

# Desafio Sicredi

Paulo Victor Sobrinho

[paulo.sobrinho@outlook.com.br](mailto:paulo.sobrinho@outlook.com.br)

**GitHub:** [https://github.com/pvsobrinho/desafio\\_sicredi\\_2024](https://github.com/pvsobrinho/desafio_sicredi_2024)

## Desafio Técnico

### Objetivo

No cooperativismo, cada associado possui um voto e as decisões são tomadas em assembleias, por votação. A partir disso, você precisa criar uma solução back-end para gerenciar essas sessões de votação.

Essa solução deve ser executada na nuvem e promover as seguintes funcionalidades através de uma API REST:

- Cadastrar uma nova pauta;
- Abrir uma sessão de votação em uma pauta (a sessão de votação deve ficar aberta por um tempo determinado na chamada de abertura ou 1 minuto por default);
- Receber votos dos associados em pautas (os votos são apenas 'Sim'/'Não'. Cada associado é identificado por um id único e pode votar apenas uma vez por pauta);
- Contabilizar os votos e dar o resultado da votação na pauta,

Para fins de exercício, a segurança das interfaces pode ser abstraída e qualquer chamada para as interfaces pode ser considerada como autorizada. A escolha da linguagem, frameworks e bibliotecas é livre (desde que não infrinja direitos de uso).

É importante que as pautas e os votos sejam persistidos e que não sejam perdidos com o restart da aplicação.

### Objetivo da aplicação

Para criar a API de acordo com o objetivo. O projeto terá as seguintes funcionalidades:

1. Cadastrar uma nova pauta;
2. Abrir uma sessão de votação;
3. Receber votos dos associados;
4. Contabilizar os votos e dar o resultado da votação;
5. Persistência dos dados para garantir que não sejam perdidos em caso de restart.

# Estratégias Comuns de Versionamento de API

Estratégia escolhida: **Versionamento no Caminho da URL (Path Versioning)**

Essa é a estratégia mais comum, onde a versão da API é indicada diretamente na URL.

## Justificativa:

A estratégia mais comum e recomendada é o versionamento no caminho da URL (Path Versioning). É a mais transparente e simples de implementar e permite uma separação clara entre versões da API, sem confundir os consumidores.

No entanto, se você precisar de maior flexibilidade ou quiser manter a URL estável, o versionamento no cabeçalho HTTP é uma ótima opção, mas exige que os clientes configurem os cabeçalhos adequadamente.

## Exemplo:

bash

/api/v1/pautas

/api/v2/pautas

## Vantagens:

- Simples de entender e implementar.
- Permite evoluir a API sem quebrar a compatibilidade com versões anteriores.
- É fácil para os consumidores verem claramente qual versão estão usando.

## Desvantagens:

- Se a API crescer muito com várias versões, pode haver replicação de código entre as versões.
- URLs podem ficar longas e repetitivas.

## Quando usar:

- Quando você precisa manter múltiplas versões da API ativas simultaneamente.
- Quando os consumidores da API são diversos e precisam de visibilidade clara das versões.

Outras estratégias de versionamento de APIs:

2. Versionamento no Header HTTP (Header Versioning)
3. Versionamento no Parâmetro de Consulta (Query Parameter Versioning)
4. Versionamento por Subdomínio (Subdomain Versioning)

## Iniciando os testes locais:

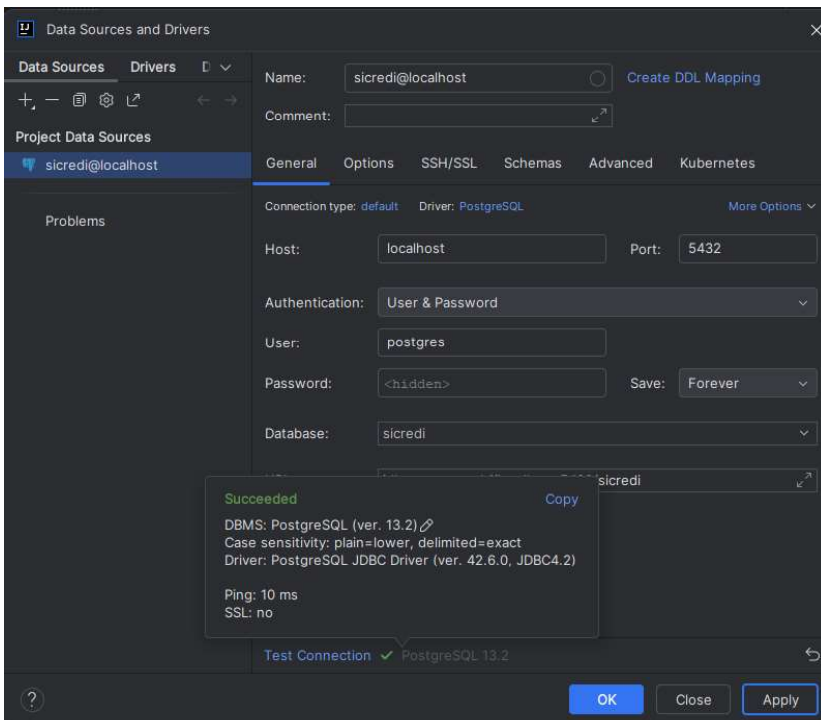
```
Windows PowerShell
O Windows PowerShell
Copyright (C) Microsoft Corporation. Todos os direitos reservados.

Instale o PowerShell mais recente para obter novos recursos e aprimoramentos! https://aka.ms/PSWindows

PS C:\Users\victo> psql -U postgres
Senha para o usuário postgres:

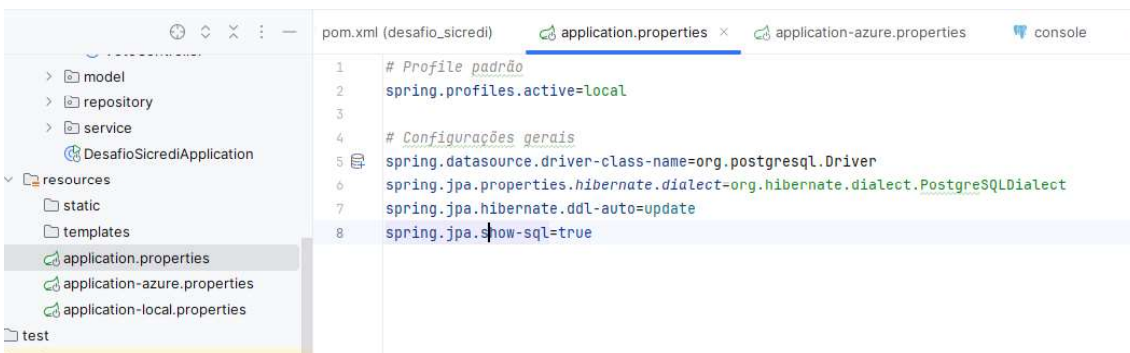
psql (17.0, servidor 13.2)
ADVERTÊNCIA: A página de código da console (850) difere da página de código do Windows (1252)
os caracteres de 8 bits podem não funcionar corretamente. Veja a página de
referência do psql "Notes for Windows users" para obter detalhes.
Digite "help" para obter ajuda.

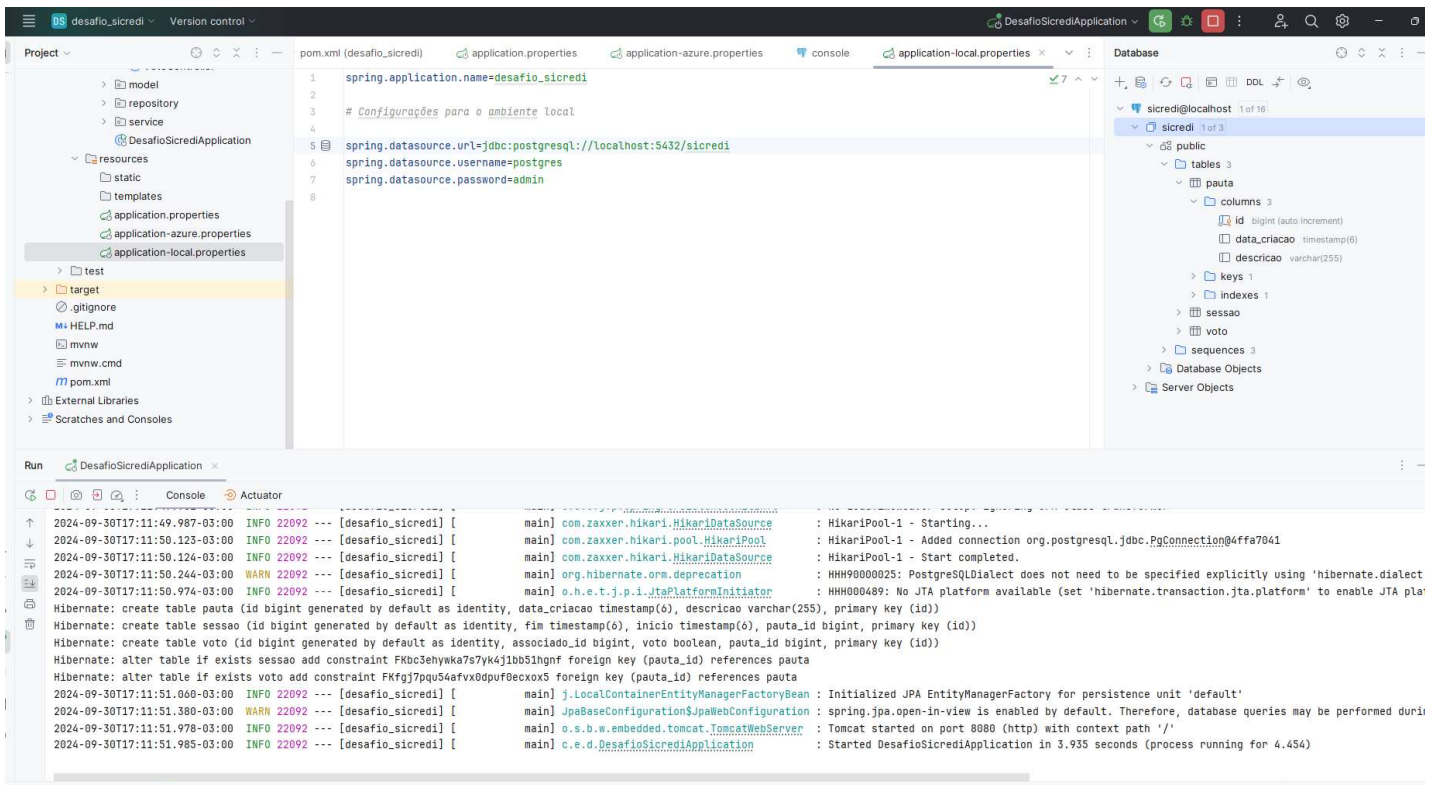
postgres=# CREATE DATABASE sicredi;
CREATE DATABASE
postgres=#
```



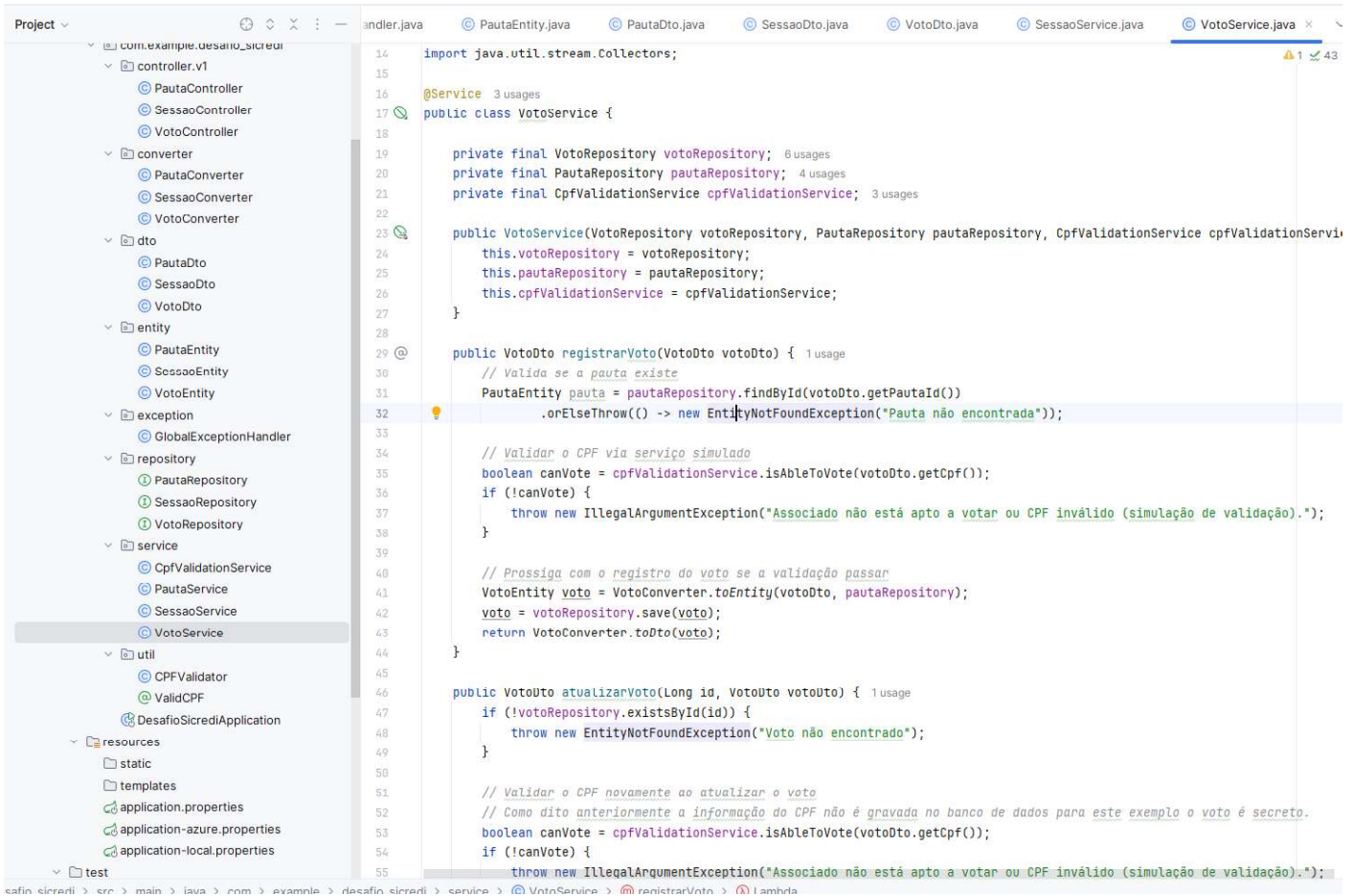
## Rodando a aplicação localmente.

Importante definir o profile local ou azure antes de executar:





## Estrutura do Projeto:



# Por qual motivo optei por não usar injeção de dependência via @Autowired?

## Vantagens de Usar Injeção por Construtor (Forma Atual)

### 1. Imutabilidade e Segurança

- Quando você usa **injeção por construtor** e declara as variáveis com final, as dependências se tornam **imutáveis**. Isso significa que uma vez que os valores são atribuídos pelo construtor, eles não podem ser alterados. Isso proporciona uma maior segurança e consistência no código, prevenindo alterações acidentais nas dependências durante a execução.

### 2. Testabilidade Facilitada

- A **injeção por construtor** facilita a **injeção manual de mocks** ou objetos falsos ao escrever testes unitários. Isso torna o código mais **testável**, já que você pode passar as dependências manualmente no construtor durante os testes.
- Exemplo de teste com injeção por construtor:

### 3. Depreciação do @Autowired em Construtores

- A partir do Spring 4.3, o **Spring** recomenda o uso da **injeção por construtor** em vez de **@Autowired** nos campos ou até mesmo nos próprios construtores, pois a **injeção por construtor** é automática quando há apenas um construtor na classe. Não é necessário usar @Autowired no construtor para que o Spring saiba qual construtor usar.

Este é uma justificativa interessante pois estou usando Java 23 e Spring 5

### 4. Evita Injeção de Campo Nulo

- Usando @Autowired nos campos, há um risco maior de inicialização tardia, o que pode causar o problema do "null pointer exception" se a injeção falhar ou ocorrer fora de ordem. Na injeção por construtor, como as dependências são resolvidas na criação da instância, esse problema é evitado.

### 5. Facilidade de Manutenção e Refatoração

- Quando as dependências são passadas pelo construtor, elas ficam visíveis imediatamente, facilitando o entendimento de quais dependências a classe precisa. Isso torna o código mais fácil de manter, especialmente em projetos maiores, onde classes complexas podem ter muitas dependências.

## Injeção com @Autowired nos Campos: Desvantagens

### • Baixa Visibilidade das Dependências

- Quando usamos @Autowired nos campos, não fica claro de imediato quais dependências a classe realmente precisa, o que pode dificultar a manutenção do código.

### • Impossibilidade de Imutabilidade

- Com a injeção de dependência diretamente nos campos, você não pode marcar os campos como final, o que significa que eles podem ser alterados acidentalmente durante a execução, potencialmente causando problemas inesperados.

### • Dificuldade nos Testes

- Testar classes que usam @Autowired diretamente nos campos pode ser mais complicado, porque você precisaria usar um framework como o Spring ou ferramentas adicionais de mock para injetar dependências nos testes. Já com injeção por construtor, você pode passar as dependências diretamente no construtor durante os testes.


**Isso pode se tornar uma dor de cabeça a medida que o projeto fica mais complexo. Realizar teste automatizado pode se tornar cada vez mais difícil, ou não a depender das escolhas arquiteturais e de quais recursos serão utilizados.**

## **Documentação da API com Swagger**



← ↻ 🏠 ⓘ localhost:8080/swagger-ui/index.html#/

Email Pessoal PUC Minas 2024 EF Google Voos Documentos Google TED Talks English learning Financas

 **Swagger**  
supported by SMARTBEAR

/v3/api-docs

# Desafio Sicredi 1.0.0 OAS3

/v3/api-docs

API para gerenciamento de sessões de votação no Desafio Sicredi

**Servers**

http://localhost:8080 - Generated server url

## Sessao

APIs relacionadas a Sessões de Votação

- GET** /api/v1/sesoes/{id} Obter sessão por ID
- PUT** /api/v1/sesoes/{id} Atualizar uma sessão existente
- GET** /api/v1/sesoes Listar todas as sessões
- POST** /api/v1/sesoes Abrir uma nova sessão de votação

## Voto

APIs relacionadas a Votos

- GET** /api/v1/votos/{id} Obter voto por ID
- PUT** /api/v1/votos/{id} Atualizar um voto existente
- GET** /api/v1/votos Listar todos os votos

Existem validações nos parâmetros e nos campos do requestBody. Fazer validação no dto é uma forma eficiente e de otimização para reutilização de código.

# Sessao

APIs relacionadas a Sessões de Votação

GET

/api/v1/sessoes/{id}

Obter sessão por ID

PUT

/api/v1/sessoes/{id}

Atualizar uma sessão existente

Atualiza os dados de uma sessão de votação existente.

Parameters

Name	Description
<b>id</b> <small>* required</small>	
integer(\$int64)	ID da sessão que será atualizada
(path)	
	<div>id</div>

Request body

required

Example Value

Schema

```
{
  "id": 0,
  "pautaId": 0,
  "inicio": "2024-09-30T21:03:28.215Z",
  "fim": "2024-09-30T21:03:28.215Z"
}
```



```
SessaoDto.java x VotoDto.java PautaDto.java SessaoService.java
1 package com.example.desafio_sicredi.dto;
2
3 import jakarta.validation.constraints.NotNull;
4 import lombok.Data;
5
6 import java.time.LocalDateTime;
7
8 @Data 19 usages
9 public class SessaoDto {
10
11     private Long id;
12
13     @NotNull(message = "O ID da pauta é obrigatório")
14     private Long pautaId;
15
16     @NotNull(message = "A data de início é obrigatória")
17     private LocalDateTime inicio;
18
19     @NotNull(message = "A data de fim é obrigatória")
20     private LocalDateTime fim;
21
22 }
```

Uso de conversores garante maior segurança ao evitar o uso de uma entidade no controlador e melhor eficiência no uso de dados. Trafegando assim apenas os dados necessários e não todo o corpo de dados da entidade. Além de garantir um código limpo para quem usa a api. Apenas dados necessários são trafegados para determinada ação.

```
SessaoDto.java  PautaConverter.java  VotoDto.java  PautaDto.java  SessaoS

1 package com.example.desafio_sicredi.converter;
2
3
4 import com.example.desafio_sicredi.dto.PautaDto;
5 import com.example.desafio_sicredi.entity.PautaEntity;
6
7 public class PautaConverter { 6 usages
8
9 @ public static PautaEntity toEntity(PautaDto dto) { 2 usages
10     PautaEntity pautaEntity = new PautaEntity();
11     pautaEntity.setId(dto.getId());
12     pautaEntity.setDescricao(dto.getDescricao());
13     pautaEntity.setDataCriacao(dto.getDataCriacao());
14     return pautaEntity;
15 }
16
17 @ public static PautaDto toDto(PautaEntity pautaEntity) { 3 usages
18     PautaDto dto = new PautaDto();
19     dto.setId(pautaEntity.getId());
20     dto.setDescricao(pautaEntity.getDescricao());
21     dto.setDataCriacao(pautaEntity.getDataCriacao());
22     return dto;
23 }
24 }
25
```

## Otimização no tratamento de exceções com o GlobalExceptionHandler

```
Project
├── converter
│   ├── PautaConverter
│   ├── SessaoConverter
│   └── VotoConverter
├── dto
│   ├── PautaDto
│   ├── SessaoDto
│   └── VotoDto
├── entity
│   ├── PautaEntity
│   ├── SessaoEntity
│   └── VotoEntity
├── exception
│   └── GlobalExceptionHandler
├── repository
│   ├── PautaRepository
│   ├── SessaoRepository
│   └── VotoRepository
├── service
│   ├── CpfValidationService
│   ├── PautaService
│   ├── SessaoService
│   └── VotoService
├── util
│   ├── CPFValidator
│   └── ValidCPF
├── DesafioSicrediApplication
├── resources
│   ├── static
│   └── templates
├── application.properties
├── application-azure.properties
├── application-local.properties
├── test
│   ├── java
│   │   └── com.example.desafio_sicredi
│   │       └── DesafioSicrediApplicationTests
└── ...

CpfValidationService.java  VotoRepository.java  SessaoRepository.java  PautaRepository.java  GlobalExceptionHandler.java

7 import org.springframework.web.server.ResponseStatusException;
8
9 @RestControllerAdvice
10 public class GlobalExceptionHandler {
11
12     // Captura a exceção ResponseStatusException para lidar com códigos de erro personalizados, como 400 e 404
13     @ExceptionHandler(ResponseStatusException.class)
14     public ResponseEntity<ErrorResponse> handleResponseStatusException(ResponseStatusException ex) {
15         ErrorResponse errorResponse = new ErrorResponse(ex.getStatusCode().value(), ex.getReason());
16         return new ResponseEntity<>(errorResponse, ex.getStatusCode());
17     }
18
19     // Captura outras exceções genéricas e retorna 500
20     @ExceptionHandler(Exception.class)
21     public ResponseEntity<ErrorResponse> handleGeneralException(Exception ex) {
22         ErrorResponse errorResponse = new ErrorResponse(HttpStatus.INTERNAL_SERVER_ERROR.value(), ex.getMessage());
23         return new ResponseEntity<>(errorResponse, HttpStatus.INTERNAL_SERVER_ERROR);
24     }
25
26     // Classe interna para formatar a resposta de erro
27     public static class ErrorResponse { 6 usages
28         private int statusCode; 3 usages
29         private String message; 3 usages
30
31         public ErrorResponse(int statusCode, String message) { 2 usages
32             this.statusCode = statusCode;
33             this.message = message;
34         }
35
36         public int getStatusCode() { no usages
37             return statusCode;
38         }
39
40         public void setStatusCode(int statusCode) { no usages
41             this.statusCode = statusCode;
42         }
43
44         public String getMessage() { no usages
45             return message;
46         }
47     }
48 }
```

# Arquitetura

O código que construímos segue **alguns** princípios da arquitetura **SOLID**, mas ainda existem pontos que podem ser aprimorados para atender completamente a esses princípios. **Não foram aplicados todos os princípios pois não existe complexidade suficiente para justificar tal uso.** Vamos analisar cada princípio do SOLID e identificar como o código atual se alinha a eles:

## 1. S - Single Responsibility Principle (SRP)

### O que o código já faz bem:

- Cada classe tem uma responsabilidade clara:
  - **Controladores** lidam com as requisições HTTP.
  - **Serviços** contêm a lógica de negócios.
  - **Repositórios** interagem com o banco de dados.
  - **Conversores** fazem a conversão entre DTOs e entidades.

### O que pode melhorar:

- O código está bem próximo de cumprir o SRP, mas uma análise constante deve ser feita para garantir que classes de serviço não fiquem sobrecarregadas com responsabilidades adicionais. Se a lógica de negócios crescer, poderá ser necessário dividir algumas responsabilidades entre serviços menores ou criar classes auxiliares.

## 2. O - Open/Closed Principle (OCP)

### O que o código já faz bem:

O design do sistema facilita a extensão sem a necessidade de modificar código existente. Por exemplo, novos endpoints ou funcionalidades podem ser adicionados sem alterar os controladores e serviços existentes.

### O que pode melhorar:

- Em alguns casos, podemos precisar de estratégias mais avançadas, como a criação de interfaces e abstrações para comportar diferentes implementações sem modificar as classes existentes. No momento, os serviços e controladores estão relativamente simples e cumprem o papel. À medida que a aplicação crescer, interfaces e estratégias de injeção de dependência serão ainda mais importantes para garantir a adesão a esse princípio.

## 3. L - Liskov Substitution Principle (LSP)

### O que o código já faz bem:

- O LSP se aplica mais a cenários onde há herança ou polimorfismo. Como o código atual usa principalmente injeção de dependência com interfaces para repositórios, não temos problemas diretos com LSP. Se decidirmos implementar classes abstratas ou subclasses, será importante garantir que essas substituições não alterem o comportamento esperado das classes de forma inesperada.

### O que pode melhorar:

- Quando começarmos a usar mais interfaces ou subclasses, precisamos garantir que todas as substituições mantenham o comportamento esperado. Isso é mais relevante para o futuro, caso o projeto se expanda com mais interfaces e heranças.

## 4. I - Interface Segregation Principle (ISP)

### O que o código já faz bem:

- O ISP sugere que as interfaces devem ser pequenas e especializadas. No momento, usamos repositórios que implementam interfaces do Spring Data JPA (JpaRepository), que são específicas e focadas nas operações de banco de dados.

### O que pode melhorar:

- À medida que a aplicação crescer, devemos considerar evitar interfaces grandes e focar em criar interfaces especializadas para funções específicas de serviços ou repositórios, evitando que classes implementem métodos que não precisam.

## 5. D - Dependency Inversion Principle (DIP)

### O que o código já faz bem:

- Estamos usando **injeção de dependências** com **repositórios** e **serviços**. Isso facilita a testabilidade e mantém o código desacoplado de implementações específicas.

### O que pode melhorar:

- Podemos ir além e introduzir mais abstrações por meio de interfaces para serviços, o que nos permitirá criar implementações alternativas dos serviços sem alterar o código que depende dessas classes.

# Teste de Operações da API

## Pauta APIs relacionadas a Pautas

**PUT** `/api/v1/pautas/{id}` Atualizar uma pauta existente

**GET** `/api/v1/pautas` Listar todas as pautas

**POST** `/api/v1/pautas` Cadastrar uma nova pauta

Cria uma nova pauta para votação.

### Parameters

No parameters

**Request body** required

```
{
  "id": 0,
  "descricao": "pauta 1",
  "dataCriacao": "2024-09-30T21:16:11.877Z"
}
```

`http://localhost:8080/api/v1/pautas`

### Server response

Code	Details
------	---------

200	
-----	--

### Response body

```
{
  "id": 1,
  "descricao": "pauta 1",
  "dataCriacao": "2024-09-30T21:16:11.877Z"
}
```

### Response headers

```
connection: keep-alive
content-type: application/hal+json
date: Mon, 30 Sep 2024 21:16:19 GMT
keep-alive: timeout=60
transfer-encoding: chunked
```

### Responses

Code	Description
------	-------------

**PUT** `/api/v1/pautas/{id}` Atualizar uma pauta existente

**GET** `/api/v1/pautas` Listar todas as pautas

Retorna todas as pautas cadastradas.

## Parameters

No parameters

Execute

## Responses

### Curl

```
curl -X 'GET' \
  'http://localhost:8080/api/v1/pautas' \
  -H 'accept: application/hal+json'
```

### Request URL

`http://localhost:8080/api/v1/pautas`

### Server response

Code	Details
------	---------

200

### Response body

```
[
  {
    "id": 1,
    "descricao": "pauta 1",
    "dataCriacao": "2024-09-30T21:16:11.877"
  }
]
```

### Response headers



#### Request URL

`http://localhost:8080/api/v1/pautas/1`

#### Server response

##### Code

##### Details

200

##### Response body

```
{
  "id": 1,
  "descricao": "ATUALIADA",
  "dataCriacao": "2024-09-30T21:49:07.284"
}
```

##### Response headers

```
connection: keep-alive
content-type: application/hal+json
date: Mon, 30 Sep 2024 22:14:15 GMT
keep-alive: timeout=60
transfer-encoding: chunked
```

#### Responses

#### Request URL

`http://localhost:8080/api/v1/pautas/4`

#### Server response

##### Code

##### Details

404

*Undocumented*

Error: response status is 404

##### Response body

```
{
  "statusCode": 404,
  "message": "Pauta não encontrada"
}
```

##### Response headers

TESTE INFORMANDO UMA PAUTA INVALIDA OU NÃO EXISTENTE:

**POST** `/api/v1/votos` Registrar um voto

Registra um voto (sim/não) associado a uma pauta.

**Parameters**

No parameters

**Request body** required

```
{
  "id": 0,
  "associadoId": 3,
  "pautaid": 3,
  "voto": true,
  "cpf": "32132321"
}
```

**Request URL**

`http://localhost:8080/api/v1/votos`

**Server response**

**Code**

**Details**

404

*Undocumented*

Error: response status is 404

**Response body**

```
{
  "statusCode": 404,
  "message": "Pauta não encontrada"
}
```

**Response headers**

`connection: keep-alive`

Teste informando uma pauta existente com cpf invalido:

**POST****/api/v1/votos** Registrar um voto

Registra um voto (sim/não) associado a uma pauta.

### Parameters

No parameters

### Request body required

```
{
  "id": 0,
  "associadoId": 3,
  "pautaId": 2,
  "voto": true,
  "cpf": "32132321"
}
```

### Server response

**Code****Details**

400

*Undocumented*

Error: response status is 400

#### Response body

```
{
  "statusCode": 400,
  "message": "CPF inválido ou associado não está apto a votar."
}
```

#### Response headers

Teste com CPF correto:

GET

/api/v1/votos

Listar todos os votos

POST

/api/v1/votos

Registrar um voto

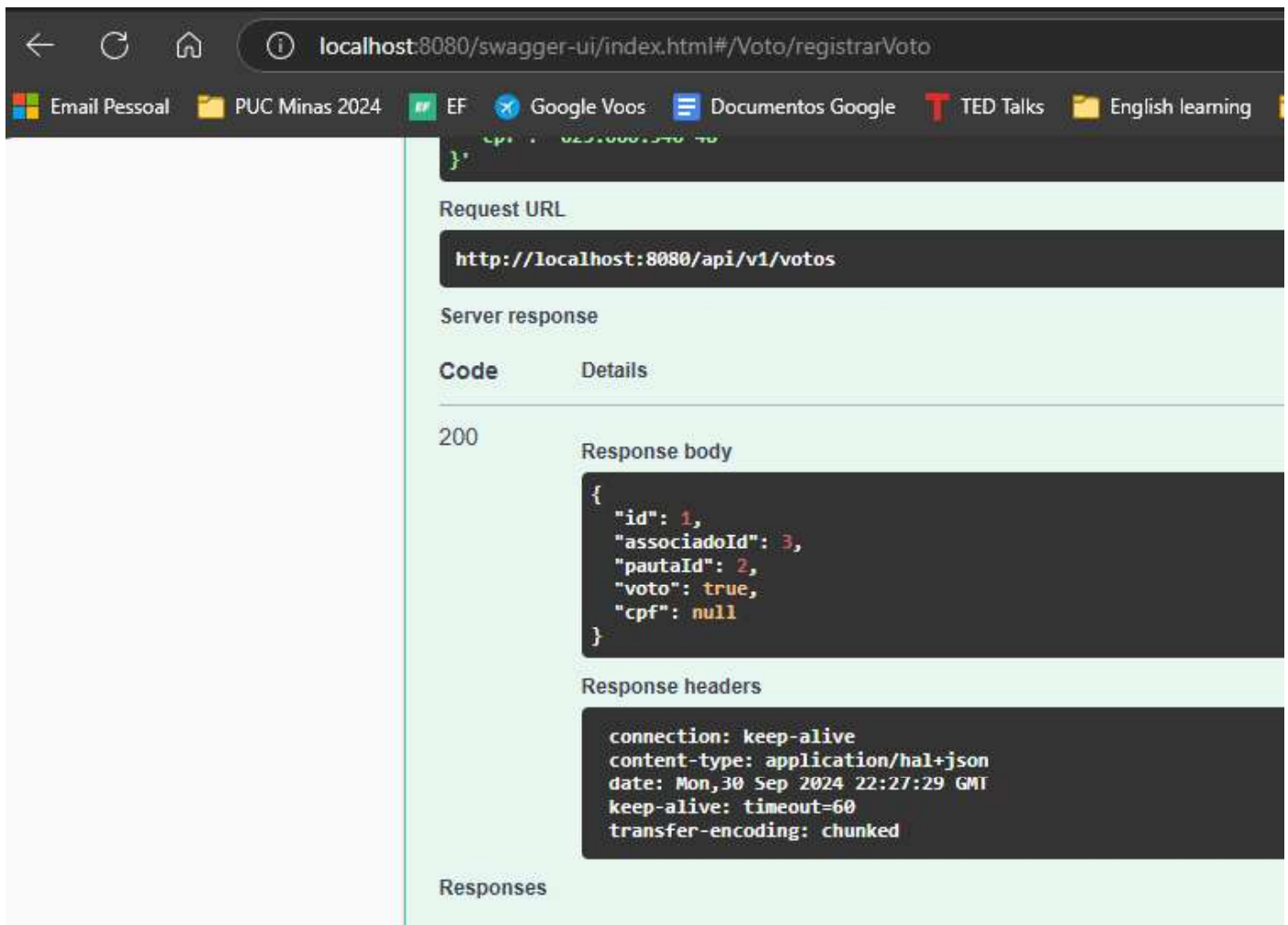
Registra um voto (sim/não) associado a uma pauta.

#### Parameters

No parameters

Request body required

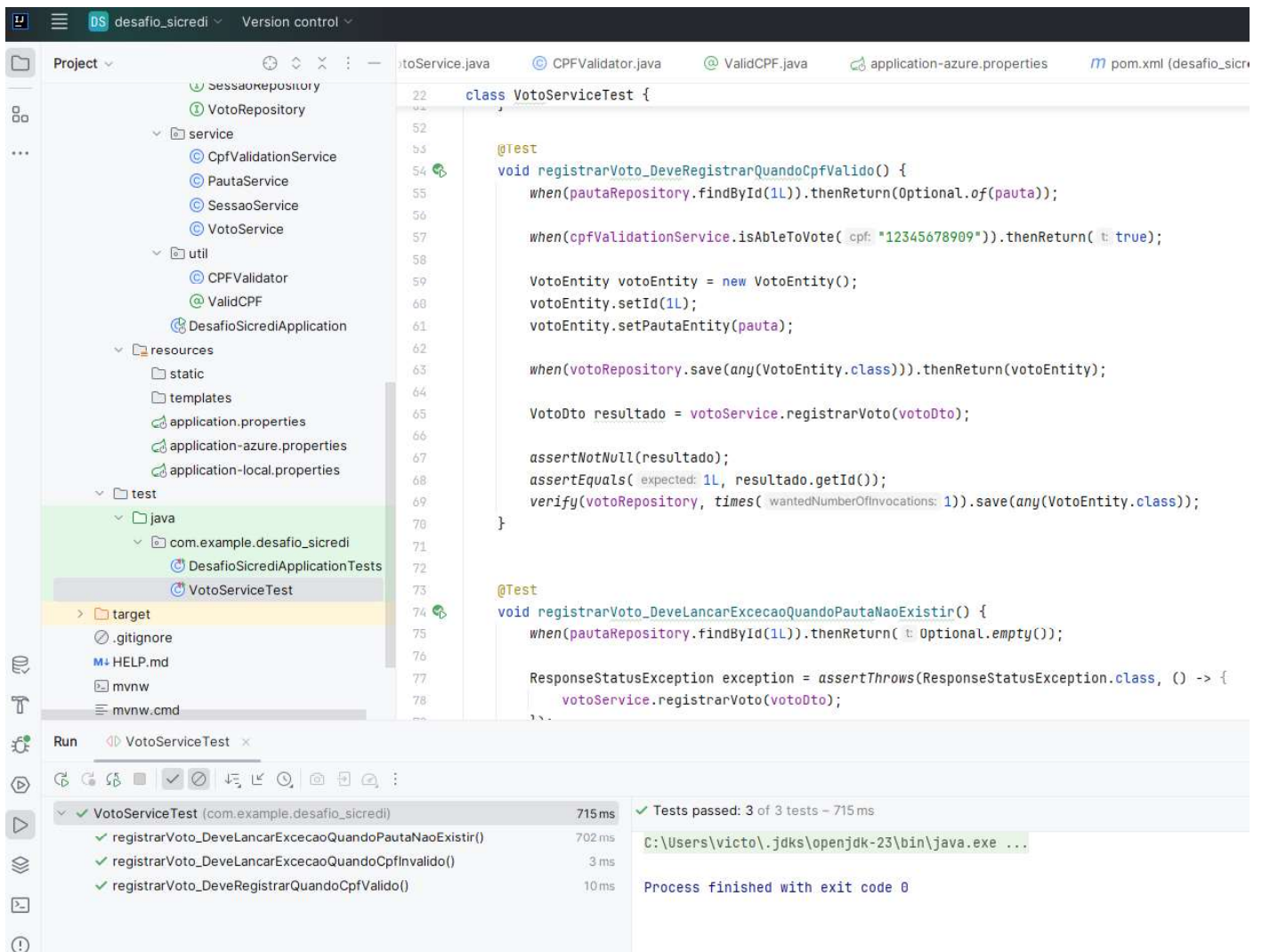
```
{  "id": 0,  "associadoId": 3,  "pautaid": 2,  "voto": true,  "cpf": "029.606.540-48"}
```



## Testes Unitários Automatizados com junit

Apesar de não ter sido solicitado no enunciado. Realizei um teste no serviço o principal. É de fundamental importância os testes unitários. Assim evitamos gerar novos bugs em ambiente de produção pelo fato de a automatização de testes capturar bugs antes de submeter novas alterações para o ambiente produtivo.

Apenas para não perder o costume criei uma classe de teste:



## Configurando a Nuvem Azure (WebApp + Banco de Dados)



←↻🏠

https://portal.azure.com/#create/Microsoft.PostgreSQLServer

Email PessoalPUC Minas 2024EFGoogle VoosDocumentos GoogleTED TalksEnglish learningFinancasDevTelegram WebPROVia

Microsoft AzurePesquisar recursos, serviços e documentos (G+/)

Página inicial >

Novo Banco de Dados do Azure para o Servidor Flexível PostgreSQL

Microsoft

⚠️ Nomes de servidor, método de conectividade de rede e redundância de backup não podem ser alterados após a criação do servidor. Revise essas opções cuidadosamente antes do provisionamento.

Assinatura \* ⓘAzure for Students

Grupo de recursos \* ⓘ(Novo) desafio\_sicredi

[Criar novo](#)

Detalhes do servidor

Insira as configurações necessárias para este servidor, incluindo a escolha de um local e a configuração dos recursos de computação e de armazenamento.

Nome do servidor \* ⓘdemopostgres123

Região \* ⓘEast US 2

Versão PostgreSQL \* ⓘ16

Tipo de carga de trabalho ⓘ

Desenvolvimento

Produção

Computação + armazenamento ⓘ

Intermitente, B1ms

1 vCores, 2 GiB de RAM, 32 GiB de armazenamento, P4 (120 IOPS)

Redundância geográfica : Disabled

[Configurar servidor](#)

Zona de disponibilidade ⓘ

Nenhuma preferência

Alta disponibilidade

Implante uma réplica em espera para a funcionalidade de failover automático. Recomendamos alta disponibilidade para todas as cargas de trabalho de produção. [Saiba mais](#)

Alta disponibilidade ⓘ

Desabilitado (SLA de 99,9%)

Mesma zona – um servidor em espera está disponível na mesma zona

Custos estimados

SKU da computaçãoUSD 12.41/mês

Gratuito até 750 horas

Standard\_B1ms (1 vCore)12.41

ArmazenamentoUSD 3.68/mês

Gratuito até 32 GB

Armazenamento selecionado 32 GiB (USD 0.12 por GiB)32 x 0.12

Largura de banda

A transferência de dados de saída entre serviços em regiões diferentes incorrerá em encargos adicionais. Qualquer transferência de dados de entrada é gratuita. [Saiba mais](#)

Total estimadoUSD 16.09/mês

As cobranças serão aplicadas se você usar acima dos limites mensais gratuitos. Verifique [seu uso](#) de serviços gratuitos. As cobranças finais aparecerão em sua moeda local.

Revisar + criar

Avançar: Rede >

Página inicial >

## PostgreSqlFlexibleServer\_dc3d387b166348b4a850f7bcefa6014c | Visão Geral

Pesquisar

Excluir Cancelar Reimplantar Baixar Atualizar

### Visão Geral

Entradas

Saídas

Modelo

## A implantação está em andamento

Nome da implantação: PostgreSqlFlexibleServer\_dc3d387b166348b... Hora de início: 30  
Assinatura: Azure for Students ID de Correlação:  
Grupo de recursos: desafio\_sicredi

### Detalhes de implantação

Recurso	Tipo
Nenhum resultado.	

Editar comentários

Página inicial > PostgreSqlFlexibleServer\_dc3d387b166348b4a850f7bcefa6014c | Visão Geral > demopostgres123

## demopostgres123 | Bancos de dados

Banco de Dados do Azure para servidor flexível do PostgreSQL

Pesquisar

Adicionar Excluir Atualizar CLI / PS Comentários

Você pode criar, visualizar e excluir PostgreSQL bancos de dados neste servidor. Observe que você não pode excluir nenhum esquema do sistema, co

Nome ↑	Conjunto de c...	Ordenação	Tipo de esque...
azure_maintenance	UTF8	en_US.utf8	System
azure_sys	UTF8	en_US.utf8	System
postgres	UTF8	en_US.utf8	User

Conectar Abrir no Power BI

- Visão geral
- Log de atividade
- IAM (Controle de acesso)
- Marcações
- Diagnosticar e resolver problemas
- Migração
- Configurações
- Computação + armazenamento
- Rede
- Bancos de dados
- Conectar

Microsoft Azure

Pesquisar recursos, serviços e documentos (G+)

Copilot

1184240@sga.pucminas.br

Página inicial > PostgreSQLFlexibleServer\_dc3d387b166348b4a8507bcefa6014c | Visão Geral > demopostgres123

demopostgres123 | Bancos de dados

Banco de Dados do Azure para servidor flexível do PostgreSQL

Pesquisar

+ Adicionar Excluir Atualizar CLI / PS Comentários

Visão geral Log de atividade IAM (Controle de acesso) Marçações Diagnosticar e resolver problemas Migração Configurações

Você pode criar, visualizar e excluir PostgreSQL bancos de dados neste servidor. Observe que você não pode excluir nenhum esquema do sistema, como azure\_sys,azure\_maintenance. Você pode se conectar ao banco de dados usando PostgreSQL ferramentas do cliente.

Nome ↑	Conjunto de c...	Ordenação	Tipo de esque...
azure_maintenance	UTF8	en_US.utf8	System
azure_sys	UTF8	en_US.utf8	System
<input checked="" type="checkbox"/> postgres	UTF8	en_US.utf8	User Conectar  Abrir no Power BI

\*\*\* Atualizando as configurações de segurança da conexão  
Atualizando as configurações de segurança da cone para demopostgres123

Microsoft Azure

Pesquisar recursos, serviços e documentos (G+)

Copilot

1184240@sga.pucminas.br

Página inicial > PostgreSQLFlexibleServer\_dc3d387b166348b4a8507bcefa6014c | Visão Geral > demopostgres123

demopostgres123 | Bancos de dados

Banco de Dados do Azure para servidor flexível do PostgreSQL

Pesquisar

+ Adicionar Excluir Atualizar CLI / PS Comentários

Visão geral Log de atividade IAM (Controle de acesso) Marçações Diagnosticar e resolver problemas Migração Configurações Computação + armazenamento Rede Bancos de dados Conectar Parâmetros do servidor Replicação

Você pode criar, visualizar e excluir PostgreSQL bancos de dados neste servidor. Observe que você não pode excluir nenhum esquema do sistema, como azure\_sys,azure\_maintenance. Você pode se conectar ao banco

Nome ↑	Conjunto de c...	Ordenação	Tipo de esque...
azure_maintenance	UTF8	en_US.utf8	System
azure_sys	UTF8	en_US.utf8	System
<input checked="" type="checkbox"/> postgres	UTF8	en_US.utf8	User  Conectar  Abrir no Power BI

Alternar para PowerShell Reiniciar Gerenciar arquivos Nova sessão Editor Visualização da Web Configurações Ajuda

Subscription used to launch your CloudShell dff51c3a-2cbf-4bdb-b02a-2dd8a959e2d4 is not registered to Microsoft.CloudShell Namespace. Please follow these instructions "hster. In future, unregistered subscriptions will have restricted access to CloudShell service.  
  
Your Cloud Shell session will be ephemeral so no files or system changes will persist beyond your current session.  
Subscription used to launch your CloudShell dff51c3a-2cbf-4bdb-b02a-2dd8a959e2d4 is not registered to Microsoft.CloudShell Namespace. Please follow these instructions "hster. In future, unregistered subscriptions will have restricted access to CloudShell service.  
  
Your Cloud Shell session will be ephemeral so no files or system changes will persist beyond your current session.  
paulo [ ~ ]\$ psql "--host=demopostgres123.postgres.database.azure.com" "--port=5432" "--dbname=postgres" "--username=demopostgres123" "--set=sslmode=require"  
Password for user demopostgres123:  
MOTD: SqlServer has been updated to Version 22!  
  
VERBOSE: Authen> psql "--host=demopostgres123.postgres.database.azure.com" "--port=5432" "--dbname=postgres" "--username=demopostgres123" "--set=sslmode=require"  
Password for user demopostgres123: ..

- ferramentas de suporte integradas para Eclipse, Visual Studio Code e Visual Studio.
- Integração de CI/CD com GitHub, Docker Hub, Azure Pipelines, Registro de Contêiner do Azure, Bitbucket e outros.
- Recursos abrangentes de diagnóstico, monitoramento e alerta com o Application Insights e o Azure Monitor.



https://portal.azure.com/#create/Microsoft.WebSite

Email Pessoal

PUC Minas 2024

EF

Google Voos

Documentos Google

TED Talks

English learning

Finanças

Microsoft Azure

Pesquisar recurso

Página inicial > desafio\_sicredi > Marketplace > Aplicativo Web >

Criar Aplicativo Web

Pilha de runtime \*

Java 21

Pilha do servidor Web Java \*

Java SE (Embedded Web Server)

Sistema Operacional \*

☒ Linux ☐ Windows

Região \*

Canada Central

Não está localizando seu Plano do Serviço de Aplicativo? Tente uma região diferente ou selecione seu Ambiente do Serviço de Aplicativo.

Planos de preços

O tipo de preço do plano do Serviço de Aplicativo determina o local, os recursos, o custo e os recursos de computação associados ao aplicativo. [Saiba mais](#)

Plano do Linux (Canada Central) \*

(Novo) ASP-desafiosicredi-a30f

[Criar novo](#)

Plano de preços

Gratuito F1 (Infraestrutura compartilhada)

Revisar + criar

< Anterior

Avançar: Banco de dados >

Alternar para PowerShell

Reiniciar

Gerenciar arquivos

Nova sessão

Editor

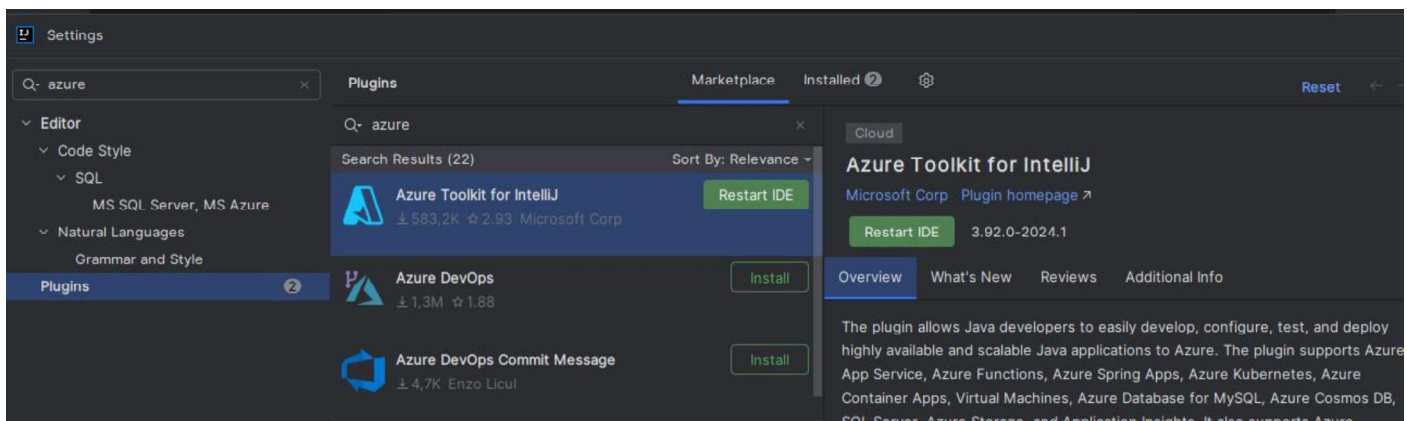
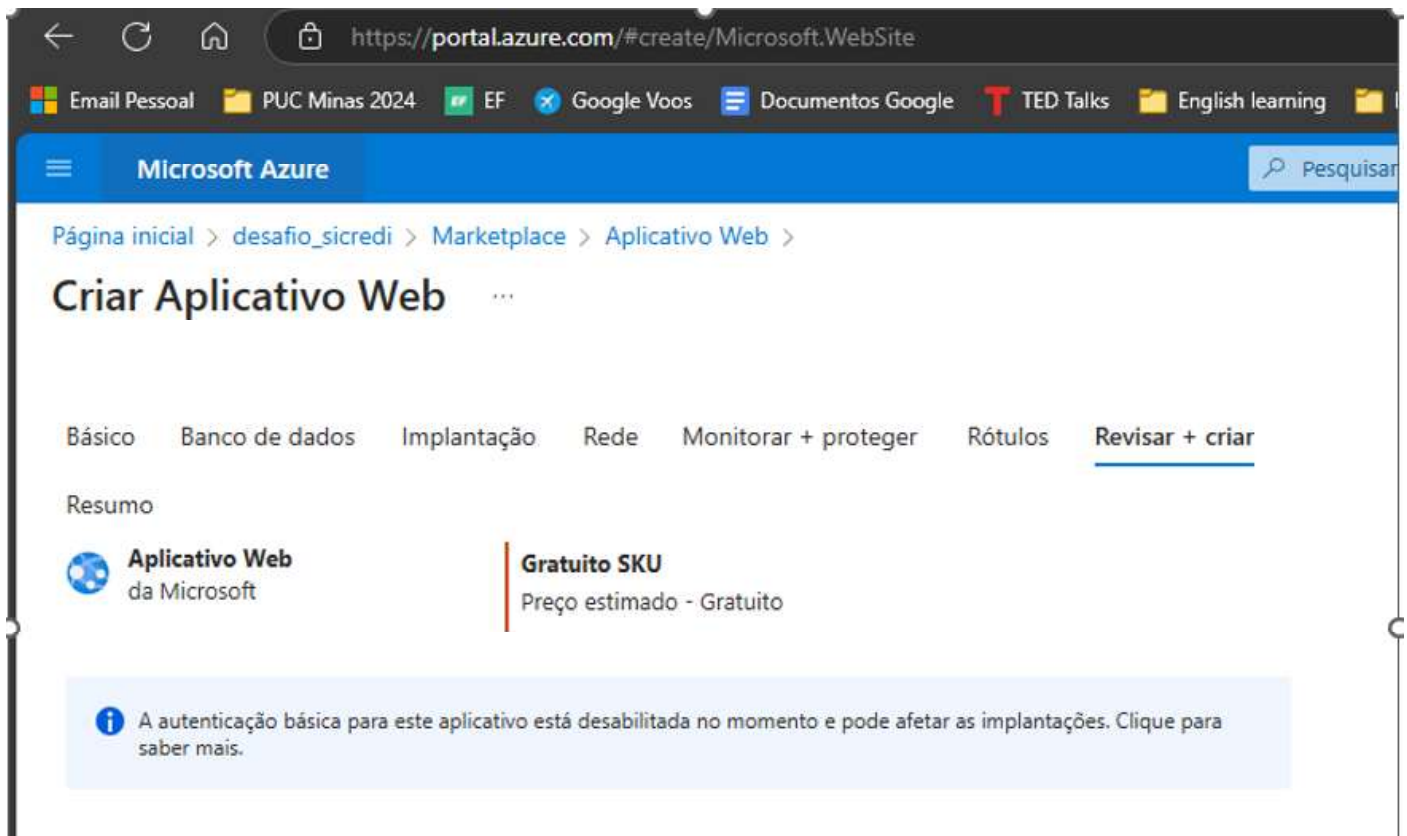
Visualização da Web

Solicitando um Cloud Shell.Succeeded.

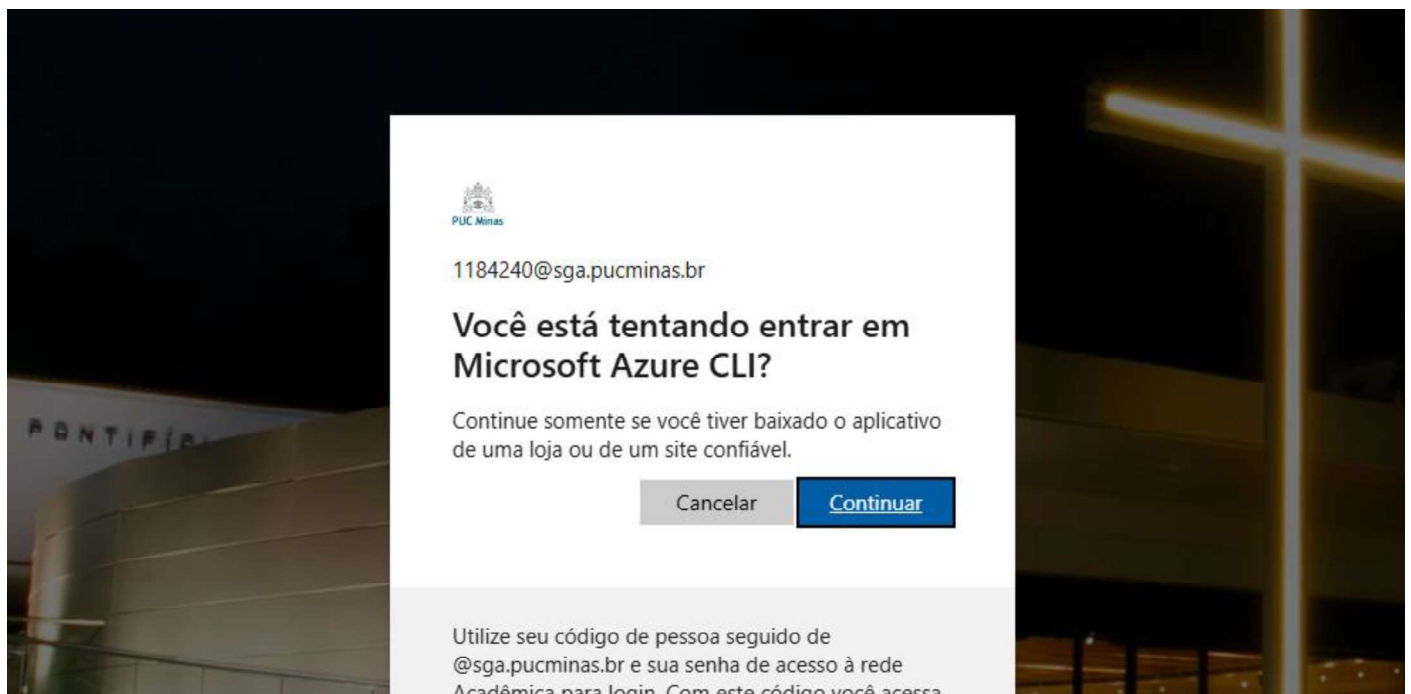
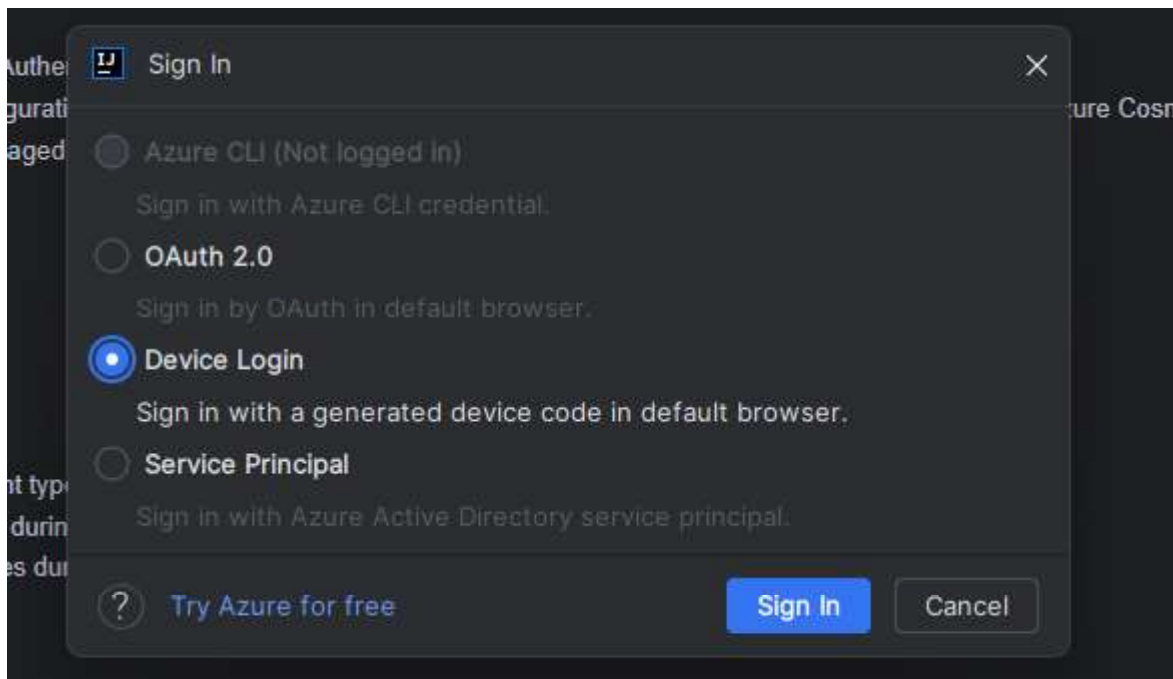
Connecting terminal...

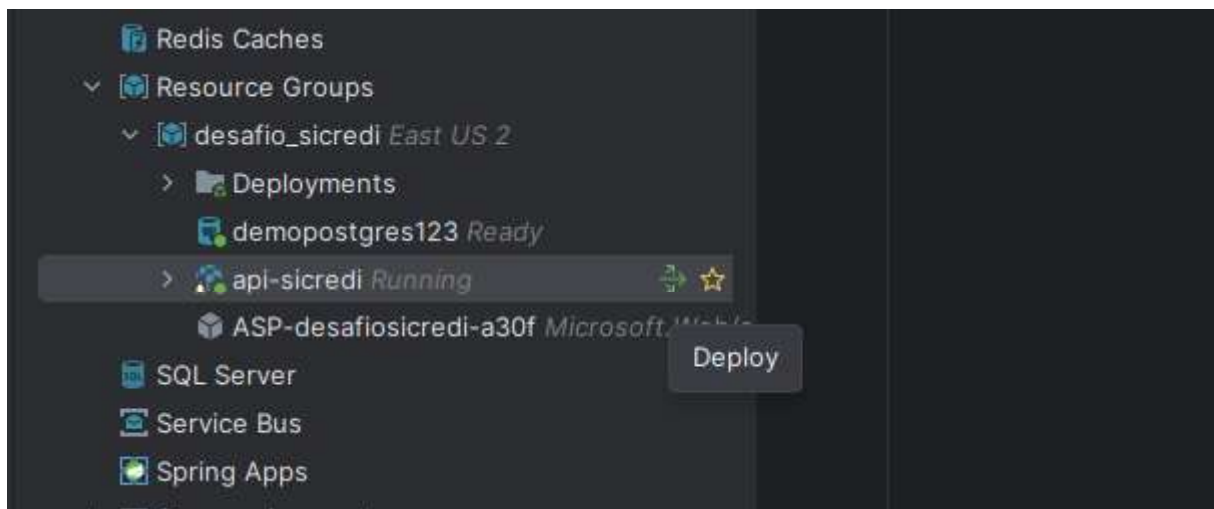
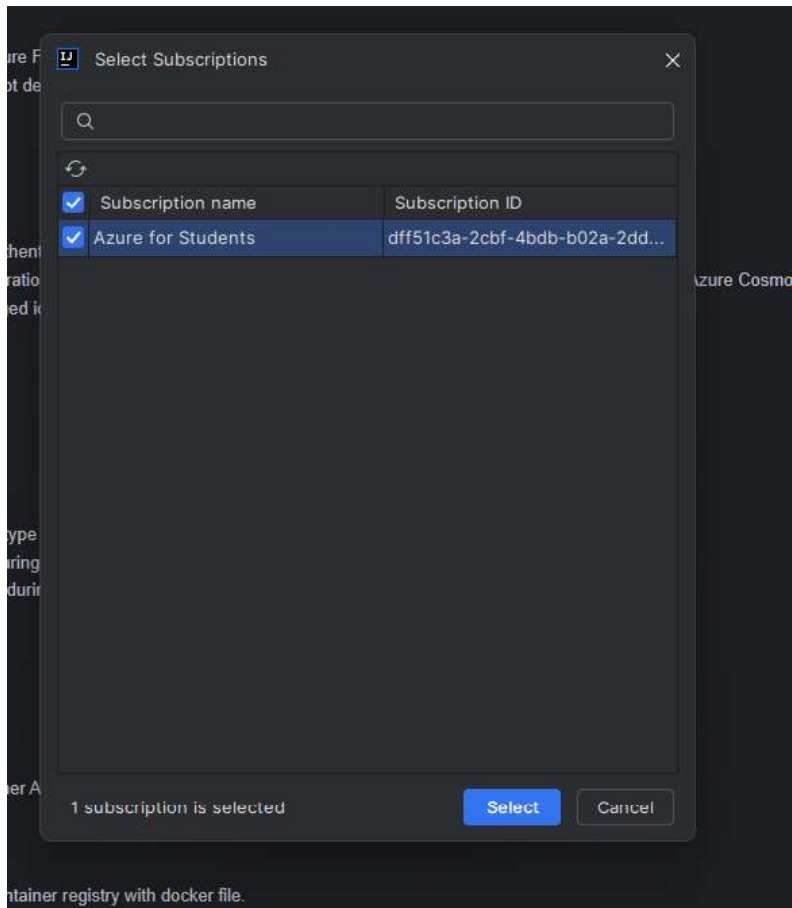
Subscription used to launch your CloudShell dff51c3a-2cbf-4bdb-b02a-2dd8a959e2d4 is not registered to use Cloud Shell. Only red subscriptions will have restricted access to CloudShell service.

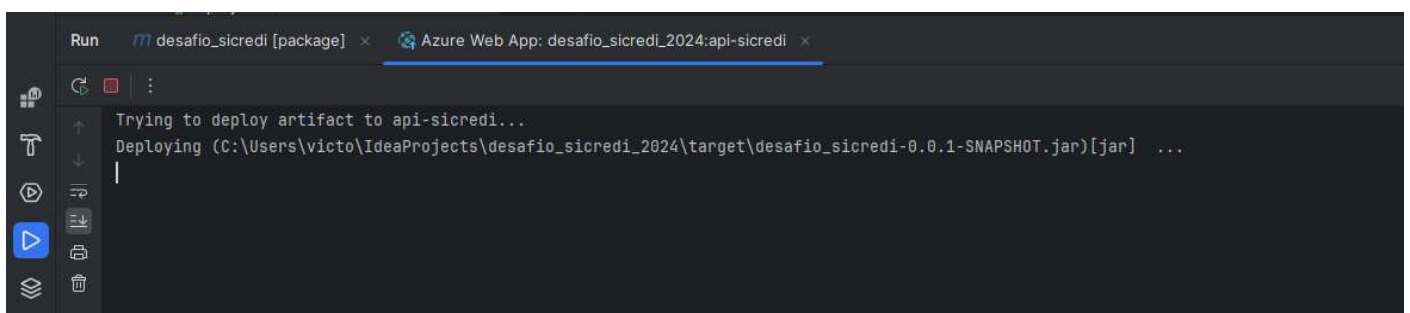
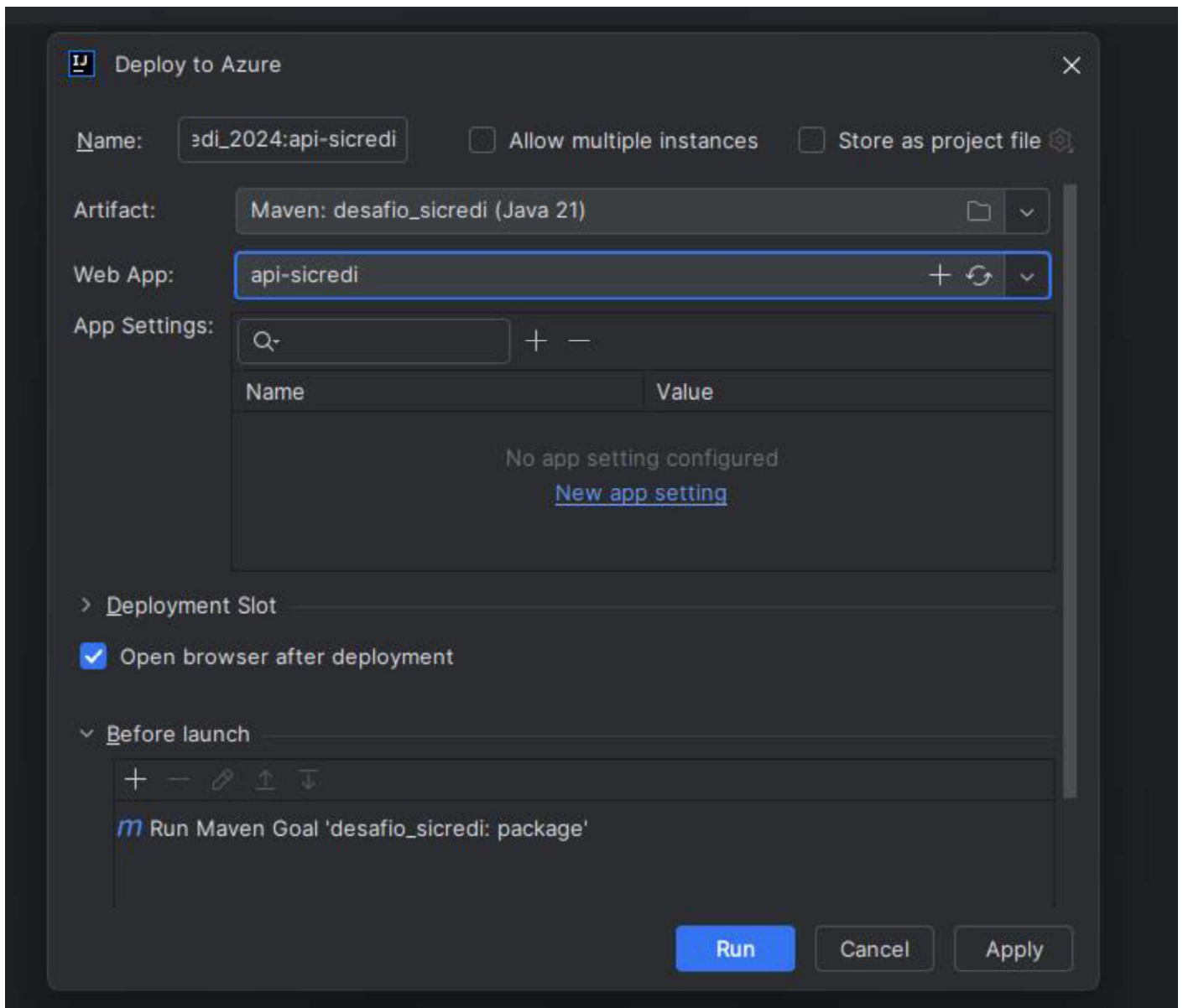
Your Cloud Shell session will be ephemeral so no files or system changes will persist beyond your current session. Press Ctrl+C to exit.











Trying to deploy artifact to api-sicredi...

Deploying (C:\Users\victo\IdeaProjects\desafio\_sicredi\_2024\target\desafio\_sicredi-0.0.1-SNAPSHOT.jar)[jar] ...

URL: <https://api-sicredi-gmdpfnebhmdabxd3.canadacentral-01.azurewebsites.net>

### **Link Swagger na Nuvem Azure:**

[api-sicredi-gmdpfnebhndabxd3.canadacentral-01.azurewebsites.net](https://api-sicredi-gmdpfnebhndabxd3.canadacentral-01.azurewebsites.net)

[api-sicredi-gmdpfnebhmdabxd3.canadacentral-01.azurewebsites.net](https://api-sicredi-gmdpfnebhmdabxd3.canadacentral-01.azurewebsites.net)

<https://api-sicredi-gmdpfnebhmdabxd3.canadacentral-01.azurewebsites.net/swagger-ui/index.html>

O serviço pode apresentar intermitência ou estar indisponível por conta das características do plano e limitação de créditos. De toda forma é possível testar localmente caso o serviço na nuvem não esteja disponível.

### **Solução alternativa para teste caso de indisponibilidade do webapp Azure:**

Rodando local não é necessário instalar banco de dados já que a aplicação aponta para base de dados do azure com firewall desativado.