica e interativa, proporcionando ao utilizador uma experiência fluida e responsiva.

Ao longo de todo este processo, a arquitetura do sistema assegura que cada camada desempenha o seu papel específico de forma eficiente e segura. O frontend cuida da interação com o utilizador, o backend processa a lógica de negócio e garante a integridade das operações, e a base de dados armazena os dados de forma persistente e estruturada. Este fluxo de dados bem definido e estruturado é fundamental para garantir a escalabilidade, segurança e eficiência do sistema como um todo.



# Tecnologias Utilizadas

---

## Frontend

No desenvolvimento do frontend, utilizamos o React.js como a biblioteca principal para a construção de interfaces de utilizador. Esta escolha permitiu-nos criar componentes reutilizáveis e de alta performance, assegurando uma experiência de utilizador dinâmica e responsiva. Através do React Router, conseguimos gerir o roteamento na aplicação, facilitando a navegação entre diferentes páginas de forma eficiente e sem a necessidade de recarregar completamente a página, proporcionando uma transição suave para os utilizadores.

Para a estilização dos componentes, recorremos ao CSS, permitindo uma personalização detalhada e estética dos elementos da interface. A biblioteca React Intl foi integrada para a internacionalização, garantindo que a aplicação pudesse suportar múltiplos idiomas e culturas, o que é essencial para assegurar a acessibilidade global e uma experiência de utilizador inclusiva.

A comunicação em tempo real entre o frontend e o backend foi viabilizada através de WebSockets. Esta tecnologia permitiu-nos oferecer atualizações instantâneas e interativas na interface do utilizador, melhorando significativamente a experiência de utilização em aplicações que requerem uma troca de dados contínua e rápida.

Para enriquecer a interface do utilizador, utilizamos diversas ferramentas de UI e bibliotecas. FontAwesome foi empregue para a adição de ícones intuitivos, enquanto React-Select foi utilizado para criar listas suspensas e configuráveis. A seleção de datas foi facilitada com o React-Datepicker, oferecendo uma interface amigável e intuitiva para os utilizadores. Além disso, bibliotecas como o Recharts foram utilizadas para a visualização de dados, permitindo a apresentação de gráficos e visualizações de forma clara e eficiente. Esta combinação de ferramentas e bibliotecas contribuiu para a criação de uma interface de utilizador robusta, funcional e visualmente apelativa.

## Backend

Na camada de backend, utilizamos a plataforma Jakarta EE, que fornece um conjunto abrangente de APIs e especificações para o desenvolvimento de aplicações corporativas robustas e escaláveis. Esta plataforma facilita a criação de soluções confiáveis, com um foco particular na modularidade e na facilidade de manutenção.

A API JAX-RS foi fundamental para a construção de serviços web RESTful, permitindo a comunicação eficiente entre o frontend e o backend através de requisições HTTPS. Isso possibilitou a criação de uma interface de programação de aplicações (API) clara e estruturada, que é essencial para a integração dos diferentes componentes do sistema.

Os Enterprise JavaBeans (EJBs) foram utilizados para encapsular a lógica de negócios da aplicação, garantindo que as transações fossem geridas de maneira segura e eficiente. Os EJBs proporcionam uma infraestrutura robusta para a implementação de funcionalidades

empresariais, assegurando a integridade e a consistência dos dados através de transações bem-definidas.

Para a gestão da base de dados foi utilizado Hibernate. Esta tecnologia facilitou a interação com a base de dados, permitindo que as entidades Java fossem mapeadas diretamente para as tabelas da base de dados, simplificando as operações de persistência de dados e assegurando uma integração perfeita entre a camada de negócios e a camada de persistência.

Finalmente, para garantir a qualidade e a robustez do código, empregámos os frameworks de testes JUnit e Mockito. JUnit foi utilizado para a realização de testes unitários, assegurando que cada componente do sistema funcionasse corretamente de forma isolada. Mockito, por sua vez, foi utilizado para mockar dependências, facilitando a criação de testes abrangentes e permitindo a validação de componentes de forma isolada, sem a necessidade de interações reais com outros componentes do sistema. Este enfoque nos testes automatizados foi crucial para assegurar a qualidade e a confiabilidade do sistema como um todo.

## Base de dados

Para o armazenamento e gestão dos dados da aplicação, optámos pelo sistema de gestão de base de dados relacional MySQL. Este sistema é utilizado para armazenar informações cruciais como dados de utilizadores, projetos, mensagens e notificações, garantindo integridade e performance. A interação com a base de dados é encapsulada através do padrão de design Data Access Objects (DAOs). Os DAOs isolam a camada de persistência da lógica de negócios, permitindo uma interface clara e consistente para operações de base de dados, facilitando a manutenção e evolução do sistema sem impactar a lógica de negócios.

# Implementação

---

## Estrutura do Código:

A estrutura do código do sistema é organizada de maneira a facilitar a manutenção, a escalabilidade e a colaboração entre desenvolvedores. No frontend, utilizamos a biblioteca React para construir uma interface de utilizador dinâmica e responsiva. O código fonte do frontend está organizado na pasta `src`, que contém todos os arquivos necessários para o funcionamento da aplicação. Dentro desta pasta, temos várias subpastas que abrigam componentes, serviços, utilitários, stores, e outros recursos.

Na raiz do projeto frontend, encontramos o arquivo `App.jsx`, que é o componente principal da aplicação e serve como ponto de entrada para a maioria dos componentes e funcionalidades. A pasta `assets` contém recursos estáticos como imagens e arquivos de média, que são utilizados em várias partes da aplicação para melhorar a experiência do utilizador.

A pasta `components` é onde são armazenados todos os componentes reutilizáveis do sistema. Estes componentes são organizados em subpastas específicos baseados na funcionalidade ou na página onde são utilizados. Por exemplo, temos subpastas como `HomePageComponents`, `InventoryComponents`, `MessagesPageComponents`, `ProjectPageComponents`, entre outros. Cada subpasta contém os componentes JSX e seus respectivos arquivos CSS em módulos para estilização.

Para a gestão do estado da aplicação, utilizamos a biblioteca Zustand, e o código relacionado a isso está localizado na pasta `stores`. Aqui, definimos vários hooks personalizados para gerenciar estados globais e compartilhar dados entre diferentes componentes da aplicação de maneira eficiente.

Os serviços que encapsulam chamadas a APIs externas ou outras operações assíncronas estão armazenados na pasta `services`. Esta pasta inclui arquivos como `userService.jsx`, `projectService.jsx`, `taskService.jsx`, entre outros. Cada serviço é responsável por um conjunto específico de operações relacionadas a uma entidade ou funcionalidade do sistema.

Os utilitários e helpers que fornecem funções auxiliares são armazenados na pasta `utils`. Esta pasta pode conter arquivos para validação de dados, configuração de constantes, cores utilizadas na aplicação, e outras funcionalidades que suportam o desenvolvimento.

Além disso, a configuração de internacionalização, que permite a tradução da aplicação para diferentes idiomas, está localizada na pasta `translations`. Esta pasta contém arquivos JSON com as traduções para diferentes idiomas.

Para a gestão de pacotes e dependências no frontend, utilizamos o npm (Node Package Manager). O arquivo `package.json` na raiz do projeto define todas as dependências e scripts necessários para a construção, teste e execução da aplicação. Este arquivo é crucial para manter as bibliotecas e ferramentas do projeto atualizadas e gerenciadas de forma eficiente.

No backend, utilizamos o Maven como ferramenta de automação de construção e gestão de dependências. O código fonte do backend está organizado na pasta `src/main/java`, onde seguimos uma estrutura de pacotes clara e modular. Dentro desta pasta, temos pacotes como `bean`, `dao`, `dto`, `entity`, `service`, `utils`, e `websocket`. Cada pacote contém classes que implementam funcionalidades específicas, promovendo a separação de preocupações e facilitando a manutenção do código.

O arquivo `pom.xml` localizado na raiz do projeto backend é o ponto central de configuração do Maven. Ele define as dependências do projeto, plugins de construção, e outras configurações necessárias para compilar e empacotar a aplicação. Com o Maven, podemos facilmente adicionar novas dependências, executar testes automatizados, e gerar builds consistentes.

Essa organização clara e modular tanto no frontend quanto no backend, aliada ao uso de ferramentas como npm e Maven, garante que o desenvolvimento seja eficiente, escalável e de fácil manutenção.

## Funcionalidades Implementadas:

---

### Classes Associadas ao Utilizador

Registo de Utilizador:

A funcionalidade de Registo de utilizador permite a criação de novas contas no sistema. A classe `UserBean` é responsável por implementar esta funcionalidade. Durante o processo de Registo, a aplicação verifica se o email ou o nome de utilizador já existem, codifica a palavra-passe do utilizador, atribui um papel padrão e laboratório, define parâmetros padrão para o novo utilizador, gera um token de confirmação e envia um email de confirmação. A integração com a base de dados é feita através do uso da API JPA, e a segurança da palavra-passe é garantida utilizando a biblioteca `BCrypt`.

Confirmação de Registo:

Após o registo, os utilizadores devem confirmar as suas contas usando um token de confirmação enviado por email. A classe `UserBean` valida o token, verifica se ele não está expirado e confirma o registo do utilizador, marcando-o como confirmado na base de dados.

Redefinição da Palavra-passe:

A funcionalidade de redefinição de palavra-passe permite que os utilizadores solicitem a redefinição da sua palavra-passe caso a tenham esquecido. O `UserBean` valida a existência do utilizador, verifica se uma solicitação de redefinição recente já foi feita, gera um novo token de redefinição e envia um email com instruções para redefinir a palavra-passe. Quando o utilizador segue as instruções e envia um novo token e palavra-passe, o sistema valida o token e atualiza a palavra-passe na base de dados.

#### Solicitação de Novo Email de Confirmação:

Caso um utilizador não tenha recebido o email de confirmação ou o tenha perdido, ele pode solicitar um novo email de confirmação. O UserBean verifica se o utilizador já está confirmado, se não, verifica se não há um intervalo muito curto desde a última solicitação de email de confirmação, e envia um novo email de confirmação.

#### Atualização do Perfil de Utilizador:

Utilizadores autenticados podem atualizar o seu perfil, incluindo nome, foto, biografia e configurações de privacidade. UserBean recupera o utilizador autenticado, valida as novas informações e atualiza o perfil na base de dados.

#### Obtenção de Informações Básicas do Utilizador:

Para permitir a visualização de perfis de utilizadores, o UserBean fornece métodos para obter informações básicas sobre um utilizador autenticado ou qualquer outro utilizador, dado o seu nome de utilizador.

#### Gestão de Sessões de Utilizador:

A classe SessionBean é responsável pela gestão de sessões de utilizadores, incluindo autenticação, validação de tokens JWT, criação e invalidação de sessões, e limpeza de sessões expiradas. Quando um utilizador faz login, a aplicação valida suas credenciais, gera tokens JWT para autenticação e sessão, e persiste a sessão na base de dados. Tokens expirados são automaticamente invalidados para garantir a segurança do sistema.

#### Login e Logout de Utilizadores:

A funcionalidade de login autentica os utilizadores e inicia uma nova sessão, enquanto a funcionalidade de logout invalida a sessão atual, garantindo que tokens não utilizados não permaneçam válidos. O SessionBean gerencia esses processos utilizando JWT para segurança de tokens e cookies para armazenamento seguro no navegador do cliente.

## Funcionalidades de Projetos

#### Criação de Projetos

A criação de projetos é gerida pela classe ProjectBean, que permite a criação de novos projetos, associando-os a utilizadores, habilidades, palavras-chave e ativos. O processo envolve a validação das datas e palavras-chave, a verificação da existência de duplicatas de nomes de projetos, a persistência do novo projeto na base de dados e a definição das relações com os utilizadores e outros elementos. Logs de atividades do projeto são criados para auditoria.

### Atualização de Projetos

A classe `ProjectBean` também gere a atualização de projetos existentes. Este processo inclui a validação das novas informações, a verificação de duplicados a atualização das relações de habilidades, keywords e ativos, e ainda a transição de estado do projeto conforme necessário. Logs de mudanças são criados para documentar as atualizações e garantir a rastreabilidade.

### Aprovação e Rejeição de Projetos

A aprovação ou rejeição de projetos é realizada pela classe `ProjectBean`. A aplicação verifica o estado atual do projeto, permite que administradores aprovem ou rejeitem projetos prontos, e atualiza o estado do projeto conforme a decisão. Notificações são enviadas para todos os membros do projeto para informá-los sobre a decisão tomada.

### Gestão de Membros do Projeto

A classe `MembershipBean` gerencia a adição e remoção de membros em projetos. Os utilizadores podem solicitar adesão a um projeto ou serem convidados por gestores de projetos. As solicitações e convites são geridos através de tokens de aceitação e notificações. A aceitação ou rejeição das solicitações são processadas com criação de logs e envio de notificações adequadas. Além disso, a remoção de membros de projetos inclui a reassociação de tarefas e a criação de logs para auditoria.

### Recuperação de Detalhes de Projetos

A aplicação permite a recuperação dos detalhes de projetos específicos através da classe `ProjectBean`. Esta funcionalidade inclui a conversão de entidades de projetos em DTOs (Data Transfer Objects) para facilitar a apresentação das informações no frontend. Os detalhes incluem informações básicas do projeto, como nome, descrição, motivação, estado, datas relevantes, membros e logs de atividades.

### Listagem de Projetos

A listagem de todos os projetos ou de projetos filtrados por critérios específicos é facilitada pela classe `ProjectBean`. Esta funcionalidade inclui a paginação e a conversão de entidades de projetos em DTOs para facilitar a exibição das informações.

### Estados e Papéis de Projetos

A aplicação fornece funcionalidades para recuperar listas de estados possíveis de projetos e papéis dentro de projetos. Estas listas são úteis para a construção de interfaces de utilizador, permitindo que utilizadores selecionem estados ou papéis ao gerirem projetos.

### Logs de Projetos

Os logs de atividades dos projetos são geridos pela classe ProjectBean. Cada mudança significativa nos projetos, como atualizações de detalhes, mudanças de estado ou adições/remoções de membros, é registada com um log correspondente. Estes logs são utilizados para auditoria e rastreamento das atividades do projeto.

### Notificações

Notificações relacionadas a projetos, como aprovações, rejeições, convites e remoções de membros, são geridas pela classe NotificationBean. Esta funcionalidade garante que todos os membros do projeto são mantidos informados sobre as mudanças e decisões importantes relacionadas aos projetos nos quais estão envolvidos.

## Gestão de Tarefas no Projeto

### Recuperação de Tarefas por Projeto

A classe `TaskBean` permite a recuperação de todas as tarefas associadas a um projeto específico. O método `getTasksByProject` valida o ID do projeto, verifica a existência do projeto na base de dados, e apresenta as tarefas associadas a esse projeto. As tarefas são convertidas para DTOs (`TaskGetDto`) para facilitar a manipulação e apresentação na camada de aplicação.

### Recuperação de Detalhes de Tarefas

Através do método `getTasksById`, é possível recuperar os detalhes de uma tarefa específica usando seu ID. O método valida o ID, verifica a existência da tarefa, e converte a entidade da tarefa para um DTO (`TaskGetDto`).

### Adição de Novas Tarefas

O método `addTask` permite adicionar novas tarefas a um projeto. Este método realiza várias validações, incluindo a existência do projeto e do utilizador responsável, a validação das datas planeadas e a verificação se o utilizador responsável é membro do projeto. Uma nova entidade de tarefa é criada e persistida na base de dados, e uma notificação é gerada para o utilizador responsável pela nova tarefa.

### Adição e Remoção de Dependências entre Tarefas

As dependências entre tarefas podem ser geridas através dos métodos `addDependencyTask` e `removeDependencyTask`. Estes métodos validam a existência das tarefas principais e dependentes, verificam se as tarefas não foram excluídas, e se pertencem ao mesmo projeto. Em seguida, a relação de dependência é criada ou removida conforme necessário.

### Atualização de Tarefas

A atualização de tarefas é gerida pelo método `updateTask`, que realiza validações dos dados atualizados, gere transições de estado das tarefas, e atualiza os detalhes da tarefa conforme necessário. O método `taskDetailedUpdate` permite atualizações detalhadas, incluindo a validação de datas planeadas, utilizadores responsáveis, executores registados e mudanças de estado. As notificações são enviadas aos utilizadores responsáveis e executores em caso de mudanças.

### Exclusão de Tarefas

O método `deleteTask` permite excluir uma tarefa identificada pelo seu ID. Este método valida a existência do utilizador autenticado, da tarefa, e da associação ao projeto. As dependências da tarefa são removidas e a tarefa é marcada como excluída.



### Conversão de Entidades de Tarefas

Os métodos `convertTaskEntityToTaskDto` e `convertTaskEntityListToTaskDtoList` são responsáveis por converter entidades de tarefas para DTOs (`TaskGetDto`), mapeando campos básicos, utilizadores responsáveis, executores registados, pré-requisitos e tarefas dependentes.

### Outras Classes e funcionalidades:

A `SkillBean` é responsável pela gestão das habilidades dos utilizadores e projetos. Inclui métodos para adicionar e remover habilidades, tanto para utilizadores como para projetos, garantindo que não ocorram duplicações. Esta classe também oferece métodos para recuperar todas as habilidades, habilidades específicas de utilizadores ou projetos, e habilidades que começam com uma letra específica.

A `LaboratoryBean` trata da gestão de laboratórios. Esta classe permite criar laboratórios se não existirem, e recuperar todos os laboratórios existentes.

A `KeywordBean` foca-se na gestão das palavras-chave associadas aos projetos. Inclui métodos para adicionar e remover palavras-chave dos projetos, garantindo que as palavras-chave não sejam duplicadas ou removidas incorretamente. Também permite recuperar todas as palavras-chave, bem como aquelas associadas a projetos específicos ou que começam com uma letra específica. Esta classe realiza a conversão das entidades de palavra-chave para DTOs (`KeywordGetDto`), facilitando a visualização e manipulação dos dados.

A `InterestBean` é utilizada para gerir os interesses dos utilizadores. Permite adicionar e remover interesses, assegurando que não há duplicados. A recuperação de interesses pode ser feita para todos os interesses, para interesses de utilizadores específicos, ou para interesses que começam com uma letra específica. A conversão das entidades de interesse para DTOs (`InterestGetDto`) também é realizada, permitindo uma gestão eficiente dos dados de interesse.

Por fim, a `AssetBean` trata da gestão dos ativos do sistema. Inclui funcionalidades para criar ativos, adicionar ativos a projetos, remover ativos, e atualizar detalhes dos ativos existentes. Também permite recuperar todos os ativos ou filtrar ativos com base em critérios específicos, convertendo as entidades de ativo para DTOs (`AssetGetDto`).

Estas classes, através das suas diversas funcionalidades, garantem uma gestão integrada e eficiente dos diferentes componentes do sistema, desde recursos humanos e materiais até

dados específicos de projetos. A integração cuidadosa dessas funcionalidades é crucial para a operação fluida e bem-organizada do sistema.

## Controlo de acesso:

---

O sistema de controlo de acesso no projeto é arquitetado de forma a garantir a segurança e a autorização adequada dos utilizadores, utilizando diversas tecnologias e práticas do Jakarta EE e Java. As principais classes e métodos relacionados a este controlo são o `AuthorizationFilter` e o `ResponseCookieFilter`, responsáveis por filtrar e modificar as requisições e respostas HTTPS, além de várias anotações customizadas para a verificação de permissões.

A arquitetura do controlo de acesso é centrada em filtros que interceptam as requisições HTTPS, validam tokens de autenticação, configuram contextos de segurança e verificam as permissões dos utilizadores. O `AuthorizationFilter` intercepta todas as requisições HTTPS que chegam ao servidor, extrai e valida o token de autenticação presente nos cookies das requisições. Se o token está prestes a expirar é renovado para manter a sessão do utilizador ativa e segura. Este filtro também configura o contexto de segurança com detalhes específicos do utilizador autenticado, como seu ID e permissões, e verifica se o utilizador tem as permissões necessárias para aceder ao recurso solicitado.

Os serviços de acesso público, que não requerem autenticação, incluem endpoints que permitem o acesso a funcionalidades essenciais sem a necessidade de validação do utilizador. Estes serviços abrangem funcionalidades como o registo de novos utilizadores, pedidos de recuperação de senha, envio de e-mails de confirmação, e acesso a informações básicas de projetos e laboratórios. Esses endpoints são identificados e geridos pelo sistema de controlo de acesso para garantir que qualquer requisição a essas funcionalidades não requer uma autenticação facilitando assim a interação inicial dos utilizadores com o sistema de forma segura e eficiente.

O `ResponseCookieFilter` modifica a resposta HTTP para incluir novos cookies de autenticação e sessão, se novos tokens forem gerados durante o processamento da requisição.

O sistema utiliza várias entidades principais, como o `AuthUserDto`, que é um Data Transfer Object contendo detalhes do utilizador autenticado, como ID, nome de utilizador e permissões. A `RoleEntity` representa os diferentes papéis que um utilizador pode ter no sistema, enquanto a `MethodEntity` representa os diferentes métodos (ações) que podem ser protegidos por permissões. A `ProjectMembershipEntity` representa a associação de utilizadores a projetos e seus respetivos papéis nesses projetos.

Os beans e DAOs são componentes essenciais no gerenciamento de dados e lógica de negócios. O `SessionBean` gere sessões de utilizadores, incluindo a validação e renovação de tokens. O `ConfigurationBean` lida com as configurações do sistema, como o tempo de expiração de sessões. O `ProjectMembershipDao` acede e manipula dados relacionados à associação de utilizadores a projetos, enquanto o `UserBean` gerencia os dados dos utilizadores.

Além disso, o sistema utiliza várias anotações personalizadas para verificar permissões. A anotação `RequiresMethodPermission` indica que o método/serviço anotado requer uma permissão específica. A anotação `RequiresProjectRolePermission` indica que o utilizador precisa de ter um grau de permissão específico num projeto para aceder ao método. A anotação `RequiresProjectMemberPermission` indica que o utilizador precisa ser membro de um projeto para aceder ao método. As anotações `RequiresPermissionByUserOnIndividualMessage` e `RequiresPermissionByUserOnIndividualMessageAllMessages` verificam se o utilizador é o remetente ou destinatário de uma mensagem individual ou tem permissão para aceder a mensagens, geralmente pelo ID do utilizador.

Os principais métodos no `AuthorizationFilter` incluem o método `filter`, que processa requisições, valida tokens, configura contextos de segurança e verifica permissões. O método `handleTokenRenewal` verifica e renova tokens de autenticação, se necessário. O método `setupThreadContext` configura o contexto de log para a requisição atual, incluindo detalhes do utilizador. O método `setSecurityContext` configura o contexto de segurança para a requisição atual, e o método `checkAuthorization` verifica se o utilizador tem as permissões necessárias para aceder ao recurso solicitado. Finalmente, o método `abortUnauthorized` aborta a requisição com um status de não autorizado se a validação ou verificação de permissões falhar.

Este sistema assegura que apenas utilizadores autorizados possam aceder a recursos protegidos, aplicando uma robusta verificação de permissões em várias camadas do sistema.

# Controlo de Acesso Baseado em Níveis de Acesso (RBAC)

---

## Implementação do RBAC no Sistema

A implementação do controlo de acesso no sistema segue o modelo RBAC (Role-Based Access Control), um paradigma amplamente reconhecido e eficaz para a gestão de permissões. Este modelo associa permissões a níveis de acesso específicos e atribui esses papéis aos utilizadores, facilitando a administração e a manutenção da segurança do sistema.

## Estrutura do RBAC

### **RoleEntity:**

Representa os diferentes papéis dentro do sistema, tais como ADMIN e STANDARD\_USER.

Cada papel tem associado um conjunto de métodos que define as permissões disponíveis para os utilizadores com esse papel.

### **MethodEntity:**

Representa as diversas operações ou funcionalidades que podem ser executadas no sistema.

Cada método tem um identificador único e uma descrição detalhada.

### **Tabela de Relação Role-Method:**

Esta tabela relaciona os papéis (RoleEntity) com os métodos (MethodEntity).

Define explicitamente quais os métodos (permissões) que são atribuídos a cada papel, permitindo uma gestão centralizada e clara das permissões.

## Benefícios do RBAC

### Gestão Simplificada de Permissões:

**Centralização:** As permissões são geridas centralmente através dos papéis. Isto significa que para alterar as permissões de um grupo de utilizadores, basta atualizar o papel correspondente.

### Facilidade de Atribuição:

Novos utilizadores podem ser rapidamente atribuídos aos papéis apropriados, sem necessidade de configurar individualmente cada permissão.

### Segurança Melhorada:

**Redução de Erros:** Atribuir permissões através de papéis minimiza o risco de erros administrativos que podem ocorrer ao gerir permissões diretamente para cada utilizador.

**Princípio do Menor Privilégio:** Utilizadores recebem apenas as permissões necessárias para realizar as suas funções, reduzindo o risco de acesso não autorizado.

#### Flexibilidade e Escalabilidade:

Facilidade de Atualização: Novas funcionalidades podem ser adicionadas ao sistema e associadas a papéis existentes através da tabela de relação, sem necessidade de reconfigurar permissões individuais.

#### Adaptação às Mudanças:

À medida que a organização cresce ou os requisitos mudam, novos papéis podem ser criados ou existentes podem ser modificados facilmente, garantindo que o sistema permaneça alinhado com as necessidades organizacionais.

A nossa implementação do RBAC, utilizando RoleEntity e MethodEntity e uma tabela de relação, proporciona um sistema de controlo de acesso robusto e flexível. Esta abordagem não só simplifica a administração e a escalabilidade do sistema, como também aumenta significativamente a segurança, garantindo que os utilizadores apenas têm acesso às funcionalidades necessárias para as suas funções

## Implementação e funcionalidade Gráfica

Ao entrar na aplicação, a primeira visualização contém no topo a secção de registo ou entrar na aplicação. No canto superior direito é ainda possível escolher a língua de navegação.

Na parte central tem um vídeo de apresentação da empresa e a lista de laboratórios, com um link de navegação para os projetos em curso em cada laboratório e um rodapé com informações gerais e contatos.

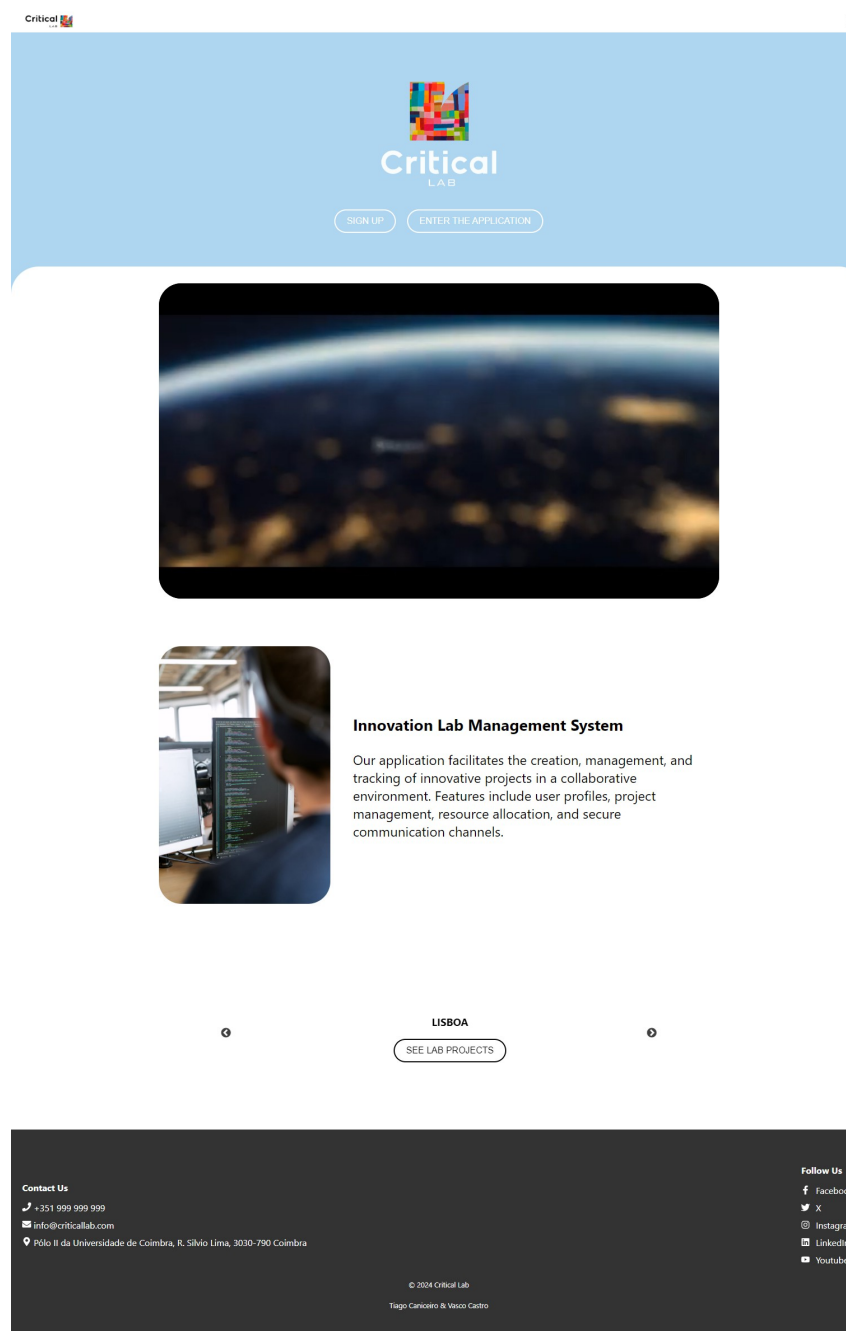


Fig.3 - Página de abertura da aplicação.

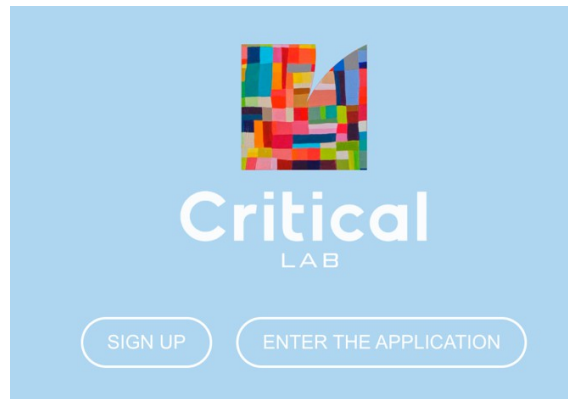


Fig.4 - Detalhe da secção de registo ou início de sessão.



Fig. 5- Detalhes da escolha da língua de navegação e da navegação entre laboratórios.

Os utilizadores não registados, tem acesso a entrar na aplicação e ver os projetos existentes, no formato de cartões, com informação sobre o número de vagas em aberto, número de membros, criador e estado do projeto.

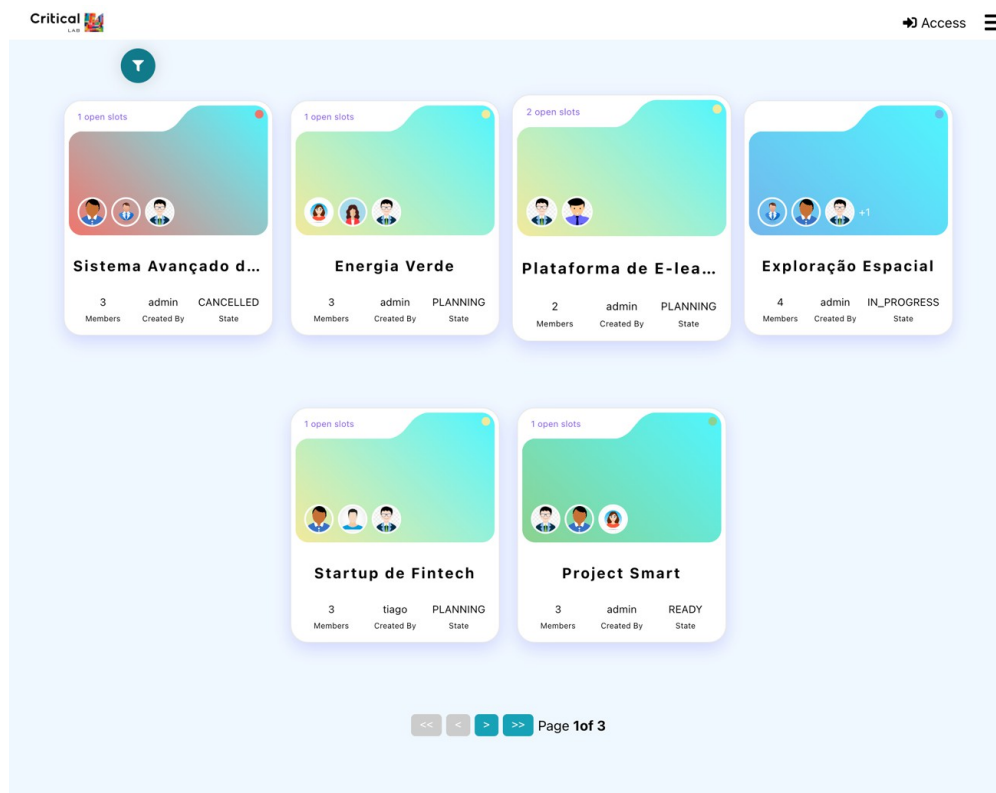


Fig.6 - Página de entrada da aplicação.

Nesta vista o utilizador tem ainda acesso a pesquisar com filtros os projetos.

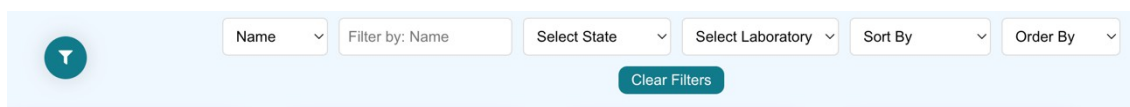


Fig.7 - Detalhe dos filtros da página de entrada da aplicação.

Assim que o utilizador tenta entrar num projeto para ver mais detalhes, surge a janela de início de sessão.

É possível também aceder à janela de início de sessão e às opções de cor do tema e de língua no canto superior direito do ecrã.

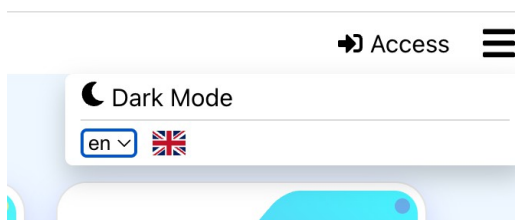


Fig.8 - Detalhe do acesso à janela de início de sessão.

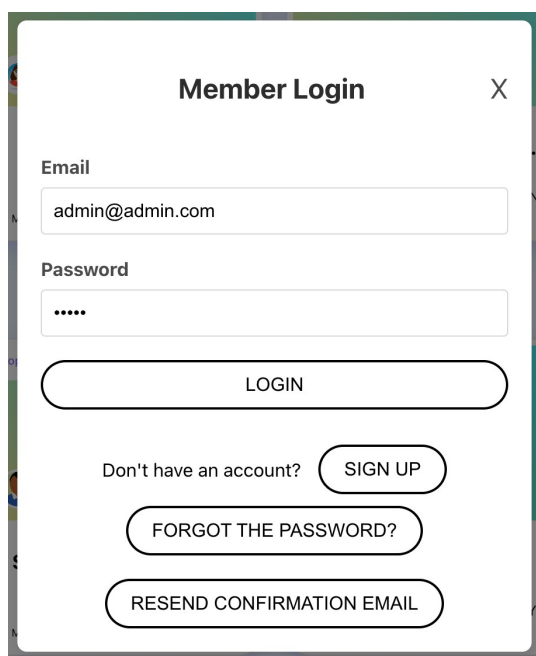


Fig.9 - Janela de início de sessão.



Caso o utilizador ainda não esteja registado, pode registar-se aqui. No menu de registo, existe um formulário onde o utilizador tem de preencher os campos identificados para um registo inicial

The figure consists of two side-by-side screenshots of a web application's registration interface. The left screenshot shows a light blue registration form with the following fields: Email (text input), Password (text input), Confirm password (text input with an information icon), Username (text input), First Name (text input), Last Name (text input), and Laboratory (dropdown menu). At the bottom are 'REGISTER' and 'BACK' buttons. The right screenshot shows the same form, but with a dark grey background and a white modal overlay titled 'Confirmation Email Sent' with an 'OK' button. The form fields behind the modal are partially filled: Email is 'aor.scrum.board@gmail.com', Password is masked with dots, Confirm password is masked with dots, Username is 'joao', Last Name is 'santos', and Laboratory is 'Lisboa'.

Fig.10 - Janela de registo e mensagem de confirmação de sucesso.

Após o registo, para o utilizador ficar com o perfil autenticado, é necessário ir ao email e validar o registo através do email de validação recebido.

No menu de iniciar sessão é também possível fazer o pedido de recuperação de password e pedido de reenvio de email de confirmação de conta caso a conta ainda não tenha sido validada

The figure consists of two side-by-side screenshots of a web application's login menu. The left screenshot shows a light blue card titled 'Recover Password' with a key icon. It has an 'Email' text input field and two buttons at the bottom: 'ASK FOR A NEW PASSWORD' and 'BACK TO LOGIN'. The right screenshot shows a light blue card titled 'Resend Confirmation Email' with an envelope icon. It has an 'Email' text input field and two buttons at the bottom: 'ASK FOR A NEW CONFIRMATION EMAIL' and 'BACK'.

Fig.119 Janelas de recuperação de password e de reenvio de email de confirmação.

No caso de início de sessão válida, entra-se na página principal da aplicação, onde se tem uma vista geral de todos os projetos e um menu lateral de navegação. Nesta página o utilizador pode alternar entre a vista de tabela e a vista de cartões e tem vários filtros disponíveis. No canto superior direito tem a informação do utilizador que tem a sessão iniciada e atalho direto para a página de perfil, e o menu de terminar sessão e alteração de tema de cor e língua da aplicação estão sempre visíveis.

NAME	DESCRIPTION	STATE	FINAL DATE	LABORATORY	NUMBER OF MEMBERS	CREATED BY	ACTIONS
Sistema Avançado de IoT	Desenvolvimento de um ...	CANCELLED	2024/07/26	Lisboa	3	admin	<a href="#">View</a> <a href="#">Plan</a>
Energia Verde	Criação de uma solução ...	PLANNING	2024/07/26	Coimbra	3	admin	<a href="#">View</a> <a href="#">Plan</a>
Plataforma de E-learning	Criação de uma platafor...	PLANNING	2024/07/25	Tomar	2	admin	<a href="#">View</a> <a href="#">Plan</a>
Exploração Espacial	Projeto de pesquisa sobr...	IN_PROGRESS	2024/08/02	Vila Real	4	admin	<a href="#">View</a> <a href="#">Plan</a>
Startup de Fintech	Desenvolvimento de um...	PLANNING	2024/07/25	Coimbra	3	tiago	<a href="#">View</a> <a href="#">Plan</a>
Project Smart	This is a comprehensive ...	READY	2024/09/01	Coimbra	3	admin	<a href="#">View</a> <a href="#">Plan</a>
Galeria de Arte Virtual	Criação de uma galeria d...	PLANNING	2024/08/01	Lisboa	3	admin	<a href="#">View</a> <a href="#">Plan</a>
Rede Social Inovadora	Desenvolvimento de um...	PLANNING	2024/07/26	Viseu	3	tiago	<a href="#">View</a>
Veículo Autónomo	Desenvolvimento de um ...	PLANNING	2024/07/26	Lisboa	3	tiago	<a href="#">View</a>
Automação Residencial	Desenvolvimento de um ...	PLANNING	2024/08/30	Tomar	1	tiago	<a href="#">View</a>

Fig.12 - Página principal na vista de tabela com todos os projetos.

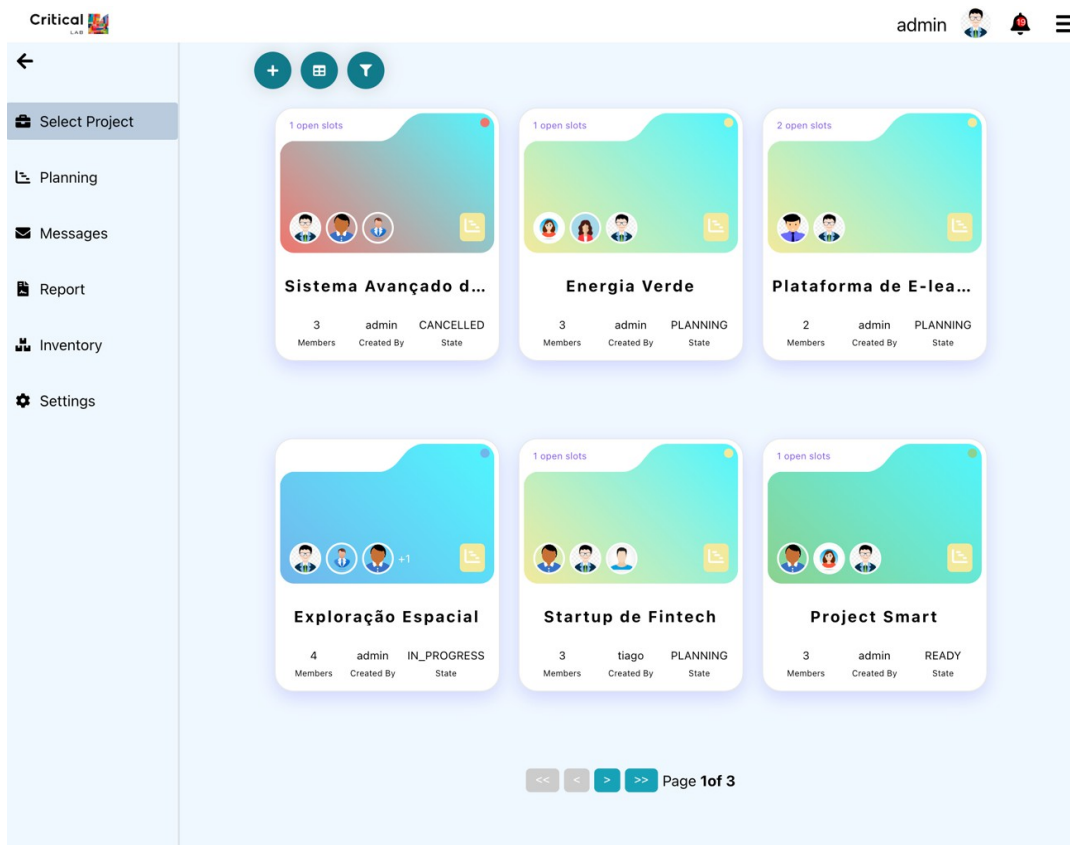


Fig.13 - Página principal na vista de cartões com todos os projetos.

Selecione o botão “Criar”, abre a janela de criação de projeto onde é necessário inserir toda a informação do projeto, incluindo as outras entidades que lhe vão estar associadas.

Fig.14 - Janela de criação de projetos (parte 1).

### Skills

Desenvolvimento Web	KNOWLEDGE	X
Psicologia	KNOWLEDGE	X
UX/UI Design	KNOWLEDGE	X

Add new
+ Add

### Keywords

Saúde	X
-------	---

Add new
+ Add

### Users

	Maria	not accepted	X
--	-------	--------------	---

Add new
+ Add

### Assets

Add new
+ Add

Submit

Fig.15 - Janela de criação de projetos (parte 2).

Selecionando um projeto, navegamos para a página do projeto, que está por defeito em modo de visualização. Para membros do projeto nível “Project Manager” há a possibilidade de abrir o projeto para edição.

Critical
admin

Energia Verde

Project Status: PLANNING

Project Name: Energia Verde
Laboratory: Coimbra
Conclusion Date: 26/07/2024

Description: Criação de uma solução de energia sustentável que utiliza painéis solares e armazenamento de energia eficiente.
Motivation: A necessidade global de reduzir a dependência de combustíveis fósseis e promover fontes de energia renováveis.

#### Users

	admin	PROJECT_MANAGER
	Manuela	NORMAL_USER
	Maria	NORMAL_USER

#### Skills

Energias Renováveis	KNOWLEDGE
Engenharia Elétrica	KNOWLEDGE
Modelagem e Simulação	KNOWLEDGE
Sistemas de Controle	KNOWLEDGE

#### Keywords

Project Chat

Fig.16 - Página de projeto em modo de visualização.

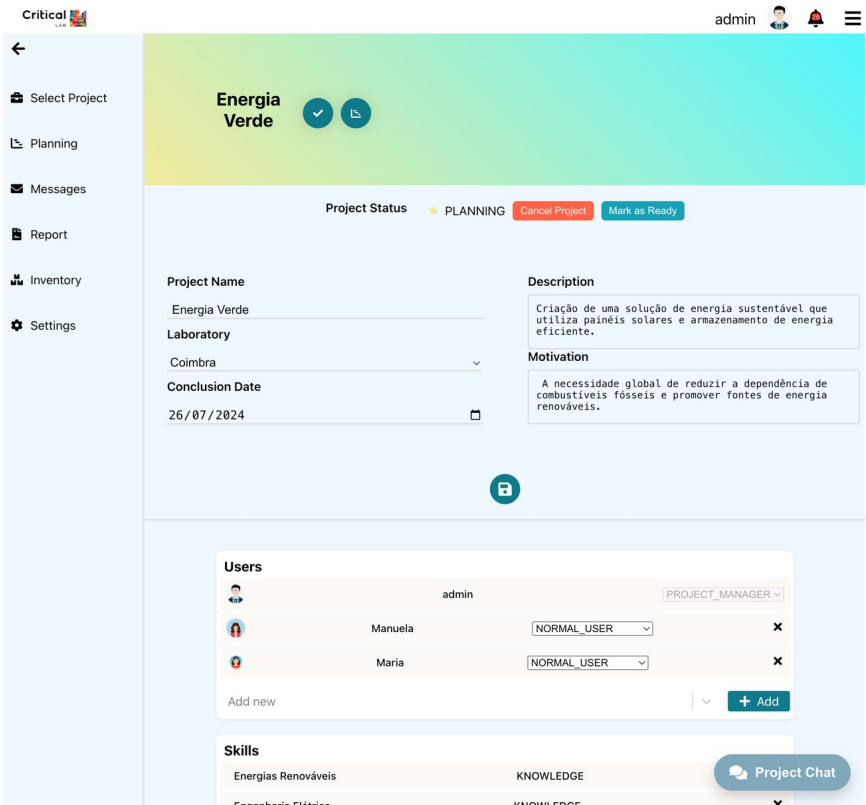


Fig.17 - Página de projeto em modo de edição.

Na edição há a possibilidade de editar as informações básicas do projeto e editar a relação com as restantes entidades.

Os membros do projeto têm diferentes níveis de acesso dentro do projeto e só os “Project Manager” podem editar esse nível.

Um utilizador que não seja membro de um projeto, pode-se candidatar a um projeto caso ainda haja vagas disponíveis e um membro a qualquer momento pode sair, caso não seja o criador do projeto.



Fig.18 - Exemplo de um utilizador que não é membro e pode-se candidatar.



Fig.19 - Exemplo de um utilizador que é membro e pode-sair do projeto.

Os membros do projeto, além de conseguirem visualizar os registos existentes do projeto, podem inserir manualmente novos.

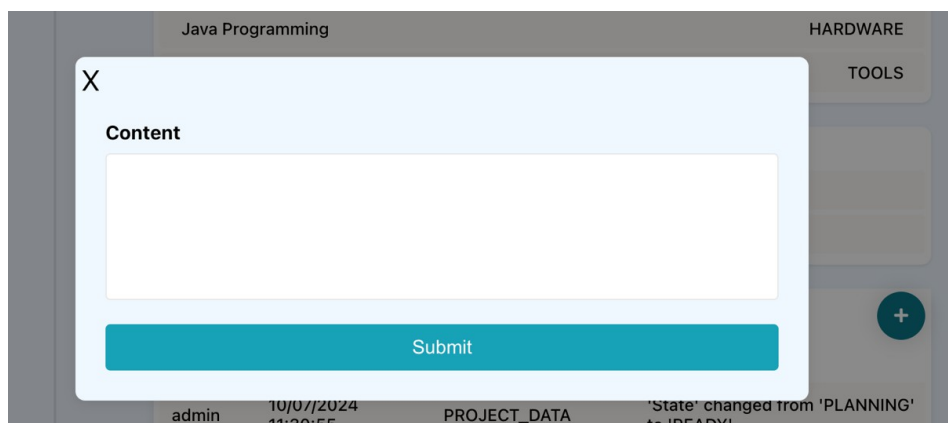


Fig.20 - Exemplo de um utilizador que é membro e pode-sair do projeto.

Quando o estado do projeto é atualizado e chega a “Ready”, o projeto fica pronto para aprovação e só um administrador de plataforma pode o aprovar. O projeto pode ser aprovado e seguir para o estado “In Progress” ou se for reprovado volta ao estado “Planning”.

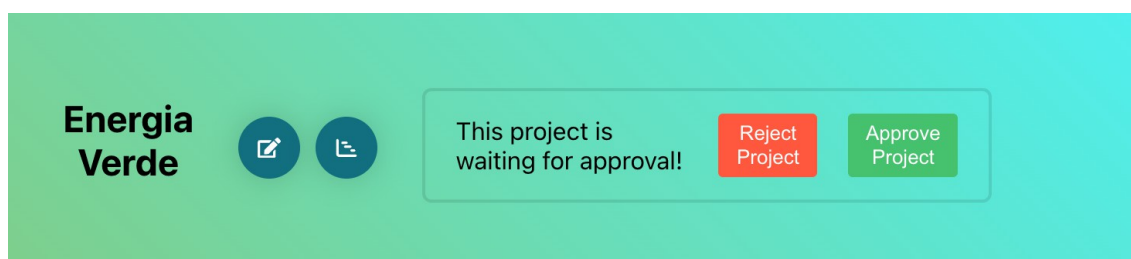


Fig.21 - Botões de aprovar ou rejeitar o projeto.

Dentro da página de cada projeto existe um chat de grupo para todos os membros do projeto.

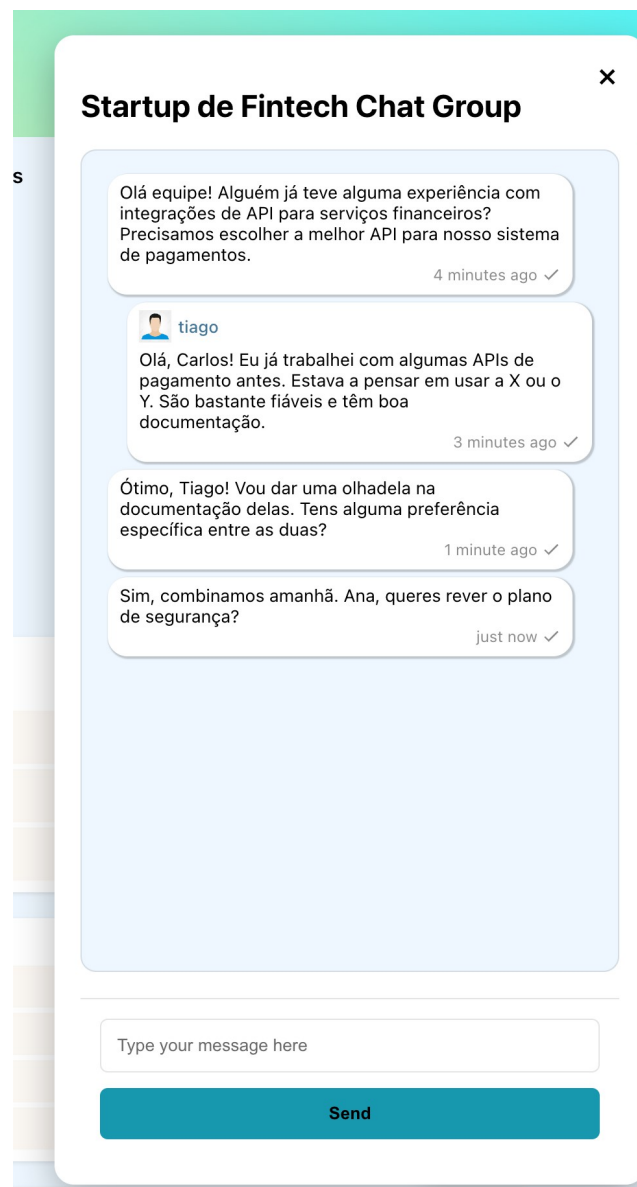


Fig.22 - Exemplo de um chat de grupo do projeto.

A partir da página de projeto é possível navegar para a página do plano de projeto, onde estão visíveis todas as tarefas do projeto num mapa de Gantt. Nesta página além de editar o mapa de Gantt, é possível navegar para as páginas de planeamento de outros projetos, criar tarefas ou editar as existentes.

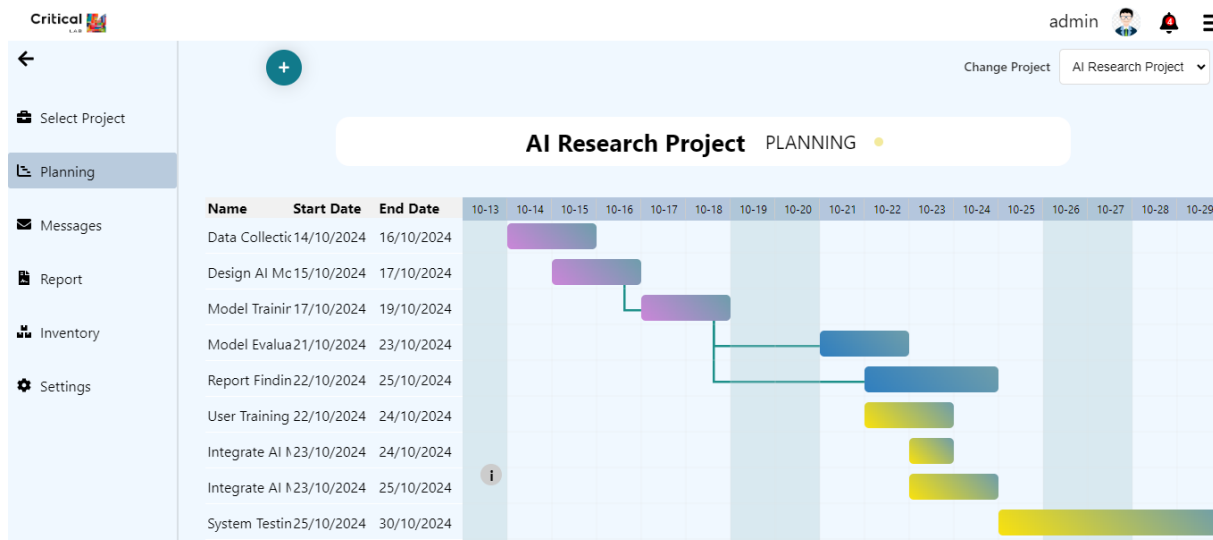


Fig.23 - Página de plano de projeto.

**Task Title**

**Description**

**Initial Planned Date**

**Duration (days)**

Fig.24 - Janela de criação de tarefa.

**Task Title**

**State**

**Description**

**Initial Planned Date**

Fig.25 - Janelas de edição de tarefa.



Na página das mensagens pessoais, o utilizador tem a opção de ver as suas mensagens enviadas e recebidas, ou criar uma mensagem. Na página das mensagens o utilizador pode fazer a pesquisa por elementos da mensagem.

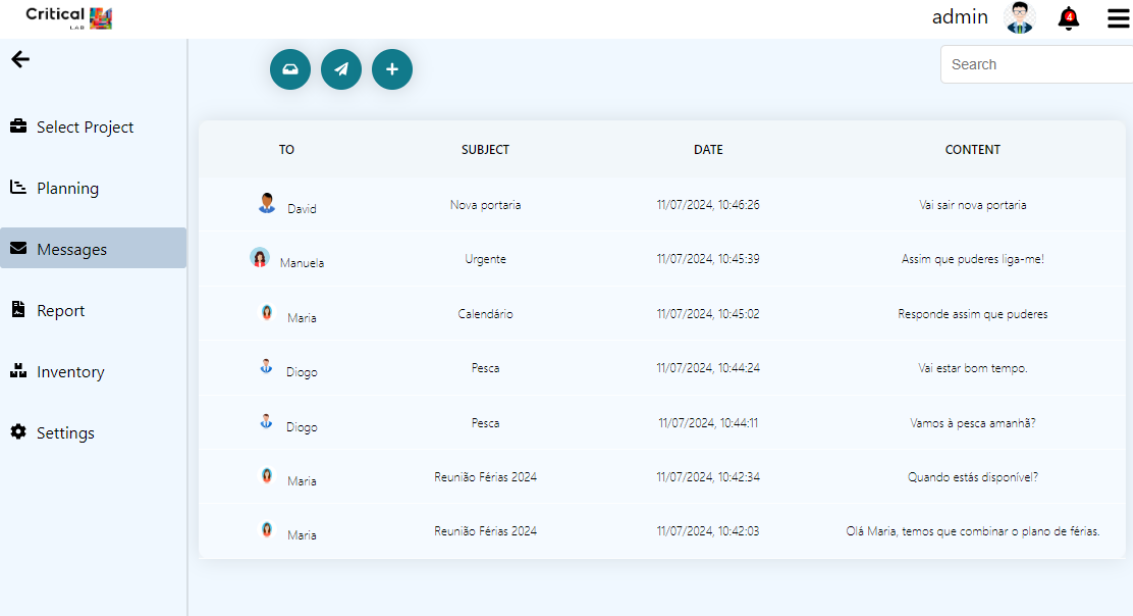


Fig.26 - Página das mensagens enviadas.

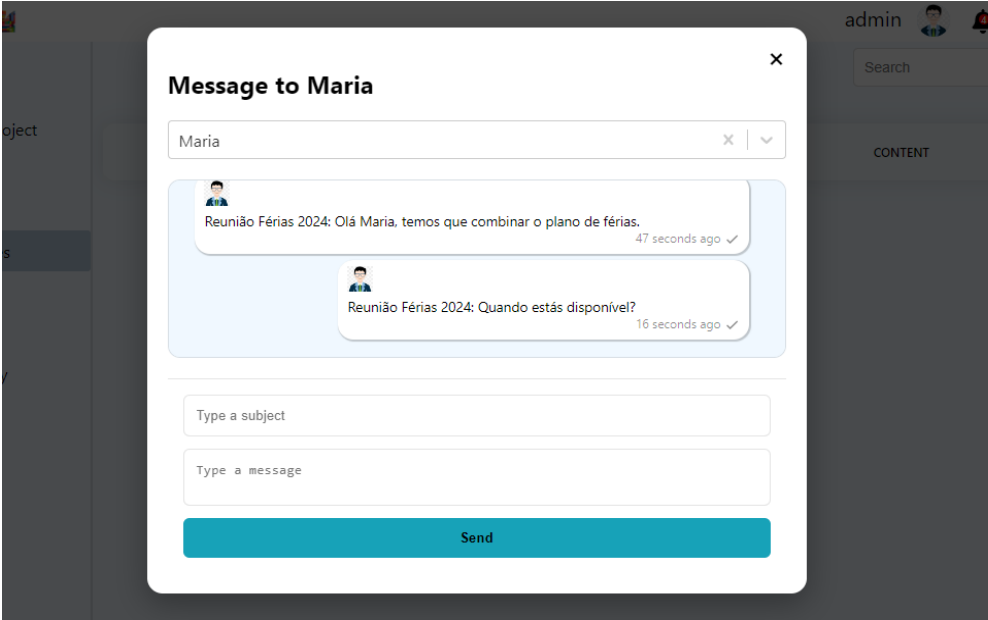


Fig.27 - Janelas de criação de mensagem.

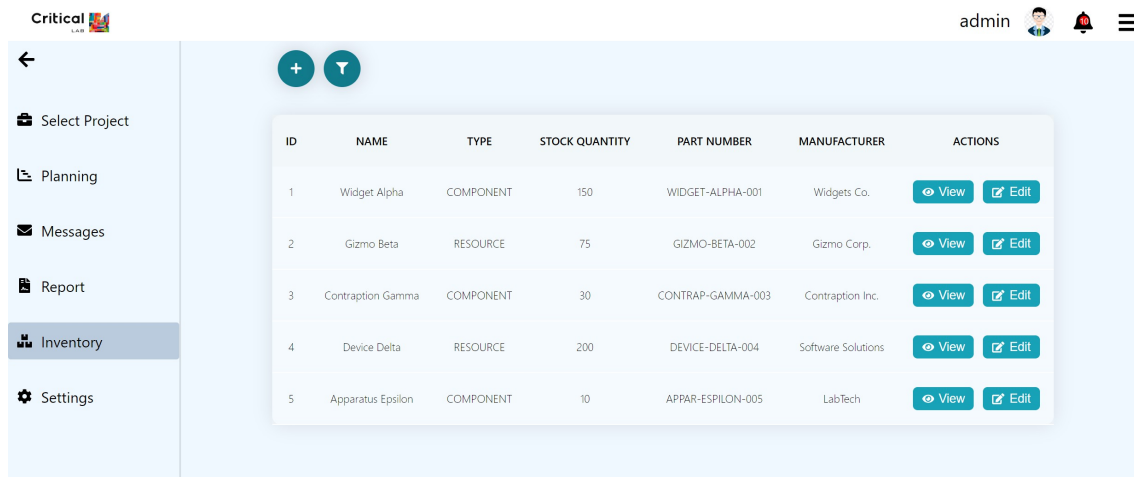
Para administradores de sistema, existe uma página onde é possível ver um dashboard com indicadores gerais, e fazer a impressão de relatórios em pdf.



Fig.28 - Página de criação de relatórios e visualização do dashboard.

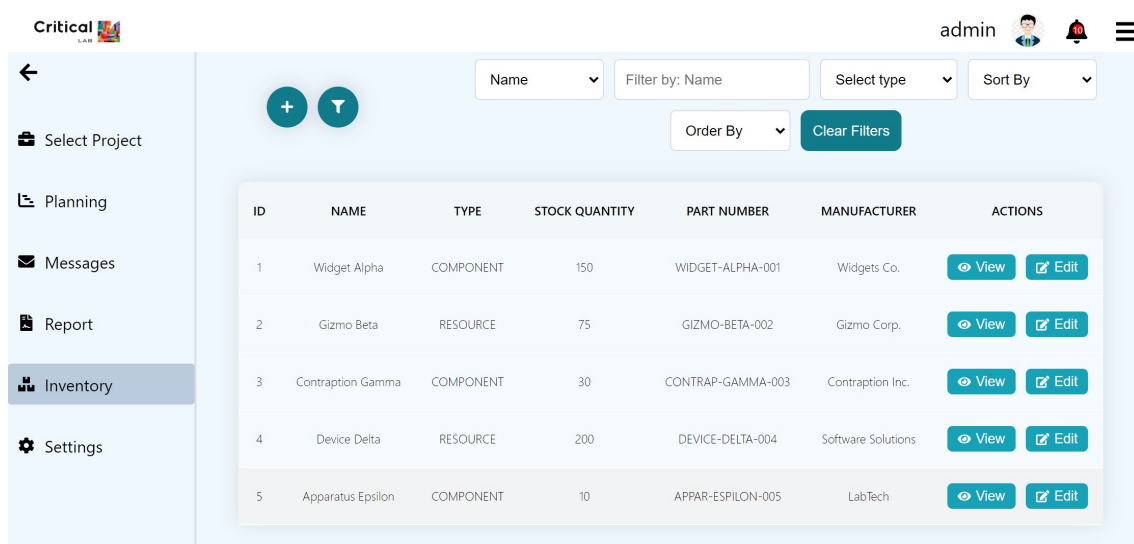
Na página do inventário, disponível a todos os utilizadores registados, é possível consultar, criar e editar os componentes e recursos.

Ao entrar na página está disponível uma tabela com todas as informações, onde o utilizador pode aplicar filtros para ver a informação organizada da forma pretendida.



ID	NAME	TYPE	STOCK QUANTITY	PART NUMBER	MANUFACTURER	ACTIONS
1	Widget Alpha	COMPONENT	150	WIDGET-ALPHA-001	Widgets Co.	<a href="#">View</a> <a href="#">Edit</a>
2	Gizmo Beta	RESOURCE	75	GIZMO-BETA-002	Gizmo Corp.	<a href="#">View</a> <a href="#">Edit</a>
3	Contraption Gamma	COMPONENT	30	CONTRAP-GAMMA-003	Contraption Inc.	<a href="#">View</a> <a href="#">Edit</a>
4	Device Delta	RESOURCE	200	DEVICE-DELTA-004	Software Solutions	<a href="#">View</a> <a href="#">Edit</a>
5	Apparatus Epsilon	COMPONENT	10	APPAR-ESPILON-005	LabTech	<a href="#">View</a> <a href="#">Edit</a>

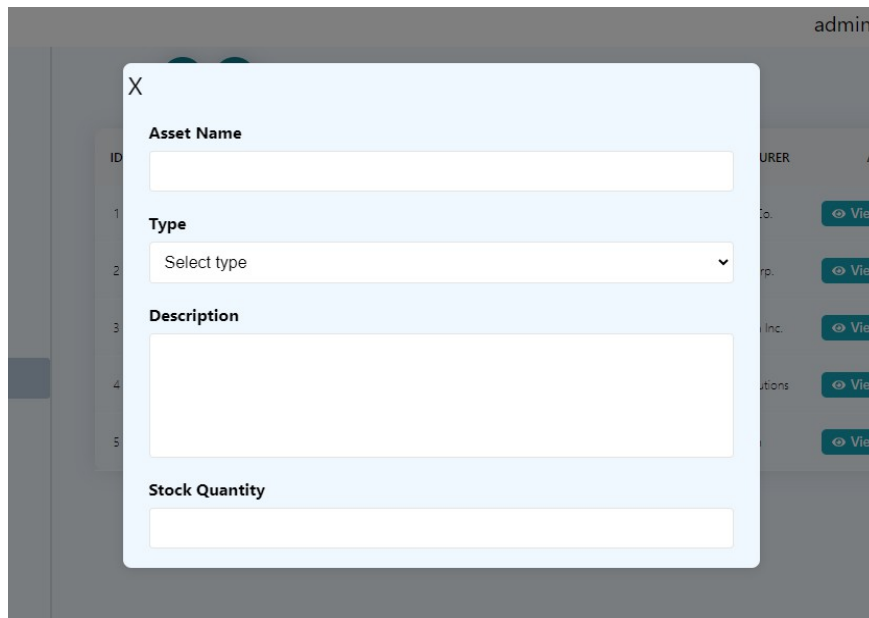
Fig.29 - Página de visualização do inventário.



ID	NAME	TYPE	STOCK QUANTITY	PART NUMBER	MANUFACTURER	ACTIONS
1	Widget Alpha	COMPONENT	150	WIDGET-ALPHA-001	Widgets Co.	<a href="#">View</a> <a href="#">Edit</a>
2	Gizmo Beta	RESOURCE	75	GIZMO-BETA-002	Gizmo Corp.	<a href="#">View</a> <a href="#">Edit</a>
3	Contraption Gamma	COMPONENT	30	CONTRAP-GAMMA-003	Contraption Inc.	<a href="#">View</a> <a href="#">Edit</a>
4	Device Delta	RESOURCE	200	DEVICE-DELTA-004	Software Solutions	<a href="#">View</a> <a href="#">Edit</a>
5	Apparatus Epsilon	COMPONENT	10	APPAR-ESPILON-005	LabTech	<a href="#">View</a> <a href="#">Edit</a>

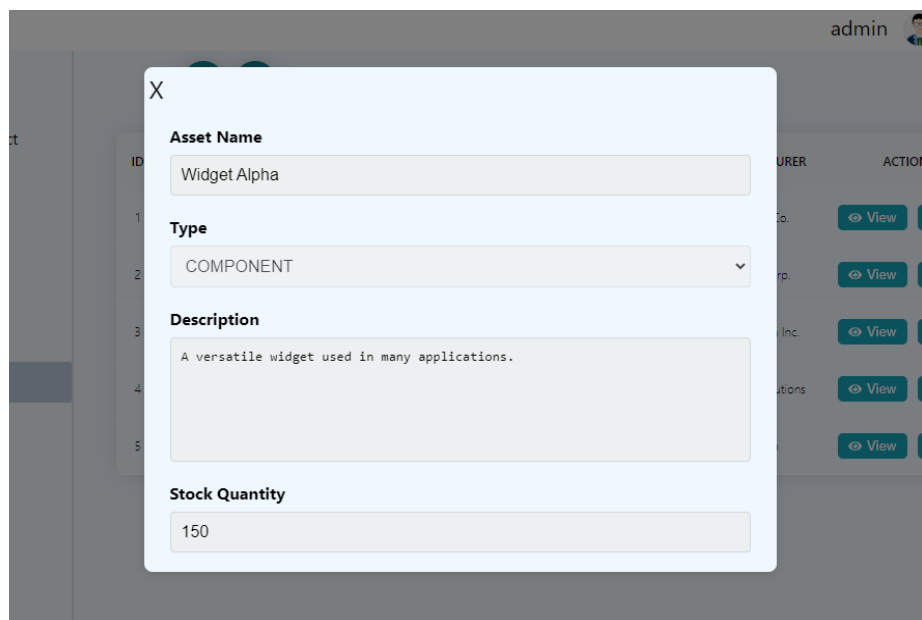
Fig.30 - Detalhe das opções de filtros.

Nessa mesma página pode navegar pelas opções de criar um ativo, onde deverá selecionar o tipo: Componente ou Recurso, visualizar ou editar um ativo existente.



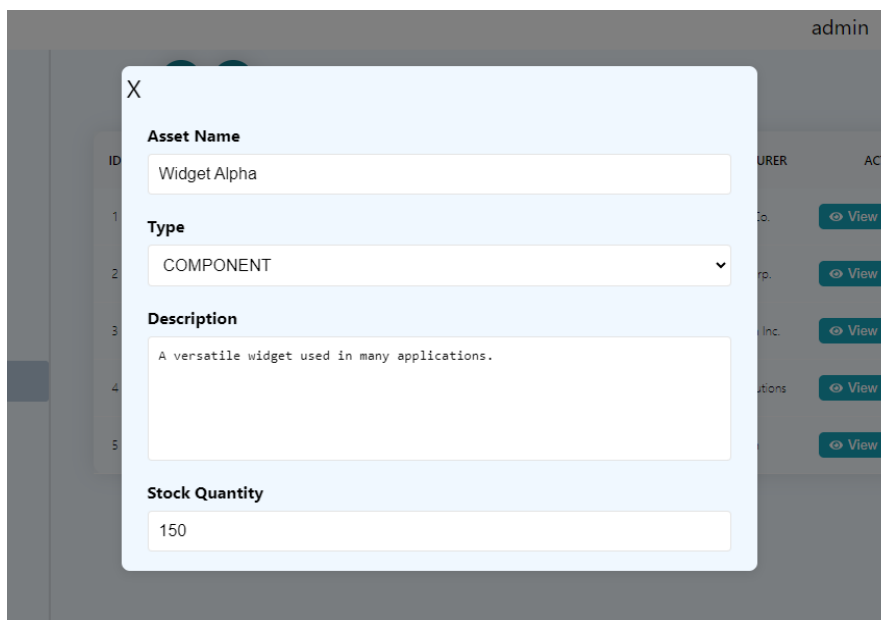
A modal form titled 'X' for creating a new asset. It contains four fields: 'Asset Name' (text input), 'Type' (dropdown menu with 'Select type' as the placeholder), 'Description' (text area), and 'Stock Quantity' (text input). The form is overlaid on a background showing a table of assets with columns for ID, Name, Type, and Actions (View, Edit, Delete).

Fig.31 - Janela de criação de um ativo.



A modal form titled 'X' for viewing an existing asset. It displays the following information: 'Asset Name' (Widget Alpha), 'Type' (COMPONENT), 'Description' (A versatile widget used in many applications.), and 'Stock Quantity' (150). The form is overlaid on a background showing a table of assets with columns for ID, Name, Type, and Actions (View, Edit, Delete).

Fig.32 - Janela de visualização de um ativo.



X

**Asset Name**

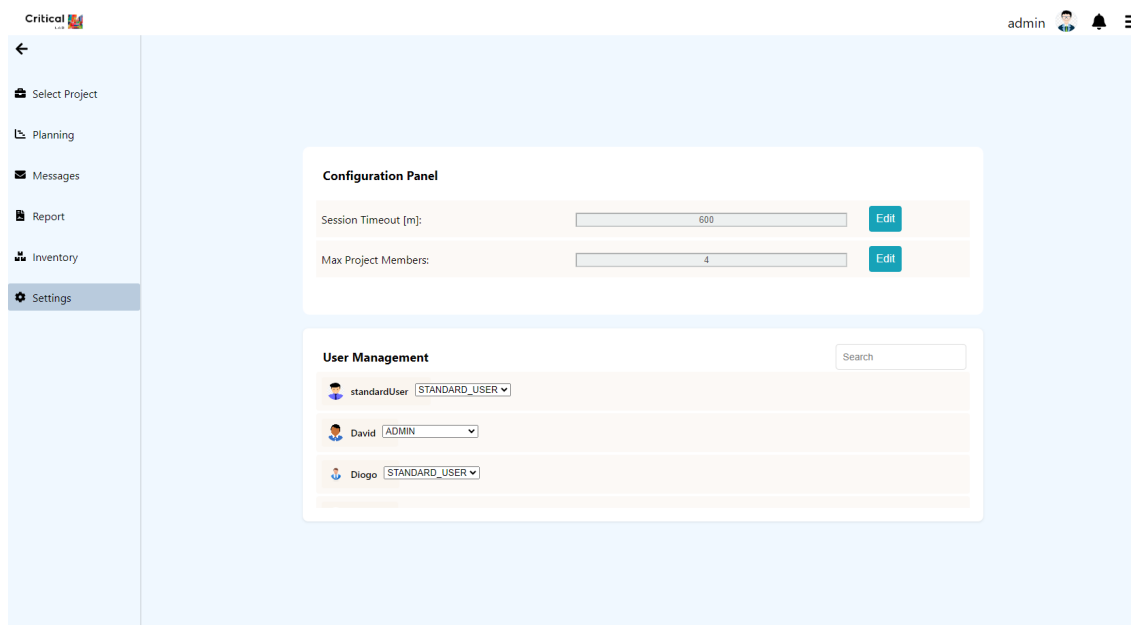
**Type**

**Description**

**Stock Quantity**

Fig.33 - Janela de edição de um ativo.

A última página disponível volta a ser uma página reservada a administradores, onde podem ser consultadas e editadas configurações gerais da aplicação bem como os níveis de acessos dos utilizadores dentro da aplicação.



Critical

admin

Select Project

Planning

Messages

Report

Inventory

Settings

**Configuration Panel**

Session Timeout [m]:
Edit

Max Project Members:
Edit

**User Management**

standardUser
STANDARD\_USER

David
ADMIN

Diogo
STANDARD\_USER

Fig. - Página de configurações.

45

## Testes e Validação

---

### Testes com Postman

Para garantir a robustez e a fiabilidade do nosso sistema, realizámos uma série de testes utilizando o Postman. Estes testes incluíram tanto cenários positivos quanto negativos, abrangendo todos os serviços expostos pelo nosso backend.

#### Testes de Criação de Projetos:

Testámos a criação de projetos através do endpoint `POST /projects/create`, verificando a criação bem-sucedida com todos os campos obrigatórios preenchidos corretamente.

Incluímos cenários negativos, como tentativas de criação de projetos sem campos obrigatórios ou com dados inválidos, para garantir que o sistema responde apropriadamente com mensagens de erro adequadas.

#### Testes de Autenticação e Registo de Utilizadores:

Verificámos a capacidade do sistema de registar novos utilizadores (`POST /user/register`), assegurando que os dados fornecidos são válidos e únicos (e.g., emails e usernames não duplicados).

Realizámos testes de autenticação (`POST /user/login`), confirmando que apenas utilizadores com credenciais corretas conseguem aceder ao sistema.

Testámos a recuperação e redefinição de palavra-passes para garantir a segurança e a funcionalidade do sistema em casos de perda de credenciais.

#### Testes de Atualização de Perfis e Permissões:

Testámos a atualização dos perfis de utilizadores (`PUT /user/updateProfile`), garantindo que todas as alterações permitidas são corretamente refletidas na base de dados.

Verificámos a capacidade de alterar papéis dos utilizadores (`PUT /user/updateRole`), assegurando que as permissões são atualizadas conforme esperado.

#### Testes de Consulta e Listagem:

Realizámos testes para os endpoints de listagem e consulta de dados, como obter detalhes de projetos e utilizadores (`GET /projects`, `GET /user/profile`), confirmando a correta recuperação das informações armazenadas.

### Testes com Mockito

Para além dos testes funcionais realizados com o Postman, desenvolvemos testes unitários utilizando a biblioteca Mockito para garantir a integridade dos nossos beans e serviços.

#### Exemplo de Teste: UserBeanTest

##### Configuração Inicial:

Utilizamos anotações como `@Mock` e `@InjectMocks` para criar instâncias simuladas dos nossos DAOs e serviços dependentes, configurando-os antes de cada teste com `MockitoAnnotations.openMocks(this)`.

Teste de Registo de Utilizador:

Simulámos a criação de um novo utilizador verificando que os métodos corretos foram chamados e que as entidades foram persistidas conforme esperado. Incluímos verificações para assegurar que o email, username, e outros campos foram corretamente definidos.

Teste de Atualização de Perfil:

Verificámos a atualização dos dados do perfil do utilizador, confirmando que todas as mudanças permitidas (nome, foto, biografia, etc.) são corretamente aplicadas e persistidas.

Teste de Redefinição de Palavra-passe:

Simulámos o processo de redefinição de palavra-passe, incluindo a verificação de tokens de redefinição e a codificação segura da nova palavra-passe.

## Testes com Jest

Implementámos testes para os nossos componentes React utilizando a biblioteca Jest. Estes testes focaram-se em garantir a correta navegação e renderização dos componentes.

Componente HomepageAside:

Realizámos testes de navegação, verificando que os links dentro do componente redirecionam para as páginas corretas (e.g., Select Project, Project Planning, Messages, etc.).

Utilizamos mocks para as lojas de estado (`useLayoutStore`, `useTranslationsStore`) e outros componentes dependentes, assegurando que os testes são isolados e focados na funcionalidade específica do componente.

Apesar de ainda estarmos a concluir os testes com Jest, já temos uma base sólida que garante a funcionalidade e a interação correta dos nossos componentes de interface.

## Benefícios e Desafios

Os testes extensivos realizados oferecem várias vantagens nos campos de Garantia de Qualidade, Segurança e Manutenção.

Permitem a identificação precoce de bugs e problemas, garantindo que o sistema funciona conforme esperado. Facilidade na identificação de regressões e na validação de novas funcionalidades. Estes esforços de testes são cruciais para a entrega de um sistema robusto e confiável.

No entanto, enfrentamos alguns desafios, como a complexidade da gestão de testes em grandes sistemas e a necessidade de manutenção contínua das suites de testes para acompanhar as mudanças no código base.

## Desafios e soluções

---

### Principais Desafios Encontrados

Durante o desenvolvimento do nosso sistema, enfrentámos vários desafios significativos, especialmente no que diz respeito ao controlo de acesso e à segurança. A complexidade de garantir um sistema seguro e eficiente exigiu uma abordagem metódica e várias iterações para superar os obstáculos encontrados.

Entre os principais desafios estava a implementação de JWT para autenticação e autorização, onde foi crucial garantir a segurança na geração e validação dos JSON Web Tokens (JWT) e gerir a validade e a renovação dos tokens para manter as sessões seguras e estáveis. Além disso, a gestão de sessões com cookies HTTP-Only foi outro ponto crítico, pois precisávamos armazenar os tokens de autenticação de forma segura, impedindo o acesso via scripts, e gerir o tempo de expiração das sessões, assegurando a renovação automática dos tokens antes do seu término.

Outro desafio foi a validação de permissões baseadas em funções, onde tivemos de implementar um sistema robusto de controlo de acesso baseado em funções (RBAC), gerindo a relação entre utilizadores, papéis e permissões de forma eficiente e escalável. Por fim, proteger a segurança e integridade dos dados contra ataques comuns como XSS, CSRF e SQL Injection e assegurar que todas as interações com a base de dados fossem seguras, garantindo a proteção dos dados dos utilizadores, foi fundamental.

Para superar esses desafios, adotámos diversas estratégias eficazes. Inicialmente, utilizámos JWT com uma chave específica para garantir a segurança dos tokens, implementando a geração e validação dos mesmos utilizando a biblioteca `io.jsonwebtoken`, criando tokens assinados com o algoritmo HS512. Desenvolvemos métodos para gerar tokens de autenticação e sessão com tempos de expiração configuráveis, garantindo assim a sua segurança e validade. Na gestão segura de sessões, utilizámos cookies HTTP-Only para armazenar os tokens de autenticação e sessão, impedindo o acesso a estes via scripts, o que reduz significativamente o risco de ataques XSS. Também implementámos um sistema de renovação de tokens, onde os tokens são automaticamente renovados se estiverem prestes a expirar, mantendo a continuidade da sessão do utilizador sem comprometer a segurança. Na implementação do RBAC, criámos entidades para representar níveis de acesso (`RoleEntity`) e métodos (`MethodEntity`), juntamente com uma tabela de relação para gerir as permissões associadas a cada papel, e implementámos filtros e verificações para garantir que apenas utilizadores com as permissões adequadas pudessem aceder a determinados recursos, utilizando anotações personalizadas para definir requisitos de permissão a nível de métodos. Para garantir a segurança dos dados, utilizámos mecanismos rigorosos de validação de entrada para evitar ataques de injeção de SQL e outras vulnerabilidades, além de implementar filtros para proteger contra ataques CSRF, assegurando que todas as requisições fossem autenticadas e válidas.



Ao longo do projeto, aprendemos várias lições valiosas que nos ajudaram a melhorar a nossa abordagem ao desenvolvimento de software seguro e eficiente. A segurança deve ser considerada desde o início do desenvolvimento, não apenas como um complemento, e a integração de práticas de segurança desde a fase de design ajudou-nos a evitar muitas vulnerabilidades comuns. A automatização de testes de segurança e a validação contínua foram cruciais para identificar e corrigir vulnerabilidades rapidamente.

Utilizar ferramentas como Mockito e Jest para testes automatizados ajudou-nos a manter a integridade do sistema. A gestão eficiente de sessões com cookies HTTP-Only e a implementação de renovação automática de tokens melhoraram significativamente a segurança e a experiência do utilizador, reduzindo o risco de sessões comprometidas. Implementar um sistema de controlo de acesso baseado em funções flexível e escalável permitiu-nos gerir permissões de forma eficiente, facilitando a adição de novos papéis e permissões conforme necessário. Por fim, o desenvolvimento do componente TaskManager, que integra um gráfico de Gantt em React, foi uma parte crucial do projeto, trazendo visualização avançada e gestão de tarefas para os utilizadores, permitindo uma visão clara e organizada das tarefas do projeto, suas dependências e prazos, facilitando a compreensão do progresso do projeto e melhorando significativamente a gestão e a coordenação das atividades.

O desenvolvimento do componente TaskManager, que integra um gráfico de Gantt em react, foi uma parte crucial do projeto, trazendo visualização avançada e gestão de tarefas para os utilizadores. Este componente foi projetado para permitir uma visão clara e organizada das tarefas do projeto, suas dependências e prazos. O gráfico de Gantt facilita a compreensão do progresso do projeto, permitindo aos utilizadores verem as tarefas em um cronograma visual, o que melhora significativamente a gestão e a coordenação das atividades.

## Conclusão

---

O projeto desenvolvido representa um sistema robusto e seguro para a gestão de projetos e utilizadores. Desde a conceção até à implementação, enfrentámos e superámos desafios importantes, especialmente na área de controlo de acesso e segurança.

O uso de JWT para autenticação e autorização trouxe uma camada adicional de segurança, assegurando que apenas utilizadores autenticados podem aceder aos recursos do sistema. Além disso, a utilização de cookies HTTP-Only para armazenamento seguro de tokens mitigou os riscos de ataques Cross-Site Scripting (XSS), melhorando a segurança geral das sessões. Esta abordagem permitiu criar um ambiente onde a segurança dos dados dos utilizadores é prioritária e constantemente monitorizada.

Um dos principais benefícios do projeto foi a implementação de um sistema de renovação automática de tokens. Este sistema prolonga a validade das sessões de forma segura e transparente para o utilizador, melhorando a experiência do utilizador sem comprometer a segurança. A capacidade de renovar automaticamente os tokens de autenticação e sessão garante que os utilizadores permanecem conectados de forma segura e contínua, sem a necessidade de reautenticação constante.

A flexibilidade oferecida pelo RBAC foi outra grande conquista. Este modelo permitiu definir e gerir níveis de acesso e permissões de forma granular, proporcionando uma gestão detalhada do acesso aos recursos do sistema. A capacidade de escalar o sistema facilmente, adicionando novos papéis e permissões conforme necessário, demonstra a adaptabilidade e a escalabilidade do sistema. Este nível de controlo de acesso garante que cada utilizador tem exatamente as permissões necessárias para realizar as suas tarefas, sem comprometer a segurança.

O projeto também se destacou na área de testes e validação. A implementação de testes automatizados com ferramentas como Postman, Mockito e Jest garantiu que o sistema foi rigorosamente testado, identificando e corrigindo rapidamente quaisquer vulnerabilidades. Esta abordagem proativa na deteção e resolução de problemas contribuiu para a criação de um sistema mais robusto e confiável, estabelecendo um padrão elevado para a qualidade e segurança do software.

A integração de medidas de segurança desde o início do desenvolvimento tornou-se uma norma, assegurando que futuros projetos beneficiem desta abordagem. A prática de incorporar segurança e testes automatizados desde as fases iniciais do desenvolvimento não só melhorou a qualidade do software, mas também reduziu significativamente o tempo associado à correção de problemas posteriores.

Outro ponto importante foi a aplicação de ferramentas de gestão ágeis, o que nos permitiu definir objetivos semanais ("sprints") e acompanhar sua obtenção de forma a garantir o cumprimento do prazo final. Isso também nos possibilitou avaliar continuamente se estávamos

atrasados ou não. A cultura de entregar código final a cada sprint é mais uma competência adquirida neste curso.

Finalmente, este trabalho é o culminar de um curso intensivo e exigente, que nos transformou completamente e nos deixa aptos a ingressar no mercado de trabalho em um período de aproximadamente 10 meses. Essa evolução não seria possível sem o apoio de todos os professores que nos acompanharam, aos quais queremos deixar nosso agradecimento, especialmente à nossa tutora do projeto final, professora Naghmeh Ivaki.