



**INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO
TOCANTINS – CAMPUS ARAGUATINS
COORDENAÇÃO DE COMPUTAÇÃO
CURSO DE LICENCIATURA EM COMPUTAÇÃO**

JULIO CÉSAR COSTA DE MACEDO

REMEDIUM: Uma aplicação de lembretes de remédios.

ARAGUATINS

2026

JULIO CESAR COSTA DE MACEDO

REMEDIUM: Uma aplicação de lembretes de remédios usando push notification.

Trabalho de Conclusão de Curso apresentado à Coordenação do Curso superior de Licenciatura em Computação da Unidade de Araguatins, do Instituto Federal do Tocantins, como exigência à obtenção do título de Licenciado, em Computação.

Orientador: Prof. Mestre Adeilson Marques da Silva Cardoso

ARAGUATINS

2026

Dados Internacionais de Catalogação na Publicação (CIP)
Bibliotecas do Instituto Federal do Tocantins

M141r Macedo, Julio Cesar Costa de
 Remedium : Uma aplicação de lembretes de remédios / Julio
Cesar Costa de Macedo. – Araguatins, TO, 2026.
 40 f. : il. color.

Trabalho de Conclusão de Curso (Licenciatura em Computação)
– Instituto Federal de Educação, Ciência e Tecnologia do Tocantins,
Campus Araguatins, Araguatins, TO, 2026.

Orientador: Me. Adeilson Marques da Silva Cardoso

1. Aplicativo móvel. 2. Lembretes. 3. Notificações push. I.
Cardoso, Adeilson Marques da Silva. II. Título.

CDD 004

A reprodução total ou parcial, de qualquer forma ou por qualquer meio, deste documento é autorizada para fins
de estudo e pesquisa, desde que citada a fonte.

Elaborado pelo sistema de geração automática de ficha catalográfica do IFTO com os dados fornecidos
pelo(a) autor(a).



Ministério da Educação
Secretaria de Educação Profissional e Tecnológica
Instituto Federal de Educação, Ciência e Tecnologia do Tocantins
Campus Araguatins

FOLHA DE APROVAÇÃO

TÍTULO DO TRABALHO: TÍTULO DO TRABALHO: REMEDIUM: UMA APLICAÇÃO DE LEMBRETES DE REMÉDIOS USANDO PUSH NOTIFICATION

AUTOR: JULIO CÉSAR COSTA DE MACEDO

ORIENTADOR: Prof. M.e Adeilson Marques da Silva Cardoso

Trabalho de Conclusão de Curso apresentado ao Instituto Federal de Educação, Ciência e Tecnologia do Tocantins, *Campus Araguatins*, como parte das exigências para a conclusão do Curso de Licenciatura em Computação.

Aprovado em 10 de dezembro de 2025.

Prof. M.e Adeilson Marques da Silva Cardoso
Presidente da Banca

Prof. Dr. : Ennio Willian Lima Silva
Primeiro avaliador

Prof. Me.: Ramasio Ferreira de Melo
Segundo avaliador



Documento assinado eletronicamente por **Adeilson Marques da Silva Cardoso, Servidor**, em 22/01/2026, às 11:07, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do [Decreto nº 8.539, de 8 de outubro de 2015](#).



Documento assinado eletronicamente por **Ramasio Ferreira de Melo, Coordenador**, em 24/03/2026, às 11:39, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do [Decreto nº 8.539, de 8 de outubro de 2015](#).



Documento assinado eletronicamente por **Ennio Willian Lima Silva, Servidor**, em 24/03/2026, às 14:32, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do [Decreto nº 8.539, de 8 de outubro de 2015](#).

AGRADECIMENTOS

Agradeço primeiramente a Deus por ter sido meu refúgio e me ajudado a prosseguir. Agradeço também à minha família por sempre estar ao meu lado, em especial aos meus pais, João Batista de Macedo Sobrinho e Gildeneia Maria Cavalcante Costa, por todo afeto e carinho ao longo desses anos. Sem o apoio financeiro e emocional deles, eu jamais conseguiria chegar até aqui. Agradeço à minha irmã, Julianne Costa de Macedo, por se fazer presente durante essa jornada. Aos meus amigos, que se tornaram minha família ao longo desses anos de faculdade, sempre pude contar com: Tainá da Silva Veras, João Pedro Nogueira Viera, Marcelo Gomes da Silva, Danillo Silva Conceição, Shakira Santos Rodrigues, Jhonatan Torres Fernandes, Kelly Luana de Sousa Silva e Tiago Azevedo da Silva. Agradeço a todos os meus amigos, por quem tenho imensa admiração. Ao meu orientador, Me. Adeilson da Silva Cardoso, que me conduziu com seu apoio e motivação, sendo um amigo em todas as horas que precisei. Ele me incentivou sempre que pensei em desistir durante a realização deste trabalho. Obrigado por tudo. Agradeço a todos os meus professores que estiveram presentes neste percurso de forma excepcional e que, em certos momentos, foram mais que docentes, foram amigos: Claudio Galvão, Marco Paiva, Alex Sousa, Ramásio Melo, Adeilson da Silva Cardoso, Rogério Pereira de Sousa e aos demais professores que me serviram de inspiração. Sou grato a todos vocês. Agradeço a esta instituição que me proporcionou inúmeras oportunidades para crescer e aprender, de forma pessoal e profissional. E a todos aqueles que contribuíram de alguma forma durante a minha formação, seja com gestos ou palavras de incentivo.

“O ontem é história, o amanhã é um mistério, mas o hoje é uma dádiva. É por isso que se chama presente.” Kung Fu Panda

RESUMO

Este trabalho apresenta o desenvolvimento do Remedium, um aplicativo móvel para dispositivos Android concebido com o propósito de fornecer lembretes de medicamentos por meio de notificações push. Diante da crescente popularização dos smartphones e de sua presença ubíqua na vida cotidiana, torna-se fundamental explorar soluções tecnológicas que auxiliem na gestão da saúde. A dificuldade em lembrar de tomar medicamentos nos horários prescritos constitui um desafio recorrente, capaz de comprometer a eficácia dos tratamentos e gerar consequências significativas, especialmente em casos de doenças crônicas ou de regimes terapêuticos complexos. O Remedium surge como uma resposta a essa problemática, com o objetivo de aprimorar a adesão ao uso correto da medicação e, conseqüentemente, melhorar os resultados de saúde dos usuários. O aplicativo foi desenvolvido com React Native no front-end, NestJS no back-end e PostgreSQL como banco de dados. Para a infraestrutura em nuvem, foram empregados recursos da Contabo, em conjunto com Docker, Traefik e Easy Panel, assegurando um ambiente robusto, escalável e de fácil manutenção. O estudo busca, portanto, analisar o processo de desenvolvimento e implementação do Remedium, bem como discutir seu impacto potencial na adesão terapêutica e no bem-estar dos usuários. Além disso, pretende contribuir para o corpo de conhecimento no campo de aplicativos móveis voltados à saúde digital, reforçando a relevância de ferramentas tecnológicas como suporte à prática clínica e à autonomia do paciente.

Palavras-chave: Aplicativo móvel; Lembretes; Notificações push; Adesão à medicação; Saúde digital.

ABSTRACT

This work presents the development of Remedium, a mobile application for Android devices designed to provide medication reminders through push notifications. Given the growing popularity of smartphones and their ubiquitous presence in everyday life, it becomes essential to explore technological solutions that support health management. The difficulty of remembering to take medications at prescribed times is a recurring challenge, capable of compromising treatment effectiveness and generating significant consequences, especially in cases of chronic diseases or complex therapeutic regimens. Remedium emerges as a response to this issue, aiming to improve adherence to proper medication use and, consequently, enhance users' health outcomes. The application was developed with React Native on the front-end, NestJS on the back end, and PostgreSQL as the database. For cloud infrastructure, resources from Contabo were employed, combined with Docker, Traefik, and Easy Panel, ensuring a robust, scalable, and easy-to-maintain environment. Therefore, the study seeks to analyze the development and implementation process of Remedium, as well as to discuss its potential impact on therapeutic adherence and users' well-being. In addition, it aims to contribute to the body of knowledge in the field of mobile health applications, reinforcing the relevance of technological tools as support for clinical practice and patient autonomy.

Keywords: Mobile application; Reminders; Push notifications; Medication adherence; Digital health.

LISTA DE FIGURAS

Figura 1 - Client Side Request in NestJS	18
Figura 2 - Instalar vs code	20
Figura 3 - Instalar nvm	21
Figura 4 - Instalar android studio.....	21
Figura 5 - Instalar docker e nestjs cli	22
Figura 6 - Criar projeto react native	22
Figura 7 - Tela básica.....	23
Figura 8 - Estilização.....	24
Figura 9 - Interface	25
Figura 10 - Instalar NestJS cli	26
Figura 11 - Estrutura da API	26
Figura 12 - Arquitetura geral do sistema Remedium	30
Figura 13 - Login	32
Figura 14 - Register.....	33
Figura 15 - Tela inicial.....	34
Figura 16 - Criar alerta	35
Figura 17 - página inicial	36
Figura 18 - Configurações	37
Figura 19 - Push notification	38

LISTA DE ABREVIACÕES E SIGLAS

API – Application Programming Interface

CLI – Command Line Interface

CSS – Cascading Style Sheets

FP – Functional Programming

FRP – Functional Reactive Programming

GUI – Graphical User Interface

HTML – HyperText Markup Language

HTTP – Hypertext Transfer Protocol

IDE – Integrated Development Environment

IFTO – Instituto Federal de Educação, Ciência e Tecnologia do Tocantins

iOS – iPhone Operating System

NMC – New Media Consortium

NVMe – Non-Volatile Memory Express

OOP – Object-Oriented Programming

UI – User Interface URL – Uniform Resource Locator

UX – User Experience

VDS – Virtual Dedicated Server

VPS – Virtual Private Server

VS Code – Visual Studio Code

SUMÁRIO

1 INTRODUÇÃO.....	13
1.1 Problemática.....	14
1.2 Justificativa	14
1.3 Objetivo	15
1.4 Objetivos específicos.....	15
2 FUNDAMENTAÇÃO TEÓRICA	15
2.1 Javascript.....	15
2.2 TypeScript.....	16
2.3 React Native	16
2.4 Front-end	16
2.5 Back end.....	17
2.6 NestJS	17
2.7 Contabo	18
2.8 Docker	19
2.9 Traefik.....	19
2.13 Easy Panel.....	19
3 PROCEDIMENTOS METODOLÓGICOS	20
3.1 Ambiente de desenvolvimento.....	20
3.2 Mobile Front-End	22
3.3 Back-end.....	25
3.4 Implementação das Notificações Push.....	27
3.4.1 Agendamento e Verificação dos Alertas.....	28
3.4.2 Tratamento de Fuso Horário (Timezone).....	28
3.4.3 Envio das Notificações Push	29
3.5 Arquitetura Geral do Sistema.....	29
4 Resultados.....	31

4.1 Acesso ao Sistema	32
4.2 Cadastro e Gerenciamento de Alertas.....	34
4.3 Configurações do Aplicativo	36
4.4 Notificações Push	37
5 CONCLUSÃO E TRABALHOS FUTUROS.....	39
REFERÊNCIAS.....	40

1 INTRODUÇÃO

Com o avanço da tecnologia, os computadores tornaram-se cada vez mais compactos, culminando na popularização dos smartphones, dispositivos de bolso multifuncionais. Segundo a Deloitte (2019), os smartphones se tornaram ubíquos no mundo contemporâneo, com cerca de 90% dos adultos possuindo um, e 95% desses dispositivos sendo utilizados diariamente.

A disseminação desses dispositivos, aliada à sua crescente capacidade de processamento e conectividade, proporcionou um terreno fértil para o desenvolvimento de aplicativos móveis inovadores. Conforme destacado por Johnson et al. (2013), a explosão de aplicativos móveis trouxe consigo uma revolução na forma como as pessoas interagem com a tecnologia, oferecendo soluções personalizadas para uma ampla gama de necessidades do usuário.

Nesse contexto, surge o Remedium, um aplicativo móvel desenvolvido para dispositivos Android, projetado para oferecer lembretes de medicamentos por meio de notificações push. A necessidade de uma gestão eficaz da medicação é reconhecida como crucial para o bem-estar e a saúde dos pacientes (Santos et al., 2018). No entanto, muitas vezes, a adesão à medicação prescrita é desafiadora, especialmente em casos de tratamentos de longo prazo ou múltiplos medicamentos (Márquez Contreras et al., 2019).

Os aplicativos surgem como uma resposta a essa lacuna, oferecendo uma solução simples e acessível para ajudar os usuários a lembrarem-se de tomar seus medicamentos de forma pontual e eficaz. Este trabalho busca explorar o desenvolvimento e a implementação, examinando seu impacto potencial na adesão à medicação e no bem-estar dos usuários.

Por meio de uma análise abrangente, pretende-se destacar não apenas a funcionalidade técnica do aplicativo, mas também sua relevância e potencial contribuição para a saúde individual. Ao longo deste trabalho, serão discutidos os aspectos teóricos e práticos relacionados ao desenvolvimento de aplicativos móveis de saúde, bem como os desafios e oportunidades encontrados durante o processo de criação.

A coleta de dados para a pesquisa sobre o aplicativo será realizada por meio de um questionário estruturado, visando obter dados quantitativos sobre diversos aspectos do aplicativo. O questionário abordará critérios relacionados à facilidade de uso do aplicativo, sua importância percebida pelos usuários, bem como o impacto positivo na saúde decorrente do seu uso. Além disso, será investigado se o aplicativo tem contribuído efetivamente para lembrar os usuários de tomarem seus medicamentos no horário correto.

Este estudo visa contribuir para o corpo de conhecimento existente no campo dos aplicativos móveis de saúde, fornecendo insights valiosos para desenvolvedores, profissionais de saúde e pesquisadores interessados na promoção da adesão à medicação e no uso eficaz da tecnologia móvel para melhorar os resultados de saúde.

1.1 Problemática

A administração correta de medicamentos é vital para a eficácia dos tratamentos, mas muitos pacientes enfrentam dificuldades em lembrar de tomar seus remédios nos horários prescritos, especialmente aqueles com regimes complexos ou que necessitam de múltiplas doses ao longo do dia. A falta de adesão aos horários de medicação pode levar a consequências graves, como a piora da condição clínica, hospitalizações desnecessárias e aumento nos custos de saúde. Métodos tradicionais de lembretes, como alarmes de relógio ou notas, muitas vezes não são suficientes para garantir a consistência necessária. Diante disso, como podemos melhorar a adesão aos horários de medicação de forma eficaz e acessível para todos os pacientes?

1.2 Justificativa

A justificativa para o desenvolvimento do aplicativo Remedium baseia-se na necessidade crescente de ferramentas eficazes para auxiliar a gerenciar seus regimes de medicação. A dificuldade em lembrar de tomar remédios nos horários corretos pode comprometer a eficácia do tratamento e a saúde do cidadão, especialmente em casos de doenças crônicas que exigem adesão rigorosa aos horários de medicação. O Remedium visa proporcionar uma solução prática e acessível, utilizando notificações push para enviar lembretes personalizados e

oportunos, promovendo assim uma melhor adesão ao tratamento e, consequentemente, melhores resultados de saúde. Ao facilitar o gerenciamento dos horários de medicação, o Remedium contribuirá para a melhoria da qualidade de vida e a otimização dos tratamentos médicos.

1.3 Objetivo

Desenvolver um aplicativo mobile para lembrete de remédios utilizando notificação push.

1.4 Objetivos específicos

- Criar interface intuitiva (React Native);
- Desenvolver API (NestJS)
- Utilizar banco PostgreSQL
- Enviar notificações programadas
- Hospedar e organizar infraestrutura (Docker, Traefik, Contaboo)

2 FUNDAMENTAÇÃO TEÓRICA

Para codificar as soluções, utilizei o Visual Studio Code (VS Code) como meu ambiente de desenvolvimento integrado (IDE). Segundo a documentação oficial do VS Code, "Visual Studio Code (VS Code) é um editor de código-fonte leve, porém poderoso, que suporta desenvolvimento em diversas linguagens de programação. Com recursos como depuração, controle de versão integrado e uma vasta coleção de extensões, o VS Code é amplamente utilizado por desenvolvedores para escrever, depurar e gerenciar código" (MICROSOFT, 2024).

2.1 Javascript

JavaScript é uma linguagem de programação versátil, amplamente utilizada no desenvolvimento de páginas web interativas e dinâmicas. Criada inicialmente por Brendan Eich em 1995, ela se tornou uma das tecnologias fundamentais da web, junto com HTML e CSS. (EICH, 1995).

2.2 TypeScript

TypeScript é uma linguagem de programação de código aberto desenvolvida pela Microsoft. É um superconjunto de JavaScript que adiciona tipos estáticos, permitindo a detecção de erros em tempo de compilação e melhorando a manutenção de grandes bases de código. (MICROSOFT, 2023).

2.3 React Native

React Native é uma estrutura de desenvolvimento de software para aplicativos móveis criada pelo Facebook. Ela permite que desenvolvedores utilizem JavaScript e React para construir aplicações para iOS e Android com uma base de código única, resultando em maior eficiência e consistência entre plataformas. (Facebook, 2015). O framework combina desenvolvimento web do React com a funcionalidade nativa dos dispositivos móveis, permitindo aos desenvolvedores criar interfaces de usuário dinâmicas e responsivas que se comportam como aplicativos nativos.

Ao usar React Native, os desenvolvedores podem escrever uma base de código comum que pode ser compartilhada entre plataformas, economizando tempo e esforço no desenvolvimento e manutenção de aplicativos móveis para diferentes sistemas operacionais. Isso é possível porque o React Native utiliza componentes nativos fornecidos pelas plataformas iOS e Android, garantindo um desempenho rápido e uma experiência de usuário nativa.

Além disso, React Native oferece um ambiente de desenvolvimento flexível e poderoso, com suporte a hot reloading, que permite aos desenvolvedores verem instantaneamente as alterações no código refletidas no aplicativo em execução, agilizando o processo de desenvolvimento e teste.

2.4 Front-end

De forma geral, segundo Alura (2023), o front-end corresponde à camada visual de sites e aplicações, isto é, a área das páginas acessível à interação dos usuários. No âmbito do desenvolvimento de aplicativos móveis, o front-end refere-se à interface com a qual os usuários mantêm contato direto, abrangendo a interface do usuário (UI) e a experiência do usuário (UX). Esse componente é responsável por

apresentar dados e informações de maneira atrativa e intuitiva, garantindo usabilidade e facilitando a interação com o sistema.

2.5 Back end

O back-end refere-se à parte do desenvolvimento web responsável pela lógica de negócios, integração com bancos de dados, autenticação de usuários, entre outros. É essencial para o funcionamento de qualquer aplicação web, pois lida com as operações no servidor e garante que os dados sejam processados e entregues corretamente ao front-end. (PATEL, 2021). Em outras palavras, o back-end é responsável por gerenciar o funcionamento interno de uma aplicação, lidando com a lógica de processamento de dados e a interação com o banco de dados.

No contexto do desenvolvimento de aplicativos móveis, o back-end desempenha um papel fundamental na comunicação entre o aplicativo e os servidores onde os dados são armazenados. Ele fornece uma interface de programação de aplicativos (API) que permite ao front-end acessar e manipular os dados de forma segura e eficiente.

O back-end é essencial para garantir o funcionamento adequado de uma aplicação, especialmente em termos de segurança, escalabilidade e desempenho. Ele pode ser desenvolvido utilizando uma variedade de tecnologias, incluindo frameworks como Node.js, Django, Ruby on Rails, entre outros.

2.6 NestJS

NestJS é um framework para a construção de aplicativos do lado do servidor eficientes e escaláveis com Node.js. Utiliza TypeScript como linguagem padrão e é fortemente inspirado pelo Angular, proporcionando uma arquitetura modular que facilita a criação de aplicações complexas." (MYŚLIWIEC, 2024).

Como a documentação diz:

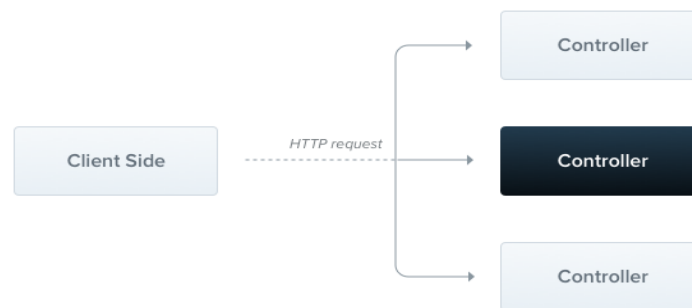
"Nest (NestJS) é uma estrutura para a construção de aplicativos Node.js do lado do servidor eficientes e escaláveis. Ele usa JavaScript progressivo, é construído e oferece suporte total a TypeScript (mas ainda permite que os desenvolvedores codifiquem em JavaScript puro) e combina elementos de OOP (Programação Orientada a Objetos), FP (Programação Funcional) e

FRP (Programação Funcional Reativa)”. (MYŚLIWIEC, 2024)

Ele utiliza por padrão typescript, que é um superset para javascript, adicionando tipagem forte e várias funcionalidades modernas linguagem javascript. Com ele é possível desenvolver uma API moderna seguindo padrões modernos, facilitando assim a criação e manutenção da API.

A arquitetura do NEST é baseada em módulos, com uma camada de controller e services. Controller são responsáveis por lidar com as solicitações recebidas e de retornar uma resposta para o cliente (MYŚLIWIEC, 2024). E dentro de cada controller existe um service, que executa uma lógica para aquele controller.

Figura 1 - Client Side Request in NestJS



Fonte: NestJs

2.7 Contabo

A infraestrutura do aplicativo foi hospedada na Contabo, provedora de serviços em nuvem que disponibiliza soluções escaláveis e de baixo custo. A Contabo oferece uma gama de soluções, incluindo Cloud VPS e Dedicated Servers, destacando-se por seus preços competitivos e a promessa de "mais desempenho pelo preço do que em qualquer outro lugar". Além disso, a plataforma enfatiza a fácil escalabilidade de seus Cloud VDS, oferecendo opções como Virtual Private Servers com NVMe rápido e Bare Metal Servers para alto poder de computação (CONTABO, 2025).

2.8 Docker

O Docker foi empregado para empacotar e distribuir a API desenvolvida em NestJS e o banco de dados PostgreSQL, assegurando portabilidade e consistência entre ambientes. Segundo a documentação oficial do Docker, "O Docker é uma plataforma de código aberto que permite aos desenvolvedores automatizar a implantação de aplicações em contêineres, garantindo que o software seja executado de maneira uniforme e consistente em qualquer ambiente" (DOCKER, 2024).

Dentro de uma vps (Virtual Private Server) da Hetzner podemos ter vários serviços simultâneos, o docker oferece uma CLI para criar, excluir, editar, inspecionar e gerenciar nossos containers dockers, porém para gerenciar múltiplos containers é muito trabalhoso por linha de comando, por isso utilizei o Portainer para gerenciar e monitorar os contêineres Docker. Segundo a documentação, Portainer é uma plataforma de gerenciamento de contêineres que oferece uma interface de usuário gráfica (GUI) fácil de usar para implantar, gerenciar e solucionar problemas de ambientes de contêineres Docker e Kubernetes (PORTAINER, 2024).

2.9 Traefik

Para o acesso dos serviços de forma remota, é preciso utilizar um serviço de proxy reverso, na qual irá redirecionar todo o tráfego na porta 443 para o serviço escolhido com base no nome de domínio. O gerenciamento do tráfego entre os serviços em contêineres foi realizado por meio do Traefik, atuando como proxy reverso e balanceador de carga. Segundo a documentação oficial, a ferramenta é um proxy reverso moderno e um balanceador de carga projetado para facilitar a implementação de micros serviços, roteando tráfego automaticamente para serviços em execução em uma infraestrutura de contêineres (TRAEFIK, 2024).

2.13 Easy Panel

O Easy Panel foi empregado como ferramenta de gerenciamento da VPS, permitindo a administração do banco de dados PostgreSQL e da API desenvolvida em NestJS. Segundo a documentação oficial, O Easy Panel é um painel de controle de servidor de código aberto que facilita a administração e gerenciamento de servidores, permitindo a instalação e gerenciamento de serviços como bancos de

dados PostgreSQL e APIs desenvolvidas com frameworks como NestJS. (EASY PANEL, 2024).

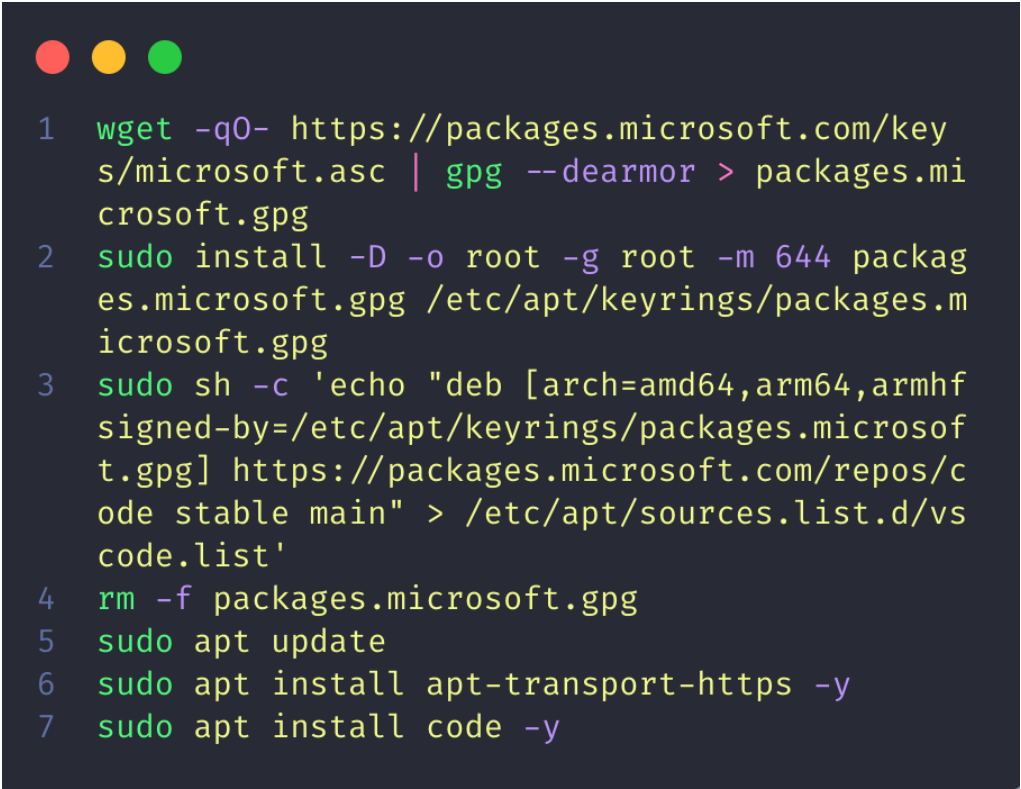
3 PROCEDIMENTOS METODOLÓGICOS

O presente projeto teve como finalidade desenvolver uma aplicação mobile para lembrete de remédios, o desenvolvimento do aplicativo móvel contou com o uso do framework Expo, que se baseia na biblioteca React para a criação de interfaces reativas. O app vai consumir uma API feita em NestJS conectada a um banco de dados PostgreSQL, a API e o banco de dados estão hospedados em um servidor em nuvem com sistema operacional Ubuntu 20.04.6.

3.1 Ambiente de desenvolvimento

Para o ambiente de desenvolvimento primeiro precisamos instalar o Visual Studio Code, podemos instalar com o seguinte bash em um sistema operacional Ubuntu 20.04.6 :

Figura 2 - Instalar vs code



```
1 wget -qO- https://packages.microsoft.com/keys/microsoft.asc | gpg --dearmor > packages.microsoft.gpg
2 sudo install -D -o root -g root -m 644 packages.microsoft.gpg /etc/apt/keyrings/packages.microsoft.gpg
3 sudo sh -c 'echo "deb [arch=amd64,arm64,armhf signed-by=/etc/apt/keyrings/packages.microsoft.gpg] https://packages.microsoft.com/repos/code stable main" > /etc/apt/sources.list.d/vscode.list'
4 rm -f packages.microsoft.gpg
5 sudo apt update
6 sudo apt install apt-transport-https -y
7 sudo apt install code -y
```

Fonte: autor

Com isso teremos um editor de texto, para codificar a aplicação, será necessário ter node.js instalado, um runtime javascript. Tanto React Native quanto Nest.js utilizam node para o processo de desenvolvimento, podemos instalar node facilmente com o seguinte bash:

Figura 3 - Instalar nvm

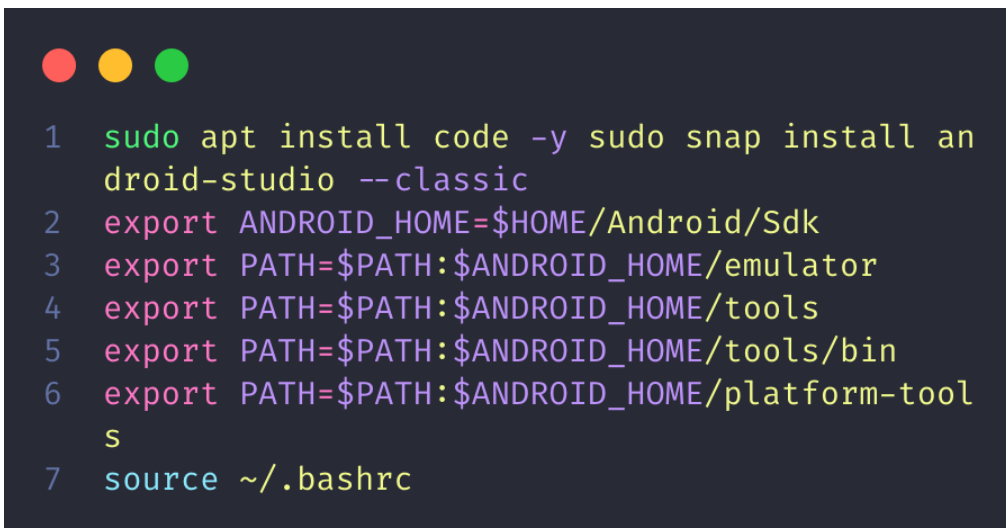
A terminal window with a dark background and three colored window control buttons (red, yellow, green) in the top-left corner. It displays two lines of code: line 1 is 'curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.39.0/install.sh | bash' and line 2 is 'nvm install 18.17.0'.

```
1 curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.39.0/install.sh | bash
2 nvm install 18.17.0
```

Fonte: autor

Para desenvolver aplicativo para android é necessário ter android studio instalado, bem como suas configurações e variáveis de ambiente, o React Native utiliza as ferramentas de build do android studio para construir e configurar a parte nativa da aplicação, podemos configurar e instalar com o seguinte bash:

Figura 4 - Instalar android studio

A terminal window with a dark background and three colored window control buttons (red, yellow, green) in the top-left corner. It displays seven lines of code: line 1 is 'sudo apt install code -y sudo snap install android-studio --classic', line 2 is 'export ANDROID_HOME=\$HOME/Android/Sdk', line 3 is 'export PATH=\$PATH:\$ANDROID_HOME/emulator', line 4 is 'export PATH=\$PATH:\$ANDROID_HOME/tools', line 5 is 'export PATH=\$PATH:\$ANDROID_HOME/tools/bin', line 6 is 'export PATH=\$PATH:\$ANDROID_HOME/platform-tools', and line 7 is 'source ~/.bashrc'.

```
1 sudo apt install code -y sudo snap install android-studio --classic
2 export ANDROID_HOME=$HOME/Android/Sdk
3 export PATH=$PATH:$ANDROID_HOME/emulator
4 export PATH=$PATH:$ANDROID_HOME/tools
5 export PATH=$PATH:$ANDROID_HOME/tools/bin
6 export PATH=$PATH:$ANDROID_HOME/platform-tools
7 source ~/.bashrc
```

Fonte: autor

Por fim para a criação do back-end com NestJS, precisamos instalar a cli e o docker, que vai ser responsável para executar nosso banco de dados, podemos instalar com o seguinte bash:

Figura 5 - Instalar docker e nestjs cli



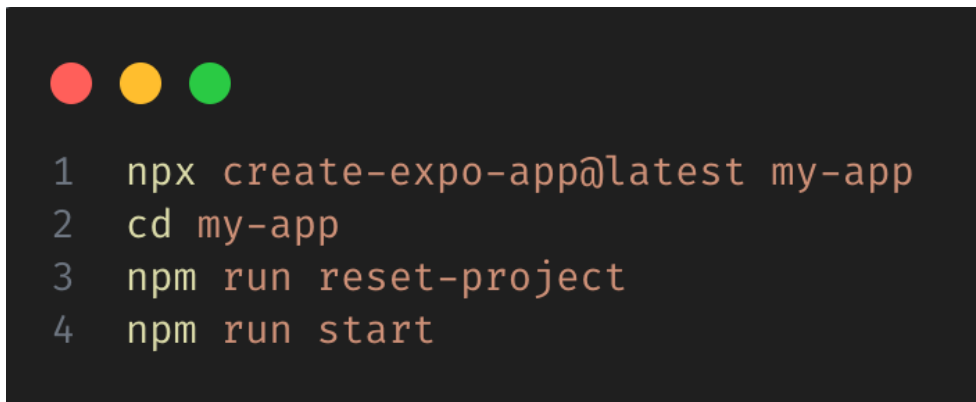
```
1 curl -fsSL https://get.docker.com -o get-docker.sh
2 sudo sh get-docker.sh
3 npm install -g @nestjs/cli
```

Fonte: autor

3.2 Mobile Front-End

Primeiramente precisamos criar um projeto React Native, a forma mais simples é utilizando a expo cli, na qual é criado um projeto pré-configurado com tudo que iremos precisar no desenvolvimento.

Figura 6 - Criar projeto react native



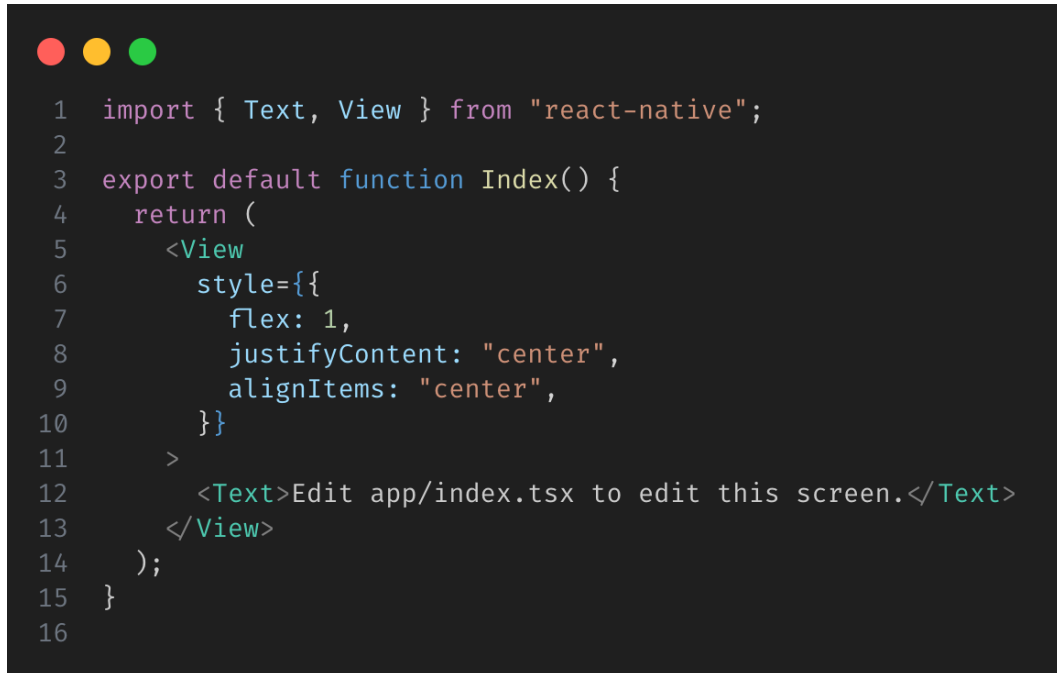
```
1 npx create-expo-app@latest my-app
2 cd my-app
3 npm run reset-project
4 npm run start
```

Fonte: autor

Com a criação do projeto, teremos uma estrutura básica para codificar as funções. No React Native existem componentes básicos para criação de interfaces, a maior parte do app foi construído em cima de 3 componentes básicos: View, Text e TextInput, onde criamos uma função que retorna as tags.

A criação de interfaces é semelhante à web, onde várias tags podem formar uma interface, com duas formas de definir estilos, *inline* ou com StyleSheet. Inline é o estilo escrito justamente com a tag, igual exemplo a seguir:

Figura 7 - Tela básica



```
1  import { Text, View } from "react-native";
2
3  export default function Index() {
4    return (
5      <View
6        style={{
7          flex: 1,
8          justifyContent: "center",
9          alignItems: "center",
10         }}
11      >
12        <Text>Edit app/index.tsx to edit this screen.</Text>
13      </View>
14    );
15  }
16
```

Fonte: autor

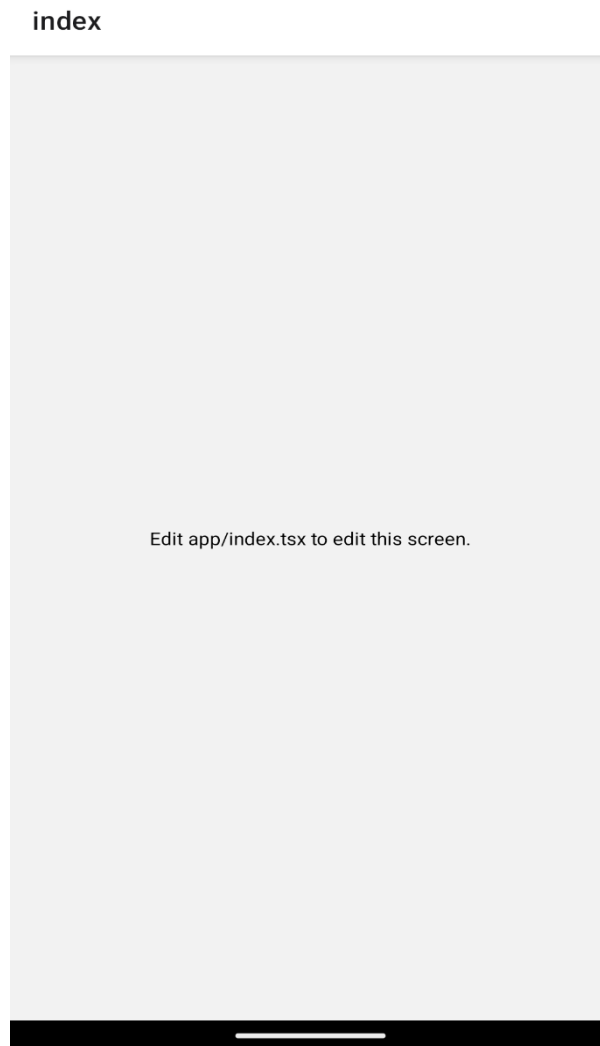
Já o Stylesheet, é uma maneira de definir estilos em um objeto separado, tornando o código mais organizado e reutilizável, podendo armazenar esse objeto em arquivos ou pasta diferentes para melhor organização e legibilidade de código.

Figura 8 - Estilização

```
1  import { Text, View, StyleSheet } from "react-native";
2
3  export default function Index() {
4    return (
5      <View style={styles.container}>
6        <Text>Edit app/index.tsx to edit this screen.</Text>
7      </View>
8    );
9  }
10
11  const styles = StyleSheet.create({
12    container: {
13      flex: 1,
14      justifyContent: "center",
15      alignItems: "center",
16    },
17  });
18
```

Fonte: autor

Com estes componentes básicos é possível construir qualquer interface no React Native, o código apresentado gera a seguinte tela.

Figura 9 - Interface

Fonte: autor

Construir interfaces com React Native é extremamente fácil e auto intuitivo se você já estiver experiência com tags html e css.

3.3 Back-end

O back-end foi criado utilizando uma framework JavaScript com nome de NestJS, para iniciar um projeto podemos utilizar o seguinte comando no terminal:

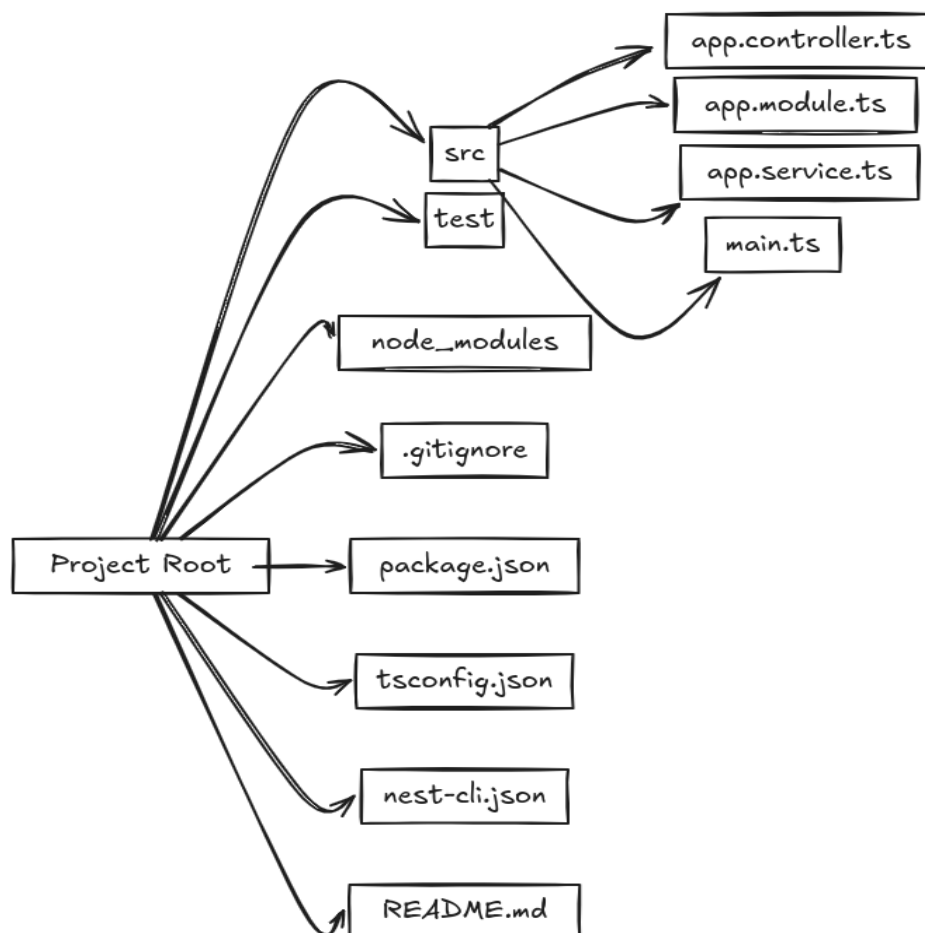
Figura 10 - Instalar NestJS cli

```
1  npm i -g @nestjs/cli
2  nest new project-name
3  cd project-name
4  npm run start
```

Fonte: autor

O NestJs segue uma estrutura de pastas modular orientada a arquivos, a imagem seguinte apresenta-se a estrutura de diretórios de um projeto básico:

Figura 11 - Estrutura da API



Fonte: autor

A Estrutura básica do NestJS se resume em 3 tipos de arquivos: controller, module, service e um arquivo main. Esse tipo de estrutura divide a responsabilidade da aplicação em camadas. Os controllers lidam com as requisições HTTP e respondem aos clientes, neste arquivo, onde serão configuradas as rotas do tipo: GET, POST, PUT, DELETE entre outros métodos HTTP.

O service encapsula a lógica de regra de negócio da aplicação, e são utilizadas dentro de um controller para manipular os dados e aplicar regras de negócio.

O module é o principal bloco na estrutura do NestJS, ele é responsável por organizar a aplicação em módulos, onde cada modulo tem sua estrutura única de controller, service e module. Esta estrutura possibilita que diferentes módulos possam ser reutilizados dentro de outros módulos e partes da aplicação, tornando a aplicação modular e escalável.

O arquivo main é o ponto de partida da aplicação, é onde o NestJS é iniciado. Neste arquivo que é feita conexão com banco de dados e outras aplicações externas, e é onde é aplicado configurações para o back-end.

3.4 Implementação das Notificações Push

A implementação das notificações push no aplicativo Remedium foi realizada por meio da plataforma OneSignal, escolhida por oferecer uma solução robusta, escalável e amplamente utilizada para o envio de notificações em aplicações móveis. A integração foi desenvolvida no back-end utilizando o framework NestJS, permitindo o controle centralizado do disparo das notificações de acordo com os alertas configurados pelos usuários.

No aplicativo mobile, o dispositivo do usuário é registrado junto ao serviço OneSignal, gerando um identificador único associado ao usuário autenticado. Esse identificador é armazenado no banco de dados e utilizado posteriormente para direcionar as notificações de forma individualizada. Dessa forma, cada usuário recebe apenas os lembretes correspondentes aos seus próprios alertas.

No back-end, foi desenvolvido um serviço específico responsável pela comunicação com a API do OneSignal. Esse serviço utiliza credenciais configuradas por meio de variáveis de ambiente, garantindo maior segurança e flexibilidade na gestão das chaves de acesso. O envio das notificações é realizado a partir da criação

de mensagens contendo título, subtítulo, corpo da notificação e dados adicionais, que são transmitidos ao OneSignal para distribuição aos dispositivos cadastrados.

3.4.1 Agendamento e Verificação dos Alertas

Para garantir que os lembretes sejam enviados no momento correto, foi implementado um mecanismo de verificação periódica utilizando o módulo de agendamento do NestJS. A cada intervalo de tempo definido, o sistema executa uma rotina responsável por analisar todos os alertas ativos armazenados no banco de dados.

Durante essa verificação, o sistema avalia o tipo de alerta configurado pelo usuário, podendo ser por intervalo de tempo, diário, semanal ou por data específica. Com base nessas informações, o sistema determina se o alerta deve ser disparado naquele momento, acionando o serviço de envio de notificações quando as condições são atendidas.

Essa abordagem permite que o controle do envio das notificações seja realizado integralmente no servidor, garantindo confiabilidade mesmo quando o aplicativo não está em execução no dispositivo do usuário.

3.4.2 Tratamento de Fuso Horário (Timezone)

Um dos principais desafios na implementação de notificações baseadas em horário é o tratamento correto do fuso horário. O servidor responsável pela execução do back-end encontra-se hospedado em um ambiente de nuvem configurado com fuso horário diferente do utilizado pelos usuários finais, que podem estar em diversas regiões geográficas.

Para evitar erros no disparo das notificações, o aplicativo coleta o fuso horário do dispositivo do usuário no momento do cadastro e o envia para o back-end, onde essa informação é armazenada juntamente com os dados do usuário. Durante o processo de verificação dos alertas, o sistema converte o horário atual do servidor para o fuso horário específico de cada usuário antes de realizar qualquer cálculo ou comparação de datas e horários.

Essa conversão garante que os alertas sejam disparados exatamente no horário esperado pelo usuário, independentemente da localização física do servidor. Dessa forma, o sistema mantém a precisão dos lembretes e evita atrasos ou

antecipações no envio das notificações, assegurando maior confiabilidade e aderência ao tratamento medicamentoso.

3.4.3 Envio das Notificações Push

Quando um alerta é identificado como válido para disparo, o sistema aciona o serviço de envio de notificações, que se comunica com a API do OneSignal. A notificação é então enviada diretamente ao dispositivo do usuário correspondente, contendo informações claras sobre o medicamento, instruções e horário do lembrete.

Após o envio, o sistema atualiza os registros do alerta no banco de dados, garantindo que o mesmo lembrete não seja enviado repetidamente de forma indevida. Essa lógica assegura o controle correto dos intervalos e a integridade do funcionamento do sistema de notificações.

3.5 Arquitetura Geral do Sistema

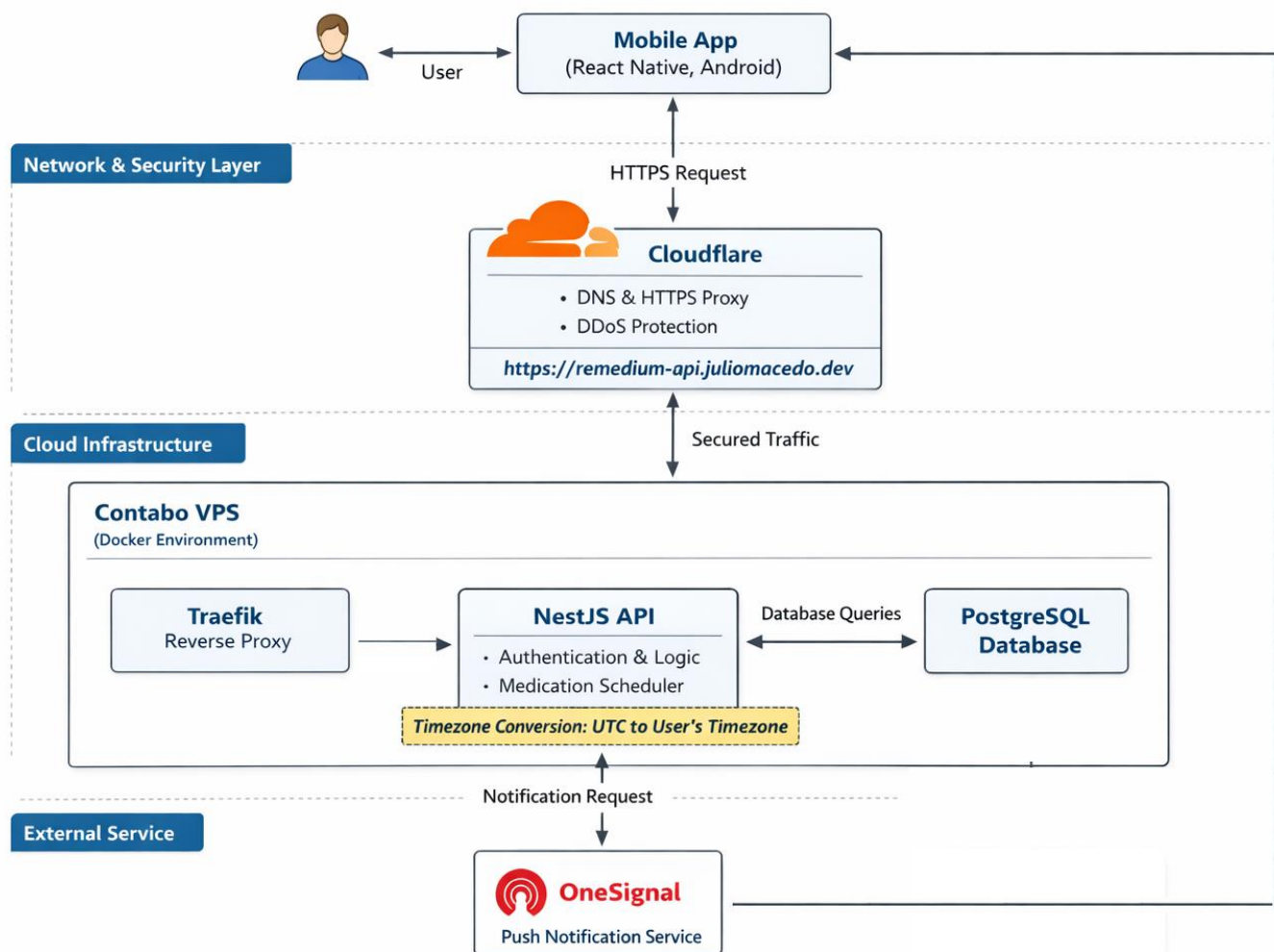
A arquitetura do sistema Remedium foi projetada com foco em modularidade, escalabilidade e segurança, adotando uma abordagem baseada em serviços distribuídos, ainda que hospedados em uma única infraestrutura de nuvem.

Adicionalmente, o sistema utiliza a plataforma Cloudflare como camada intermediária de segurança e performance. Os domínios da aplicação são gerenciados pela Cloudflare, que atua como proxy reverso externo, oferecendo recursos como proteção contra ataques de negação de serviço (DDoS), cache de conteúdo, ocultação do endereço IP real do servidor e otimização do tráfego HTTPS. Essa abordagem contribui para aumentar a disponibilidade, a segurança e a confiabilidade da aplicação.

A arquitetura adotada permite que todos os componentes do back-end operem de forma integrada e eficiente, mantendo clareza na separação de responsabilidades e possibilitando futuras expansões, como escalabilidade horizontal dos serviços ou distribuição da infraestrutura em múltiplos servidores. Presando pela máxima segurança, o bando de dados postgre não é exposto na internet, sendo somente acessível internamente na rede docker, como a api NestJS está na mesma rede docker, ela consegue se comunicar de forma segura e rápida com o banco de dados.

A seguir na figura 12 um diagrama resume o design system do sistema Remedium:

Figura 12 - Arquitetura geral do sistema Remedium



Fonte: autor

4 RESULTADOS

O aplicativo Remedium foi testado em ambiente Android, demonstrando funcionamento adequado em toda as suas principais funcionalidades. A seguir, apresentam-se os resultados por meio de capturas de tela obtidas durante a utilização prática do sistema.

4.1 Acesso ao Sistema

O sistema permite que o usuário realize login, como mostra a figura 12 a seguir:

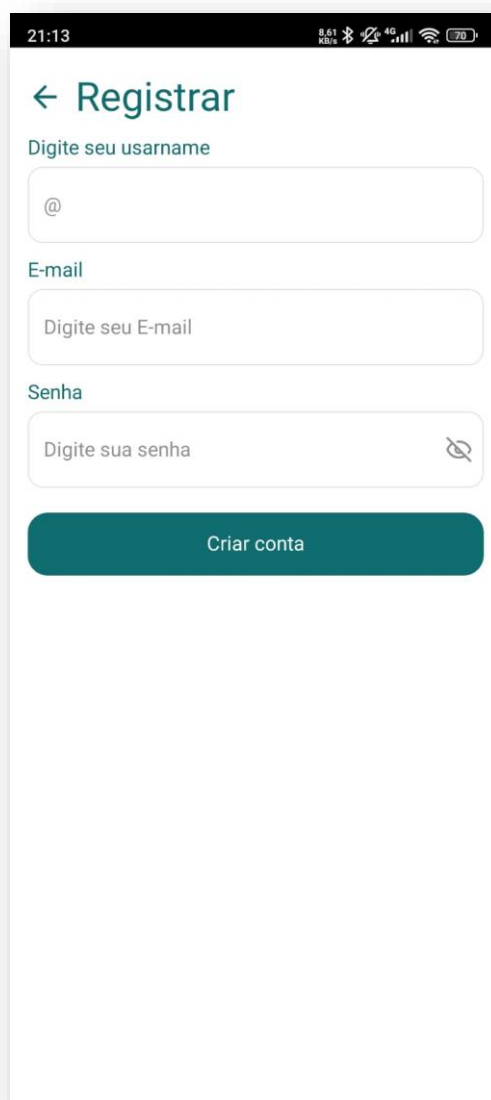
Figura 13 - Login

A mobile application login screen. At the top, a status bar shows the time 21:13, 77.8% battery, and various connectivity icons. The screen has a white background with a teal header area. The word "Login" is displayed in teal. Below it, the label "E-mail" is followed by a text input field with the placeholder "Digite seu E-mail". The label "Senha" is followed by a text input field with the placeholder "Digite sua senha" and a small eye icon to toggle password visibility. Below the password field is a link "Esqueceu a senha?". At the bottom, there are two buttons: a teal button labeled "Entrar" and a white button with a teal border labeled "Criar conta".

Fonte: autor

ou crie uma conta como demonstra na figura 13 a seguir garantindo controle individualizado de alertas de medicação.

Figura 14 - Register

A mobile application registration screen titled 'Registrar' with a back arrow. It features three input fields: 'Digite seu username' with an '@' icon, 'E-mail' with the placeholder 'Digite seu E-mail', and 'Senha' with the placeholder 'Digite sua senha' and a toggle icon. A teal 'Criar conta' button is at the bottom.

21:13 8.61 KB/s 4G 70

← Registrar

Digite seu username

@

E-mail

Digite seu E-mail

Senha

Digite sua senha

Criar conta

Fonte: autor

Para assegurar a privacidade dos dados e a integridade das informações trocadas entre cliente e servidor, foi implementado um mecanismo de autenticação segura baseada em JSON Web Token (JWT). Essa abordagem, amplamente utilizada em arquiteturas modernas, consiste na geração de um token assinado digitalmente pelo servidor, o qual é enviado ao cliente após a validação das credenciais.

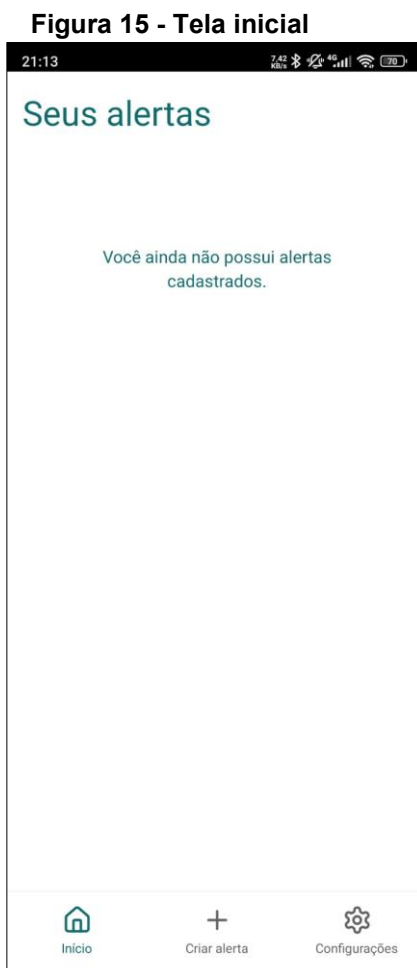
A cada requisição subsequente, o cliente anexa o token no cabeçalho da requisição, permitindo que o servidor autentique o usuário de forma eficiente e sem a

necessidade de retransmitir a senha. Essa estratégia proporciona maior segurança contra-ataques de interceptação e session hacking, além de favorecer a escalabilidade do sistema, visto que o JWT é autossuficiente para armazenar as informações de autenticação.

4.2 Cadastro e Gerenciamento de Alertas

Após o cadastro e a realização do login com suas credenciais, você será direcionado para página inicial da app. Nesta tela o usuário é informado caso ainda não possua alertas cadastrados, e caso possua, mostrará seus alertas.

O aplicativo conta com uma barra de navegação na parte inferior com as opções de início, criar alerta e configurações como é demonstrado pela figura:



Fonte: autor

Acessando a barra de navegação inferior denominada de “Criar Alerta” o usuário poderá cadastrar seus alertas, nesta página é possível registrar dados como

nome do medicamento, dosagem, instruções e intervalo de uso como demonstra na figura 15:

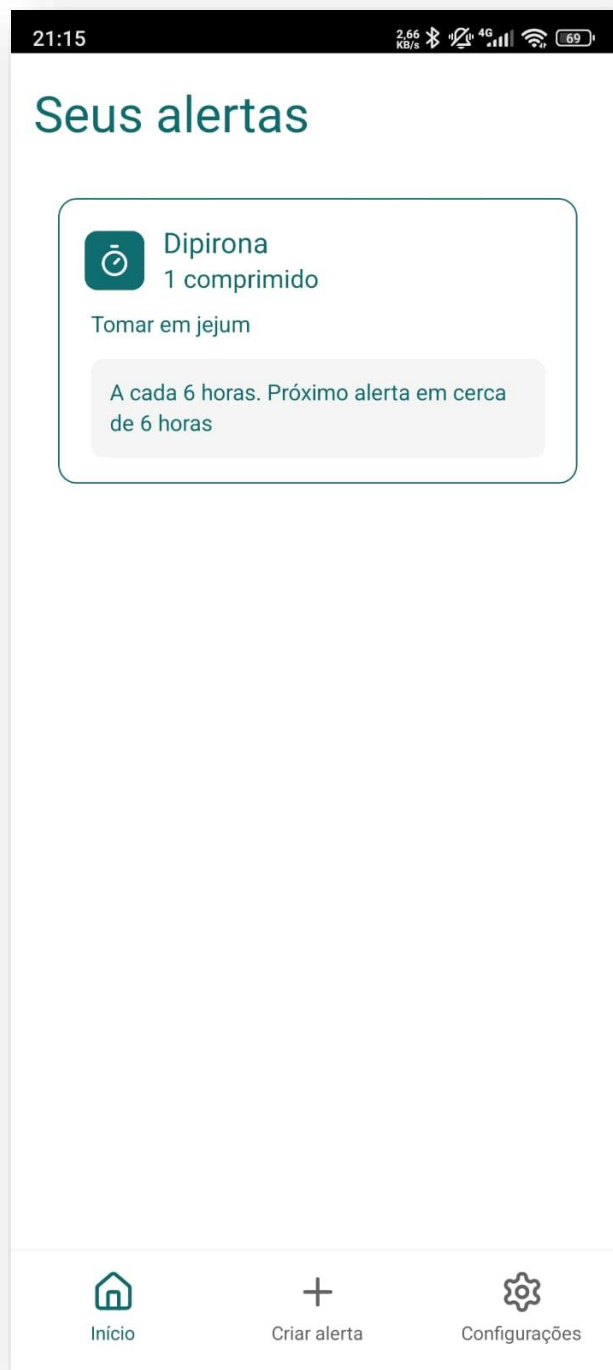
Figura 16 - Criar alerta

The screenshot displays a mobile application interface for creating a medication alert. At the top, the status bar shows the time 21:15, data speed 0.42 KB/s, 4G connectivity, and 69% battery. The main title is 'Criar alerta' in a large, dark teal font. Below the title, there are four input fields with teal labels: 'Nome do remédio' (containing 'Dipirona'), 'Dosagem' (containing '1 comprimido'), 'Instruções' (containing 'Tomar em jejum'), and 'Tipo de alerta'. The 'Tipo de alerta' field is a dropdown menu currently showing 'Intervalo'. Below this, the 'Intervalo de alerta' section contains a button labeled 'A cada 6 horas'. At the bottom of the form is a large teal button labeled 'Criar alerta'. The bottom navigation bar features three icons: a house for 'Início', a plus sign for 'Criar alerta' (which is highlighted in teal), and a gear for 'Configurações'.

Fonte: autor

Após o cadastro todos os seus alertas irão aparecer na tela inicial do aplicativo em cards com um pequeno resumo sobre o alerta, como mostra a figura a seguir:

Figura 17 - página inicial



Fonte: autor

4.3 Configurações do Aplicativo

No meu de configurações permite alterar tema, cor de interface, idioma e ativar/desativar notificações. Essa flexibilidade favorece a personalização da experiência de uso.

Figura 18 - Configurações

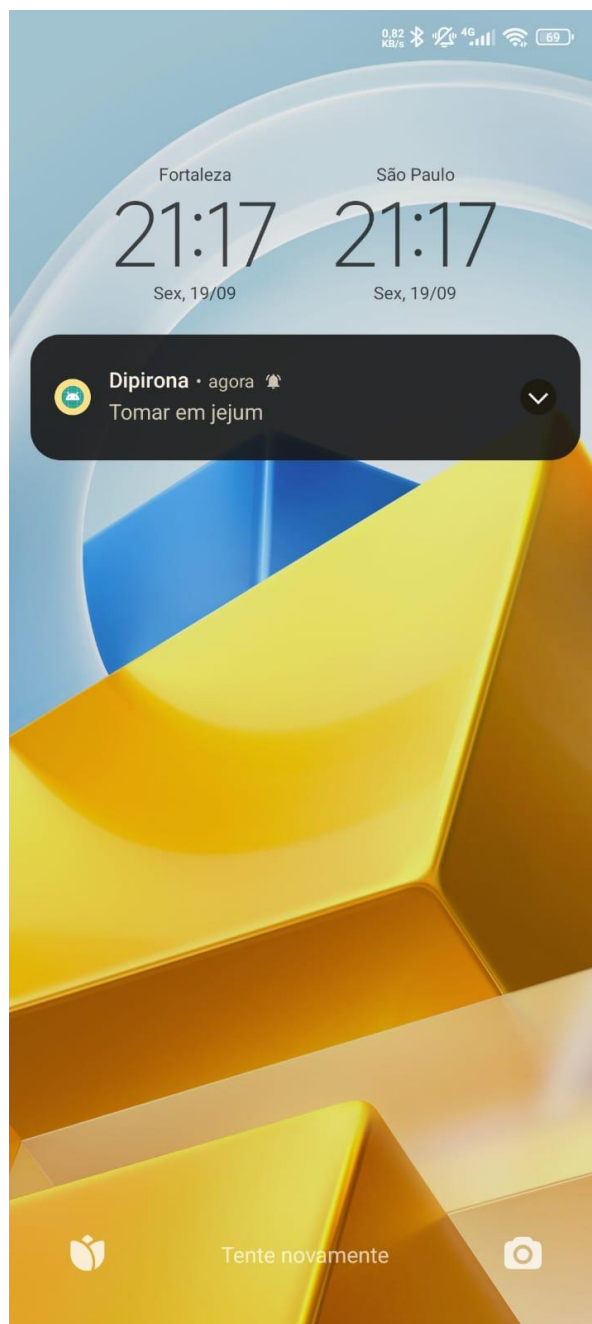


Fonte: autor

4.4 Notificações Push

O principal resultado esperado — envio de lembretes — foi alcançado. O aplicativo gerou notificações push no horário programado (Figura 18), assegurando a efetividade do sistema como ferramenta de apoio à adesão terapêutica.

Figura 19 - Push notification



Fonte: autor

5 CONCLUSÃO E TRABALHOS FUTUROS

O desenvolvimento do Remedium possibilitou a criação de um aplicativo móvel funcional para lembretes de medicamentos, atendendo ao objetivo proposto no início deste trabalho. Os resultados demonstraram que o sistema cumpre as funções básicas de cadastro, edição e exclusão de alertas, além de enviar notificações push nos horários configurados, o que pode contribuir diretamente para melhorar a adesão ao tratamento medicamentoso.

Como trabalhos futuros, destacam-se as seguintes possibilidades:

- Suporte multiplataforma: expansão para iOS, ampliando o alcance do aplicativo.
- Relatórios de uso: permitir que o usuário acompanhe estatísticas de adesão, como doses tomadas e esquecidas.
- Integração com profissionais de saúde: possibilitar que médicos ou cuidadores recebam relatórios de acompanhamento.
- Integração com dispositivos vestíveis: uso de smartwatches para alertas mais práticos.
- Recursos de inteligência artificial: análise de padrões de uso para sugerir melhores horários ou detectar riscos de esquecimento.

Dessa forma, conclui-se que o Remedium é uma solução viável, com potencial de evolução significativa, reforçando a importância de tecnologias móveis aplicadas à saúde digital.

REFERÊNCIAS

CONTABO. (2025) **Contabo Landing Page**. Disponível em: <<https://contabo.com/en/>>. Acesso em: 05 jun. 2025.

DELOITTE. (2019). **Global Mobile Consumer Survey: The Nordic Cut. Deloitte**. Disponível em: <<https://view.deloitte.nl/rs/502-WIB-308/images/deloitte-nl-tmt-gmcs2019-report.pdf>> Acesso em: 05 jun. 2025.

DOCKER. **Docker Documentation**. Disponível em: <<https://docs.docker.com>> Acesso em: 05 jun. 2025.

EASY PANEL. **Easy Panel Documentation**. Disponível em: <<https://easypanel.io/docs>> Acesso em: 05 jun. 2025.

EICH, Brendan. **JavaScript**. 1995. Disponível em: <<https://developer.mozilla.org/en-US/docs/Web/JavaScript>> Acesso em: 5 jun. 2025.

FACEBOOK. **React Native**. Disponível em: <<https://reactnative.dev/>> Acesso em: 5 jun. 2025.

JOHNSON, L. et al. (2013). **NMC Horizon Report: 2013 Higher Education Edition. The New Media Consortium**. Disponível em: <<https://eric.ed.gov/?id=ED559358>> Acesso em: 5 jun. 2025.

MÁRQUEZ CONTRERAS, E et al. (2019). **Intervenciones para mejorar la adherencia al tratamiento en pacientes con hipertensión arterial: revisión sistemática**. Atención Primaria, 51(2), 100-109.

MICROSOFT. **TypeScript: JavaScript with syntax for types**. 2023. Disponível em: <<https://www.typescriptlang.org/>> Acesso em: 5 jun. 2025.

MICROSOFT. **Visual Studio Code Documentation**. Disponível em: <<https://code.visualstudio.com/docs>> Acesso em: 05 jun. 2025.

MYŚLIWIEC, Kamil. **NestJS - A progressive Node.js framework**. 2023. Disponível em: <<https://nestjs.com/>> Acesso em: 5 jun. 2025.

NESTJS. **The open source platform designed for the future. Build enterprise**. Disponível em: <<https://nestjs.com/>> Publicado em 2017. Acesso em 9 abr. 2024.

PATEL, Praj. **What is Backend Development?**. 2021. Disponível em: <<https://www.freecodecamp.org/news/what-is-backend-development/>> Acesso em: 5 jun. 2025.

PORTAINER. **Portainer Documentation**. Disponível em: <<https://docs.portainer.io/>> Acesso em: 05 jun. 2025.

SANTOS, T., Oliveira, M., & Primo, R. (2018). **Adesão à terapia medicamentosa em pacientes com doenças crônicas. Revista de Pesquisa: Cuidado é Fundamental Online**, 10(2), 549-554.

SOUTO, Mauro. **O que é Front-end Back-end e Full Stack - aprenda as diferenças entre essas áreas**. Alura, 24 jan. 2023. Disponível em: <R> Acesso em: 09 jun. 2025.

TRAEFIK. **Traefik Documentation**. Disponível em: <<https://doc.traefik.io/traefik/>> Acesso em: 05 jun. 2025.

VOLTOLINI, Ramon. **Por que as notificações de aplicativos são importantes?** Disponível em: <<https://www.tecmundo.com.br/apps/43525-por-que-as-notificacoes-de-aplicativos-sao-importantes-.htm#ixzz30TQMPZ3O>> Acesso em: 1 abr. 2025.