UNIVERSIDADE DE FORTALEZA
VICE REITORIA DE GRADUAÇÃO
CENTRO DE CIÊNCIAS TECNOLÓGICAS
TECNÓLOGO EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

NOME DA EQUIPE

DOCUMENTAÇÃO TÉCNICA

App Ponto Estagiário

FORTALEZA
2025

NOME DA EQUIPE

DOCUMENTAÇÃO TÉCNICA

App Ponto Estagiário

Este documento contém a documentação técnica do Nome do Projeto desenvolvido na componente curricular T197 - Desenv Plataformas Móveis como requisito para obtenção de nota.

Supervisora: Lyndainês Santos, Esp.

FORTALEZA

2025

SUMÁRIO

1. INTRODUÇÃO

1.1. CONTEXTO E JUSTIFICATIVA

Estagiários, em diferentes áreas, precisam cumprir uma carga horaria estabelecida em contrato. Contudo, muitas vezes, o registro de frequência ainda é feito de forma manual ou em sistema pouco acessíveis. Esse processo pode gerar falhas no acompanhamento das horas, dificuldades na prestação de contas e até prejuízos para o aluno e para a instituição.

O app Ponto Estagiário surge como uma solução simples e prática, focada exclusivamente no público de estagiários, permitido registrar entrada e saída de forma rápida, visualizar o histórico e gerar relatórios básicos de horas cumpridas.

A importância do projeto está em proporcionar organização, transparência e autonomia aos estagiários, reduzindo erros no controle de frequência e facilitando a comunicação com supervisores.

1.2. OBJETIVOS

Objetivo Geral

Desenvolver um aplicativo mobile que registre e acompanhe a frequência dos estagiários, possibilitando maior controle sobre entrada, saída e carga horaria.

Objetivo Específicos:

- *Permitir cadastro e autenticação de usuários (estagiários).*
- *Registrar pontos de entrada e saída com data e hora.*
- *Exibir histórico de registro e total de horas diárias e semanais.*
- *Gerar relatórios básicos de frequência para acompanhamento.*
- *Oferecer interface simples e intuitiva, com cores de status para indicar situação do ponto.*
- *Possibilitar recuperação de senha em caso de esquecimento.*

1.3. ESCOPO E DELIMITAÇÃO

- ***Escopo:***

- *Cadastro e login de estagiários.*
- *Tela de informações do usarios.*
- *Registro do ponto (entrada/saída).*
- *Historico de registro de ponto.*
- *Calculo de carga horaria diária e semanal.*
- *Relatorios básicos em formato visual e exportável.*
- *Tela de recuperação de senha.*
- *Notificações via ícones no app sobre status de ponto.*

- ***Delimitação (Fora do Escopo):***
  - *O sistema não terá modulo de geolocalização.*
  - *Não sera implementado controle de múltiplos perfis (focado apenas nos estagiários)*
  - Não haverá integração com sistema de rh ou folha de pagamento.
  - Não sera implementado modulo financeiro ou de gestão de contratos.
  - Não incluira autenticação biométrica na versão inicial.

2. ENGENHARIA DE REQUISITOS

2.1. REQUISITOS FUNCIONAIS (RFs)

ID	Nome do Requisito	Importância	Descrição	Categoria
RF01	Cadastro de usuários	OBRIGATÓRIO	O sistema deve permitir o cadastro de estagiários com e-mail, nome e senha.	Funcional
RF02	Login	OBRIGATÓRIO	O usuário deve poder acessar o app com suas credenciais.	Funcional
RF03	Recuperar senha	OBRIGATÓRIO	O usuário deve poder redefinir senha via e-mail em caso de esquecimento.	Funcional
RF04	Tela de informações	IMPORTANTE	Exibir informações básicas do estagiário cadastrado (nome, e-mail).	Funcional
RF05	Registrar ponto (entrada/saída)	OBRIGATÓRIO	O sistema deve permitir registrar horários de entrada e saída.	Funcional
RF06	Histórico de ponto	OBRIGATÓRIO	O app deve exibir os pontos registrados em ordem cronológica.	Funcional
RF07	Cálculo de horas	OBRIGATÓRIO	O sistema deve calcular carga horária diária e semanal do estagiário.	Funcional
RF08	Exportação de relatórios	OBRIGATÓRIO	O usuário deve poder exportar relatórios de frequência por período.	Funcional
RF09	Status de ponto (cores)	OBRIGATÓRIO	O sistema deve indicar status com cores (verde = registrado, vermelho = falta, laranja = pendente).	Funcional
RF10	Notificações de ausência	IMPORTANTE	O app deve notificar o estagiário caso não registre entrada/saída.	Funcional
RF11	Justificar ponto	IMPORTANTE	O usuário deve poder justificar faltas ou atrasos.	Funcional

2.2 REQUISITOS NÃO FUNCIONAIS (RNFs)

- **Desempenho**

    - RNF01: O tempo de inicialização do app não deve exceder 3 segundos.
    - RNF02: O carregamento do histórico deve ocorrer em até 1 segundo para 30 registros.
- **Usabilidade**
    - RNF03: Interface deve seguir padrões de design simples e intuitivos, priorizando fácil uso por estagiários.
    - RNF04: O app deve utilizar cores padronizadas para status de ponto.
- **Compatibilidade**
    - RNF05: Funcionar em dispositivos Android a partir da versão 9.0 (API 28).
- **Segurança**
    - RNF06: Senhas devem ser armazenadas de forma segura (hash + salt).
    - RNF07: Recuperação de senha deve ser feita exclusivamente via e-mail.

3. PROJETO E ARQUITETURA DO SOFTWARE

3.1. PROJETO DE DADOS: INTEGRAÇÃO COM API E PERSISTÊNCIA LOCAL

O App Ponto Estagiário foi desenvolvido seguindo a arquitetura cliente-servidor, onde o aplicativo mobile (cliente) se comunica com uma API REST (servidor) para realizar operações como cadastro, login, registro e consulta de pontos.

A arquitetura foi baseada no padrão MVVM (Model-View-ViewModel) para separar responsabilidades:

- Model: Responsável pela estrutura dos dados (usuário, ponto, relatório).
- View: Telas do aplicativo, que exibem os dados ao usuário.
- ViewModel: Camada intermediária que gerencia regras de negócio, comunicação com a API e banco de dados local.

Para garantir uso offline, foi implementada uma camada de persistência local (SQLite via Room ou alternativa), que armazena registros de ponto e histórico. Quando a conexão com a internet é restabelecida, os dados são sincronizados com a API.

3.2 Projeto Banco de Dados

O banco de dados foi projetado para ser simples e otimizado para consultas rápidas. Ele é dividido em duas principais entidades: Usuário e Ponto.

Modelo Entidade-Relacionamento (MER)

Tabela Usuário

Campo	**Tipo de dado**	**PK/FK**	**Nulo?**	**Descrição**
id_usuario	UUID	PK	Não	Identificador único do usuário.
nome	VARCHAR(100)		Não	Nome completo do estagiário.
email	VARCHAR(100)	Único	Não	E-mail usado para login.
senha_hash	VARCHAR(255)		Não	Senha criptografada (hash + salt).
data_cadastro	TIMESTAMP		Não	Data de criação do usuário.

Tabela Ponto

Campo	Tipo de dado	PK/FK	Nulo?	Descrição
id_ponto	UUID	PK	Não	Identificador único do ponto.
id_usuario	UUID	FK	Não	Referência ao estagiário.
data_registro	DATE		Não	Data do registro.
hora_entrada	TIME		Sim	Horário de entrada.
hora_saida	TIME		Sim	Horário de saída.
status	VARCHAR(20)		Não	Status do ponto (registrado, falta, pendente).
justificativa	TEXT		Sim	Texto explicando falta ou atraso.

3.2. Projeto de API

A API REST é responsável pela comunicação entre o app e o servidor.
As principais rotas previstas são:

Autenticação e Usuário

- *POST /api/usuarios → Cadastro de novo estagiário.*
- *POST /api/login → Login com e-mail e senha.*
- *POST /api/recuperar-senha → Envio de e-mail para redefinir senha.*
- *GET /api/usuarios/{id} → Retorna informações básicas do estagiário.*

Ponto

- *POST /api/pontos → Registrar ponto (entrada ou saída).*
- *GET /api/pontos/{id_usuario} → Listar pontos de um estagiário.*
- *GET /api/pontos/{id_usuario}/relatorio?periodo=semanal → Gerar relatório semanal.*
- *PUT /api/pontos/{id_ponto}/justificativa → Inserir justificativa em caso de falta ou atraso.*

Notificações

- *POST /api/notificacoes → Enviar notificação ao estagiário caso não registre ponto.*

Fluxo de Navegação do Aplicativo

O fluxo de navegação foi projetado para ser simples e intuitivo:

1. ***Splash Screen** → Logo do App (carregamento inicial).*
2. ***Login***
    - *Acesso com e-mail e senha.*
    - *Link para "Recuperar senha".*
    - *Botão para "Criar Conta".*
3. ***Tela Inicial (Home)***
    - *Botão grande para **Entrada**.*
    - *Botão grande para **Saída**.*
    - *Status atual (verde, vermelho, laranja).*
    - *Acesso rápido ao histórico.*
4. ***Histórico de Registros***
    - *Lista de dias com horários registrados.*
    - *Opção de filtrar por semana/mês.*
5. ***Relatórios***
    - *Resumo de horas diárias e semanais.*
    - *Exportação em PDF/Excel.*
6. ***Tela de Informações do Usuário***
    - *Dados básicos (nome, e-mail, data de cadastro).*
7. ***Tela de Recuperação de Senha***
    - *Envio de e-mail para redefinir senha.*

4. TECNOLOGIAS E FERRAMENTAS

4.1. STACK DE TECNOLOGIAS

***Instrução:** Substitua completamente a stack web pelas tecnologias, linguagens e bibliotecas usadas no desenvolvimento do seu aplicativo mobile.*

Exemplo de Texto:

- ***Linguagem:*** *Kotlin, por ser a linguagem oficial para o desenvolvimento Android, oferecendo segurança (null-safety) u concisão.*
- ***Arquitetura:*** *Android Architecture Components (ViewModel, LiveData, StateFlow, Room).*
- ***UI Toolkit:*** *Jetpack Compose, para a construção declarativa e moderna da interface do usuário.*
- ***Injeção de Dependência:*** *Hilt, para simplificar a injeção de dependências no projeto.*
- ***Consumo de API:*** *Retrofit 2 e OkHttp 3, para realizar chamadas de rede à API REST de forma eficiente.*
- ***Banco de Dados Local:*** *Room, para persistência de dados e implementação de cache offline.*
- ***Carregamento de Imagens:*** *Coil, uma biblioteca moderna e performática para carregar imagens da rede.*

4.2. FERRAMENTAS DE DESENVOLVIMENTO

Instrução: *Liste as ferramentas que apoiaram o processo de desenvolvimento, desde a escrita do código até o gerenciamento das tarefas da equipe.*

Exemplo de Texto:

- ***IDE:*** *Visual Studio Code foi a IDE padrão para toda a equipe, devido à sua leveza, extensibilidade e terminal integrado.*
- ***Controle de Versão:*** *Git, com o repositório hospedado no GitHub. Adotamos o fluxo de trabalho "GitFlow", com branches separadas para* `develop`*,* `features` *e* `main`*.*
- ***Gerenciamento de Projeto:*** *Trello foi utilizado para gerenciar as tarefas. Criamos um quadro Kanban com as colunas "A Fazer", "Em Andamento", "Em Teste" e "Concluído".*
- ***Ferramenta de API:*** *Insomnia foi usado para testar os endpoints da API durante o desenvolvimento.*

5. IMPLEMENTAÇÃO E RESULTADOS

5.1. TELAS DO SISTEMA

Instrução: *Esta é a vitrine do seu projeto. Insira imagens (screenshots) das principais telas da sua aplicação. Cada imagem deve ter uma legenda curta e clara explicando sua finalidade. Dê preferência a telas que demonstrem as funcionalidades mais importantes (RFs) que você listou na seção 2.*

Exemplo de Texto:

Figura 1: Tela de Login *(Legenda: A tela de login é o ponto de entrada do sistema, garantindo o acesso seguro através de autenticação por e-mail e senha.)*

[INSERIR SCREENSHOT DA TELA DE LOGIN AQUI]

Figura 2: Dashboard Principal com Visão Geral do Estoque *(Legenda: Após o login, o administrador tem acesso a um dashboard com métricas rápidas sobre o inventário, como o número de peças disponíveis, reservadas e vendidas.)*

[INSERIR SCREENSHOT DO DASHBOARD AQUI]

Figura 3: Tela de Cadastro de Laje de Pedra *(Legenda: Formulário detalhado para o cadastro de uma nova peça no inventário, permitindo o upload de foto e a inserção de todas as especificações técnicas.)*

[INSERIR SCREENSHOT DO FORMULÁRIO DE CADASTRO AQUI]

6. Conclusão

6.1 TRABALHOS FUTUROS

Instrução: *Nenhum projeto está 100% completo. Liste aqui as melhorias e novas funcionalidades que você gostaria de implementar no futuro. Pense em como o aplicativo poderia se tornar ainda mais útil para o usuário. Isso demonstra visão de produto e consciência das limitações do trabalho atual.*

Exemplo de Texto:

Com a base sólida do aplicativo estabelecida, identificamos diversas oportunidades para evoluir e agregar ainda mais valor ao sistema RochaForte no futuro. As principais propostas são:

- **Funcionalidades Offline Avançadas:** Expandir a capacidade offline para permitir não apenas a consulta, mas também a **criação e edição de pedidos** sem conexão com a internet. Os dados seriam sincronizados automaticamente com o servidor assim que uma conexão fosse restabelecida.
- **Identificação de Lajes por QR Code:** Implementar uma funcionalidade que utilize a câmera do dispositivo para ler um QR Code fixado em cada laje de pedra. Isso permitiria ao vendedor ou gerente de estoque acessar instantaneamente os detalhes da peça, eliminando a necessidade de busca manual e agilizando o processo de inventário.
- **Notificações Push em Tempo Real:** Desenvolver um sistema de notificações push para alertar os vendedores sobre eventos importantes, como a confirmação de um pedido, a chegada de um novo lote de material ou uma alteração no status de uma laje que ele esteja monitorando.
- **Otimização para Tablets e Modo Paisagem:** Adaptar a interface do usuário para oferecer uma experiência otimizada em telas maiores, como as de tablets, que são frequentemente utilizados em balcões de vendas. Isso incluiria layouts de duas colunas e melhor aproveitamento do espaço horizontal.
- **Versão para a Plataforma iOS:** Iniciar o desenvolvimento da versão do aplicativo para iOS, a fim de atender a todos os potenciais usuários da empresa, independentemente do sistema operacional de seus dispositivos móveis.