UNIFG

Simulador de Banco em Python

Danilo Dayvid Lucas

2023.2

UNIFG

Objetivo do Projeto

- Criar um simulador de banco básico.
- Demonstrar operações bancárias como criar conta, acessar conta, depositar e sacar.

2023.2

UNIFG

Tecnologias Utilizadas

- Linguagem de Programação: Python
- Ferramentas: IDE (PyCharm)

2023.2

UNIFG

Requisitos Funcionais do Negócio

Acesso à Conta

Os usuários devem poder acessar uma conta existente usando o número da conta, permitindo a realização de operações.

Consultar Saldo

Os usuários devem poder verificar o saldo atual de sua conta.

Realizar Depósito

Os usuários devem poder fazer depósitos em suas contas, aumentando o saldo disponível.

Realizar Saque

Os usuários devem poder efetuar saques de suas contas, reduzindo o saldo disponível.

2023.2

Requisitos Não Funcionais do Negócio

Desempenho

As operações do simulador devem ser executadas de maneira eficiente, proporcionando uma resposta rápida ao usuário.

Usabilidade

A interface do usuário deve ser amigável e intuitiva, permitindo que usuários sem conhecimento técnico realizem operações facilmente.

Segurança

As transações financeiras devem ser seguras, protegendo as informações sensíveis dos clientes.

2023.2

UNIFG

Requisitos Não Funcionais do Negócio

Escalabilidade

O sistema deve ser capaz de lidar com um aumento razoável no número de contas e transações sem comprometer significativamente o desempenho.

Testabilidade

O sistema deve ser projetado de maneira que seja fácil de testar, facilitando a criação e execução de testes para garantir a qualidade do software.

2023.2

UNIFG

Requisitos Funcionais do Usuário

Cadastro de conta

O usuário deve poder se cadastrar no sistema, fornecendo informações como nome e saldo inicial para criar uma nova conta bancária.

Login

sistema deve permitir que o usuário faça login com suas credenciais para acessar sua conta.

Consultar Saldo

Os usuários devem poder verificar o saldo atual de sua conta após o login.

Realizar Operações Bancárias

O usuário deve poder realizar operações bancárias, incluindo depósitos e saques, para gerenciar o saldo de sua conta.

2023.2

Requisitos Não Funcionais do Usuário

Disponibiliidade

A interface do usuário deve ser intuitiva e fácil de usar, permitindo que usuários de diferentes níveis de habilidade compreendam e executem as operações facilmente.

Tempo de Resposta

O sistema deve fornecer respostas rápidas às interações do usuário, garantindo uma experiência eficiente.

Facilidade de Aprendizado

O sistema deve ser projetado de maneira a facilitar a aprendizagem para novos usuários, proporcionando uma curva de aprendizado suave.

2023.2

UNIFG

Requisitos Funcionais do Sistema

Persistência de Dados

As informações das contas, incluindo detalhes do cliente e saldos, devem ser persistentes entre as sessões do programa.

Segurança das Transações

O sistema deve garantir a segurança das transações bancárias, protegendo as informações sensíveis dos usuários.

Relatório de Transações

O sistema deve ser capaz de gerar relatórios de transações, fornecendo um histórico detalhado das atividades na conta.

2023.2

Requisitos Não Funcionais do Sistema

Desempenho

O sistema deve ser responsivo e eficiente, proporcionando tempos de resposta rápidos durante as operações.

Segurança da Aplicação

O sistema deve incorporar práticas de segurança robustas para proteger contra ameaças como injeção de SQL, ataques de força bruta, etc.

Disponibilidade

O sistema deve ter uma alta disponibilidade, minimizando períodos de inatividade não planejados e garantindo acesso constante aos usuários.

Manutenabilidade

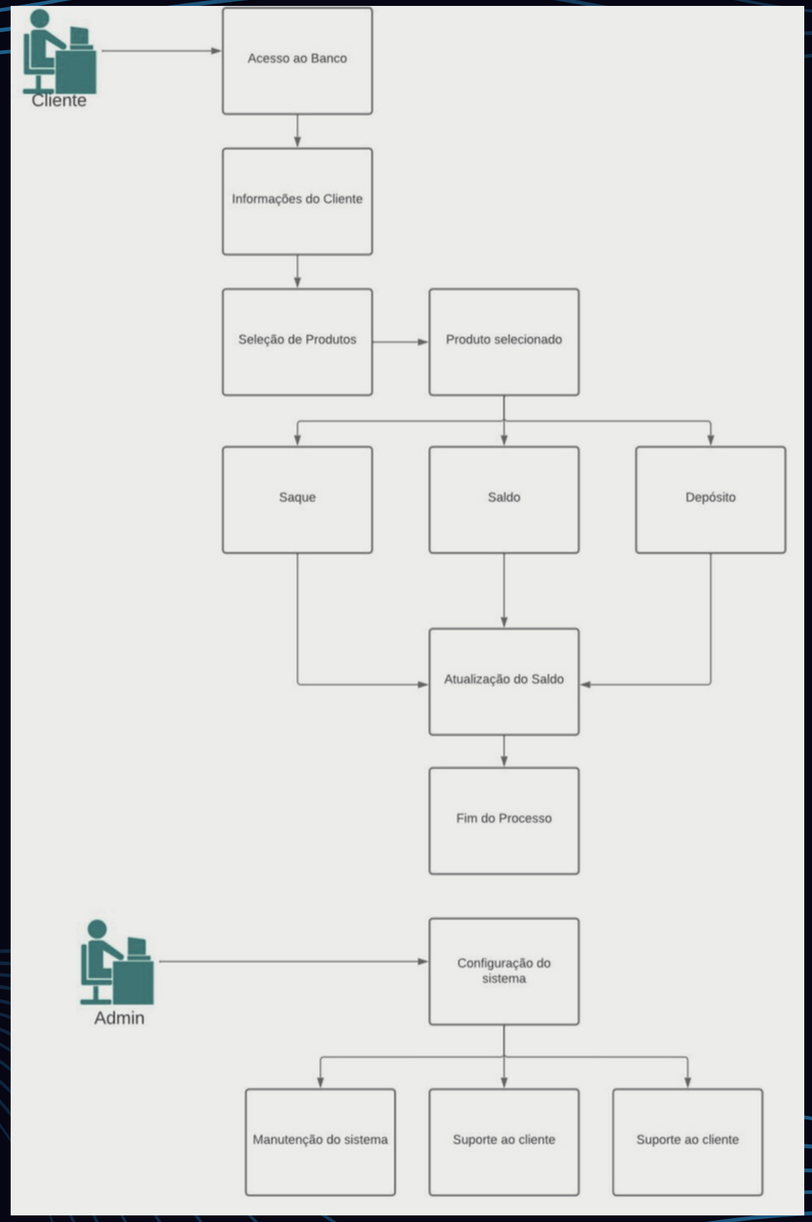
O código-fonte deve ser modular e bem documentado, facilitando a manutenção e futuras atualizações.

UNIFG

Plano de Teste

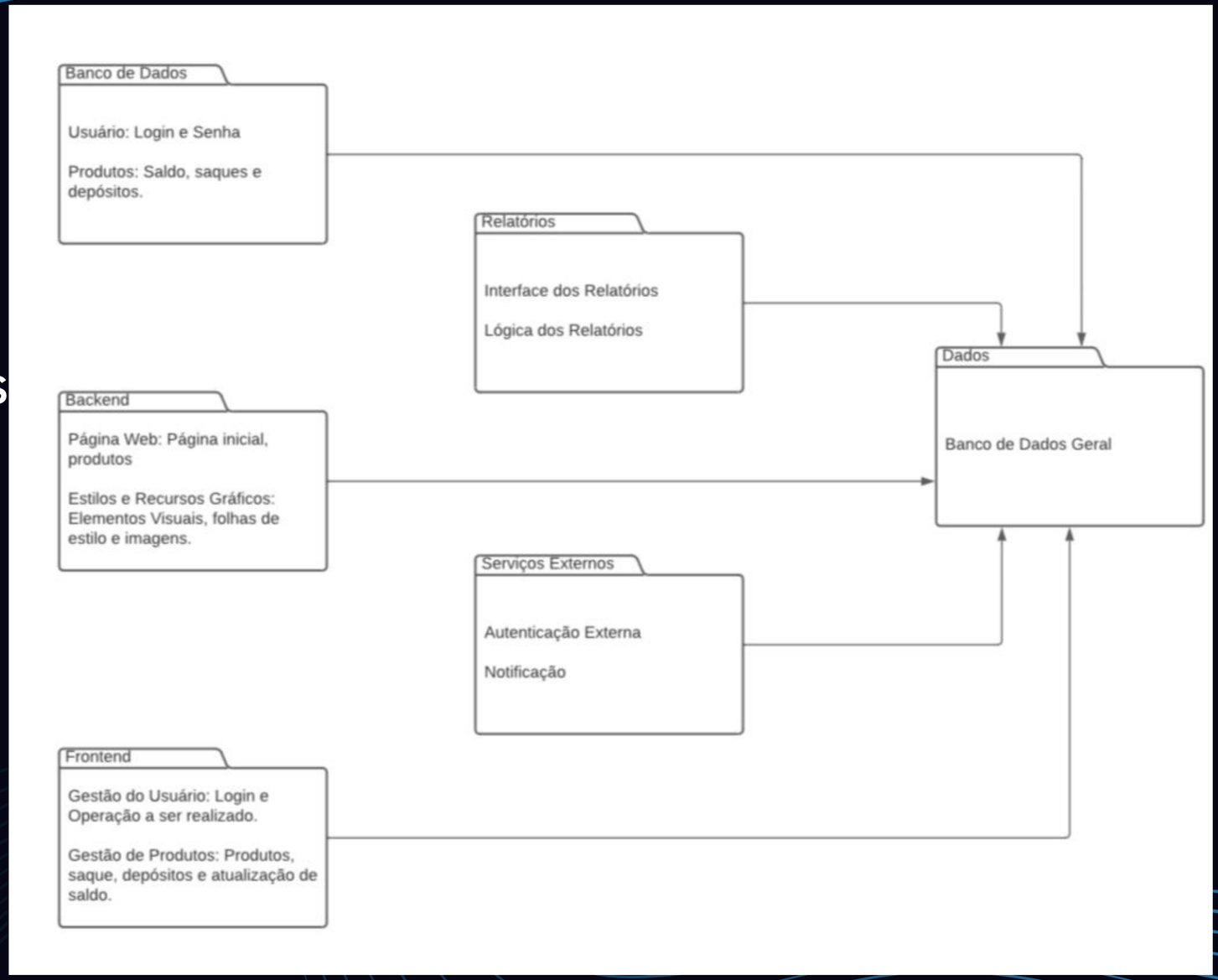
- **Cadastro de Conta**
  - Criar uma nova conta com dados válidos
  - Cadastro com informações válidas
  - Garantir a unicidade de cada conta nova
- **Acesso à Conta**
  - Login com credencias corretas
  - Bloqueiro de acesso com credenciais incorretas
- **Realizar Saque**
  - Saque com saldo suficiente
  - Saque com saldo insuficiente
  - Saque com valores inválidos
- **Realizar Depósito**
  - Depósito com valor válido
  - Depósito com valor negativos ou inválidos
  - Atualização do saldo após despósito

Diagrama de Caso de Uso



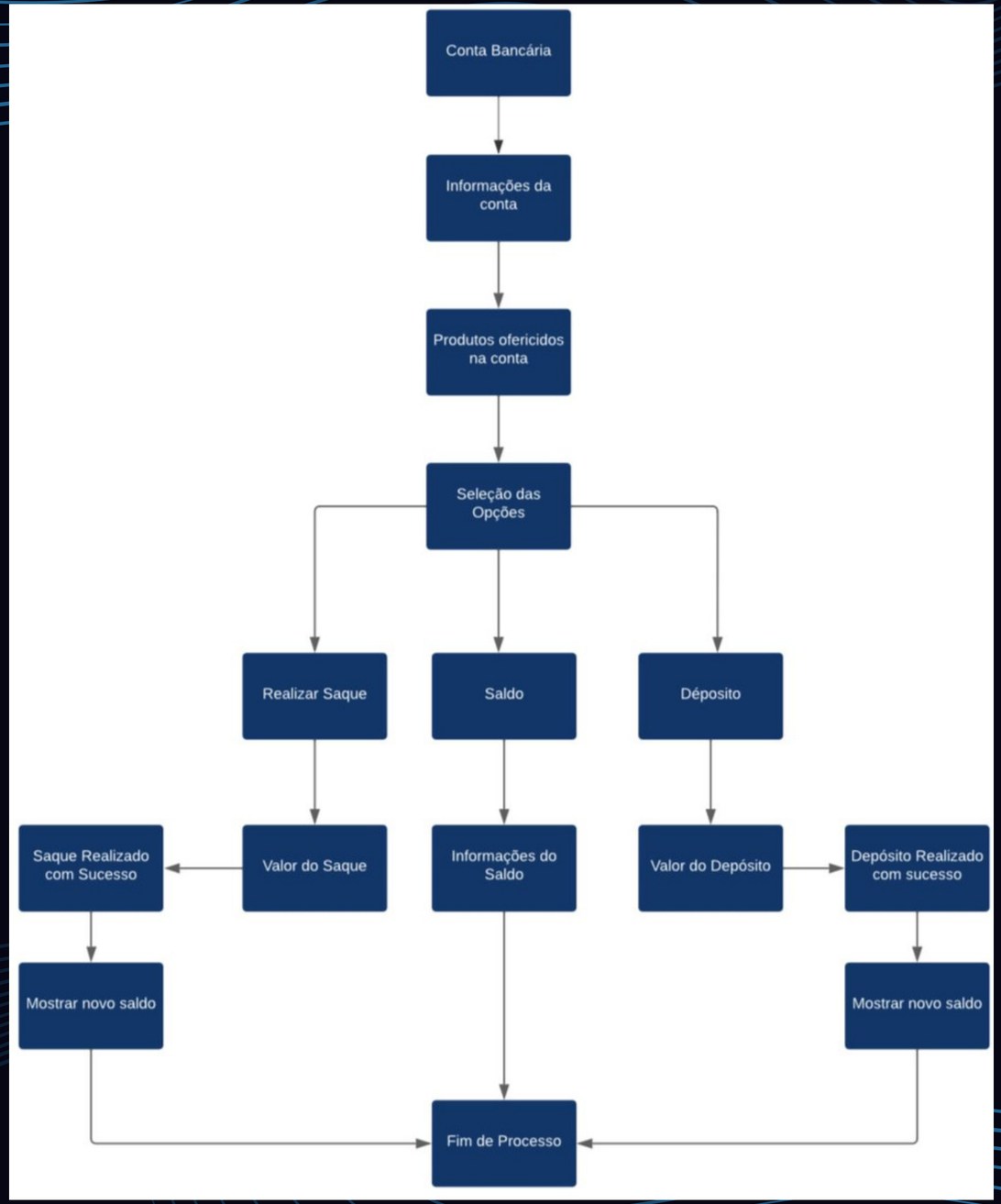
2023.2

Diagrama de Pacotes



UNIFG

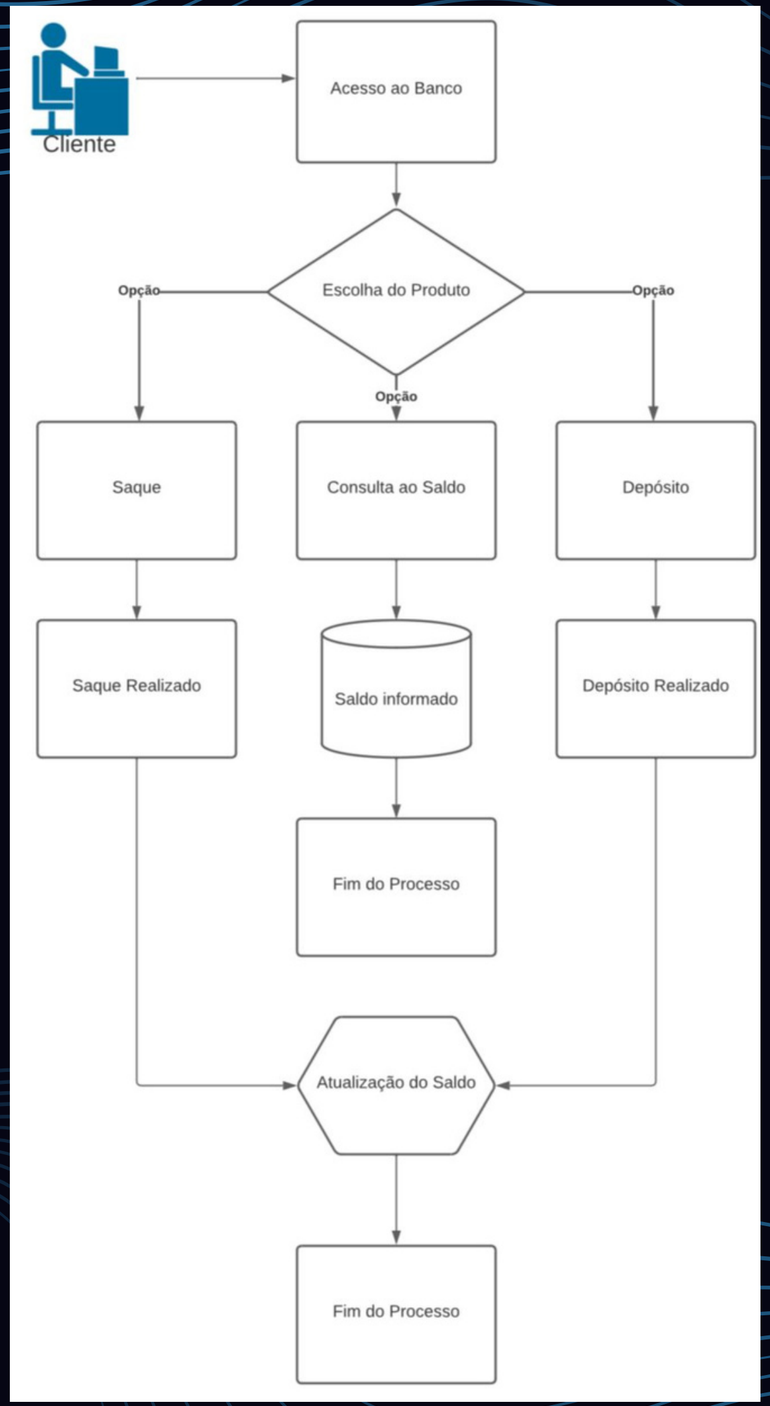
Diagrama Hierárquico



2023.2

UNIFG

Diagrama de Pacotes



2023.2

UNIFG

Conclusão

Em resumo, a criação do simulador de banco em Python ofereceu uma introdução prática ao desenvolvimento de software nessa linguagem. Ao estruturar classes e implementar operações bancárias básicas, ganhamos uma compreensão valiosa de como construir sistemas simples em Python. Ao longo do projeto, focamos na clareza do código, na eficiência das operações e na interação amigável com o usuário. A simplicidade da linguagem Python tornou a implementação ágil e acessível, permitindo uma rápida iteração e teste das funcionalidades.