Caio Henrique Bastos Marques, N° 8

Murilo Felipe dos Santos, N° 23

Murilo Ferraz Barbosa, N° 24

Paulo Ricardo de Carvalho, N° 26

Levantamento de Requisitos (LR) - Plataforma Adytum (Projeto Final)

1. Introdução

Este documento detalha os requisitos funcionais e não-funcionais para o desenvolvimento da **Plataforma Adytum**, uma aplicação web completa destinada ao aprimoramento contínuo de desenvolvedores. Diferente do protótipo inicial, este projeto final contemplará todas as camadas da aplicação, incluindo um backend robusto, persistência de dados e funcionalidades em tempo real. O Adytum será uma Single Page Application (SPA), com um design moderno, profissional e inspirador, que combina a clareza e usabilidade da Microsoft com uma identidade visual única de tons de azul intenso e roxo vibrante.

1.1. Propósito

O propósito deste documento é:

- Definir de forma abrangente as funcionalidades que a plataforma Adytum deverá oferecer.
- Estabelecer os critérios de qualidade e as restrições que o sistema deverá atender (requisitos não-funcionais).
- Servir como um contrato entre as partes interessadas para o desenvolvimento do projeto final.
- Fornecer a base técnica para a implementação, testes e validação da plataforma no contexto do Trabalho de Conclusão de Curso (TCC).

1.2. Escopo

O escopo deste projeto abrange o desenvolvimento de uma **aplicação web completa**, incluindo:

- **Frontend:** Interface de usuário interativa e responsiva (SPA).
- **Backend:** Lógica de negócio, API RESTful para comunicação com o frontend.
- **Banco de Dados:** Persistência e gerenciamento de dados de usuários, desafios,

artigos, etc.
- **Módulo de Execução de Código:** Ambiente seguro para compilar e executar código de usuários.

1.3. Definições, Acrônimos e Abreviações

- **Adytum:** Nome da plataforma.
- **SPA:** Single Page Application.
- **UI/UX:** User Interface / User Experience.
- **LR:** Levantamento de Requisitos.
- **RF:** Requisito Funcional.
- **RNF:** Requisito Não-Funcional.
- **TCC:** Trabalho de Conclusão de Curso.
- **API:** Application Programming Interface.
- **CRUD:** Create, Read, Update, Delete.
- **ORM:** Object-Relational Mapping.
- **JWT:** JSON Web Token (para autenticação).
- **IDE:** Integrated Development Environment.
- **CDN:** Content Delivery Network.
- **ACE Editor:** Editor de código.

---

2. Usuários do Sistema

O sistema Adytum interagirá com os seguintes perfis de usuários:

- **2.1. Usuário Não Autenticado:**
  - Acessa a plataforma pela primeira vez.
  - Pode visualizar apenas a página inicial (Home).
  - Pode iniciar os processos de cadastro e login.
  - Não tem acesso a conteúdos protegidos (Desafios, Dashboard, etc.).
- **2.2. Usuário Autenticado (Comum):**
  - Registrado e logado na plataforma.
  - Acesso a todos os módulos de estudo (Desafios Técnicos, Simulador de Entrevistas, Boas Práticas, Dicas de Carreira).
  - Pode visualizar seu Dashboard de progresso pessoal.
  - Suas ações (submissões de desafios, respostas de entrevistas) são persistidas.
- **2.3. Administrador:**
  - Usuário com privilégios administrativos.
  - Acesso a um Painel Administrativo com funcionalidades CRUD para gerenciar conteúdo e usuários.

- Pode monitorar métricas gerais da plataforma.

---

3. Requisitos Funcionais (RFs)

Os requisitos funcionais descrevem o que o sistema *deverá fazer* para atender às necessidades dos usuários e do negócio.

ID	Requisito Funcional	Prioridade	Perfil(s)
RF001	**Gestão de Usuários (Autenticação e Autorização):**		
RF001.1	O sistema deve permitir o **cadastro de novos usuários**, incluindo nome, e-mail, senha e confirmação de senha, com persistência em banco de dados.	Altíssima	Não Autenticado
RF001.2	O sistema deve permitir o **login de usuários existentes** através de e-mail/usuário e senha, com autenticação e geração de token/sessão.	Altíssima	Não Autenticado
RF001.3	O sistema deve permitir que usuários logados realizem **logout**, invalidando sua sessão/token.	Alta	Autenticado, Admin
RF001.4	O sistema deve permitir o **login de administradores** com credenciais específicas, com	Alta	Não Autenticado

	autenticação e autorização para acesso ao painel de administração.		
RF001.5	O sistema deve **restringir o acesso** a todas as funcionalidades de estudo (exceto a Home) a usuários autenticados. Tentativas de acesso não autorizado devem redirecionar para a página de login.	Altíssima	Todos
RF001.6	O sistema deve **restringir o acesso** ao Painel Administrativo apenas a usuários com perfil de Administrador.	Altíssima	Todos
RF001.7	O sistema deve permitir a **recuperação de senha** para usuários (via e-mail, por exemplo).	Média	Autenticado
RF002	**Navegação e Interface Geral:**		
RF002.1	O sistema deve apresentar uma **página inicial (Home)** atrativa e informativa, acessível a todos os usuários.	Altíssima	Todos
RF002.2	O sistema deve	Altíssima	Todos

	implementar uma **navegação SPA**, com transições suaves entre as seções sem recarregar a página.		
RF002.3	O sistema deve exibir um **header dinâmico** que reflita o status de autenticação (Não Logado, Usuário Comum Logado, Administrador Logado) e o perfil do usuário.	Altíssima	Todos
RF002.4	O sistema deve fornecer um **menu hamburguer** para navegação em dispositivos móveis.	Alta	Todos
RF003	**Módulo de Desafios Técnicos:**		
RF003.1	O sistema deve exibir uma **lista paginada de desafios técnicos**, com filtros por dificuldade, linguagem e tags, e ordenação (RF011 do protótipo agora com backend real).	Altíssima	Autenticado
RF003.2	O sistema deve permitir que o usuário selecione um desafio e visualize seu **enunciado completo, exemplos de entrada/saída e um editor de código**	Altíssima	Autenticado

	(Ace Editor).		
RF003.3	O sistema deve permitir que o usuário **submeta seu código** para avaliação.	Altíssima	Autenticado
RF003.4	O sistema deve **executar o código submetido** em um ambiente seguro (sandboxed) e **validar sua correção** contra casos de teste pré-definidos (RF013 do protótipo agora real).	Altíssima	Autenticado
RF003.5	O sistema deve apresentar o **resultado da submissão** (passou/falhou nos testes, tempo de execução, uso de memória, erros de compilação/execução).	Altíssima	Autenticado
RF003.6	O sistema deve **persistir as submissões** dos usuários (código, resultado, data) para histórico.	Alta	Autenticado
RF003.7	O sistema deve exibir um **temporizador** para desafios com limite de tempo, com controle de backend.	Alta	Autenticado

RF004	**Módulo de Simulador de Entrevistas Comportamentais:**		
RF004.1	O sistema deve apresentar **perguntas de entrevista comportamental** armazenadas em banco de dados, de forma sequencial ou aleatória.	Alta	Autenticado
RF004.2	O sistema deve permitir ao usuário **registrar sua resposta** em um textarea e salvar para revisão.	Alta	Autenticado
RF004.3	O sistema deve fornecer **feedback inteligente** sobre a resposta, usando análise de texto (NLP básica ou integrações) para sugestões de melhoria (RF017 agora inteligente).	Média	Autenticado
RF004.4	O sistema deve **persistir as respostas** do usuário para revisão futura.	Alta	Autenticado
RF005	**Módulo de Boas Práticas de Código:**		
RF005.1	O sistema deve exibir conteúdo sobre **boas práticas de código**,	Alta	Autenticado

	categorizado por tópico, com explicações e exemplos de código formatados.		
RF005.2	O sistema deve permitir a **pesquisa e filtragem** de conteúdo sobre boas práticas.	Média	Autenticado
RF006	**Módulo de Dicas e Artigos de Carreira:**		
RF006.1	O sistema deve exibir uma **listagem paginada de artigos e dicas de carreira**, com filtros por tags e categorias.	Alta	Autenticado
RF006.2	O sistema deve permitir ao usuário visualizar o **conteúdo completo de um artigo**, incluindo texto, imagens e vídeos (se houver).	Alta	Autenticado
RF007	**Dashboard de Progresso do Usuário:**		
RF007.1	O sistema deve apresentar um **dashboard personalizado** com o progresso do usuário, incluindo métricas reais (número de desafios concluídos, taxa de acerto, tempo médio).	Altíssima	Autenticado

RF007.2	O sistema deve exibir **gráficos dinâmicos** (ex: barras, pizza) que representem o desempenho e o progresso do usuário ao longo do tempo ou por categoria.	Alta	Autenticado
RF007.3	O sistema deve exibir o **histórico de submissões** de desafios do usuário.	Alta	Autenticado
RF008	**Painel Administrativo (CRUD Completo):**		
RF008.1	O sistema deve permitir ao administrador **gerenciar (CRUD)** usuários (visualizar, editar perfil, desativar).	Altíssima	Admin
RF008.2	O sistema deve permitir ao administrador **gerenciar (CRUD)** desafios (criar, editar, excluir, adicionar casos de teste, definir dificuldade/linguagem).	Altíssima	Admin
RF008.3	O sistema deve permitir ao administrador **gerenciar (CRUD)** artigos e dicas de carreira (criar, editar, excluir, categorizar).	Altíssima	Admin

RF008.4	O sistema deve permitir ao administrador **gerenciar (CRUD)** perguntas do simulador de entrevistas.	Alta	Admin
RF008.5	O sistema deve exibir **métricas e relatórios gerais** da plataforma (número de usuários, desafios ativos, submissões diárias).	Alta	Admin

---

4. Requisitos Não-Funcionais (RNFs)

Os requisitos não-funcionais descrevem as qualidades e restrições que o sistema deverá possuir.

ID	Requisito Não-Funcional	Prioridade
RNF001	**Usabilidade:** A interface do usuário deve ser intuitiva, consistente e fácil de usar, com um fluxo de trabalho lógico que minimize a curva de aprendizado.	Altíssima
RNF002	**Performance:**	
RNF002.1	O tempo de carregamento inicial da página (Home) não deve exceder 3 segundos em uma conexão banda larga padrão.	Alta
RNF002.2	As transições entre as seções SPA devem ser suaves e responsivas, não excedendo 500ms.	Alta

RNF002.3	As requisições de API (login, submissão de desafio, carregamento de lista) devem ter um tempo de resposta médio inferior a 500ms, excluindo o tempo de execução de código para desafios.	Alta
RNF002.4	O módulo de execução de código para desafios deve retornar o resultado da avaliação em no máximo 5 segundos para códigos simples (excluindo tempo de compilação/teste de grandes volumes de dados).	Alta
RNF003	**Segurança:**	
RNF003.1	As senhas dos usuários devem ser armazenadas de forma segura (hash e salt).	Altíssima
RNF003.2	A comunicação entre frontend e backend deve ser segura (HTTPS/SSL).	Altíssima
RNF003.3	O sistema deve implementar autenticação baseada em tokens (JWT) ou sessões seguras.	Altíssima
RNF003.4	Deve haver validação de entrada de dados (input validation) rigorosa no frontend e, principalmente, no backend para prevenir ataques (ex: SQL Injection, XSS).	Altíssima
RNF003.5	O ambiente de execução de código dos desafios deve ser isolado (sandboxed) para	Altíssima

	prevenir acessos indevidos ao sistema principal.	
RNF004	**Confiabilidade:**	
RNF004.1	O sistema deve ter um tempo de atividade (uptime) mínimo de 95% (excluindo manutenções planejadas).	Média
RNF004.2	O sistema deve exibir mensagens de erro claras e informativas ao usuário em caso de falhas na operação.	Alta
RNF04.3	Dados de usuário e conteúdo devem ser persistidos de forma segura e com mecanismos de backup (implementação de backup não faz parte do escopo, mas o sistema deve suportá-lo).	Alta
RNF005	**Manutenibilidade:**	
RNF005.1	O código-fonte deve ser bem documentado, modular e seguir padrões de código (ex: convenções de nomenclatura, estilo).	Altíssima
RNF005.2	A arquitetura da aplicação (frontend, backend, banco de dados) deve ser clara e compreensível para facilitar futuras manutenções e extensões.	Altíssima
RNF006	**Compatibilidade:**	
RNF006.1	O sistema deve ser compatível com os navegadores web mais recentes (Chrome,	Alta

	Firefox, Edge, Safari) e suas duas últimas versões estáveis.	
RNF006.2	O design deve ser totalmente responsivo, adaptando-se a diferentes tamanhos de tela (desktop, tablet, mobile) sem perda de funcionalidade ou usabilidade.	Altíssima
RNF007	**Estética e UI/UX:**	
RNF007.1	O design visual deve ser **limpo, moderno e profissional**, com inspiração clara na **estética da Microsoft** (clareza, uso de espaçamento, tipografia legível, componentes bem definidos).	Altíssima
RNF007.2	A paleta de cores deve ser baseada em **azul intenso (#2980B9) e roxo vibrante (#8E44AD)** como cores de destaque, com neutros (#FFFFFF, #F7F8FC, #2C3E50, #7F8C8D, #DDE3EC) e cores de feedback (#27AE60, #C0392B, #F39C12).	Altíssima
RNF007.3	A tipografia deve utilizar 'Inter' ou 'Segoe UI' para textos gerais e 'Fira Code' ou 'Consolas' para blocos de código.	Alta
RNF007.4	O sistema deve incorporar **imagens e/ou vídeos de fundo** de forma estratégica e otimizada (Header, Home, Resolução de Desafios, Dicas de Carreira, Dashboard),	Alta

	adicionando profundidade visual sem prejudicar a legibilidade do conteúdo.	

---

5. Requisitos Técnicos

Estes requisitos detalham as tecnologias, arquitetura e ferramentas a serem utilizadas no desenvolvimento da plataforma Adytum.

- **5.1. Arquitetura da Aplicação:**
  - **Frontend (SPA):** Cliente-side renderizado, comunicando-se com o backend via API RESTful.
  - **Backend:** API RESTful que serve dados para o frontend e gerencia a lógica de negócio e persistência.
  - **Banco de Dados:** Relacional ou Não-Relacional para armazenamento de dados da aplicação.
  - **Módulo de Execução de Código:** Serviço separado ou componente do backend para executar e testar código de usuário em ambiente isolado.
- **5.2. Linguagens de Programação:**
  - **Frontend:** HTML5, CSS3, JavaScript (Vanilla ou um framework JS leve, a ser definido na fase de projeto detalhado).
  - **Backend:** [Sugestão: Python com Flask/Django, Node.js com Express, Java com Spring Boot, etc. **Escolha a mais apropriada para o seu TCC**].
- **5.3. Banco de Dados:**
  - [Sugestão: PostgreSQL, MySQL, MongoDB, etc. **Escolha a mais apropriada para o seu TCC**].
  - Uso de um ORM (Object-Relational Mapper) para interação entre o backend e o banco de dados.
- **5.4. Bibliotecas/Frameworks (Frontend):**
  - **Ace Editor:** Para o editor de código.
  - **Font Awesome (CDN):** Para ícones.
  - **Google Fonts (CDN):** Para tipografia.
  - **[Opcional]:** Biblioteca de gráficos para o Dashboard (ex: Chart.js, D3.js).
- **5.5. Bibliotecas/Frameworks (Backend):**
  - [Framework do backend escolhido (ex: Flask, Express, Spring Boot)].
  - Biblioteca para hashing de senhas.
  - Biblioteca para gerenciamento de JWT/sessões.
  - [Opcional]: Biblioteca para validação de entrada de dados.
- **5.6. Ambiente de Execução de Código:**
  - Utilização de contêineres (ex: Docker) ou outras tecnologias de sandbox para

isolar a execução do código do usuário.
    - Suporte a múltiplas linguagens de programação (ex: Python, JavaScript, Java) para execução de desafios.
- **5.7. Ferramentas de Desenvolvimento:**
    - Git para controle de versão.
    - Ferramentas de linha de comando (CLI) do Node.js, Python, ou Java (dependendo da escolha do backend).
    - Um IDE (ex: Visual Studio Code, IntelliJ IDEA).
- **5.8. Implantação (Deployment):**
    - Plataforma de hospedagem web (ex: Heroku, AWS EC2, Google Cloud Run, Vercel/Netlify para frontend estático + backend separado).

---

6. Regras de Negócio

Estas regras definem o comportamento e as restrições inerentes à lógica da plataforma.

- **RN001:** Usuários devem ter um e-mail único para cadastro.
- **RN002:** Senhas devem ter requisitos mínimos de complexidade (ex: 8 caracteres, maiúsculas, minúsculas, números, símbolos).
- **RN003:** Apenas um administrador pode criar ou editar desafios, artigos e perguntas de entrevista.
- **RN004:** O sistema deve registrar a data e hora de cada submissão de código de desafio.
- **RN005:** Um desafio é considerado "concluído" apenas se todas as submissões passarem em todos os casos de teste.
- **RN006:** A pontuação do usuário no Dashboard é calculada com base nos desafios concluídos e no desempenho das submissões.
- **RN007:** A funcionalidade de recuperação de senha deve incluir um token de uso único e tempo limitado.
