ENIDH - Computação Distribuída

Relatório de Desenvolvimento – Fase 1 e 2

GeoSolar: Aplicação Web Contentorizada com Acesso REST a Serviço Interno e Externo

Autores:
Paulo Oliveira - 13479
Rafael Cosme - 13475
Francisco Gonçalves - 14069

Curso: Licenciatura em Engenharia Informática e de Computadores
Instituição: Escola Superior Náutica Infante D. Henrique
Data: 21 de junho de 2025

Conteúdo

1 Introdução

Este relatório documenta o desenvolvimento da aplicação web **GeoSolar**, que permite visualizar, pesquisar e explorar estações solares a nível mundial. O projeto foi dividido em duas fases: a primeira focada na visualização e gestão local de dados geográficos, e a segunda na integração de serviços REST externos.

2 Objetivos do Projeto

- Desenvolver uma aplicação web responsiva para visualização de dados solares.
- Implementar uma arquitetura modular com contentores Docker a partir do projeto existente.
- Utilizar Swagger Codegen para gerar uma API RESTful.
- Integrar autenticação de utilizadores com ativação por email.
- Criar uma interface com mapa interativo e tabela pesquisável.
- Consumir serviços REST externos e integrá-los na aplicação.

3 Arquitetura da Aplicação

A aplicação é composta por três serviços principais, todos orquestrados com Docker Compose:

- **PostgreSQL (Utilizadores)**: Base de dados relacional que armazena as informações dos utilizadores.
- **PostgreSQL (Estações)**: Base de dados relacional de tabela única que armazena as estações solares.
- **API REST (Swagger)**: Criada com Swagger Codegen e Flask para servir dados e controlar a lógica da aplicação.
- **WebApp**: Aplicação em Flask que serve o frontend e comunica com a API.

3.1 Orquestração com Docker

Os serviços são definidos em `docker-compose.yml` para facilitar a instalação e execução em qualquer sistema com Docker.

Figura 1: Arquitetura dos serviços contentorizados no docker

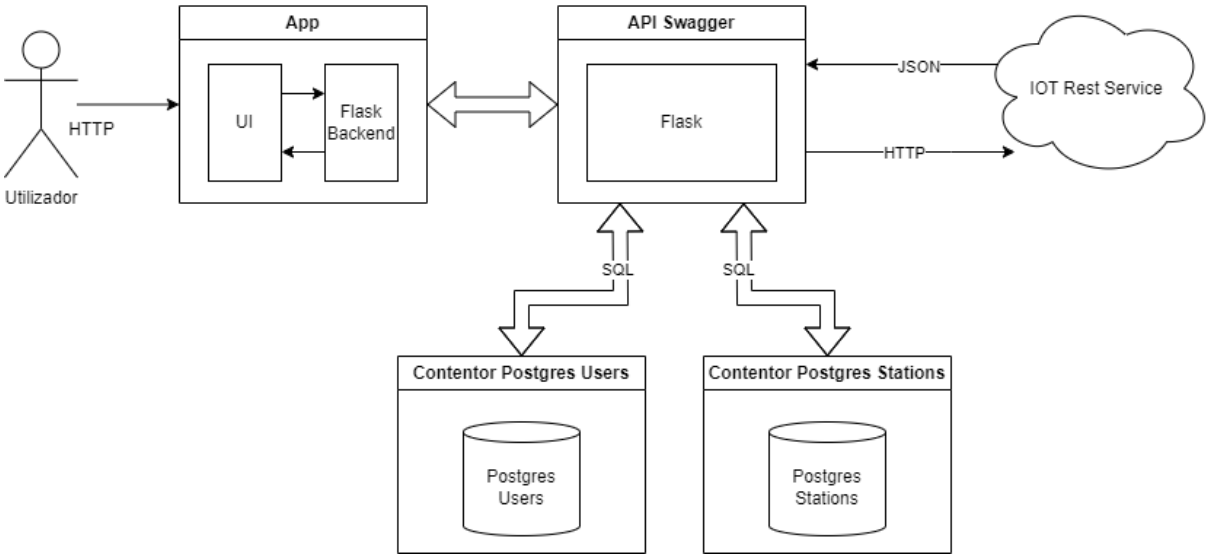

Figura 2: Diagrama dos serviços contentorizados

3.2 Importação de Dados

Os dados foram obtidos do repositório oficial:
https://globalenergymonitor.org/projects/global-solar-power-tracker/
E foram importados via DBeaver para PostgreSQL utilizando scripts SQL.

4 Funcionalidades da Aplicação (Fase 1)

4.1 Mapa Interativo

Foi utilizado `Leaflet.js` para construir um mapa mundial interativo onde cada estação é representada por um marcador com popup informativo.

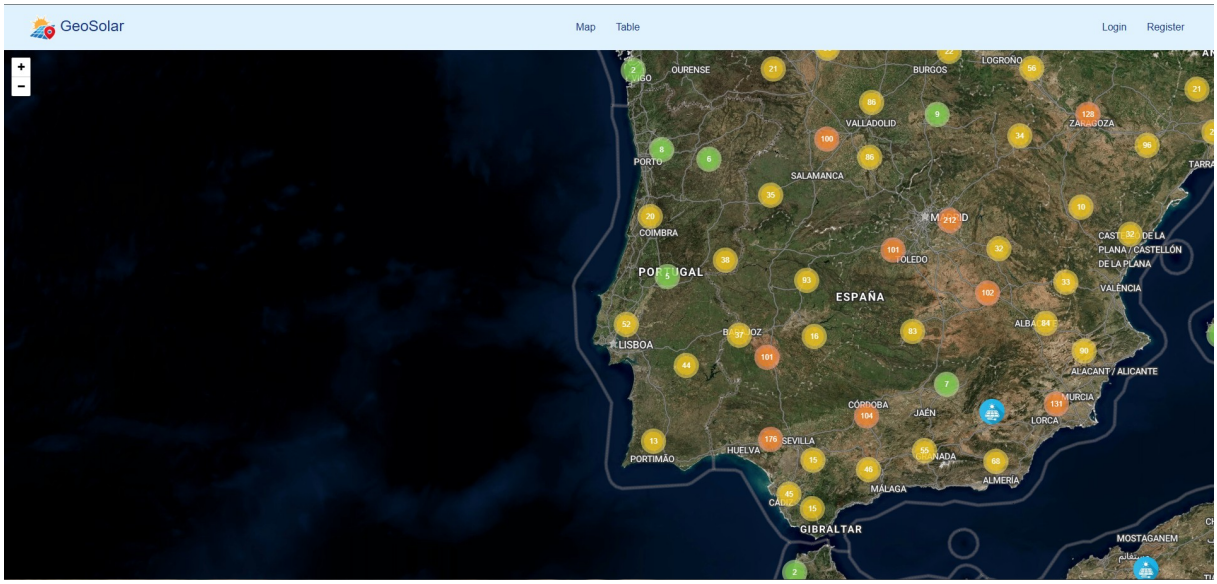

Figura 3: Funcionalidade - Mapa

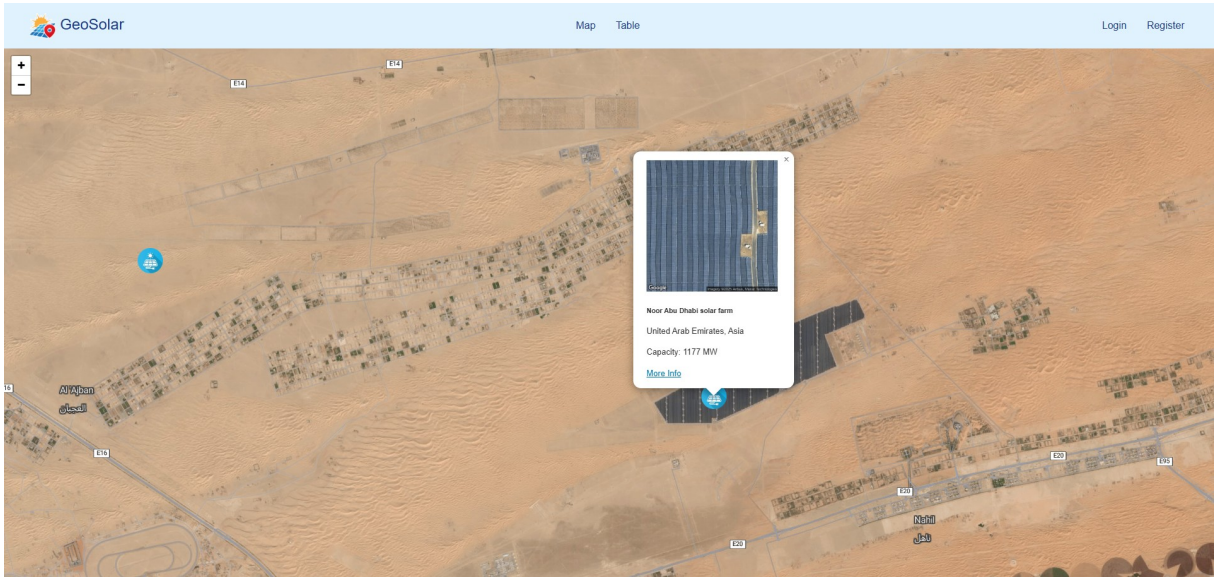

Figura 4: Funcionalidade - Mapa

4.2 Tabela Pesquisável

A aplicação apresenta uma tabela dinâmica com todas as estações, onde é possível filtrar por nome, país, capacidade, etc. Cada linha contém um botão para localizar a estação no mapa. Esta tabela é caracterizada pelo "lazy-loading"onde são carregadas 50 estações de cada vez (ordenadas pela capacidade em MW) sempre que o visualizador se aproxima do limite inferior da tabela.

Figura 5: Funcionalidade - Tabela

4.3 Sistema de Autenticação

Inclui:

- Registo de conta com verificação por email
- Login apenas após ativação
- Não há distinção de experiência de utilizador autenticado

5 Fase 2: Integração com Serviços REST

Dois serviços REST foram disponibilizados pelo docente. A aplicação passou a importar no ato do carregamento dados destes serviços e representá-los como novas estações no mapa, onde:

- IOT1: stream1/jpg weather/values weather/position
- IOT2: stream2/jpg socket/values socket/position

5.1 Processamento e Integração

Estes dados externos foram acessados apenas por REST, e não também por MQTT como no planeamento inicial.

5.2 Visualização

Estas novas "estações"são visíveis no mapa, distinguidas por um icone de uma relâmpago, e incluem imagens e informações adicionais para distinção.

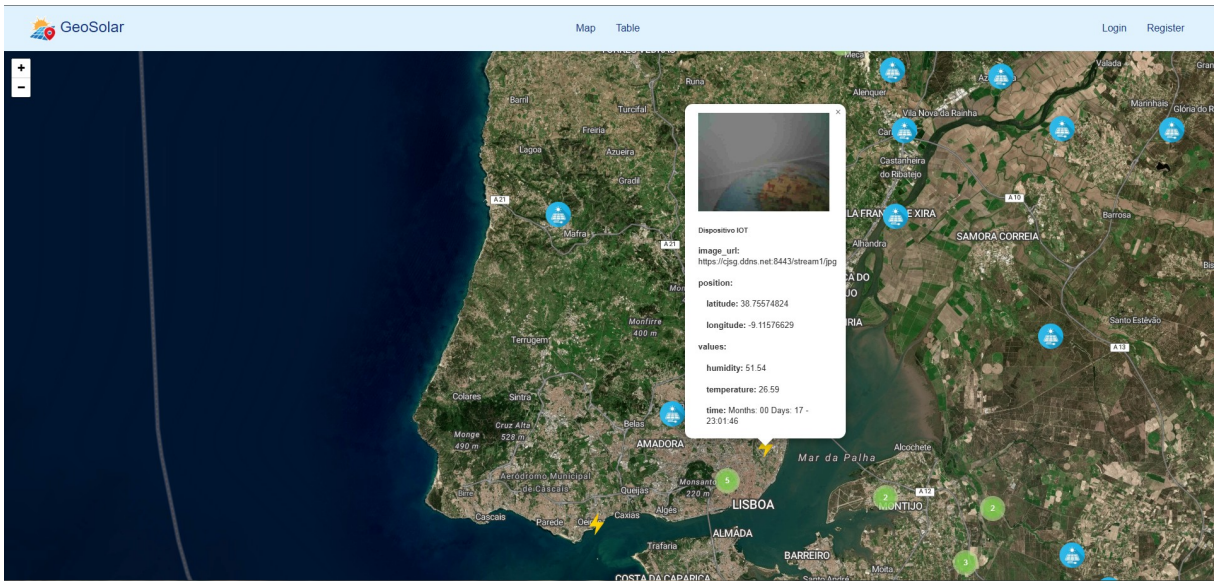

Figura 6: Dados IoT no Mapa

6 API REST: Swagger + Flask

A API REST utilizada na aplicação foi gerada automaticamente através do Swagger Codegen, com base numa especificação OpenAPI. A API é responsável por gerir o acesso às estações solares, utilizadores e operações auxiliares.

6.1 Descrição Geral

- **Base URL:** `/api`
- **Tecnologia:** Flask + Swagger Codegen
- **Formato de Resposta:** JSON
- **Autenticação:** Token (quando aplicável)

6.2 Endpoints

`Utilizadores`

Figura 7: Swagger Utilizadores

Estações, Health e IoT

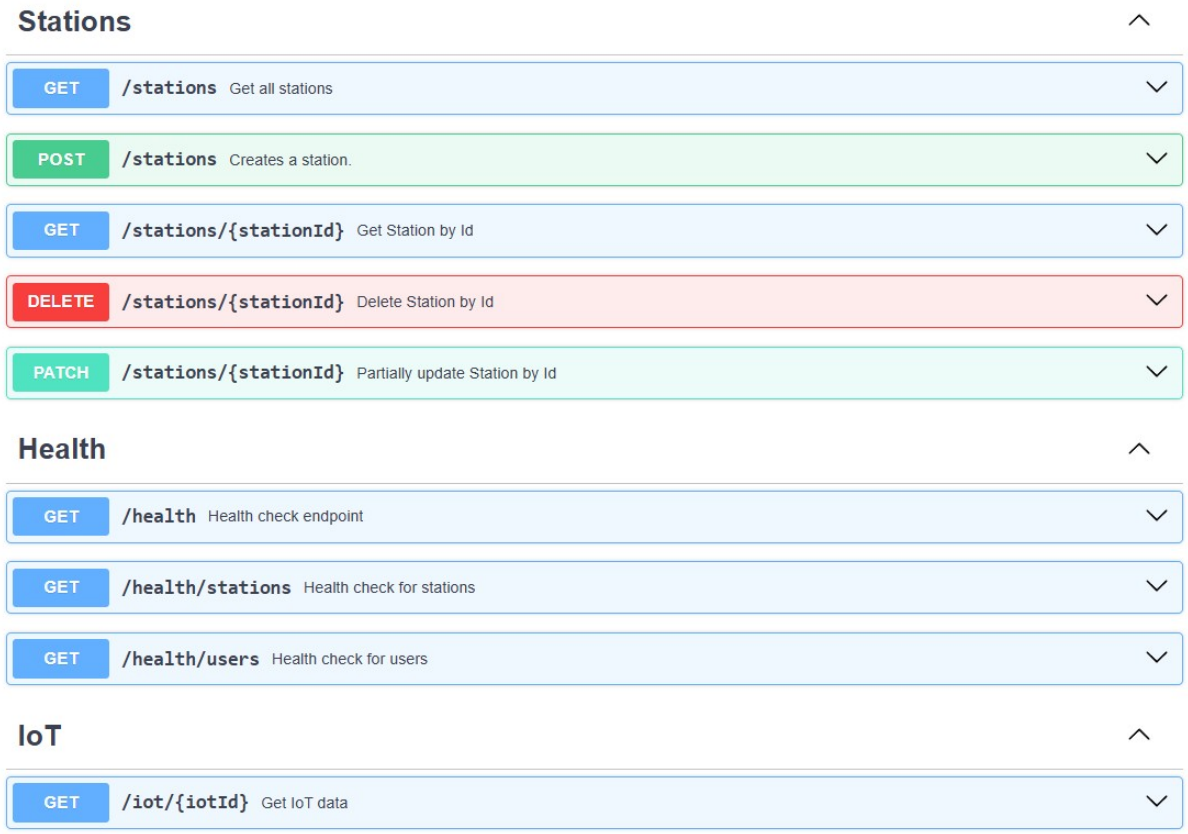

Figura 8: Swagger Estações, Health e IoT

6.3 Considerações

Todos os controladores e helpers gerados seguem a convenção RESTful. A separação entre frontend e backend é feita de forma clara, com a API a servir exclusivamente dados JSON.

A documentação completa da API pode ser acedida através do Swagger UI embutido na aplicação durante o desenvolvimento. Foi utilizado Swagger Codegen para gerar uma API RESTful com base numa especificação OpenAPI. A estrutura resultante inclui:

- Controladores: Definem as rotas da API.
- Models: Representações dos dados(schemas) transferidos.

7 Conclusões e Trabalho Futuro

A aplicação GeoSolar cumpre maior parte dos objetivos definidos para as duas fases. Destaca-se pela sua modularidade, usabilidade e integração de dados em tempo real.

7.1 Melhorias Futuras

- Autenticação com OAuth (Google, GitHub)
- Upload de estações via CSV
- Sistema de permissões por tipo de utilizador
- Dashboard estatístico com gráficos
- Cache de serviços REST externos
- Lazy-Loading do Mapa

Repositório

Código-fonte disponível em:
https://github.com/pjgoliveira21/GeoSolar/tree/main/Docker