PROJETO A3

UC: SISTEMAS DISTRIBUÍDOS E MOBILE

3º SEMESTRE

SALVADOR – BA

2025.1

PROJETO A3

UC: SISTEMAS DISTRIBUÍDOS E MOBILE

Alunos:

Ágatha Brandão Borges - 12724145681

Amanda Francisca Ramos Pereira - 12724143394

Maurício Gabriel Leal da Silva - 12724157145

Pablo Ernesto da Cunha Guerreiro - 12724111921

Paulo Henrique Silva de Almeida - 12724141651

Yara Carolina Leite dos Santos - 12725138638

Projeto solicitado pelos professores
Eduardo Sidney e Danilo Miguel,
que ministram a unidade curricular,
Sistemas Distribuídos e Mobile.

SALVADOR - BA
2025.1

Sumário

Descrições do projeto

O **GestãoPro** é uma plataforma de acompanhamento de Tarefas Corporativas, tem como objetivo otimizar o acompanhamento e a gestão das tarefas realizadas pelos colaboradores de uma empresa. Ele é dividido em três níveis de acesso:

1. **Funcionário**: Pode visualizar as tarefas que lhe foram atribuídas, registrar o andamento das atividades e marcar as tarefas como concluídas. Essa funcionalidade garante maior controle individual e transparência no cumprimento das responsabilidades.

2. **Supervisor**: Responsável por criar, atribuir e gerenciar as tarefas dos funcionários sob sua supervisão. Pode acompanhar o progresso das atividades em tempo real, fazer ajustes quando necessário e garantir que os prazos sejam cumpridos.

3. **Gerente**: Possui acesso a relatórios gerenciais detalhados sobre o desempenho das equipes, andamento das tarefas e produtividade geral. Esses relatórios auxiliam na tomada de decisões estratégicas e no planejamento organizacional.

O sistema promove a comunicação eficiente entre os níveis hierárquicos, melhora a produtividade e permite uma visão clara do andamento das atividades dentro da empresa.

Requerimentos de software para aplicação (bibliotecas e linguagem)

1. Linguagem de Programação: A aplicação foi desenvolvida utilizando tecnologias modernas, separando claramente as responsabilidades entre o front-end e o back-end: Front-end: Utiliza as linguagens HTML, CSS e JavaScript para a construção das interfaces gráficas da aplicação. Essa combinação permite a criação de páginas web responsivas, interativas e com boa experiência de uso para os colaboradores, supervisores e gerentes. Back-end: Desenvolvido em Java, utilizando o framework Spring Boot, que oferece uma arquitetura robusta, segura e escalável. O Spring Boot facilita a criação de APIs RESTful, integração com bancos de dados e implementação de autenticação, além de fornecer uma estrutura organizada com suporte ao padrão MVC (Model-View-Controller).

2. Banco de Dados: A aplicação utiliza o SQLite como sistema de gerenciamento de banco de dados. O SQLite é uma solução leve, de fácil configuração e integrada diretamente ao ambiente da aplicação, eliminando a necessidade de instalação de um servidor de banco de dados externo. Ideal para aplicações de pequeno a médio porte, pois oferece desempenho eficiente, portabilidade e armazenamento em um único arquivo. No contexto da aplicação, o SQLite é responsável por armazenar e gerenciar informações como: Dados dos colaboradores (funcionários, supervisores e gerentes); Tarefas atribuídas e concluídas; Status e histórico de atividades e Informações para geração de relatórios.

3. Bibliotecas/Frameworks: Spring Boot DevTools: Ferramenta de desenvolvimento com recarga automática e melhorias para produtividade local.

- Spring Web: Permite a criação de APIs RESTful e controle das rotas HTTP.
- Spring Data JPA: Facilita a persistência de dados com uso de repositórios baseados em JPA.
- JDBC API: Comunicação direta com o banco SQLite, quando necessário.
- Lombok: Reduz a verbosidade do código Java com anotações como `@Getter`, `@Setter`, `@Builder` etc.
- OpenFeign: Facilita a comunicação com APIs externas via cliente HTTP declarativo.

4. Ferramentas de Desenvolvimento: Durante o desenvolvimento da aplicação, foram utilizadas duas ferramentas principais: **Visual Studio Code (VS Code)**: Utilizado para a construção e edição das páginas front-end da aplicação (HTML, CSS e JavaScript). **IntelliJ IDEA**: Utilizado para o desenvolvimento do back-end da aplicação, implementado em Java com o framework Spring Boot.

Instrução para instalação e execução da aplicação

1. Pré-requisitos: Instale os seguintes softwares:

- Java JDK 21 ou superior
- Maven ou Gradle
- IntelliJ IDEA (para o back-end)
- Visual Studio Code (para o front-end)
- Navegador moderno (Chrome, Firefox etc.)

2. Configurar e Executar o Back-end (Spring Boot + SQLite)

1. Abra o projeto no IntelliJ IDEA.
2. No arquivo `application.properties`, configure o banco SQLite:

```properties
spring.datasource.url=jdbc:sqlite:database/tarefas.db
spring.datasource.driver-class-name=org.sqlite.JDBC
spring.jpa.hibernate.ddl-auto=update
```

3. No `pom.xml`, adicione a dependência do SQLite:

```xml
<dependency>
  <groupId>org.xerial</groupId>
  <artifactId>sqlite-jdbc</artifactId>
  <version>3.36.0.3</version>
</dependency>
```

4. Execute a aplicação (via terminal ou botão "Run" no IntelliJ):

```bash
./mvnw spring-boot:run
```

A API estará disponível em: `http://localhost:8080`

3. Executar o Front-end (HTML/CSS/JS)

1. Abra a pasta do front-end.
2. Abra o arquivo index.html.
3. Execute com a extensão Live Server ou abra o arquivo direto no navegador.

Justificativa para a abordagem de comunicação escolhida.

A aplicação adota uma arquitetura de comunicação baseada em API REST, onde o front-end (desenvolvido em HTML, CSS e JavaScript) se comunica com o back-end (implementado em Java utilizando o framework Spring Boot) por meio de requisições HTTP que trocam dados no formato JSON.

🔹 **Motivação da Escolha**

Essa abordagem foi escolhida por sua capacidade de promover uma separação clara entre a interface do usuário e a lógica de negócio, permitindo que cada camada evolua de forma independente. Isso torna o sistema mais modular, escalável e fácil de manter.

🔹 **Funcionamento da Comunicação**

No front-end, o usuário interage com formulários e botões que, por meio de JavaScript (usando a API fetch()), enviam requisições ao back-end. Essas requisições seguem os métodos padrões do protocolo HTTP, como:

- GET – para buscar tarefas
- POST – para criar ou concluir tarefas
- PUT/DELETE – para atualizações ou remoções, se necessário

O back-end processa essas requisições, realiza operações sobre o banco de dados SQLite, e responde ao front-end com informações estruturadas em JSON.

🔹 **Aderência aos Objetivos do Projeto**

Como o sistema lida com diferentes perfis de usuários (funcionários, supervisores e gerentes), que acessam e manipulam tarefas em tempo real, a comunicação via API REST garante:

- Clareza no controle de permissões
- Organização das operações por perfil
- Facilidade para gerar relatórios e registrar histórico de ações

PROJETO A3

UC: USABILIDADE, DESENVOLVIMENTO WEB, MOBILE E JOGOS

3º SEMESTRE

SALVADOR – BA

2025.1