

Desenvolvimento de Sistemas Backend

1. Introdução

Com a crescente demanda por serviços de internet, operadoras necessitam de soluções eficientes para gerenciar os planos e assinaturas de seus clientes. Este projeto propõe o desenvolvimento de um sistema backend para operadoras de internet, utilizando a Arquitetura Limpa proposta por Robert Martin. O sistema visa manter o cadastro de clientes e planos, gerenciar assinaturas e garantir integração com sistemas externos.

2. Arquitetura do Sistema

O sistema é composto por um serviço principal, denominado **ServicoGestao**, e um microsserviço auxiliar: **ServicoFaturamento**.

2.1 ServicoGestao

Responsável pela gestão de clientes, planos e assinaturas, além de processar a cobrança e atualizar o status de assinaturas.

2.2 ServicoFaturamento

Registra os pagamentos efetuados e armazena os históricos de pagamentos realizados pelos clientes.

3. Aplicando a Arquitetura Limpa e Princípios SOLID

A implementação segue os seguintes princípios:

- Single Responsibility Principle (SRP): Cada serviço tem uma responsabilidade bem definida.
- **Open/Closed Principle (OCP)**: Novas funcionalidades podem ser adicionadas sem modificar classes existentes.
- **Liskov Substitution Principle (LSP)**: As classes derivadas podem substituir as classes base sem impactar o sistema.
- **Interface Segregation Principle (ISP)**: Interfaces são criadas para atender apenas os serviços necessários.
- **Dependency Inversion Principle (DIP)**: As dependências são injetadas para manter baixo acoplamento.

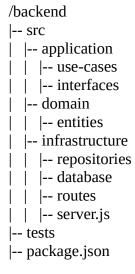
4. Padrões de Projeto Utilizados

Foram adotados diversos padrões de projeto para garantir a escalabilidade e manutenção do sistema:

- Repository Pattern: Para acesso aos dados e separação da lógica de negócio.
- Factory Pattern: Para criação de instâncias de serviços.
- Service Layer Pattern: Adotado para dividir as regras de negócio e os controladores HTTP, garantindo que as rotas tenham apenas a responsabilidade de receber requisições e devolver respostas.

5. Implementação do Sistema

A implementação foi realizada utilizando Node.js com Express, seguindo a Arquitetura Limpa. A estrutura do projeto foi organizada da seguinte forma:



Os endpoints foram implementados conforme especificado no template do Postman. As principais tecnologias utilizadas incluem:

- Node.js
- Express
- Sequelize (ORM para banco de dados SQL)
- MySQL (banco de dados utilizado)

6. Desafios e Soluções

Durante o desenvolvimento, alguns desafios foram enfrentados:

 Gerenciamento de Assinaturas: Foi implementado um sistema de controle de fidelidade e expiração de planos.

- **Performance**: A utilização de um banco de dados relacional otimizado garantiu respostas rápidas para as consultas de assinaturas.
- Segurança: Foram implementadas validações nos endpoints para garantir a integridade dos dados enviados e recebidos.

7. Conclusão

O desenvolvimento do **ServicoGestao** e **ServicoFaturamento** seguiu as melhores práticas de engenharia de software, aplicando Arquitetura Limpa e princípios SOLID. A divisão do sistema em serviços garante escalabilidade e integração eficiente, tornando-o uma solução robusta para operadoras de internet.

Foram feitas melhorias significativas na estrutura do código, garantindo que ele esteja modularizado e fácil de manter. A implementação dos testes automatizados e a separação das camadas do sistema aumentaram a confiabilidade da aplicação.

Próximos passos incluem testes de integração, implementação de autenticação e documentação adicional para deploy.

8. Referências

- Arquitetura Limpa (Clean Architecture) Robert C. Martin: https://blog.cleancoder.com/uncle-bob/2012/08/13/the-clean-architecture.html
- **Princípios SOLID Explicação e Exemplos**: https://www.digitalocean.com/community/tutorials/solid-principles-object-oriented-design
- **Sequelize ORM Documentação Oficial**: https://sequelize.org/docs/v6/
- Express.js Guia Oficial: https://expressjs.com/
- Node.js Documentação Oficial: https://nodejs.org/en/docs/
- **Padrões de Projeto em JavaScript**: https://refactoring.guru/pt-br/design-patterns