**Projeto Arquitetural - Edu Planner**

**Versão:** 1.0  
**Data:** 30/06/2025  
**Autores:** Pedro Dylan Fernandes, Felipe Haiashida, Isaac Alves, Rafael Cardoso, Davi Bertasone Kirisawa, João Pedro Ibiapina, Ryan Macedo, Louise Sampaio  
**Professora:** Fabiana Gomes Marinho

### **1. Introdução**

Este documento apresenta a arquitetura do Edu Planner, um sistema de gerenciamento de estudos focado em produtividade e gamificação. Define tecnologias, componentes, fluxos e boas práticas que orientarão seu desenvolvimento e manutenção.

### **2. Requisitos Arquiteturais**

#### 2.1 Requisitos Funcionais (Resumo)

* Gerenciar tarefas com datas, prioridades e dificuldades
* Executar ciclos Pomodoro
* Aplicar sistema de níveis e recompensas
* Exibir calendário integrado
* Permitir customização de perfil

#### 2.2 Requisitos Não Funcionais

* Performance: tempo de resposta < 2s em 90% das requisições
* Escalabilidade: modularização para futuras expansões
* Usabilidade: foco em mobile, acessível WCAG 2.1 AA
* Segurança: JWT stateless, criptografia AES local

### **3. Visão Arquitetural Geral**

* **Estilo:** Modular Monolítico, com separação clara por camadas
* **Frontend:** React Native (mobile first)
* **Backend:** Node.js com NestJS (Domain, Infra, App, Interface)
* **Banco de Dados:** SQLite local (armazenamento offline)
* **Autenticação:** JWT stateless com MFA opcional

Padrões utilizados: - Repository Pattern - DTO (Data Transfer Object) - MVC parcial (em camadas) - Camada de serviços assíncronos com BullMQ

### **4. Componentes e Módulos**

| Componente | Responsabilidade | Tecnologia |
| --- | --- | --- |
| Auth | Login, refresh JWT, MFA | NestJS + JWT |
| Task Manager | CRUD de tarefas, calendário, prazos | NestJS + SQLite |
| Pomodoro Engine | Ciclo de tempo, controle de sessões | Node.js Worker |
| Gamification | Sistema de níveis, pontos e loja de itens | Node.js |
| Profile Manager | Gerenciamento e personalização de perfil | NestJS |
| Frontend App | Interface mobile do usuário | React Native |

### **5. Modelagem**

#### 5.1 Diagrama de Componentes

*(Inserir diagrama no estilo C4 Model)*

#### 5.2 Diagrama de Fluxo

* Fluxo de Login
* Fluxo de Criação de Tarefa
* Fluxo do Pomodoro e atribuição de pontos

### **6. Fluxos Arquiteturais Importantes**

* Login JWT: Stateless, com refresh token de 15 dias
* Ciclo Pomodoro: Controle assíncrono, independente da interface
* Gamificação: Pontuação após tarefas e ciclos concluídos
* Personalização: Leitura e aplicação dos temas e avatares salvos

### **7. Decisões Arquiteturais**

* **Stateless JWT:** evita dependência de sessão
* **Banco local:** prioriza acesso offline
* **NestJS:** modularização robusta e TypeScript nativo
* **React Native:** maior alcance entre Android e iOS

### **8. Qualidade Arquitetural**

| Qualidade | Estratégia adotada |
| --- | --- |
| Performance | Processamento local, fluxos assíncronos |
| Segurança | JWT, criptografia AES, HTTPS em APIs externas |
| Usabilidade | Interface intuitiva e acessível |
| Escalabilidade | Componentização modular para futura nuvem |

### **9. Riscos Arquiteturais e Mitigações**

| Risco | Probabilidade | Impacto | Mitigação |
| --- | --- | --- | --- |
| Problemas de sincronização | Média | Médio | Estrutura de eventos e logs locais |
| Lentidão em dispositivos fracos | Alta | Alto | Processos leves e otimização visual |
| Crescimento não planejado | Média | Médio | Modularização para escalabilidade |

### **10. Evolução da Arquitetura**

* Integração futura com Google/Firebase Auth
* Sincronização cloud para múltiplos dispositivos
* Possível migração do banco para PostgreSQL + Prisma
* Ampliação do sistema de notificações

**Fim do Projeto Arquitetural - Edu Planner**