**Conexão ao banco MySQL com Sequelize**

**- O que é uma ORM?**

Uma ORM, ou Object-Relational Mapping, é uma técnica de programação que visa facilitar a interação entre um sistema de gerenciamento de banco de dados relacional (como MySQL, PostgreSQL, SQLite, etc.) e um programa orientado a objetos. A ideia principal é mapear objetos do código-fonte para tabelas no banco de dados e vice-versa, eliminando a necessidade de escrever consultas SQL manualmente e permitindo que os desenvolvedores usem objetos e métodos para manipular dados.

As ORMs desempenham um papel crucial em ambientes onde o código é predominantemente orientado a objetos, como em muitas linguagens de programação modernas. Elas ajudam a abstrair a complexidade das operações de banco de dados, proporcionando uma interface mais intuitiva e facilitando a manutenção do código.

**ORM Sequilize**

O Sequelize é uma ORM (Object-Relational Mapping) para Node.js que suporta vários bancos de dados relacionais, incluindo MySQL, PostgreSQL, SQLite e MSSQL. ORM é um conceito que visa facilitar a interação entre sistemas de gerenciamento de banco de dados relacionais e código JavaScript, permitindo que os desenvolvedores usem objetos e métodos para manipular dados em vez de escrever consultas SQL manualmente.

npm install sequelize

**MySQL2**

O mysql2 é um driver MySQL para Node.js. Em termos simples, é uma biblioteca que permite que você se conecte a um banco de dados MySQL a partir do ambiente Node.js e execute consultas SQL de forma programática. O mysql2 é uma evolução do antigo driver mysql para Node.js, oferecendo melhor desempenho e suporte a algumas funcionalidades adicionais.

npm install mysql2

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

**sequelize-config.js:**

// *Importando o Sequelize*

*import* Sequelize *from* "sequelize"

// *Criando os dados de conexão com o banco de dados*

constconnection=**new** *Sequelize*({

*dialect*:'mysql',

*host*:'localhost',

*username*:'root',

*password*:'admin',

    // *Comente essa linha na primeira execução da aplicação*

// *database: 'loja',*

*timezone*:"-03:00"

})

*export* *default* connection

**index.js:**

// *Importando o Sequelize*

*import* connection *from* "./config/sequelize-config.js"

// *Realizando a conexão com o banco de dados*

connection*.authenticate*()*.then*(()=> {

    console*.log*("Conexão com o banco de dados feita com sucesso!")

})*.catch*((error) => {

    console*.log*(error)

})

// *Criando o banco de dados se ele não existir*

connection*.query*(`CREATE DATABASE IF NOT EXISTS loja;`)*.then*(() => {

    console*.log*("O banco de dados está criado.")

})*.catch*((error) => {

    console*.log*(error)

})

Interface gráfica do usuário

Descrição gerada automaticamente com confiança baixa

Interface gráfica do usuário, Texto

Descrição gerada automaticamente­­

**Cliente.js:**

*import* Sequelize *from* "sequelize"

*import* connection *from* "../config/sequelize-config.js"

constCliente=connection*.define*('clientes',{

*nome*:{

type: *Sequelize.*STRING,

*allowNull*: *false*

},

*cpf*:{

type: *Sequelize.*STRING,

*allowNull*: *false*

},

*endereco*:{

type: *Sequelize.*STRING,

*allowNull*: *false*

}

})

Cliente*.sync*({force: false})

*export* *default* Cliente

**index.js:**

// *Importando os Models*

*import* Cliente *from* "./models/Cliente.js"

Interface gráfica do usuário, Aplicativo, Word

Descrição gerada automaticamente

Interface gráfica do usuário, Tabela

Descrição gerada automaticamente



**ClientesControllers.js:**

*import* express *from* "express"

*import* Cliente *from* "../models/Cliente.js"

constrouter= *express.Router*()

// *ROTA CLIENTES*

router*.get*("/clientes", function(req, res){

    Cliente*.findAll*()*.then*(clientes => {

        res*.render*("clientes", {

            clientes: clientes

        })

    })

})

*export* *default* router //*exportando o módulo "router"*

**clientes.ejs**

<%- include ('partials/header.ejs') %>

<%- include ('partials/navbar.ejs') %>

<div *class*="container">

    <div *class*="card">

        <div *class*="card-header">

            <h1>Clientes</h1>

        </div>

        <div *class*="card-body">

            <table>

                <thead>

                    <tr>

                        <th>Nome:</th>

                        <th>CPF:</th>

                        <th>Endereço:</th>

                    </tr>

                </thead>

                <tbody>

                    <% clientes.forEach(cliente => { %>

                    <tr>

                        <td><%= cliente.nome %></td>

                        <td><%= cliente.cpf %></td>

                        <td><%= cliente.endereco %></td>

                    </tr>

                    <% }) %>

                </tbody>

            </table>

        </div>

    </div>

</div>

<%- include ('partials/footer.ejs') %>

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente