

**Faculdade de Informática e Administração Paulista**

**Nome dos Integrantes:**

**Herbert Santos de Sousa: RM553227**

**João Pedro Pereira: RM553698**

**Enzo Franco Rocha: RM553643**

**Java Advanced**

**São Paulo**

**2024**

# Revolucionando o Mercado Odontológico

## Cronograma de Desenvolvimento da Sprint 1

Atividade	Responsável	Data Início	Data Término
Desenvolvimento das Entidades	Herbert Santos	01/09/2024	10/09/2024
Implementação de Controladores	Enzo Franco	11/09/2024	20/09/2024
Diagrama de Classes e DER	João Pedro	21/09/2024	25/09/2024
Configuração do Banco de Dados	João Pedro	26/09/2024	30/09/2024
Implementação de Testes de API	Herbert Santos	01/10/2024	05/10/2024

Na primeira sprint do projeto "Revolucionando o Mercado Odontológico", cada membro da equipe desempenhou papéis cruciais para o desenvolvimento da solução. Segue um resumo das atividades realizadas por cada integrante:

- Herbert Santos:** Foi responsável pelo desenvolvimento das entidades do projeto. Ele definiu as classes de domínio e mapeou as entidades com JPA e Hibernate, garantindo o correto encapsulamento e a tipagem adequada dos atributos. Além disso, conduziu a implementação de testes de API, criando e executando testes para verificar a funcionalidade dos endpoints e a correta persistência e recuperação dos dados.
- Enzo Franco:** Ficou encarregado da implementação dos controladores. Ele desenvolveu os controladores REST da aplicação, assegurando que as requisições HTTP fossem corretamente tratadas e que a aplicação seguisse os princípios RESTful. Enzo também garantiu a organização dos controladores e a implementação de métodos para CRUD de forma eficiente.
- João Pedro:** Foi o responsável por criar o diagrama de classes e o diagrama de entidade-relacionamento (DER), representando as relações entre as entidades e detalhando os relacionamentos e constraints envolvidas. Ele também cuidou da configuração do banco de dados, integrando a aplicação com o SGBD relacional (Oracle), configurando os parâmetros de conexão e validando a integridade dos dados.

Essas contribuições permitiram o desenvolvimento de uma base sólida para a aplicação, assegurando que todos os requisitos da primeira sprint fossem atendidos com qualidade e dentro do cronograma estipulado.

### Cronograma de Desenvolvimento da Sprint 2

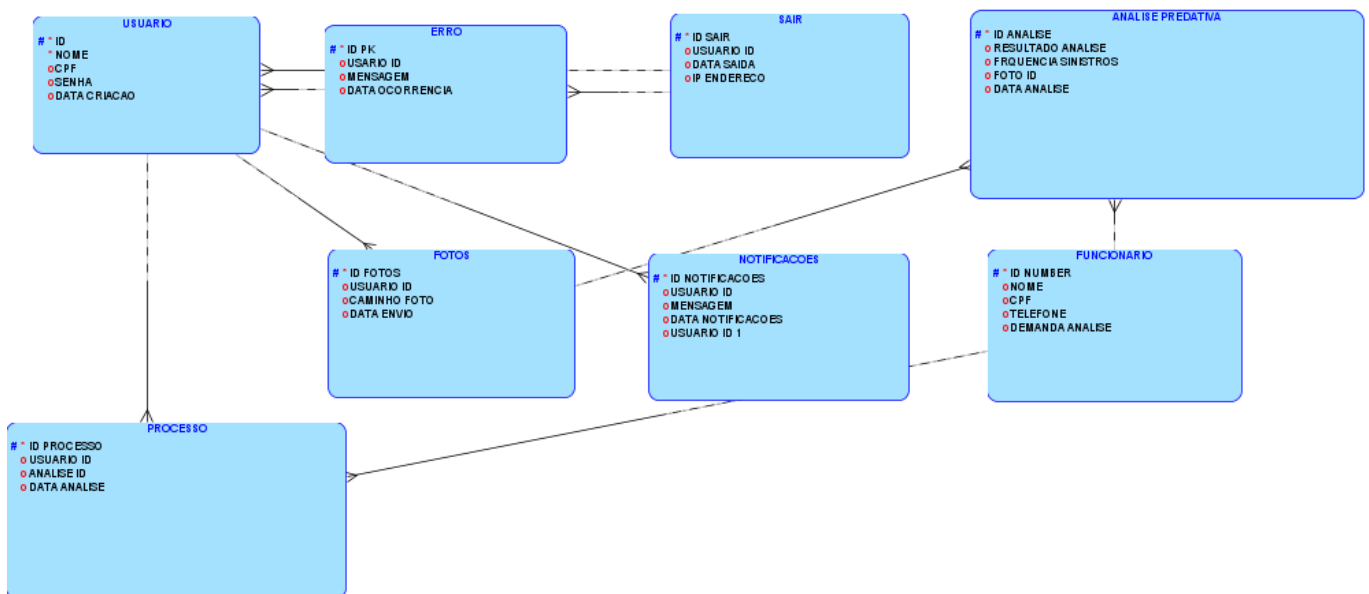
Atividade	Responsável	Data Início	Data Término
Criação da Classe AnalisePreditiva	Herbert Santos	06/10/2024	02/11/2024
Implementação de Controller, DTO, Repository e Service para a Classe Funcionario	Herbert Santos	15/09/2024	03/11/2024
Ajustes na Classe Notificação	Enzo Franco	26/10/2024	30/10/2024
Eliminação da Classe Sair	Herbert Santos	01/11/2024	02/11/2024
Implementação de HATEOAS (Nível 3 de Maturidade)	Herbert Santos	25/10/2024	04/11/2024
Uso de Lombok para Otimização de Código	João Pedro	10/10/2024	01/11/2024
Execução de Procedures via Aplicação Java	Herbert Santos	25/10/2024	04/11/2024
Diagramas MER e DER atualizados	João Pedro	30/10/2024	07/11/2024
Arquivo Postman_Collection adicionado	Herbert Santos	01/11/2024	02/11/2024

### Evoluções do Projeto Feitas Sprint 2

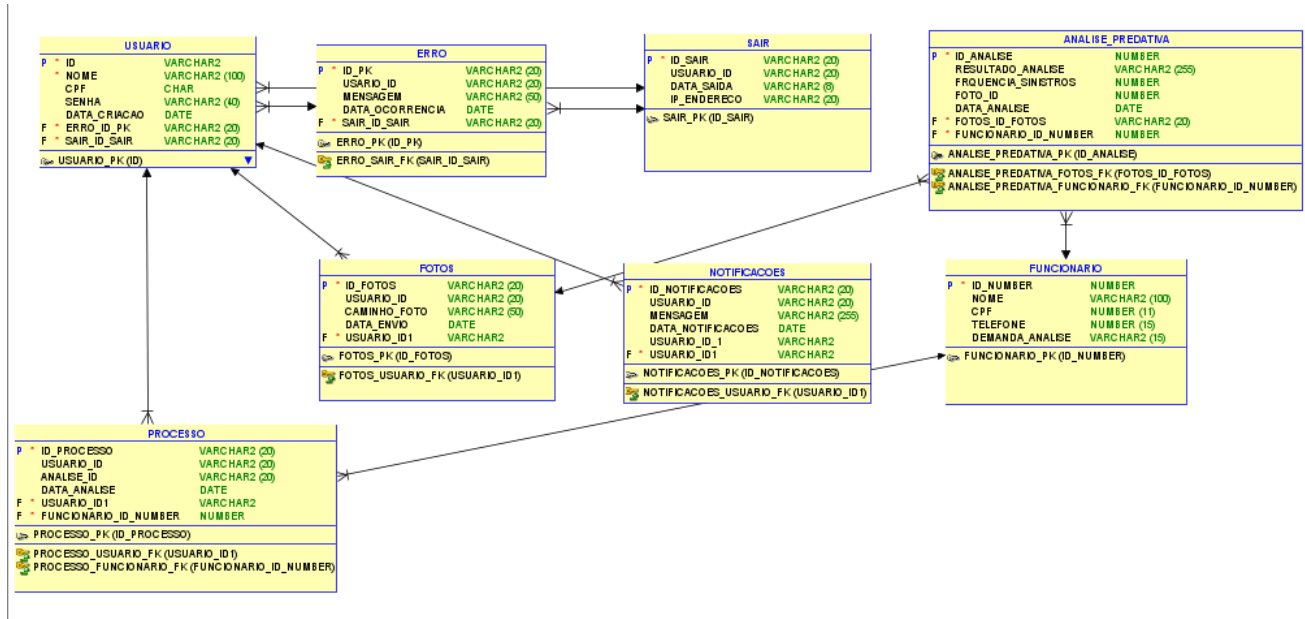
- Herbert Santos:** foi responsável por uma parte significativa das evoluções no projeto. Ele desenvolveu a classe Analise\_Preditiva, que foi projetada para analisar as fotos dos clientes e prever a aptidão para planos odontológicos e a frequência de sinistros, conectando-se a um modelo de machine learning. Herbert também implementou o controlador, DTO, repositório e serviço para a classe Funcionário, garantindo uma estrutura robusta e funcional. Além disso, ele realizou a eliminação da classe "Sair", considerada desnecessária, e implementou a integração de HATEOAS, levando a API ao nível 3 de maturidade de Richardson. Finalizando, ele executou a implementação de procedures na aplicação Java, permitindo operações de INSERT, UPDATE e DELETE, além da adição do arquivo Postman\_Collection.
- Enzo Franco:** contribuiu com ajustes específicos na classe Notificação, reformulando-a para que tivesse um propósito mais claro e coeso no projeto, alinhando-a às novas funcionalidades implementadas e ao fluxo de dados esperado.

- **João Pedro:** focou na otimização do código com o uso de Lombok, o que simplificou a manutenção e reduziu a verbosidade do projeto, tornando a base de código mais limpa e eficiente, além de atualização no MER e DER

### Diagrama de Classes



### Diagrama de Entidade-Relacionamento (DER)



### breve explicação sobre os relacionamentos e as constraints envolvidas

O banco de dados desenvolvido contém tabelas inter-relacionadas para gerenciar usuários, fotos, erros, processos, notificações, registros de saída, análises preditivas e dados de funcionários. As tabelas utilizam chaves primárias para garantir unicidade em cada registro e chaves estrangeiras para estabelecer relações entre as entidades. As constraints ON DELETE CASCADE e ON DELETE SET NULL definem o comportamento para exclusões, onde registros dependentes são excluídos ou têm seus valores nulos, conforme o caso.

Cada tabela é estruturada com integridade referencial e constraints de unicidade, como no campo cpf em usuarios, garantindo que cada usuário tenha um CPF único. Além disso, uma sequência (funcionario\_seq) foi configurada para gerar IDs automaticamente para os registros de funcionários, assegurando valores únicos. Esses relacionamentos e constraints promovem a integridade e a consistência dos dados no sistema.

## Instruções para Rodar a Aplicação

### Pré-requisitos

- JDK 17 ou superior
- Maven instalado
- Banco de dados Oracle ou outro banco de dados relacional configurado

### Passos para Rodar:

#### 1. Clone o repositório do GitHub:

```
git clone https://github.com/seu-usuario/repositorio.git
```

#### 2. Navegue até a pasta do projeto:

```
cd nome-do-projeto
```

#### 3. Configure as credenciais do banco de dados no arquivo `application.properties`:

```
spring.datasource.url=jdbc:oracle:thin:@localhost:1521:orcl  
spring.datasource.username=seu-usuario  
spring.datasource.password=sua-senha  
spring.jpa.hibernate.ddl-auto=update
```

**4. Compile o projeto:**

mvn clean install

**5. Execute a aplicação:**

mvn spring-boot:run

**6. Acesse a aplicação através da URL:**

http://localhost:8081

**Lista de endpoints para testar no Postman com base nos controladores fornecidos, usando o endereço base <http://localhost:8081>.**

**Listagem de Endpoints:****Endpoints de AnalisePreditiva**

- **GET /api/analises:** Lista todas as análises preditivas.
- **GET /api/analises/{id}:** Busca uma análise preditiva específica por ID.
- **POST /api/analises:** Cria uma nova análise preditiva.
- **DELETE /api/analises/{id}:** Exclui uma análise preditiva por ID.

**Endpoints de Erro**

- **GET /erros:** Lista todos os erros.
- **GET /erros/{id}:** Busca um erro específico por ID.
- **POST /erros:** Cria um novo erro.
- **PUT /erros/{id}:** Atualiza um erro existente por ID.
- **DELETE /erros/{id}:** Exclui um erro por ID.

**Endpoints de Foto**

- **GET /fotos:** Listar todas as fotos.
- **GET /fotos/{id}:** Buscar foto específica por ID.
- **POST /fotos:** Criar uma nova foto.
- **PUT /fotos/{id}:** Atualizar uma foto existente por ID.
- **DELETE /fotos/{id}:** Deletar uma foto por ID.

**Endpoints de Funcionario**

- **GET /api/funcionarios:** Listar todos os funcionários.
- **GET /api/funcionarios/{id}:** Buscar funcionário específico por ID.
- **POST /api/funcionarios:** Criar um novo funcionário.
- **DELETE /api/funcionarios/{id}:** Excluir um funcionário por ID.

**Endpoints de Notificacao**

- **GET /notificacoes:** Listar todas as notificações.

- **GET /notificacoes/{id}**: Buscar notificação específica por ID.
- **POST /notificacoes**: Criar uma nova notificação.
- **PUT /notificacoes/{id}**: Atualizar uma notificação existente por ID.
- **DELETE /notificacoes/{id}**: Deletar uma notificação por ID.

### Endpoints de Processo

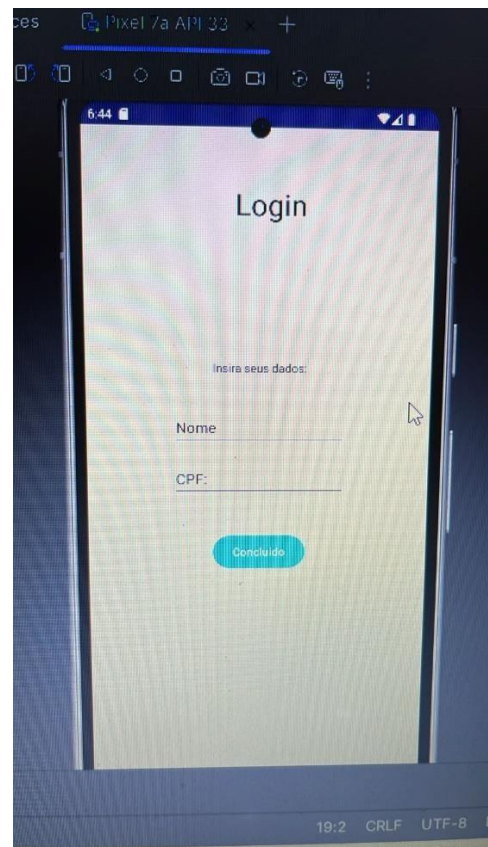
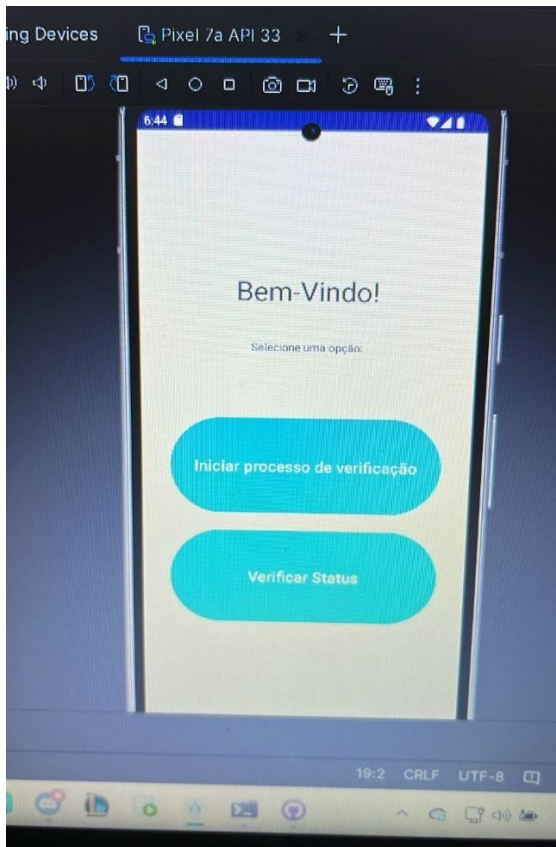
- **GET /processo** – Lista todos os processos.
- **GET /processo/{id}** – Busca um processo pelo ID.
- **POST /processo** – Cria um novo processo.
- **PUT /processo/{id}** – Atualiza um processo pelo ID.
- **DELETE /processo/{id}** – Deleta um processo pelo ID.

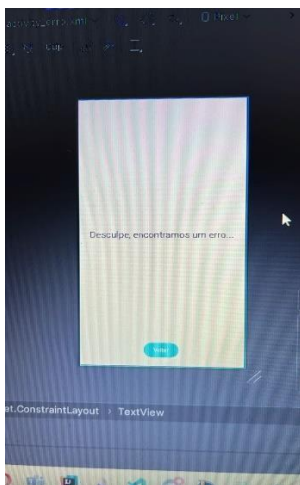
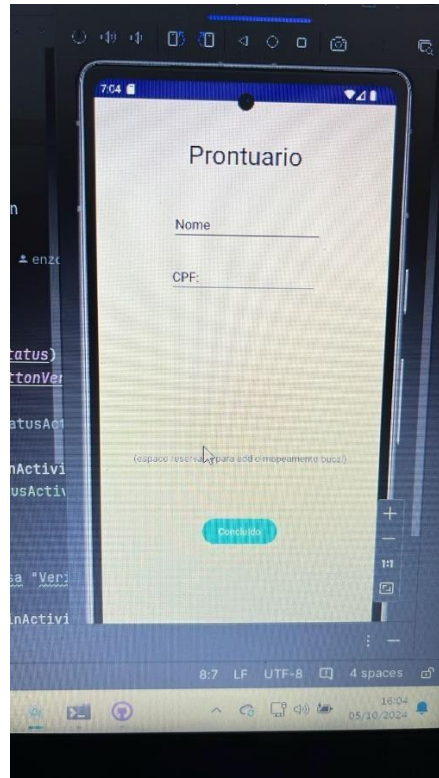
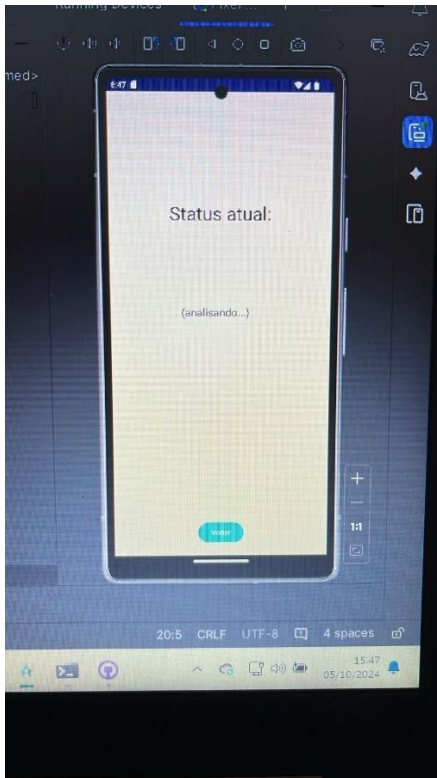
### Endpoints de Usuario

- **GET /usuarios** – Lista todos os usuários.
- **GET /usuarios/{id}** – Busca um usuário pelo ID, com links HATEOAS para navegação.
- **POST /usuarios** – Cria um novo usuário.
- **PUT /usuarios/{id}** – Atualiza um usuário pelo ID.
- **DELETE /usuarios/{id}** – Deleta um usuário pelo ID.

## Imagens Explicativas e Arquitetura







### Vídeo de Apresentação/Pitch:

Acesse o vídeo de apresentação da proposta tecnológica, o público-alvo da aplicação e os problemas que a aplicação se propõe a solucionar no link abaixo:

**LINK:** [https://www.youtube.com/watch?v=4pkziGL\\_koo](https://www.youtube.com/watch?v=4pkziGL_koo)

### Link do Github:

Todos os artefatos produzidos estão disponíveis no GitHub e os professores têm acesso ao repositório através do link:

**LINK:** <https://github.com/HerbertSsouza/SprintOdonto2>

### Teste dos Endpoints completo:

The screenshot shows a REST client interface with the following components:

- Top Bar:** Overview, GET http://local, GET http://local, GET http://localh, +, v, No environment, v.
- Collection Bar:** HTTP, New Collection / http://localhost:8083, Save, Share.
- Request Bar:** GET, http://localhost:8081/api/funcionarios, Send.
- Params Bar:** Params, Auth, Headers (7), Body, Scripts, Settings, Cookies.
- Query Params Table:**

Key	Value	Description	...	Bulk Edit
Key	Value	Description		
- Body Section:**
  - Status: 200 OK, 222 ms, 487 B, globe icon, e.g., ...
  - Tab: Pretty (selected), Raw, Preview, Visualize, JSON, v, icon, icon.
  - JSON Response (Pretty):

```

1  [
2    {
3      "id": 1,
4      "nome": "João Silva",
5      "cpf": "12345678901",
6      "cargo": "Analista de TI",
7      "salario": 3500.0,
8      "dataAdmissao": "2023-10-15",
9      "telefone": "11987654321",
10     "email": "joao.silva@email.com",
11     "links": [
12       {
13         "rel": "self",
14         "href": "http://localhost:8081/api/funcionarios/1"
15       },
16       {
17         "rel": "funcionarios",
18         "href": "http://localhost:8081/api/funcionarios"

```

Overview

GET http://local

GET http://local

GET http://lo

x

+

▼

No environment

▼

⌵

HTTP

New Collection / http://localhost:8083

Save

▼

Share

⌵

GET

▼

http://localhost:8081/processo

Send

▼

Params

Auth

Headers (7)

Body

Scripts

Settings

Cookies

</>

↕

i

Query Params

	Key	Value	Description	...	Bulk Edit
	Key	Value	Description		

Body

▼

200 OK

• 783 ms

• 901 B

• 🌐

• e.g.

• ...

Pretty

Raw

Preview

Visualize

JSON

▼

⌵

📄

🔍

```
4      "usuarioid": null,
5      "analiseId": 10,
6      "dataAnalise": "2024-10-01"
7    },
8    {
9      "idProcesso": 2,
10     "usuarioId": 2,
11     "analiseId": 20,
12     "dataAnalise": "2024-10-01"
13   },
14   {
15     "idProcesso": 3,
16     "usuarioId": 3,
17     "analiseId": 30,
18     "dataAnalise": "2024-10-01"
19   },
20   {
21     "idProcesso": 4,
```

Workspaces ▾ More ▾

🔍 👤 ⚙️ 🔔 🌐 Upgrade ▾ — □

👤 Overview GET http://local GET http://local GET http://local + ▾

🗑️ No environment ▾

📁 New Collection / http://localhost:8083

💾 Save ▾ Share

GET ▾ http://localhost:8081/erros

Send ▾

Params Auth Headers (7) Body Scripts Settings

Cookies

Query Params

	Key	Value	Description	⋮ Bulk Edit
	Key	Value	Description	

Body ▾

200 OK • 411 ms • 2.19 KB • 🌐 📄 ⋮

Pretty Raw Preview Visualize JSON ▾ 📄 🔍

```
4      "usuarioId": null,
5      "mensagem": "Erro de teste 1",
6      "dataOcorrencia": "2024-10-01",
7      "links": [
8        {
9          "rel": "self",
10         "href": "http://localhost:8081/erros/1"
11       },
12       {
13         "rel": "all-erros",
14         "href": "http://localhost:8081/erros"
15       }
16     ],
17     },
18     {
19       "idErro": 2,
20       "usuarioId": 2,
21       "mensagem": "Erro de teste 2",
```

OverviewGET http://localGET http://localGET http://local

No environment

New Collection / http://localhost:8083

SaveShare

GET

http://localhost:8081/fotos

Send

ParamsAuthHeaders (7)BodyScriptsSettings

Cookies

Query Params

	Key	Value	Description	Bulk Edit
	Key	Value	Description	

Body

200 OK • 402 ms • 1.02 KB

PrettyRawPreviewVisualizeJSON

```
1 [
2   {
3     "idFotos": 2,
4     "caminhoFoto": "/challenge/transferir2.jfif",
5     "dataEnvio": "2024-10-01",
6     "usuarioId": 2
7   },
8   {
9     "idFotos": 3,
10    "caminhoFoto": "/challenge/transferir3.jfif",
11    "dataEnvio": "2024-10-01",
12    "usuarioId": 3
13  },
14  {
15    "idFotos": 4,
16    "caminhoFoto": "/challenge/transferir4.jfif",
17    "dataEnvio": "2024-10-01",
18    "usuarioId": 4
```

OverviewGET http://localGET http://localGET http://local

No environment

New Collection / http://localhost:8083

SaveShare

GET

http://localhost:8081/api/analises

Send

ParamsAuthHeaders (7)BodyScriptsSettings

Cookies

Query Params

	Key	Value	Description	Bulk Edit
	Key	Value	Description	

Body

200 OK • 422 ms • 166 B

Overview

GET http://localhost:8080

GET http://localhost:8080

GET http://localhost:8080

+

▼

No environment

▼

HTTP

New Collection / http://localhost:8083

Save

▼

Share

GET

▼

http://localhost:8081/usuarios

Send

▼

Params

Auth

Headers (7)

Body

Scripts

Settings



Cookies

Query Params

	Key	Value	Description	...	Bulk Edit
	Key	Value	Description		

Body

▼

200 OK • 511 ms • 5.74 KB •  |  ...

Pretty


Raw


Preview


Visualize

JSON

▼







```
130      "idProcesso": 4,
131      "usuarioId": 99,
132      "analiseId": 40,
133      "dataAnalise": "2024-10-01",
134      "links": []
135    },
136  ],
137  "notificacoes": [
138    {
139      "idNotificacoes": 4,
140      "mensagem": "Notificação 4",
141      "dataNotificacao": "2024-10-01",
142      "lida": false,
143      "usuarioId": 4,
144      "links": [
145        {
146          "rel": "self",
147          "href": "/notificacoes/4"
```



Overview GET http://localhost GET http://localhost x GET http://localhost + No environment

New Collection / http://localhost:8083 Save Share

GET http://localhost:8081/notificacoes Send

Params Auth Headers (7) Body Scripts Settings Cookies

Query Params

Key	Value	Description	Bulk Edit
Key	Value	Description	

Body 200 OK • 1887 ms • 1.11 KB

Pretty Raw Preview Visualize JSON

```
1 [
2   {
3     "idNotificacoes": 2,
4     "mensagem": "Notificação 2",
5     "dataNotificacao": "2024-10-01",
6     "lida": false,
7     "usuarioId": 2
8   },
9   {
10    "idNotificacoes": 3,
11    "mensagem": "Notificação 3",
12    "dataNotificacao": "2024-10-01",
13    "lida": false,
14    "usuarioId": 3
15  },
16  {
17    "idNotificacoes": 4,
18    "mensagem": "Notificação 4"
```