UNIVERSIDADE DE PERNAMBUCO

ESCOLA POLITÉCNICA DE PERNAMBUCO

GRADUAÇÃO EM ENGENHARIA DA COMPUTAÇÃO

Documento de Projeto de Software

Sistema de Gerenciamento de Monitorias, Estágios e   
Teses de Conclusão de Curso (TCCs)

Grupo:

Bruno César Protasio de Lima Filho

Daniel Guilherme Pereira de Meneses

Paulo Renato Ferraz de Carvalho

RECIFE

2025

### Design Arquitetural do Sistema de Gestão Acadêmica

### 1. Contexto do Projeto

#### 1.1. Objetivo deste Documento

Este artefato estabelece o design técnico e a fundação arquitetural para o **Sistema de Gestão Acadêmica**. O propósito é servir como um guia definitivo para a equipe de desenvolvimento, convertendo os requisitos de negócio em uma especificação técnica detalhada para a construção e implementação do software.

#### 1.2. Visão Geral do Sistema

O escopo do sistema é criar uma plataforma web centralizada para administrar o ciclo de vida completo das atividades de **monitoria**, **estágio** e **Trabalho de Conclusão de Curso (TCC)**. A solução visa otimizar os processos de cadastro, acompanhamento e avaliação, unificando a experiência para todos os envolvidos: alunos, professores e a administração do curso.

### 2. Estratégia Arquitetural

#### 2.1. Modelo Arquitetural

A arquitetura adotada é a de **Camadas (Layered Architecture)**, com uma separação clara entre o cliente (Frontend) e o servidor (Backend). O backend implementará o padrão **Model-View-Controller (MVC)**, expondo suas funcionalidades através de uma **API REST**.

Esta escolha se justifica pela alta **separação de responsabilidades (SoC)**, que promove:

* **Baixo Acoplamento:** As camadas podem evoluir de forma independente.
* **Alta Coesão:** Cada componente tem uma responsabilidade bem definida.
* **Facilidade de Manutenção:** A lógica é isolada e mais fácil de depurar e estender.
* **Testabilidade:** As regras de negócio podem ser testadas sem depender da interface do usuário.

#### 2.2. Componentes e Responsabilidades das Camadas

O fluxo de dados no sistema seguirá uma trajetória unidirecional: View → Controller → Service → Repository → Database.

* **Camada de Apresentação (View):**
  + **Tecnologia:** Frontend (ex: Single-Page Application em React, Angular ou Vue).
  + **Responsabilidade:** É a interface com o usuário. Renderiza os dados recebidos da API e captura as interações do usuário, sem conter lógica de negócio.
* **Camada de API (Controller):**
  + **Tecnologia:** Backend (Endpoints da API REST).
  + **Responsabilidade:** Funciona como a porta de entrada para o backend. Recebe requisições HTTP, valida os dados de entrada (payloads) e aciona a camada de Serviço para executar a lógica correspondente.
* **Camada de Negócio (Service):**
  + **Tecnologia:** Backend (Classes de serviço).
  + **Responsabilidade:** É o núcleo do sistema. Implementa as regras de negócio, orquestra as operações e processa os dados, garantindo a integridade e a lógica dos casos de uso.
* **Camada de Acesso a Dados (Repository):**
  + **Tecnologia:** Backend (Implementação do *Repository Pattern*).
  + **Responsabilidade:** Abstrai e centraliza toda a comunicação com o banco de dados. Isola o restante da aplicação dos detalhes de implementação da persistência (SQL, etc.).
* **Camada de Domínio (Model):**
  + **Tecnologia:** Backend (POJOs/POCOs).
  + **Responsabilidade:** Modela os dados do sistema através de:
    - **Entities:** Objetos que mapeiam diretamente as tabelas do banco de dados.
    - **DTOs (Data Transfer Objects):** Objetos simples e seguros para transferir dados entre as camadas, especialmente para a camada de apresentação.

### 3. Mapeamento da Análise para o Projeto

Esta seção traduz os artefatos da fase de análise em componentes concretos do projeto de software.

#### 3.1. Componentes de Interface (Views)

| **Artefato de Análise (Tela)** | **Componente de Projeto (View)** | **Racional** |
| --- | --- | --- |
| Tela\_Login,  Tela\_Criar\_Conta, Tela\_Recuperar\_Senha | AuthView | Agrupa todas as funcionalidades de autenticação em um único módulo visual para o usuário não autenticado. |
| Tela\_Visualizar\_Painel, Tela\_Atualizar\_Cadastro  Tela\_Logoff | DashboardView, PerfilUsuarioView | O DashboardView é a tela principal pós-login. PerfilUsuarioView é uma seção dedicada à edição de dados do usuário. A ação de Logoff será um botão no layout principal. |
| Tela\_Submeter\_Candidatura, Tela\_Acompanhar\_Candidatura  Tela\_Cancelar\_Candidatura | GerenciamentoCandidaturasView | Consolida o ciclo de vida da candidatura (submissão, acompanhamento, cancelamento) em uma única interface para o aluno. |
| Tela\_Enviar\_Documentos, Tela\_Cancelar\_Envio\_Documentos | ComponenteDocumentoView | Abstrai a funcionalidade de upload em um componente reutilizável, aplicável em diferentes contextos do sistema. |
| Tela\_Visualizar\_Orientandos, Tela\_Filtrar\_Orientandos | OrientandosView | Combina a listagem de orientandos com os controles de filtro em uma única tela para o professor. |
| Tela\_Aceitar\_Orientacao, Tela\_Rejeitar\_Orientacao | AvaliacaoOrientacaoView | Centraliza a decisão do professor (aceitar/rejeitar) em uma tela de avaliação da solicitação de orientação. |
| Tela\_Aprovar\_Relatório, Tela\_Rejeitar\_Relatório | AvaliacaoRelatorioView | Unifica o processo de avaliação de relatórios, apresentando as opções de aprovação e rejeição ao professor. |
| Tela\_Ver\_Status\_Documentos, Tela\_Ver\_Prazos | AcompanhamentoAtividadeView | Centraliza o acompanhamento de uma atividade em andamento (TCC, Estágio, etc.) para o aluno, mostrando o status dos documentos e pendências. |
| Tela\_Ver\_Feedback, Tela\_Adicionar\_Feedback, Tela\_Remover\_Feedback | DetalheEntregaView | Consolida a visualização de um documento ou entrega com a seção de feedbacks, permitindo a leitura (aluno) e a gestão (professor). |
| Tela\_Ver\_Histórico\_Orientando | HistoricoOrientacaoView | Exibe ao professor o histórico cronológico completo de uma atividade de um aluno orientado, incluindo todas as submissões e feedbacks. |
| Tela\_Registrar\_Horas\_Monitoria, Tela\_Diario\_Monitoria | RegistroMonitoriaView | Oferece a interface específica (formulário) para que o aluno Monitor possa registrar e gerenciar suas atividades periódicas. |

#### 3.2. Estrutura de Dados (Model & Repositories)

| **Artefato de Análise (Entidade/Coleção)** | **Componentes de Projeto** | **Racional** |
| --- | --- | --- |
| Usuario, ColecaoUsuarios | Usuario (Entity), UsuarioDTO, UsuarioRepository | A Entity espelha a tabela do BD. O DTO garante a transferência segura de dados. O Repository encapsula o acesso aos dados do usuário. |
| Candidatura, ColecaoCandidaturas | Candidatura (Entity), CandidaturaDTO, CandidaturaRepository | Isola a lógica de persistência no Repository e desacopla a representação de dados (Entity vs. DTO). |
| Monitoria, Estagio (Entidades) | ProcessoAcademico (Entity), ProcessoAcademicoDTO, ProcessoAcademicoRepository | Generaliza diferentes processos acadêmicos em uma única estrutura, simplificando o modelo de dados através de um campo "tipo". |
| Documento, ColecaoDocumentos | Documento (Entity), DocumentoDTO, DocumentoRepository | Define uma estrutura formal para persistência e manipulação de documentos, com acesso centralizado via Repository. |
| Email, Senha, Sessão | (Não são Entidades Persistentes) | São considerados dados transitórios ou atributos de outras entidades, não necessitando de tabelas próprias no banco de dados. |

#### 3.3. Lógica de Negócio (Controllers & Services)

| **Artefato de Análise (Controlador)** | **Componentes de Projeto** | **Racional** |
| --- | --- | --- |
| Controlador\_Login, Controlador\_Criar\_Conta | AuthController, AuthService | Agrupa todos os serviços relacionados à autenticação e segurança da API em um único ponto de entrada. |
| Controlador\_Atualizar\_Cadastro, Controlador\_Visualizar\_Painel | UsuarioController, UsuarioService | Dedicado às operações que o usuário pode realizar em seu próprio perfil e dados. |
| Controlador\_Submeter\_Candidatura, Controlador\_Cancelar\_Candidatura | CandidaturaController, CandidaturaService | Gerencia o fluxo completo das candidaturas, desde a submissão até o cancelamento, contendo toda a lógica de negócio associada. |
| Controlador\_Aceitar\_Orientacao, Controlador\_Rejeitar\_Orientacao | AvaliacaoOrientacaoController, AvaliacaoOrientacaoService | Encapsula a lógica de negócio do professor ao avaliar e decidir sobre as solicitações de orientação dos alunos. |
| Controlador\_Aprovar\_Relatório, Controlador\_Rejeitar\_Relatório | AvaliacaoRelatorioController, AvaliacaoRelatorioService | Isola a lógica de avaliação de relatórios, incluindo a aplicação de regras e a notificação dos resultados. |
| Controlador\_Definir\_Prazos, Controlador\_Gerar\_Certificado | AdminController, AdminService | Consolida as funcionalidades de alta permissão, restritas ao perfil de Funcionário Administrativo. |
| Controlador\_Definir\_Regras\_Curso, Controlador\_Trocar\_Orientador | CoordenadorController, CoordenadorService | Agrupa as operações estratégicas e de gestão que são de responsabilidade exclusiva do Coordenador de Curso. |