CENTRO UNIVERSITÁRIO SENAC

Vinicius Cunha Martins

DESENVOLVIMENTO API REST
CATÁLOGO DE PRODUTO

Nova Iguaçu
2023

Vinicius Cunha Martins

CATÁLOGO DE PRODUTO

Trabalho de Produção de Texto Individual apresentado ao SENAC – Nova Iguaçu, como exigência parcial para obtenção do grau de Tecnólogo em Sistemas para Internet.

Nova Iguaçu
2023

Link do repositório Github: https://github.com/vinecunha/catalogoroupas.git

Primeira Parte:

Vamos considerar o ramo de uma empresa de roupas como exemplo para a modelagem da tabela de roupa.

A modelagem da tabela de roupa pode ser feita da seguinte forma:

Tabela "roupa":

id_roupa (chave primária)
nome_roupa (texto)
descrição (texto)
preço (decimal)
quantidade (inteiro)
data de criação (data/hora)
data de atualização (data/hora)

Segunda Parte:

```
// Rota POST para adicionar uma nova roupa
router.post('/roupa', async (req, res) => {
    const novaRoupa = {
        "nome_roupa": "Camiseta",
        "descricao": "Uma camiseta de algodão",
        "preco": 29.99,
        "quantidade": 10,
        "data_de_criacao": "2023-05-14",
        "data_de_atualizacao": "2023-05-14"
      }
    const { nome_roupa, descricao, preco, quantidade, data_de_criacao, data_de_atu-
alizacao } = novaRoupa;

    // Realize a validação dos dados recebidos (pode ser mais complexa dependendo
dos requisitos do sistema)
    if (!nome_roupa || !descricao || !preco || !quantidade) {
      return res.status(400).json({ mensagem: 'Dados incompletos. Todos os campos
são obrigatórios.' });
    }

    try {
      const result = await con.query('INSERT INTO roupa (nome_roupa, descricao,
preco, quantidade, data_de_criacao, data_de_atualizacao) VALUES (?, ?, ?, ?, ?,
?)', [nome_roupa, descricao, preco, quantidade, data_de_criacao, data_de_atualiza-
cao]);

      // Em caso de sucesso, retorna uma resposta de sucesso
      return res.status(201).json({ mensagem: 'Roupa incluída com sucesso!', roupa:
{ id: result.insertId, nome_roupa, descricao, preco, quantidade, data_de_criacao,
data_de_atualizacao } });
```

```
    } catch (error) {
      console.error('Erro ao salvar a roupa:', error);
      // Em caso de erro, retorna uma resposta de erro
      return res.status(500).json({ mensagem: 'Ocorreu um erro ao salvar a roupa.'
});
    }
  });
```

```
// Rota para buscar roupas por nome
router.get('/roupa', async (req, res) => {
    const { nome } = req.query;

    try {
      const [roupas] = await con.query('SELECT * FROM roupa WHERE nome_roupa LIKE
?', [`%${nome}%`]);

      // Em caso de sucesso, retorna as roupas encontradas
      return res.status(200).json({ roupas });
    } catch (error) {
      console.error('Erro ao buscar as roupas:', error);
      // Em caso de erro, retorna uma resposta de erro
      return res.status(500).json({ mensagem: 'Ocorreu um erro ao buscar as rou-
pas.' });
    }
  });
```

```
// Rota para atualizar uma roupa por ID
router.put('/roupa/:id', (req, res) => {
    const { id } = req.params;
    const novaRoupaAtualizada = {
        "nome_roupa": "Camiseta 2",
        "descricao": "Uma segunda camiseta de algodão",
        "preco": 59.99,
        "quantidade": 10,
        "data_de_criacao": "2023-05-14",
        "data_de_atualizacao": "2023-05-14"
      }
    const { nome_roupa, descricao, preco, quantidade, data_de_criacao, data_de_atu-
alizacao } = novaRoupaAtualizada;

    // Realiza a validação dos dados recebidos (pode ser mais complexa dependendo
dos requisitos do sistema)
    if (!nome_roupa || !descricao || !preco || !quantidade) {
      return res.status(400).json({ mensagem: 'Dados incompletos. Todos os campos
são obrigatórios.' });
    }

    // Monta a query para atualizar a roupa no banco de dados
    const query = `
      UPDATE roupa
```

```
      SET nome_roupa = ?, descricao = ?, preco = ?, quantidade = ?, data_de_criacao
= ?, data_de_atualizacao = ?
      WHERE id_roupa = ?
    `;
    const values = [nome_roupa, descricao, preco, quantidade, data_de_criacao,
data_de_atualizacao, id];

    // Executa a query de atualização no banco de dados
    con.query(query, values, (err, result) => {
      if (err) {
        // Em caso de erro, retorna uma resposta de erro
        return res.status(500).json({ mensagem: 'Ocorreu um erro ao atualizar a
roupa.' });
      }

      // Verifica se a atualização foi bem-sucedida
      if (result.affectedRows === 0) {
        // Caso nenhum registro tenha sido afetado, significa que a roupa não foi
encontrada
        return res.status(404).json({ mensagem: 'Roupa não encontrada.' });
      }

      // Em caso de sucesso, retorna uma resposta de sucesso
      return res.status(200).json({ mensagem: 'Roupa atualizada com sucesso!' });
    });
  });
```

```
// Rota para excluir uma roupa por ID
router.delete('/roupa/:id', (req, res) => {
    const { id } = req.params;

    // Monta a query para excluir a roupa no banco de dados
    const query = `
      DELETE FROM roupa
      WHERE id_roupa = ?
    `;
    const values = [id];

    // Executa a query de exclusão no banco de dados
    con.query(query, values, (err, result) => {
      if (err) {
        // Em caso de erro, retorna uma resposta de erro
        return res.status(500).json({ mensagem: 'Ocorreu um erro ao excluir a
roupa.' });
      }

      // Verifica se a exclusão foi bem-sucedida
      if (result.affectedRows === 0) {
        // Caso nenhum registro tenha sido afetado, significa que a roupa não foi
encontrada
        return res.status(404).json({ mensagem: 'Roupa não encontrada.' });
```

```
    }

    // Em caso de sucesso, retorna uma resposta de sucesso
    return res.status(200).json({ mensagem: 'Roupa excluída com sucesso!' });
  });
});
```