Relatório Técnico – Módulo de Empréstimo e Devolução

1. Introdução

Este documento apresenta uma descrição técnica do desenvolvimento do módulo de **Empréstimo e Devolução** do sistema **Biblioteca Tabajara**, implementado em linguagem Python. O módulo integra-se aos componentes de **Usuários** e **Catálogo**, viabilizando a gestão completa de empréstimos de livros, controle de prazos e devoluções, com base em regras de negócio pré-definidas. O trabalho segue a metodologia de **Desenvolvimento Orientado a Testes (TDD)** e aplica testes automatizados de unidade, contrato e integração, conforme as boas práticas da Engenharia de Software.

2. Objetivos

O principal objetivo do projeto consiste em projetar e validar, através de testes automatizados, o módulo responsável pelo controle de empréstimos e devoluções de exemplares. Busca-se garantir o correto funcionamento das regras de negócio associadas, entre elas: Limite máximo de empréstimos simultâneos (3 para alunos e 5 para professores); Definição de prazos distintos de devolução (14 dias para alunos e 28 dias para professores); Impedimento de empréstimo de exemplares indisponíveis; Bloqueio de duplicidade de empréstimos ativos para o mesmo usuário e livro; Incremento do estoque no momento da devolução. A validação do sistema foi realizada por meio de testes unitários, de contrato e de integração, assegurando cobertura total dos fluxos principais e das exceções previstas.

3. Metodologia de Desenvolvimento (TDD)

O desenvolvimento do módulo foi conduzido seguindo a metodologia **Test Driven Development (TDD)**, baseada nos ciclos sequenciais *Red* $\rightarrow$ *Green* $\rightarrow$ *Refactor*. Em cada ciclo, o grupo elaborou primeiramente um teste que necessariamente falha (*Red*), em seguida implementou a menor porção de código capaz de fazê-lo passar (*Green*), e, por fim, realizou melhorias de estrutura e legibilidade (*Refactor*). Esse processo permitiu a construção incremental e controlada do sistema, garantindo alta coesão entre as classes e funções e reduzindo o risco de regressões durante o desenvolvimento.

4. Estrutura do Projeto

O projeto foi estruturado em camadas lógicas, visando modularidade e separação de responsabilidades: app/ models.py # Entidades principais (User, Book, Loan) repositories.py # Repositórios em memória simulando os módulos externos services/ loan_service.py # Regras de negócio de empréstimo e devolução tests/ unit/ # Testes unitários (10 casos) contract/ # Testes de contrato e validação (5 casos) integration/ # Testes de integração (5 casos) Essa organização promove independência entre as partes do sistema e facilita a realização de testes automatizados,

seguindo o princípio da *Injeção de Dependência*.

5. Testes Automatizados e Validação do Sistema

Foram implementadas três categorias de testes: **Testes Unitários:** validam individualmente as regras de negócio e comportamentos das funções de empréstimo e devolução. **Testes de Contrato:** verificam a integridade das estruturas de dados e a consistência das funções de serialização e desserialização. **Testes de Integração:** simulam o fluxo completo entre os módulos de Usuários, Catálogo e Empréstimos. Todos os testes foram escritos utilizando o framework **Pytest**, garantindo automação e rastreabilidade completa dos resultados. Os testes cobrem cenários de sucesso, falhas esperadas e condições de borda, assegurando a conformidade com os requisitos do trabalho.

6. Resultados e Análise

A execução da suíte de testes evidenciou a robustez do módulo implementado. O sistema respeitou integralmente as restrições definidas nas regras de negócio e apresentou comportamento consistente em todas as operações críticas. O relatório de execução do Pytest demonstrou a passagem de 100% dos casos de teste, sem falhas ou regressões detectadas.