

Projeto Arquitetural - NexuS

Recife, 04 de julho de 2025

Ficha Técnica

Equipe Responsável pela Elaboração:

Daniel Dias Lopes Farias (Engenharia da Computação)
Gabriel Correia de Albuquerque (Engenharia da Computação)
Mário Stela Guerra (Engenharia da Computação)
Sérgio Henrique de Andrade Lima Filho (Engenharia da Computação)

Público Alvo

Este documento arquitetural destina-se a gestores acadêmicos, professores orientadores, alunos de graduação, equipe de TI e mantenedora institucional que venham a desenvolver e utilizar o sistema.

Versão 1.0 - Recife, julho de 2025

Dúvidas, críticas e sugestões devem ser encaminhadas por escrito para o seguinte endereço eletrônico:

ddlf@ecomp.poli.br

Recomendamos que o assunto seja identificado com o título desta obra. Alertamos ainda para a importância de se identificar o endereço e o nome completos do remetente para que seja possível o envio de respostas.

Introdução

O presente **Documento de Projeto Arquitetural** detalha a estrutura de software e os componentes necessários para a implementação do sistema **NexuS** (Núcleo de Experiências Supervisionadas), conforme especificado no Documento de Requisitos (SRS) versão 1.0 de junho de 2025 e baseado nos diagramas de classes gerados na fase de projeto. O NexuS é uma plataforma acadêmica integrada voltada à gestão digital de atividades supervisionadas de alunos universitários, incluindo Monitoria, Estágio Curricular Obrigatório e Trabalho de Conclusão de Curso (TCC). Sua concepção parte da necessidade observada, expressa no Documento de Requisitos (SRS), de modernizar os fluxos institucionais que tradicionalmente dependem de papeladas físicas, e-mails descentralizados e processos não rastreáveis.

Contexto e Justificativa

A arquitetura do NexuS foi projetada para atender tanto os requisitos funcionais descritos no SRS — como o cadastro e gerenciamento de vagas de monitoria, envio de relatórios de estágio e submissão de propostas de TCC — quanto os requisitos não funcionais como **usabilidade, segurança, rastreabilidade e modularidade**. Entre as diretrizes arquiteturais centrais, destacam-se a **separação de responsabilidades em camadas bem definidas**, o suporte à **escalabilidade horizontal**, a **rastreabilidade de eventos e documentos**, e a compatibilidade com diferentes perfis de usuários, tais como **alunos, professores e a escolaridade**.

O sistema será construído sobre uma **arquitetura em camadas**, composta por:

- **Camada de apresentação:** responsável pela interface com o usuário e renderização das telas prototipadas no Figma;
- **Camada de aplicação:** encarregada da orquestração dos casos de uso e das interações REST;
- **Camada de domínio:** contendo as entidades centrais como Aluno, Monitoria, Estágio, TCC, Documentos, entre outras, bem como os serviços que implementam a lógica de negócio;
- **Camada de infraestrutura:** onde se encontram os mecanismos de persistência, integrações com serviços externos como armazenamento de arquivos, autenticação, envio de notificações e controle de acesso.

Base de Modelagem

Esta arquitetura foi definida a partir dos modelos de análise desenvolvidos na fase anterior, especialmente o **diagrama de classes** que representa a modelagem conceitual do sistema. Com base nele, foram extraídos os principais **módulos do sistema**, organizados por áreas funcionais (Monitoria, Estágio e TCC), que foram refinados em componentes de projeto com responsabilidades delimitadas.

Premissas Arquiteturais

A arquitetura proposta considera aspectos operacionais importantes, como a necessidade de **registros auditáveis** para garantir a integridade dos fluxos institucionais (conforme previsto na regra de negócio RN10), e o uso de **assinatura digital com carimbo do tempo** (RN03) para garantir validade jurídica em documentos enviados e assinados dentro do sistema. Outro ponto crítico abordado pela arquitetura é a **separação de perfis de acesso** com diferentes níveis de permissão, o que impacta diretamente o desenho dos controladores e serviços de autorização.

Finalidade

Este documento tem como finalidade não apenas servir como guia de referência para a equipe de desenvolvimento durante a implementação, mas também como instrumento de comunicação entre os membros do time técnico e os stakeholders institucionais. O conteúdo aqui descrito irá detalhar cada camada do sistema, explicitar os componentes principais e suas responsabilidades, definir fluxos de interação entre os módulos, apresentar os principais pontos de integração externa (como sistema de autenticação e armazenamento de documentos), e indicar o modelo de implantação esperado, que inclui o uso de **contêineres Docker**, **versionamento via GitHub** e **pipelines de integração e entrega contínua (CI/CD)**.


Camada de Apresentação

A camada de apresentação do **NexuS** é o ponto de contato direto entre os usuários e os serviços oferecidos pelo sistema. Essa camada foi concebida para ser clara, responsiva, acessível e coerente com a identidade visual proposta no protótipo interativo desenvolvido no Figma. Ela não apenas traduz os requisitos funcionais descritos no SRS em elementos visuais e interativos, como também cumpre requisitos não funcionais de usabilidade, navegabilidade, acessibilidade e aderência a boas práticas de desenvolvimento front-end.


A arquitetura da camada será orientada a componentes e implementada com a biblioteca React, que permite a composição modular da interface em unidades reutilizáveis e de fácil manutenção. O estilo visual será definido com o uso do Tailwind CSS, uma biblioteca utilitária que facilita a estilização responsiva e mantém a consistência com o design system. O roteamento de páginas será controlado via React Router, possibilitando navegação entre os módulos de forma fluida e sem recarregamento da página. A comunicação com a API da aplicação poderá ser feita através do Axios, que encapsula chamadas HTTP, cuidando da serialização de dados, inclusão do token JWT nos headers e tratamento de respostas e erros.

Estrutura das Telas e Componentes

Cada tela corresponde a um conjunto de funcionalidades descritas no SRS. A interface foi dividida em páginas, cada uma representando um módulo funcional (TCC, Monitoria e Estágio), além de páginas genéricas como Login e Perfil.

Link para o documento de telas:  Documento de Telas - Sistema Nexus.pdf

Link para o protótipo:  Figma

Link para o SRS:  DocumentoDeRequisitosNexus.pdf

Tela de Login

- **LoginForm**: coleta credenciais (e-mail e senha), envia ao endpoint `/api/usuarios/authenticate` e armazena o JWT retornado no `localStorage`, redireciona o usuário de acordo com o tipo de perfil.
- Relaciona-se à tela 1.1 do documento de telas.

Tela de Recuperação de Senha

- **RecuperacaoSenhaForm**: coleta credenciais (e-mail) e envia uma mensagem com link para redefinição de senha.
- Relaciona-se à tela 1.2 do documento de telas.

Tela de Início

- Composta por três elementos principais:
 - **WelcomeBanner**: renderiza uma saudação personalizada com um texto institucional de boas-vindas e uma pequena descrição.
 - **StatusAcademicoCard**: mostra, em formato de resumo, a situação atual do usuário nos três módulos principais. Exibe, por exemplo, o número de propostas de TCC ativas, monitorias em andamento ou relatórios pendentes de estágio.
 - **NewsCarousel**: carrossel rotativo com mensagens institucionais, prazos e orientações sobre uso do sistema. Cada slide é vinculado a um módulo ou regra de negócio.
 - Relaciona-se à tela 1.4 do documento de telas.

Módulo de Monitoria

- **Visão do Aluno (Tela 2.1 do documento de telas):**
 - **Inscricao:** permite que o aluno se inscreva em vagas de monitoria abertas. Telas: 2.1.1, 2.1.1.1, 2.1.1.2 do documento de telas.
 - **MinhasMonitorias:** permite que o aluno acesse as disciplinas em que é Monitor. Telas: 2.1.2 e 2.1.2.1 do documento de telas.
 - **DisciplinaMonitoria:** permite que o aluno acesse as monitorias das disciplinas que está cursando para trocar mensagens com os monitores e visualizar atividades cadastradas. Telas: 2.1.3 e 2.1.3.1 do documento de telas.
 - **SubmissãoRelatorio:** permite a submissão do relatório final de estágio para a conclusão das atividades como monitor. Telas: 2.1.4 e 2.1.4.1 do documento de telas.
- **Visão do Professor (Tela 3.1 do documento de telas):**
 - **CadastrarMonitoria:** permite que o professor cadastre vagas de monitoria para as suas disciplinas. Telas: 3.1.1 e 3.1.1.1 do documento de telas.
 - **VisualizarPropostasMonitoria:** permite que o professor acesse e avalie as propostas de monitores para as suas disciplinas. Telas: 3.1.2 e 3.1.2.1 do documento de telas.
 - **AcompanharMonitoria:** permite que o professor acesse as monitorias das suas disciplinas para trocar mensagens com os monitores, visualizar atividades cadastradas e agendar encontros virtuais. Telas: 3.1.3, 3.1.3.1, 3.1.3.2 do documento de telas.
- **Visão da Escolaridade (Tela 4.2 do documento de telas):**
 - **GerenciarMonitorias:** permite que o administrador gerencie as monitorias do sistema para visualizar, editar, bloquear ou excluir vagas de monitoria. Telas: 4.2.1 do documento de telas.

Módulo de Estágio

- **Visão do Aluno (Tela 2.2 do documento de telas):**
 - **CadastrarEstagio:** permite que o aluno cadastre um novo estágio obrigatório ou não obrigatório, além de conseguir selecionar o professor orientador. Telas: 2.2.1, 2.2.1.1 e 2.2.1.2 do documento de telas.
 - **SubmissaoDocumentos:** permite que o aluno anexe os documentos

necessários para o desenvolvimento do seu estágio. Telas: 2.2.2, 2.2.2.1 e 2.2.2.2 do documento de telas.

- **Visão do Professor (Tela 3.2):**

- **VisualizarPropostasEstagio**: permite que o professor acesse e avalie proposta para orientação de estágio. Telas: 3.2.1 do documento de telas.
- **SubmissaoDocumentos**: permite que o professor corrija e anexe os documentos necessários para o acompanhamento de estágio de um orientando. Telas: 3.2.2, 3.2.2.1 e 3.2.2.2 do documento de telas.
- **AcompanharEstagio**: permite que o professor envie mensagens, agende encontros e visualize os documentos enviados pelo orientando. Telas: 3.2.3, 3.2.3.1 e 3.2.3.2 do documento de telas.

- **Visão da Escolaridade (Tela 4.3 do documento de telas):**

- **GerenciarPropostasEstagio**: permite que a escolaridade visualize e avalie os estágios do aluno, para validar documentos e exigir possíveis correções. Telas: 4.3.1 do documento de telas.

Módulo de TCC

- **Visão do Aluno (Tela 2.3 do documento de telas):**

- **SubmeterPropostaTCC**: permite que o aluno cadastre sua proposta de TCC e escolha seu professor orientador. Telas: 2.3.2, 2.3.2.1 e 2.3.2.2 do documento de telas.
- **AcompanhamentoTCC**: permite que o aluno converse, tire dúvidas, adicione documentos e agende encontros com o orientador. Telas: 2.3.1 do documento de telas.
- **SubmeterVersaoFinalTCC**: permite que o aluno envie a versão final do seu TCC para validação do orientador e finalização do processo. Telas: 2.3.3, 2.3.3.1 do documento de telas.

- **Visão do Professor (Tela 3.3 do documento de telas):**

- **VisualizarPropostasTCC**: permite que o professor visualize e avalie as propostas recebidas para orientação de TCC. Telas: 3.3.1 do documento de telas.
- **AcompanhamentoTCCProfessor**: permite que o professor converse, tire dúvidas, visualize e corrija documentos, além de agendar encontros com o orientando. Telas: 3.3.2, 3.3.2.1 e 3.3.2.2 do

documento de telas.

- **Visão da Escolaridade (Tela 4.4):**

- **AcompanharVinculoTCC:** permite que a escolaridade acompanhe os vínculos ativos de TCC. Telas: 4.4.2 do documento de telas.
- **ConfirmarOrientacaoTCC:** permite que a escolaridade formalize a orientação dos trabalhos conclusão de curso. Telas: 4.4.1 do documento de telas.

Elementos Globais

- **NavbarNavigation:** menu superior com links para os módulos e logout.
- **HeaderBar:** exibe saudação, avatar do usuário, e ícone de notificações.
- **Perfil:** exibe informações do usuário logado. Tela 1.3 do documento de telas.

Camada de Aplicação

A camada de aplicação do NexuS é responsável por coordenar toda a lógica de negócio, orquestrar o fluxo de informações entre as interfaces de usuário e os componentes internos do sistema, além de garantir a correta execução dos casos de uso definidos no SRS.

Essa camada atua como o cérebro da aplicação, encapsulando regras, controlando o acesso aos serviços e gerenciando a interação entre as diferentes entidades e processos do domínio. Ao isolar a lógica de aplicação, promove-se uma arquitetura mais robusta, testável, escalável e de fácil manutenção.

Sua estrutura é organizada em torno de controladores, interfaces (fachadas) e serviços, compondo um conjunto de módulos que seguem o princípio da separação de responsabilidades. Os controladores implementam as regras específicas de cada subsistema (TCC, Estágio, Monitoria, Usuários e Comunicação), enquanto as interfaces definem contratos claros e padronizados que viabilizam a comunicação com as camadas externas de forma desacoplada.

Para unificar e simplificar o acesso a esses serviços, foi concebida uma fachada central, que age como um ponto de entrada único para a camada de apresentação. Essa fachada implementa todas as interfaces de subsistema e delega as operações para os respectivos controladores, garantindo coesão e reduzindo o acoplamento entre as camadas.

Os diagramas de pacotes, classes e interfaces desenvolvidos nesta etapa refletem a estrutura lógica da camada de aplicação, evidenciando os fluxos de dependência, responsabilidades e pontos de integração. Eles ajudam a visualizar como cada módulo se conecta, destacando o papel da fachada como intermediária entre a interface gráfica e a lógica central do sistema.

Essa abordagem modular e orientada a interfaces facilita futuras evoluções do sistema, integração com novos módulos e aplicação de testes unitários e de integração, contribuindo para a qualidade global do NexuS.

Arquitetura - Diagrama de Classes

Diagrama Geral de Classes

Link do diagrama:  [Classes Total.pdf](#)

Esse diagrama inicial mostra toda a arquitetura do projeto, sem a implementação de fachada ou interface de subsistemas. Dessa forma, temos todas as telas, controladores e entidades do software.

Classes de TCC

- **AgendamentoTCC**: gerencia o agendamento das reuniões do TCC;
- **PropostaTCC**: representa a proposta inicial submetida pelo aluno para aprovação;
- **AvaliacaoFinalTCC**: registra a avaliação final do TCC, podendo ser aprovada ou não;
- **DocumentoTCC**: armazena documentos relacionados ao TCC, como monografia, ficha de avaliação, etc;
- **VinculoTCC**: gerencia a associação entre aluno, orientador e o TCC;
- **ReuniaoOrientacao**: controla reuniões entre aluno e orientador, com datas e observações;

Classes de Estágio

- **AvaliacaoEstagio**: armazena avaliações feitas pelo orientador ou supervisor ao final do estágio;
- **DocumentoEstagio**: documentos obrigatórios para o estágio (relatório, ficha de frequência, etc.);
- **SolicitacaoEstagio**: solicitações de aprovação de estágio feitas pelos alunos;
- **VinculoEstagio**: vínculo formal entre aluno, instituição e empresa para o estágio;

Classes de Monitoria

- **AtividadeMonitoria**: atividades atribuídas ao monitor durante o semestre;

- **EditalMonitoria**: edital público de abertura de vagas para monitoria;
- **InscricaoMonitoria**: inscrições dos alunos no processo seletivo da monitoria;
- **RelatorioFinalMonitoria**: relatório final submetido pelo monitor ao término do período;
- **VinculoMonitoria**: vínculo entre o aluno selecionado e a disciplina monitorada;

Classes de Usuário

- **Usuario**: classe base para autenticação e gerenciamento geral de usuários.
- **Aluno**: extensão da classe **Usuario**, representa estudantes.
- **Professor**: extensão da classe **Usuario**, representa docentes.
- **Escolaridade**: usuário administrador, representa a escolaridade da instituição.

Classes de Comunicação

- **Mensagem**: representa mensagens trocadas entre usuários no sistema;
- **Notificacao**: notificações enviadas aos usuários para alertas e atualizações;
- **Conversa**: agrupamento de mensagens, representando uma thread ou chat;

Classes Gerais

- **Disciplina**: representa disciplinas oferecidas no curso, estando relacionadas principalmente ao professor e a monitoria;
- **AreaConhecimento**: área temática à qual um TCC ou disciplina pertence;
- **Documento**: classe genérica para representar documentos em geral no sistema;

Classes View

- **TelaAgendamentoTCC**: interface para agendar reuniões de TCC;
- **TelaAvaliacaoEstagio**: interface para avaliação de estágio;
- **TelaAvaliacaoFinalTCC**: interface de avaliação final do TCC;
- **TelaCadastroUsuario**: cadastro de novos usuários;
- **TelaConversas**, **TelaMensagens**, **TelaNotificacoes**: comunicação entre usuários;
- **TelaGerenciamentoUsuarios**, **TelaPerfilUsuario**: gestão e visualização de dados do usuário;
- **TelaGestaoDocumentosEstagio**, **TelaGestaoDocumentosTCC**: gestão de documentos;
- **TelaInscricaoMonitoria**, **TelaPublicarEdital**, **TelaSelecaoMonitores**, **TelaRelatorioMonitoria**: monitoria;
- **TelaPropostaTCC**, **TelaReuniaoEstagio**, **TelaSolicitacaoEstagio**, **TelaLogin**: operações diversas relacionadas a TCC, estágio e autenticação;

Classes Controladoras

- **ControladorTCC**: coordena regras e fluxos do TCC;
- **ControladorEstagio**: controla processos de estágio;
- **ControladorMonitoria**: gerencia atividades de monitoria;

- **ControladorUsuario**: autenticação, cadastro e perfis;
- **ControladorComunicacao**: gerencia mensagens e notificações;


Classes Utilitárias

- **StatusAgenda, StatusEdital, StatusInscricao, StatusRelatorio, StatusUsuario, StatusVinculo**: enumeradores que indicam estados de processos.
- **TipoDocumentoEstagio, TipoDocumentoTCC, TipoNotificacao**: tipos de documentos e notificações;

Dessa forma, podemos dividir as classes em módulos para facilitar no entendimento do sistema e na sua manutenção. Seguem abaixo os principais módulos do projeto.

Diagrama de Classes - Módulo de Comunicação

Apresenta as classes responsáveis pela comunicação entre usuários, como **Mensagem**, **Notificacao** e **Conversa**. Mostra suas propriedades e relacionamentos, destacando como os dados de interação são estruturados.

Link do diagrama:  [Classes_comunicacao.pdf](#)

- **Controlador Comunicação**: Classe controller responsável por gerenciar o fluxo de dados dentro desse módulo do sistema;
- **Conversa**: Classe representando uma conversa dentro do sistema entre dois usuários. Dessa forma, possuindo um conjunto de mensagens atreladas a essa única conversa;
- **Mensagem**: Representa uma mensagem entre dois usuários numa conversa;
- **Notificação**: Classe representando uma notificação que pode ser enviada para qualquer usuário do sistema;

Diagrama de Classes - Módulo Estágio

Inclui as classes relacionadas ao gerenciamento de estágios, como **AvaliacaoEstagio**, **DocumentoEstagio**, **SolicitacaoEstagio** e **VinculoEstagio**, facilitando entender a estrutura de dados e as regras de negócio associadas ao processo de estágio, fundamental para alterações e integrações.

Link do diagrama:  [Classes_estagio.pdf](#)

- **AvaliacaoEstagio**: Registra a avaliação final de um estágio, com nota, data e observações, e permite calcular a nota total e solicitar correções;
- **DocumentoEstagio**: Representa cada documento submetido no processo de estágio (plano de atividades, ficha, etc.), com status e links para o arquivo.
- **SolicitacaoEstagio**: Modela o pedido de estágio do aluno, incluindo dados da empresa, cronograma e lista de documentos iniciais.
- **VinculoEstagio**: Mantém o vínculo formal entre aluno e empresa, com datas de início/fim e estado (ativo, rescindido, etc.).

- **ControladorEstagio:** Orquestra os casos de uso do estágio: submeter documentos, avaliar, encerrar vínculos, gerar relatórios.

Diagrama de Classes - Módulo Geral

Mostra classes utilitárias ou genéricas, como **Disciplina**, **AreaConhecimento** e **Documento**, que são compartilhadas por diferentes módulos.

Link do diagrama:  [Classes_geral.pdf](#)

- **Disciplina:** Define uma disciplina do curso, com nome, código, carga horária e método para verificar pré-requisitos de alunos;
- **AreaConhecimento:** Representa áreas e sub-áreas de conhecimento;
- **Documento:** Classe genérica de documento usada por vários módulos para upload, validação e geração de URL;

Diagrama de Classes - Módulo de Monitoria

Foca nas classes específicas do processo de monitoria: **AtividadeMonitoria**, **EditaiMonitoria**, **InscricaoMonitoria**, **RelatorioFinalMonitoria**, **VinculoMonitoria**, permitindo estudar o fluxo de monitoria isoladamente, facilitando análise e evolução de regras específicas do processo de seleção e acompanhamento.

Link do diagrama:  [Classes_monitoria.pdf](#)

- **AtividadeMonitoria:** Registra cada atividade de monitoria (anexos, carga horária, estado) e valida sua carga
- **EditaiMonitoria:** Controla a abertura de vagas, com professor responsável, datas e status de inscrição.
- **InscricaoMonitoria:** Representa a inscrição do aluno, com documentos anexos, status (aprovado, em correção etc.) e nota.
- **RelatorioFinalMonitoria:** Representa o relatório final da monitoria, sendo entregue pelo monitor;
- **VinculoMonitoria:** Vínculo entre aluno, monitoria e professor;
- **ControladorMonitoria:** Gerencia o fluxo completo: publicação de edital, seleção de inscritos, geração de relatórios.

Diagrama de Classes - Módulo de TCC

Abrange as classes referentes ao gerenciamento de TCC, como **AgendamentoTCC**, **PropostaTCC**, **AvaliacaoFinalTCC**, **DocumentoTCC**, **VinculoTCC** e **ReuniaoOrientacao**.

Link do diagrama:  [Classes_TCC.pdf](#)

- **AgendamentoTCC:** Agenda reuniões entre o orientador e orientando;

- **DocumentoTCC:** Documento submetido (capítulos, anexos), com tipo, data de envio e método de avaliação.
- **AvaliacaoFinalTCC:** Registra avaliação final, com nota, parecer e lista de agendamentos/documentos analisados.
- **VinculoTCC:** Controla a relação aluno-orientador-TCC, incluindo reuniões de orientação e ata.
- **ControladorTCC:** Orquestra casos de uso de TCC: submissão de proposta, agendamento, avaliação final.

Diagrama de Classes - Módulo do Usuário

Apresenta as classes **Usuario**, **Aluno**, **Professor** e **Escolaridade**, incluindo hierarquia e atributos, além de organizar a modelagem de usuários e perfis, esclarecendo relações e facilitando futuras alterações em autenticação ou controle de acesso.

Link do diagrama:  [Classes_Usuario.pdf](#)

- **Usuario:** Classe base com autenticação, perfil, status e últimos acessos.
- **Aluno / Professor:** Extensões de Usuário representando os principais usuários desse sistema.
- **Escolaridade:** Nível de formação, com curso, carga horária e pré-requisitos.
- **ControladorUsuario:** Responsável por login, cadastro, alteração de senha e gerenciamento de perfis.

Diagrama de Classes - Módulo Utilitário

Mostra classes de apoio e enums, como **StatusAgenda**, **StatusEdital**, **TipoNotificacao**, **Datetime**, entre outros.

Link do diagrama:  [Classes_util.pdf](#)

- **Enums de Status:** StatusUsuario, StatusVinculo, StatusInscricao, StatusRelatorio, StatusEdital, StatusAgenda, etc.
- **TipoDocumentoEstagio / TipoDocumentoTCC / TipoNotificacao:** Definem categorias e governam validações de formato ou fluxo de aprovação.

Diagrama de Classes - Módulo de Views

Exibe as classes correspondentes às telas do sistema, refletindo os fluxos de interface com o usuário

Link do diagrama:  [Classes_view.pdf](#)

Diagrama de Classes - Módulo de Controladores

Exibe os controladores do sistema: `ControladorTCC`, `ControladorEstagio`, `ControladorMonitoria`, `ControladorUsuario` e `ControladorComunicacao`.

Link do diagrama: [PDF Classes_Controllers.pdf](#)

Arquitetura do Projeto

No início do projeto, o diagrama de classes foi desenvolvido de forma abrangente e direta, apresentando todas as classes de domínio, controladores, views, utilitários e relacionamentos em um único modelo global. Essa abordagem inicial é importante para visualizar **todas as entidades do sistema**, identificar relacionamentos diretos entre os objetos e garantir a cobertura de todos os requisitos funcionais descritos no SRS.

Entretanto, ao evoluir o projeto para uma arquitetura mais modular, escalável e manutenível, percebeu-se a necessidade de **reduzir o acoplamento** entre as diferentes partes do sistema e de **organizar melhor as responsabilidades**.

Com isso, o modelo inicial passou por uma transição arquitetural, incorporando dois conceitos fundamentais:

Subsistemas

Para organizar a complexidade e separar preocupações, o sistema foi dividido em **subsistemas temáticos** (por exemplo, TCC, Estágio e Monitoria). Cada subsistema agrupa classes de domínio, controladores e regras específicas daquele contexto.

Essa separação facilita:

- Evolução e manutenção de cada módulo de forma independente.
- Entendimento isolado de funcionalidades complexas.
- Reutilização futura em outros sistemas ou versões.

Como padrão de projeto para cada subsistema, teremos a definição de uma interface de implementação do subsistema e uma fachada dedicada para cada subsistema do software.


Subsistema TCC:

O Subsistema TCC é responsável por gerenciar todo o fluxo de trabalho relacionado ao Trabalho de Conclusão de Curso, incluindo submissão de propostas, agendamento de bancas, avaliações finais e reuniões de orientação. Este subsistema foi organizado para facilitar o acompanhamento completo do TCC, tanto para alunos quanto para orientadores e avaliadores.

Link do diagrama: [PDF SubsistemaTCC.pdf](#)

Subsistema de Monitoria:

O Subsistema Monitoria gerencia todo o processo de monitoria acadêmica: publicação de editais, inscrição de alunos, seleção, atividades, relatórios e encerramento. Seu objetivo é automatizar e formalizar o acompanhamento dos monitores ao longo do período letivo.

Link do diagrama:  Subsistema_Monitoria.pdf

Subsistema Estágio:

O Subsistema Estágio controla todo o ciclo de vida dos estágios curriculares: solicitações, aprovação de documentos, avaliações e encerramento de vínculos. Este subsistema facilita o gerenciamento tanto para alunos quanto para professores e coordenadores, garantindo rastreabilidade de todo o processo.


Link do diagrama:  SubsistemaEstagio.pdf

Fachada

A fachada é uma camada adicional que centraliza o acesso aos controladores, atuando como ponto único de entrada para a camada de apresentação, simplificando a comunicação com os subsistemas.

Vantagens:

- Unifica a interface do sistema para a camada de UI, ocultando detalhes de implementação.
- Promove maior encapsulamento e controle de fluxo.
- Facilita a manutenção futura e a evolução das regras de negócio.

Link do diagrama:  Fachada.pdf

Camada de Infraestrutura

A camada de infraestrutura do sistema NexuS é a base técnica que sustenta a execução da aplicação, oferecendo suporte à persistência de dados, à comunicação com serviços externos e à gestão de aspectos transversais como autenticação, segurança, envio de arquivos, notificações e logs. Essa camada será composta por implementações concretas de repositórios, configurações de integração e adaptações técnicas que interagem diretamente com recursos da plataforma.

A infraestrutura do NexuS é projetada para ser modular e substituível, permitindo abstrair detalhes de implementação e garantir a manutenção e extensibilidade do sistema. Os artefatos definidos na camada de aplicação (repositórios e serviços) serão aqui implementados utilizando tecnologias específicas, como Spring Data

JPA, Amazon S3, Firebase Cloud Messaging, JWT, e bibliotecas de assinatura digital com carimbo do tempo.

Persistência de Dados

Toda a persistência será realizada através do Spring Data JPA, que fornece implementações automáticas para os repositórios definidos na camada de domínio. O sistema estará conectado a um banco de dados PostgreSQL, estruturado com schemas relacionais que refletem as entidades do modelo conceitual citado na camada de aplicação.

Cada entidade persistente é anotada com `@Entity` e possui mapeamentos claros de seus atributos, relações (OneToMany,ManyToOne, etc.) e constraints (unique, not null, etc).

Armazenamento de Documentos

Para garantir conformidade com o requisito RN03 (documentos com validade jurídica), todo documento enviado é armazenado em um bucket Amazon S3, com controle de versão habilitado e metadados para rastreamento. A manipulação dos arquivos é feita através da SDK oficial da AWS, encapsulada em um serviço `FileStorageService`.

Arquivos são nomeados de forma determinística com hash, identificador de usuário e timestamp, evitando conflitos e garantindo rastreabilidade. O link de download é protegido por tempo de expiração e requer token JWT ativo.

Autenticação e Segurança

O controle de autenticação será realizado por meio de tokens JWT (JSON Web Tokens), emitidos pelo AuthService ao realizar login. Cada requisição à API é interceptada por um filtro (`JwtRequestFilter`), que valida o token e injeta o contexto do usuário logado. Perfis de acesso (Aluno, Professor e Escolaridade) são usados para habilitar ou restringir operações conforme regras de negócio.

A configuração de segurança será feita via Spring Security com CORS habilitado apenas para domínios autorizados. Endpoints sensíveis utilizam HTTPS com TLS 1.3 e cabeçalhos seguros (HSTS, CSP, X-Frame-Options) são aplicados via middleware.

Modelo de Implantação

A implantação do sistema NexuS será realizada com base em uma arquitetura de microsserviços encapsulados em contêineres, visando escalabilidade, portabilidade e facilidade de manutenção. Cada componente da arquitetura lógica — incluindo as camadas de aplicação, infraestrutura, e a própria interface web — será empacotado de forma isolada e implantado em um ambiente orquestrado, com automação de builds, testes e entregas contínuas.

Contêinerização

A aplicação será dividida em ao menos três imagens principais Docker:

- **nexus-backend**: aplicação Java com Spring Boot que implementa a API REST, conectada ao banco PostgreSQL.
- **nexus-frontend**: aplicação React buildada como static site, servida via Nginx.
- **nexus-worker** (opcional): serviço de background para envio de notificações, processamento assíncrono de assinaturas digitais e rotinas agendadas.

Além disso, serviços externos serão utilizados como dependências:

- **PostgreSQL**: banco de dados relacional, com volume persistente.
- **Amazon S3**: para armazenar documentos enviados e assinados.
- **RabbitMQ**: para tarefas assíncronas e filas de eventos.
- **Prometheus + Grafana**: para métricas de saúde e observabilidade.
- **Kibana**: para registro e análise de logs.

Esses serviços serão definidos em um arquivo `docker-compose.yml` para ambientes de desenvolvimento e testes locais, e migrados para orquestração via Kubernetes (ou similar) em produção.

Ambientes

Serão mantidos pelo menos três ambientes:

- **Desenvolvimento (DEV):** utilizado para desenvolvimento local e integração contínua. Inclui ferramentas de debug, hot reload e testes automatizados.
- **Homologação (HML):** espelho do ambiente de produção com dados fictícios, usado para validação final por professores e coordenadores.
- **Produção (PRD):** ambiente final, acessado por alunos e professores, com segurança reforçada, monitoramento ativo e backup automático.

Considerações Finais e Conclusão

O projeto arquitetural do NexuS foi cuidadosamente elaborado para refletir não apenas os requisitos funcionais e não funcionais especificados no Documento de Requisitos (SRS), mas também os objetivos institucionais de digitalização, rastreabilidade e modernização dos processos acadêmicos relacionados à Monitoria, Estágio Curricular Obrigatório e Trabalho de Conclusão de Curso. Com base em uma arquitetura multicamadas bem definida — composta por apresentação, aplicação, domínio e infraestrutura —, o sistema foi projetado para garantir modularidade, escalabilidade, segurança e extensibilidade.

Ao longo deste documento, foram detalhadas todas as camadas envolvidas, suas responsabilidades, tecnologias adotadas, estratégias de integração e correspondência com as funcionalidades prototipadas no Figma. A camada de apresentação, centrada na experiência do usuário, traduz as operações acadêmicas em uma interface moderna e acessível. A camada de domínio consolida as regras de negócio da instituição, organizando entidades e serviços em módulos coerentes e desacoplados. A infraestrutura, por sua vez, assegura o suporte operacional necessário ao funcionamento do sistema, lidando com persistência, segurança, assinatura digital, armazenamento de arquivos e monitoramento contínuo.

O modelo de implantação, baseado em contêineres Docker e pipelines CI/CD, foi pensado para oferecer flexibilidade, confiabilidade e agilidade tanto no desenvolvimento quanto na manutenção da solução, preparando o sistema para operar em ambientes distribuídos, com alto grau de automação e visibilidade operacional. Com a conclusão deste projeto arquitetural, estabelece-se uma base sólida para o desenvolvimento, implementação e evolução do NexuS, assegurando que as decisões técnicas estejam alinhadas aos objetivos pedagógicos e administrativos da instituição.

