Projeto com Node.js

Thyago de Sousa Gonçalves

1. Código do index.html

O arquivo index.html contém um formulário simples com campos para ID, nome, e-mail e telefone, além de botões para cadastrar e atualizar.

Formulário index.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Cadastro de Alunos</title>
</head>
<body>
  <h1>Cadastro de Alunos</h1>
  <form action="/cadastrar" method="POST">
    <label for="id">ID:</label>
    <input type="text" id="id" name="id"><br>
    <label for="nome">Nome:</label>
    <input type="text" id="nome" name="nome"><br>
    <label for="email">Email:</label>
    <input type="email" id="email" name="email"><br>
    <label for="telefone">Telefone:</label>
    <input type="text" id="telefone" name="telefone"><br>
    <button type="submit">Cadastrar</button>
  </form>

  <h1>Atualizar Aluno</h1>
  <form action="/atualizar" method="POST">
    <label for="id">ID:</label>
    <input type="text" id="id" name="id"><br>
    <label for="nome">Nome:</label>
    <input type="text" id="nome" name="nome"><br>
    <label for="email">Email:</label>
    <input type="email" id="email" name="email"><br>
```

```
    <label for="telefone">Telefone:</label>
    <input type="text" id="telefone" name="telefone"><br>
    <button type="submit">Atualizar</button>
  </form>
</body>
</html>
```

2. Código do app.js

O arquivo app.js é responsável por definir as rotas. Ele inclui a rota principal para exibir o formulário (index.html), uma rota para cadastrar um novo aluno, e outra para atualizar os dados de um aluno.

```
const express = require('express');
const path = require('path');
const bodyParser = require('body-parser');
const dao = require('./dao');

const app = express();
app.use(bodyParser.urlencoded({ extended: true }));

// Servir a página inicial
app.get('/', (req, res) => {
  res.sendFile(path.join(__dirname, 'index.html'));
});

// Rota para cadastrar aluno
app.post('/cadastrar', (req, res) => {
  const { id, nome, email, telefone } = req.body;
  dao.gravarAluno(id, nome, email, telefone);
  res.send('Aluno cadastrado com sucesso!');
});

// Rota para atualizar aluno
app.post('/atualizar', (req, res) => {
  const { id, nome, email, telefone } = req.body;
  dao.atualizarAluno(id, nome, email, telefone);
  res.send('Aluno atualizado com sucesso!');
});
```

```
// Iniciar o servidor
app.listen(3000, () => {
  console.log('Servidor rodando na porta 3000');
});
```

3. Código do dao.js

O arquivo dao.js contém os métodos para gravar e atualizar os dados no banco de dados MySQL.

```
const mysql = require('mysql');

const connection = mysql.createConnection({
  host: 'localhost',
  user: 'root',
  password: '1234',
  database: 'escola'
});

connection.connect((err) => {
  if (err) {
    console.error('Erro ao conectar ao banco de dados: ', err);
    return;
  }
  console.log('Conectado ao banco de dados MySQL.');
});

// Método para gravar aluno
function gravarAluno(id, nome, email, telefone) {
  const query = 'INSERT INTO aluno (id, nome, email, telefone) VALUES (?, ?, ?, ?)';
  connection.query(query, [id, nome, email, telefone], (err, results) => {
    if (err) {
      console.error('Erro ao inserir aluno: ', err);
      return;
    }
    console.log('Aluno inserido com sucesso.');
  });
}

// Método para atualizar aluno
```

```
function atualizarAluno(id, nome, email, telefone) {
   const query = 'UPDATE aluno SET nome = ?, email = ?, telefone = ? WHERE id = ?';
   connection.query(query, [nome, email, telefone, id], (err, results) => {
      if (err) {
         console.error('Erro ao atualizar aluno: ', err);
         return;
      }
      console.log('Aluno atualizado com sucesso.');
   });
}

module.exports = { gravarAluno, atualizarAluno };
```