**Documentação Técnica da Api Projeto Back-end**

* **Iniciar a Documentação da Api**

Foi pedida uma documentação técnica que retrata a realização de uma Api para um sistema de controle de locação de uma biblioteca.

O intuito de uma API é trocar dados entre sistemas diferentes ela permite a criação de novas funcionalidades. Ela também adiciona, atualiza, deleta e lista locações da biblioteca. Cada locação tem o nome do livro, nome do cliente, a data de locação e data de devolução.

• **Banco de Dados**

**Estrutura da tabela**

Foi criado uma tabela no banco de dados MySQL para o controle de locações da biblioteca.

CREATE DATABASE biblioteca\_db;

USE biblioteca\_db;

CREATE TABLE users (

id\_usuario INT AUTO\_INCREMENT PRIMARY KEY,

nome VARCHAR (100),

telefone VARCHAR (15),

email VARCHAR (255),

senha VARCHAR(100)

);

INSERT INTO users (nome, telefone, email, senha ) VALUES

("Ana Maria", "8691234790", "anamaria@gmail.com", "ana123"),

("João Lima", "1234790086", "joãolima@gmail.com", "joao56"),

("Karina Silva", "97698674837", "karinasilva@gmail.com", "ka20");

CREATE TABLE livros (

id\_livro INT AUTO\_INCREMENT PRIMARY KEY,

titulo VARCHAR (255),

autor VARCHAR (255),

isbn VARCHAR(100),

categoria VARCHAR(100)

);

INSERT INTO livros (titulo, autor, isbn, categoria) VALUES

("A Hipótese do Amor", "Ali Halzewood", '20240101', "Comédia Romantica"),

("O Fabricante de Lágrimas", "Erim Doom", '20240102', "Dark Romance"),

("A Guerra dos Tronos", "George M. Martin", '20240103', "Fantasia"),

("Princesa das Cinzas","Laura Sebastian", '20240104', "Fantasia");

CREATE TABLE Locações (

ID INT PRIMARY KEY AUTO\_INCREMENT,

ID\_Livro INT,

ID\_Usuário INT,

Data\_Locação DATE,

Data\_Devolução DATE,

Status VARCHAR(20)

);

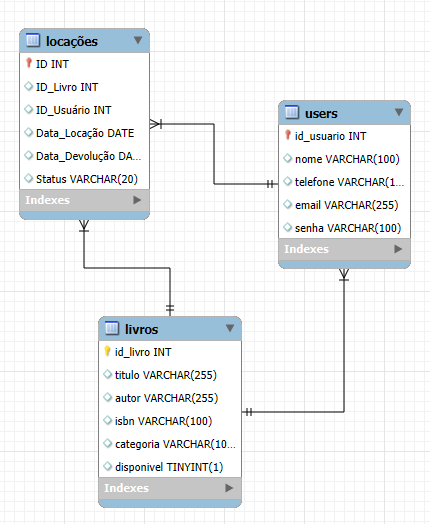
INSERT INTO locações (ID\_Livro , ID\_Usuário , Data\_Locação, Data\_Devolução, Status) VALUES

(2, 1, '2024-10-20', '2024-11-05', 'concluído'),

(3, 3, '2024-10-21', '2024-11-06', 'concluído'),

(1, 2, '2024-10-22', '2024-11-07', 'ativo');

* **Diagrama Entidade-Relacionamento (ERD):** Representação visual e seus relacionamentos.



• Estrutura do Projeto

Logo abaixo está uma descrição com pastas e arquivos do desenvolvimento da Api.

BIBLIOTECA/

├── config/ # armazenará arquivos de configuração

│ └── db.js

├── controllers/ #armazenará os arquivos de controladores da aplicação

│ └── authController.js

│ └── locacoesController.js

├── middlewares/ # para verificar se o token JWT está presente e válido

│ └── authMiddleware.js

├── routes/ #armazenará os arquivos de definição de rotas da aplicação

│ └── auth.js # responsáveis por solicitar a redefinição de senha e

realizar a redefinição.

│ └── locacoes.js

├── services/ # configurar o serviço de envio de e-mails.

│ └── emailService.js

├── .env #armazenar variáveis de ambiente

├── .gitignore

├── package.json # Arquivos com configurações padrão

├── package-lock.json # gerenciar as dependências e scripts do projeto

├── README

├── server.js #Configurar a conexão com o banco de dados

* **Teste das API**

Para testar todas as funcionalidades da API, foi realizado teste usando o Isomnia.

* **Listar todos as locações**
* Método: GET
* URL: <http://localhost:3000/biblioteca/locacoes>
* Resposta de Sucesso:

[

{

"ID": 1,

"ID\_Livro": 2,

"ID\_Usuário": 1,

"Data\_Locação": "2024-10-20T03:00:00.000Z",

"Data\_Devolução": "2024-11-05T03:00:00.000Z",

"Status": "concluído"

},

{

"ID": 2,

"ID\_Livro": 3,

"ID\_Usuário": 3,

"Data\_Locação": "2024-10-21T03:00:00.000Z",

"Data\_Devolução": "2024-11-06T03:00:00.000Z",

"Status": "concluído"

},

{

"ID": 3,

"ID\_Livro": 1,

"ID\_Usuário": 2,

"Data\_Locação": "2024-10-22T03:00:00.000Z",

"Data\_Devolução": "2024-11-07T03:00:00.000Z",

"Status": "ativo"

}

]

**Adicionar uma nova locação**

* Método: POST
* URL: <http://localhost:3000/biblioteca/locacoes>
* Código JSON

{ "ID": 4,

"ID\_Livro": 4,

"ID\_Usuário": 4,

"Data\_Locação": "2024-10-22T03:00:00.000Z",

"Data\_Devolução": "2024-11- 07T00:00.000Z",

"Status": "ativo"

}

* Resposta de Sucesso:

Locação adicionada com sucesso !!

* **Atualizar uma locação existente (atualização completa)**

Requisição:

* Método: PUT
* URL: http://localhost:3000/biblioteca/locacoes/2
* Corpo:

Código JSON

{

"ID": 2,

"ID\_Livro": 3,

"ID\_Usuário": 2,

"Data\_Locação": "2024-10-22",

"Data\_Devolução": "2024-11- 22",

"Status": "ativo”

}

* Resposta de Sucesso:

Locação atualizada com sucesso !!

* **Atualizar uma locação existente (atualização parcial)**

Requisição:

* Método: PATCH
* URL: http://localhost:3000/biblioteca/locacoes/2
* Corpo:

Código JSON

{

"Data\_Locação": "2024-11-22",

"Data\_Devolução": "2024-12- 22",

"Status": "ativo”

}

* Resposta de Sucesso:

Locação atualizada com sucesso !!

* **Deletar uma locação**

Requisição:

* Método: Delete
* URL: : http://localhost:3000/biblioteca/locacoes/4
* Resposta de sucesso:

Locação deletada com sucesso