

**Proposta de Trabalho Prático 2**

**Integração de Sistemas de Informação**

21110 – Flávio Pereira

**Licenciatura em Engenharia de Sistemas Informáticos**

**3ºano**

Barcelos | Dezembro, 2023

Conteúdo

[Introdução 3](#_Toc154845846)

[Processo de Documentação 4](#_Toc154845847)

[Documentação Criada pelo Doxygen 5](#_Toc154845848)

[Visão Geral do Projeto 6](#_Toc154845849)

[ETL 6](#_Toc154845850)

[Serviços de dados Covid 7](#_Toc154845851)

[Serviços de dados de Geolocalização 12](#_Toc154845852)

[Demonstração dos serviços dados Covid e Geolocalização. 14](#_Toc154845853)

[Serviços de Autenticação 15](#_Toc154845854)

[Demonstração serviços de Autenticação 19](#_Toc154845855)

[Dashboard 20](#_Toc154845856)

[Publicação na Cloud 25](#_Toc154845857)

[Conclusão 26](#_Toc154845858)

# Introdução

No decorrer deste projeto, foi utilizada uma extensa quantidade de dados coletados durante a pandemia de Covid-19 em diversos países da Europa, abrangendo o período de 2020 a 2022. O foco principal foi direcionado aos dados referentes ao ano de 2022, complementando-os com um serviço de geolocalização. Essa abordagem visa proporcionar uma compreensão mais aprofundada sobre a evolução da pandemia ao longo desse período, permitindo uma representação visual dos dados, por exemplo, em um mapa.

Esta etapa do projeto se integra à Fase 1, na qual foi demonstrada a utilidade das ferramentas de ETL( [Extract, transform, load](https://pt.wikipedia.org/wiki/Extract,_transform,_load) ) e solidificados os conceitos de integração de sistemas. Foram utilizados serviços web e integradas APIs externas, além da apresentação dos serviços por meio de um painel e disponibilização na nuvem ([Computação em nuvem](https://pt.wikipedia.org/wiki/Computa%C3%A7%C3%A3o_em_nuvem)).

Dessa forma, o projeto não apenas explora a manipulação e análise de dados relevantes da pandemia, mas também destaca a aplicação prática de ferramentas de integração, evidenciando a utilidade e eficácia desses serviços em um contexto mais amplo.

# Processo de Documentação

O processo de documentação foi realizado através do [Doxygen](https://pt.wikipedia.org/wiki/Doxygen) e [Graphviz](https://graphviz.org/), uma ferramenta eficiente para gerar documentação a partir de código-fonte. A escolha do Doxygen baseou-se na sua capacidade de analisar automaticamente o código e criar documentação consistente e bem formatada.

Esse processo consistiu nos seguintes passos:

1. **Integração do Doxygen:** O Doxygen foi integrado ao projeto para analisar o código-fonte automaticamente.
2. **Configuração do Doxyfile:** Um ficheiro de configuração (Doxyfile) foi criado para especificar as configurações desejadas para a criação da documentação.
3. **Marcação do Código:** O código-fonte foi marcado com comentários especiais Doxygen para descrever classes, métodos, parâmetros e outros elementos significativos.
4. **Execução do Doxygen:** O Doxygen foi executado para processar os comentários marcados e criar documentação em diferentes formatos como [HTML](https://pt.wikipedia.org/wiki/HTML) e [LaTeX](https://pt.wikipedia.org/wiki/LaTeX) (que foi usado para criar um ficheiro em formato PDF), utilizando também o Graphviz para gerar gráfico.

# Documentação Criada pelo Doxygen

Documentação HTML: A documentação em HTML pode ser acessada na pasta HTML criada pelo Doxygen (presente dentro da pasta de projeto) onde a documentação foi no ficheiro "index.html". A documentação em HTML é mais amigável e interativa, permitindo uma procura mais fácil pelas diferentes seções do projeto.

Documentação em PDF: Para a versão em PDF, **foi utilizado** o [MiKTeX](https://miktex.org/) para instalar os componentes necessários e o [TeXworks](https://tug.org/texworks/) para compilar o ficheiro LaTeX gerado pelo Doxygen. O ficheiro PDF está incluído na pasta do projeto. A documentação em PDF fornece uma versão estática e portátil da documentação.

# Visão Geral do Projeto

Este projeto consiste em uma REST API destinada à gestão e análise de dados relacionados à pandemia de Covid-19 para além da criação de uma Dashboard para utilização da mesma. Abaixo, destacam-se os principais componentes e funcionalidades do projeto:

## ETL

Inicialmente foi utilizado um CSV [Dataset](https://www.ecdc.europa.eu/en/publications-data/covid-19-testing) que incluiu informações relativas à testagem da Covid-19 na época da pandemia (2020-2022), que continha dados convenientes, tais como, País, Região, Ano-Semana, Nivel de Testagem, Novos Casos, Testes Efetuados, População, Taxa de Posivitidade e Origem da Testagem. Foi utilizado o MySQL para guardar dados relativos aos anos numa base de dados.

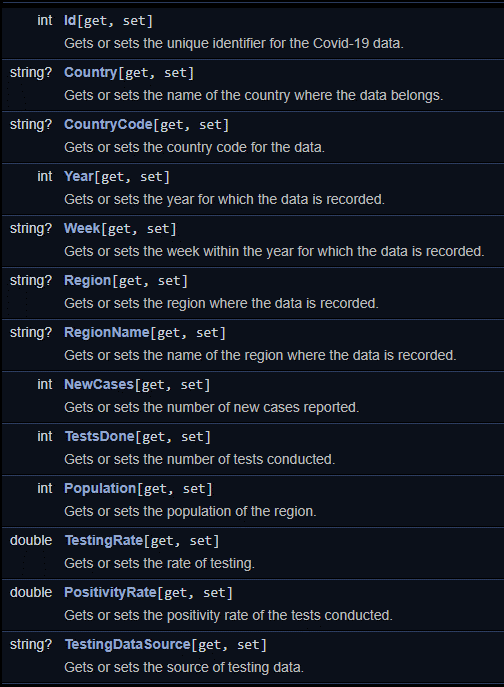
A primeira transformação recebeu o ficheiro CSV como data input, contudo os dados neste dataset precisavam de ser limpos e estruturados, então foram efetuadas uma série de operações de Seleção, Separação e Filtragem para alterar o desejado.

Uma imagem com diagrama, mapa, texto, file

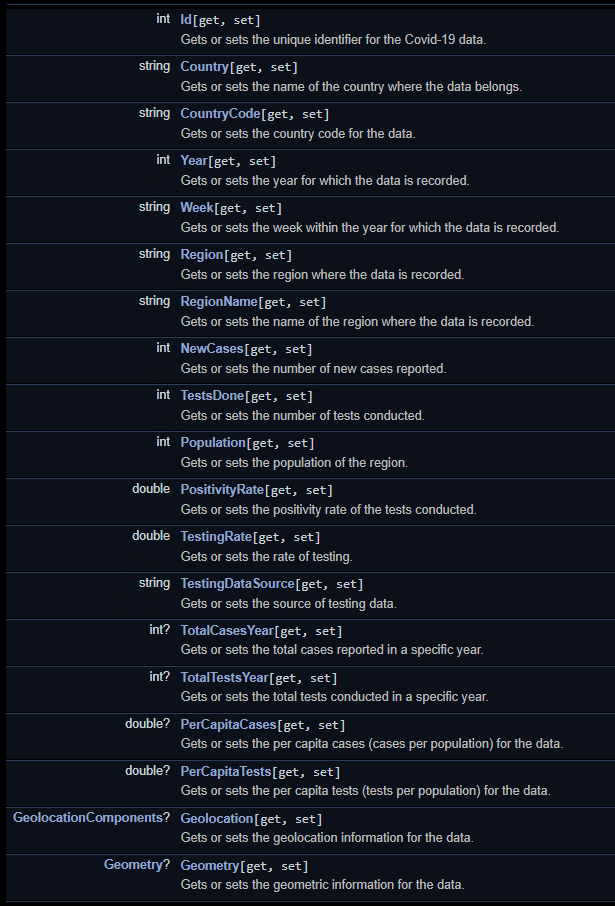
Descrição gerada automaticamente

## Serviços de dados Covid

Foram criados dois modelos, primeiramente a classe [*CovidAPI.Models.CovidData*](https://flaww1.github.io/CovidAPI/html/d5/d96/class_covid_a_p_i_1_1_models_1_1_covid_data.html) representa dados da COVID-19 armazenados na base de dados. Inclui as seguintes propriedades:



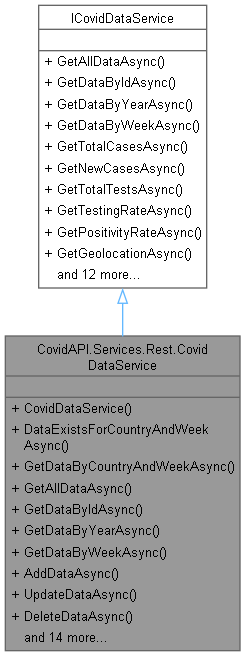
E a classe [*CovidAPI.Models.CovidDataDTO*](https://flaww1.github.io/CovidAPI/html/d8/dd7/class_covid_a_p_i_1_1_models_1_1_covid_data_d_t_o.html#details) representa dados da COVID-19 armazenados na base de dados. Inclui as seguintes propriedades:



O uso de dois modelos pode ser justificado pela necessidade de separar a representação interna dos dados na base de dados da forma como esses dados são apresentados ao utilizador ou consumidor externo.

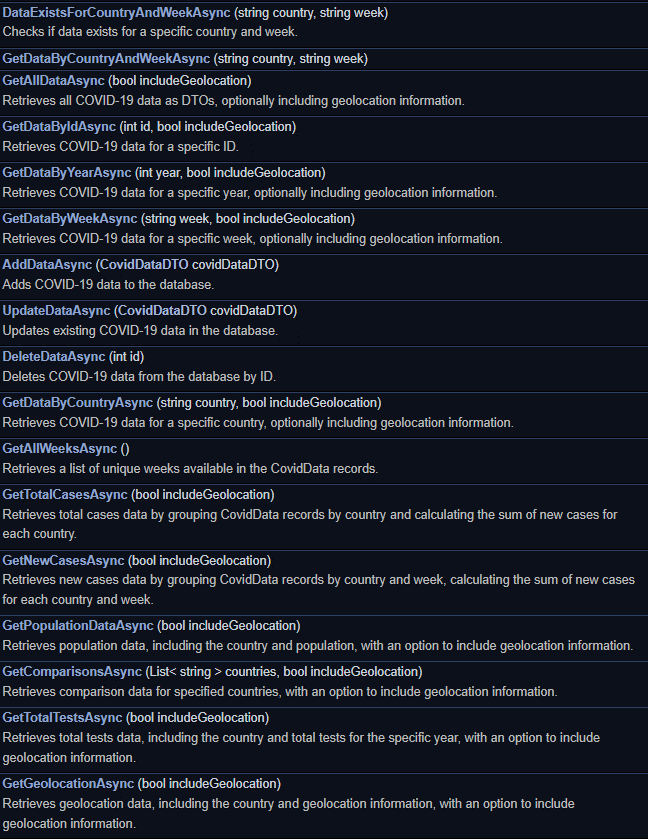
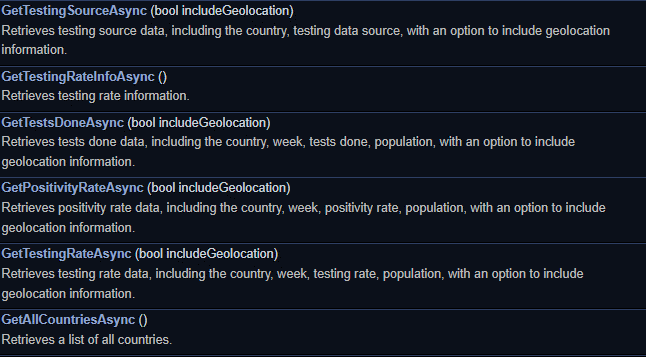
O modelo **CovidData** pode ser otimizado para atender aos requisitos de armazenamento e manipulação de dados no backend, enquanto o modelo **CovidDataDTO** é projetado para fornecer uma estrutura clara e simplificada para apresentação, evitando o envio de dados excessivos ou sensíveis ao cliente. Essa abordagem favorece a modularidade e a eficiência na comunicação entre os diferentes componentes do sistema.

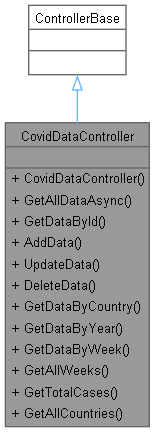
De seguida foi criado o serviço [*CovidAPI.Services.Rest.*CovidDataService](https://flaww1.github.io/CovidAPI/html/db/d89/class_covid_a_p_i_1_1_services_1_1_rest_1_1_covid_data_service.html) que desempenha um papel importante na aplicação, é responsável por manipular operações relacionadas a dados da COVID-19, incluindo procura, atualização e exclusão de dados.



A seta que vai do **CovidDataService** até a interface **ICovidDataService** representa a relação de implementação. Isso significa que o **CovidDataService** está a implementar as funcionalidades definidas na interface **ICovidDataService**. Essa relação é um acordo, onde o serviço promete fornecer as operações especificadas pela interface.

O serviço contém gama abrangente de funções:



O [*CovidDataController*](https://flaww1.github.io/CovidAPI/html/d7/dcf/class_covid_data_controller.html) é o controlador responsável por operações relacionadas a dados da COVID-19. Atua como uma interface para lidar com requisições relacionadas a esses dados.

O **CovidDataController** é um controlador que herda de **ControllerBase**. No contexto do [ASP.NET Core](https://dotnet.microsoft.com/pt-br/learn/aspnet/what-is-aspnet-core), **ControllerBase** é uma classe base para controladores da Web API.

A seta da imagem que vai do **CovidDataController** para **ControllerBase** indica que **CovidDataController** herda de **ControllerBase**. Essa herança implica que **CovidDataController** possui todas as funcionalidades de **ControllerBase** e, portanto, pode ser tratado como um controlador da Web API no contexto do ASP.NET Core.

Essa relação de herança é fundamental no desenvolvimento do ASP.NET Core, pois fornece ao **CovidDataController** acesso às funcionalidades básicas de um controlador, permitindo que responda a requisições [HTTP](https://pt.wikipedia.org/wiki/Hypertext_Transfer_Protocol), manipule parâmetros, retorne resultados e muito mais, de acordo com os padrões da arquitetura [RESTful](https://aws.amazon.com/pt/what-is/restful-api/) da Web API.

Uma imagem com texto, captura de ecrã, Tipo de letra

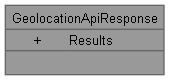
Descrição gerada automaticamenteFunções do controlador:

## Serviços de dados de Geolocalização

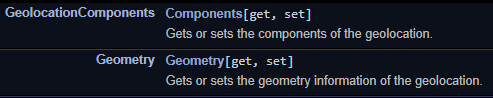
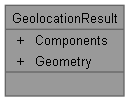
É uma componente do sistema responsável por lidar com operações relacionadas à geolocalização. Este serviço é utilizado para interagir com uma API de geolocalização, neste caso a [OpenCage Geocoding API](https://opencagedata.com/) para obter informações sobre determinadas coordenadas geográficas dado a entrada de um nome de um país.

Foi criados múltiplos modelos: [GeolocationAPIResponse](https://flaww1.github.io/CovidAPI/html/d3/d27/class_geolocation_api_response.html), [GeolocationResult](https://flaww1.github.io/CovidAPI/html/d5/dfc/class_geolocation_result.html), [GeolocationComponents](https://flaww1.github.io/CovidAPI/html/df/dc6/class_geolocation_components.html) e [Geometry](https://flaww1.github.io/CovidAPI/html/db/d0b/class_geometry.html).

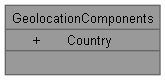
**GeolocationAPIResponse** representa a resposta da API de geolocalização.



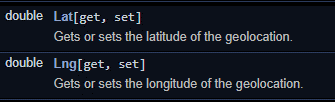
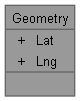
**GeolocationResult** representa o resultado de geolocalização.



Uma imagem com texto, Tipo de letra, captura de ecrã, file

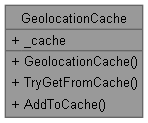
Descrição gerada automaticamente**GeolocationComponents** representa os componentes de uma geolocalização. 

**Geometry** representa as informações de geometria de uma geolocalização.



O principal propósito desse serviço é facilitar a obtenção, atualização e gestão de dados de geolocalização.

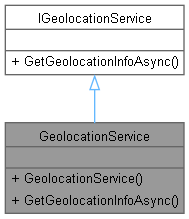
Uma imagem com texto, captura de ecrã, Tipo de letra

Descrição gerada automaticamenteFoi decidido também incluir uma classe [GeolocationCache](https://flaww1.github.io/CovidAPI/html/de/d39/class_geolocation_cache.html) que representa um cache utilizado para armazenar dados de geolocalização, com o objetivo de evitar chamadas redundantes à API armazenando temporariamente os resultados num ficheiro. 

Este diagrama visualiza a estrutura da classe **GeolocationCache** e como ela lida com a adição e consulta de dados de geolocalização em um cache interno.

Aqui está um exemplo de como os dados são armazenados no ficheiro ***cache.json*** para a Chéquia: 

*O pedido completo de dados covid-19 junto com a integração do serviço de geolocalização vai ser futuramente exemplificado.*

A classe [GeolocationService](https://chat.openai.com/c/e50b2671-f535-41fa-b903-46866ace680a) representa o serviço para obter informações de geolocalização específicas de um país

A seta no gráfico, de **GeolocationService** para **IGeolocationService**, indica que a classe **GeolocationService** implementa a interface **IGeolocationService**. Isto promove flexibilidade e manutenção de código.

Em suma, a integração do serviço de geolocalização com o serviço de dados COVID cria uma sinergia valiosa para o sistema. Ao unir esses serviços, podem-se enriquecer os dados epidemiológicos com informações geográficas relevantes. Isso não apenas aprimora a precisão das análises e visualizações, mas também oferece insights geolocalizados cruciais para entender a propagação da COVID-19.

## Demonstração dos serviços dados Covid e Geolocalização.

Uma imagem com captura de ecrã, texto, Tipo de letra, file

Descrição gerada automaticamenteUtilizando o [Postman](https://www.postman.com/), aqui está a exemplificação de um Pedido GET de dados da semana 41:

Uma imagem com texto, captura de ecrã, menu, Tipo de letra

Descrição gerada automaticamenteUma imagem com texto, captura de ecrã, menu

Descrição gerada automaticamente

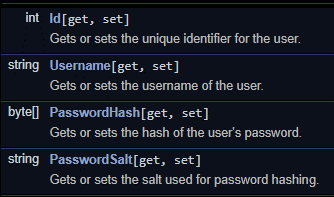
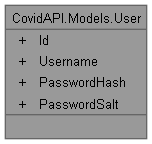
Esta resposta é um conjunto de dados formatados em JSON, representando informações específicas sobre a situação da COVID-19.

Contém um identificador único para o conjunto de dados, o nome do país associado aos dados, o código do país, o ano ao qual os dados se referem, a semana dentro do ano, o nome e código da região (não explorada neste projeto mas pode ser utilizado para melhor precisão de dados), novos casos da semana, numero total de testes da semana, população total do país, taxa de positividade, taxa de testagem (número de testes realizados por 1000 pessoas), fonte de dados, número total de casos registados durante o ano, número total de testes realizados durante o ano, número de casos e testes por pessoa, e informações de geolocalização/coordenadas geográficas do país (latitude e longitude).

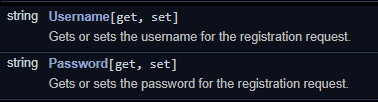
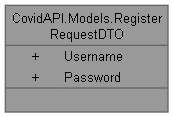
## Serviços de Autenticação

Para o serviço de autenticação foram definidos três modelos: [CovidAPI.Models.User](https://flaww1.github.io/CovidAPI/html/db/ddf/class_covid_a_p_i_1_1_models_1_1_user.html), [CovidAPI.Models.RegisterRequestDTO](https://flaww1.github.io/CovidAPI/html/d5/d91/class_covid_a_p_i_1_1_models_1_1_register_request_d_t_o.html) e [CovidAPI.Models.LoginRequestDTO](https://flaww1.github.io/CovidAPI/html/d0/d21/class_covid_a_p_i_1_1_models_1_1_login_request_d_t_o.html).

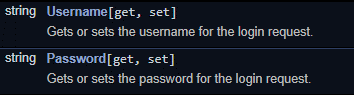
**User** armazena informações sobre os utilizadores.



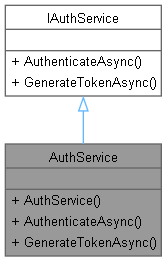
**RegisterRequestDTO** é utilizado para transferir dados relacionados ao registo de um utilizador.



**LoginRequestDTO** também é utilizado para transferir dados, mas desta vez para solicitações de login.



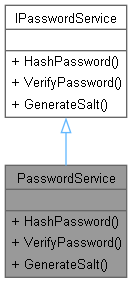
A junção dos três modelos cria uma estrutura coesa para o serviço de autenticação na aplicação.

A classe [AuthService](https://flaww1.github.io/CovidAPI/html/d1/de2/class_auth_service.html) é responsável pela autenticação do utilizador e pela criação de tokens [JWT](https://jwt.io/).

Uma imagem com texto, captura de ecrã, Tipo de letra

Descrição gerada automaticamenteA seta indica que a classe **AuthService** implementa todos os membros (métodos, propriedades, etc.) definidos pela interface **IAuthService**.

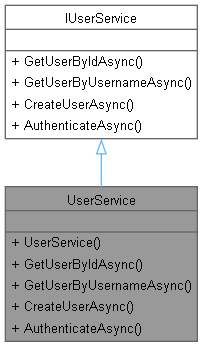
Esta classe “trabalha” com outros serviços para garantir a segurança e eficiência de autenticação da aplicação:

Com a classe [PasswordService](https://flaww1.github.io/CovidAPI/html/d9/df0/class_password_service.html), que é responsável por operações relacionadas à password, como o hash da password, verificação de password e criação de salt utilizando [BCrypt](https://www.npmjs.com/package/bcrypt) (biblioteca de encriptação).

Uma imagem com texto, captura de ecrã, Tipo de letra

Descrição gerada automaticamenteA seta indica que a classe **PasswordService** implementa todos os membros (métodos, propriedades, etc.) definidos pela interface **IPasswordService**.

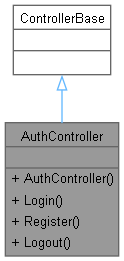
Estas operações são úteis em contextos de autenticação, onde é crucial armazenar passwords de forma segura e eficiente, minimizando riscos de exposição.

A classe [UserService](https://flaww1.github.io/CovidAPI/html/d8/d40/class_user_service.html) é responsável por operações relacionadas aos utilizadores incluindo autenticação e recuperação de informações do utilizador.

Uma imagem com texto, captura de ecrã, Tipo de letra

Descrição gerada automaticamenteA seta indica que a classe **UserService** implementa todos os membros (métodos, propriedades, etc.) definidos pela interface **IUserService**.

Estas operações são essenciais para a gestão de utilizadores, permitindo a autenticação segura, recuperação de informações do utilizador e criação de novos utilizadores no sistema. O uso de operações assíncronas reflete a natureza potencialmente demorada de algumas dessas operações, como a interação com uma base de dados.

A classe [AuthController](https://flaww1.github.io/CovidAPI/html/df/da5/class_auth_controller.html) é o controlador responsável por operações relacionadas à autenticação.

O **AuthController** é um controlador que herda de **ControllerBase**, já explicado anteriormente

Uma imagem com texto, captura de ecrã, Tipo de letra, número

Descrição gerada automaticamenteA seta da imagem que vai do **AuthController** para **ControllerBase** indica que **AuthController** herda de **ControllerBase**. Essa herança implica que **AuthController** possui todas as funcionalidades de **ControllerBase** e, portanto, pode ser tratado como um controlador da Web API no contexto do ASP.NET Core.

Estas operações são fundamentais para a gestão da autenticação de utilizadores. O método Login gera um token JWT após autenticar com sucesso um utilizador. O método Logout encerra a sessão do utilizador autenticado. O método Register cria um novo utilizador e fornece um token JWT em caso de sucesso.

## Demonstração serviços de Autenticação

Novamente, utilizando o [Postman](https://www.postman.com/), aqui está a exemplificação do registo e login de um utilizador.

Uma imagem com texto, captura de ecrã, Tipo de letra, file

Descrição gerada automaticamente

Uma imagem com texto, captura de ecrã, Tipo de letra, file

Descrição gerada automaticamente **Entrada**:

**Resposta**: Atribuição de um ID e mensagem de sucesso

Uma imagem com texto, captura de ecrã, Tipo de letra, file

Descrição gerada automaticamenteUma imagem com texto, captura de ecrã, Tipo de letra

Descrição gerada automaticamente

Uma imagem com texto, captura de ecrã, Tipo de letra

Descrição gerada automaticamente**Resposta**: Token JWT é criado após login.

O utilizador é registado na base de dados com uma password encriptada.



## Dashboard

Para a criação da Dashboard foi utilizada a framework [React](https://react.dev/) em javascript.

Foi utilizada a biblioteca [axios](https://axios-http.com/docs/intro) para interagir com os endpoints da API, a biblioteca [leaflet.js](https://leafletjs.com/) para a utilização de mapas em conjunto com o [openstreetmap](https://www.openstreetmap.org/), e a biblioteca [recharts](https://recharts.org/en-US/) para inclusão de gráficos.

**Login/Registo:**

Uma imagem com texto, software, Ícone de computador, captura de ecrã

Descrição gerada automaticamente

Uma imagem com texto, design, captura de ecrã

Descrição gerada automaticamente

**Página Principal:**

Na página principal é possível selecionar uma semana e ver os dados de cada país num mapa e algumas métricas referentes aos dados covid dessa semana. Cada país e marcado com um ponto na devida latitude e longitude fornecida pela API.

Por exemplo, semana 1

Uma imagem com texto, captura de ecrã, Tipo de letra

Descrição gerada automaticamente

Uma imagem com mapa, texto, atlas

Descrição gerada automaticamente

Ao clicar num ponto, dados sobre a semana nesse país, são mostrados.

Uma imagem com texto, mapa, atlas, captura de ecrã

Descrição gerada automaticamente

Uma imagem com texto, número, recibo, captura de ecrã

Descrição gerada automaticamenteÉ possível adicionar pontos também, por exemplo, Portugal não tem um ponto na semana 1, se criarmos um ponto com dados na semana 1 para Portugal a API automaticamente vai buscar os dados relativos à geolocalização em Portugal, guarda na cache da API e o ponto é assim colocado com os dados relativos.

Uma imagem com texto, mapa, atlas, captura de ecrã

Descrição gerada automaticamenteFoi gerado o ponto para Portugal na semana 1:

Também é possível apagar e atualizar pontos:

Uma imagem com texto, captura de ecrã, Tipo de letra, número

Descrição gerada automaticamenteUma imagem com texto, captura de ecrã, Tipo de letra, número

Descrição gerada automaticamenteAtualização do valor de novos casos em Portugal:

Deleção do ponto:

Uma imagem com mapa, atlas, texto

Descrição gerada automaticamente

Isto coloca em prática uma variedade de rotas e funções presentes na API .

Mais abaixo, também se pode selecionar um país e a respetiva métrica para listar os dados relativos a todas as semanas desse ano num gráfico.

Por exemplo, Bélgica – Novos Casos

Uma imagem com texto, captura de ecrã, diagrama, Gráfico

Descrição gerada automaticamente

Uma imagem com texto, file, captura de ecrã, Tipo de letra

Descrição gerada automaticamenteApesar de o gráfico não estar a 100% e estar praticamente a mostrar dados em loop, demonstra que é possível utilizar os dados para obter informação histórica. Outro exemplo:

## Publicação na Cloud

A publicação no [Microsoft Azure](https://azure.microsoft.com/pt-pt/) foi um pouco problemática, umas vezes publicava outras não, era mostrado um erro de swagger e um pacote .net que ficou por resolver, mas apesar dos endpoints não funcionarem na Cloud, o swagger foi lido.

Uma imagem com texto, captura de ecrã, software, Ícone de computador

Descrição gerada automaticamente

No geral gostava bastante de experimentar trabalhar em Cloud visto que é algo que me despertou interesse mas infelizmente não foi possível explorar tudo adequadamente neste projeto.

Também procurei utilizar [Docker](https://www.docker.com/) mas não tive essa oportunidade.

# Conclusão

O desenvolvimento deste projeto abordou de forma abrangente a gestão e análise de dados relacionados à pandemia de Covid-19, enfatizando a integração de diversas ferramentas e serviços. A seguir, destacam-se algumas conclusões relevantes:

* **Abordagem Abrangente:** O projeto envolveu uma extensa coleta de dados da pandemia de Covid-19 na Europa, com foco no ano de 2022. A utilização de um serviço de geolocalização complementou esses dados, permitindo uma representação visual em mapas.
* **Integração de Sistemas e Ferramentas:** A Fase 1 do projeto demonstrou a utilidade das ferramentas ETL (Extract, Transform, Load) e solidificou os conceitos de integração de sistemas. A integração de serviços web, APIs externas e a apresentação por meio de um painel contribuíram para uma compreensão mais ampla.
* **Documentação Eficiente:** O processo de documentação, realizado pelo Doxygen, proporcionou uma visão clara da estrutura do código-fonte. A documentação em HTML e PDF facilita a compreensão e manutenção do projeto.
* **Serviços de Dados Covid e Geolocalização:** A implementação de serviços dedicados aos dados da Covid-19 e geolocalização trouxe uma abordagem modular. A separação de modelos internos e externos permitiu uma melhor adaptação às necessidades de armazenamento e apresentação.
* **Demonstrações Práticas:** A utilização do Postman para exemplificar solicitações GET de dados e autenticação ofereceu uma visão prática da funcionalidade da API. As demonstrações na Dashboard, com interação de mapas e gráficos, adicionaram uma camada visual e interativa aos dados.
* **Serviços de Autenticação:** A implementação de um serviço de autenticação, com modelos específicos e criação de tokens JWT, contribuiu para a segurança e controlol de acesso à aplicação.
* **Desafios e Oportunidades Futuras:** Os desafios encontrados na publicação na nuvem e a exploração limitada do Docker representam oportunidades de aprimoramento futuro. A utilização de serviços em nuvem, como o Microsoft Azure, pode ser uma direção promissora para a expansão do projeto.

Em suma, o projeto não apenas atingiu seus objetivos principais relacionados à análise de dados da Covid-19, mas também evidenciou a importância da integração de serviços, documentação eficiente e a aplicação prática de conceitos em um contexto mais amplo.