LimpaMais: aplicativo mobile para prestação de serviços de limpeza

Gabriel de Barros Gambôa, Fernando Sambinelli

Instituto Federal de Educação, Ciência e Tecnologia de São Paulo, Campus Hortolândia (IFSP) Avenida Thereza Ana Cecon Breda, s/n, Vila São Pedro - Hortolândia – SP – Brasil

gabriel.gamboa@aluno.ifsp.edu.br,sambinelli@ifsp.edu.br

Abstract. *In Brazil, the impact caused by the coronavirus on the income of informal sector workers, more specifically day laborers, was very high, causing a high rate of unemployment. Thinking about the large use of cell phones by the population, this work aims to create an application for mobile devices in order to connect cleaning service providers with potential customers.*

Resumo. *No Brasil, o impacto causado pelo coronavírus na renda de trabalhadores do ramo informal, mais especificamente diaristas, foi muito alto, causando um grande índice de desemprego. Pensando na grande utilização de celulares pela população, este trabalho tem como objetivo a criação de um aplicativo para dispositivos móveis com o intuito de conectar prestadores de serviço de limpeza com potenciais clientes.*

1. Introdução

No contexto da pandemia da COVID-19, o grande número de mortos e o isolamento social forçado causou inúmeras mudanças no modo de vida de todas as populações ao redor do mundo, inclusive no Brasil. Além disso, também causou drásticas alterações no mercado de trabalho, que acabou alavancando uma enorme onda de desemprego com impactos ainda mais severos para aqueles que estão vinculados à informalidade (Costa, 2020).

Segundo uma pesquisa realizada em 2020 pela Organização Internacional do Trabalho, o impacto causado dentro do mercado é crítica para trabalhadores desprotegidos e para os grupos vulneráveis que estão dentro do mercado de trabalho informal, já que laboram sem um registro formal e excluídos do contexto da legislação trabalhista (OIT, 2020). As trabalhadoras domésticas foram um dos grupos mais afetados pelas consequências causadas pela pandemia (Bohrer, 2021).

De acordo com uma pesquisa publicada pelo Instituto Locomotiva, 39% dos empregadores de diaristas optaram por encerrar o serviço desses profissionais sem manter o seu pagamento (Instituto Locomotiva, 2020), acarretando numa grande dificuldade financeira para as trabalhadoras.

Tendo sua utilização cada vez mais acentuada pela sociedade nos dias atuais, os dispositivos móveis continuam ganhando grande destaque, inclusive no Brasil. Segundo Meirelles (2021), a pesquisa anual realizada pelo Centro de Tecnologia da Informação Aplicada (FGVcia) mostra que já são aproximadamente 242 milhões de celulares em uso no Brasil, mais de um *Smartphone* por habitante. Essa transformação digital ocorreu de forma ainda mais acelerada em 2020 devido à pandemia (FGVcia apud Meirelles, 2021).

Considerando a ampla adoção dos celulares no Brasil e a necessidade de incremento de renda das trabalhadoras domésticas afetadas pela pandemia, o objetivo deste trabalho foi desenvolver uma aplicação para dispositivos móveis denominado "LimpaMais" para conectar prestadores de serviço de limpeza com clientes que desejarem solicitar por seus serviços, com o intuito de motivar e auxiliar financeiramente diaristas que sofreram as consequências

causadas pelo coronavírus no Brasil, abrindo um espaço para que continuem utilizando o aplicativo mesmo após o fim da pandemia como meio de oferecerem seus serviços.

Este artigo está estruturado da seguinte maneira: a Seção 2 apresenta o referencial teórico, na Seção 3 são descritos os trabalhos correlatos do projeto, a Seção 4 descreve as metodologias utilizadas na construção do projeto, a Seção 5 mostra a fase de desenvolvimento e a Seção 6 mostra a conclusão.

2. Referencial Teórico

O objetivo desta seção é apresentar os fundamentos teóricos envolvidos neste trabalho. Inicialmente, são apresentadas as principais teorias sobre prestação de serviço no contexto digital. Em seguida, é realizada uma breve revisão sobre o desenvolvimento de aplicações móveis.

2.1 Prestação de serviços no contexto digital

Devido à crescente transformação tecnológica, o mundo do trabalho virtual começou a ganhar destaque. Diversas empresas começaram a construir plataformas para dispositivos móveis com o intuito de aproximar de forma otimizada e eficaz a oferta de prestadores de serviços e a demanda de clientes que desejarem contratar esses trabalhadores (Carelli, Cavalcanti e Fonseca, 2020).

O termo *Gig Economy* (Economia de Bico) surgiu para representar uma parcela do mercado de trabalho em que os trabalhadores buscam um sustento por trabalhos temporários através de plataformas digitais (Lisboa, 2021). O tamanho das empresas que agem dentro da economia de bico variam muito, desde a Uber e Airbnb, que oferecem serviços de transporte e hospedagem, até empresas que sequer se aproximam do tamanho dessas empresas (Oitaven, Carelli e Casagrande, 2018).

Com a viralização das plataformas voltadas à economia de bico, esse modo de trabalho começou a ganhar reconhecimento, sendo compreendido massivamente por duas formas de trabalho principais: ‘‘*crowdwork*’’ e “*on-demand*”. O primeiro abrange a realização de tarefas por grupos de trabalhadores a partir de uma plataforma online, e o segundo trata da realização de trabalhos tradicionais (como de limpeza e transporte, por exemplo), demandados dentro de um aplicativo (Oitaven, Carelli e Casagrande, 2018). A possibilidade de ter essa conexão virtualizada pelos trabalhos “*on-demand*” promoveu com bastante facilidade o encontro dos prestadores de serviço com os clientes, algo que dificilmente aconteceria por meios presenciais ou físicos (Carelli, Cavalcanti e Fonseca, 2020).

2.2 Desenvolvimento de aplicações móveis

A popularização dos dispositivos móveis ao redor do mundo fez com que o desenvolvimento de aplicações para esses dispositivos ganhasse sua devida atenção. Existem diversos sistemas operacionais para os dispositivos existentes atualmente, e para cada tipo de sistema operacional existem linguagens de programação específicas para ser utilizada na construção dos aplicativos, como por exemplo o Java e o Kotlin para a plataforma Android e o Objective-C para a plataforma iOS (White, 2013), sendo essas duas as plataformas mais frequentemente usadas famosas entre os dispositivos (Goasduff, 2021).

As aplicações criadas através dessas linguagens específicas são chamadas de nativas, pois conseguem, através das *Application Programming Interfaces (API*s*)* disponibilizadas,

acesso aos recursos de hardware do dispositivo, como câmera, geolocalização e microfone (White, 2013). Os aplicativos nativos são distribuídos pelas próprias empresas que cuidam de cada um dos sistemas operacionais através das suas lojas de aplicativos, podendo haver uma verificação dos aplicativos antes de sua publicação dentro das lojas (White, 2013). Em contramão, existem aplicativos não nativos, que são projetados para serem compatíveis em múltiplos sistemas operacionais que são construídos com apenas um único código, geralmente escrito em HTML, CSS e Javascript (Martins, 2020). Uma das principais desvantagens ao construir uma aplicação híbrida é o fato dos recursos nativos do dispositivo não poderem ser acessados diretamente através da linguagem utilizada. "Os aplicativos não nativos podem acessar alguns desses recursos, como a câmera ou a localização do usuário, mas o fazem usando métodos não ideais", cita Martins (2020).

Antes do surgimento das ferramentas para construção de aplicativos híbridos, desenvolver aplicativos para plataformas Android e iOS era complexo, pois segundo Martins (2020) era necessário que as empresas contratassem uma equipe de desenvolvimento para cada sistema operacional, sendo que uma equipe programaria usando Java, para o sistema operacional Android, e outra equipe atuaria com Objective-C no iOS. Além do tempo de desenvolvimento elevado para criar duas vezes o mesmo aplicativo para plataformas diferentes, o custo consequentemente era maior, devido a necessidade em ter times de desenvolvimento com conhecimento em cada uma das linguagens de programação citadas.

Com o avanço da tecnologia, foram surgindo diversas bibliotecas e *frameworks* a fim de facilitar o processo de desenvolvimento de aplicativos móveis híbridos, tornando o processo mais produtivo. Atualmente, há uma enorme gama de *frameworks* específicos para cada tipo de necessidade. Dentre os mais comuns que utilizam as tecnologias padrão da web para a criação das aplicações, existem também aqueles que necessitam de apenas uma única base de código para desenvolver aplicativos *cross-platform* nativos, sendo os mais populares Flutter com a linguagem Dart, React Native com a linguagem JavaScript e Xamarin com a linguagem C# (Rees, 2021). Apesar das suas diferenças de desempenho e linguagens utilizadas, a utilização destes *frameworks* no desenvolvimento de aplicações móveis trazem uma grande semelhança que é na questão na utilização de componentes, pequenos trechos de código isolados que podem ser reutilizáveis e adaptáveis em outras partes do programa (Rees, 2021). A Figura 1 mostra um exemplo comparativo de como os *frameworks* Flutter e React Native conseguem se comunicar com os recursos nativos do dispositivo.

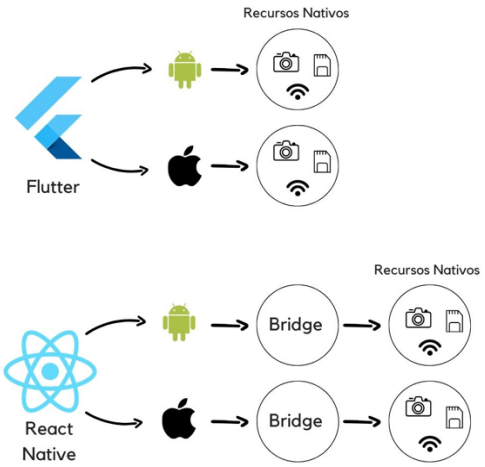


Figura 1. Diferença entre Flutter e React Native (Andrade, 2020).

3. Trabalhos correlatos

Esta seção tem o objetivo de descrever as principais funcionalidades de alguns aplicativos existentes relacionados que serviram como base para a proposta do desenvolvimento deste trabalho.

3.1 *Service Advisor*

Service Advisor é uma aplicação para dispositivos móveis criada por Cairo Araújo Santos e Miguel Gomes Aleixo Ferreira Lima que possui um objetivo similar ao LimpaMais: mediar a comunicação entre solicitantes e prestadores de serviço. Dentro do aplicativo, os usuários do tipo prestadores podem oferecer seus serviços de diversos tipos para que fiquem disponíveis para serem pesquisados e solicitados por clientes que desejarem contratar seus serviços. (Santos e Lima, 2020). A Figura 2 mostra a interface de pesquisa por serviços no aplicativo.

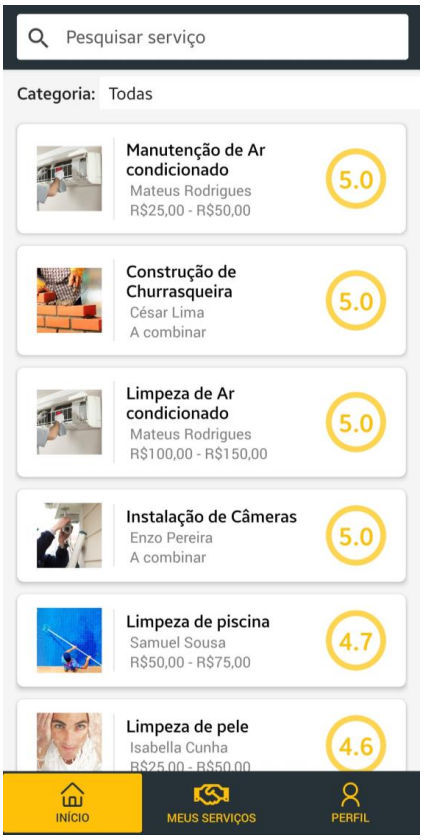


Figura 2. Tela pesquisa de serviços do Service Advisor (Santos e Lima, 2020)

3.2 *Yellow List*

Yellow List é um projeto de aplicativo móvel criado para a plataforma Android por André Felipe Angeloni Rodrigues, Fábio Stefanini da Silveira e Mateus Guilherme Fuini que disponibiliza uma forma de localizar prestadores de serviço utilizando a função de geolocalização do aparelho. Dentro do aplicativo, é possível realizar uma autenticação via rede social como o Facebook ou realizar um cadastro simples para acessar a plataforma. Pensando no contexto de serviços, o solicitante pode emitir uma qualificação do serviço prestado para que outros usuários consigam visualizar sobre as avaliações de determinado serviço (Rodrigues, Silveira e Fuini , 2019). A Figura 3 mostra a interface de login do aplicativo.

Figura 3. Tela inicial do Yellow List (Rodrigues, Silveira e Fuini , 2019)

4. Metodologias

Inicialmente, foi efetuado o levantamento de aplicativos semelhantes que têm como base a prestação e solicitação de serviços para analisar como seria estruturado as regras de negócio e funcionalidades da aplicação. Após isso, para a definição dos requisitos foram criadas histórias de usuário, juntamente com a criação de protótipos de média fidelidade com a ferramenta de prototipação Figma, a fim de validar entradas e estrutura das interfaces. Definidos os requisitos, foi feito um esquema da solução técnica para definir as tecnologias a serem utilizadas para o desenvolvimento do projeto, assim como a arquitetura.

Anterior ao início do desenvolvimento do aplicativo e da *API*, foi realizada a modelagem das tabelas que irão servir para armazenar os dados da aplicação. Após isso, criou-se o banco de dados relacional através do sistema gerenciador de banco de dados PostgreSQL..

Para cada funcionalidade, foi criado e implementado dentro de uma *API* desenvolvida em NodeJS rotas correspondentes para consultar, alterar ou excluir registros dentro do banco de dados, fornecendo sempre ao final das requisições uma resposta ao usuário sobre a ação realizada. As funcionalidades mais importantes, como cadastro no sistema, login, consulta e agendamento dos serviços foram primeiramente implementadas.

A fim de construir o aplicativo móvel que será utilizado pelos usuários finais, utilizou-se o *framework* Flutter para criar a aplicação tanto para iOS quanto para Android. As interfaces implementadas foram desenvolvidas utilizando-se o padrão *Material Design* da Google (Google, 2021).

No fim do desenvolvimento, será feita a validação do aplicativo através de uma coleta de *feedbacks* de potenciais clientes da aplicação sobre sua satisfação em relação ao aplicativo. A coleta será realizada através do Google Forms.

5. Desenvolvimento

A Seção abaixo demonstra as fases do desenvolvimento do trabalho, interfaces, funcionalidades e execução da aplicação.

5.1 Entendimento da visão geral de negócio

Primeiramente, foi definido e criado um fluxo da visão geral de negócio do aplicativo para representar como o aplicativo seria utilizado pelos clientes e diaristas. A Figura 4 demonstra o fluxo criado.

O fluxo inicia-se quando um usuário entra no aplicativo a fim de buscar por diaristas (passo 1) para realizar um serviço de limpeza. Quando encontrar a diarista que deseja, o usuário realiza uma solicitação de agendamento (2) com informações pertinentes à solicitação, como data, horário e endereço do local. Ao confirmar a solicitação, a diarista recebe uma notificação pelo aplicativo (3) para confirmar ou não o agendamento do serviço (4). Depois de confirmar, um registro de agendamento é criado e notificado ao usuário (5). Ao chegar no dia do agendamento, a diarista realiza o serviço de sua diária ao usuário (6) e , após o término do trabalho, o usuário pode avaliar o serviço prestado (7), realizando no fim uma atualização da reputação da diarista dentro do aplicativo.

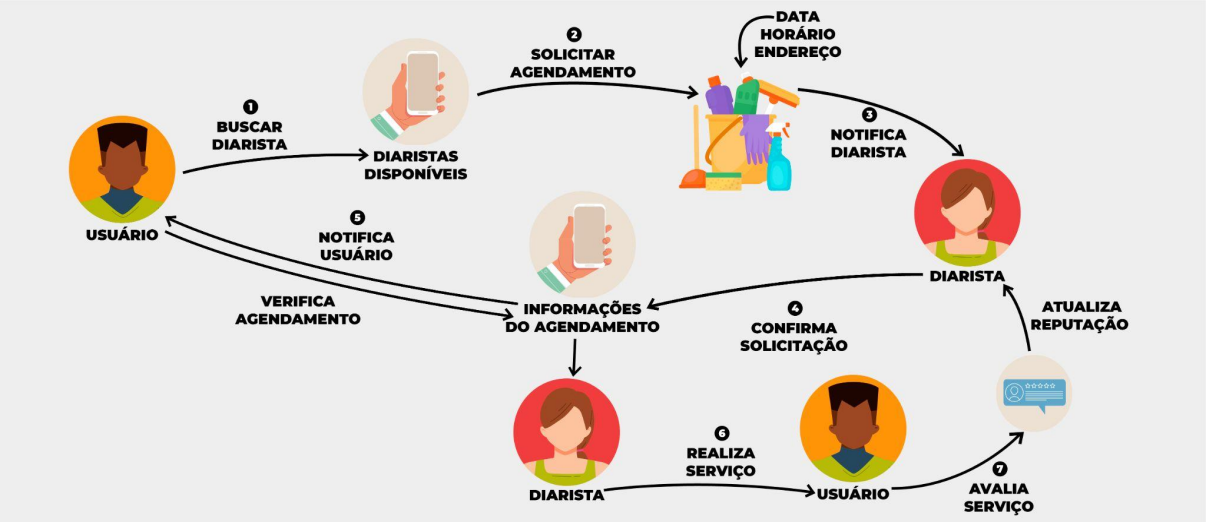
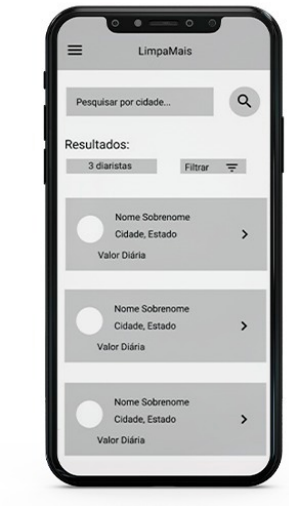


Figura 4. Visão de negócio. Fonte: elaborado pelo autor.

5.2 Prototipação da Interface e Histórias de Usuário

Para criar-se uma ideia das telas do aplicativo, foram criados os protótipos das interfaces que serão utilizadas. O objetivo da prototipação é elaborar um modelo de um produto final que não pretenda demonstrar sua qualidade técnica, mas sim sua ideia de usabilidade, estilização e propósito (Noleto, 2020).

Juntamente com o desenvolvimento dos protótipos, foram sendo criados e definidos os critérios de aceitação e histórias de usuário para cada uma das interfaces. A proposta das histórias é levantar requisitos de uma aplicação pensando pela perspectiva do cliente, fornecendo à equipe de desenvolvimento a capacidade necessária para visualizar o que o usuário precisa (Longo e Silva, 2014). A Figura 4 demonstra um exemplo de uma interface prototipada juntamente com sua respectiva história de usuário e seus critérios de aceite.



Pesquisar por diaristas

Como um solicitante de serviços de limpeza
Eu quero pesquisar por diaristas próximas à minha localização
Para visualizar todas as diaristas disponíveis

Critérios de aceite:

- Deve haver uma caixa de pesquisa para pesquisar a cidade das diaristas e um botão para confirmar a pesquisa
- Quando o botão de pesquisa for pressionado, o aplicativo deve realizar uma busca pelas diaristas através da cidade mencionada e mostrar o resultado da quantidade de diaristas retornadas através da consulta pela cidade
- Deve haver a opção de filtrar as diaristas encontradas por valor da diária (maior ou menor) e também a distância entre o solicitante e a prestadora
- Ao clicar em uma diarista, é necessário abrir uma tela de detalhes com suas informações.

Figura 5. Protótipo da tela de pesquisa por diaristas. Fonte: elaborado pelo autor.

5.4 Criação da solução técnica

Na etapa da criação da solução técnica, foi realizada uma definição da arquitetura geral da aplicação, juntamente com a definição das tecnologias e ferramentas necessárias para o desenvolvimento do aplicativo LimpaMais e da *API*. A Figura 6 mostra o esquema criado para representar a visão geral da arquitetura do sistema.

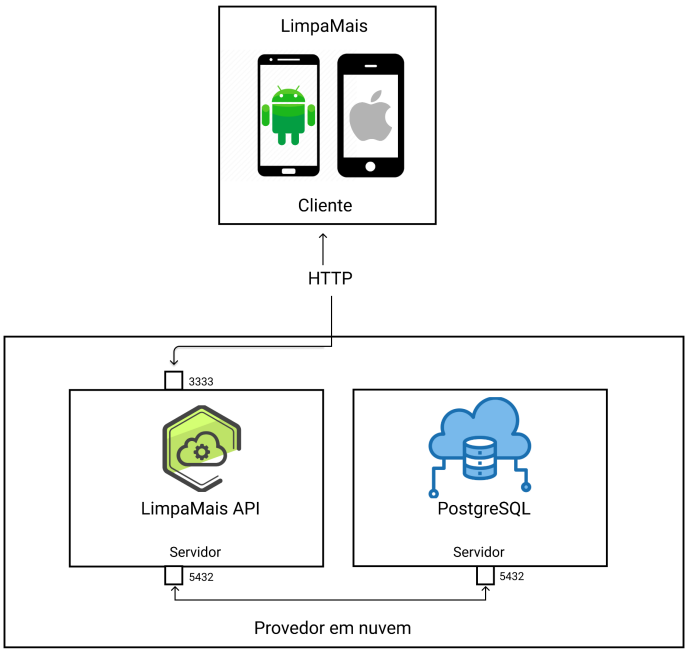


Figura 6.Visão geral da arquitetura da aplicação. Fonte: elaborado pelo autor.

Nesse modelo, a arquitetura é separada entre dois componentes principais: o cliente e o provedor em nuvem. Os celulares com o aplicativo LimpaMais instalados em seu sistema, tanto Android quanto iOS, funcionarão como clientes que irão consumir a *API* que, juntamente com o banco de dados, estão hospedados em servidores disponíveis pelo provedor. Para realizar a comunicação com o servidor em nuvem, a aplicação utiliza o protocolo HTTP,

através da porta 3333. Já a comunicação entre a *API* e o banco de dados postgreSQL é realizada através da porta 5432.

5.4 Modelagem do Banco de Dados

A modelagem de um Banco de Dados é um passo muito importante para qualquer desenvolvimento de software, pois é com ela que os desenvolvedores da aplicação conseguem ter uma visão geral das entidades envolvidas dentro do negócio. O principal objetivo de uma modelagem dos dados, segundo Heuser (2001), é fornecer uma descrição dos tipos de informações que estão armazenadas dentro do Banco de Dados. Com isso, ela facilita a forma de definir o modo e a maneira como os dados da aplicação serão persistidos.

Foi utilizado o modelo conceitual para representar a modelagem das entidades e dados relacionados dentro do trabalho. Para Heuser (2004), esse modelo registra a estrutura que o Banco de Dados terá, independentemente do Sistema de Gerenciamento de Banco de Dados (*SGBD*) utilizado. A Figura 7 ilustra a modelagem realizada para o Banco de Dados da aplicação.

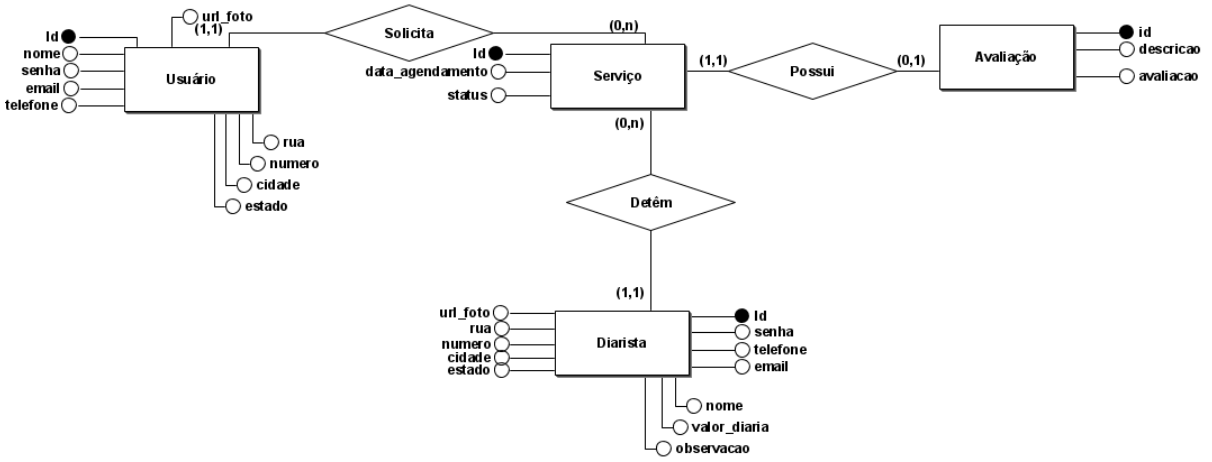


Figura 7. Diagrama Entidade-Relacionamento do projeto. Fonte: elaborado pelo autor.

5.2 Banco de Dados PostgreSQL

Para a persistência de dados na aplicação, foi utilizado o sistema gerenciador de banco de dados PostgreSQL. Este SGBD teve um recente aumento de sua popularidade por ter uma alta confiabilidade, bons recursos de consulta e integridade de dados (Carvalho, 2017).

5.2 Desenvolvimento da *API*

Após a finalização dos protótipos das interfaces e a modelagem do banco de dados, foi iniciado a codificação da API, uma aplicação que permite realizar uma integração entre dois sistemas para a realização de troca de dados e informações (Fernandes, 2018). A API foi desenvolvida utilizando-se a arquitetura REST (*Representational State Transfer*), que define um conjunto de princípios a serem utilizados em sua implementação (Tilkov, 2007). O aplicativo LimpaMais realiza chamadas à API por meio do protocolo HTTP (*HyperText*

Transfer Protocol). A tabela 1 mostra algumas das rotas que foram implementadas, assim como uma descrição do que essa requisição realiza.

Tabela 1. Rotas implementadas dentro da API.

URI	Método HTTP	Descrição
/services	POST	Criar um serviço
/services/{id}	GET	Listar as informações de um serviço
/services/{id}	PATCH	Atualiza as informações de um serviço
/services/{id}/ratings	POST	Adicionar uma avaliação ao serviço
/services/{id}/ratings	GET	Listar o comentário de um serviço

Para realizar as requisições que lidam com adição ou atualização de registros no banco de dados, o aplicativo LimpaMais faz as chamadas à API utilizando uma estrutura de dados em formato de texto chamado de JSON (*JavaScript Object Notation*) no corpo da sua requisição. A API então recebe essa estrutura e trata os dados através da lógica que foi implementada.

Para a implementação da *API*, foi utilizado a plataforma Node JS, uma ferramenta criada para desenvolver aplicações escaláveis utilizando Javascript no lado do servidor, juntamente com o *Express*, um *framework* que facilita o desenvolvimento de *APIs*. O servidor em que a aplicação está hospedada foi disponibilizado pelo *Heroku*, uma plataforma que permite aos desenvolvedores realizarem implantações e gerenciamento de suas aplicações sem a necessidade de configurar um servidor próprio para realizar a hospedagem.

5.3 Desenvolvimento do aplicativo

Após a finalização do desenvolvimento da *API*, foi dado início à implementação do aplicativo. Para sua construção, foi utilizado o Flutter, um *framework* criado pelo Google muito utilizado para desenvolvimento de aplicações móveis, de código aberto e que possui como linguagem de programação base o Dart para a construção dos aplicativos.

5.4 Funcionalidades da aplicação

Neste tópico, serão mostradas algumas das funcionalidades que foram desenvolvidas dentro do aplicativo

5.4.1. Página de Login

Ao inicializar o aplicativo, é possível visualizar a tela inicial para autenticação na aplicação, onde o usuário ou a diarista fornece seu e-mail e senha. Se os dados mencionados refletirem em algum usuário já registrado na aplicação, o aplicativo abre a tela principal em que mostra uma lista de diaristas previamente cadastrados. Na mesma tela, é possível clicar em cima do texto para criar uma nova conta para realizar cadastro no sistema. A Figura 8 representa a interface da tela de login.

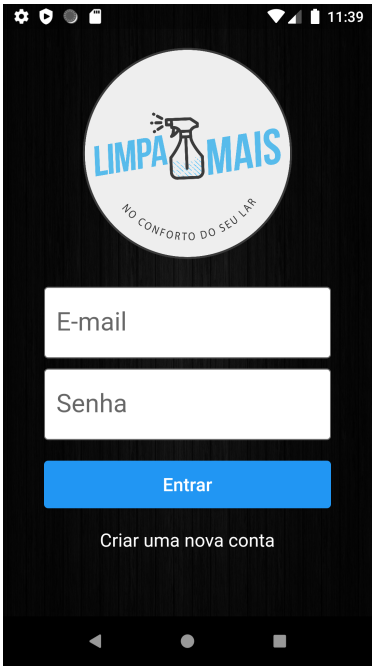


Figura 8. Autenticação do Usuário. Fonte: elaborado pelo autor.

5.4.2. Listagem de Diaristas

Quando o usuário realizar login no aplicativo, ele será redirecionado para a página principal que contém uma lista das diaristas que possuem um cadastro prévio dentro da base de dados da aplicação. Nessa tela, é possível o usuário filtrar as diaristas pela cidade em que estão localizadas, como mostra a Figura 9. Ao clicar em uma diarista específica, o aplicativo direciona o usuário para a tela de detalhes, onde mostra todas as informações da prestadora e seus serviços.

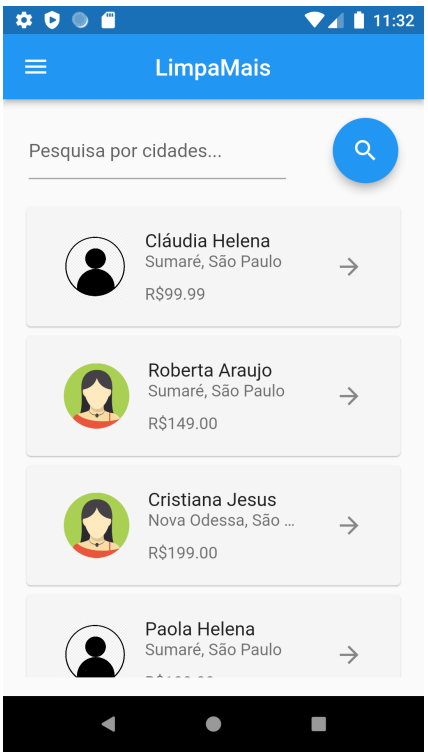


Figura 9. Interface com a lista de diaristas disponíveis. Fonte: elaborado pelo autor.

5.4.3. Detalhes da diarista

Quando o usuário selecionar a diarista desejada para realizar uma solicitação de agendamento de serviço, o aplicativo será redirecionado para a interface de detalhes dessa diarista, onde é mostrado as informações da diarista, o valor da sua diária e caso houver, as avaliações feitas pelos usuários dos últimos serviços que prestou. Logo após as informações, como mostra a Figura 10, há um botão para o usuário realizar uma solicitação de agendamento, onde o aplicativo irá redirecionar o usuário para uma página onde irá selecionar o dia desejado para o serviço.

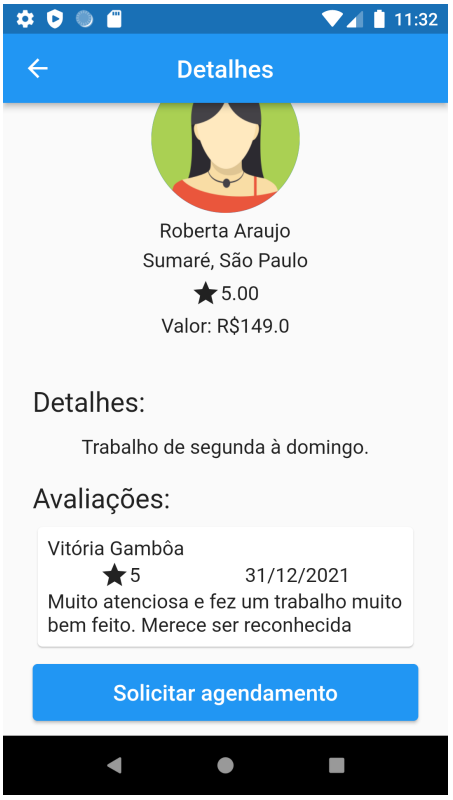


Figura 10. Interface de informações da diarista. Fonte: elaborado pelo autor.

5.4.3. Solicitações de Usuários

As diaristas ao realizarem autenticação no aplicativo irão ser redirecionadas para sua página inicial que contém todos os seus agendamentos, tanto os já agendados quanto os que ainda faltam sua confirmação ou rejeição, dependendo da escolha da diarista. Na lista de agendamentos a confirmar, a diarista pode selecionar o *cartão* desejado para abrir a tela de informações do usuário que solicitou o agendamento. A Figura 11 demonstra a interface principal de uma diarista.

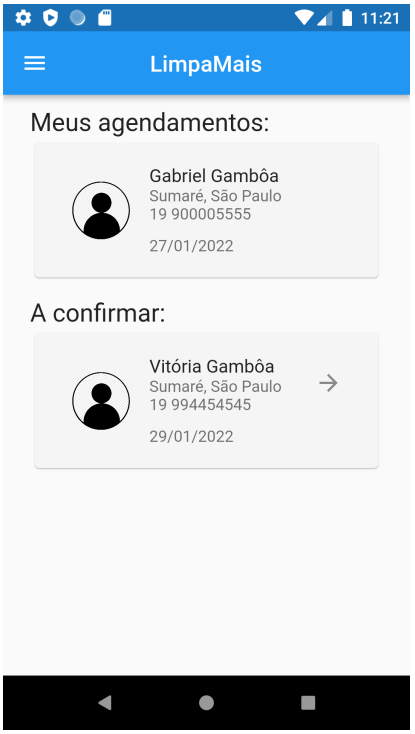


Figura 11. Interface inicial de uma Diarista. Fonte: elaborado pelo autor.

5.4.4. Informações da solicitação de serviço

Ao ser criado pelo usuário, a solicitação será mostrada pelo aplicativo na página inicial da diarista. Nessa página, é exibido todas as informações do usuário que realizou o pedido de limpeza, mostrando sua localização, a data de agendamento escolhida e seu número de telefone para eventual contato. Na tela, também é mostrado dois botões para a trabalhadora poder recusar ou aceitar o convite de agendamento. A Figura 12 mostra uma solicitação de um cliente para a diarista.

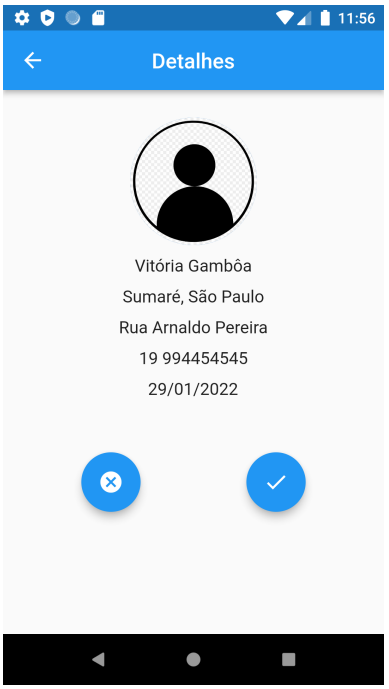


Figura 12. Interface de informações do agendamento. Fonte: elaborado pelo autor.

6. Conclusão

A fim de auxiliar todas as diaristas que foram afetadas pela epidemia do Coronavírus no Brasil e levando em conta o grande crescimento no uso de dispositivos móveis no país, no presente trabalho foi construído o aplicativo LimpaMais para servir como uma plataforma para conectar prestadores de serviço de limpeza com potenciais clientes. Com essa aplicação, busca-se motivar e auxiliar financeiramente as diaristas que sofreram as consequências causadas pelo surgimento e crescimento da Covid-19 no Brasil.

Com o desenvolvimento deste trabalho, foi possível aplicar diversos conhecimentos adquiridos em diversas disciplinas ao decorrer do Curso Superior de Análise e Desenvolvimento de Sistemas. Dentre elas, estão: arquitetura de software, análise orientada a objetos, engenharia de software e banco de dados I e II. Ademais, foram adquiridos diversos conhecimentos relacionados a construções de *APIs REST* e programação em multiplataforma para dispositivos móveis com a utilização do *framework* Flutter.

O aplicativo foi parcialmente desenvolvido, dando o foco principal às suas principais funcionalidades para atingir o objetivo da aplicação, como o agendamento do serviço de limpeza e a confirmação ou rejeição por parte da diarista.

Como trabalhos futuros, algumas funcionalidades que não foram implementadas no período de desenvolvimento, como por exemplo login com redes sociais, editar as informações do seu perfil e permitir que o usuário avalie um serviço prestado ainda podem ser desenvolvidas. A publicação do aplicativo nas lojas Appstore e Google Play também é uma sugestão para trabalho futuro, assim como uma coleta de feedback de potenciais clientes da aplicação sobre sua satisfação em relação ao aplicativo, através de um formulário do Google Forms, para a validação do aplicativo.

Referencias

Andrade, Ana. O que é Flutter ? TreinaWeb, 2020. Disponível em: https://www.treinaweb.com.br/blog/o-que-e-flutter. Acesso em: 28 jun. 2021.

Bohrer, Larissa. Trabalhadoras domésticas estão entre as mais afetadas pela pandemia. Disponível em: https://www.redebrasilatual.com.br/trabalho/2021/04/trabalhadoras-domesticas-estao-entre-as-mais-afetadas-pela-pandemia/. Acesso em: 11 nov. 2021.

Carelli, Rodrigo; Cavalcanti, Tiago; Fonseca, Vanessa. FUTURO DO TRABALHO: Os efeitos da Revolução Digital na sociedade. Brasília: ESMPU, 2020. Disponível em: http://35.238.111.86:8080/xmlui/bitstream/handle/123456789/411/Carelli_Rodrigo_Futuro%20do%20trabalho.pdf?sequence=1&isAllowed=y. Acesso em: 09 ago. 2021.

Carvalho, Vinicius. PostgreSQL: Banco de dados para aplicações web modernas. 1.ed. Casa do Código, 2017.

Costa, Simone. Pandemia e desemprego no Brasil. Revista de Administração Pública, Rio de Janeiro, jul./ago. 2020 Disponível em: https://www.scielo.br/j/rap/a/SGWCFyFzjzrDwgDJYKcdhNt/?lang=pt. Acesso em: 15 set. 2021.

Fernandes, André. O que é API? Entenda de uma maneira simples. Vertigo, 2018. Disponível em: https://vertigo.com.br/o-que-e-api-entenda-de-uma-maneira-simples/. Acesso em: 7 jul. 2021.

Goasduff, Laurence. Gartner Says Worldwide Smartphone Sales Grew 26% in First Quarter of 2021. Gartner, 2021. Disponível em: https://www.gartner.com/en/newsroom/press-releases/2021-06-07-1q21-smartphone-market-share. Acesso em: 27 set. 2021.

Google. 2021. Material Design. Disponível em: https://material.io/design. Acesso em: 19 jul. 2021.

Heuser, Carlos. Projeto de Banco de Dados. 4.ed. Porto Alegre: Instituto de informática da UFRGS, Sagra Luzzatto, 2001.

Instituto Locomotiva, 39% dos patrões dispensaram diaristas sem pagamento durante pandemia, aponta pesquisa. Disponível em: https://www.ilocomotiva.com.br/single-post/2020/04/23/g1-39-dos-patr%C3%B5es-dispensaram-diaristas-sem-pagamento-durante-pandemia-aponta-pesquisa. Acesso em: 02 jul. 2021.

International Labour Organization. (2020, 7 de abril). ILO Monitor: Covid-19 and the world of work. Second Edition. Updated estimates and analysis. Genebra, Switzerland:Autor. Recuperado de https://www.ilo.org/wcmsp5/groups/public/@dgreports/@dcomm/documents/briefingnote/wcms_740877.pdf.

Lisboa, Anna. GIG ECONOMY E AS (RE)CONFIGURAÇÕES DE TRABALHO. Revista Manus Iuris, Mossoró, abr. 2021. Disponível em: https://periodicos.ufersa.edu.br/index.php/rmi/article/view/10457/10649. Acesso em: 28 jul. 2021.

Longo, Hugo; Silva, Madalena. A Utilização de Histórias de Usuários no Levantamento de Requisitos Ágeis. Universidade Federal de Santa Catarina. p. 30. 2014.

Martins, Rafael. Aplicativo Híbrido x Nativo: Qual o melhor para o seu projeto?. Agence, 2020. Disponível em: https://www.agence.com.br/aplicativo-hibrido-x-nativo/. Acesso em: 29 jun. 2021.

Meirelles, Fernando. Panorama do Uso de TI no Brasil. Portal FGV, 2021. Disponível em: https://portal.fgv.br/en/node/22931. Acesso em: 30 jun. 2021.

Noleto, Cairo. Prototipagem: o que é, quais os tipos e dicas para montar o seu protótipo! TreinaWeb, 2020. Disponível em: https://blog.betrybe.com/tecnologia/prototipagem/. Acesso em: 10 ago. 2021.

Oitaven, Juliana; Carelli, Rodrigo; Casagrande, Cássio. Empresas de Transporte, plataformas digitais e a relação de emprego: um estudo do trabalho subordinado sob aplicativos. Brasília: Ministério Público do Trabalho, 2018. Disponível em: https://csb.org.br/wp-content/uploads/2019/01/CONAFRET_WEB-compressed.pdf

Rees, Jason. Flutter Vs React-Native Vs Xamarin. Dzone, 2020. Disponível em: https://dzone.com/articles/flutter-vs-react-native-vs-xamarin. Acesso em: 31 ago. 2021.

Rodrigues, André; Silveira, Fábio; Fuini, Mateus. YELLOW LIST: APLICATIVO ANDROID PARA LOCALIZAR PRESTADORES DE SERVIÇO UTILIZANDO GPS E INTEGRAÇÃO COM REDES SOCIAIS. TCC (Gestão da Tecnologia da Informação) - Faculdade de Tecnologia São Paulo. Itapira, p.37. 2019.

Santos, Carlos; Lima, Miguel. SERVICE ADVISOR: UM APLICATIVO MÓVEL PARA MEDIAÇÃO ENTRE SOLICITANTES E PRESTADORES DE SERVIÇOS. TCC (Sistemas de Informação) - Centro Universitário Municipal de Franca. Franca, p. 26. 2020.

Tilkov, Stefan. A Brief Introduction to REST, 2007. Disponível em: https://www.infoq.com/articles/rest-introduction/. Acesso em: 13 dez. 2021.

White, James. Going native (or not): Five questions to ask mobile applications developers. The Australiasian Medical Journal, 2013. Disponível em: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3575060/. Acesso em: 09 ago. 2021.