INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA PARAÍBA

Projeto Final – Sistema de Gestão de Biblioteca Digital

Objetivo: Desenvolver um sistema que permite usuários cadastrarem, pesquisarem, alugarem e avaliarem livros digitais. Também possui áreas administrativas, de recomendação e estatísticas. Cada equipe ficará responsável por um módulo independente.

Tecnologia: Linguagem Java, JavaScript ou Python

Equipe 1: Gerenciamento de Usuários e Perfis
Membros: Erik e Eduardo

1. Cadastro e autenticação:
    - Criar formulário de cadastro e login.
    - Implementar verificação de credenciais (email/senha).
    - Gerar tokens ou sessões para usuários logados.
2. Perfis de usuário:
    - Definir perfis: Administrador, Leitor Comum, Leitor Premium.
    - Restringir funcionalidades conforme o perfil (ex: apenas administradores podem excluir livros).
3. Controle de acesso:
    - Criar mecanismos para impedir acesso não autorizado a rotas/métodos.
4. Notificações ao usuário:
    - Ex: "Seu livro está atrasado", "Novo livro adicionado em sua categoria favorita".

Padrões que devem ser aplicado:

- **Singleton**: Controlador de sessão/logado (UserSessionManager).
- **Factory Method:** Criação de objetos `Usuario`, `Administrador`, `LeitorPremium`.
- **Proxy:** Verificação de permissões antes de executar métodos.
- **Observer:** Sistema de notificações (usuário se inscreve para receber avisos).

Equipe 2: Catálogo e Cadastro de Livros

Membros: Kayo e Marcelo

1. Cadastro de livros:
    - Título, autor, ano, editora, ISBN, sinopse, categoria.
    - Interface para administradores adicionarem e editarem livros.
2. Visualização e busca:
    - Interface para listagem com filtros por categoria, autor, palavras-chave.
    - Sistema de paginação.
3. Organização por coleções:
    - Ex: Trilogias, séries ou autores específicos.
4. Metadados extras:
    - Destaques, tags, capas alternativas, resumos estendidos.

Padrões que devem ser utilizados:

- **Builder**: Criação de objetos `Livro` com muitos atributos.
- **Composite**: Representar uma coleção como um conjunto de livros.
- **Repository**: Abstração da persistência de dados dos livros.
- **Decorator**: Adicionar comportamentos extras ao objeto `Livro` (ex: `LivroComTag`, `LivroDestacado`).

Equipe 3: Empréstimos, Reservas e Devoluções
Membros: José Jefferson e Marcos Paulo

1. Solicitação de empréstimo:
    a. Verificar disponibilidade do livro.
    b. Validar regras conforme tipo de usuário.

2. Controle de prazos e renovações:
    a. Cálculo de data de devolução.
    b. Permitir renovação se o livro não estiver reservado.

3. Multas e notificações:
    a. Aplicar multas por atraso.
    b. Enviar alertas ao usuário sobre prazo final.

4. Reservas:
    a. Usuário pode reservar livro atualmente indisponível.
    b. Notificação quando o livro estiver disponível.

5. Histórico de ações do usuário:
    a. Exibir histórico completo de empréstimos e devoluções.

Padrões que devem ser aplicados:

- **State:** Controle do estado do livro (Disponível, Emprestado, Reservado, Atrasado).
- **Strategy:** Diferentes estratégias de empréstimo (tempo, limite).
- **Command:** Cada ação do usuário (emprestar, renovar, devolver) encapsulada como comando.
- **Memento:** Armazenar snapshots do histórico de transações para possível restauração/consulta.

Equipe 4: Avaliação, Comentários e Recomendação de Livros
Membros: José Isaias e José Tiago

1. Sistema de avaliações:
    - Nota de 1 a 5 estrelas.
    - Comentários abertos e resenhas de livros.
2. Média de avaliação por livro:
    - Atualizar automaticamente conforme novas avaliações.

3. Sistema de recomendação:
    - Com base nos livros mais lidos, mais avaliados, ou parecidos com os já lidos pelo usuário.
4. Filtros e personalização:
    - Recomendação por autor, categoria, faixa etária etc.
5. Ranking e destaques:
    - Livros mais bem avaliados, mais comentados, tendências semanais.

Padrões que devem ser aplicados:

- **Observer**: Atualiza a média de avaliação assim que um novo comentário é registrado.
- **Chain of Responsibility:** Permite aplicar uma cadeia de critérios para gerar recomendações.
- **Adapter:** Integração com APIs externas (simuladas).
- **Flyweight:** Compartilhamento de objetos repetidos em avaliações (ex: comentários genéricos).

Equipe 5: Administração e Relatórios do Sistema
Membros: Mateus e Robert

1. Painel administrativo:
    - Visualização do status geral do sistema.
    - Gráficos e dashboards de uso.
2. Estatísticas do sistema:
    - Livros mais emprestados
    - Categorias mais buscadas
    - Usuários mais ativos
3. Geração de relatórios:
    - Exportar dados em formatos (PDF, CSV, Excel).
    - Relatórios semanais, mensais e anuais.
4. Monitoramento de desempenho:
    - Quantidade de livros cadastrados, avaliações por período, reservas pendentes.

Padrões que devem ser aplicados:

- **Facade:** Centraliza as operações de geração de relatórios e estatísticas.
- **Template Method:** Define o esqueleto dos relatórios com partes personalizáveis.
- **Iterator:** Navega entre grandes conjuntos de dados.
- **Bridge:** Permite trocar o formato de saída (PDF, CSV) sem alterar a lógica dos dados.

Integração entre as Equipes

- As **interfaces dos módulos devem ser bem definidas** desde o início:

- Ex: `ILivroService`, `IUsuarioService`, `IAvaliacaoService`, `IRelatorioService`

Fluxo entre os módulos:

- **Equipe 1** fornece dados de usuários para todas as outras.
- **Equipe 2** compartilha informações dos livros com Empréstimos (Equipe 3) e Avaliações (Equipe 4).
- **Equipe 3** consulta usuários e livros para controlar empréstimos e gera dados para relatórios.
- **Equipe 4** interage com Catálogo e com os usuários, gerando feedback e recomendação.
- **Equipe 5** recebe dados de todas as outras para gerar estatísticas e relatórios.