Relatório de Desenvolvimento de Sistemas Backend - Fase 2

Descrição da Arquitetura

O projeto foi desenvolvido com uma arquitetura baseada em microsserviços, utilizando o api-gateway como ponto de entrada unificado e os serviços ServicoGestao, ServicoFaturamento e ServicoPlanosAtivos. A arquitetura segue os princípios da Clean Architecture adaptados para microsserviços, com as seguintes camadas:

- - Domínio: Contém as entidades e interfaces que definem os modelos de negócios e contratos de acesso aos dados.
- - Aplicação: Inclui os use cases e serviços que implementam a lógica de negócios.
- - Interface: Representada pelos controllers que expõem os endpoints da API.
- - Infraestrutura: Implementa os repositórios concretos e a integração com RabbitMQ.

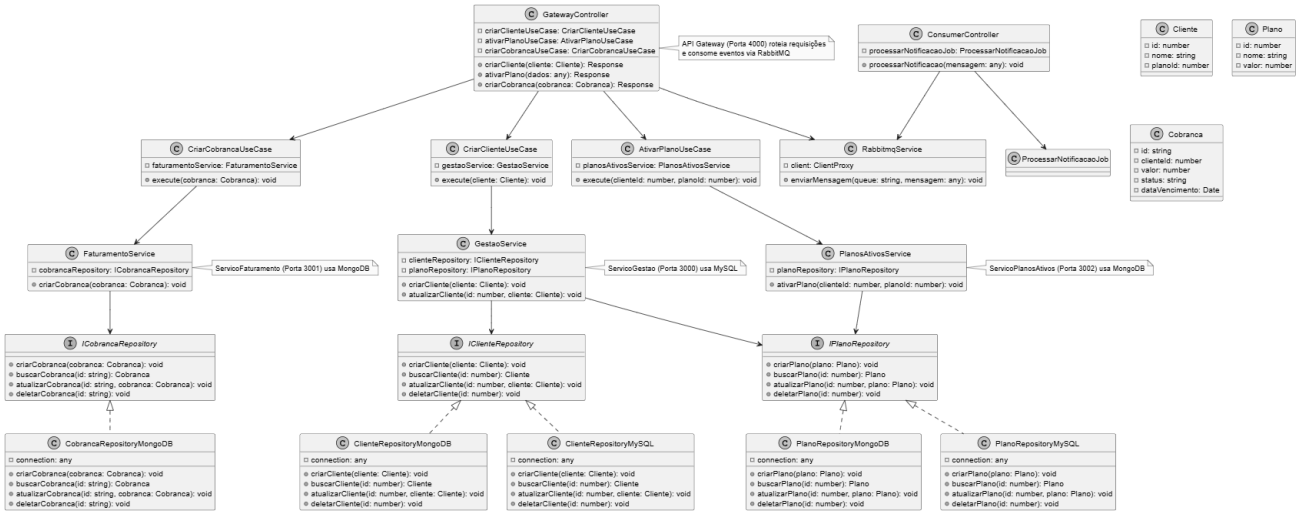
Essa separação permite escalabilidade e troca de tecnologias sem impactar a lógica de negócios.

Princípios SOLID Aplicados

- - S (Single Responsibility Principle): Cada classe tem uma única responsabilidade.
- - O (Open/Closed Principle): As interfaces permitem adicionar novas implementações sem modificar o código existente.
- - L (Liskov Substitution Principle): Repositórios substituem suas interfaces sem alterar o comportamento.
- - I (Interface Segregation Principle): As interfaces são específicas, contendo apenas os métodos necessários.
- - D (Dependency Inversion Principle): Os use cases dependem de abstrações.

Diagrama UML

O diagrama UML ilustra as relações entre as classes e interfaces do projeto.



Desafios Enfrentados

- - Erros de Conexão com RabbitMQ: Corrigido ajustando noAck para true e simplificando configuração.
- - Tipagem com Banco de Dados: Corrigido uso de AxiosResponse com RxJS.
- - Configuração Híbrida HTTP/RabbitMQ: Adicionada integração com fallback HTTP.

Manual de Execução

Pré-requisitos:

- - Node.js (versão 18.x ou superior)
- - MySQL
- - MongoDB
- - RabbitMQ

Instale as Dependências:

Navegue até cada pasta e execute: npm install.

Configuração (.env):

Cada pasta contém um arquivo .env.example com as configurações padrão. Renomeie ou crie um arquivo .env em cada pasta, mantendo as URLs fornecidas e incluindo suas credenciais (usuário e senha) conforme necessário.

Execução dos Serviços:

- cd api-gateway -> npm start
- cd servico-gestao -> npm start
- cd servico-faturamento -> npm start
- cd servico-planos-ativos -> npm start

Testar Endpoints com Postman:

- Importe a coleção Brenda_Desenvolvimento_de_Sistemas_backend_Fase-2.postman_collection.json

Estrutura do Projeto

- - api-gateway/: Gateway que roteia requisições e consome eventos.
- - servico-gestao/: Gerencia clientes e planos com MySQL.
- - servico-faturamento/: Gerencia cobranças com MongoDB.
- - servico-planos-ativos/: Gerencia planos ativos com MongoDB.
- - Brenda_Desenvolvimento_de_Sistemas_backend_Fase-2.postman_collection.json

Conclusão

A Fase 2 implementa uma arquitetura de microsserviços robusta, com integração via RabbitMQ e fallback HTTP. Os desafios foram superados com ajustes na configuração e tipagem, resultando em um sistema escalável e testável.