

Atualizando Transações com PUT e PATCH

Nesta aula, vamos aprender a criar rotas e controladores para atualizar transações na nossa API usando os métodos **PUT** e **PATCH**.

Objetivos da Aula:

- Compreender como criar rotas para atualizar transações
- Configurar controladores para manipular as requisições de atualização de transações

1. Configurando as Rotas para Atualizar Transações

Atualizar o Arquivo transactions.js

Abra o arquivo **transactions.js** dentro da pasta **routes**:

Vamos adicionar novas rotas **PUT** e **PATCH** para atualizar transações.

Adicione o seguinte código ao arquivo **transactions.js**:

Código JavaScript

```
const express = require('express'); // Importa o framework Express
const router = express.Router(); // Cria um novo roteador
const transactionsController = require('../controllers/transactionsController'); // Importa o controlador de transações
```

```
// Definindo uma rota para obter todas as transações
router.get('/', transactionsController.getAllTransactions);
```

```
// Definindo uma rota para adicionar uma nova transação
router.post('/', transactionsController.addTransaction);
```

```
// Definindo uma rota para atualizar uma transação existente (substituição completa)
⇒ router.put('/:id', transactionsController.updateTransactionPut);
```

```
// Definindo uma rota para atualizar uma transação existente (atualização parcial)
⇒ router.patch('/:id', transactionsController.updateTransactionPatch);
```

```
// Exportando o roteador
module.exports = router;
```

Explicação do Código:

- **router.put('/:id', transactionsController.updateTransactionPut);**: Define uma rota PUT para atualizar uma transação específica.

- `router.patch('/:id', transactionsController.updateTransactionPatch);` Define uma rota PATCH para atualizar uma transação específica.

Configurar o Controlador para Atualizar Transações

Abra o arquivo `transactionsController.js` dentro da pasta `controllers`:

Vamos adicionar novas funções para atualizar transações usando **PUT** e **PATCH**.

Adicione o seguinte código ao arquivo `transactionsController.js`:

Código JavaScript

```
const db = require('../config/db'); // Importa a conexão com o banco de dados
```

```
// Função para obter todas as transações
```

```
const getAllTransactions = (req, res) => {
  db.query('SELECT * FROM transactions', (err, results) => {
    if (err) {
      console.error('Erro ao obter transações:', err);
      res.status(500).send('Erro ao obter transações');
      return;
    }
    res.json(results);
  });
};
```

```
// Função para adicionar uma nova transação
```

```
const addTransaction = (req, res) => {
  const { date, amount, description, category, account, user_id } = req.body;
  db.query(
    'INSERT INTO transactions (date, amount, description, category, account, user_id) VALUES'
    '(?, ?, ?, ?, ?, ?)',
    [date, amount, description, category, account, user_id],
    (err, results) => {
      if (err) {
        console.error('Erro ao adicionar transação:', err);
        res.status(500).send('Erro ao adicionar transação');
        return;
      }
      res.status(201).send('Transação adicionada com sucesso');
    }
  );
};
```

```
// Função para atualizar uma transação existente (substituição completa)
const updateTransactionPut = (req, res) => {
  const { id } = req.params;
  const { date, amount, description, category, account, user_id } = req.body;
  db.query(
    'UPDATE transactions SET date = ?, amount = ?, description = ?, category = ?, account = ?,
    user_id = ? WHERE id = ?',
    [date, amount, description, category, account, user_id, id],
    (err, results) => {
      if (err) {
        console.error('Erro ao atualizar transação:', err);
        res.status(500).send('Erro ao atualizar transação');
        return;
      }
      res.send('Transação atualizada com sucesso');
    }
  );
};
```

```
// Função para atualizar uma transação existente (atualização parcial)
const updateTransactionPatch = (req, res) => {
  const { id } = req.params;
  const fields = req.body;
  const query = [];
  const values = [];

  for (const [key, value] of Object.entries(fields)) {
    query.push(`${key} = ?`);
    values.push(value);
  }

  values.push(id);

  db.query(
    `UPDATE transactions SET ${query.join(', ')} WHERE id = ?`,
    values,
    (err, results) => {
      if (err) {
        console.error('Erro ao atualizar transação:', err);
        res.status(500).send('Erro ao atualizar transação');
        return;
      }
      res.send('Transação atualizada com sucesso');
    }
  );
};
```

```
module.exports = {
  getAllTransactions,
  addTransaction,
  updateTransactionPut,
  updateTransactionPatch
};
```

Explicação do Código:

- **const updateTransactionPut = (req, res) => {...};** Define uma função arrow chamada updateTransactionPut que recebe dois parâmetros, req (requisição) e res (resposta). Esta função será usada para atualizar uma transação existente no banco de dados com uma substituição completa.
- **const updateTransactionPatch = (req, res) => {...};** Define uma função arrow chamada updateTransactionPatch que recebe dois parâmetros, req (requisição) e res (resposta). Esta função será usada para atualizar uma transação existente no banco de dados com uma atualização parcial.

Parte Prática (1 hora):

Ação:

1. Atualizar o arquivo **transactions.js** para definir as rotas de atualização de transações.
2. Atualizar o arquivo **transactionsController.js** para definir os controladores de atualização de transações.
3. Testar as rotas **PUT** e **PATCH** para **/api/transactions/:id** para garantir que transações podem ser atualizadas.

Teste Prático com Insomnia:

Vamos usar o Insomnia para testar as rotas PUT e PATCH para **/api/transactions/:id**.

Testando a Rota PUT para Atualizar Transações

1. **Abra o Insomnia.**
2. **Crie uma Nova Requisição:**
 - No menu lateral esquerdo, clique no botão "+" ao lado de "Debug" para adicionar uma nova requisição.
 - Dê um nome à sua requisição, por exemplo, "Update Transaction (PUT)".
 - Selecione o método HTTP como "PUT".
 - Clique em "Create".
3. **Configure a URL da Requisição:**
 - Na barra de URL, insira **http://localhost:3000/api/transactions/1** (onde 1 é o id da transação que deseja atualizar).

4. Configure o Corpo da Requisição:

- Na aba "Body", selecione "JSON".
- Adicione o seguinte JSON:

Código JSON

```
{
  "date": "2023-07-08",
  "amount": 300.00,
  "description": "Viagem atualizada",
  "category": "Lazer",
  "account": "Cartão de Crédito",
  "user_id": 1
}
```

5. Envie a Requisição:

- Clique no botão "Send" para enviar a requisição.

6. Verifique a Resposta:

- Verifique o painel de resposta no Insomnia.
- A resposta deve indicar que a transação foi atualizada com sucesso.

Exemplo de Resposta Esperada para Atualização com PUT

Código JSON

```
{
  "message": "Transação atualizada com sucesso"
}
```

Testando a Rota PATCH para Atualizar Transações

1. Crie uma Nova Requisição:

- No menu lateral esquerdo, clique no botão "+" ao lado de "Debug" para adicionar uma nova requisição.
- Dê um nome à sua requisição, por exemplo, "Update Transaction (PATCH)".
- Selecione o método HTTP como "PATCH".
- Clique em "Create".

2. Configure a URL da Requisição:

- Na barra de URL, insira `http://localhost:3000/api/transactions/1` (onde 1 é o id da transação que deseja atualizar).

3. Configure o Corpo da Requisição:

- Na aba "Body", selecione "JSON".
- Adicione o seguinte JSON:

Código JSON

```
{
  "amount": 400.00,
  "description": "Viagem parcialmente atualizada"
}
```

4. Envie a Requisição:

- Clique no botão "Send" para enviar a requisição.

5. Verifique a Resposta:

- Verifique o painel de resposta no Insomnia.
- A resposta deve indicar que a transação foi atualizada com sucesso.

Exemplo de Resposta Esperada para Atualização com PATCH

Código JSON

```
{
  "message": "Transação atualizada com sucesso"
}
```

Resolução de Problemas Comuns

Erro 404 (Not Found)

- Verifique se o servidor está rodando corretamente.
- Certifique-se de que a URL está correta e que a rota `/api/transactions/:id` está configurada no `server.js`.

Erro 500 (Internal Server Error)

- Verifique os logs do servidor para identificar a causa do erro.
- Certifique-se de que a conexão com o banco de dados está configurada corretamente.