

UNIVERSIDADE FEDERAL DO PARANÁ

FERNANDO GUILHERME GORSKI

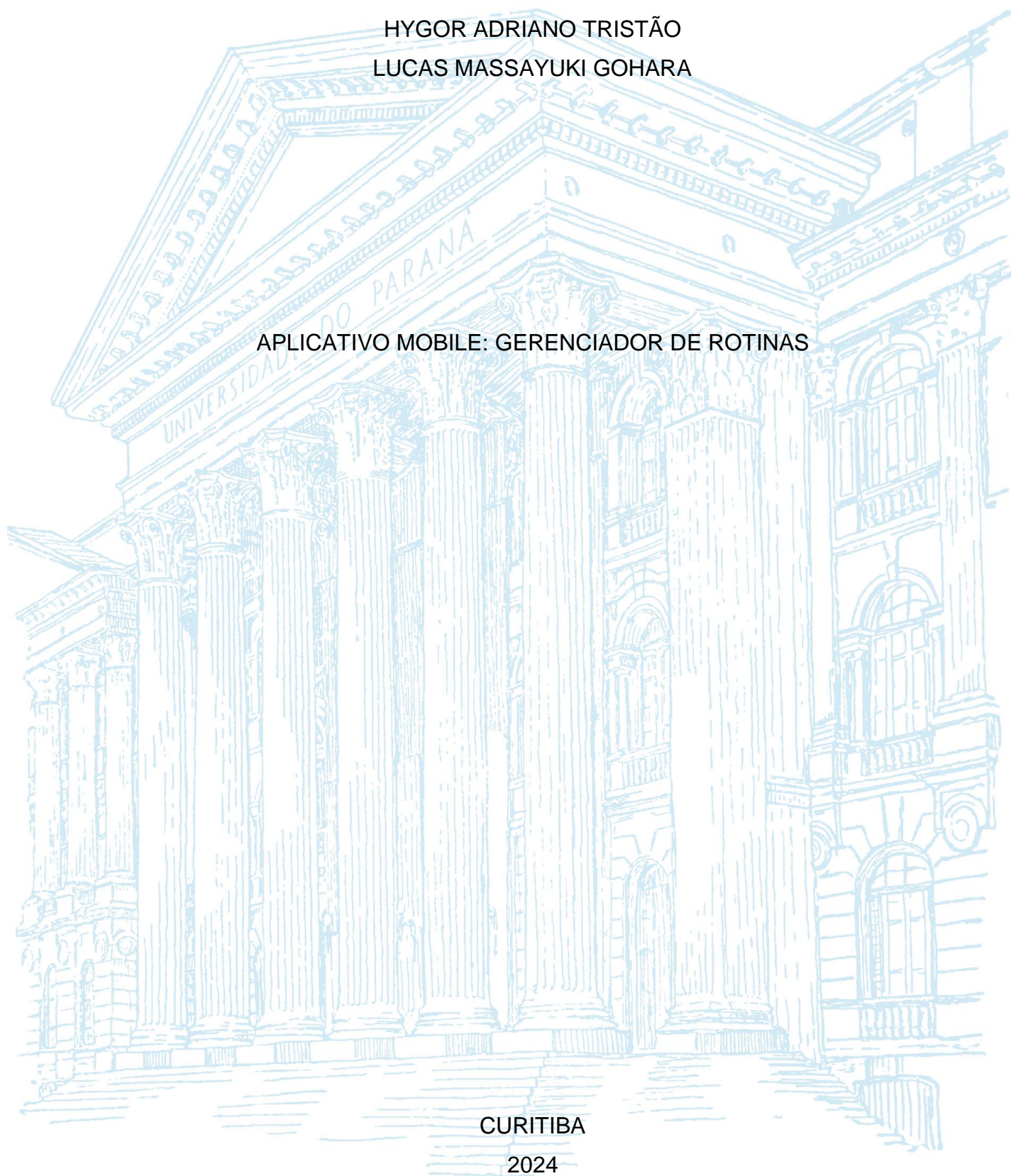
HYGOR ADRIANO TRISTÃO

LUCAS MASSAYUKI GOHARA

APLICATIVO MOBILE: GERENCIADOR DE ROTINAS

CURITIBA

2024



FERNANDO GUILHERME GORSKI  
HYGOR ADRIANO TRISTÃO  
LUCAS MASSAYUKI GOHARA

## APLICATIVO MOBILE: GERENCIADOR DE ROTINAS

Trabalho apresentado ao curso de Tecnologia em Análise e Desenvolvimento de Sistemas, no Setor de Educação Profissional e Tecnológica, na Universidade Federal do Paraná, sendo realizado no 1º Semestre de 2024, em cumprimento das exigências da disciplina Desenvolvimento para Dispositivos Móveis, como parte da nota parcial.

Orientador: Prof. Roberson Cesar Araujo.

CURITIBA

2024

## LISTA DE FIGURAS

FIGURA 1: Diagrama de Casos de Uso .....	8
FIGURA 2: Diagrama de Entidade Relacionamento .....	9
FIGURA 3: Protótipo - Tela de AutoCadastro.....	11
FIGURA 4: Protótipo - Tela de Login.....	12
FIGURA 5: Protótipo - Tela Inicial .....	13
FIGURA 6: Protótipo - Tela de Adicionar Tarefa .....	14
FIGURA 7: Protótipo - Tela de Selecionar Categoria .....	15
FIGURA 8: Protótipo - Tela de Selecionar Data e Hora .....	16
FIGURA 9: Protótipo - Tela de Selecionar Notificação .....	17
FIGURA 10: Protótipo - Tela de Adicionar Comentários .....	17
FIGURA 11 : Demonstração do uso de componentes React Native .....	18
FIGURA 12: Demonstração do uso de React Navigation.....	19
FIGURA 13: Demonstração do uso de StyleSheet.....	20
FIGURA 14: Demonstração do uso de Função Lambda .....	21
FIGURA 15: Demonstração do Uso de Streams .....	21
FIGURA 16: Demonstração de uso do Aplicativo 1 .....	22
FIGURA 17: Demonstração de uso do Aplicativo 2.....	23
FIGURA 18: Demonstração de uso do Aplicativo 3.....	24
FIGURA 19: Demonstração de uso do Aplicativo 4.....	25

## SUMÁRIO

<b>1 INTRODUÇÃO .....</b>	<b>5</b>
1.1 AMBIENTE DE DESENVOLVIMENTO .....	5
1.2 MEIOS DE COMUNICAÇÃO.....	6
<b>2 OBJETIVO DO PROJETO.....</b>	<b>7</b>
2.1 ESTRUTURA BÁSICA .....	7
2.1.1 Requisitos Mínimos .....	7
2.1.2 Diagrama de Casos de Uso .....	8
2.1.3 Diagrama Físico de Banco de Dados .....	8
<b>3 PROTOTIPAÇÃO.....</b>	<b>10</b>
<b>4 FUNCIONALIDADES DO SISTEMA.....</b>	<b>18</b>
4.1 CONCEITOS APLICADOS.....	18
4.1.1 Componentes React Native.....	18
4.1.2 React Navigation .....	19
4.1.3 StyleSheet.....	20
4.1.4 Função Lambda.....	21
4.1.5 Streams .....	21
<b>5 APRESENTAÇÃO DOS RESULTADOS .....</b>	<b>22</b>
<b>6 CONSIDERAÇÕES FINAIS .....</b>	<b>26</b>
<b>REFERÊNCIAS.....</b>	<b>27</b>
<b>ANEXO 1 – RELATÓRIO DE EXECUÇÃO DO PROJETO .....</b>	<b>28</b>

## 1 INTRODUÇÃO

No mundo agitado de hoje, é cada vez mais importante manter-se organizado e produtivo. Com inúmeras responsabilidades e compromissos, muitas vezes é fácil perder o controle das tarefas diárias. Neste contexto, a tecnologia desempenha um papel crucial ao fornecer ferramentas que auxiliam na gestão eficaz do tempo e na organização das atividades cotidianas. Este documento descreve o processo de desenvolvimento de um aplicativo de gerenciamento de rotinas, projetado para ajudar os usuários a planejar, acompanhar e executar suas tarefas de forma eficiente. O aplicativo será desenvolvido utilizando a estrutura React Native, permitindo assim a sua execução em dispositivos móveis.

### 1.1 AMBIENTE DE DESENVOLVIMENTO

O Desenvolvimento do projeto de Aplicativo de Gerenciador de Rotinas será conduzido utilizando a estrutura React Native, reconhecida por sua eficiência na criação de aplicativos móveis multiplataforma, especialmente quando combinada com TypeScript, que traz vantagens como tipagem estática e autocompletar de código, aumentando a produtividade e a confiabilidade do desenvolvimento. Além disso, o Expo será utilizado como uma ferramenta complementar para simplificar e acelerar o desenvolvimento, fornecendo acesso a APIs nativas sem a necessidade de configurações complexas.

Para realizar solicitações à API, será empregado o Axios, uma biblioteca amplamente utilizada para fazer requisições HTTP de forma assíncrona. Os componentes fundamentais do React Native, como FlatList para exibição de listas, TouchableOpacity para botões com feedback tátil, e Text para exibição de texto, serão utilizados para construir a interface do aplicativo de maneira intuitiva e responsiva.

A navegação entre telas será implementada utilizando o React Navigation, uma biblioteca que oferece uma ampla gama de opções de navegação e gestão de estado para aplicativos React Native. Além disso, o StyleSheet será utilizado para estilização dos componentes, garantindo uma apresentação visual consistente e agradável para o usuário.

Em termos de conceitos de programação, o projeto incorporará elementos da orientação a objetos, como herança e encapsulamento, em linha com as práticas modernas de desenvolvimento de software. A implementação de classes abstratas permitirá a definição de estruturas comuns e reutilizáveis, enquanto o uso de funções lambda proporcionará uma sintaxe concisa e expressiva para o tratamento de eventos e processamento de dados.

Além disso, serão aplicados conceitos avançados de programação em Java, como trabalho com streams e threads, para garantir a eficiência e a escalabilidade do aplicativo, especialmente em operações que envolvam processamento de grandes volumes de dados ou interações assíncronas com a API.

## 1.2 MEIOS DE COMUNICAÇÃO

Nos projetos de desenvolvimento de software contemporâneos, a comunicação eficiente é fundamental para o sucesso da colaboração entre membros da equipe. Para facilitar essa colaboração e a troca de informações durante o projeto, foram utilizados dois meios de comunicação essenciais: o GitHub e o WhatsApp.

## 2 OBJETIVO DO PROJETO

O objetivo principal deste projeto é desenvolver um aplicativo de gerenciamento de rotinas que ofereça aos usuários uma maneira eficiente de organizar suas tarefas diárias, semanais e mensais. O aplicativo será projetado para ajudar os usuários a planejar, acompanhar e executar suas tarefas de forma eficaz, facilitando a gestão do tempo e aumentando a produtividade.

### 2.1 ESTRUTURA BÁSICA

Neste capítulo, detalharemos a estrutura fundamental do aplicativo de gerenciamento de rotinas, começando pelos requisitos mínimos e em seguida apresentando o diagrama de casos de uso e incluindo também o diagrama de classes e o diagrama de banco de dados.

#### 2.1.1 Requisitos Mínimos

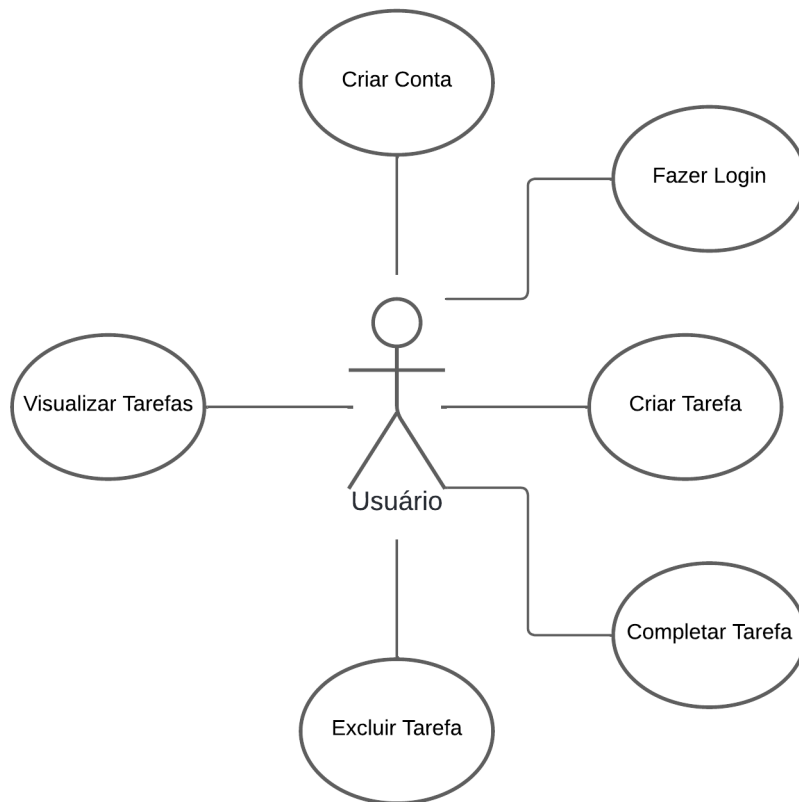
Os requisitos são essenciais para definir o escopo e as funcionalidades do aplicativo. Eles representam as necessidades e expectativas dos usuários, bem como os objetivos do projeto. Os requisitos mínimos que estabelecem as funcionalidades essenciais que o aplicativo deve oferecer aos usuários. São eles:

1. Cadastro de Tarefas: Os usuários devem poder criar, visualizar, editar e excluir tarefas de maneira intuitiva e eficiente.
2. Categorização de Tarefas: As tarefas devem ser organizadas em categorias, permitindo aos usuários classificá-las de acordo com suas preferências e necessidades.
3. Visualização de Calendário: Os usuários devem ter acesso a uma visualização de calendário que mostre suas tarefas agendadas, facilitando o planejamento e acompanhamento de suas rotinas.

### 2.1.2 Diagrama de Casos de Uso

O diagrama de casos de uso é uma ferramenta visual que descreve as interações entre os usuários e o sistema. Ele representa as principais funcionalidades do aplicativo e as relações entre elas. A representação do Diagrama de Casos de Uso para o aplicativo de gerenciador de rotinas pode ser visualizada na Figura 1.

FIGURA 1: Diagrama de Casos de Uso



FONTE: Os Autores (2024)

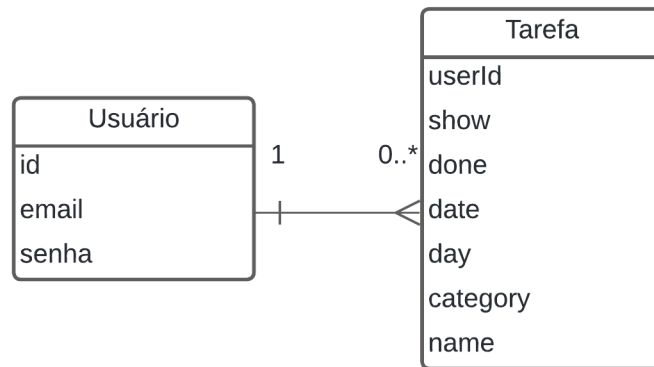
### 2.1.3 Diagrama Físico de Banco de Dados

O diagrama físico de banco de dados é uma representação visual da estrutura do banco de dados utilizado pelo aplicativo, mostrando as tabelas, seus atributos e os relacionamentos entre elas. Esse diagrama fornece uma visão detalhada da organização dos dados dentro do sistema, incluindo as chaves



primárias, chaves estrangeiras e outros elementos que definem a integridade e a consistência dos dados. A representação do Diagrama Físico de Banco de Dados para o aplicativo de gerenciador de rotinas pode ser visualizada na Figura 2.

FIGURA 2: Diagrama de Entidade Relacionamento



FONTE: Os Autores (2024)

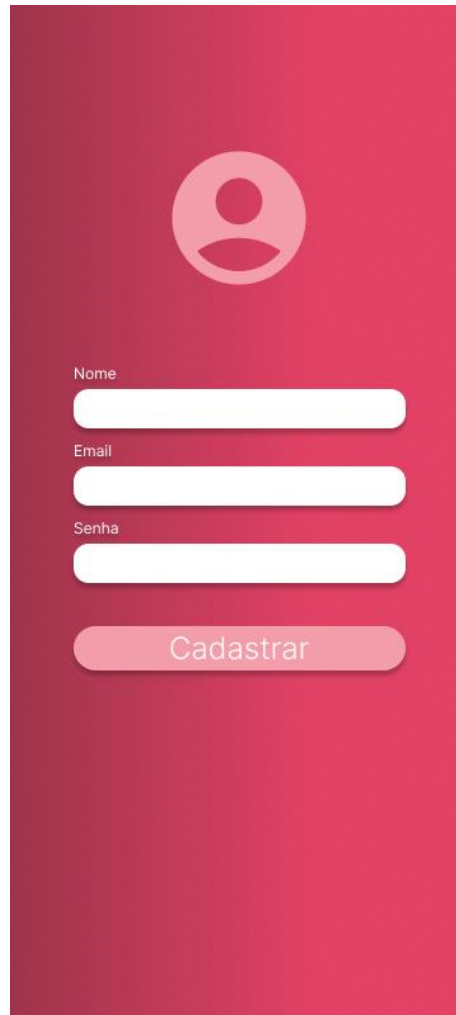
### 3 PROTOTIPAÇÃO

Neste capítulo, será apresentada a prototipação do aplicativo de gerenciamento de rotinas, mostrando os principais fluxos de navegação e as telas fundamentais do sistema. A prototipação é uma etapa crucial no processo de desenvolvimento de software, pois permite validar conceitos e testar a usabilidade do sistema antes da implementação final. Para esta fase, foi utilizada a ferramenta Figma, que permite criar interfaces de usuário de forma rápida e iterativa.

O objetivo da prototipação é fornecer uma representação visual das funcionalidades do aplicativo, incluindo a disposição dos elementos na tela, as interações do usuário e a navegabilidade entre as diferentes telas. Isso permite que a equipe de desenvolvimento e as partes interessadas visualizem e entendam como o aplicativo funcionará na prática, identificando possíveis melhorias e ajustes necessários.

Ao longo deste capítulo, serão apresentados os protótipos das telas principais do aplicativo, destacando os elementos de interface e as funcionalidades associadas a cada uma. Esses protótipos servirão como base para o desenvolvimento posterior do aplicativo, garantindo que os requisitos mínimos estabelecidos sejam atendidas de forma eficaz e intuitiva.

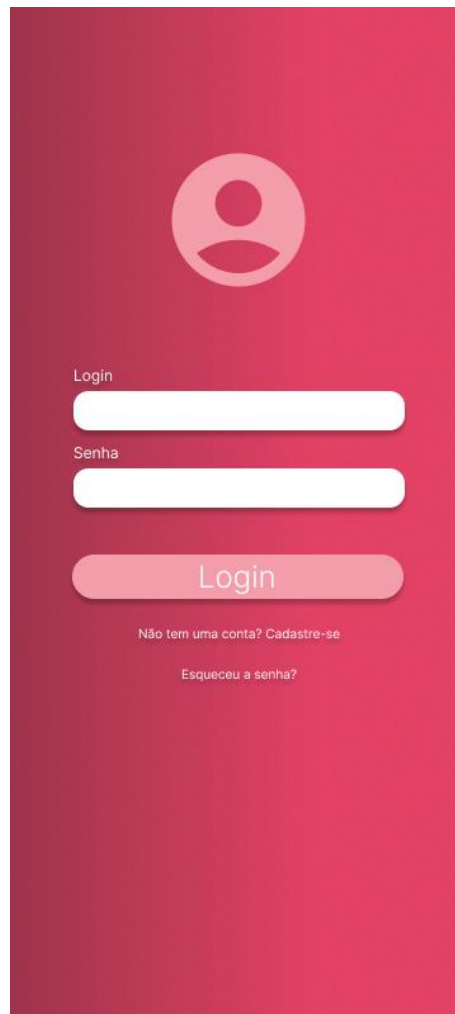
FIGURA 3: Protótipo - Tela de AutoCadastro



Protótipo de tela de AutoCadastro. A interface possui um fundo rosa-claro com um gradiente. No topo, há um ícone de perfil de usuário em tons de rosa. Abaixo, há três campos de entrada de texto brancos com bordas arredondadas, cada um precedido por um rótulo em cinza: 'Nome', 'Email' e 'Senha'. No final, há um botão retangular com cantos arredondados, cor de fundo rosa e o texto 'Cadastrar' em branco.

FONTE: Os Autores (2024)

FIGURA 4: Protótipo - Tela de Login



The image shows a mobile app login screen prototype. It features a solid red background with a subtle gradient. At the top center is a white circular icon representing a user profile. Below this, the word "Login" is displayed in a small, light gray font. Underneath is a white, rounded rectangular input field. Below the first field, the word "Senha" is displayed in a small, light gray font, followed by another identical white, rounded rectangular input field. Centered below these fields is a large, rounded rectangular button with a light red gradient and the word "Login" in white. At the bottom of the screen, there are two lines of small, light gray text: "Não tem uma conta? Cadastre-se" and "Esqueceu a senha?".

FONTE: Os Autores (2024)

FIGURA 5: Protótipo - Tela Inicial



FONTE: Os Autores (2024)

FIGURA 6: Protótipo - Tela de Adicionar Tarefa

18:55 96%

Tarefas Atuais Nova Tarefa

Nome

Categoria Selecionar

Data e Horário Selecionar

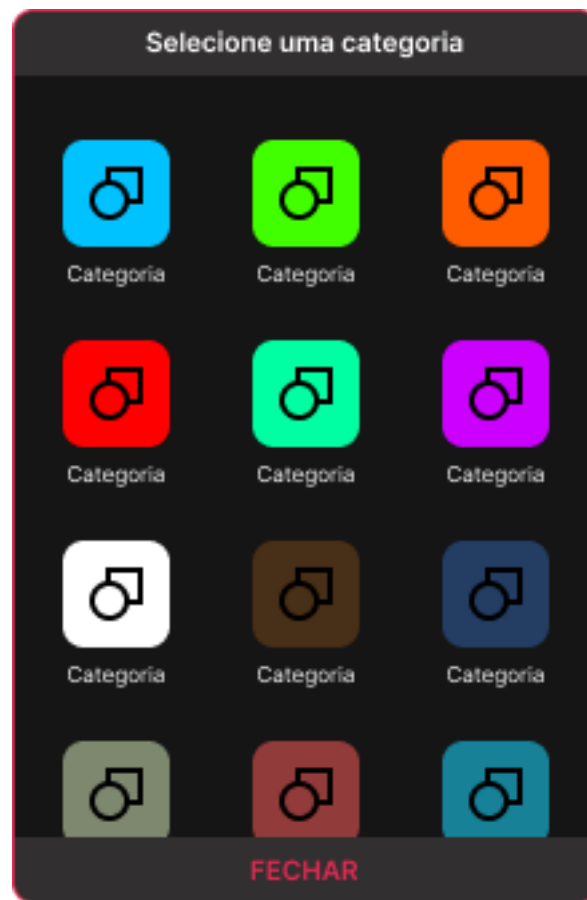
Notificações Adicionar

Comentários Adicionar

Home Tarefas

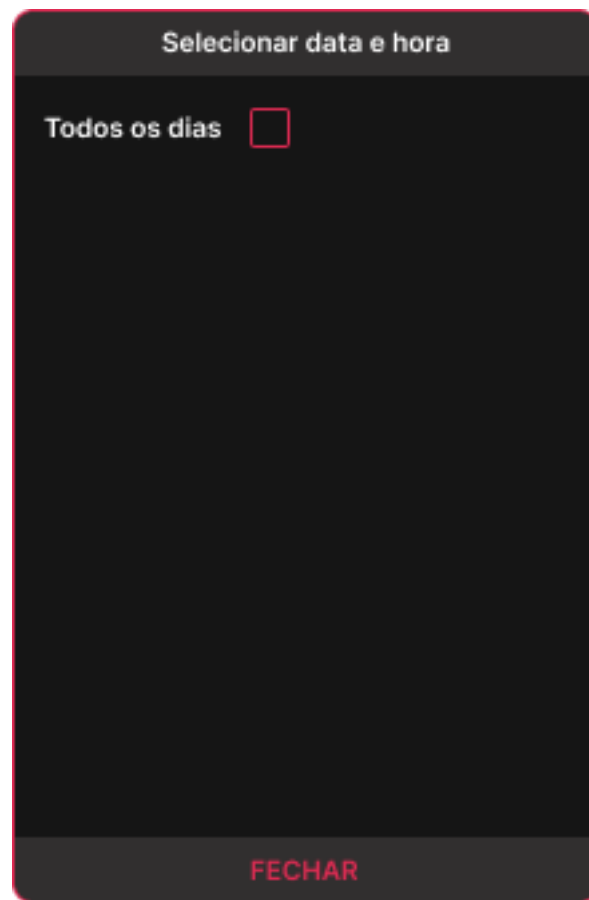
FONTE: Os Autores (2024)

FIGURA 7: Protótipo - Tela de Selecionar Categoria



FONTE: Os Autores (2024)

FIGURA 8: Protótipo - Tela de Selecionar Data e Hora



Selecione data e hora

Todos os dias ☐

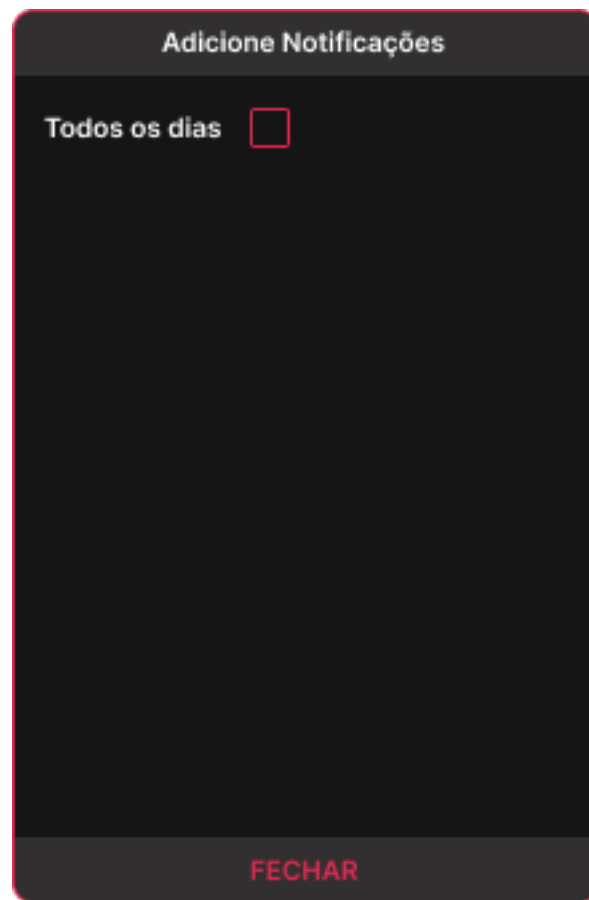
FECHAR

Detailed description: This is a mobile app prototype for a date and time selection screen. It features a dark theme with a black background. At the top, there is a dark gray header bar with the text 'Selecione data e hora' in white. Below the header, the text 'Todos os dias' is displayed in white, followed by a small, empty square checkbox. The bottom of the screen has a dark gray footer bar with the word 'FECHAR' in red, indicating a close or cancel action.

FONTE: Os Autores (2024)



FIGURA 9: Protótipo - Tela de Selecionar Notificação



Adicione Notificações

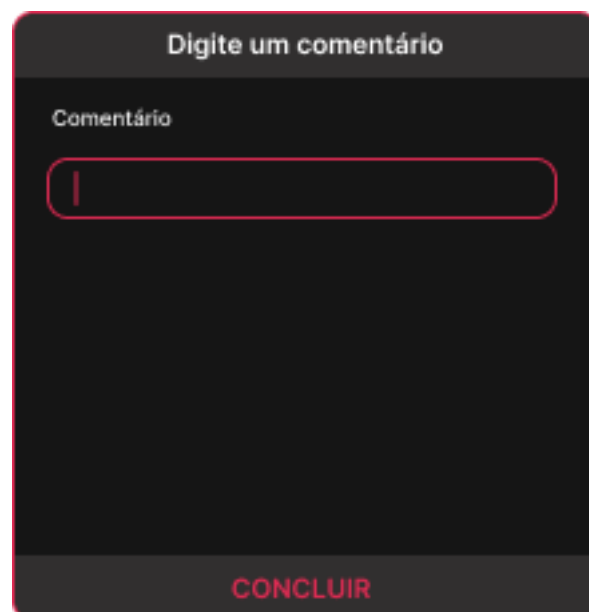
Todos os dias ☐

FECHAR

Detailed description: This is a mobile app prototype for selecting notifications. It features a dark theme. At the top, there is a header bar with the text 'Adicione Notificações' in white. Below the header, the main area is dark gray and contains the text 'Todos os dias' followed by an unchecked checkbox. At the bottom, there is a dark gray footer bar with the red text 'FECHAR'.

FONTE: Os Autores (2024)

FIGURA 10: Protótipo - Tela de Adicionar Comentários



Digite um comentário

Comentário

CONCLUIR

Detailed description: This is a mobile app prototype for adding comments. It features a dark theme. At the top, there is a header bar with the text 'Digite um comentário' in white. Below the header, the main area is dark gray and contains the label 'Comentário' above a text input field. At the bottom, there is a dark gray footer bar with the red text 'CONCLUIR'.

FONTE: Os Autores (2024)

## 4 FUNCIONALIDADES DO SISTEMA

Neste capítulo, serão apresentadas as principais funcionalidades do sistema de gerenciamento de rotinas, detalhando como cada conceito foi aplicado para implementar essas funcionalidades.

### 4.1 CONCEITOS APLICADOS

Cada conceito será detalhado em termos de sua aplicação prática e contribuição para o desenvolvimento do aplicativo.

#### 4.1.1 Componentes React Native

FIGURA 11 : Demonstração do uso de componentes React Native

```

1  import React, {useState,useContext} from "react";
2  import {View,Text,Pressable} from 'react-native';
3  import { TextInput } from "react-native-gesture-handler";
4  import styles from "../Styles";
5  import { useAuth } from "../context/AuthContext";
6
7
8  const LoginScreen = ({navigation}:any) =>{
9    const {signIn} = useAuth();
10   const [email, setEmail] = useState('');
11   const [password, setPassword] = useState('');
12
13   return(
14     <View style={styles.container}>
15       <Text>Login :</Text>
16       <TextInput value={email} onChangeText={text => setEmail(text)} inputMode='email' style={styles.input} />
17       <Text>Senha :</Text>
18       <TextInput value={password} onChangeText={text => setPassword(text)} secureTextEntry={true} style={styles.input} />
19       <Pressable style={styles.botao} onPress={()=>signIn(email,password)}>
20         <Text>Login</Text>
21       </Pressable>
22       <Pressable onPress={()=>navigation.navigate('Cadastro')}>
23         <Text>Não tem conta ? Cadastre-se</Text>
24       </Pressable>
25       <Pressable onPress={()=>navigation.navigate('Recuperar')}>
26         <Text>Esqueceu a senha ?</Text>
27       </Pressable>
28     </View>
29   )
30 }

```

FONTE: Os Autores (2024)

Na Figura 10, o componente *LoginScreen* utiliza os componentes *View*, *Text*, *TextInput* e *Pressable*. *View* é usado como contêiner, *Text* para exibir textos, *TextInput* para os campos de entrada de email e senha, e *Pressable* para os botões de ação e links de navegação.

#### 4.1.2 React Navigation

FIGURA 12: Demonstração do uso de React Navigation

```
1  import 'react-native-gesture-handler';
2  import React from 'react';
3  import { createStackNavigator } from '@react-navigation/stack';
4  import LoginScreen from '../screens/LoginScreen';
5  import RegisterScreen from '../screens/RegisterScreen';
6  import RestoreScreen from '../screens/RestoreScreen';
7
8  const Stack = createStackNavigator();
9
10
11  export function AuthStack() {
12    return(
13      <Stack.Navigator>
14        <Stack.Screen name="Login" component={LoginScreen} />
15        <Stack.Screen name="Cadastro" component={RegisterScreen} />
16        <Stack.Screen name="Recuperar" component={RestoreScreen} />
17      </Stack.Navigator>
18    )
19  }
```

FONTE: Os Autores (2024)

Na Figura 11, o React Navigation é implementado criando um navegador de pilha (*Stack Navigator*) usando *createStackNavigator* de *@react-navigation/stack*. O navegador (*AuthStack*) configura três telas: *LoginScreen*, *RegisterScreen* e *RestoreScreen*, permitindo a navegação entre elas.

### 4.1.3 StyleSheet

FIGURA 13: Demonstração do uso de StyleSheet

```
1  import {StyleSheet} from 'react-native'
2
3  const styles = StyleSheet.create({
4    container:{
5      flex:1,
6      backgroundColor: '#DF2B53',
7      justifyContent: 'center',
8      paddingLeft:60
9    },
10   input:{
11     backgroundColor: '#FFFFFF',
12     width:260,
13     height:30,
14     borderRadius:12,
15     margin:10,
16     paddingLeft:3
17   },
18   botao:{
19     backgroundColor: '#F49EAB',
20     width:260,
21     height:35,
22     borderRadius:24,
23     justifyContent: 'center',
24     paddingLeft:110,
25   }
26 },
27 );
28
29 export default styles;
```

FONTE: Os Autores (2024)

Na Figura 12, os estilos são definidos usando *StyleSheet.create* do *react-native*. O objeto *styles* define estilos para o contêiner principal (*container*), os campos de entrada (*input*) e os botões (*botao*), que são aplicados aos componentes correspondentes no código.

#### 4.1.4 Função Lambda

FIGURA 14: Demonstração do uso de Função Lambda

```
const RegisterScreen = () => {  
  const [nome, setNome] = useState('');  
  const [email, setEmail] = useState('');
```

FONTE: Os Autores (2024)

Na figura 14, pode-se ver a utilização de uma função lambda, sendo ela representada por “() => {}”

#### 4.1.5 Streams

FIGURA 15: Demonstração do Uso de Streams

```
useEffect(() => {  
  if (taskContext) {  
    const newTasksByDate: AgendaSchedule = taskContext.tasks.reduce((acc, task) => {  
      const date = task.date;  
      if (!acc[date]) acc[date] = [];  
      acc[date].push({ ...task, height: 50, day: date });  
      return acc;  
    }, {} as AgendaSchedule);  
    setTasksByDate(newTasksByDate);  
  }  
});
```

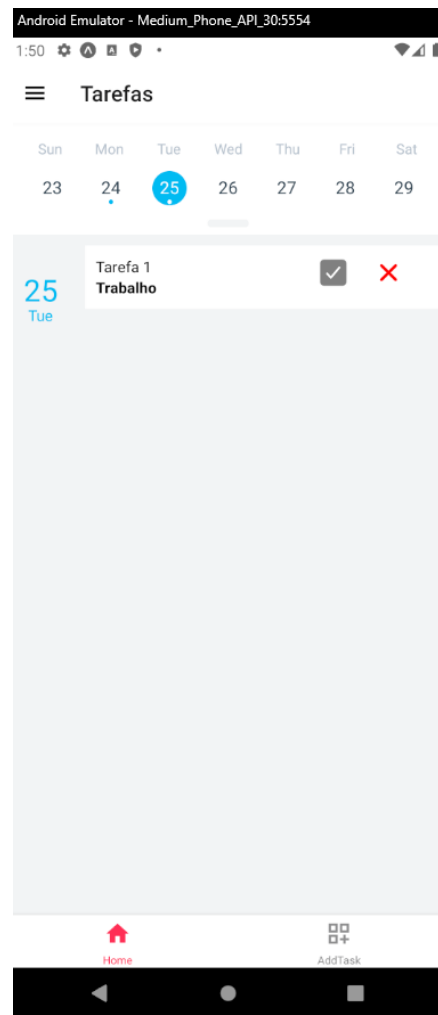
FONTE: Os Autores (2024)

Na figura 15, pode-se ver a utilização de um método Stream, o *reduce*. Neste caso, está reduzindo as tarefas em um objeto *AgendaSchedule*, onde as tarefas são agrupadas por data.

## 5 APRESENTAÇÃO DOS RESULTADOS

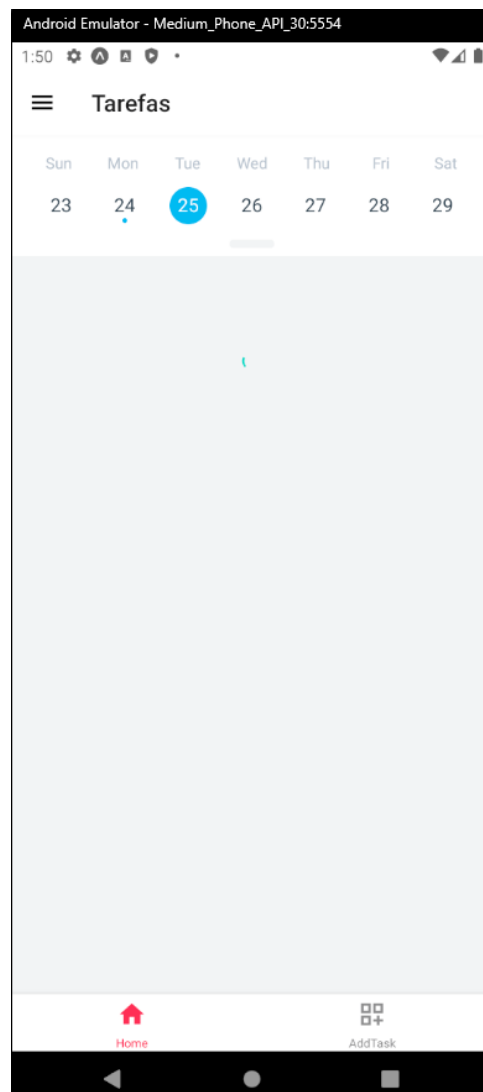
Neste capítulo, apresentamos os resultados obtidos após o desenvolvimento e implementação do sistema de gerenciamento de rotinas. A funcionalidade do sistema foi avaliada em relação à capacidade de criar e gerenciar tarefas, assim como a eficácia das funcionalidades de categorização. Destacamos também a usabilidade do aplicativo, incluindo a intuitividade da interface do usuário, facilidade de navegação e clareza das instruções fornecidas.

FIGURA 16: Demonstração de uso do Aplicativo 1



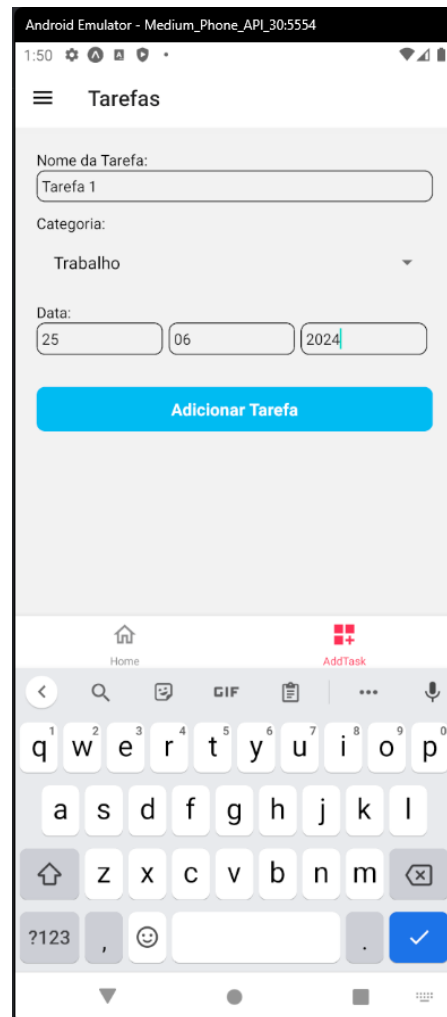
FONTE: Os Autores (2024)

FIGURA 17: Demonstração de uso do Aplicativo 2



FONTE: Os Autores (2024)

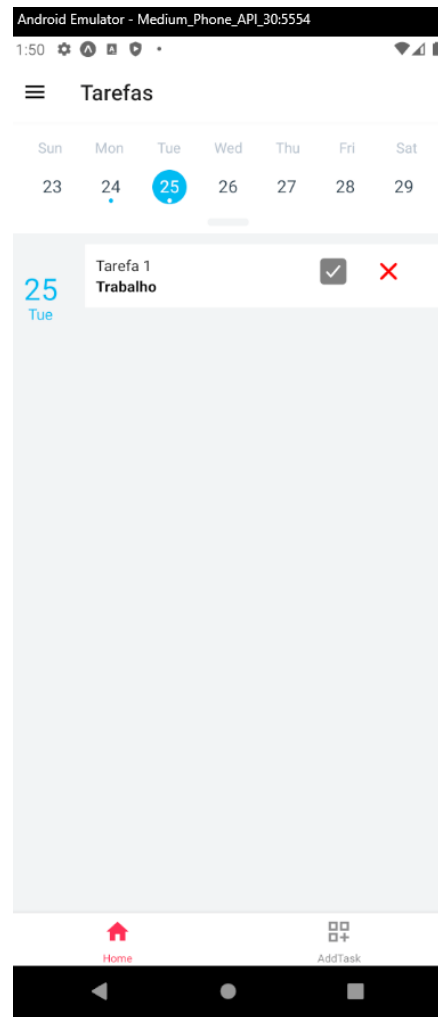
FIGURA 18: Demonstração de uso do Aplicativo 3



FONTE: Os Autores (2024)



FIGURA 19: Demonstração de uso do Aplicativo 4



FONTE: Os Autores (2024)

## **6 CONSIDERAÇÕES FINAIS**

O desenvolvimento do Sistema de Gerenciamento de Rotinas demonstrou a importância de uma abordagem estruturada e colaborativa para a criação de aplicativos móveis eficientes. Desde a fase inicial de levantamento de requisitos até a entrega final, cada etapa do projeto foi planejada e executada, resultando em um produto que atende às necessidades dos usuários de forma eficaz. A aplicação de tecnologias modernas como React Native, Expo, garantiu um desempenho robusto e uma experiência de usuário intuitiva. O sucesso deste projeto reforça a relevância de práticas de desenvolvimento ágil e destaca a importância de uma comunicação clara e contínua entre todos os envolvidos.

## REFERÊNCIAS

REACT NATIVE. Disponível em: <https://reactnative.dev/>.

AXIOS. Disponível em: <https://www.npmjs.com/package/axios>.

GITHUB. Disponível em: <https://github.com/>.

FIGMA. Disponível em: <https://www.figma.com/>.

## **ANEXO 1 – RELATÓRIO DE EXECUÇÃO DO PROJETO**

### **Semana 1: Levantamento de Requisitos**

Durante esta semana, foram realizadas reuniões entre os membros da equipe para definir os requisitos do sistema. Foram identificadas as principais funcionalidades esperadas e documentadas no documento de escopo do projeto.

### **Semana 2: Elaboração de Casos de Uso**

Na segunda semana, os requisitos levantados foram traduzidos em casos de uso. Cada funcionalidade do sistema foi detalhada em casos de uso.

### **Semana 3 e 4: Design e Prototipação das Telas**

Durante estas duas semanas, a equipe concentrou-se no design e prototipação das telas do aplicativo. Utilizando a ferramenta Figma, foram criados wireframes e protótipos interativos das telas principais, como a tela de login, tela de lista de tarefas, tela de criação de tarefa, entre outras.

### **Semana 5, 6, 7 e 8: Desenvolvimento**

Durante estas quatro semanas, foi realizado o desenvolvimento do aplicativo. Utilizando React Native com TypeScript, as telas foram implementadas de acordo com os protótipos criados anteriormente.

### **Semana 9: Testes**

Na nona semana, foram realizados testes para verificar o funcionamento correto do sistema. Foram identificados e corrigidos bugs e problemas de usabilidade encontrados durante os testes.

### Semana 10: Revisão e Desenvolvimento

Na semana 10, foi realizada uma revisão final do projeto. Foram verificados todos os requisitos e casos de uso para garantir que todas as funcionalidades foram implementadas corretamente. Foi necessário corrigir algumas implementações.

### Semana 11: Ajustes e Entrega

Na última semana, foi realizado ajustes nos estilos e foi produzido o vídeo para apresentação e entrega do projeto.