Obrigatórios e Script SQL Server

FinTECH é uma aplicação para um banco digital, com o objetivo de registrar as principais operações para um cliente.
As funcionalidades estão descritas a seguir:

1. Funcionalidades Básicas
As funcionalidades básicas são esperadas nos projetos, e contemplam:

Login:
A entrada no sistema deve ser realizada com o número da agência e da conta, além de uma senha, quando for realizado por um cliente.
Como administrador, deve haver um usuário e uma senha.

Criação e Manutenção de Contas:
Como gerente (administrador) o usuário poderá registrar novas contas. No ato do cadastro, a conta é considerada simples e, ao longo do tempo, novos recursos poderão ser implementados, como por exemplo, cartão de credito internacional, redução de tarifas, entre outras.

Transações
As transações esperadas são aquelas comuns a todas as contas:
• Depósito
• Saque
• Transferência (sujeita a cobrança)
• Pix (sem cobrança)
• Pagamentos de boletos
Quando a transação envolver duas contas, o resultado deve refletir nas duas. Por exemplo, uma transferência deve aparecer no extrato das duas, com seu respectivo histórico.

Saldos e Extratos
A opção de saldo deve apenas mostrar o saldo atual do cliente, enquanto a opção de extrato deve apresentar as últimas transações, levando em conta uma data de início válida.

~~**2. Funcionalidades complementares**~~
~~Estas funcionalidades visam enriquecer o projeto, mas não são considerados como implementações obrigatórias.~~

~~Aplicações Financeiras:~~
~~Havendo saldo suficiente na conta, este pode ser aplicado em determinado investimento. Neste caso, deverá haver uma lista de investimentos disponíveis.~~
~~O rendimento de cada aplicação é fixo (por exemplo, 1% ao mês).~~
~~Deverá haver um recurso que permite atualizar o rendimento com base no período aplicado.~~
~~Além disso, se a aplicação não permitir resgate antecipado, este deve ser bloqueado~~

~~pelo sistema.~~

User Stories
Login
User Story:
Como um cliente,
Quero realizar login no sistema usando o número da agência, número da conta e minha senha,
Para que eu possa acessar minha conta de forma segura.
Especificações:
• O sistema deve oferecer um campo para agência, conta e senha.
• Deve haver validação para garantir que os campos não estejam vazios.
• Para administradores, o login será feito com nome de usuário e senha.
• Senhas devem ser armazenadas usando hash seguro.

Critérios de Aceitação:
1. Usuários inválidos devem receber uma mensagem de erro: "Agência, conta ou senha incorretos."
2. O login deve ser bloqueado após 5 tentativas consecutivas incorretas, com aviso de "Conta
bloqueada. Entre em contato com o suporte."
3. O administrador pode visualizar logs de acesso no painel administrativo.
4. Autenticação via token JWT deve ser implementada para sessões seguras.

Criação e Manutenção de Contas
User Story:
Como um administrador,
Quero cadastrar novas contas bancárias no sistema,
Para que os clientes possam utilizar os serviços bancários da FinTECH.
Especificações:
• Deve haver um formulário de cadastro com campos: nome completo, CPF, endereço, telefone,
email, tipo de conta (simples ou especial), agência e senha inicial.
• O sistema deve gerar automaticamente o número da conta. (DEFINIR SE VAI SER NO BANCO OU NO BACK)
• Deve ser possível editar dados do cliente posteriormente (exceto CPF e número da conta).
Critérios de Aceitação:
1. Apenas administradores autenticados podem acessar a funcionalidade.
2. Ao criar uma conta, o administrador deve receber uma confirmação de sucesso.
3. Se algum campo obrigatório estiver ausente, o sistema deve exibir uma mensagem clara, por
exemplo, "O campo CPF é obrigatório."
4. Uma conta criada deve estar ativa por padrão, com saldo inicial igual a zero.

Transações
User Story: Depósito
Como um cliente,

Quero realizar depósitos em minha conta,
Para que meu saldo seja atualizado com o valor depositado.
Especificações:
• O cliente deve informar o valor a ser depositado.
• O depósito pode ser realizado por transferência entre contas.
Critérios de Aceitação:
1. O valor do depósito deve ser adicionado ao saldo da conta.
2. O extrato do cliente deve mostrar a transação com data, hora e valor.
3. Valores negativos ou não numéricos devem ser rejeitados.

User Story: Saque
Como um cliente,
Quero realizar saques em minha conta,
Para que eu possa retirar dinheiro de acordo com meu saldo.
Especificações:
• O cliente deve informar o valor do saque.
• O sistema deve verificar se há saldo suficiente antes de concluir a transação.
Critérios de Aceitação:
1. Se o saldo for insuficiente, o sistema deve exibir: "Saldo insuficiente para realizar o saque."
2. Após o saque, o saldo deve ser atualizado imediatamente.
3. O extrato deve registrar a transação com detalhes.

User Story: Transferência
Como um cliente,
Quero transferir valores para outra conta,
Para que eu possa pagar ou transferir dinheiro para outros clientes.
Especificações:
• O cliente deve informar a conta de destino e o valor.
• O sistema deve aplicar tarifas, quando aplicáveis, e mostrá-las antes da confirmação.
Critérios de Aceitação:
1. A transferência só será concluída se a conta de destino for válida e houver saldo suficiente.
2. Ambas as contas devem registrar a transação no extrato.
3. Uma mensagem de sucesso deve ser exibida após a conclusão.

User Story: Pix
Como um cliente,
Quero realizar transferências via Pix,
Para que eu possa transferir dinheiro para outras contas de forma rápida e sem cobrança.

Especificações:
• Deve ser possível cadastrar chaves Pix (CPF, email ou telefone).
• A transferência deve ser concluída em tempo real.
Critérios de Aceitação:
1. O cliente deve visualizar uma confirmação instantânea após a conclusão da transferência.
2. Transações via Pix devem aparecer no extrato como "Pix enviado para [Chave Pix]."

User Story: Pagamento de Boletos
Como um cliente,
Quero pagar boletos diretamente pela minha conta,
Para que eu possa quitar minhas obrigações financeiras de forma prática.
Especificações:
• O cliente deve informar o código de barras do boleto ou fazer upload do arquivo.
• O sistema deve calcular e mostrar o saldo final após o pagamento.
Critérios de Aceitação:
1. O boleto só será pago se o saldo for suficiente.
2. O extrato deve registrar a transação com a descrição do beneficiário.

Saldos e Extratos
User Story: Saldo
Como um cliente,
Quero consultar meu saldo atual,
Para saber quanto tenho disponível na minha conta.
Especificações:
• O saldo deve ser mostrado em tempo real na interface.
Critérios de Aceitação:
1. O cliente deve ver o saldo em formato monetário (R$ 0,00).
2. Caso não haja conexão com o servidor, o sistema deve exibir: "Não foi possível recuperar o
saldo. Tente novamente mais tarde."

User Story: Extrato
Como um cliente,
Quero consultar meu extrato bancário,
Para ver as últimas transações realizadas na minha conta.
Especificações:
• Deve ser possível filtrar o extrato por período (data inicial e final).
Critérios de Aceitação:
1. O extrato deve mostrar transações em ordem cronológica.
2. Cada entrada deve incluir: data, hora, descrição e valor (positivo ou negativo).

~~**Aplicações Financeiras**~~
~~**User Story:**~~
~~**Como** um cliente,~~
~~**Quero** aplicar meu saldo em investimentos,~~
~~**Para que** eu possa obter rendimentos sobre o valor aplicado.~~
~~**Especificações:**~~
~~• O cliente pode escolher entre investimentos com ou sem resgate antecipado.~~
~~• O sistema deve mostrar projeções de rendimento.~~
~~**Critérios de Aceitação:**~~
~~1. Aplicações bloqueadas não podem ser resgatadas antes do vencimento.~~
~~2. Rendimentos devem ser calculados corretamente com base no período aplicado.~~
~~3. O extrato deve registrar a aplicação e o resgate (quando aplicável).~~

Script SQL Server

```
--CRIAÇÃO TABELA TIPO DE USUARIO
CREATE TABLE Tipo_usuario(
ID_tipo_usuario INT PRIMARY KEY IDENTITY (1,1) NOT NULL,
tipo_usuario VARCHAR (255) NOT NULL);

INSERT INTO Tipo_usuario(tipo_usuario)
VALUES
('Cliente'),
('ADM'),
('Cliente'),
('Cliente'),
('Cliente')

SELECT *FROM Tipo_usuario
---------------------------------------------------------------------------------------------------------------
--CRIAÇÃO TABELA USUARIO
CREATE TABLE Usuario(
ID_usuario INT PRIMARY KEY IDENTITY (1,1) NOT NULL,
Nome VARCHAR(150) NOT NULL,
CPF VARCHAR(50) UNIQUE NOT NULL,
Telefone VARCHAR(50) UNIQUE NOT NULL,
Email VARCHAR (50) UNIQUE NOT NULL,
Data_cadastro DATETIME DEFAULT GETDATE(),
ID_tipo_usuario INT NOT NULL,
FOREIGN KEY (ID_tipo_usuario) REFERENCES Tipo_usuario (ID_tipo_usuario)
)

--Populando tabela Usuario
INSERT INTO Usuario (Nome, CPF, Telefone, Email, ID_tipo_usuario)
VALUES
('Carlos Silva', '123.456.789-00', '(11) 98765-4321', 'carlos.silva@yahoo.', 2), --- adm
('Ana Souza', '234.567.890-11', '(21) 98765-1234', 'ana.souza@outlook.com', 1), ---cliente
('Lucas Oliveira', '345.678.901-22', '(31) 99876-5432', 'lucas.oliveira@hotmail.com', 1),
---cliente
('Maria Costa', '456.789.012-33', '(41) 98765-8765', 'maria.costa@gmail.com', 1), ---cliente
('Maria da Paz Costa', '456.789.012-33', '(41) 98765-8765' , 'maria.costa@gmail.com',1)
---Verificação de duplicidade

SELECT
        u.ID_usuario,
        u.Nome AS Nome_Usuario,
        u.CPF,
        u.Email,
```

```
        u.Telefone,
        u.Data_cadastro,
        t.tipo_usuario AS Tipo_Usuario
FROM USUARIO u
INNER JOIN Tipo_Usuario t ON u.ID_tipo_usuario = t.ID_tipo_usuario;

SELECT*FROM Usuario
-------------------------------------------------------------------------------------------------------------
--Tabela: Endereços
CREATE TABLE Endereco(
ID_endereco INT PRIMARY KEY IDENTITY (1, 1) NOT NULL,
Rua VARCHAR (200) NOT NULL,
Numero INT,
Complemento VARCHAR (50),
CEP VARCHAR (9) NOT NULL,
Bairro VARCHAR (200),
Cidade VARCHAR (100),
Estado VARCHAR (2),
ID_usuario INT,
FOREIGN KEY (ID_usuario) REFERENCES Usuario (ID_usuario)
)

--Populando tabela Endereco

INSERT INTO Endereco (Rua, Numero, Complemento, CEP, Bairro, Cidade, Estado,
ID_usuario)
VALUES
('Rua das Flores', 123, 'Apto 101', '12345-678', 'Centro', 'São Paulo', 'SP', 6),
('Avenida Paulista', 1000, 'Sala 301', '98765-432', 'Bela Vista', 'São Paulo', 'SP', 7),
('Rua da Consolação', 200, 'Casa 2', '54321-123', 'Consolação', 'São Paulo', 'SP', 8),
('Rua do Riacho', 300, 'Bloco B', '12321-234', 'Itaim Bibi', 'São Paulo', 'SP', 9),
('Rua da Cachoeira', 876, 'Apto 123', '12321-234', 'Vila Olimpia', 'São Paulo', 'SP', 5) ---
Verificação de duplicidade

SELECT *FROM Endereco

SELECT
        e.ID_endereco,
        e.Rua,
        e.Numero,
        e.Complemento,
        e.CEP,
        e.Bairro,
        e.Cidade,
        e.Estado,
        u.Nome AS Nome_Usuario,
        t.tipo_usuario AS Tipo_Usuario
FROM Endereco e
```

```
INNER JOIN USUARIO u ON e.Id_usuario = u.Id_usuario
INNER JOIN Tipo_Usuario t ON u.ID_tipo_usuario = t.ID_tipo_usuario;
-----------------------------------------------------------------------------------------------------------
 --Tabela: CONTA
CREATE TABLE Conta (
ID_conta INT PRIMARY KEY IDENTITY(1,1) NOT NULL,
Conta INT NOT NULL UNIQUE,
Agencia INT NOT NULL,
Saldo DECIMAL (10, 2),
Data_abertura DATETIME,
Senha VARCHAR(200) NOT NULL,
ID_tipo_usuario INT NOT NULL,
FOREIGN KEY (ID_tipo_usuario) REFERENCES Tipo_usuario(ID_tipo_usuario)
);

--Populando Tabela Conta
INSERT INTO Conta (Conta, Agencia, Saldo, Data_abertura, Senha, ID_tipo_usuario)
VALUES
(123456, 101, 5000.00, GETDATE(), 'senha_adm_123', 2),
(234567, 102, 1500.00, GETDATE(), 'senha_cliente_234', 1),
(345678, 103, 2000.00, GETDATE(), 'senha_cliente_345', 1),
(456789, 104, 3000.00, GETDATE(), 'senha_cliente_456', 1),
(456789, 104, 567.00, GETDATE (), 'senha-cliente467',1) --- Verificação de duplicidade

SELECT* FROM Conta
-----------------------------------------------------------------------------------------------------------
-- Tabela Transações
CREATE TABLE Transacoes (
ID_transacao INT IDENTITY (1,1) PRIMARY KEY NOT NULL,
tipo_transacao VARCHAR (50) NOT NULL,
ID_conta_origem INT NOT NULL,
ID_conta_destino INT NOT NULL,
Valor FLOAT NOT NULL,
Data_hora DATETIME NOT NULL
FOREIGN KEY (ID_conta_origem) REFERENCES Conta(ID_conta),
FOREIGN KEY (ID_conta_destino) REFERENCES Conta(ID_conta)
);

INSERT INTO Transacoes (tipo_transacao, ID_conta_origem, ID_conta_destino, Valor,
Data_hora)
VALUES
('Transferência', 2, 3, 500.00, GETDATE()),
('Pix', 4, 2, 200.00, GETDATE()),
('Saque', 2, 4, 1000.00, GETDATE()),
('Deposito', 3, 4, 1500.00, GETDATE());


SELECT *FROM Transacoes
```

```
-----------------------------------------------------------------------------------------------------------------
 --Tabela Cadastro Pix
CREATE TABLE Cadastro_PIX (
ID_pix_cadastro INT PRIMARY KEY IDENTITY(1, 1) NOT NULL,
Chave_cadastrada_pix VARCHAR (200),
Chave_pix VARCHAR (200),
FOREIGN KEY (ID_pix_cadastro) REFERENCES Usuario(ID_usuario)
);

INSERT INTO Cadastro_PIX (Chave_cadastrada_pix, Chave_pix)
VALUES
('Email' , 'ana.souza@outlook.com'),
('Telefone' , '(31) 99876-5432'),
('Telefone' , '(41) 98765-8765'),
('Telefone', '(41) 98765-8765') --Verificação de duplicidade

SELECT *FROM Cadastro_PIX
SELECT *FROM Usuario
-----------------------------------------------------------------------------------------------------------------
--Tabela Saque
CREATE TABLE Saque (
ID_saque INT PRIMARY KEY IDENTITY(1,1) NOT NULL,
ID_conta INT NOT NULL,
tipo_transacao VARCHAR (100),
Taxa FLOAT,
ID_transacao INT NOT NULL,
FOREIGN KEY (ID_transacao) REFERENCES Transacoes(ID_transacao)
);

INSERT INTO Saque (ID_conta, tipo_transacao, Taxa, ID_transacao)
VALUES
(2, 'Saque Bancário', 2.50, 1),  -- saque da transação 1
(3, 'Saque Caixa Eletrônico', 3.00, 2),  -- saque da transação 2
(4, 'Saque Online', 1.50, 3) -- saque da transação 3

SELECT *FROM Saque
SELECT *FROM Conta
-----------------------------------------------------------------------------------------------------------------
--Tabela Deposito
CREATE TABLE Deposito (
ID_deposito INT PRIMARY KEY IDENTITY(1,1) NOT NULL,
ID_conta_origem INT NOT NULL,
ID_conta_destino INT NOT NULL,
Valor FLOAT,
Data_hora DATETIME DEFAULT GETDATE() NOT NULL,
ID_transacao INT NOT NULL,
FOREIGN KEY (ID_transacao) REFERENCES Transacoes(ID_transacao),
FOREIGN KEY (ID_conta_origem) REFERENCES Conta(ID_conta),
```

```
FOREIGN KEY (ID_conta_destino) REFERENCES Conta(ID_conta)
);

INSERT INTO Deposito (ID_conta_origem, ID_conta_destino, Valor, Data_hora,
ID_transacao)
VALUES
(1, 2, 1000.00, GETDATE(), 1),  -- transação 1
(2, 3, 1500.00, GETDATE(), 2),  -- transação 2
(3, 4, 2000.00, GETDATE(), 3)  -- transação 3

SELECT *FROM Deposito
-------------------------------------------------------------------------------------------------------------------
--
--Tabela Pix
CREATE TABLE Pix (
ID_pix INT PRIMARY KEY IDENTITY(1,1) NOT NULL,
ID_conta_origem INT NOT NULL,
ID_conta_destino INT NOT NULL,
Valor FLOAT,
Data_hora DATETIME DEFAULT GETDATE() NOT NULL,
ID_transacao INT NOT NULL,
FOREIGN KEY (ID_transacao) REFERENCES Transacoes(ID_transacao),
FOREIGN KEY (ID_conta_origem) REFERENCES Conta(ID_conta),
FOREIGN KEY (ID_conta_destino) REFERENCES Conta(ID_conta)
);

INSERT INTO Pix (ID_conta_origem, ID_conta_destino, Valor, Data_hora, ID_transacao)
VALUES
(2, 3, 500.00, GETDATE(), 1),  -- Pix da transação 1
(3, 2, 750.00, GETDATE(), 2),  -- Pix da transação 2
(2, 4, 1200.00, GETDATE(), 3),  -- Pix da transação 3
(4, 3, 1500.00, GETDATE(), 4)  -- Pix da transação 4

SELECT *FROM Pix
-------------------------------------------------------------------------------------------------------------------
-- Tabela extrato
CREATE TABLE Extrato(
ID_extrato INT PRIMARY KEY IDENTITY (1,1) NOT NULL,
Data_emissao DATETIME,
tipo_transacao VARCHAR (100),
ID_transacao INT NOT NULL,
FOREIGN KEY (ID_transacao) REFERENCES Transacoes (ID_transacao)
);

INSERT INTO Extrato (Data_emissao, tipo_transacao, ID_transacao)
VALUES
(GETDATE(), 'Transferência', 1),  -- extrato da transação 1
(GETDATE(), 'Pix', 2),  -- extrato da transação 2
```

```
(GETDATE(), 'Saque', 3)  -- extrato da transação 3

SELECT *FROM Extrato
```