UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DEPARTAMENTO ACADÊMICO DE INFORMÁTICA
CURSO DE ESPECIALIZAÇÃO EM TECNOLOGIA JAVA

WESLEI PATRICK CAMBRUZZI

APLICATIVO MOBILE PARA GERENCIAMENTO DE REEMBOLSO DE QUILOMETRAGEM

MONOGRAFIA DE ESPECIALIZAÇÃO

PATO BRANCO
2020

WESLEI PATRICK CAMBRUZZI

APLICATIVO MOBILE PARA GERENCIAMENTO DE REEMBOLSO DE QUILOMETRAGEM

Monografia de especialização apresentado ao Curso de Especialização em Tecnologia Java, do Departamento Acadêmico de Informática, da Universidade Tecnológica Federal do Paraná, Câmpus Pato Branco, como requisito parcial para obtenção do título de Especialista.

Orientadora: Profa. Andreia Scariot Beulke

PATO BRANCO
2020

Ministério da Educação
Universidade Tecnológica Federal do Paraná
Câmpus Pato Branco
Departamento Acadêmico de Informática
Curso de Especialização em Tecnologia Java

TERMO DE APROVAÇÃO

APLICATIVO MOBILE PARA GERENCIAMENTO DE REEMBOLSO DE QUILOMETRAGEM

por

WESLEI PATRICK CAMBRUZZI

A avaliação deste trabalho de conclusão de curso foi realizada em 09 de março de 2020, como requisito parcial para a obtenção do título de especialista em Tecnologia Java. Após a apresentação o candidato foi arguido pela banca examinadora composta pelos professores Andréia Scariot Beulke (orientadora), Beatriz Terezinha Borsoi e Vinicius Pegorini, membros de banca. Em seguida foi realizada a deliberação pela banca examinadora que considerou o trabalho aprovado.

_____
Andréia Scariot Beulke
Profa. Orientadora (UTFPR)

_____
Beatriz Terezinha Borsoi
(UTFPR)

_____
Vinicius Pegorini
(UTFPR)

_____
Vinicius Pegorini
Coordenador do curso

A Folha de Aprovação assinada encontra-se na Coordenação do Curso.

RESUMO

Cada vez mais as empresas estão revendo seus procedimentos, num intuito de permanecerem no mercado competitivo na qual atuam. A organização e a gestão de custos e despesas têm grande destaque neste sentido, inclusive no âmbito da modernização e informatização dos processos. Visando redução dos custos para a empresa, a necessidade de informatizar o processo de controle de quilometragem, pago pelo deslocamento dos seus funcionários, ao utilizarem veículo próprio para o deslocamento ao trabalho. Neste trabalho foi desenvolvido um aplicativo que possibilite registrar os veículos e seus proprietários para posterior reembolso das despesas que o empregado teve na utilização de veículo próprio para a execução de serviços relacionados à empresa. Para a identificação do veículo será utilizada a leitura de um *QR Code* e um relatório das informações registradas.

Palavras-chave: Android. Mobile. Java. React Native.

ABSTRACT

More and more companies are reviewing their procedures, in order to remain in the competitive market in which they operate. The organization and management of costs and expenses has a great emphasis in this regard, including in the scope of modernization and computerization of processes. Aiming to reduce costs for the company, the need to computerize the mileage control process, paid for the displacement of employees of a company, when using their own vehicle for commuting to and from work. Thus, the development of an application that makes it possible to register the vehicles and, consequently, the owners who went to the company with their own vehicle, for the identification of the vehicle will be used the reading of a QR Code and finally a report of the registered information.

Keywords: Android. Mobile. Java. React Native.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTAGENS DE CÓDIGO

LISTA DE SIGLAS

API	*Application Programming Interface*
HTTP	*Hypertext Transfer Protocol*
IDE	*Integrated Development Environment*
JSON	*JavaScript Object Notation*
JVM	*Java Virtual Machine*
QR Code	*Quick Response Code*
URL	*Uniform Resource Locator*

SUMÁRIO

1 INTRODUÇÃO

Este capítulo apresenta a introdução do trabalho que abrange as considerações iniciais, os seus objetivos e a justificativa. O capítulo é finalizado com a apresentação do texto por meio da descrição sumária dos próximos capítulos.

1.1 CONSIDERAÇÕES INICIAIS

Cada vez mais as empresas estão revendo seus procedimentos, num intuito de permanecerem no mercado que está cada vez mais competitivo. A organização e a gestão de custos e despesas tem grande destaque neste sentido, inclusive no âmbito da modernização e informatização dos processos.

De acordo com Chiavenato (2014), em uma era de mudanças, de incessante desenvolvimento tecnológico, o administrador moderno precisa construir habilidades conceituais a fim de proporcionar novos rumos e caminhos para sua organização.

A tecnologia da informação pode colaborar para um melhor controle dos gastos das empresas, visando maior assertividade e agilidade no processamento de dados, e com a criação de aplicativos mais complexos reduzirem os gastos desnecessários ocorrentes de lançamentos manuais e com pouca confiabilidade.

O aplicativo desenvolvido como resultado deste trabalho propõe a informatização do controle de quilometragem, pago pelo deslocamento dos funcionários de uma empresa ao utilizarem veículo próprio para traslado de trabalho.

1.2 OBJETIVOS

O objetivo geral apresenta o resultado principal obtido da realização deste trabalho e os objetivos específicos o complementam.

1.2.1 Objetivo Geral

Desenvolver um aplicativo para dispositivos móveis para identificar veículos de funcionários utilizados para transporte ao trabalho.

1.2.2 Objetivos Específicos

- Utilizar o *Quick Response Code* (QR Code) como forma de identificação dos veículos dos colaboradores utilizados para o deslocamento ao local de trabalho.
- Permitir que a empresa controle os dados dos colaboradores que estão usando o seu veículo para posterior reembolso das despesas relacionadas à quilometragem realizada para a execução dos serviços relacionados aos assuntos da empresa;
- Disponibilizar um relatório com informações para efetuar o reembolso de gastos com combustível.

1.3 JUSTIFICATIVA

Um reembolso acontece quando a empresa devolve o dinheiro gasto por um funcionário com, por exemplo, o uso de veículo próprio para deslocamento até o trabalho. Nesse caso, o valor de reembolso é decorrente do combustível em veículo próprio, que o funcionário gastou.

O reembolso pode ser tanto de despesa com combustível quanto de reembolso de quilometragem. Em ambos os formatos os valores não integram o salário contratual do empregado. A escolha entre um ou outro depende do planejamento da empresa, de um acordo entre ela e o funcionário ou de cada situação específica.

A realização deste trabalho consiste em um aplicativo para dispositivos móveis que visa fazer a leitura via QR Code dos dados do veículo e do colaborador para que, posteriormente, a empresa possa reembolsá-lo quanto aos gastos com combustível utilizado para locomoção. O aplicativo permite que um colaborador registre somente veículos que se encontram no estacionamento, para posteriormente gerar informação para o pagamento.

Para o desenvolvimento do trabalho foi utilizada a tecnologia Java e React Native que se apresenta como uma boa alternativa no desenvolvimento de aplicativos para múltiplas plataformas. Além disso, oferecer uma melhor integração entre as funções nativas do aparelho, em razão de ele não converter o código para linguagens nativas, mas realizar a comunicação com o código nativo.

1.4 ESTRUTURA DO TRABALHO

Este trabalho está organizado em capítulos, dos quais este é o primeiro e apresenta as considerações iniciais e contextualização do trabalho, incluindo os objetivos e a justificativa. No Capítulo 2 estão os materiais utilizados no desenvolvimento do trabalho. Os materiais se referem às tecnologias e ferramentas utilizadas e o método contém a metodologia adotada para o desenvolvimento deste trabalho, visando alcançar os objetivos propostos. O Capítulo 3 a modelagem do sistema contendo os requisitos funcionais e não funcionais, diagrama de casos de uso e de entidade e relacionamento do banco de dados, as telas e a implementação do sistema. No Capítulo 4 está a conclusão do trabalho e, por fim, estão as referências utilizadas no texto.

2 FERRAMENTAS, TECNOLOGIAS E PROCEDIMENTOS

Neste capítulo são apresentadas as ferramentas e as tecnologias utilizadas no desenvolvimento do aplicativo.

2.1 FERRAMENTAS E TECNOLOGIAS

As ferramentas e as tecnologias utilizadas para a modelagem e o desenvolvimento do projeto estão listadas no Quadro 1.

Quadro 1 – Tecnologias e ferramentas utilizadas na modelagem e na implementação do aplicativo

Ferramenta / Tecnologia	**Versão**	**Disponível em**	**Aplicação**
IntelliJ IDEA	2018.1.3	https://www.jetbrains.com/ide/	*Integrated Development Environment* (IDE) para desenvolvimento da API.
Java	8.0	http://www.oracle.com/technetwork/java/javaee/downloads/index.html	Linguagem para desenvolvimento da API
Javascript	ECMAScript 2018		Linguagem de programação utilizada para o desenvolvimento do aplicativo.
MySQL	5.7	http://www.mysql.com/	Banco de dados.
MySQL Workbench	6.3.8	http://dev.mysql.com/downloads/workbench/	Software para gerenciamento de bases de dados MySQL.

React Native	0.60	https://reactnative.dev/	Para desenvolver aplicativos para os sistemas Android e IOS de forma nativa.
Spring Framework	5.2.4	https://spring.io/	Framework *open source* para a plataforma Java.
Visual Studio Code	1.16.1	https://code.visualstudio.com/	Editor de código-fonte utilizado para desenvolvimento do aplicativo.

Fonte: Autoria própria (2020).

2.1.1 Java

Java é tanto uma linguagem quanto uma plataforma. Isso quer dizer que, além da linguagem, o programador Java também conta com um conjunto de APIs que facilitam o desenvolvimento ao oferecerem soluções para situações comuns de desenvolvimento (DEVMEDIA, 2020).

Uma das grandes vantagens do Java é ser multiplataforma, o que possibilita a execução de aplicativos em qualquer sistema operacional ou hardware, desde que o interpretador Java esteja instalado. Os programas desenvolvidos são emulados por meio de uma máquina virtual, conhecida como *Java Virtual Machine* (JVM), que converte os *bytecodes* (codificação do programa) para uma linguagem que a máquina entenda .

2.1.1 Spring Framework

Spring Framework é um *framework* voltado para o desenvolvimento de aplicações corporativas para a plataforma Java, baseado nos conceitos de inversão de controle e injeção de dependências (WEISSMANN, 2014).

2.1.1 Javascript

JavaScript é uma linguagem de programação conhecida como a linguagem de *script* para páginas *web*, usada também em vários outros ambientes sem navegador, tais como node.js, Apache CouchDB e Adobe Acrobat. JavaScript é uma linguagem baseada em protótipos, multi-paradigma e dinâmica, suportando estilos de orientação a objetos, imperativos e declarativos (como, por exemplo, a programação funcional) (MDN, 2020).

Essa linguagem permite implementar funcionalidades mais complexas como mostrar em tempo real conteúdos atualizados, mapas interativos, animações gráficas em 2D/3D, vídeos (MDN, 2020).

De acordo com Fernandes (2020), a linguagem JavaScript vem sendo muito utilizada em desenvolvimento *mobile*, tanto por meio de ferramentas híbridas como Ionic quanto tecnologias mais modernas como React Native.

2.1.2 React Native

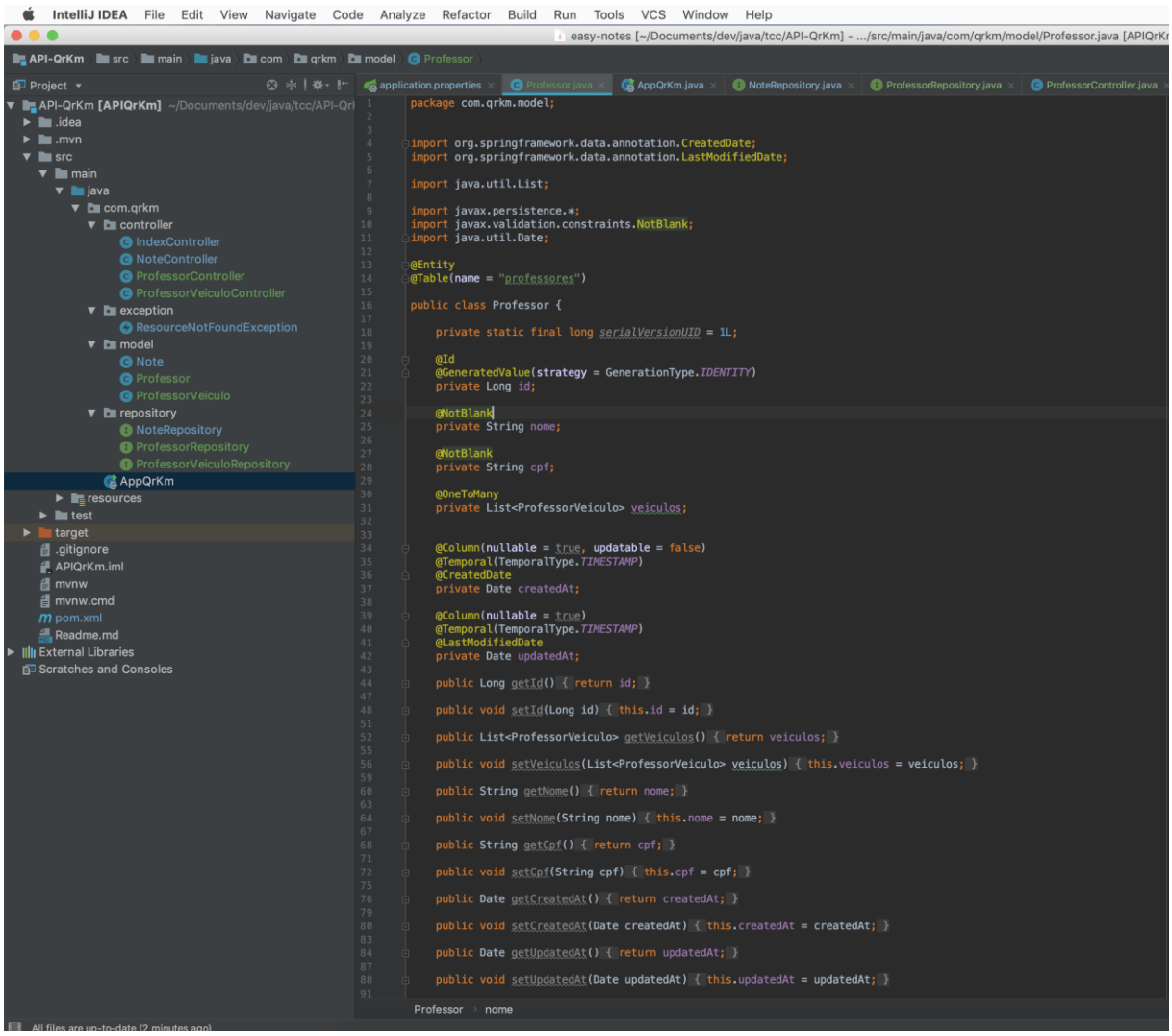
Para o desenvolvimento do aplicativo, foi escolhida uma das tecnologias atualmente mais utilizadas no mercado para aplicativos híbridos. Utilizando o React Native é possível escrever códigos em JavaScript e o mesmo renderiza para código nativo.

De acordo com Devmedia (2020), o React Native consiste em uma série de ferramentas que viabilizam a criação de aplicações móveis nativas para a plataforma iOS e Android, utilizando o que há de mais moderno no desenvolvimento *front-end*. O *stack* do React Native permite utilizar ECMAScript 6, CSS Flexbox, JSX, diversos pacotes do NPM, entre outros. Também permite fazer *debug* na mesma IDE utilizada para o desenvolvimento nativo com essas plataformas.

2.1.3 IntelliJ IDEA

IntelliJ IDEA é uma *Integrated Development Environment* (IDE) que oferece excelente assistência no ambiente de codificação e características de aumento de produtividade para Java EE, Spring, GWT, Grails, Play e outras estruturas, juntamente com ferramentas de implantação para a maioria dos servidores de aplicativos. Ele inclui um conjunto de ferramentas que trabalham: suporte para Maven, Gradle; integração com o Git, SVN (L3SOFTWARE, 2020). A Figura 1 exibe a tela inicial do IntelliJ IDEA.

Figura 1 - IntelliJ IDEA

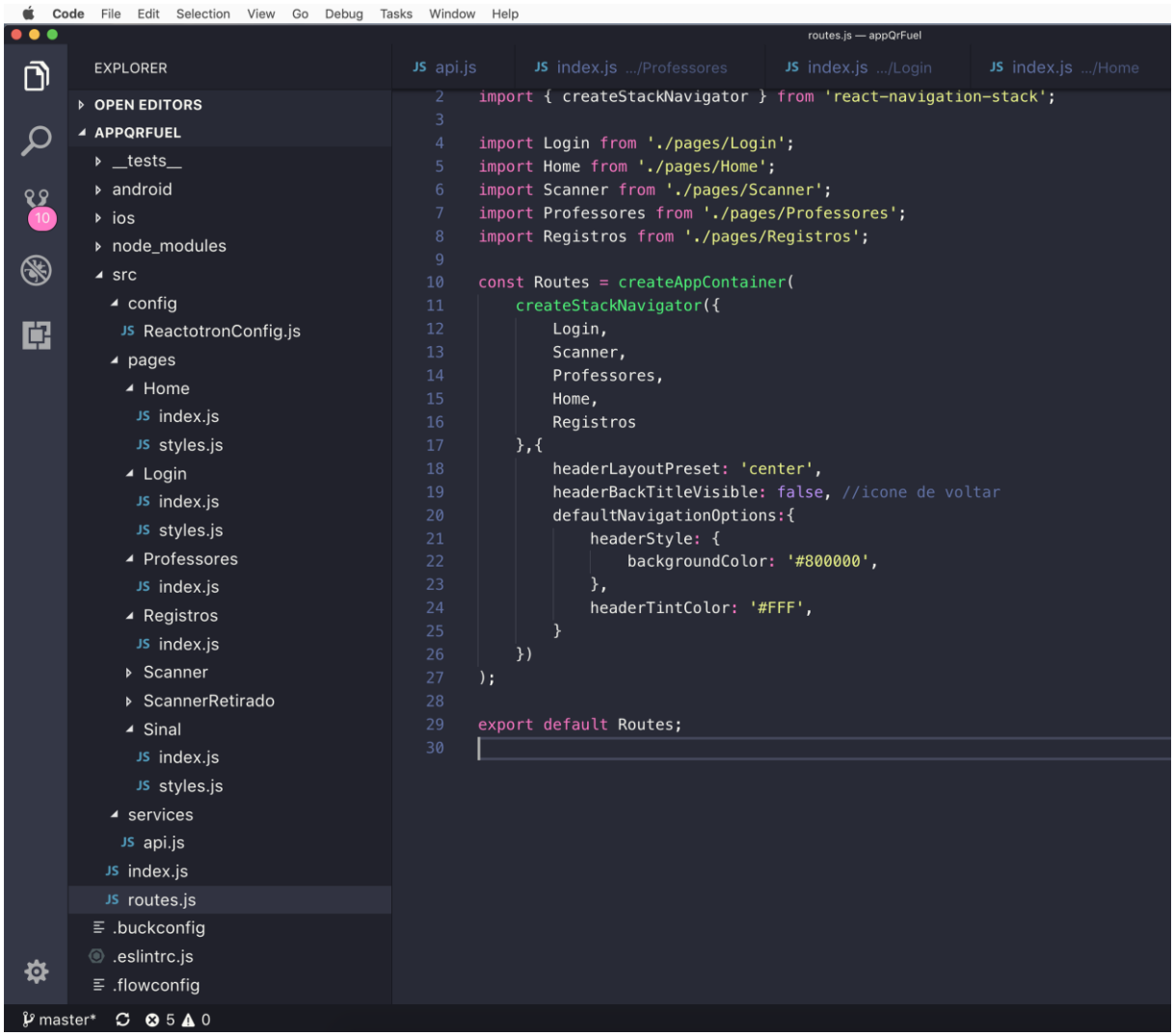


Fonte: Autoria própria (2020).

2.1.4 Visual Studio Code

Para o desenvolvimento do aplicativo foi utilizado o editor de código fonte Visual Studio Code que é leve, porém muito eficiente. Está disponível para Windows, macOS e Linux. Ele vem com suporte interno para JavaScript, TypeScript e Node.js e possui um rico ecossistema de extensões para outras linguagens (como C ++, C #, Java, Python, PHP, Go) e tempos de execução (como .NET e Unity) (VISUALSTUDIO, 2020). A Figura 2 exibe a tela inicial do Visual Studio Code.

Figura 2 - Visual Studio Code



Fonte: Autoria própria (2020).

3 RESULTADO

Este capítulo apresenta o resultado do desenvolvimento do trabalho, que é um aplicativo para informatização do controle de quilometragem, pago pelo deslocamento dos funcionários de uma empresa.

3.1 ESCOPO DO SISTEMA

O aplicativo tem por objetivo informatizar o controle de reembolso de despesas com combustível quando os colaboradores de uma empresa utilizam veículo próprio para deslocamento de trabalho. O aplicativo permite que um colaborador registre somente veículos que se encontram no estacionamento, para posteriormente gerar informação para o pagamento. Isso porque somente os proprietários ou responsáveis pelo veículo poderão ser ressarcido quando às despesas com combustível. Um funcionário que utilizou a carona como meio de transporte não deverá ser reembolsado, uma vez que não teve despesas com transporte. A seguir são descritas as principais funcionalidades do escopo do aplicativo.

- Leitura do *QR Code* que estará disponível nos veículos dos funcionários, para confirmar que o veículo realmente estava na empresa.
- Listagem dos funcionários e respectivas informações sobre os veículos, como cor, modelo e placa.
- Relatório das informações obtidas por meio da leitura do QR Code dos veículos, como quantidade de registros por mês e funcionário.

3.2 MODELAGEM DO SISTEMA

Para o desenvolvimento do aplicativo foram estabelecidos alguns requisitos, listados nos Quadros 2.

Quadro 2 - Requisitos funcionais

Identificação	Nome	Descrição
RF001	Leitura QR Code	Realizar o registro da leitura dos QR Code.
RF002	Consultar funcionário	Consultar informações dos respectivos funcionários.

RF003	Consultar registros	Consultar informações dos registros obtidos por meio da leitura do QR Code.

Fonte: Autoria própria (2020).

No Quadro 3 estão os requisitos não funcionais definidos.

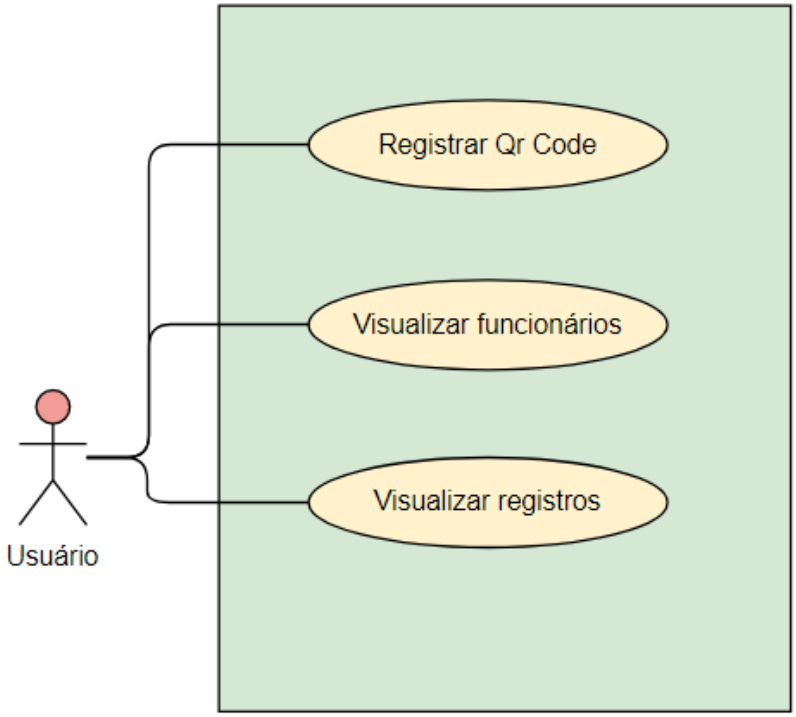
Quadro 3 - Requisitos não funcionais

Identificação	Nome	Descrição
RNF001	Acesso à internet	O aplicativo necessitará de conexão com a internet.
RNF002	Texto QR Code	O texto do QR Code deverá receber o código do item no banco de dados.
RNF003	Autenticação no aplicativo	O *email* utilizado como *login*, deverá ser armazenado na memória do celular (*Local Storage*), para que nos próximos acessos o aplicativo não exiba mais a tela de autenticação no sistema.

Fonte: Autoria própria (2020).

Fundamentado nos requisitos funcionais e não funcionais elencados para o desenvolvimento do aplicativo, foi elaborado o diagrama de caso de uso conforme a Figura 5.

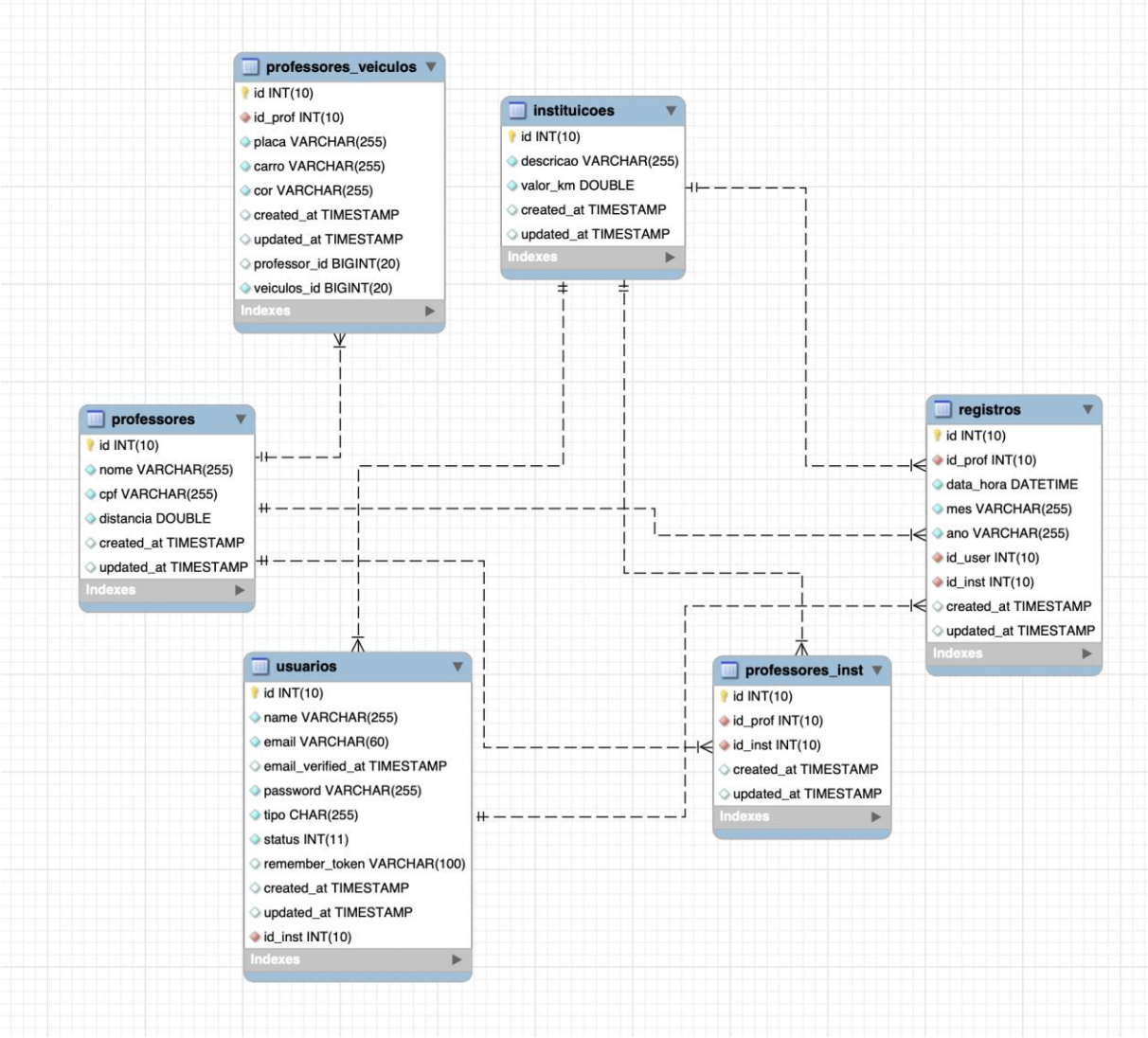
Figura 3 - Diagrama de caso de uso



Fonte: Autoria própria (2020).

A Figura 4 exibe o diagrama de entidades e relacionamentos do banco de dados com seus respetivos campos e tipos.

Figura 4 - Diagrama de Entidades e Relacionamentos



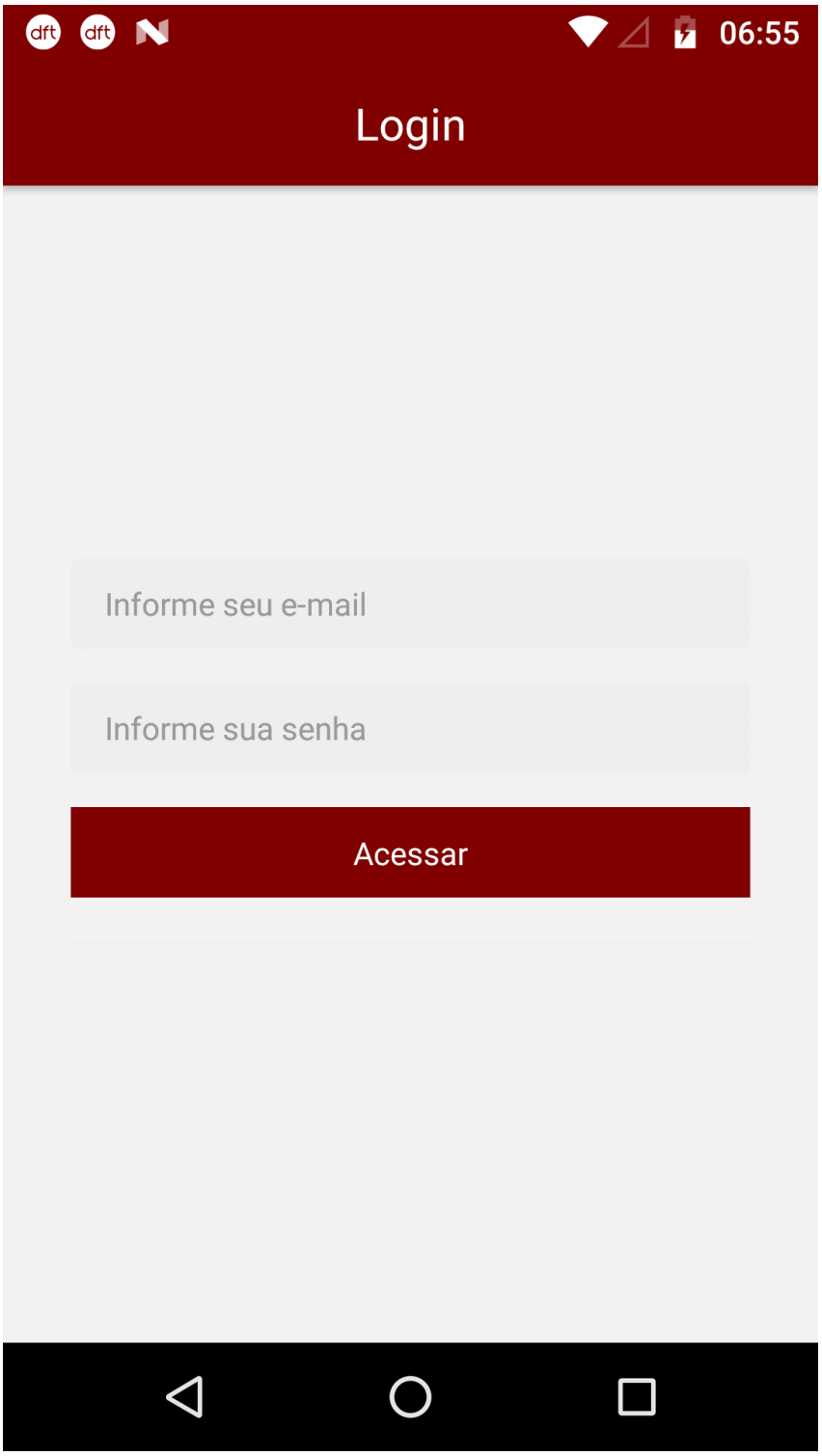
Fonte: Autoria própria (2020).

3.3 APRESENTAÇÃO DO SISTEMA

Nesta sessão serão apresentadas as telas e as funcionalidades do aplicativo. A apresentação será realizada por meio de imagens e descrição delas.

Para realizar o acesso do sistema será necessário efetuar autenticação realizada por meio de e-mail e senha. Figura 5 apresenta a tela de autenticação do sistema. A implementação , foi realizada de forma que o usuário realizar ao realizar o *login* permaneça autenticado até realizar o *logoff* do aplicativo.

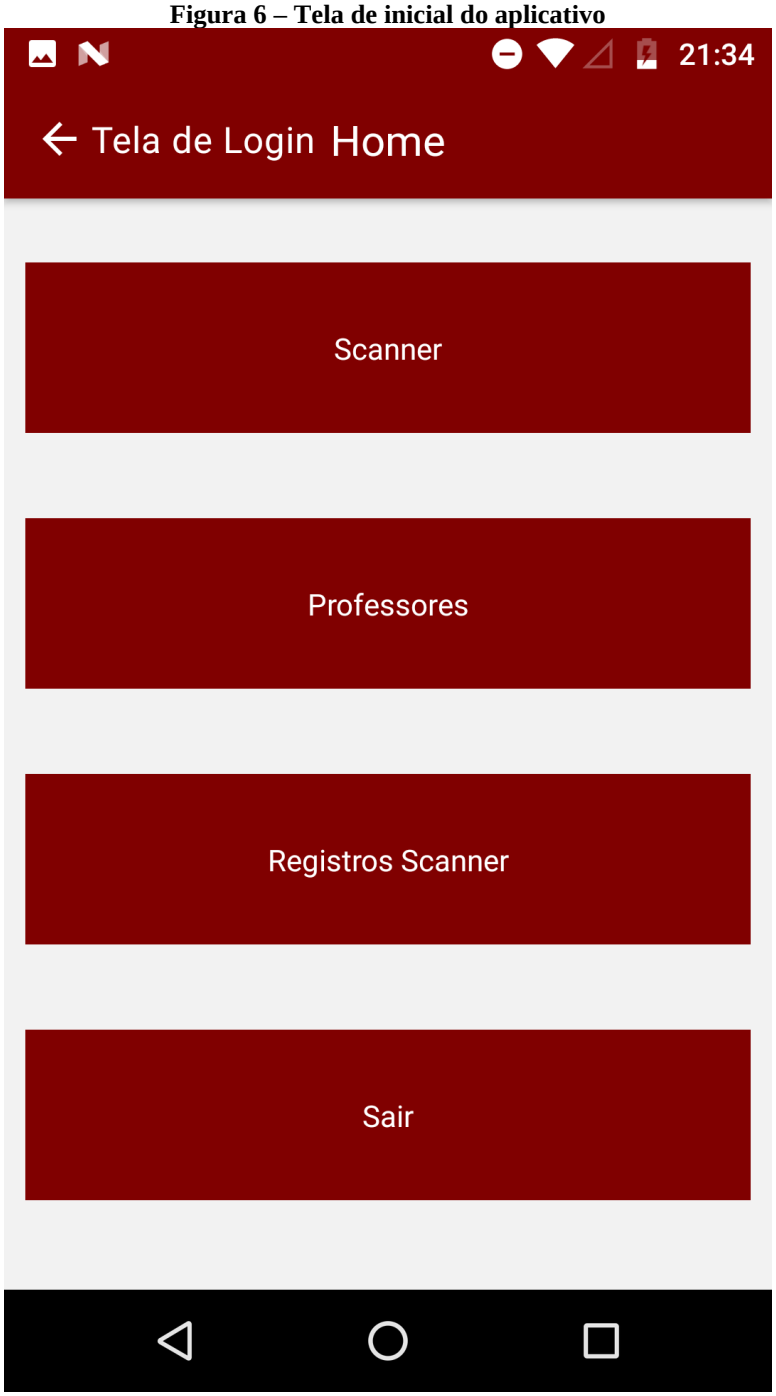
Figura 5 – Tela de autenticação do aplicativo



Fonte: Autoria própria (2020).

A Figura 6 apresenta a tela inicial do aplicativo permite realizar a navegação entre as funcionalidades.

Figura 6 – Tela de inicial do aplicativo



Fonte: Autoria própria (2020).

A Figura 7 apresenta a tela inicial do aplicativo na qual é exibido o QR Code para fazer a leitura dos dados.

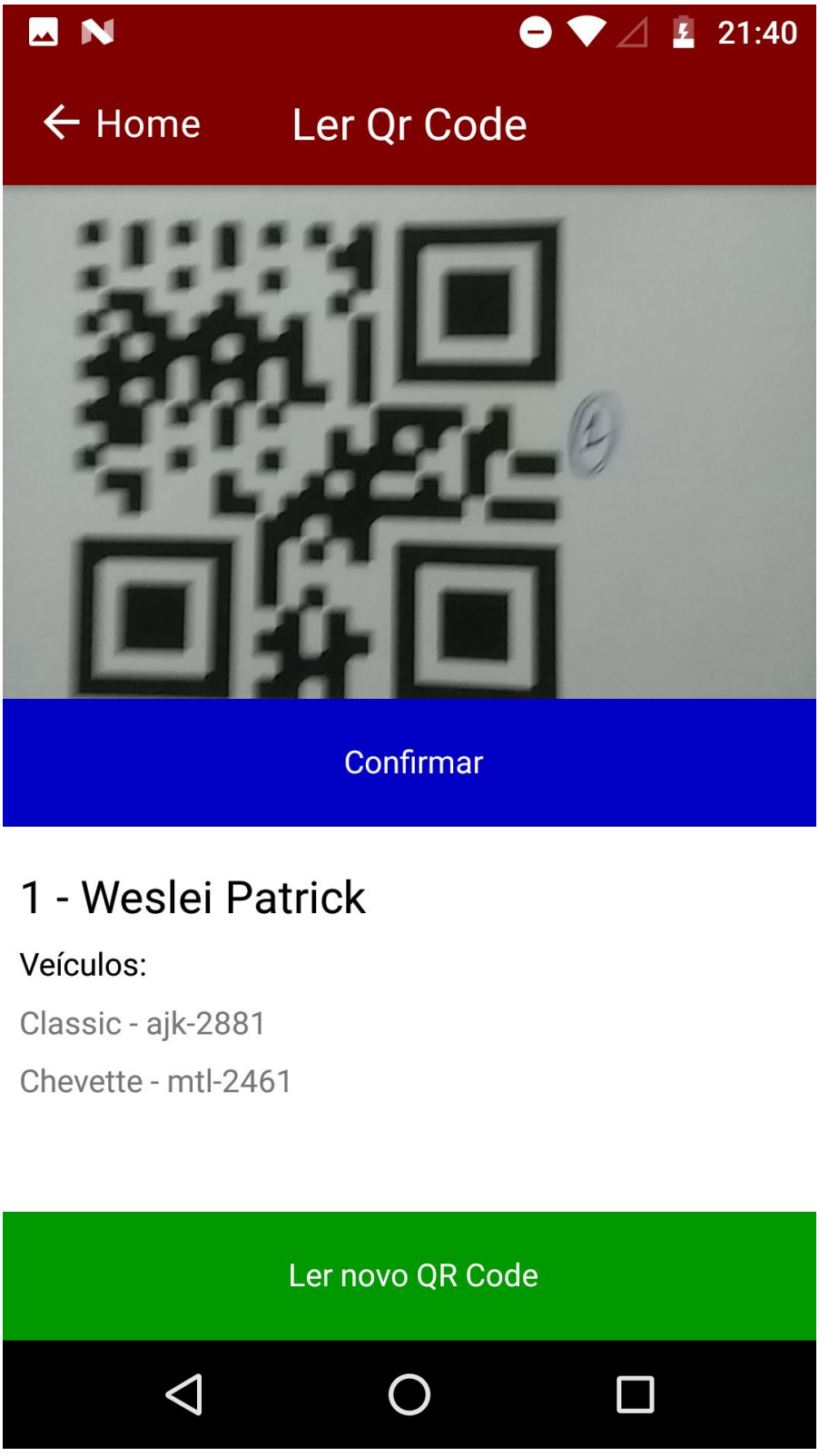
Figura 7 – Tela de inicial do aplicativo

Fonte: Autoria própria (2020).

Após a leitura do QR Code, serão apresentados os dados do usuário correspondente, como o nome e o(s) veículo(s) e, após a confirmação dos dados, o aplicativo inserirá um

registro no banco de dados com os dados do funcionário, data e hora e os dados do usuário que realizou o registro.

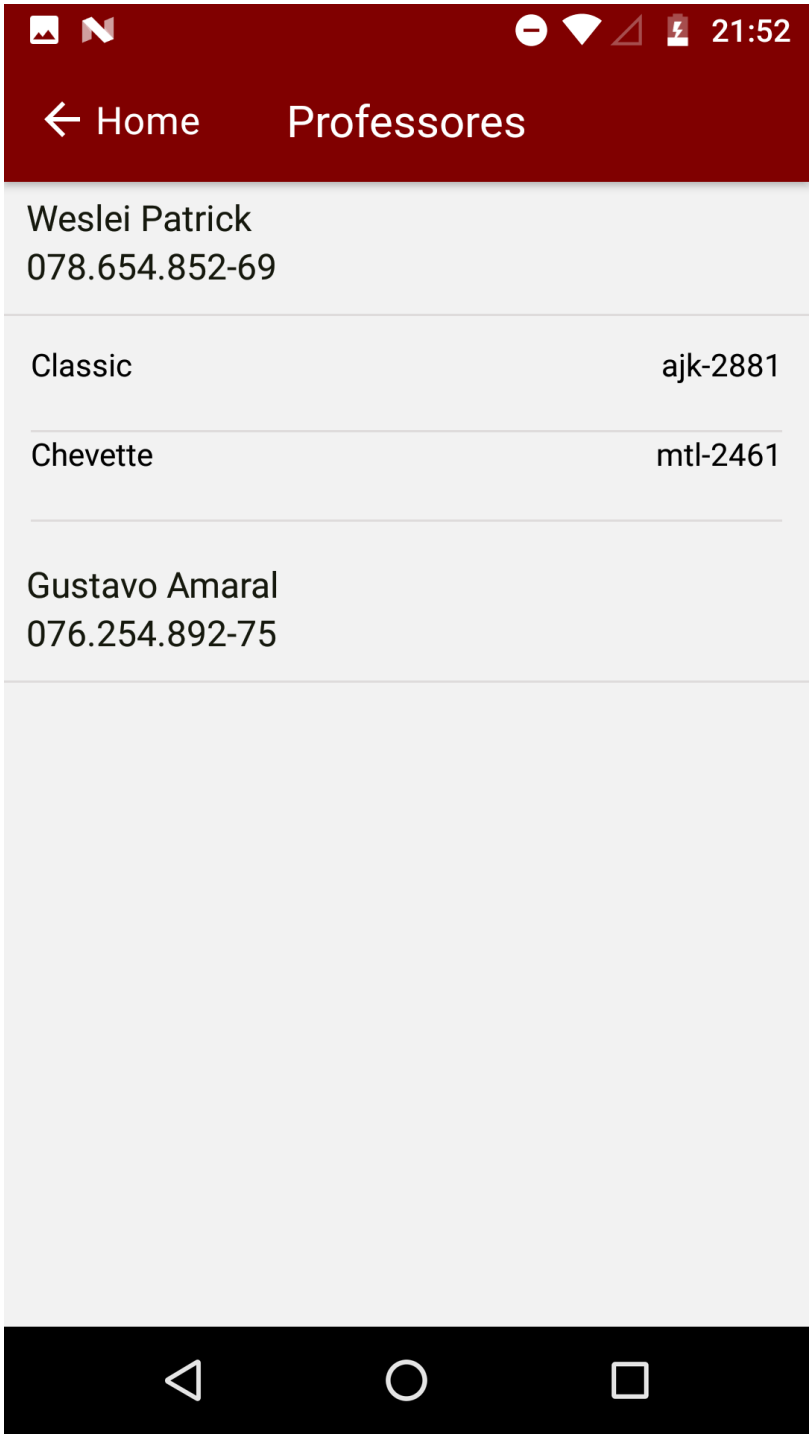
Figura 8 – Tela de inicial do aplicativo



Fonte: Autoria própria (2020).

A Figura 9 apresenta a funcionalidade para visualizar os dados dos funcionários e de seus veículos cadastrados.

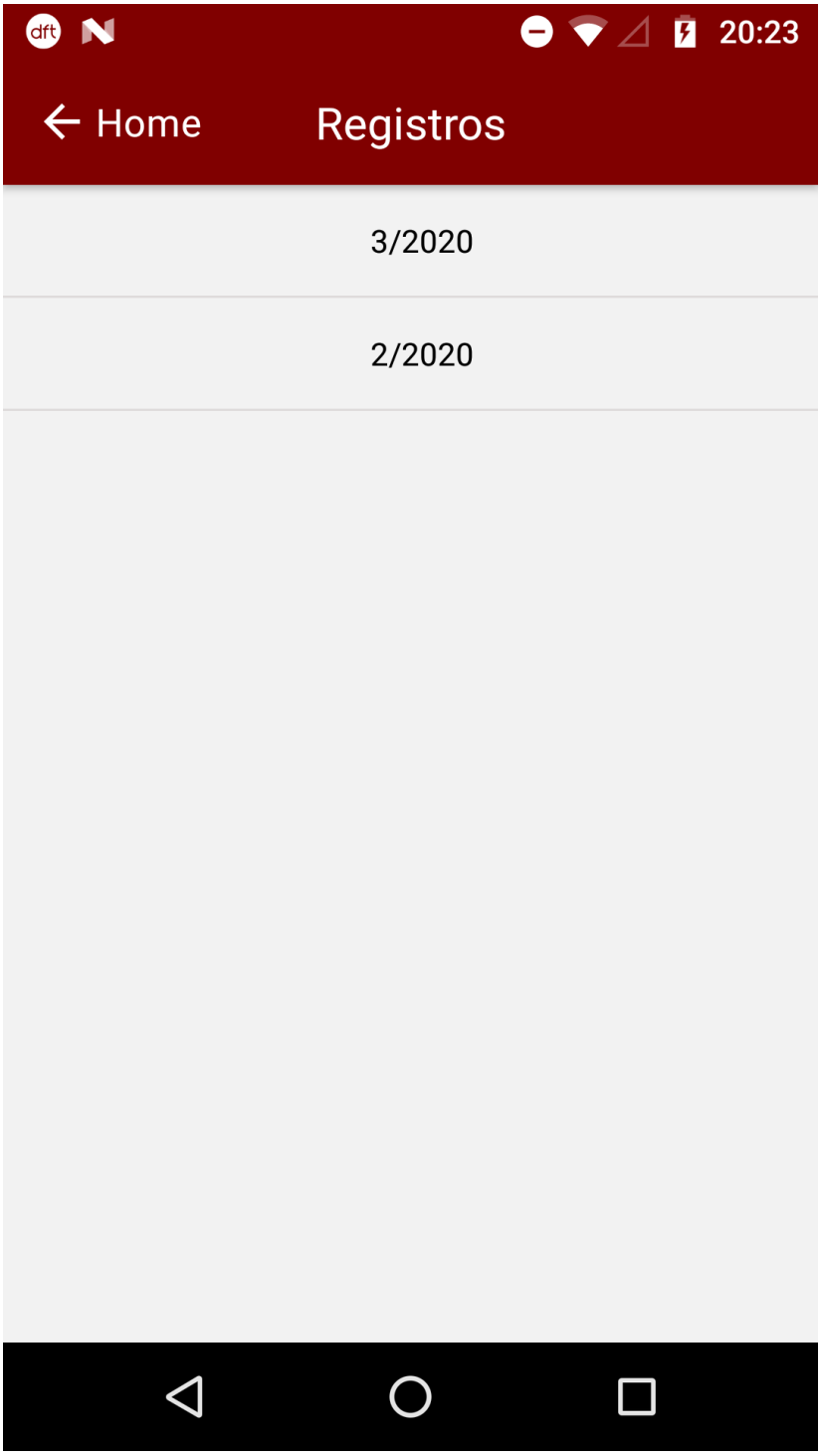
Figura 9 – Tela de inicial do aplicativo



Fonte: Autoria própria (2020).

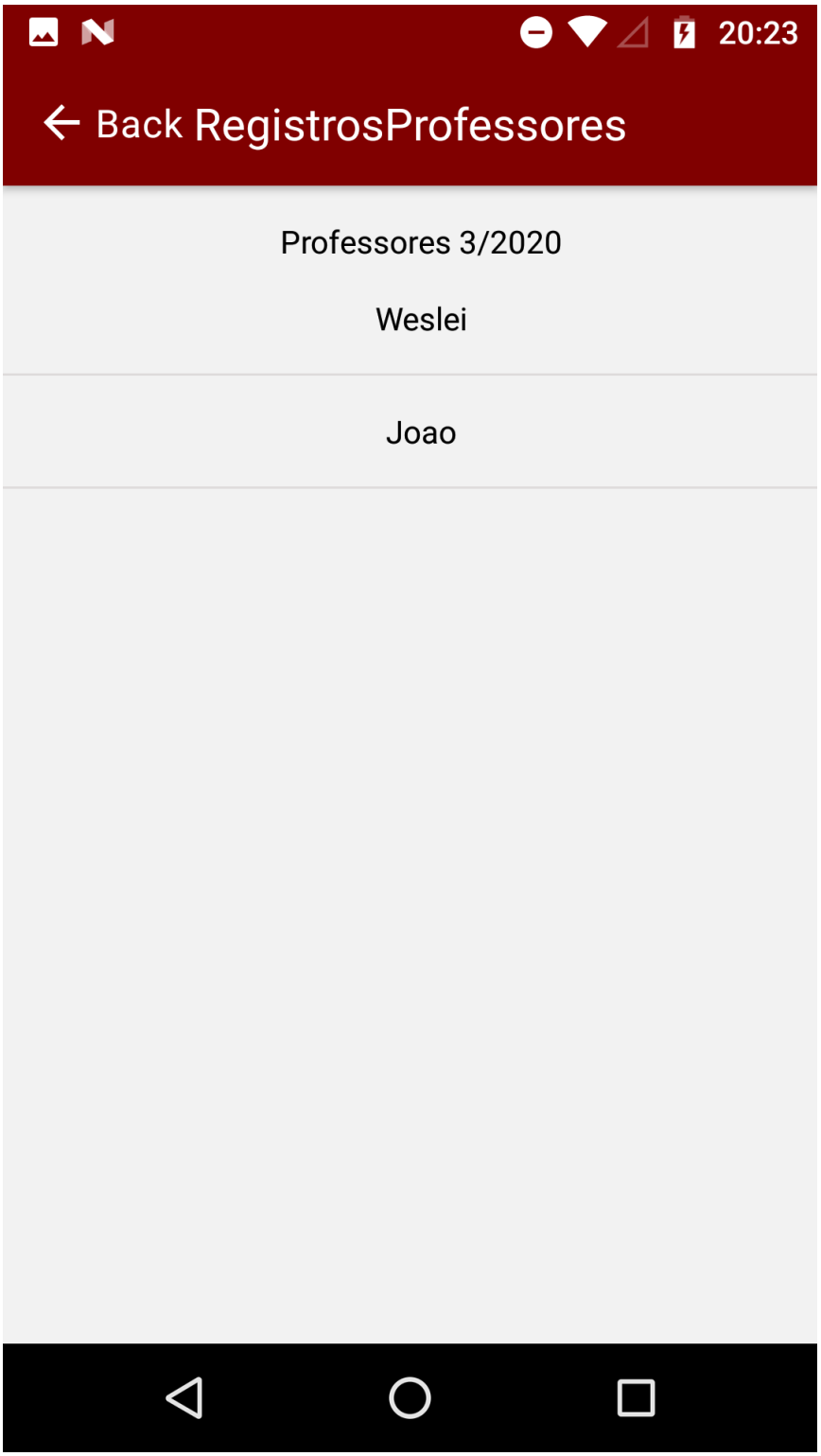
A Figura 10, 11 e 12 apresenta a funcionalidade para visualizar os registros por funcionário, sendo possível consultar quais dias o veículo de determinado funcionário estava no estacionamento.

Figura 10 – Tela de inicial do aplicativo



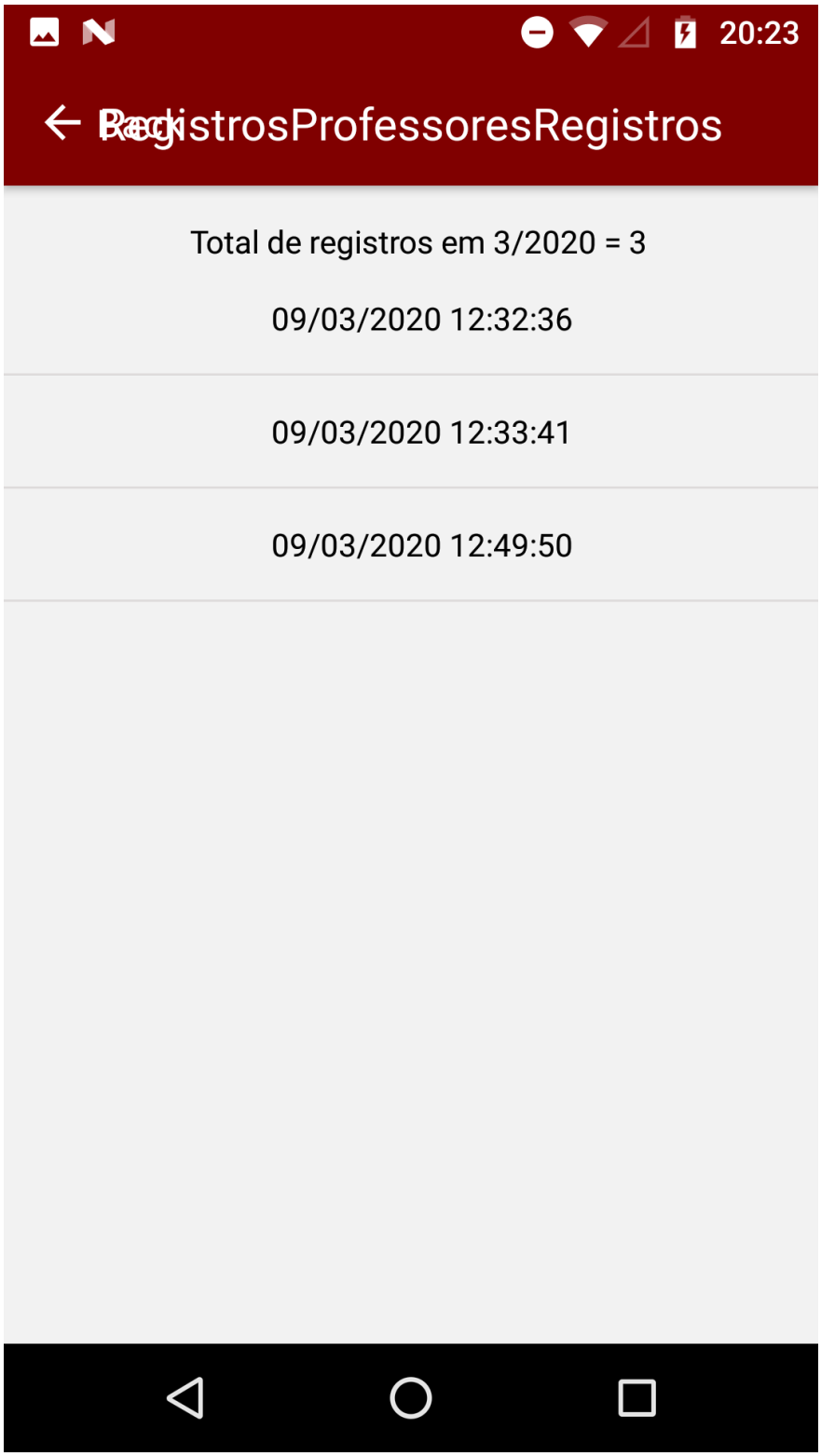
Fonte: Autoria própria (2020).

Figura 11 – Tela de inicial do aplicativo



Fonte: Autoria própria (2020).

Figura 12 – Tela de inicial do aplicativo



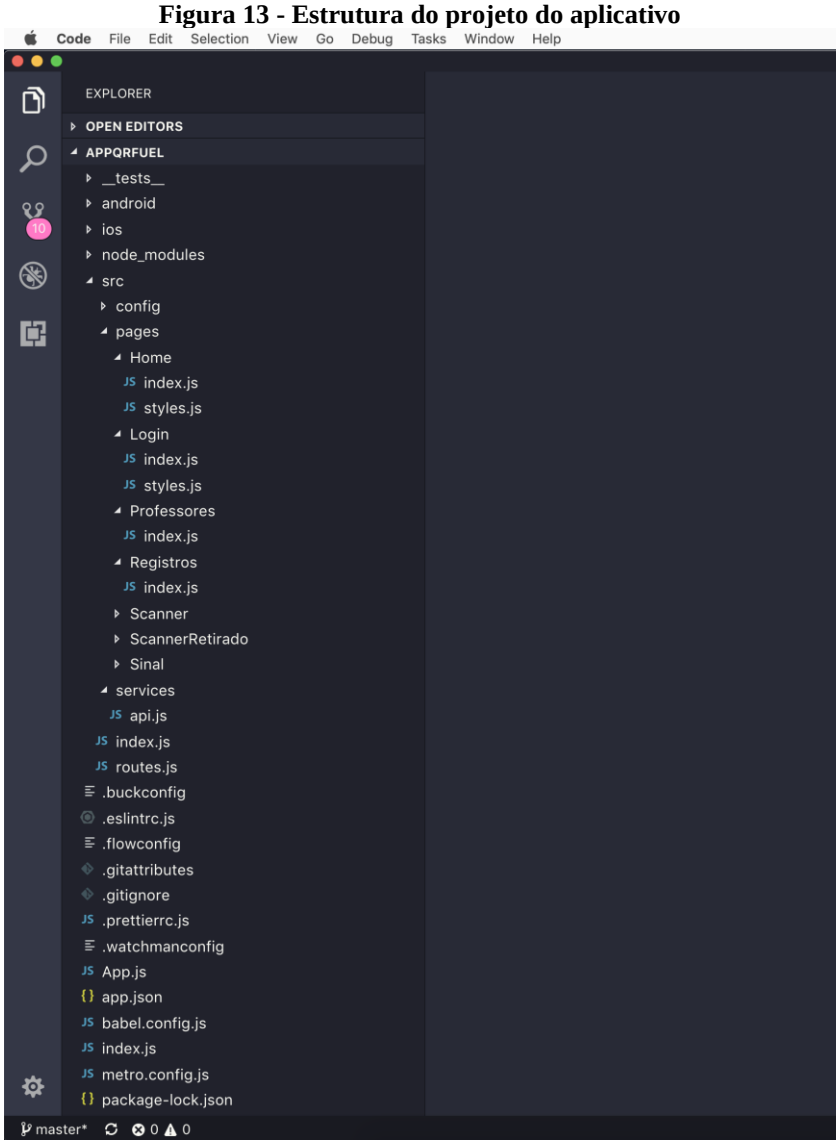
Fonte: Autoria própria (2020).

3.4 IMPLEMENTAÇÃO DO SISTEMA

Nesta sessão são apresentados os principais códigos implementados para o funcionamento do aplicativo. Inicialmente foi preparado o ambiente de desenvolvimento. As principais tecnologias utilizadas foram Rect Native para o desenvolvimento do aplicativo, Java e o *framework* Spring para o desenvolvimento da *Application Programming Interface* (API). Para a comunicação do aplicativo e API foi utilizado o padrão REST, que é baseado no protocolo *Hypertext Transfer Protocol* (HTTP), utilizando os verbos HTTP, GET, POST. Para os dados transferidos foi utilizado o formato *JavaScript Object Notation* (JSON).

A Figura 11 apresenta a estrutura utilizada para organização do projeto do aplicativo.

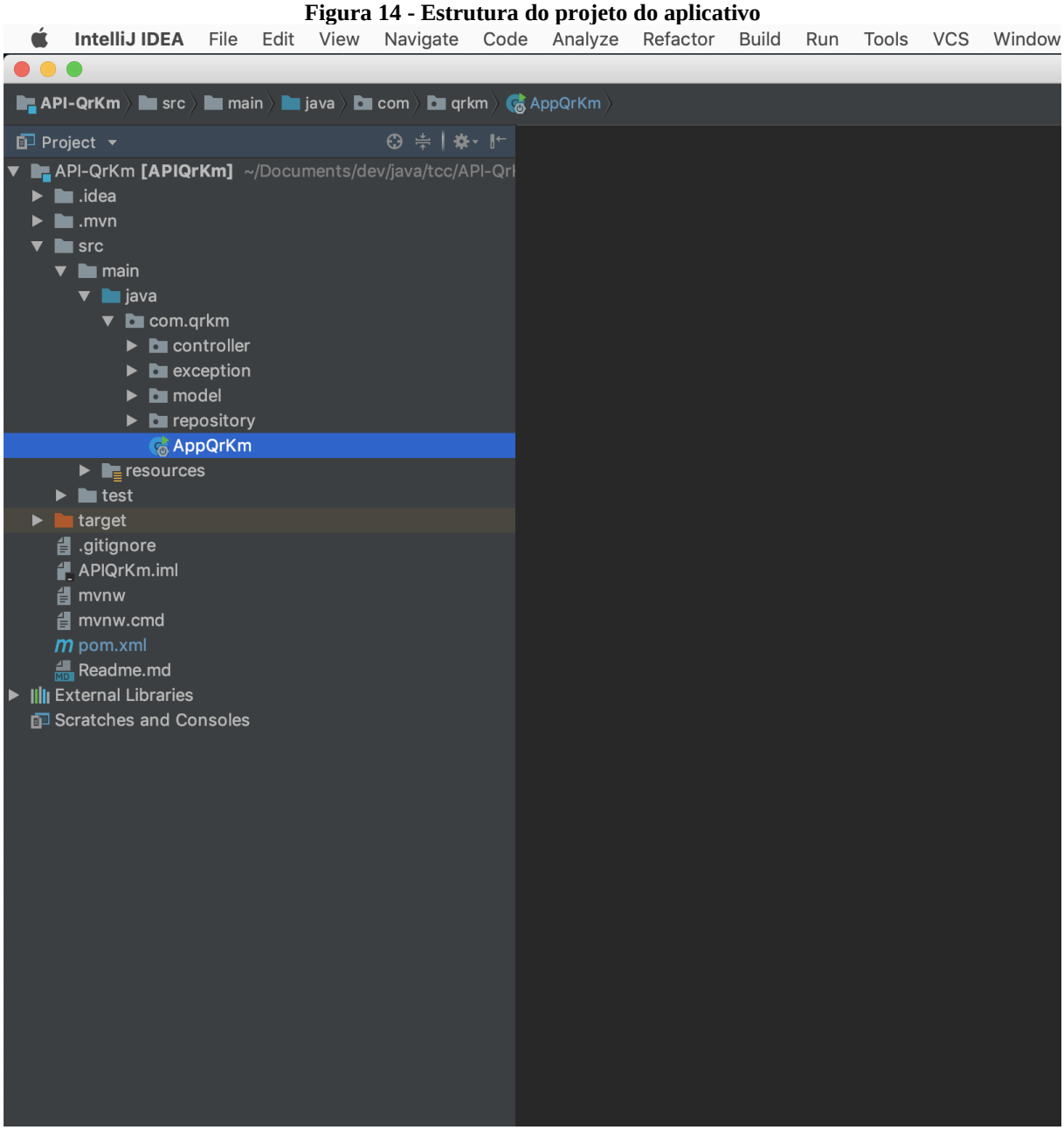
Figura 13 - Estrutura do projeto do aplicativo



Fonte: Autoria própria (2020).

A Figura 12 demonstra a estrutura utilizada para organizacão do projeto da API.

Figura 14 - Estrutura do projeto do aplicativo



Fonte: Autoria própria (2020).

A Listagem 1 refere-se a diversos acesso ao aplicativo. Para que o usuário não precise informar os dados de autenticação sempre que for acessar o aplicativo, foi utilizado o recurso de *AsyncStorage* que possibilita armazenar informações no dispositivo mesmo finalizando o aplicativo. Dessa forma, após o usuário acessar o aplicativo, no próximo acesso é verificado se há dados armazenados no *AsyncStorage* e, caso exista, ele é redirecionado para a tela inicial.

Listagem 1 - Armazenar e recuperar informações do *AsyncStorage*

```
getDadosLogin = async () => {

    let dadosLogin = await AsyncStorage.getItem('dadosLogin')

    if( dadosLogin != null ){
        this.redirHome();
    }
}

storeLogin = async (dados) => {
    await AsyncStorage.setItem('dadosLogin',
        JSON.stringify(
            dados
        )
    );
}

redirHome = () => {
    const { navigation } = this.props;
    navigation.navigate('Home');
}
```

Fonte: Autoria própria (2020).

Uma peculiaridade do *AsyncStorage* no React Native é que o método que o utiliza deve ser *async* e a chamada *await*, isso indica que o método será assíncrono e deverá aguardar a resposta.

A Listagem 2 é referente a requisição HTTP realizada no React Native, para enviar as informações para a API e a retornará se os dados informados estão corretos ou não. Para realizar as requisições HTTP no aplicativo foi utilizado o *feth* que é nativo do JavaScript.

Listagem 2 - Requisição HTTP para realizar autenticação

```
37      autenticar = () => {
38
39          let values = {
40              email : this.state.login,
41              password : this.state.senha
42          }
43
44          fetch(`${API}/api/login`,
45          {
46              method: 'post',
47              headers: {
48                  'Accept': 'application/json',
49                  'Content-Type': 'application/json'
50              },
51              body: JSON.stringify(values)
52          })
53          .then( resp => resp.json() )
54          .then( dados => {
55              this.tratarRetornoLogin( dados );
56          })
57          .catch( erro => {
58            alert( "Falha ao realizar o login" )
59          })
60      }//autenticar
```

Fonte: Autoria própria (2020).

Uma das principais funcionalidades do aplicativo é a leitura do QR Code, para identificar a qual funcionário pertence determinado veículo e realizar o registro, para trabalhar com a leitura do QR Code foi utilizado o pacote react-native-camera. Conforme a Listagem 3 é possível ver a utilização do componente que usará a câmera para leitura do QR Code.

Listagem 3 - Utilizando componente de câmera

```
{/* Camera */}
<View style={{flex: 4, backgroundColor: 'powderblue'}}>
    <RNCamera
        ref={ref => {
            this.camera = ref;
        }}
        style={{
            flex: 1,
            width: '100%',
            height: '30%'
        }}
        type={RNCamera.Constants.Type.back}
        flashMode={RNCamera.Constants.FlashMode.on}
        onGoogleVisionBarcodesDetected={this.barcodeRecognized}
    >
    </RNCamera>
</View>
```

Fonte: Autoria própria (2020).

Com a utilização do *framework* Spring, a inicialização da API é realizada por meio da classe Principal AppQrKm.java, que contém o método *main*, conforme exibido na Listagem 4.

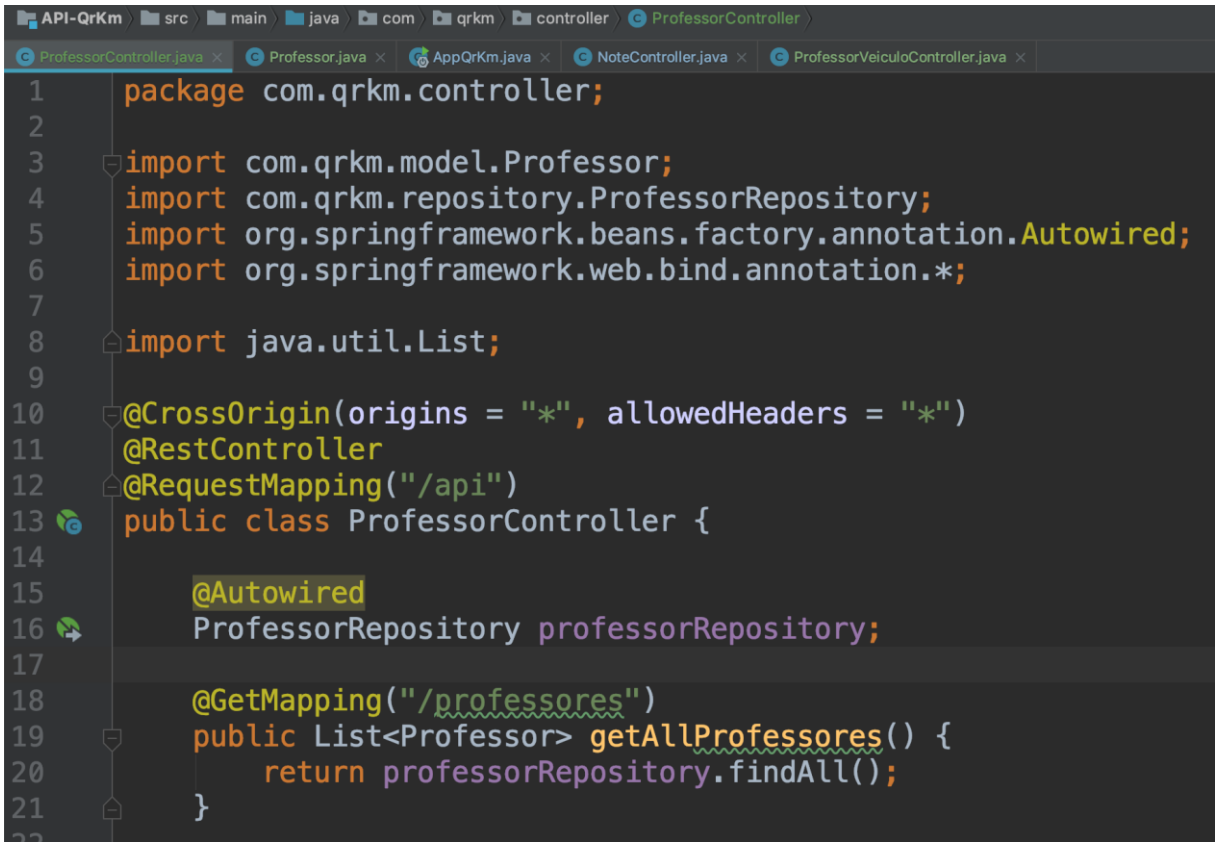
Listagem 4 - Classe principal do Spring

```
1   package com.qrkm;
2
3   import org.springframework.boot.SpringApplication;
4   import org.springframework.boot.autoconfigure.SpringBootApplication;
5   import org.springframework.data.jpa.repository.config.EnableJpaAuditing;
6
7   @SpringBootApplication
8   @EnableJpaAuditing
9   public class AppQrKm {
10
11      public static void main(String[] args) {
12          SpringApplication.run(AppQrKm.class, args);
13      }
14  }
15
```

Fonte: Autoria própria (2020).

No padrão do *framework* Spring as classes que são responsáveis por receber as requisições e tratar, delegar o que deve ser executado, são as classes controladoras, nas quais é possível definir algumas configurações, como *Uniform Resource Locator* (URL), formato que retornará as informações, conforme exibido na Listagem 5.

Listagem 5 - Classe ProfessorController



Fonte: Autoria própria (2020).

Para realizar o mapeamento das tabelas do banco de dados, é necessário realizar a anotação @Table para saber qual entidade está sendo referenciada e para os atributos é utilizada a anotação @Column para realizar o mapeamento das colunas do banco de dados. O código dessa anotação é apresentado na Listagem 6.

Listagem 6 - Anotação @Table

```java
package com.qrkm.model;


import org.springframework.data.annotation.CreatedDate;
import org.springframework.data.annotation.LastModifiedDate;

import java.util.List;

import javax.persistence.*;
import javax.validation.constraints.NotBlank;
import java.util.Date;

@Entity
@Table(name = "professores")

public class Professor {

    private static final long serialVersionUID = 1L;

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    @NotBlank
    private String nome;

    @NotBlank
    private String cpf;

    @OneToMany
    private List<ProfessorVeiculo> veiculos;


    @Column(nullable = true, updatable = false)
    @Temporal(TemporalType.TIMESTAMP)
    @CreatedDate
    private Date createdAt;

    @Column(nullable = true)
    @Temporal(TemporalType.TIMESTAMP)
    @LastModifiedDate
    private Date updatedAt;

    public Long getId() { return id; }

    public void setId(Long id) { this.id = id; }

    public List<ProfessorVeiculo> getVeiculos() { return veiculos; }
```

Fonte: Autoria própria (2020).

4 CONSIDERAÇÕES FINAIS

Este trabalho teve como objetivo apresentar o desenvolvimento de um aplicativo *mobile* híbrido utilizando a tecnologia React Native. Durante o desenvolvimento do projeto foram utilizados recursos de câmera para leitura de QR Code, comunicação com API REST e *frameworks*, como Spring para desenvolvimento da API, entre outros.

Os objetivos propostos foram cumpridos, destacando-se o desenvolvimento de um aplicativo que identifique os veículos e consequentemente os proprietários que foram até o local de trabalho com veículo próprio e gerar relatório das informações coletadas.

As dificuldades encontradas foram as de trabalhar com tecnologias desconhecidas. Contudo, com os desafios do projeto foi possível adquirir conhecimento e realizar a implementação necessária, sendo, assim, agregando muito valor em termos de aprendizado.

Como implementação futuras em termos de melhorias e agregação de funcionalidades, destaca-se, a implementação do Spring Security e JWT para que seja possível realizar requisições somente por aplicações autorizadas.

Finalizando, pode-se afirmar que houve assertividade em termos das tecnologias utilizadas para o desenvolvimento do projeto, pois observou-se qualidade nos resultados obtidos e agilidade no desenvolvimento.

REFERÊNCIAS

CHIAVENATO, Idalberto. **Introdução à teoria geral da administração.** 4. ed. Barueri - Sp: Manole, 2014. 536 p

DEVMEDIA. **Guia de React Native: Primeiros Passos.** Disponível em: https://www.devmedia.com.br/react-native/. Acesso em: 01 mar. 2020.

L3SOFTWARE. **JetBrains – IntelliJ IDEA.** Disponível em: https://l3software.com.br/produto/jetbrains-intellij-idea/. Acesso em: 01 mar. 2020.

VISUALSTUDIO. **Documentação para código do Visual Studio.** Disponível em: https://code.visualstudio.com/docs. Acesso em: 01 mar. 2020.

MDN. **JavaScript.** Disponível em: https://developer.mozilla.org/pt-BR/docs/Web/JavaScript. Acesso em: 01 mar. 2020.

FERNANDES, Diego. **O que é JavaScript?** Disponível em: https://rocketseat.com.br/starter/curso-gratuito-javascript. Acesso em: 01 mar. 2020.

DEVMEDIA. **Guia Completo de Java.** Disponível em: https://www.devmedia.com.br/guia/linguagem-java/38169. Acesso em: 28 fev. 2020.

DEVMEDIA. **Introdução às plataformas Java.** Disponível em: https://www.devmedia.com.br/introducao-as-plataformas-java/29544. Acesso em: 28 fev. 2020.

WEISSMANN, Henrique Lobo. **Vire o jogo com Spring Framework.** São Paulo: Casa do Código, 2014.