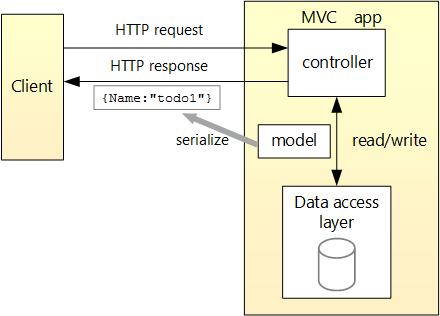
Desenvolvimento de API Web

Desafio Tecnológico

Neste desafio tecnológico você deverá implementar uma API Web usando que disponibilize os dados armazenados em um banco de dados relacional ou não relacional, no formato json. A aplicação Web deverá estar estruturada em uma arquitetura MVC conforme o diagrama abaixo:



# Sistema de Gerenciamento de Aeroportos

Um aeroporto é um aeródromo com instalações estendidas, principalmente para transporte aéreo comercial. Os aeroportos geralmente consistem em uma área de pouso, que compreende um espaço aberto acessível por via aérea, incluindo pelo menos uma superfície operacionalmente ativa, como uma pista para um avião decolar e pousar ou um heliporto, e muitas vezes inclui utilitários adjacentes edifícios como torres de controle, hangares e terminais, para manutenção e monitoramento de aeronaves. Aeroportos maiores podem ter pátios aeroportuários, pontes de taxiway, centros de controle de tráfego aéreo, instalações de passageiros, como restaurantes e lounges, e serviços de emergência. Em alguns países, em particular nos EUA, os aeroportos também costumam ter um ou mais operadores de base fixa, atendendo à aviação geral.

**Fonte**: WIKIPEDIA. Airport. Disponível em <https://en.wikipedia.org/wiki/Airport>

Crie uma API REST que forneça os dados no formato json, para gerenciar o cadastro de aeroportos espalhados pelo mundo todo.

# Visão Geral da API

Esta seção descreve as principais características da API que será implementada.

| **HTTP** | **URL** | **Descrição** |
| --- | --- | --- |
| **GET** | **/api/v1/aeroportos** | Obter todos os aeroportos. |
| **GET** | **/api/v1/aeroportos/{iata}** | Obter um aeroporto pelo código aeroportuário IATA. |
| **POST** | **/api/v1/aeroportos** | Adicionar um novo aeroporto |
| **PUT** | **/api/v1/aeroportos/{iata}** | Atualizar um aeroporto existente. |
| **DELETE** | **/api/v1/aeroportos/{iata}** | Excluir um aeroporto. |

# Dicionário de Dados

Esta seção apresenta definições e representações dos itens de dados que serão processados pela aplicação. As informações de metadados apresentados neste dicionário de dados incluem o nome do campo, o tipo de dados, se o campo pode estar vazio ou não, ou seja, se ele é obrigatório ou não, os valores máximo e mínimo aceitos, o valor padrão inicial daquele determinado campo, bem como, informações adicionais na coluna observação.

| **Campo** | **Tipo** | **Descrição** |
| --- | --- | --- |
| **id\_aeroporto** | Inteiro | Chave primária que identifica de maneira única cada aeroporto. |
| **nome\_aeroporto** | Texto | Nome do aeroporto. |
| **codigo\_iata** | Texto | O código aeroportuário IATA é uma sigla composta por três letras, utilizada para designar os aeroportos em todo o mundo. |
| **cidade** | Texto | Nome da cidade onde está localizado o aeroporto. |
| **codigo\_pais\_iso** | Texto | Código (ISO 3166-1) para nomes de países que utiliza um sistema de 2 letras (alfa-2), por exemplo: Brasil (BR), Estados Unidos (US) |
| **latitude** | Real | Um número real representando a latitude. |
| **longitude** | Real | Um número real representando a longitude. |
| **altitude** | Real | Um número real representando a altitude. |

No website **Open Flights** <https://openflights.org/data.html>, você encontrará disponível para download gratuitamente, um conjunto de dados com informações dos aeroportos para você testar sua aplicação.

# Tecnologias

A API REST poderá ser implementada usando qualquer linguagem de programação orientada a objetos ou qualquer framework web que dê suporte ao padrão MVC. Porém, o(s) professor(es) poderão dar suporte apenas nas tecnologias vistas em sala de aula.

# Links e Recursos Úteis

* CRUD REST using Spring Boot 2, Hibernate, JPA, and MySQL: <https://www.oracle.com/br/technical-resources/articles/dsl/crud-rest-sb2-hibernate.html>
* Tutorial: criar uma API Web com o ASP.NET Core: <https://learn.microsoft.com/pt-br/aspnet/core/tutorials/first-web-api?view=aspnetcore-6.0&tabs=visual-studio-code>
* REST API TUTORIAL: <https://www.restapitutorial.com/>

# Glossário

Este glossário fornece definições de palavras e abreviações que você precisa saber para entender e construir o software com sucesso.

* Código IATA: O código aeroportuário IATA é uma sigla composta por três letras, utilizada para designar os aeroportos em todo o mundo. É definido pela Associação Internacional de Transportes Aéreos (em inglês: International Air Transport Association — IATA), que tem sede em Toronto, no Canadá
* REST: REST (Representational State Transfer) refere-se a um grupo de restrições de design dentro da arquitetura de software que geram sistemas distribuídos eficientes, confiáveis e escaláveis. Um sistema é denominado RESTful quando adere a todas essas restrições. A ideia básica do REST é que um recurso, por exemplo um documento, seja transferido com seu estado bem definido, padronização de operações e formatos, ou serviços que se autodenominam RESTful, quando modificam diretamente o tipo de documento, ao invés de desencadear ações em algum lugar. Como o HTTP, protocolo padrão por trás da "World Wide Web", a "Web" também transfere documentos e links de hipertexto. APIs HTTP simples são às vezes chamadas de APIs RESTful, serviços RESTful ou simplesmente serviços REST, embora não adotem todas as restrições REST necessariamente. Os iniciantes podem presumir que uma API REST significa um serviço HTTP que pode ser chamado usando bibliotecas e ferramentas padrões da web.

# Entrega

Coloque seu código no github e cole abaixo a URL do seu repositório.

| **URL Github:** |  |
| --- | --- |

Pesquise na web sobre [Repositórios do Github](https://docs.github.com/pt/enterprise-server@3.3/repositories/creating-and-managing-repositories/about-repositories), em seguida, o representante do grupo deverá:

1. **Criar um Repositório:** [criar](https://docs.github.com/pt/enterprise-server@3.3/repositories/creating-and-managing-repositories/creating-a-new-repository) um novo repositório para o grupo com o nome do projeto.
2. **Adicionar os Colaboradores:** adicionar todos os integrantes do grupo como [colaboradores](https://docs.github.com/pt/account-and-profile/setting-up-and-managing-your-github-user-account/managing-access-to-your-personal-repositories/inviting-collaborators-to-a-personal-repository) do repositório.

Após o líder da equipe criar e adicionar os colaboradores, o próximo passo é a equipe organizar o repositório do GitHub. Para isso, o grupo deverá:

1. **Criar Arquivos:** criar no diretório raiz do repositório do seu projeto os seguintes arquivos:
   1. **README.md**
   2. **LICENSE.md**
   3. **CONTRIBUTING.md**
2. **Editar o Arquivo README:** Com base nesse [artigo](https://blog.rocketseat.com.br/como-fazer-um-bom-readme/#-demonstra-o-da-aplica-o) e neste [exemplo](https://github.com/profdiegoaugusto/template-repositorio), estruture e preencha o seu arquivo README.md usando a linguagem de marcação [markdown.](https://guides.github.com/features/mastering-markdown/) Nesta estapa do projeto, seu arquivo deverá ter as seguintes seções:
   1. Logo, Capa ou Banner do projeto
   2. Título do Projeto
   3. Descrição (Resumo) do Projeto
   4. Status do Projeto [Opcional]
   5. Índice/Sumário
   6. Requisitos Funcionais
   7. Tecnologias Usadas
   8. Contribuição
   9. Autores
   10. Licença
   11. Agradecimentos

**Observações**

* O conteúdo do arquivo **LICENSE.md** pode ser gerado automaticamente no momento da criação do repositório. Caso não tenha feito, leia os seguintes artigos e defina a licença do seu projeto:
  + [**https://blog.rocketseat.com.br/como-fazer-um-bom-readme/#-licen-a**](https://blog.rocketseat.com.br/como-fazer-um-bom-readme/#-licen-a)
  + [**http://escolhaumalicenca.com.br/**](http://escolhaumalicenca.com.br/)
  + [**https://docs.github.com/pt/github/creating-cloning-and-archiving-repositories/licensing-a-repository**](https://docs.github.com/pt/github/creating-cloning-and-archiving-repositories/licensing-a-repository)
* O arquivo **CONTRIBUTING.md** use o mesmo disponível no repositório exemplo**.**
* O repositório <https://github.com/profdiegoaugusto/template-repositorio> contém o exemplo do mínimo necessário para avaliação do seu trabalho. Seja livre para usar sua criatividade (p.ex.: https://shields.io/category/license), ser autêntico e deixar o repositório com a cara do seu grupo; podem usar badges, emojis e gifs que agreguem valor ao repositório. Cuidado apenas para não ficar muito fantasioso.

# Sistema de Gerenciamento de Aeroportos

Os Sistemas Baseados em Sensores, são sistemas cujo principal objetivo é coletar dados a partir de um conjunto de sensores e processá-los de alguma forma. Os principais requisitos de tais sistemas são: confiabilidade, mesmo em condições ambientais hostis, e manutenabilidade. O Instituto Nacional de Meteorologia (INMET) coordena um conjunto de [estações meteorológicas](https://mapas.inmet.gov.br/) <https://mapas.inmet.gov.br/> espalhadas pelo Brasil, cujo objetivo é informar dados meteorológicos observados nas estações meteorológicas distribuídas no território nacional, que coletam dados através dos sensores meteorológicos

Crie uma API REST para gerenciar as estações meteorológicas brasileiras que coletam dados através dos sensores, para o Banco de Dados Meteorológicos do INMET.

# Visão Geral da API

Esta seção descreve as principais características da API que será implementada.

| **HTTP** | **URL** | **Descrição** |
| --- | --- | --- |
| **GET** | **/api/v1/estacoes** | Obter todas as estações. |
| **GET** | **/api/v1/estacoes/{id}** | Obter uma estação por ID |
| **POST** | **/api/v1/estacoes** | Adicionar uma nova estação. |
| **PUT** | **/api/v1/estacoes/{id}** | Atualizar uma estação existente. |
| **DELETE** | **/api/v1/estacoes/{id}** | Excluir uma estação. |

# Dicionário de Dados

Esta seção apresenta definições e representações dos itens de dados que serão processados pela aplicação. As informações de metadados apresentados neste dicionário de dados incluem o nome do campo, o tipo de dados, se o campo pode estar vazio ou não, ou seja, se ele é obrigatório ou não, os valores máximo e mínimo aceitos, o valor padrão inicial daquele determinado campo, bem como, informações adicionais na coluna observação.

| **Campo** | **Tipo** | **Descrição** |
| --- | --- | --- |
| **id\_estacao** | Inteiro | Chave primária que identifica de maneira única cada estação metereológica. |
| **nome\_estacao** | Texto | Nome da estação metereológica. |
| **cod\_regiao** | Texto | Um ou dois caracteres representando a região brasileira onde está localizada a estação, por exemplo: Norte (N), Nordeste (NE), Sudeste (SE), Sul (S), Centro Oeste (CO). |
| **uf** | Texto | Código Unidade Federativa (UF) do estado |
| **codigo\_wmo** | Texto | Código WMO da estação meteorológica, por exemplo: A001 |
| **latitude** | Real | Um número real representando a latitude. |
| **longitude** | Real | Um número real representando a longitude. |
| **altitude** | Real | Um número real representando a altitude. |
| **data\_fundacao** | Data | Data de fundação da estação meteorológica no formato YYYY-MM-DD, por exemplo: 2000-05-07. |

No website do **Instituto Nacional de Meterologia (INMET)** <https://portal.inmet.gov.br/paginas/catalogoaut>, você encontrará disponível para download gratuitamente, um conjunto de dados com informações dos aeroportos para você testar sua aplicação.

# Tecnologias

A API REST poderá ser implementada usando qualquer linguagem de programação orientada a objetos ou qualquer framework web que dê suporte ao padrão MVC. Porém, o(s) professor(es) poderão dar suporte apenas nas tecnologias vistas em sala de aula.

# Links e Recursos Úteis

* CRUD REST using Spring Boot 2, Hibernate, JPA, and MySQL: <https://www.oracle.com/br/technical-resources/articles/dsl/crud-rest-sb2-hibernate.html>
* Tutorial: criar uma API Web com o ASP.NET Core: <https://learn.microsoft.com/pt-br/aspnet/core/tutorials/first-web-api?view=aspnetcore-6.0&tabs=visual-studio-code>
* REST API TUTORIAL: <https://www.restapitutorial.com/>

# Glossário

Este glossário fornece definições de palavras e abreviações que você precisa saber para entender e construir o software com sucesso.

* WMO: A Organização Meteorológica Mundial (World Meteorological Organization - WMO) é uma agência especializada da Organização das Nações Unidas. É sucessora da Organização Meteorológica Internacional, criada em 1873.
* REST: REST (Representational State Transfer) refere-se a um grupo de restrições de design dentro da arquitetura de software que geram sistemas distribuídos eficientes, confiáveis e escaláveis. Um sistema é denominado RESTful quando adere a todas essas restrições. A ideia básica do REST é que um recurso, por exemplo um documento, seja transferido com seu estado bem definido, padronização de operações e formatos, ou serviços que se autodenominam RESTful, quando modificam diretamente o tipo de documento, ao invés de desencadear ações em algum lugar. Como o HTTP, protocolo padrão por trás da "World Wide Web", a "Web" também transfere documentos e links de hipertexto. APIs HTTP simples são às vezes chamadas de APIs RESTful, serviços RESTful ou simplesmente serviços REST, embora não adotem todas as restrições REST necessariamente. Os iniciantes podem presumir que uma API REST significa um serviço HTTP que pode ser chamado usando bibliotecas e ferramentas padrões da web.

# Entrega

Coloque seu código no github e cole abaixo a URL do seu repositório.

| **URL Github:** |  |
| --- | --- |

Pesquise na web sobre [Repositórios do Github](https://docs.github.com/pt/enterprise-server@3.3/repositories/creating-and-managing-repositories/about-repositories), em seguida, o representante do grupo deverá:

1. **Criar um Repositório:** [criar](https://docs.github.com/pt/enterprise-server@3.3/repositories/creating-and-managing-repositories/creating-a-new-repository) um novo repositório para o grupo com o nome do projeto.
2. **Adicionar os Colaboradores:** adicionar todos os integrantes do grupo como [colaboradores](https://docs.github.com/pt/account-and-profile/setting-up-and-managing-your-github-user-account/managing-access-to-your-personal-repositories/inviting-collaborators-to-a-personal-repository) do repositório.

Após o líder da equipe criar e adicionar os colaboradores, o próximo passo é a equipe organizar o repositório do GitHub. Para isso, o grupo deverá:

1. **Criar Arquivos:** criar no diretório raiz do repositório do seu projeto os seguintes arquivos:
   1. **README.md**
   2. **LICENSE.md**
   3. **CONTRIBUTING.md**
2. **Editar o Arquivo README:** Com base nesse [artigo](https://blog.rocketseat.com.br/como-fazer-um-bom-readme/#-demonstra-o-da-aplica-o) e neste [exemplo](https://github.com/profdiegoaugusto/template-repositorio), estruture e preencha o seu arquivo README.md usando a linguagem de marcação [markdown.](https://guides.github.com/features/mastering-markdown/) Nesta estapa do projeto, seu arquivo deverá ter as seguintes seções:
   1. Logo, Capa ou Banner do projeto
   2. Título do Projeto
   3. Descrição (Resumo) do Projeto
   4. Status do Projeto [Opcional]
   5. Índice/Sumário
   6. Requisitos Funcionais
   7. Tecnologias Usadas
   8. Contribuição
   9. Autores
   10. Licença
   11. Agradecimentos

**Observações**

* O conteúdo do arquivo **LICENSE.md** pode ser gerado automaticamente no momento da criação do repositório. Caso não tenha feito, leia os seguintes artigos e defina a licença do seu projeto:
  + [**https://blog.rocketseat.com.br/como-fazer-um-bom-readme/#-licen-a**](https://blog.rocketseat.com.br/como-fazer-um-bom-readme/#-licen-a)
  + [**http://escolhaumalicenca.com.br/**](http://escolhaumalicenca.com.br/)
  + [**https://docs.github.com/pt/github/creating-cloning-and-archiving-repositories/licensing-a-repository**](https://docs.github.com/pt/github/creating-cloning-and-archiving-repositories/licensing-a-repository)
* O arquivo **CONTRIBUTING.md** use o mesmo disponível no repositório exemplo**.**
* O repositório <https://github.com/profdiegoaugusto/template-repositorio> contém o exemplo do mínimo necessário para avaliação do seu trabalho. Seja livre para usar sua criatividade (p.ex.: https://shields.io/category/license), ser autêntico e deixar o repositório com a cara do seu grupo; podem usar badges, emojis e gifs que agreguem valor ao repositório. Cuidado apenas para não ficar muito fantasioso.