Modelagem API

**\*TRANSAÇÕES\***

**Criar Transação (POST)**

Endpoint: POST /transacoes

Requisição:

{

"remetente\_id": "id\_do\_remetente",

"destinatario\_id": "id\_do\_destinatario",

"valor": 100.0

}

Retorno: "Transação Criada com sucesso"

Erros esperados:

404 Not Found: Não existe usuário remetente ou destinatário com este ID.

403 Forbidden: Saldo insuficiente no remetente.

400 Bad Request: O valor da transação deve ser maior que zero.

**Consultar Transação (GET)**

Endpoint: GET /transacoes/{transacao-id}

Retorno:

{

" id" : "id\_da\_transacao",

" data\_transacao" : "data\_da\_transacao",

"remetente\_id": "id\_do\_remetente",

"destinatario\_id": "id\_do\_destinatario",

"valor": 100.0,

"status": "concluida"

}

Erros esperados:

404 Not Found: Transação não encontrada.

400 Bad Request: ID da transação inválido.

**Erros esperados:**

404 Not Found: Transação não encontrada.

400 Bad Request: ID da transação inválido.

**Atualizar Status da Transação (PATCH)**

Endpoint: PATCH /transacoes/{transacao-id}

Requisição:

{

"status": "concluida"

}

Retorno: "Status atualizado!"

Erros esperados:

404 Not Found: Não existe transação com este ID.

400 Bad Request: O status da transação deve ser um dos valores permitidos (por exemplo, "pendente", "concluida", "cancelada").

**Cancelar Transação (DELETE)**

Endpoint: DELETE /transacoes/{transacao-id}

Retorno: "Transação Cancelada!"

Erro esperado:

404 Not Found: Transação não encontrada.

400 Bad Request: Id da transação inválido.

### **Transferência de Fundos entre Usuários**

**Ator Principal: Usuário**

**Pré-condições:**

* O usuário possui saldo suficiente para realizar a transação.
* Os IDs do remetente e do destinatário são válidos.

**Fluxo Principal:**

1. O usuário envia uma solicitação para criar uma transação usando o método POST para o endpoint /transacoes, fornecendo as informações necessárias (ID do remetente, ID do destinatário e valor).
2. O sistema verifica se o remetente e o destinatário existem. Se algum deles não existir, retorna um erro 404 Not Found.
3. O sistema verifica se o saldo do remetente é suficiente para a transação. Se não for, retorna um erro 403 Forbidden.
4. O sistema verifica se o valor da transação é maior que zero. Se não for, retorna um erro 400 Bad Request.
5. Se todas as verificações passarem, a transação é criada e retorna um código de status 201 Created, juntamente com os detalhes da transação.

**Fluxos Alternativos:**

* Se houver um erro durante a criação da transação (por exemplo, falha no servidor), retorna um código de erro apropriado (por exemplo, 500 Internal Server Error).

**Pós-condições:**

* A transação é criada com sucesso e reflete a dedução do saldo do remetente e a adição ao saldo do destinatário.

**Outros Casos de Uso:**

* Consultar Transação (GET): Permite que o usuário obtenha detalhes sobre uma transação específica, fornecendo o ID da transação.
* Atualizar Status da Transação (PATCH): Permite que o usuário atualize o status de uma transação (por exemplo, de "pendente" para "concluída").
* Cancelar Transação (DELETE): Permite que o usuário cancele uma transação específica, fornecendo o ID da transação.

**\*USUÁRIOS\***

**Criar Usuario(POST)**

Endpoint: POST /usuario

Requisição:

{

"nome": "nome",

"data\_nascimento": "data\_nascimento",

"telefone": "telefone",

"endereco":{

"cidade": "cidade",

"bairro":"bairro",

"CEP":"CEP",

"Numero": "numero

},

"CPF": “1234567-89”

}

Retorno: "Usuário criado com sucesso"

Erros esperados:

400 Bad Request: O valor do cpf inválido.

**Consultar Transação (GET)**

Endpoint: GET /usuario/{usuario-id}

Retorno:

{

" id" : "id\_do\_usuario",

"nome": "nome",

"data\_nascimento": "data\_nascimento",

"telefone": "telefone",

"endereco":{

"cidade": "cidade",

"bairro":"bairro",

"CEP":"CEP",

"Numero": "numero

},

"CPF": “1234567-89”

"saldo": “9999999999,99”

Erros esperados:

404 Not Found: Id do usuário não encontrado.

400 Bad Request: Id do usuário inválido.

### **Gestão de Usuários**

**Ator Principal:** Administrador do sistema

**Pré-condições:**

* O administrador possui permissões adequadas para acessar e gerenciar usuários.

**Fluxo Principal:**

1. O administrador envia uma solicitação para criar um novo usuário usando o método POST para o endpoint /usuario, fornecendo as informações necessárias (nome, data de nascimento, telefone, endereço, CPF).
2. O sistema valida o formato do CPF fornecido. Se for inválido, retorna um erro 400 Bad Request.
3. Se o CPF for válido, o sistema cria o usuário com sucesso e retorna uma mensagem "Usuário criado com sucesso".

**Fluxos Alternativos:**

* Se houver um erro durante a criação do usuário (por exemplo, CPF já existente), retorna um código de erro apropriado.

**Pós-condições:**

* O novo usuário é adicionado ao sistema e pode ser consultado posteriormente.

**Outro Caso de Uso:**

* Consultar Usuário (GET): Permite que o administrador obtenha detalhes sobre um usuário específico, fornecendo o ID do usuário.