

Instituto Politécnico de Viseu

Escola Superior de Tecnologia e Gestão de Viseu

Departamento de Informática

Unidade Curricular: Aplicações para a Internet II

Relatório Relativo a Aplicações para a Internet II

Tema: Softinsa

Realizado por: Guilherme Félix – 25172

João Cruz – 25178

Viseu, 2024

Instituto Politécnico de Viseu

Escola Superior de Tecnologia e Gestão de Viseu

Departamento de Informática

Relatório relativo a Aplicações para a Internet II

Curso de Licenciatura em Engenharia Informática

Unidade Curricular de Aplicações para a Internet II

Softinsa

Ano Letivo 2023/24

Viseu, 2024

**ÍNDICE**

[***1. Introdução 1***](#_gjdgxs)

[***2. Backend 2***](#_30j0zll)

[**Models 2**](#_1fob9te)

[**Rotas 4**](#_2et92p0)

[**Controllers 5**](#_3dy6vkm)

[Middleware 7](#_kh9422dku2dc)

[Utils 8](#_vv7ej9vohy7u)

[***3. Frontend 11***](#_35nkun2)

[API 11](#)

[Auth 11](#)

[Avaliação 13](#_qphbgi1aub1m)

[Calendário 14](#_9yxs2ujen9tb)

[Estabelecimentos 15](#_tqhj966g2skp)

[Eventos 16](#_la7h4to9xr8n)

[Home 18](#_5uzp9ydp13vo)

[Notificação 19](#_v03bg5b33q5k)

[Utilizador 20](#_ne33znanxqxj)

[***4. Conclusões 22***](#_2p2csry)

# Introdução

O projeto desenvolvido visa criar uma plataforma abrangente para gestão de eventos e estabelecimentos, com funcionalidades que vão desde a autenticação de usuários até a avaliação de serviços. Utilizando uma arquitetura de aplicação web, o sistema é dividido em frontend e backend, proporcionando uma interface intuitiva para os usuários e uma robusta infraestrutura de dados e lógica no servidor. Este relatório detalha cada componente do sistema, explicando sua implementação e papel dentro do projeto.

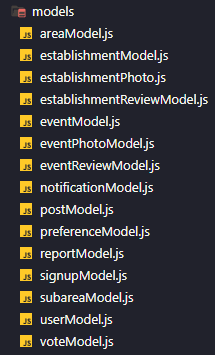
# Backend

O backend é a base da aplicação, responsável por gerenciar dados e lógica da mesma. Desenvolvido com Node.js e Express, ele inclui configuração de servidor, definição de rotas, controladores para manipulação de dados, modelos de dados com Postgres e middleware para autenticação e autorização.

## Models

Os modelos são definidos utilizando o Postgres e representam as entidades do sistema, como usuários, eventos e estabelecimentos. Cada modelo especifica a estrutura dos dados e as validações necessárias.

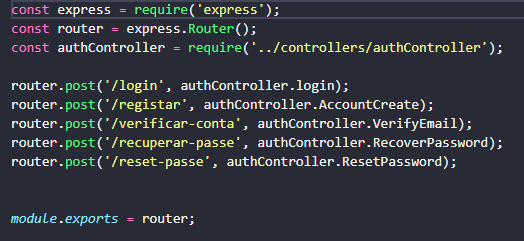
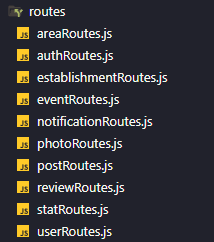
* **areaModel.js**: Define a estrutura dos dados para áreas, incluindo campos como nome e descrição.
* **establishmentModel.js**: Modelo para estabelecimentos, com informações como nome, localização e avaliações.
* **establishmentPhoto.js**: Armazena as fotos associadas aos estabelecimentos.
* **establishmentReviewModel.js**: Armazena as avaliações dos estabelecimentos.
* **eventModel.js**: Modelo para eventos, incluindo detalhes como título, data e descrição.
* **notificationModel.js**: Define a estrutura para notificações enviadas aos usuários.
* **postModel.js**: Armazena postagens feitas pelos usuários.
* **reviewModel.js**: Modelo para armazenar avaliações gerais.
* **statModel.js**: Armazena estatísticas relacionadas às atividades do usuário e do sistema.
* **userModel.js**: Modelo para usuários, contendo dados como nome, email, senha e perfil.
* **validateModel.js**: Armazena as informações de validação de dados inseridos no sistema.



## Rotas

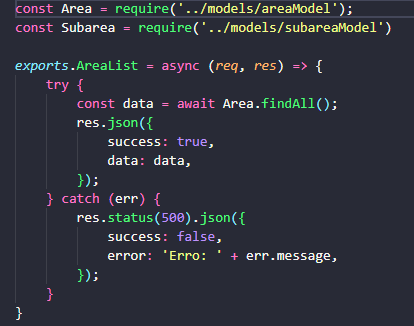
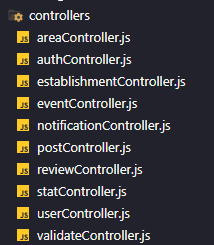
As rotas são pontos de entrada para as requisições HTTP. Cada rota é mapeada para um controlador específico que lida com a lógica da requisição.

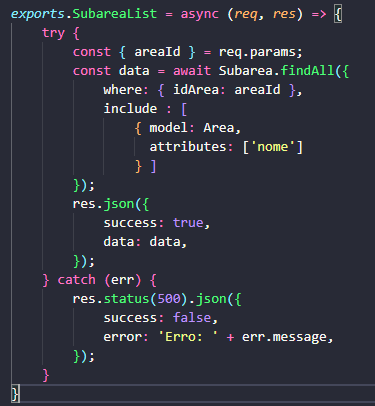
* **areaRoutes.js**: Inclui endpoints para operações CRUD em áreas.
* **userRoutes.js**: Gere operações de usuários, como o registro e o login.
* **establishmentRoutes.js**: Gere operações relacionadas a estabelecimentos.
* **eventRoutes.js**: Inclui endpoints para a criação, a atualização e a listagem de eventos.
* **notificationRoutes.js**: Gere notificações.
* **postRoutes.js**: Lida com operações de criação e gerenciamento de postagens.
* **reviewRoutes.js**: Inclui endpoints para avaliações.
* **statRoutes.js**: Gere estatísticas do sistema.
* **validateRoutes.js**: Lida com operações de validação de dados.



## Controllers

* **areaController.js**: Implementa a lógica para gerenciar áreas, como criar, atualizar e eliminar.
* **authController.js**: Lida com a autenticação de usuários, incluindo o login e o registro.
* **establishmentController.js**: Gere operações relacionadas a estabelecimentos, como a criação, a edição e a listagem.
* **eventController.js**: Implementa a lógica para criar, atualizar e listar eventos.
* **notificationController.js**: Lida com o envio e gerenciamento de notificações.
* **postController.js**: Gere o posto a qual o usuário está associado.
* **reviewController.js**: Lida com a lógica de criação e gerenciamento de avaliações.
* **statController.js**: Gere e atualiza estatísticas do sistema.
* **userController.js**: Gere operações de usuário, incluindo autenticação e atualização de perfil.
* **validateController.js**: Lida com a validação de dados inseridos no sistema.

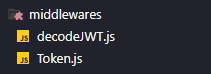




## Middleware

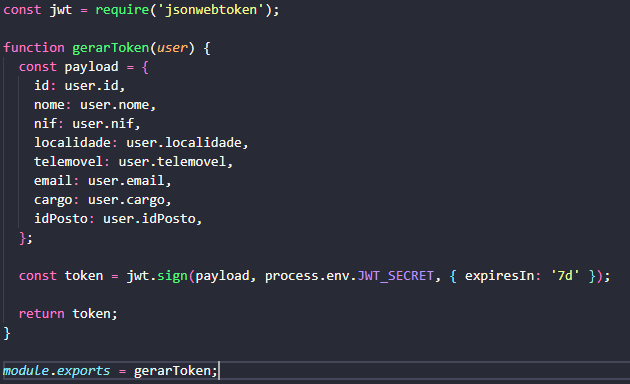
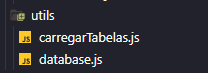
Os middlewares são funções intermediárias que processam requisições antes de chegarem aos controladores. Eles são usados para tarefas como a autenticação e a autorização.

* decodeJWT.js: Decodifica tokens JWT para verificar a identidade do usuário.
* Token.js: Verifica a validade do token de autenticação.



## Utils

A pasta “ utils” contém utilitários que são usados em várias partes do sistema para realizar tarefas comuns e facilitar o desenvolvimento.

* **carregarTabelas.js**: Script para carregar tabelas com dados iniciais no banco de dados. Ele facilita a inserção de dados necessários para testes e desenvolvimento, garantindo que o sistema tenha dados básicos para funcionar corretamente desde o início.
* **database.js**: Conecta ao postgres, isso permite que os respectivos models criem as tabelas necessárias para a gestão de dados.

# Frontend

## API

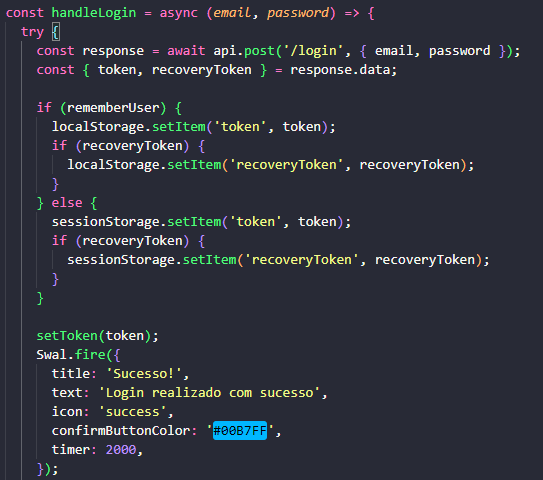
A API conecta o frontend e o backend, fornecendo os endpoints para as operações como criação de eventos, avaliação de estabelecimentos e autenticação de usuários. Ela é documentada para facilitar o desenvolvimento e a integração.



## Auth

Autenticação e autorização são essenciais para a segurança da aplicação. O sistema usa JWT (JSON Web Tokens) para gerenciar sessões de usuário. Middleware específicos garantem que apenas usuários autenticados possam acessar certas rotas.

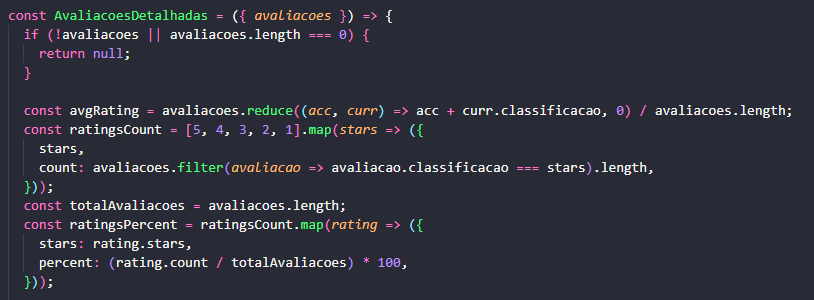
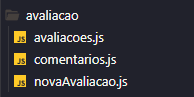




## Avaliação

A funcionalidade de avaliação permite que usuários avaliem estabelecimentos e eventos. Isso é gerenciado por controladores específicos e modelos que armazenam as avaliações no banco de dados.

* **reviewController.js**: Lida com a lógica de criação e gerenciamento de avaliações.
* **establishmentReviewModel.js**: Modelo para armazenar avaliações de estabelecimentos.
* **avaliacoes.js**: Mostra as avaliações num dado estabelecimento ou evento, bem como mostrar num grafico, as médias das avaliações totais.
* **comentarios.js**: Mostra os comentários num dado estabelecimento ou evento.



## 

## 

## 

## Calendário

O calendário permite a visualização e gerenciamento de eventos. Utiliza bibliotecas como FullCalendar para exibir eventos de forma intuitiva.

O template usado foi o calendário disponibilizado pelo adminLTE.



## 

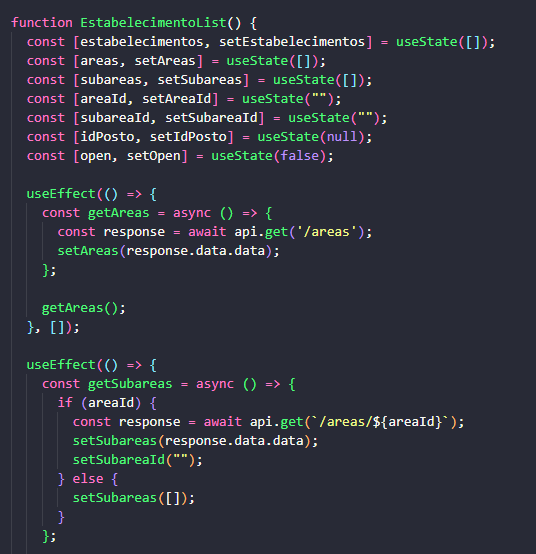
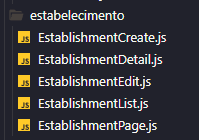
## 

## 

## 

## Estabelecimentos

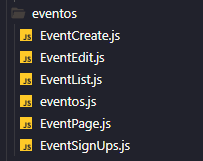
A funcionalidade de estabelecimentos inclui a criação, atualização e exibição de estabelecimentos no sistema. Cada estabelecimento pode ter avaliações e fotos associadas.



## 

## Eventos

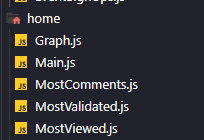
Gerenciamento de eventos, incluindo a criação, edição e exibição. Os eventos são exibidos no calendário e podem ser avaliados pelos usuários.



## Home

A página inicial agrega diversas funcionalidades, apresentando dados como os itens mais comentados, mais validados e mais visualizados.

* **Main.js**: Página principal após o login, abrange várias funções para facilitar a visualização dos itens mais vistos/comentados/validados.
* **MostComments.js**: Exibe os itens mais comentados.
* **MostViewed.js**: Exibe os itens mais visualizados.
* **MostValidated**.js: Exibe os itens mais validados.
* **Graph.js**: Exibe um gráfico das quantidades de eventos a acontecer brevemente, nos respectivos meses.



## 

## Notificação

O sistema de notificações mantém os usuários informados sobre eventos importantes e atualizações, enviando alertas conforme necessário.



## 

## 

## 

## 

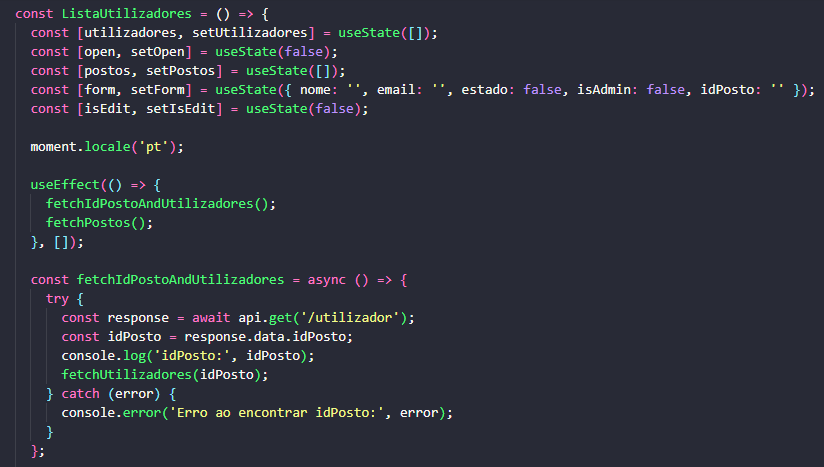
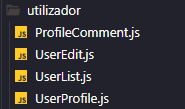
## 

## 

## 

## Utilizador

Funcionalidades relacionadas aos perfis de usuário, incluindo a capacidade de editar informações pessoais e visualizar o perfil.



# Conclusões

O desenvolvimento deste projeto proporcionou uma plataforma robusta e funcional para a gestão de eventos e estabelecimentos. Utilizando uma arquitetura bem definida, conseguimos integrar de forma eficaz o frontend e o backend, proporcionando uma experiência de usuário fluida e uma gestão eficiente dos dados. O uso de tecnologias modernas como React, Node.js e JWT garantiu uma aplicação segura e escalável.

Este relatório detalha as principais partes do sistema, explicando a implementação e o papel de cada componente, proporcionando uma visão clara do funcionamento da aplicação e dos desafios superados durante o desenvolvimento.