T2 - Programação Modular

Grupo 6 - 3WB

PAULO HENRIQUE LOPES MONÇÃO - 2310736
PEDRO ANTERO SIMAS DE OLIVEIRA - 2310970
PEDRO MESQUITA MAIA - 2312664
LUANA NOBRE DA COSTA – 2310204

Objetivo da aplicação

A aplicação tem como objetivo gerenciar uma biblioteca, oferecendo funcionalidades que facilitam o controle de empréstimos de livros, a administração de usuários, o gerenciamento de multas e a geração de relatórios. O sistema permite que os livros do acervo sejam criados, editados, excluídos e listados de forma eficiente. Além disso, a aplicação gerencia todo o processo de empréstimo de livros, registrando a data de devolução prevista e permitindo a finalização dos empréstimos, associando sempre o livro e o usuário a cada operação.

Especificação de requisitos

Requisitos Funcionais:

1.1. Módulo de Livros

- O sistema deve permitir o cadastro de novos livros com informações como título, autor, ano, número de exemplares, e editora.
- O sistema deve possibilitar a consulta de livros por diferentes critérios, como título, autor ou ano.
- O sistema deve permitir a atualização das informações dos livros, como título, autor, ou disponibilidade.
- O sistema deve permitir a remoção de livros do catálogo da biblioteca.
- O sistema deve registrar a disponibilidade de exemplares de cada livro no catálogo.

1.2. Módulo de Usuários

- O sistema deve permitir o cadastro de novos usuários, incluindo dados como nome, endereço, telefone e e-mail
- O sistema deve permitir a consulta de usuários por meio de identificação como nome, telefone ou e-mail.
- O sistema deve permitir a atualização de informações do usuário, como endereço, e-mail e telefone.
- O sistema deve permitir a desativação de usuários, bloqueando-os de realizar novos empréstimos.

1.3. Módulo de Empréstimos

- O sistema deve permitir o registro de empréstimos de livros por usuários cadastrados, verificando a disponibilidade de exemplares.
- O sistema deve permitir o registro de devoluções de livros, atualizando automaticamente a disponibilidade dos livros no catálogo.
- O sistema deve verificar se o usuário possui multas ou bloqueios antes de permitir novos empréstimos.
- O sistema deve permitir que usuários façam o agendamento de empréstimos, caso o livro esteja indisponível no momento.

1.4. Módulo de Multas

- O sistema deve calcular automaticamente a multa em caso de atraso na devolução de livros, considerando os dias de atraso.
- O sistema deve permitir o registro de quitação de multas pelos usuários.
- Além das multas por atraso, o sistema deve permitir registrar multas específicas por danos ou perda de livros.

Requisitos Não Funcionais:

2.1 Desempenho

- O sistema deve garantir uma resposta fluida nas operações mais comuns, como busca de livros e consulta de status de empréstimos, de forma que o usuário não perceba lentidão.
- Deve ser capaz de lidar com um grande número de transações e consultas sem comprometer significativamente a experiência do usuário.

2.2 Escalabilidade

- O sistema deve permitir expansão gradual em termos de número de usuários e quantidade de dados, sem necessidade de reestruturação significativa da arquitetura.
- Deve ser projetado para acomodar novos módulos e funcionalidades conforme o crescimento da biblioteca ou mudanças nas demandas dos usuários.

2.3 Manutenção

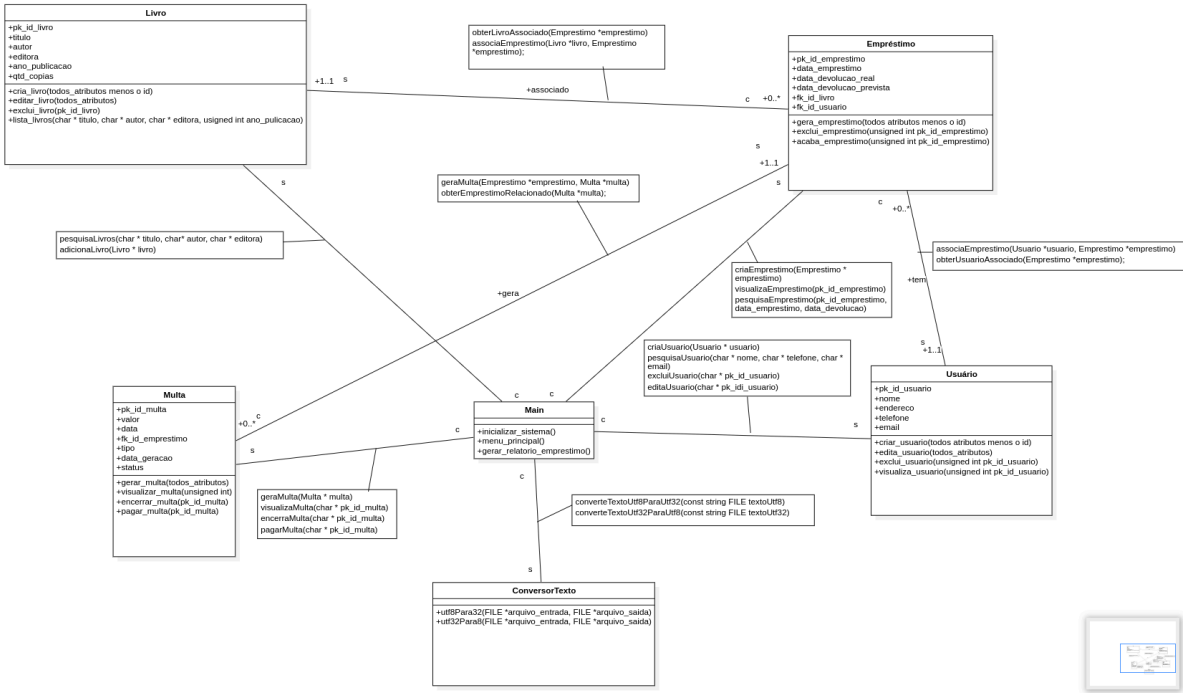
- O código e a arquitetura do sistema devem ser projetados para facilitar futuras manutenções e atualizações, com modularidade e documentação adequada.
- O sistema deve ser fácil de atualizar com novas versões ou funcionalidades, sem necessidade de interrupção prolongada no serviço.

2.4 Segurança

- O sistema deve garantir a segurança dos dados sensíveis, protegendo informações pessoais e transações contra acessos não autorizados, por meio de mecanismos como autenticação e controle de permissões.

- Deve seguir boas práticas de segurança, incluindo o uso de políticas de backup e recuperação de dados para proteger contra perdas.

Diagrama de arquitetura de módulos



Definição do desenvolvedor responsável por cada módulo

Módulos e desenvolvedor responsável:

- Livro - Pedro Simas
- Empréstimo - Pedro Mesquita
- Multa - Pedro Mesquita
- Usuário - Luana Nobre
- Main - Paulo Henrique
- ConversorTexto - Paulo Henrique

Programa testador

Empréstimo:
Void testaEmprestimo(char * descricao, int resultadoEsperado, int resultadoObtido)

Ex:
resultado = gera_emprestimo("", "usuario123");
testa("Geração de empréstimo com livro ausente", 1, resultado);

Multa:
Void testaEmprestimo(char * descricao, int resultadoEsperado, int resultadoObtido)

Ex:
resultado = gera_multa("emprestimo456", 50);
testa("Falha no sistema ao gerar multa", 5, resultado);

Casos de teste

0: Sucesso.
1: Erro de parâmetro inválido.
2: Erro de item não encontrado.
3: Erro de operação inválida (item já existente).
4: Erro de dependência ou restrição (tentativa de excluir livro com empréstimo ativo).
5: Erro desconhecido.

Programa de teste automatizado de cada módulo

Livro

cria_livro(char * nome, char * autor, char * editora, unsigned int ano_publicacao, unsigned int qtd_copias)
(1) Título ou autor ausente (parâmetro inválido).
(3) Tentativa de criar um livro com ID (isbn) já existente (livro duplicado).
(5) Falha ao acessar banco de dados (erro de sistema).

edita_livro(unsigned int pk_id_livro, char * nome, char * autor, char * editora, unsigned int ano_publicacao, unsigned int qtd_copias)
(1) Tentar editar com campos de dados inválidos (número de cópias negativo).
(2) Editar um livro inexistente (ID não encontrado).
(5) Falha ao atualizar o livro no banco de dados (erro de sistema).

exclui_livro(unsigned int pk_id_livro)
(2) Tentar excluir um livro inexistente (ID não encontrado).
(4) Tentar excluir um livro que tem empréstimos ativos.
(5) Falha ao excluir o livro no banco de dados (erro de sistema).

lista_livros(char * titulo, char * autor, char * editora, unsigned int ano_publicacao)
(5) Falha ao acessar a lista de livros (erro de sistema).

Emprestimo

gera_emprestimo(char * data_emprestimo, char * data_devolucao_real, char * data_devolucao_prevista, id_livro, id_usuario)
(1) Parâmetros inválidos (livro ou usuário ausente).
(3) Tentar gerar um empréstimo para um livro já emprestado.
(4) Tentativa de empréstimo por um usuário bloqueado ou com multas pendentes.
(5) Falha no processo de gerar empréstimo (erro de sistema).

exclui_emprestimo(unsigned int pk_id_emprestimo)
(2) Tentativa de excluir um empréstimo inexistente (ID não encontrado).
(3) Tentativa de excluir um empréstimo já encerrado.
(5) Falha ao excluir o empréstimo no sistema (erro de sistema).

acaba_emprestimo(unsigned int pk_id_emprestimo)
(2) Tentar encerrar um empréstimo inexistente.
(4) Tentativa de encerrar um empréstimo sem devolver o livro.
(5) Falha ao encerrar o empréstimo no sistema.

PesquisaEmprestimo(unsigned int pk_id_emprestimo)
(1) Parâmetros de pesquisa inválidos (datas em formato incorreto).
(2) Nenhum empréstimo encontrado para os parâmetros fornecidos.
(5) Erro ao realizar pesquisa de empréstimos no banco de dados.

visualizaEmprestimo(unsigned int pk_id_emprestimo)
(2) Tentativa de visualizar um empréstimo inexistente.

(5) Falha ao acessar detalhes do empréstimo no banco de dados.

Multa

gera_multa(unsigned int valor, char * data, unsigned int, char tipo, char * data_geracao, char status)
(1) Parâmetros inválidos ao gerar multa (valor negativo).
(3) Tentativa de gerar multa para um empréstimo que já tem multa.
(5) Falha no processo de geração de multa (erro de sistema).

visualiza_multa(unsigned int pk_id_multa)
(2) Tentativa de visualizar uma multa inexistente.
(5) Falha ao acessar detalhes da multa no banco de dados.

encerrar_multa(unsigned int pk_id_multa)
(2) Tentar encerrar uma multa inexistente.
(3) Tentar encerrar uma multa que já foi encerrada.
(5) Falha ao encerrar a multa no sistema.

pagar_multa(unsigned int pk_id_multa)
(2) Tentativa de pagar uma multa inexistente.
(3) Tentativa de pagar uma multa já paga.
(5) Falha no processamento do pagamento no sistema.

Usuario

criaUsuario(char * nome, char * endereco, unsigned int telefone, char * email)
(1) Nome, telefone ou email ausente ou inválido.
(3) Tentativa de criar um usuário com ID já existente.
(5) Falha ao criar o usuário no banco de dados.

editaUsuario(unsigned int pk_id_usuario, char * nome, char * endereco, unsigned int telefone, char * email)
(1) Dados inválidos na tentativa de edição (email em formato incorreto).
(2) Tentar editar um usuário inexistente.
(5) Falha ao atualizar informações do usuário no sistema.

excluiUsuario(unsigned int pk_id_usuario)
(2) Tentativa de excluir um usuário inexistente.
(4) Tentativa de excluir um usuário que tem empréstimos ativos.

(5) Falha ao excluir o usuário no banco de dados.

visualizaUsuario(unsigned int pk_id_usuario)
(2) Tentativa de visualizar um usuário inexistente.(5) Falha ao acessar detalhes do usuário no banco de dados.

Funções Gerais

inicializar_sistema()
(5) Falha ao carregar arquivos de configuração ou banco de dados.
(5) Falha ao inicializar componentes essenciais (erro de sistema).

menu_principal()
(1) Entrada inválida ao selecionar uma opção do menu.
(5) Falha ao processar a navegação de menu (erro de sistema).

gera_relatorio_emprestimo()
(2) Nenhum empréstimo encontrado para gerar o relatório.
(5) Falha ao gerar o relatório (erro de sistema).

ConversorTexto

converteTextoUtf8ParaUtf32(const string FILE textoUtf8)
(1) Arquivo de entrada inválido ou formato de texto incorreto.
(5) Falha ao realizar conversão UTF-8 para UTF-32 (erro de sistema).

converteTextoUtf32ParaUtf8(const string FILE textoUtf32)
(1) Arquivo de entrada inválido ou formato de texto incorreto.
(5) Falha ao realizar conversão UTF-32 para UTF-8 (erro de sistema).

Casos de teste integrados da aplicação

Casos de Teste para a Aplicação de Biblioteca:

Caso de Teste Integrado 1: Realizar Empréstimo e Gerar Multa

Módulos envolvidos: Livro, Empréstimo, Multa, Usuário

- **Objetivo:** Testar se o sistema realiza um empréstimo corretamente e, em caso de atraso na devolução, gera uma multa relacionada ao usuário e ao livro.
- **Pré-condição:** Livro e Usuário já cadastrados no sistema.
- **Entrada:**
    - Criar um empréstimo com data de devolução prevista em 2024-10-20.
    - Não registrar a devolução até 2024-10-25.
- **Saída esperada:** O empréstimo é registrado com sucesso. Ao passar da data prevista de devolução, uma multa é gerada automaticamente e associada ao empréstimo.
- **Resultado esperado:** O sistema permite visualizar o empréstimo e a multa relacionados ao usuário e ao livro.

Caso de Teste Integrado 2: Excluir Usuário com Empréstimos Ativos

Módulos envolvidos: Usuário, Empréstimo, Multa

- **Objetivo:** Verificar se o sistema impede a exclusão de um usuário que possui empréstimos ativos ou multas pendentes.
- **Pré-condição:** Um usuário com empréstimos ativos e uma multa pendente.
- **Entrada:** Tentar excluir o usuário que possui um empréstimo ativo e uma multa pendente.
- **Saída esperada:** O sistema bloqueia a exclusão do usuário e exibe uma mensagem de erro.
- **Resultado esperado:** O usuário não é excluído, e o sistema informa que há empréstimos ou multas pendentes.

Caso de Teste Integrado 3: Finalizar Empréstimo e Atualizar Status de Livro

Módulos envolvidos: Livro, Empréstimo

- **Objetivo:** Verificar se, ao finalizar um empréstimo, o status do livro é atualizado corretamente no sistema, tornando-o disponível para outros usuários.
- **Pré-condição:** Um empréstimo ativo para um livro no sistema.
- **Entrada:** Chamar a função para finalizar o empréstimo.

- **Saída esperada:** O empréstimo é finalizado, e o status do livro é atualizado para "disponível" para novos empréstimos.
- **Resultado esperado:** O sistema permite visualizar que o empréstimo foi finalizado e que o livro está disponível novamente.

Caso de Teste Integrado 4: Criar Usuário e Realizar Empréstimo

Módulos envolvidos: Usuário, Empréstimo, Livro

- **Objetivo:** Verificar se o sistema permite criar um novo usuário, realizar um empréstimo de um livro e associar o empréstimo ao usuário recém-cadastrado.
- **Pré-condição:** O sistema está em execução com livros cadastrados.
- **Entrada:** Criar um novo usuário e realizar um empréstimo de um livro disponível para este usuário.
- **Saída esperada:** O usuário é criado com sucesso e o empréstimo é associado ao usuário e ao livro corretamente.
- **Resultado esperado:** O sistema permite visualizar o empréstimo feito pelo usuário e o status do livro como "emprestado".

Caso de Teste Integrado 5: Converter Texto e Salvar Multa Gerada em Relatório

Módulos envolvidos: Multa, ConversorTexto, Main

- **Objetivo:** Verificar se, após a geração de uma multa, o sistema permite salvar as informações da multa em um arquivo de texto, utilizando a conversão de UTF-8 para UTF-32.
- **Pré-condição:** Uma multa já gerada para um empréstimo.
- **Entrada:** Chamar a função para gerar um relatório do empréstimo e da multa, e converter o arquivo de texto de UTF-8 para UTF-32.
- **Saída esperada:** O relatório é gerado e convertido para UTF-32 corretamente.
- **Resultado esperado:** O sistema salva o arquivo de texto convertido com as informações da multa e do empréstimo.

Caso de Teste Integrado 6: Criar Relatório de Usuários com Empréstimos Ativos e Livros Emprestados

Módulos envolvidos: Usuário, Empréstimo, Livro, Main

- **Objetivo:** Verificar se o sistema gera corretamente um relatório de todos os usuários com empréstimos ativos e os livros associados.
- **Pré-condição:** Existem empréstimos ativos e usuários no sistema.
- **Entrada:** Chamar a função para gerar um relatório dos usuários com empréstimos ativos.
- **Saída esperada:** O sistema gera um relatório com as informações de cada usuário e o livro que está com eles.
- **Resultado esperado:** O relatório é gerado corretamente, listando todos os usuários e os livros emprestados.