INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO PIAUÍ
CAMPUS PICOS
TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

KARIELLY DE CARVALHO

REGISTRO DE PONTO DIGITAL

PICOS, PIAUÍ
2025

KARIELLY DE CARVALHO

REGISTRO DE PONTO DIGITAL

Trabalho de Conclusão de Curso (Relatório Técnico de Software) apresentado como exigência parcial para obtenção do diploma do Curso de Análise e Desenvolvimento de Sistemas do Instituto Federal de Educação, Ciência e Tecnologia do Piauí, Campus Picos.

Orientador: Prof. M^{e}. Jesiel Viana da Silva

PICOS, PIAUÍ
2025

KARIELLY DE CARVALHO

REGISTRO DE PONTO DIGITAL

Trabalho de Conclusão de Curso (Relatório Técnico de Software) apresentado como exigência parcial para obtenção do diploma do Curso de Análise e Desenvolvimento de Sistemas do Instituto Federal de Educação, Ciência e Tecnologia do Piauí, Campus Picos.

Aprovada em _____/_____/_____.

BANCA EXAMINADORA:

Prof. M^{e}. Jesiel Viana da Silva(Orientador)
Instituto Federal do Piauí (IFPI)

Prof. M^{e}. Aislan Rafael Rodrigues de Sousa
Instituto Federal do Piauí (IFPI)

Prof. M^{e}. João Paulo Lima do Nascimento
Instituto Federal do Piauí (IFPI)

PICOS, PIAUÍ
2025

Dedico este trabalho à minha família, cujo apoio e incentivo tornaram possível esta nova jornada.

AGRADECIMENTOS

Agradeço primeiramente a Deus, por me conceder sabedoria, saúde e força durante toda esta jornada acadêmica.

Ao meu orientador, professor Jesiel Viana, pela dedicação, paciência e ótimas orientações ao longo do desenvolvimento deste trabalho.

À minha família, especialmente aos meus pais, Juvan Luís de Carvalho e Alessandra Rodrigues de Carvalho, por todo amor, apoio incondicional e incentivo desde o início da minha trajetória — sem vocês, nada disso seria possível. À minha irmã, Fernanda Monique de Carvalho, e ao meu irmão, Levi Rodrigues de Carvalho, pelo companheirismo e carinho. À minha sobrinha, Aurora Monique de Carvalho, que, com sua doçura e alegria, iluminou muitos dos meus dias e me deu ainda mais motivação para seguir em frente.

Ao meu namorado, Jean Carlos Rodrigues Sousa, por todo incentivo e parceria nos momentos mais desafiadores desta caminhada. Sua presença foi importante para que eu chegasse até aqui, e sua confiança em mim fez toda a diferença.

A todos que, de alguma forma, contribuíram para a realização deste trabalho, deixo o meu mais sincero agradecimento.

"A tecnologia move o mundo."

— Steve Jobs

LISTA DE TABELAS

Tabela 1 – Requisitos funcionais do sistema de Registro de Ponto 19
Tabela 2 – Requisitos não funcionais do sistema de Registro de Ponto 20
Tabela 3 – Resumo das tecnologias e versões utilizadas no projeto 23

LISTA DE ABREVIATURAS E SIGLAS

ABNT	Associação Brasileira de Normas Técnicas
CLT	Consolidação das Leis do Trabalho
CSS	Cascading Style Sheets (Folhas de Estilo em Cascata)
IFPI	Instituto Federal de Educação, Ciência e Tecnologia do Piauí
MPE	Micro e Pequenas Empresas
ORM	Mapeador Objeto-Relacional
SSR	Server-Side Rendering (Renderização no Lado do Servidor)
API	Interface de Programação de Aplicação

SUMÁRIO

1	**INTRODUÇÃO**	**9**
1.1	Justificativa	10
1.2	Objetivos	10
1.2.1	Geral	10
1.2.2	Específicos	11
1.3	Metodologia	11
1.3.1	Procedimentos de Pesquisa Exploratória	11
1.3.1.1	Pesquisa Bibliográfica	11
1.3.1.2	Análise de Mercado e Soluções Concorrentes	11
1.3.2	Metodologia de Desenvolvimento do Software (MVP)	12
1.3.2.1	Modelo de Processo de Desenvolvimento	12
1.3.2.2	Verificação Técnica e Testes	12
2	**REFERENCIAL TEÓRICO**	**13**
2.1	Controle de Jornada de Trabalho	13
2.1.1	Evolução dos Sistemas de Ponto: Do Manual ao Digital	13
2.1.2	Legislação Brasileira e a Portaria 671/2021	13
2.2	Tecnologias Habilitadoras	14
2.2.1	Geolocalização (GPS) no Controle de Frequência	14
2.2.2	Metodologia de Desenvolvimento de Software: MVP	14
2.3	Arquitetura e Tecnologias Adotadas	15
2.3.1	Arquitetura Frontend com Next.js	15
2.3.2	Arquitetura Backend com NestJS	15
2.3.3	Sistema de Gerenciamento de Banco de Dados: PostgreSQL	15
2.3.4	Segurança e Autenticação	16
2.3.4.1	Autenticação com JSON Web Tokens (JWT)	16
2.3.4.2	Controle de Acesso Baseado em Papéis (RBAC)	16
2.4	Tendências e Futuro da Gestão de Ponto	16
3	**REQUISITOS DO SISTEMA**	**18**
3.1	REQUISITOS FUNCIONAIS	18
3.2	REQUISITOS NÃO FUNCIONAIS	18
4	**TECNOLOGIAS ENVOLVIDAS**	**21**
4.1	Tecnologias Front-End	21
4.1.1	Next.js	21

4.1.2 React.js . . . 21
4.1.3 TypeScript . . . 21
4.1.4 Tailwind CSS . . . 21
4.2 Tecnologias Back-End . . . 22
4.2.1 NestJS . . . 22
4.2.2 Prisma . . . 22
4.2.3 PostgreSQL . . . 22
4.3 Resumo das Tecnologias e Versões . . . 22

5 ARQUITETURA E MODELAGEM DO SISTEMA . . . 24
5.1 Visão Geral da Arquitetura . . . 24
5.2 Modelagem de Dados . . . 25
5.2.1 Entidades Principais . . . 25
5.2.2 Relacionamentos e Cardinalidades . . . 26
5.3 Arquitetura do Back-End . . . 27
5.3.1 Estrutura Modular . . . 27
5.3.2 Camadas Arquiteturais . . . 28
5.3.3 Principais Funcionalidades . . . 28
5.4 Arquitetura do Front-End . . . 28
5.4.1 Estrutura de Componentes . . . 29
5.4.2 Padrões Arquiteturais Implementados . . . 29
5.4.3 Gerenciamento de Estado . . . 30
5.5 Fluxo de Autenticação e Autorização . . . 30
5.5.1 Componentes de Segurança . . . 30
5.5.2 Validação de Geolocalização . . . 31
5.6 Integração com Serviços Externos . . . 31
5.6.1 Google Maps API . . . 31
5.6.2 Geolocation API . . . 32
5.7 Considerações de Performance e Escalabilidade . . . 32

Referências . . . 34

1 INTRODUÇÃO

A gestão da jornada de trabalho é um elemento crucial para o bom funcionamento das organizações, pois influencia diretamente tanto a conformidade com a legislação quanto a eficiência operacional. No Brasil, o artigo 74 da Consolidação das Leis do Trabalho (CLT)[1] determina que empresas com mais de 20 colaboradores devem manter o registro de ponto de seus funcionários (Brasil, 1943). No entanto, micro e pequenas empresas (MPEs) enfrentam desafios significativos nesse aspecto, especialmente devido ao alto custo de sistemas eletrônicos de controle de ponto. Como alternativa, muitas recorrem a métodos manuais, considerados mais acessíveis inicialmente, mas que acabam aumentando os riscos de imprecisões, perdas de dados e até fraudes no acompanhamento das horas trabalhadas, comprometendo a eficácia na gestão do tempo (Miranda, 2023). A adoção de sistemas digitais de registro de ponto oferece benefícios importantes. Essas soluções eliminam a necessidade de equipamentos físicos específicos, permitindo o uso de dispositivos já disponíveis, como tablets, smartphones ou computadores. Dessa forma, tornam-se opções mais práticas e econômicas para a gestão eficiente do banco de horas (Florindo; Bianchi, 2022).

De acordo com Gomes (2023), o registro de ponto digital tem se consolidado como uma solução eficaz para empresas que buscam otimizar a gestão de suas equipes sem comprometer o orçamento. Esse modelo proporciona maior precisão e transparência no acompanhamento das horas trabalhadas, além de reduzir os custos operacionais. Em contrapartida, a permanência no uso de métodos manuais pode levar a falhas significativas na supervisão da jornada de trabalho, prejudicando tanto empregadores quanto empregados. Um monitoramento inadequado impacta negativamente o cumprimento das obrigações legais e a transparência necessária para garantir a confiança mútua entre as partes (Abreu, 2016).

Este projeto tem como objetivo e desenvolver uma solução digital acessível e eficiente. Para isso, será desenvolvido um sistema de registro de ponto digital que utiliza tecnologias como escaneamento de QR Code e geolocalização. Essa abordagem busca otimizar o controle das horas trabalhadas, proporcionando maior precisão, autonomia aos funcionários e conformidade com as exigências legais, além de reduzir custos e riscos associados aos métodos manuais.

Segundo Gomes (2023), a adoção de sistemas digitais para o registro de ponto oferece uma alternativa prática e econômica para empresas com recursos limitados, enquanto Mariotti e Kienetz (2011) enfatizam que esses sistemas desempenham um papel essencial na melhoria dos processos internos e no cumprimento das normas trabalhistas. Assim, os sistemas digitais não apenas aumentam a eficiência, mas

[1] https://www.planalto.gov.br/ccivil_03/decreto-lei/del5452.htm

também promovem a transparência e a segurança jurídica para empregadores e funcionários.

1.1 JUSTIFICATIVA

Este projeto visa aprimorar o gerenciamento e controle das horas trabalhadas, beneficiando tanto empresas quanto funcionários, ao modernizar e tornar mais seguro o processo de gestão de pessoas por meio de soluções tecnológicas. Muitas micro e pequenas empresas (MPEs) ainda enfrentam dificuldades para substituir registros manuais por sistemas digitais. Métodos tradicionais, como planilhas e livros de ponto, são suscetíveis a erros humanos e não garantem a segurança necessária, podendo resultar em inconsistências no controle de jornada e problemas trabalhistas (Florindo; Bianchi, 2022).

Diante desse cenário, este estudo se torna importante para analisar e desenvolver uma solução digital acessível, segura e eficiente. Além de atender às exigências legais, um sistema digital pode reduzir custos operacionais, otimizar o controle de jornada e fortalecer a transparência nas relações de trabalho. Com isso, há potencial para impactos positivos significativos, como aumento da produtividade, redução de erros manuais e maior satisfação dos colaboradores (Gomes, 2023).

De acordo com Longo e Watanabe (2019), a transformação digital tem impulsionado mudanças rápidas e contínuas nas organizações, reformulando produtos, serviços e processos internos. Essas mudanças afetam diretamente as relações de trabalho e tornam indispensável a adoção de soluções inovadoras para acompanhar a evolução do mercado. Assim, este estudo se justifica pela necessidade de desenvolver uma ferramenta digital que atenda a essas novas demandas, promovendo benefícios para empregadores e funcionários, além de contribuir para a modernização e eficiência da gestão empresarial.

1.2 OBJETIVOS

1.2.1 Geral

Desenvolver um MVP (Minimum Viable Product) de sistema web de Registro de Ponto com geolocalização para pequenas e médias empresas, permitindo que funcionários registrem suas entradas e saídas através de smartphones com validação de localização, oferecendo dashboard gerencial para acompanhamento de bancos de horas, gestão de justificativas e relatórios automatizados, visando proporcionar maior controle, transparência e eficiência no controle de ponto.

1.2.2 Específicos

- Analisar as principais ferramentas digitais de registro de ponto disponíveis no mercado, identificando suas funcionalidades, limitações e requisitos para garantir um controle eficiente da jornada de trabalho.
- Identificar as principais dificuldades enfrentadas por empresas e funcionários no controle de ponto.
- Desenvolver a arquitetura do MVP utilizando Next.js e NestJS, implementando o sistema de autenticação JWT e os perfis de acesso (funcionário e gerente).
- Implementar o módulo de registro de ponto para o funcionário, com validação de geolocalização por raio configurável via smartphone.
- Construir o painel gerencial, englobando a gestão de funcionários, o fluxo de aprovação de justificativas, o cálculo automático de banco de horas e a emissão de relatórios.

1.3 METODOLOGIA

A metodologia adotada para este projeto combinou uma pesquisa exploratória e qualitativa com uma abordagem ágil de engenharia de software, estruturando o trabalho em duas frentes principais e complementares: a primeira, focada no embasamento teórico e na análise de mercado para a fundamentação do projeto, e a segunda, na construção prática e iterativa do Mínimo Produto Viável (MVP).

1.3.1 Procedimentos de Pesquisa Exploratória

1.3.1.1 Pesquisa Bibliográfica

A pesquisa bibliográfica foi fundamental para compreender os conceitos fundamentais de controle de ponto, legislação trabalhista brasileira e tecnologias de geolocalização. Foram analisados artigos científicos, livros técnicos e documentações oficiais sobre sistemas de registro de ponto, com foco especial na Consolidação das Leis do Trabalho (CLT) e suas exigências para empresas com mais de 20 funcionários. A pesquisa também abrangeu estudos sobre usabilidade em sistemas móveis e boas práticas de desenvolvimento web.

1.3.1.2 Análise de Mercado e Soluções Concorrentes

Foi realizada uma análise exploratória das principais ferramentas digitais de registro de ponto disponíveis no mercado brasileiro, identificando suas funcionalidades, limitações e requisitos para garantir um controle eficiente da jornada de trabalho. Esta

pesquisa permitiu identificar lacunas no mercado, especialmente para pequenas e médias empresas que não possuem recursos para sistemas complexos e caros. A análise incluiu sistemas como PontoTel, Tangerino, PontoMais e outros, mapeando suas características técnicas, custos e adequação para diferentes portes de empresa.

1.3.2 Metodologia de Desenvolvimento do Software (MVP)

1.3.2.1 Modelo de Processo de Desenvolvimento

O desenvolvimento do MVP seguiu uma abordagem ágil com prototipagem evolutiva, utilizando a metodologia Frontend-First — uma estratégia que prioriza a construção da experiência do usuário, desenvolvendo inicialmente todas as interfaces com dados estáticos (mockados) utilizando Next.js com TypeScript e componentes Radix UI. O backend foi desenvolvido posteriormente com NestJS, construindo a API de forma direcionada para atender às necessidades já estabelecidas no frontend.

A arquitetura foi desenvolvida seguindo princípios de separação de responsabilidades, com frontend e backend como serviços independentes, utilizando PostgreSQL como banco de dados principal e implementando autenticação JWT para controle de acesso diferenciado entre funcionários e gestores.

1.3.2.2 Verificação Técnica e Testes

Para a verificação do MVP, considerando o estágio de desenvolvimento, foi implementada uma abordagem de verificação interna em duas frentes. A primeira consistiu na Verificação de Requisitos Funcionais, onde cada funcionalidade especificada foi testada sistematicamente para garantir seu correto funcionamento. Os principais módulos verificados foram:

- Registro de ponto com validação por geolocalização;
- Gestão de funcionários, perfis e departamentos;
- Fluxo de envio e aprovação de justificativas;
- Cálculo automático e exibição do banco de horas.

A segunda frente foi uma Avaliação Heurística da interface do usuário, baseada nas 10 Heurísticas de Usabilidade de Jakob Nielsen. Esta análise focou nos principais fluxos de interação, com atenção especial à usabilidade em dispositivos móveis, considerando que o registro de ponto é realizado via smartphone. A avaliação identificou potenciais problemas de usabilidade e garantiu que o MVP adere a boas práticas de design, como consistência visual, feedback ao usuário e prevenção de erros.

2 REFERENCIAL TEÓRICO

2.1 CONTROLE DE JORNADA DE TRABALHO

O controle da jornada de trabalho é um pilar fundamental na relação entre empregadores e empregados, sendo essencial para garantir a conformidade com a legislação trabalhista, assegurar a remuneração correta das horas trabalhadas e proteger os direitos de ambas as partes com transparência (Seguros, 2024). A evolução dos métodos de controle reflete diretamente o avanço tecnológico e as mudanças nas dinâmicas de trabalho.

2.1.1 Evolução dos Sistemas de Ponto: Do Manual ao Digital

Historicamente, o controle de jornada era realizado por meios manuais, como o livro de ponto ou o relógio cartográfico. Tais métodos, embora simples, são extremamente vulneráveis a fraudes, erros de preenchimento e rasuras, gerando insegurança jurídica e administrativa. A manipulação intencional ou o simples erro humano no registro manual podem resultar em pagamentos incorretos de horas extras, adicional noturno e outras verbas, culminando em prejuízos financeiros e um aumento significativo no risco de ações trabalhistas (forpeople, 2023).

A transição para sistemas digitais representa um avanço substancial em segurança, precisão e eficiência. Ao automatizar a coleta e o cálculo das horas, os sistemas eletrônicos minimizam a ocorrência de erros e fraudes, além de fornecerem dados em tempo real para a gestão de equipes, otimizando processos do departamento de Recursos Humanos (RH) e reduzindo custos operacionais (Geovictoria, 2024).

2.1.2 Legislação Brasileira e a Portaria 671/2021

A Consolidação das Leis do Trabalho (CLT), em seu Artigo 74, estabelece a obrigatoriedade do controle de jornada para empresas com mais de 20 funcionários. A regulamentação dos sistemas eletrônicos de ponto, por sua vez, foi modernizada pela Portaria 671 do Ministério do Trabalho e Previdência (MTP), de novembro de 2021, que unificou e substituiu as portarias anteriores (1510 e 373), simplificando as regras e introduzindo novas modalidades de registro (Cardoso, 2024).

A portaria define três tipos principais de Registradores Eletrônicos de Ponto (REP):

- **REP-C (Convencional):** O relógio de ponto tradicional, físico, que emite comprovantes impressos.

- **REP-A (Alternativo):** Sistemas e aplicativos online que permitem o registro via computador ou dispositivos móveis, validados por Convenção ou Acordo Coletivo de Trabalho.
- **REP-P (Programa):** A categoria mais moderna, que engloba softwares e sistemas em nuvem para registro de ponto, incluindo coletores de marcações, armazenamento de dados e o programa de tratamento de ponto. Esta modalidade exige a emissão de comprovante de registro de ponto por meio eletrônico ou impresso e a geração do Arquivo Fonte de Dados (AFD) conforme padrões legais (TOTVS, 2024).

O sistema desenvolvido neste projeto, que utiliza um aplicativo de smartphone com geolocalização, enquadra-se na categoria **REP-P**, por se tratar de um programa de computador que executa o registro e o tratamento dos dados de jornada de forma digital e segura.

2.2 TECNOLOGIAS HABILITADORAS

2.2.1 Geolocalização (GPS) no Controle de Frequência

A tecnologia de geolocalização, popularizada pelo Sistema de Posicionamento Global (GPS), tornou-se uma ferramenta estratégica para empresas com equipes externas, em regime de home office ou em locais de trabalho variáveis. Ao integrar o GPS a um aplicativo de ponto, o sistema captura as coordenadas geográficas (latitude e longitude) do colaborador no exato momento da marcação (Senior Sistemas, 2023).

Essa funcionalidade oferece um nível adicional de segurança e transparência, permitindo ao gestor verificar se o registro foi realizado dentro de um perímetro pré-autorizado, como a sede da empresa ou o local de um cliente. Isso inibe fraudes comuns, como o buddy punching — prática na qual um colega registra o ponto pelo outro — e fornece um respaldo jurídico robusto para o empregador (Senior Sistemas, 2023).

Sob a ótica da Lei Geral de Proteção de Dados (LGPD), o uso da geolocalização para controle de ponto é legal, desde que o tratamento dos dados esteja estritamente ligado à finalidade de controle da jornada de trabalho. É imperativo que o colaborador seja informado de maneira clara sobre a coleta desses dados e que a empresa colete apenas as informações estritamente necessárias, garantindo a privacidade do funcionário fora do horário de trabalho (Factorial, 2024).

2.2.2 Metodologia de Desenvolvimento de Software: MVP

A abordagem de desenvolvimento de um Produto Mínimo Viável (MVP, do inglês *Minimum Viable Product*) foi adotada neste projeto. A estratégia consiste em

construir uma versão inicial do sistema contendo apenas as funcionalidades essenciais para resolver o problema central do público-alvo (Corrales, 2024). O objetivo é lançar o produto rapidamente no mercado para coletar feedback real de usuários e, a partir daí, orientar as próximas fases do desenvolvimento de forma iterativa.

Os principais benefícios dessa metodologia incluem a redução de riscos e custos, a aceleração do ciclo de aprendizado da equipe e a garantia de que o produto final esteja verdadeiramente alinhado às necessidades do mercado. Empresas como Dropbox, Foursquare e Zappos são exemplos clássicos de sucesso que iniciaram suas trajetórias com um MVP para validar suas propostas de valor antes de investir em um desenvolvimento em larga escala (SoftKraft, 2024).

2.3 ARQUITETURA E TECNOLOGIAS ADOTADAS

2.3.1 Arquitetura Frontend com Next.js

Para a construção da interface do usuário (*frontend*), optou-se pelo framework Next.js. Em comparação com outras bibliotecas e frameworks populares como Vue.js ou Angular, o Next.js se destaca por sua arquitetura híbrida que otimiza a performance e a experiência do desenvolvedor (Shah, 2024). Seus recursos de Renderização no Lado do Servidor (SSR - *Server-Side Rendering*) e Geração de Site Estático (SSG - *Static Site Generation*) são cruciais para a performance da aplicação, resultando em tempos de carregamento mais rápidos e melhor indexação por mecanismos de busca (SEO) (Strapi, 2024).

2.3.2 Arquitetura Backend com NestJS

No desenvolvimento do *backend*, a escolha foi o NestJS, um framework Node.js progressivo. Diferente de alternativas mais flexíveis e menos estruturadas como o Express.js, o NestJS adota uma arquitetura opinativa e modular, fortemente inspirada no Angular. Essa estrutura organizada facilita a escalabilidade, a manutenção e a testabilidade do código, tornando-o ideal para aplicações corporativas robustas e complexas. A modularidade do NestJS permite que a aplicação seja dividida em componentes independentes e reutilizáveis, promovendo um código mais limpo e organizado (Dev&Deliver, 2024).

2.3.3 Sistema de Gerenciamento de Banco de Dados: PostgreSQL

A persistência dos dados é gerenciada pelo PostgreSQL, um sistema de gerenciamento de banco de dados relacional (SGBDR) de código aberto. Para uma aplicação de controle de ponto, onde a integridade e a consistência dos dados são críticas (registros de ponto, informações de funcionários, etc.), um banco de dados

SQL como o PostgreSQL é superior a alternativas NoSQL como o MongoDB (Amazon Web Services, 2024a). O PostgreSQL garante a conformidade com os princípios ACID (Atomicidade, Consistência, Isolamento e Durabilidade) e oferece tipos de dados avançados, que são fundamentais para manter a confiabilidade e a precisão das informações trabalhistas (Amazon Web Services, 2024b).

2.3.4 Segurança e Autenticação

2.3.4.1 Autenticação com JSON Web Tokens (JWT)

A autenticação do sistema é baseada em JSON Web Tokens (JWT), um padrão aberto (RFC 7519) para a criação de tokens de acesso que afirmam um determinado número de "claims"(informações). Em uma arquitetura de microsserviços ou em aplicações de página única (SPA), o JWT é vantajoso por ser *stateless*: o servidor não precisa armazenar o estado da sessão do usuário. Após o login, o cliente recebe um token assinado que é enviado a cada requisição subsequente para validar a identidade e as permissões do usuário (Stack Overflow, 2019). Para mitigar riscos, é crucial validar o algoritmo do token (‘alg‘) no servidor e usar chaves de assinatura fortes, evitando vulnerabilidades conhecidas (PentesterLab, 2024).

2.3.4.2 Controle de Acesso Baseado em Papéis (RBAC)

Para gerenciar as permissões de acesso dentro do sistema, foi implementado o modelo de Controle de Acesso Baseado em Papéis (RBAC - *Role-Based Access Control*). O RBAC simplifica a administração de permissões ao associá-las a "papéis"(como "Funcionário"e "Gerente") em vez de a usuários individuais. Um usuário recebe acesso a um conjunto de funcionalidades com base nos papéis que lhe são atribuídos. Esse modelo centraliza a gestão de acesso, reduz a complexidade administrativa e fortalece a segurança ao garantir que os usuários possam acessar apenas as informações e funcionalidades estritamente necessárias para o desempenho de suas funções (IBM, 2024).

2.4 TENDÊNCIAS E FUTURO DA GESTÃO DE PONTO

O campo da gestão de Recursos Humanos está em constante transformação, impulsionado por tecnologias emergentes. Sistemas de ponto, como o desenvolvido neste projeto, estão na vanguarda dessa evolução. Tendências futuras apontam para uma integração ainda maior com tecnologias como Inteligência Artificial (IA) para análise preditiva de absenteísmo e biometria avançada, como o reconhecimento facial, para aumentar ainda mais a segurança e agilizar o processo de marcação de ponto (Oitchau, 2024; Vala, 2024). A contínua evolução dessas ferramentas promete revolucionar a

forma como as empresas gerenciam sua força de trabalho, tornando os processos cada vez mais inteligentes, eficientes e seguros.

3 REQUISITOS DO SISTEMA

Este capítulo apresenta a especificação dos requisitos funcionais e não funcionais do sistema de Registro de Ponto, obtidos através da análise das necessidades operacionais de empresas que necessitam controlar a jornada de trabalho de seus funcionários. Os requisitos foram organizados de forma a atender aos principais desafios de gestão de ponto eletrônico e controle de presença.

3.1 REQUISITOS FUNCIONAIS

Os requisitos funcionais descrevem as funcionalidades que o sistema de Registro de Ponto deve executar. A Tabela 1 apresenta os requisitos organizados por módulos funcionais.

3.2 REQUISITOS NÃO FUNCIONAIS

Os requisitos não funcionais definem os critérios de qualidade e restrições técnicas que o sistema deve atender. A Tabela 2 apresenta esses requisitos organizados por categorias.

Tabela 1 – Requisitos funcionais do sistema de Registro de Ponto

ID	Módulo	Descrição
Autenticação e Autorização		
RF01	Registro	Cadastro de usuários com validação de dados pessoais e profissionais
RF02	Login	Autenticação segura via e-mail e senha com controle de sessões
RF03	Permissões	Controle de acesso com diferentes níveis de usuário (administrador, funcionário)
Gestão Multi-empresa		
RF04	Empresas	Criação e gestão de perfis de empresa com dados cadastrais completos
RF05	Isolamento	Separação completa de dados entre diferentes empresas
Gestão de Funcionários		
RF06	Funcionários	Cadastro completo de funcionários com dados pessoais, profissionais e de contato
RF07	Departamentos	Criação e gestão de departamentos organizacionais
RF08	Cargos	Definição de cargos e responsabilidades dos funcionários
RF09	Horários	Configuração de horários de trabalho personalizados por funcionário
Registro de Ponto		
RF10	Registro	Registro de entrada, saída, início e fim de intervalo com geolocalização
RF11	Validação	Verificação automática de localização dentro do raio permitido
RF12	Status	Controle de status dos registros (aprovado, pendente, rejeitado)
RF13	Histórico	Consulta de histórico completo de registros de ponto
Justificativas		
RF14	Solicitação	Criação de justificativas para registros fora do raio ou problemas técnicos
RF15	Aprovação	Sistema de aprovação e rejeição de justificativas pelos gestores
RF16	Tipos	Categorização de justificativas (reunião externa, viagem a serviço, problemas técnicos)
Controle de Horas		
RF17	Banco de Horas	Cálculo automático do saldo de horas trabalhadas vs. previstas
RF18	Extras	Controle de horas extras e débitos de jornada
RF19	Relatórios	Geração de relatórios de horas por período
Dashboard e Métricas		
RF20	Dashboard	Visualização de métricas em tempo real (presentes, ausentes, horas extras)
RF21	Gráficos	Apresentação gráfica de dados de presença e produtividade
RF22	Alertas	Sistema de notificações para justificativas pendentes e relatórios disponíveis
Consultas e Filtros		
RF23	Busca	Pesquisa avançada de funcionários e registros com múltiplos filtros
RF24	Filtros	Filtros por período, departamento, status e tipo de registro
RF25	Exportação	Exportação de dados para relatórios e análises externas

Tabela 2 – Requisitos não funcionais do sistema de Registro de Ponto

ID	Categoria	Descrição
Usabilidade		
RNF01	Responsividade	Interface adaptável a diferentes dispositivos móveis e desktops
RNF02	Navegação	Interface intuitiva com navegação clara e feedback visual adequado
RNF03	Acessibilidade	Formulários guiados com indicadores de progresso em processos complexos
Segurança		
RNF04	Autenticação	Sistema de login seguro com controle de sessões e tokens JWT
RNF05	Autorização	Controle de acesso baseado em perfis de usuário e empresa
RNF06	Isolamento	Separação total de dados entre diferentes empresas
RNF07	Validação	Validação rigorosa de todas as entradas de dados e geolocalização
Performance		
RNF08	Carregamento	Tempo de resposta adequado para carregamento de páginas e registros
RNF09	Consultas	Otimização de consultas ao banco de dados para relatórios e histórico
RNF10	Escalabilidade	Suporte a crescimento do volume de dados e usuários simultâneos
Confiabilidade		
RNF11	Disponibilidade	Sistema disponível 24/7 para registro de ponto em qualquer horário
RNF12	Backup	Backup automático dos dados de registro e configurações
RNF13	Recuperação	Capacidade de recuperação de dados em caso de falhas
Manutenibilidade		
RNF14	Arquitetura	Código organizado em módulos bem definidos e reutilizáveis
RNF15	Padrões	Seguimento de padrões de codificação estabelecidos
RNF16	Documentação	API e código adequadamente documentados
Portabilidade		
RNF17	Containerização	Sistema deployável em containers para diferentes ambientes
RNF19	Banco de Dados	Compatibilidade com diferentes sistemas de banco de dados
Integração		
RNF20	Geolocalização	Integração com serviços de geolocalização para validação de localização

4 TECNOLOGIAS ENVOLVIDAS

Este capítulo apresenta as principais tecnologias que serão utilizadas no desenvolvimento do projeto, abrangendo tanto o front-end quanto o back-end. Serão descritas as linguagens de programação, frameworks, bibliotecas e ferramentas empregadas, bem como as razões para sua escolha e como cada uma contribui para o desenvolvimento do projeto.

4.1 TECNOLOGIAS FRONT-END

4.1.1 Next.js

O Next.js é um framework para aplicações web construído sobre o React, que permite a renderização no lado do servidor (Server-Side Rendering — SSR) e a geração de sites estáticos. Foi escolhido por oferecer, junto ao React, uma plataforma robusta, organizada e escalável, o que contribui diretamente para a qualidade e estrutura do desenvolvimento do projeto (Vercel, 2025).

4.1.2 React.js

O React.js é uma biblioteca JavaScript para criação de interfaces de usuário, baseada em componentes reutilizáveis. Foi escolhida por ser compatível com JavaScript e TypeScript, facilitar a modularização. (Meta Platforms, Inc., 2025).

4.1.3 TypeScript

O TypeScript é uma linguagem de programação que adiciona tipagem estática ao JavaScript, permitindo identificar erros ainda durante o desenvolvimento. Foi escolhido por melhorar a legibilidade, a manutenção do código e por oferecer maior segurança no desenvolvimento (Microsoft, 2025).

4.1.4 Tailwind CSS

O Tailwind CSS é um framework de folhas de estilo em cascata (Cascading Style Sheets — CSS) baseado em classes utilitárias. Ele permite a criação rápida de interfaces responsivas e customizáveis, com menos necessidade de escrever CSS manualmente. Foi escolhido por agilizar o desenvolvimento visual e garantir consistência no design (Tailwind Labs, 2025).

4.2 TECNOLOGIAS BACK-END

4.2.1 NestJS

O NestJS é um framework para construção de aplicações Node.js escaláveis e eficientes. Baseado em TypeScript, ele utiliza conceitos do Angular, como módulos, controladores e serviços, para estruturar o código de forma organizada e modular. O NestJS facilita a criação de APIs (Interfaces de Programação de Aplicações) robustas e de fácil manutenção (NestJS Contributors, 2025).

4.2.2 Prisma

O Prisma é um ORM (Mapeador Objeto-Relacional) moderno para Node.js e TypeScript. Ele simplifica a interação com o banco de dados, oferecendo uma API intuitiva e tipada para consultas e manipulação de dados. O Prisma facilita a manutenção da consistência dos dados e melhora a produtividade no desenvolvimento (Prisma Data, Inc., 2025).

4.2.3 PostgreSQL

O PostgreSQL é um sistema de gerenciamento de banco de dados relacional de código aberto, conhecido por sua robustez, desempenho e conformidade com padrões. Ele suporta uma ampla variedade de tipos de dados e funcionalidades avançadas, sendo uma escolha sólida para aplicações que requerem integridade e escalabilidade dos dados (The PostgreSQL Global Development Group, 2025).

4.3 RESUMO DAS TECNOLOGIAS E VERSÕES

A Tabela 3 apresenta um resumo das principais tecnologias utilizadas no projeto, incluindo suas respectivas versões.

Tabela 3 – Resumo das tecnologias e versões utilizadas no projeto

Tecnologia	Versão	Categoria	Propósito
Frontend			
Next.js	15.3.5	Framework	Renderização e roteamento
React	19.0.0	Biblioteca	Interface de usuário
TypeScript	5.x	Linguagem	Tipagem estática
Tailwind CSS	4.x	Framework CSS	Estilização responsiva
Radix UI	1.x/2.x	Biblioteca	Componentes acessíveis
React Hook Form	7.60.0	Biblioteca	Gerenciamento de formulários
Zod	3.25.75	Biblioteca	Validação de dados
Recharts	2.15.4	Biblioteca	Gráficos e visualizações
Backend			
NestJS	11.0.1	Framework	API e estrutura do servidor
TypeORM	0.3.25	ORM	Mapeamento objeto-relacional
PostgreSQL	8.16.3	Banco de Dados	Armazenamento de dados
JWT	11.0.0	Biblioteca	Autenticação e autorização
Passport	0.7.0	Biblioteca	Estratégias de autenticação
Bcrypt	6.0.0	Biblioteca	Criptografia de senhas
Class Validator	0.14.2	Biblioteca	Validação de dados
Ferramentas de Desenvolvimento			
ESLint	9.x	Linter	Análise estática de código
Prettier	3.4.2	Formatador	Formatação de código
Jest	29.7.0	Framework	Testes automatizados
Docker	-	Containerização	Deploy e isolamento

5 ARQUITETURA E MODELAGEM DO SISTEMA

Este capítulo apresenta a arquitetura e modelagem do sistema de registro de ponto eletrônico desenvolvido, detalhando a estrutura organizacional do *software*, as tecnologias empregadas, os padrões arquiteturais adotados e o modelo de dados implementado. O sistema foi concebido seguindo uma arquitetura distribuída, com separação clara entre *frontend* e *backend*, permitindo escalabilidade, manutenibilidade e flexibilidade na evolução da aplicação.

5.1 VISÃO GERAL DA ARQUITETURA

O sistema de registro de ponto eletrônico foi desenvolvido utilizando uma arquitetura em três camadas (3-tier), separando claramente as responsabilidades entre apresentação, lógica de negócio e persistência de dados. Esta abordagem segue os princípios da arquitetura SOA (*Service-Oriented Architecture*) e REST (*Representational State Transfer*), promovendo a interoperabilidade e facilitando a integração com sistemas externos.

A Figura 1 apresenta a visão geral da arquitetura do sistema, destacando os principais componentes e suas interações.

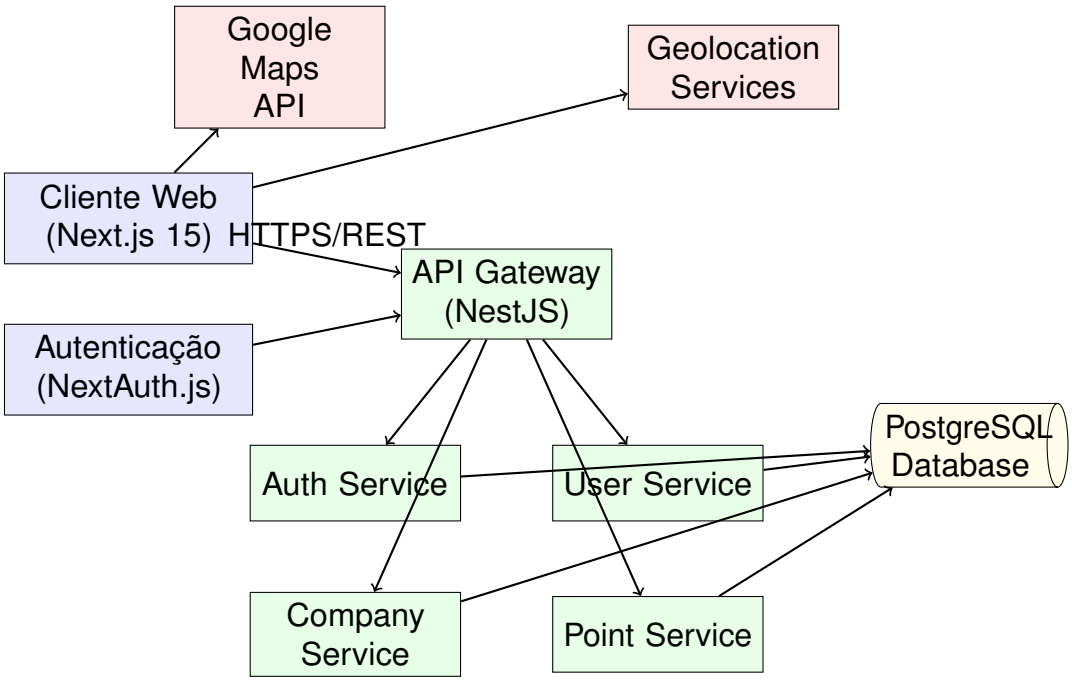

Figura 1 – Arquitetura geral do sistema de registro de ponto

A arquitetura adotada proporciona as seguintes vantagens:

- **Separação de Responsabilidades**: Cada camada possui responsabilidades bem definidas, facilitando a manutenção e evolução do sistema;

- **Escalabilidade**: A arquitetura permite escalar horizontalmente cada componente de forma independente;
- **Testabilidade**: A separação em camadas facilita a implementação de testes unitários e de integração;
- **Reutilização**: Os serviços do *backend* podem ser consumidos por diferentes clientes;
- **Segurança**: A centralização da lógica de autenticação e autorização no *backend* garante maior segurança.

5.2 MODELAGEM DE DADOS

O modelo de dados foi projetado seguindo os princípios da normalização de banco de dados e os padrões do Domain-Driven Design (DDD). O sistema utiliza PostgreSQL como sistema de gerenciamento de banco de dados, aproveitando suas características de robustez, conformidade com ACID e suporte a tipos de dados complexos.

A Figura 2 apresenta o diagrama de classes UML do modelo de dados, destacando as principais entidades, seus atributos, métodos e relacionamentos.

5.2.1 Entidades Principais

O modelo de dados é composto pelas seguintes entidades principais:

Empresa Representa a organização que utiliza o sistema. Armazena informações corporativas, configurações de geolocalização (latitude, longitude, raio permitido) e políticas de tolerância para registros de ponto.

Usuario Modela os usuários do sistema (proprietários, administradores e funcionários). Implementa controle de acesso baseado em papéis (RBAC - *Role-Based Access Control*) através do campo `papel`.

Departamento Representa as divisões organizacionais da empresa, permitindo estruturação hierárquica e controle administrativo.

Cargo Define as posições/funções dentro da organização, incluindo informações salariais base para cálculos de folha de pagamento.

InformacoesTrabalhistas Consolida dados contratuais do funcionário, incluindo data de admissão, carga horária semanal e salário.

HorarioFuncionario Modela os horários de trabalho individuais, permitindo flexibilidade na definição de expedientes por dia da semana.

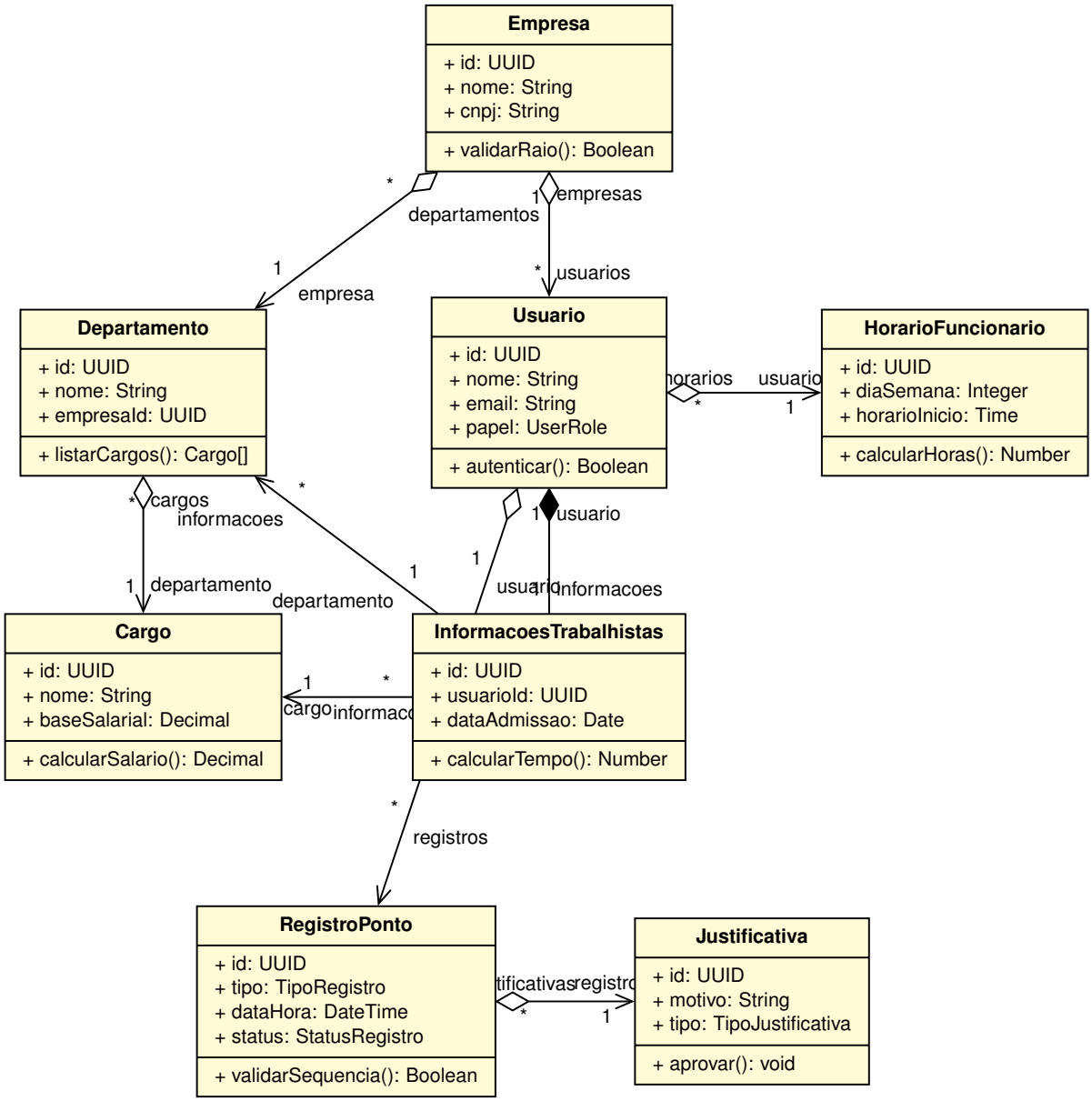

Figura 2 – Diagrama de classes UML do modelo de dados

RegistroPonto Armazena os registros de ponto dos funcionários, incluindo validação geográfica e controle de status (aprovado, pendente, justificado).

Justificativa Implementa o sistema de justificativas para registros irregulares, com workflow de aprovação.

5.2.2 Relacionamentos e Cardinalidades

Os relacionamentos entre as entidades seguem as regras de negócio do domínio:

- Uma empresa pode ter múltiplos usuários, departamentos e configurações de horário (1:N);

- Cada usuário pertence a uma única empresa, mas pode ter múltiplos registros de ponto e horários (1:N);
- As informações trabalhistas mantêm relacionamento 1:1 com o usuário, garantindo integridade referencial;
- Registros de ponto podem ter múltiplas justificativas, permitindo rejeição e reenvio (1:N).

5.3 ARQUITETURA DO BACK-END

O *backend* foi desenvolvido utilizando o *framework* NestJS, que implementa os padrões arquiteturais do Angular para o ambiente Node.js. A escolha do NestJS se justifica pela robustez na implementação de APIs RESTful, suporte nativo a TypeScript, sistema de injeção de dependências e extensa capacidade de integração com bibliotecas do ecossistema Node.js.

5.3.1 Estrutura Modular

A arquitetura do *backend* segue o padrão modular do NestJS, organizando funcionalidades em módulos coesos e fracamente acoplados. A Figura 3 ilustra a estrutura modular implementada.

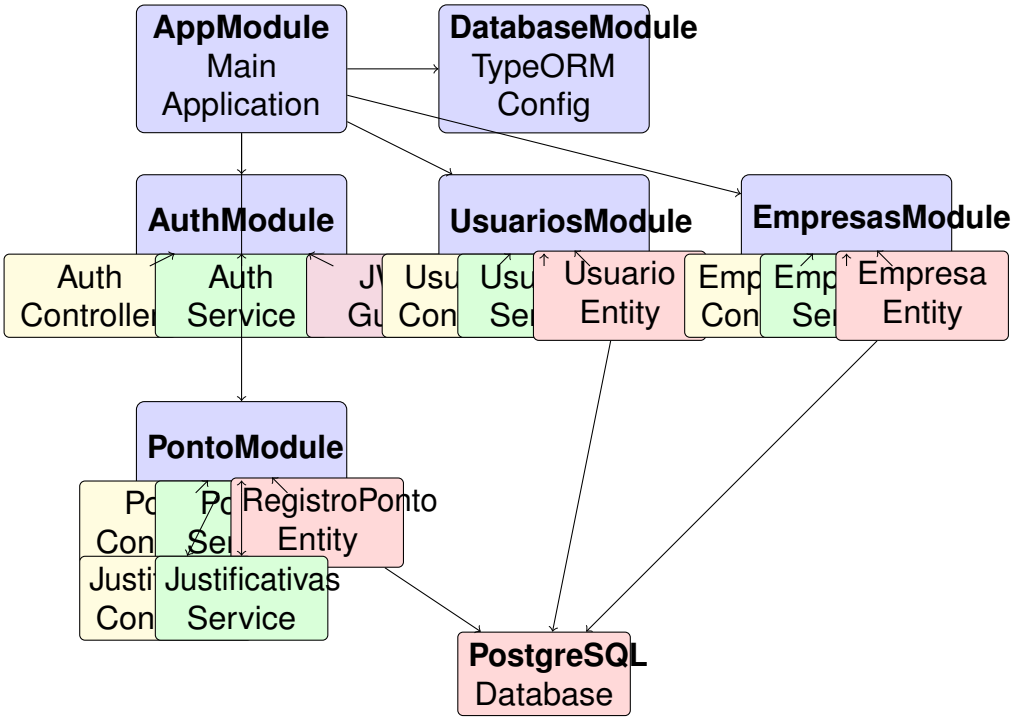

Figura 3 – Arquitetura modular do backend NestJS

5.3.2 Camadas Arquiteturais

O *backend* implementa uma arquitetura em camadas bem definidas:

Camada de Apresentação (Controllers) Responsável por receber e processar requisições HTTP, validar parâmetros de entrada e retornar respostas adequadas. Utiliza decoradores do NestJS para mapeamento de rotas e validação automática através das bibliotecas `class-validator` e `class-transformer`.

Camada de Negócio (Services) Concentra a lógica de negócio da aplicação, implementando regras específicas do domínio como validação de geolocalização, cálculo de banco de horas e workflows de aprovação de justificativas.

Camada de Acesso a Dados (Repositories) Implementada através do TypeORM, fornece abstração para operações de persistência, mapeamento objeto-relacional e gerenciamento de transações.

Camada de Segurança (Guards e Middlewares) Implementa autenticação JWT, autorização baseada em papéis e validações de segurança transversais.

5.3.3 Principais Funcionalidades

O *backend* implementa as seguintes funcionalidades principais:

- **Autenticação e Autorização**: Sistema completo baseado em JWT com refresh tokens e controle de sessão;
- **Gestão de Usuários**: CRUD completo com validação de dados e controle de status;
- **Controle de Ponto**: Registro com validação geográfica, sequenciamento obrigatório e cálculo automático de horas;
- **Sistema de Justificativas**: Workflow completo com aprovação/rejeição e rastreabilidade;
- **Relatórios**: Cálculo de banco de horas, extração de dados e relatórios gerenciais.

5.4 ARQUITETURA DO FRONT-END

O *frontend* foi desenvolvido utilizando Next.js 15 com React 19, aproveitando as funcionalidades avançadas de Server-Side Rendering (SSR), Static Site Generation (SSG) e o novo App Router. Esta escolha tecnológica proporciona excelente performance, SEO otimizado e experiência de usuário superior.

5.4.1 Estrutura de Componentes

A aplicação segue os padrões de design de componentes do React, implementando uma arquitetura baseada em componentes reutilizáveis e contextualizada. A Figura 4 apresenta a estrutura organizacional do *frontend*.

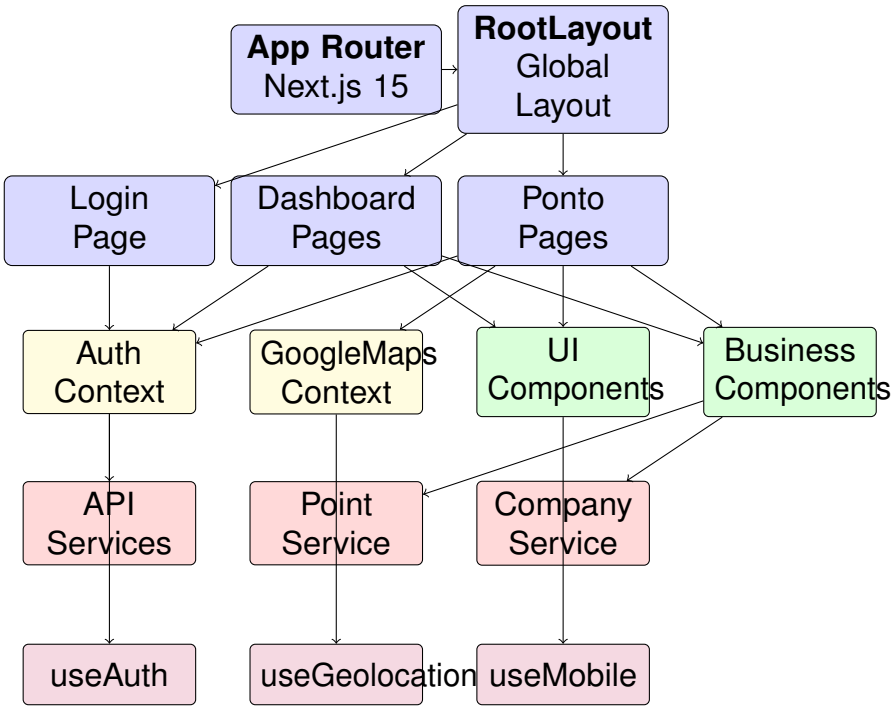

Figura 4 – Arquitetura de componentes do frontend Next.js

5.4.2 Padrões Arquiteturais Implementados

O *frontend* implementa diversos padrões arquiteturais modernos:

Component-Based Architecture Toda a interface é construída através de componentes React reutilizáveis, seguindo os princípios de Single Responsibility e composição.

Context API Implementação de contextos globais para autenticação, configurações de mapas e estado da aplicação, evitando prop drilling e centralizando estados compartilhados.

Custom Hooks Encapsulamento de lógicas complexas em hooks customizados, promovendo reutilização e separação de responsabilidades.

Service Layer Abstração das chamadas de API através de serviços especializados, facilitando manutenção e testes.

Design System Implementação de um sistema de design consistente utilizando Radix UI e Tailwind CSS, garantindo acessibilidade e responsividade.

5.4.3 Gerenciamento de Estado

O gerenciamento de estado da aplicação utiliza uma abordagem híbrida:

- **Estado Local**: Componentes individuais gerenciam seus estados internos através do hook `useState`;
- **Estado Global**: Contextos React para dados compartilhados (autenticação, configurações);
- **Estado do Servidor**: React Query (TanStack Query) para cache e sincronização de dados remotos;
- **Estado de Formulários**: React Hook Form para validação e gerenciamento eficiente de formulários.

5.5 FLUXO DE AUTENTICAÇÃO E AUTORIZAÇÃO

O sistema implementa um fluxo de autenticação robusto baseado em JWT (*JSON Web Tokens*) com integração entre *frontend* e *backend*. A Figura 5 ilustra o processo completo de autenticação.

Figura 5 – Fluxo de autenticação e autorização

5.5.1 Componentes de Segurança

O sistema de segurança é composto pelos seguintes elementos:

JWT Strategy Implementação da estratégia Passport.js para validação de tokens JWT no *backend*;

NextAuth.js Biblioteca de autenticação para Next.js que gerencia sessões, tokens e callbacks;

Guards Implementação de guards NestJS para proteção de rotas e validação de permissões;

Middleware Middleware Next.js para proteção de rotas do *frontend* baseado em papéis;

RBAC Sistema de controle de acesso baseado em papéis com três níveis: funcionário, administrador e proprietário.

5.5.2 Validação de Geolocalização

Um dos diferenciais do sistema é a validação automática de geolocalização para registros de ponto. O processo utiliza a API de Geolocalização do navegador integrada com o Google Maps API para validação de presença física no local de trabalho.

A Figura 6 apresenta o algoritmo de validação geográfica implementado.

O algoritmo implementa a fórmula de Haversine para cálculo preciso de distância entre coordenadas geográficas, considerando a curvatura da Terra. A validação permite configuração flexível de raio de tolerância por empresa e políticas diferenciadas para registros fora da área permitida.

5.6 INTEGRAÇÃO COM SERVIÇOS EXTERNOS

O sistema integra-se com serviços externos para expandir suas funcionalidades:

5.6.1 Google Maps API

A integração com o Google Maps API proporciona:

- Seleção visual de localização da empresa através de interface interativa;
- Geocodificação reversa para conversão de coordenadas em endereços;
- Validação de endereços e otimização de coordenadas;
- Visualização de mapas para administradores supervisionarem registros.

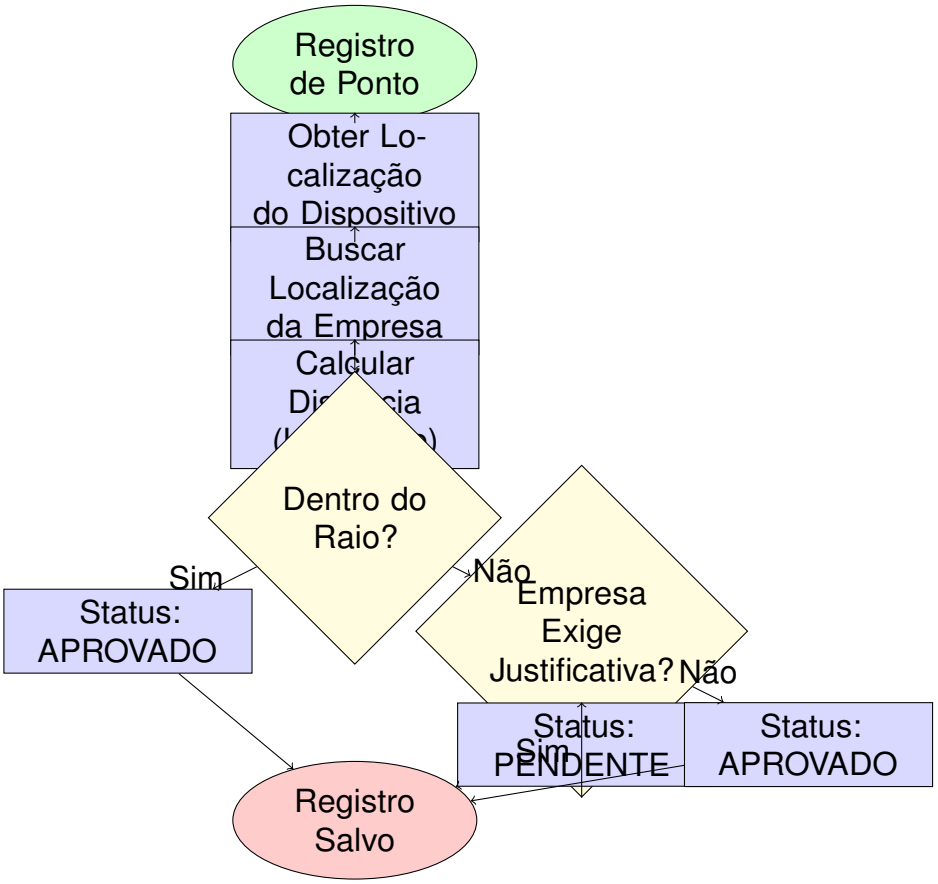

Figura 6 – Algoritmo de validação geográfica

5.6.2 Geolocation API

A API nativa de geolocalização do navegador fornece:

- Coordenadas precisas do dispositivo do funcionário;
- Detecção automática de localização sem intervenção manual;
- Tratamento de erros e permissões negadas;
- Otimização de precisão baseada em contexto.

5.7 CONSIDERAÇÕES DE PERFORMANCE E ESCALABILIDADE

A arquitetura implementada considera aspectos fundamentais de performance e escalabilidade:

Cache Inteligente Implementação de cache em múltiplas camadas: browser cache, Redis para sessões e cache de consultas no TypeORM;

Lazy Loading Carregamento sob demanda de componentes React e módulos Next.js;

Database Optimization Índices otimizados, consultas eficientes e relacionamentos bem estruturados;

API Pagination Implementação de paginação em todas as consultas que retornam listas;

Horizontal Scaling Arquitetura preparada para escalamento horizontal através de load balancers e múltiplas instâncias.

A arquitetura apresentada demonstra uma implementação robusta e moderna, seguindo as melhores práticas da engenharia de *software* e proporcionando uma base sólida para evolução e manutenção do sistema de registro de ponto eletrônico.

REFERÊNCIAS

ABREU, Lucas Gennari Silva. Sistema de controle de ponto auxiliado por aplicativo Android, 2016. Citado na p. 9.

AMAZON WEB SERVICES. **Qual é a diferença entre o MongoDB e o PostgreSQL?** [*S. l.: s. n.*], 2024. Acesso em: 27 ago. 2025. Disponível em: https://aws.amazon.com/pt/compare/the-difference-between-mongodb-and-postgresql/. Citado na p. 16.

AMAZON WEB SERVICES. **Qual é a diferença entre o MySQL e o PostgreSQL?** [*S. l.: s. n.*], 2024. Acesso em: 27 ago. 2025. Disponível em: https://aws.amazon.com/pt/compare/the-difference-between-mysql-vs-postgresql/. Citado na p. 16.

BRASIL. **Consolidação das Leis do Trabalho (CLT)**. [*S. l.*]: Imprensa Nacional, 1943. Citado na p. 9.

CARDOSO, Eluth. **Portaria 671: regras para marcação de ponto nas empresas**. [*S. l.: s. n.*], 2024. Acesso em: 27 ago. 2025. Disponível em: https://www.contabeis.com.br/noticias/71604/portaria-671-regras-para-marcacao-de-ponto-nas-empresas/. Citado na p. 13.

CORRALES, Victor. **MVP (Mínimo Produto Viável): o que é, como fazer e exemplos para se inspirar**. [*S. l.: s. n.*], 2024. Acesso em: 27 ago. 2025. Disponível em: https://www.rdstation.com/blog/marketing/mvp-minimo-produto-viavel/. Citado na p. 15.

DEV&DELIVER. **Level up Your Backend: A C-Suite Guide to Mastering Modular Architecture in Nest.js Applications**. [*S. l.: s. n.*], 2024. Acesso em: 27 ago. 2025. Disponível em: https://devanddeliver.com/blog/backend/level-up-your-backend-a-c-suite-guide-to-mastering-modular-architecture-in-nest-js-applications. Citado na p. 15.

FACTORIAL. **Ponto por geolocalização: é legal? O que diz a lei?** [*S. l.: s. n.*], 2024. Acesso em: 27 ago. 2025. Disponível em: https://factorialhr.com.br/blog/ponto-por-geolocalizacao/. Citado na p. 14.

FLORINDO, Glênio Henrique Carvalho; BIANCHI, Renata Alexandre. **Desenvolvimento de Aplicação para Registro de Ponto Inteligente**. 2022. Trabalho de Graduação – Faculdade de Tecnologia de Franca - "Dr. Thomaz Novelino", Franca, SP. Orientador: Profª Dra. Jaqueline Brigladori Pugliesi. Citado nas pp. 9, 10.

FORPEOPLE. **Controle de ponto manual: entenda os riscos desse sistema**. [*S. l.: s. n.*], 2023. Acesso em: 27 ago. 2025. Disponível em: https://forpeople.io/recursos-humanos/controle-de-ponto-manual-entenda-os-riscos-desse-sistema/. Citado na p. 13.

GEOVICTORIA. **Ponto manual ou digital: qual o melhor para a sua empresa?** [*S. l.: s. n.*], 2024. Acesso em: 27 ago. 2025. Disponível em: https://www.geovictoria.com.br/br/blog/ponto-manual-ou-digital-qual-o-melhor-para-sua-empresa/. Citado na p. 13.

GOMES, José Victor Magalhães. **PontoUp - Sistema de Gerenciamento e Registro de Ponto Eletrônico**. 2023. Tese (Doutorado) – Universidade Federal Rural do Semi-Árido. Citado nas pp. 9, 10.

IBM. **O que é o controle de acesso baseado em funções (RBAC)?** [*S. l.: s. n.*], 2024. Acesso em: 27 ago. 2025. Disponível em: https://www.ibm.com/br-pt/think/topics/rbac. Citado na p. 16.

LONGO, Maria Tereza; WATANABE, Carolina Yukari Veludo. Transformação Digital: Uma análise das principais barreiras e dificuldades em micro e pequenas empresas. *In:* ANAIS do XXII SEMEAD - Seminários em Administração. São Paulo, Brasil: [*s. n.*], 2019. Disponível em: https://login.semead.com.br/22semead/anais/arquivos/1022.pdf. Citado na p. 10.

MARIOTTI, Isabel; KIENETZ, Taiani Bacchi. **Perspectivas da Utilização do Ponto Eletrônico de Acordo com a Portaria 1.510/09**. 2011. Tese (Doutorado) – Universidade Federal de Santa Maria. Citado na p. 9.

META PLATFORMS, INC. **React – A JavaScript library for building user interfaces**. [*S. l.: s. n.*], 2025. https://reactjs.org/. Acesso em: 27 mar. 2025. Citado na p. 21.

MICROSOFT. **TypeScript - JavaScript with syntax for types**. [*S. l.: s. n.*], 2025. https://www.typescriptlang.org/. Acesso em: 27 mar. 2025. Citado na p. 21.

MIRANDA, Izabella. **Controle de ponto: conheça os tipos e escolha o ideal para a sua empresa**. Acesso em: 10 mar. 2025. 2023. Disponível em: https://www.contabeis.com.br/noticias/60426/controle-de-ponto-conheca-os-tipos-e-escolha-o-ideal-para-a-sua-empresa/. Citado na p. 9.

NESTJS CONTRIBUTORS. **NestJS - A progressive Node.js framework for building efficient, reliable and scalable server-side applications**. [*S. l.: s. n.*], 2025. https://nestjs.com/. Acesso em: 27 mar. 2025. Citado na p. 22.

OITCHAU. **Como fazer controle de ponto por reconhecimento facial?** [*S. l.: s. n.*], 2024. Acesso em: 27 ago. 2025. Disponível em: https://www.oitchau.com.br/blog/como-fazer-controle-de-ponto-por-reconhecimento-facial/. Citado na p. 16.

PENTESTERLAB. **JWT Vulnerabilities & Attacks: The Ultimate Guide**. [*S. l.: s. n.*], 2024. Acesso em: 27 ago. 2025. Disponível em: https://pentesterlab.com/blog/jwt-vulnerabilities-attacks-guide. Citado na p. 16.

PRISMA DATA, INC. **Prisma - Next-generation Node.js and TypeScript ORM**. [*S. l.: s. n.*], 2025. https://www.prisma.io/. Acesso em: 27 mar. 2025. Citado na p. 22.

SEGUROS, Equipe Compare. **A Importância da Jornada de Trabalho Regulada pela CLT**. [*S. l.: s. n.*], 2024. https://compareplanodesaude.com.br/empresarial/clt-beneficios/importancia-jornada-trabalho-regulada-clt/. Acessado em: 2025-08-27. Citado na p. 13.

SENIOR SISTEMAS. **Controle de ponto com geolocalização: como funciona e quais as vantagens**. [*S. l.: s. n.*], 2023. Acesso em: 27 ago. 2025. Disponível em: https://www.senior.com.br/blog/controle-de-ponto-geolocalizacao. Citado na p. 14.

SHAH, Chintan. **Next.js vs Angular vs Vue.js: A Detailed Comparison**. [*S. l.: s. n.*], 2024. Acesso em: 27 ago. 2025. Disponível em: https://www.brilworks.com/blog/javascript-web-frameworks-comparison/. Citado na p. 15.

SOFTKRAFT. **11 Minimum Viable Product Examples: From Zero to Hero**. [*S. l.: s. n.*], 2024. Acesso em: 27 ago. 2025. Disponível em: https://www.softkraft.co/minimum-viable-product-examples/. Citado na p. 15.

STACK OVERFLOW. **Qual método usar para deixar um usuário logado? JWT (Cookie vs Session)**. [*S. l.: s. n.*], 2019. Acesso em: 27 ago. 2025. Disponível em: https://pt.stackoverflow.com/questions/376924/qual-m%C3%A9todo-usar-para-deixar-um-usu%C3%A1rio-logado-jwt-cookie-vs-session. Citado na p. 16.

STRAPI. **SSR vs SSG in Next.js: Differences, Advantages and Use Cases**. [*S. l.: s. n.*], 2024. Acesso em: 27 ago. 2025. Disponível em: https://strapi.io/blog/ssr-vs-ssg-in-nextjs-differences-advantages-and-use-cases. Citado na p. 15.

TAILWIND LABS. **Tailwind CSS - Rapidly build modern websites without ever leaving your HTML**. [*S. l.: s. n.*], 2025. https://tailwindcss.com/. Acesso em: 27 mar. 2025. Citado na p. 21.

THE POSTGRESQL GLOBAL DEVELOPMENT GROUP. **PostgreSQL - The world's most advanced open source database**. [*S. l.: s. n.*], 2025. https://www.postgresql.org/. Acesso em: 27 mar. 2025. Citado na p. 22.

TOTVS. **Portaria 671/21: o que é, mudanças e guia completo**. [*S. l.: s. n.*], 2024. Acesso em: 27 ago. 2025. Disponível em: https://espacolegislacao.totvs.com/portaria-671/. Citado na p. 14.

VALA, Ana. **Tendências de RH para 2025: o que esperar do futuro do trabalho?** [*S. l.: s. n.*], 2024. Acesso em: 27 ago. 2025. Disponível em: https://www.feedz.com.br/blog/tendencias-de-rh-2025/. Citado na p. 16.

VERCEL. **Next.js - The React Framework**. [*S. l.: s. n.*], 2025. https://nextjs.org/. Acesso em: 27 mar. 2025. Citado na p. 21.