**Prova QA**

1. Quais cenários de teste você especificaria para validar a funcionalidade do aplicativo bancário desenhado abaixo? Liste todas as premissas adotadas.

Texto

Descrição gerada automaticamente

**Regras de Validação:**

Campo **Cliente destino** aceita apenas clientes validos e cadastrados na base, em caso de o cliente seja valido, mas está bloqueado no sistema, deve emitir a seguinte mensagem: *“O cliente está bloqueado.”*

Campo **Valor** aceita somente oque estiver em **Saldo** na Conta.

Campo **Data de efetivação**, é possível criar em datas futuras, esta operação é chamada de **Agendamento**, datas **passadas** não é possível.

**Todos os campos são obrigatórios.**

**Mensagens:**

MSG01: Transferência Ocorreu com Sucesso!

MSG02: Transferência não realizada!

MSG03: Cliente destino é obrigatório.

MSG04: Valor é obrigatório.

MSG05: Data de efetivação é obrigatória.

MSG06: Cliente não encontrado.

MSG07: Cliente bloqueado.

MSG08: Valor é menor que o Saldo.

MSG09: Valor é maior que o Saldo.

MSG10: Você acaba de fazer um agendamento para a [Data Futura], certifique-se de haver Saldo na data especifica.

**Plano de Testes:**

Validação do **App de Banco**, em um Processo de **Transferência Bancária**

**CT001** – Validação com Sucesso da Transferência

Dado que estou com o Aplicativo aberto na tela de Transferência de valores entre correntistas.

E preencho o ‘Cliente destino’

E preencho o ‘Valor’

E seleciono uma ‘Data de efetivação’

Quando pressiono ‘Transferir’

Então o App deve informar que a “Transferência Ocorreu com Sucesso!”

**CT002** – Validação do Cancelamento

Dado que estou com o Aplicativo aberto na tela de Transferência de valores entre correntistas.

Quando pressiono ‘Cancelar’

Então o App deve emitir a seguinte mensagem: “Transferência não realizada!”

**CT003** – Validação de campos Obrigatórios – Cliente destino

Dado que estou com o Aplicativo aberto na tela de Transferência de valores entre correntistas.

E preencho o ‘Valor’

E seleciono uma ‘Data de efetivação’

Quando pressiono ‘Transferir’

Então o App informar a seguinte mensagem: “Cliente destino é obrigatório.”

**CT004** – Validação de campos Obrigatórios – Valor

Dado que estou com o Aplicativo aberto na tela de Transferência de valores entre correntistas.

E preencho o ‘Cliente destino’

E seleciono uma ‘Data de efetivação’

Quando pressiono ‘Transferir’

Então o App informar a seguinte mensagem: “Valor é obrigatório.”

**CT005** – Validação de campos Obrigatórios – Data de efetivação

Dado que estou com o Aplicativo aberto na tela de Transferência de valores entre correntistas.

E preencho o ‘Cliente destino’

E seleciono uma ‘Valor’

Quando pressiono ‘Transferir’

Então o App informar a seguinte mensagem: “Data de efetivação é obrigatória.”

**CT006** – Validação com Cliente destino invalido

Dado que estou com o Aplicativo aberto na tela de Transferência de valores entre correntistas.

E preencho o ‘Cliente destino’ com um ‘cliente invalido’

E preencho o ‘Valor’

E seleciono uma ‘Data de efetivação’

Quando pressiono ‘Transferir’

Então o App deve informar que a “Cliente não encontrado.”

**CT007** – Validação com Cliente destino Bloqueado

Dado que estou com o Aplicativo aberto na tela de Transferência de valores entre correntistas.

E preencho o ‘Cliente destino’ com um ‘cliente bloqueado’

E preencho o ‘Valor’

E seleciono uma ‘Data de efetivação’

Quando pressiono ‘Transferir’

Então o App deve informar que a “Cliente bloqueado.”

**CT008** – Validação com Valor Negativo

Dado que estou com o Aplicativo aberto na tela de Transferência de valores entre correntistas.

E preencho o ‘Cliente destino’

E preencho o ‘Valor’ com Valor igual a ‘-1’

E seleciono uma ‘Data de efetivação’

Quando pressiono ‘Transferir’

Então o App deve informar que a “Valor é menor que o Saldo.”

**CT009** – Validação com Valor Acima

Dado que estou com o Aplicativo aberto na tela de Transferência de valores entre correntistas.

E preencho o ‘Cliente destino’

E preencho o ‘Valor’ com Valor igual a ‘200,01’

E seleciono uma ‘Data de efetivação’

Quando pressiono ‘Transferir’

Então o App deve informar que a “Valor é maior que o Saldo.”

**CT010** – Validação com Datas Futuras

Dado que estou com o Aplicativo aberto na tela de Transferência de valores entre correntistas.

E preencho o ‘Cliente destino’

E preencho o ‘Valor’

E seleciono uma ‘Data de efetivação’ em uma ‘Data Futura’

Quando pressiono ‘Transferir’

Então o App deve informar que a “Você acaba de fazer um agendamento para a [Data Futura], certifique-se de haver Saldo na data especifica.”

**CT011** – Validação com Datas Futuras com Valores Maiores

Dado que estou com o Aplicativo aberto na tela de Transferência de valores entre correntistas.

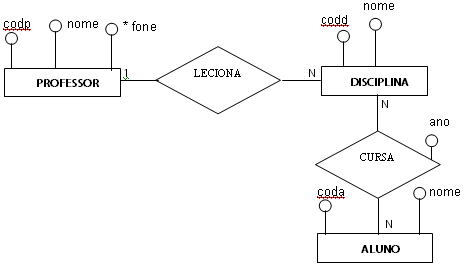
E preencho o ‘Cliente destino’

E preencho o ‘Valor’ com o Valor maior que o Saldo

E seleciono uma ‘Data de efetivação’ em uma ‘Data Futura’

Quando pressiono ‘Transferir’

Então o App deve informar que a “Você acaba de fazer um agendamento para a [Data Futura], certifique-se de haver Saldo na data especifica.”

1. Considere o esquema Entidade / Relacionamento abaixo:

* Escreva a consulta SQL para listar o nome de todos os alunos matriculados na disciplina de Cálculo do professor João.

SELECT

Alunos.nome AS Aluno,

Disciplina.nome AS Disciplina

Professor.name AS Professor

FROM Alunos

JOIN Disciplina ON Alunos.codd = Disciplina.codd

JOIN Professor ON Disciplina.codp = Professor.codp

WHERE Disciplina.nome = 'Cálculo' AND Professor.nome = 'João'

* Escreva a consulta SQL para exibir a quantidade de alunos por disciplinas.

SELECT Disciplina.nome AS Disciplina,

COUNT(Alunos.coda) AS QuantidadeAlunos

FROM Alunos JOIN Disciplina ON Alunos.codd = Disciplina.codd

GROUP BY Disciplina.nome

* Escreva a consulta SQL para listar as disciplinas que todos os professores lecionam.

SELECT Disciplina.nome

FROM Disciplina

GROUP BY Disciplina.nome

HAVING COUNT(DISTINCT Disciplina.codp) = (SELECT COUNT(\*)

FROM Professor)

* Escreva a consulta SQL que exibe o total de professores.

SELECT COUNT(\*) AS TotalProfessores

FROM Professor

* Escreva a consulta SQL para listar todos os alunos que cursaram alguma disciplina do ano 2000 até 2020.

SELECT Alunos.nome   
 FROM Alunos JOIN Disciplina ON Alunos.codd = Disciplina.codd

WHERE Disciplina.ano BETWEEN 2000 AND 2020

1. **Para validarmos a sua técnica de automação de testes, vamos utilizar uma api de desenvolvimento referente a plataforma Trello.**

Caso não conheça o Trello, acessar: [www.trello.com](http://www.trello.com)

Link da api para desenvolvimento: <https://developers.trello.com/docs/api-introduction>

Com o end-point acima, crie testes automatizados para as seguintes ações:

* **Cadastrar um board**
* **Cadastrar um card**
* **Excluir um card**
* **Excluir um board**

**Obs: Utilizar o Cypress como ferramenta de automação.**

**Para a automação de testes realizada, ilustre através de um mapa mental, a sua estratégia de testes para construção do seu projeto.**

**Mapa Mental:**

**Trello API Automation**

**Setup**

* + **baseUrl: https://api.trello.com/1**
  + **API Authentication: Utilização das chaves key e token via Cypress environment variables.**

1. **Cadastrar um Board**
   * **Request Method: POST**
   * **URL: ${baseUrl}/boards/**
   * **Query String Parameters:**
     + **name: 'Board de Teste'**
     + **key: Cypress.env('TRELLO\_KEY')**
     + **token: Cypress.env('TRELLO\_TOKEN')**
   * **Response Handling:**
     + **Status Check: Verifica se o status da resposta é 200.**
     + **Save Board ID: Armazena o ID do board para uso posterior.**
2. **Cadastrar um Card**
   * **Request Method: POST**
   * **URL: ${baseUrl}/cards**
   * **Query String Parameters:**
     + **name: 'Card de Teste'**
     + **idList: this.boardId (utiliza o ID do board criado)**
     + **key: Cypress.env('TRELLO\_KEY')**
     + **token: Cypress.env('TRELLO\_TOKEN')**
   * **Response Handling:**
     + **Status Check: Verifica se o status da resposta é 200.**
     + **Save Card ID: Armazena o ID do card para uso posterior.**
3. **Excluir um Card**
   * **Request Method: DELETE**
   * **URL: ${baseUrl}/cards/${this.cardId}**
   * **Query String Parameters:**
     + **key: Cypress.env('TRELLO\_KEY')**
     + **token: Cypress.env('TRELLO\_TOKEN')**
   * **Response Handling:**
     + **Status Check: Verifica se o status da resposta é 200.**
4. **Excluir um Board**
   * **Request Method: DELETE**
   * **URL: ${baseUrl}/boards/${this.boardId}**
   * **Query String Parameters:**
     + **key: Cypress.env('TRELLO\_KEY')**
     + **token: Cypress.env('TRELLO\_TOKEN')**
   * **Response Handling:**
     + **Status Check: Verifica se o status da resposta é 200.**

**Após a conclusão do teste, publique no github, todo material produzido, e não esqueça de nos disponibilizar o link da sua avaliação.**

<https://github.com/Migzinho/verx/>