UNIVERSIDADE FEDERAL DE CAMPINA GRANDE
CENTRO DE ENGENHARIA ELÉTRICA E INFORMÁTICA
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

EDUARDO AFONSO NUNES DA SILVA

TERMOBAKE: APLICATIVO PARA DIGITALIZAR INDICADORES DE TEMPERATURAS DE FORNOS CONTÍNUOS NA INDÚSTRIA LOCAL DE BISCOITOS

CAMPINA GRANDE - PB

2023

EDUARDO AFONSO NUNES DA SILVA

TERMOBAKE: APLICATIVO PARA DIGITALIZAR INDICADORES DE TEMPERATURAS DE FORNOS CONTÍNUOS NA INDÚSTRIA LOCAL DE BISCOITOS

Trabalho de Conclusão Curso apresentado ao Curso Bacharelado em Ciência da Computação do Centro de Engenharia Elétrica e Informática da Universidade Federal de Campina Grande, como requisito parcial para obtenção do título de Bacharel em Ciência da Computação.

Orientador : Wilkerson de Lucena Andrade

CAMPINA GRANDE - PB

2023

EDUARDO AFONSO NUNES DA SILVA

TERMOBAKE: APLICATIVO PARA DIGITALIZAR INDICADORES DE TEMPERATURAS DE FORNOS CONTÍNUOS NA INDÚSTRIA LOCAL DE BISCOITOS

Trabalho de Conclusão Curso apresentado ao Curso Bacharelado em Ciência da Computação do Centro de Engenharia Elétrica e Informática da Universidade Federal de Campina Grande, como requisito parcial para obtenção do título de Bacharel em Ciência da Computação.

BANCA EXAMINADORA:

Wilkerson de Lucena Andrade

Orientador – UASC/CEEI/UFCG

Roberto Medeiros de Faria

Examinador – UASC/CEEI/UFCG

Melina Mongiovi Cunha Lima Sabino

Professor da Disciplina TCC – UASC/CEEI/UFCG

Trabalho aprovado em: 17 de Novembro de 2023.

CAMPINA GRANDE - PB

RESUMO

O controle de temperatura de fornos industriais para uma empresa de biscoitos é essencial para a qualidade do produto. Porém, nem toda empresa possui uma forma digital para gerenciar as informações coletadas durante a fabricação dos produtos e, por isso, seus registros são realizados em papéis e posteriormente repassado para planilhas digitais. Nesse sentido, software com objetivos parecidos estão escassos. Portanto, o propósito deste trabalho é entregar uma aplicação android a fim de modernizar a indústria local de biscoitos, a qual utiliza fornos contendo oito zonas, requerendo atenção para suas temperaturas. Com o uso do aplicativo, o processo de apontamento de temperaturas será agilizado e o fabricante poderá ter um acompanhamento controlado mais intuitivo. Além disso, não haverá a necessidade do uso de papel.

TERMOBAKE: APLICATIVO PARA DIGITALIZAR INDICADORES DE TEMPERATURAS DE FORNOS CONTÍNUOS NA INDÚSTRIA LOCAL DE BISCOITOS

ABSTRACT

Controlling the temperature of industrial ovens for a cookie industry is essential for the cookie quality. However, not every company has a digital way to manage the information collected during the manufacturing of the products, and as a result, this information is recorded on paper and later transferred to spreadsheets. In this regard, software with similar purposes is scarce. Therefore, the purpose of this project is to provide an Android application to modernize the local cookie industry, which has ovens with eight zones that require attention to their temperatures. With the use of this application, the temperature recording process will be streamlined, and the manufacturer will be able to have a more intuitive and controlled monitoring. Additionally, there will be no need for paper usage.

TermoBake: Aplicativo para digitalizar indicadores de temperaturas de fornos contínuos na indústria local de biscoitos

Eduardo Afonso Nunes da Silva
Universidade Federal de Campina Grande, Campina Grande, Paraíba, Brasil
eduardo.afonso.silva@ccc.ufcg.edu.br

Dr. Wilkerson de Lucena Andrade
Universidade Federal de Campina Grande Campina Grande, Paraíba, Brasil
wilkerson@computacao.ufcg.edu.br

RESUMO

O controle de temperatura de fornos industriais para uma empresa de biscoitos é essencial para a qualidade do produto. Porém, nem toda empresa possui uma forma digital para gerenciar as informações coletadas durante a fabricação dos produtos e, por isso, seus registros são realizados em papéis e posteriormente repassado para planilhas digitais. Nesse sentido, software com objetivos parecidos estão escassos. Portanto, o propósito deste trabalho é entregar uma aplicação Android a fim de modernizar a indústria local de biscoitos, a qual utiliza fornos contendo oito zonas, requerendo atenção para suas temperaturas. Com o uso do aplicativo, o processo de apontamento de temperaturas será agilizado e o fabricante poderá ter um acompanhamento mais intuitivo e controlado. Além disso, não haverá a necessidade do uso de papel.

Keywords

Indicadores de temperatura, Empresas locais de biscoito, Gerenciamento, Kotlin, *Firebase*, Android.

1. INTRODUÇÃO

A indústria de biscoitos é muito importante para a sociedade, pois eles são consumidos diariamente, seja no café da manhã, lanches ou sobremesas. Nesse contexto, segundo a ABIMAPI (Associação Brasileira das Indústrias de Massas Alimentícias e Pães & Bolos Industrializados)[1], o mercado de biscoitos valia R$29,20 bilhões em 2022, mostrando o seu valor para a economia.

O projeto em questão envolve a indústria local de biscoitos que utiliza forno contínuo para a sua produção. Estes fornos são divididos em diversas áreas denominadas zonas, em que cada uma delas possui tanto uma temperatura na área superior (teto), como uma inferior (lastro). Então, faz-se importante o controle preciso da temperatura de cada zona do forno para garantir a qualidade do produto.

Atualmente, na indústria local, o registro das temperaturas do forno é feito de maneira manual, usando papel, seguido pelo processo de digitalização desses dados em planilhas eletrônicas para análise posterior.

Apesar de existirem softwares patenteados para análise sensoriais de alimentos e sistema para ouvidoria colaborativa, a exemplo de um sistema para suporte na análise sensorial do café, acessível no banco de patentes do INPI, Instituto Nacional da Propriedade Intelectual, há a escassez de softwares patenteados com objetivo de tratamento de dados ou obtenção de dados de calor, fazendo-se necessário um software capaz de digitalizar os dados, especialmente para uma rede local, com menos acesso à tecnologia.

Portanto, com o intuito de otimizar o processo de coleta de dados de temperatura nas diferentes zonas do forno, um *software* foi desenvolvido. Agora, o processo, anteriormente realizado em duas etapas, pode ser realizado em apenas uma. Além disso, o software proporciona uma interface intuitiva para a visualização desses dados.

O trabalho está dividido em quatro seções. A primeira seção apresenta uma introdução e a contextualiza a necessidade do sistema. A segunda seção descreve a aplicação desenvolvida, com base nos requisitos coletados durante reuniões com um professor pesquisador de Engenharia de Alimentos da Universidade Federal de Campina Grande. A terceira, detalha o processo de desenvolvimento do sistema e os desafios enfrentados. Por fim, a última seção aborda uma avaliação do sistema e sugestões para decisões futuras que a aplicação pode seguir.

2. SOLUÇÃO

A solução proposta é um aplicativo (*TermoBake*) disponível para a plataforma Android, cujo principal objetivo é agilizar o processo de apontamento de temperaturas das zonas de um forno contínuo de assar biscoitos. Atualmente, é um procedimento realizado em duas etapas, a primeira, é coletar os dados em papel, para então colocá-los em planilha eletrônica.

2.1 Funcionalidades

As funcionalidades foram definidas a partir de reuniões com um dos professores pesquisadores da Universidade Federal de Campina Grande, Rennan Gusmão, cuja área de pesquisa envolve computação aplicada à Engenharia de Alimentos e tem contato direto com uma empresa local no segmento da

indústria de biscoitos. Com base nessas reuniões, foram determinadas as principais funcionalidades do sistema. Além disso, a paleta de cores e o *layout* foram baseados em *softwares* já desenvolvidos por outros alunos e pesquisadores em parceria com laboratórios da universidade.

2.1.1 Apontamento de temperaturas

Um usuário é capaz de realizar o registro das temperaturas de cada zona de um determinado forno. Cada zona possui uma temperatura teto no canto superior e uma temperatura lastro no canto inferior. Ambas possuem valores recomendados que podem ser fixos ou fazerem parte de uma faixa de variação. Um exemplo de um forno contínuo pode ser visto na Figura 1.

Figura 1. Exemplo de forno contínuo.

2.1.2 Listagem dos apontamentos realizados

Uma das formas de visualização dos apontamentos realizados é através de uma listagem, essa funcionalidade é dividida em duas telas. A primeira exibe a data dos apontamentos realizados, e, ao pressionar o apontamento específico, a tela de listagem de temperatura de cada zona daquele apontamento será exibido.

2.1.3 Visualização de gráfico dos apontamentos realizados filtrados por zona do forno e data

Além da exibição em listagem, o funcionário poderá visualizar os registros realizados através de um gráfico e controlar qual a zona e a data dos apontamentos.

2.1.4 Visualização de relatório dos dados

Deve ser possível visualizar um relatório contendo os apontamentos das temperaturas registradas pelo usuário dentro da própria aplicação e através de um arquivo em *PDF*.

2.2 Arquitetura

Esta seção apresenta a arquitetura da solução, bem como as interfaces desenvolvidas no sistema e, por fim, os testes unitários.

2.2.1 Front-end

O aplicativo foi desenvolvido nativamente utilizando a linguagem de programação Kotlin, a linguagem recomendada pelo Google para desenvolvimento Android. A Figura 2 apresenta a arquitetura *MVP (Model-View-Presenter)*, um dos padrões arquiteturais mais conhecidos no desenvolvimento focado em dispositivos móveis [2], utilizado para o desenvolvimento da aplicação. O *MVP* é uma variação do *MVC* (Model-View-Controller), onde o *controller* é substituído pelo *presenter*, esse último é responsável pelo processamento de dados no modelo e delegação de eventos na camada de visualização.

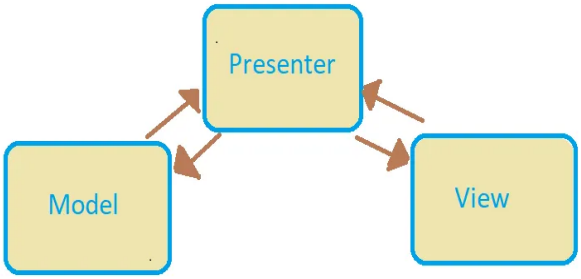

Figura 2. Diagrama da arquitetura MVP

2.2.2 Back-end

Optou-se por utilizar a plataforma *Firebase*, um *Baas* (Backend as a service)[3]. Um *Baas* é um tipo de serviço que fornece às aplicações web e mobile uma maneira de armazenamento em nuvem, notificações push e integração com outros serviços da rede. Os serviços utilizados para essa aplicação foram o de banco de dados *(Firestore Database)* e autenticação por credenciais. A Figura 3 representa o fluxo da comunicação entre uma aplicação e a plataforma do *Firebase*.

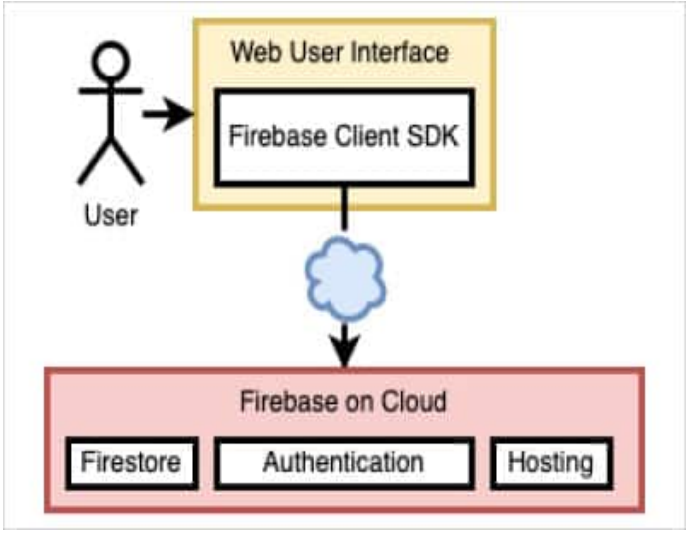

Figura 3. Diagrama de comunicação com *Firebase*

2.3 Interface do Sistema

A primeira tela com a qual o usuário interage é a de *login*, como mostrado na Figura 4, na qual se fez uso dos serviços de autenticação do *Firebase*[4] para garantir a segurança. Após estar logado no sistema, serão mostradas algumas opções de funcionalidades para o usuário.

Figura 4. Tela de login e tela inicial

Ao escolher a primeira opção, o usuário será direcionado para tela de registro de temperaturas, como apresentado na Figura 5. Nesta tela, ele deverá preencher o formulário com as temperaturas correspondentes a cada zona do forno, uma por vez. Para facilitar a navegação, o usuário pode alternar entre as zonas ao clicar nos botões em formato de seta ou deslizando o formulário para o lado.

Além disso, caso insira alguma temperatura fora da faixa recomendada, o aplicativo exibirá uma caixa de diálogo informando no momento que alterar a zona do forno. Adicionalmente, ao pressionar o botão de confirmação, haverá a opção de adicionar observações adicionais ao registro, oferecendo um maior nível de detalhamento e personalização, demonstrados na Figura 6.

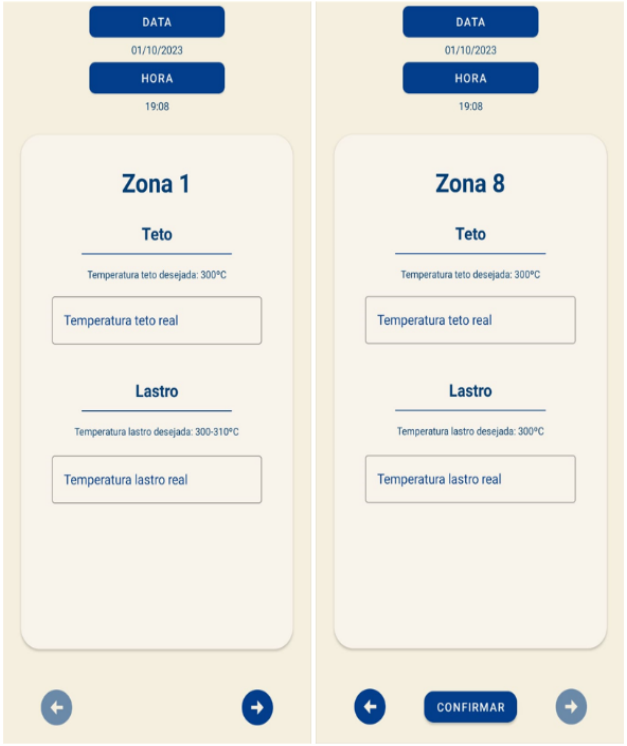

Figura 5. Tela de registro de temperaturas

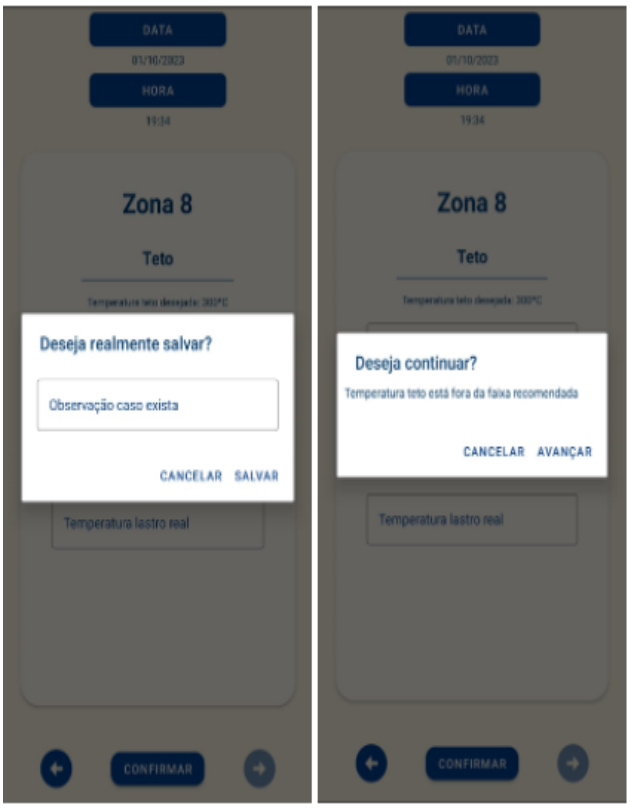

Figura 6. Diálogo de observação para o registro e informação de temperatura fora da faixa recomendada

Entre as demais opções, destacam-se a listagem de indicadores, a exibição do gráfico e a geração de relatórios.

A listagem é subdividida em duas telas, elas podem ser vistas na figura 7, a primeira é a listagem dos apontamentos com suas datas, juntamente à funcionalidade de exclusão através de um botão. Cada item nessa lista é selecionável, permitindo ao usuário visualizar as temperaturas associadas às zonas do forno.

Ao selecionar o registro, o usuário será direcionado para a segunda tela. Nessa página, na parte inferior, é exibida a observação digitada pelo usuário durante o registro. Adicionalmente, é possível realizar a edição das temperaturas de cada zona ao pressionar o botão específico.

Figura 7. Listagem de registros e de temperaturas com observação

Na terceira opção, tem-se a possibilidade de uma visualização mais abrangente, exibida na Figura 8, das temperaturas cadastradas por zona e data, apresentadas através de um gráfico. Neste gráfico, os pontos azuis representam a temperatura teto do forno em momentos específicos, enquanto os pontos em rosa refletem a temperatura lastro no mesmo período. Por conseguinte, os demais pontos no gráficos correspondem às temperaturas recomendadas tanto para o teto quando para o lastro da respectiva zona, proporcionando uma comparação visual das temperaturas registradas em relação às faixas ideais.

Por fim, tem-se a opção de geração de relatório, o usuário tem acesso a um relatório geral dentro da própria aplicação, como representado na Figura 9, ou no formato de *PDF*, como na figura 10. Nesta tela, o usuário escolhe a data e consegue ver a média das temperaturas registradas naquele período, a temperatura máxima e a mínima. Ademais, pode ver cada uma das temperaturas separadamente em um arquivo *PDF* gerado pela aplicação.

Figura 8. Gráfico de apontamento por zona e data.

Figura 9. Tela de relatório

Data e horário do registro: 28/10/2023 02:00:12

Zona	Temperatura teto	Temperatura teto recomendada	Temperatura lastro	Temperatura lastro recomendada
1	300,000ºC	300	310,000ºC	300-310
2	280,000ºC	300-320	290,000ºC	300
3	250,000ºC	300	280,000ºC	300
4	300,000ºC	300	299,000ºC	300
5	300,000ºC	300	310,000ºC	300
6	320,000ºC	300-340	330,000ºC	300
7	380,000ºC	300	390,000ºC	300
8	300,000ºC	300	300,000ºC	300

Data e horário do registro: 28/10/2023 01:42:12

Zona	Temperatura teto	Temperatura teto recomendada	Temperatura lastro	Temperatura lastro recomendada
1	300,000ºC	300	308,000ºC	300-310
2	0,000ºC	300-320	0,000ºC	300
3	0,000ºC	300	0,000ºC	300
4	0,000ºC	300	0,000ºC	300
5	0,000ºC	300	0,000ºC	300
6	0,000ºC	300-340	0,000ºC	300
7	0,000ºC	300	0,000ºC	300
8	0,000ºC	300	0,000ºC	300

Figura 10. Relatório em pdf gerado pela aplicação

2.4 Estrutura da aplicação

Na Figura 11, há a estruturação de pastas e arquivos para aplicação, proposta pela arquitetura *MVP*. A pasta *adapter* possui algumas classes associadas à listagem no Android, pois para usar os recursos de maneira otimizada, existem alguns contratos no qual o desenvolvedor precisa seguir. As classes são baseadas em um padrão de projeto arquitetural estrutural chamado *adapter*. Nos *services*, tem-se contratos para utilização dos serviços do *firebase*, e, por fim, há o pacote *util*, utilizado para classes utilitárias na aplicação

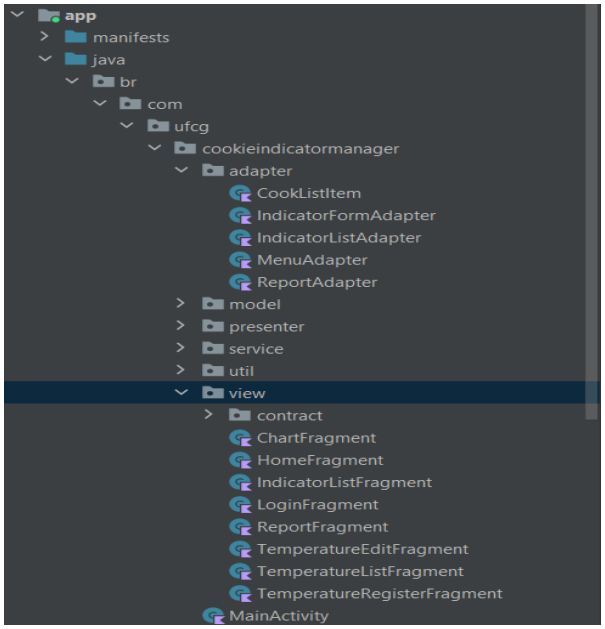

Figura 11. Estrutura de arquivos da aplicação android.

2.5 Estrutura de testes

A Figura 12 exibe a divisão dos arquivos e pastas dos testes. Nesta aplicação, não foi utilizada ferramenta de análise estática. No pacote "*mocks*", estão as classes que são utilizadas como dependência de outras classes. Os *mocks*[5] são utilizados para verificar se os passos do código funcionam corretamente, tirando o foco do funcionamento de dependências externas. Devido ao fato de determinadas funções receberem como parâmetros outras funções como *callback*[6], o uso de frameworks como o *Mockito*[7], por exemplo, torna-se um pouco complexo neste caso. Entretanto, o *mockito* foi utilizado para verificar as chamadas aos métodos da camada de visão pelos apresentadores correspondentes após processar os dados. Além disso, o padrão *MVP* se demonstrou um facilitador para os testes unitários, pois tornou possível os testes do comportamento da camada visão sem precisar de um framework interceptando, o que evitou trabalho de mockar componentes como caixas de diálogos do próprio Android.

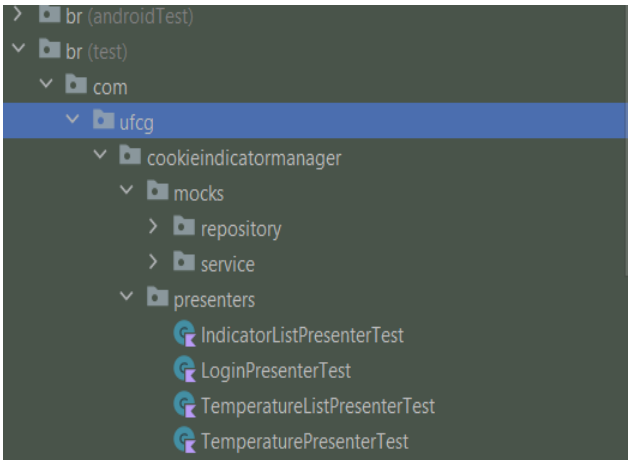

Figura 12. Estrutura de arquivos de testes unitários

3. DESENVOLVIMENTO

Esta seção apresenta como foi realizado o processo de desenvolvimento do sistema, desde suas etapas, como os desafios encontrados e aprendizados.

3.1 Etapas

O processo de desenvolvimento foi dividido em quatro etapas. A primeira etapa foi responsável pelo levantamento de requisitos, realizada através de reuniões presenciais com um professor pesquisador da UFCG e engenheiro de alimentos, o qual resultou em algumas histórias de usuário:

- Como usuário, desejo cadastrar apontamento de temperaturas teto e lastro para cada zona do forno, definindo a hora e a data de cadastro para que eu possa ter armazenado o processo de passamento do biscoito.
- Como usuário, desejo poder cadastrar uma observação para um apontamento de temperaturas para que seja possível informar algo sobre o forno em determinado momento.
- Como usuário, desejo poder excluir apontamentos a fim de remover dados inutilizados.

- Como usuário, desejo poder ver a lista de apontamentos cadastrados para que eu tenha acesso aos dados do forno.
- Como usuário, desejo ter acesso a um gráfico por zoneamento do forno em um determinado dia para analisar o funcionamento do forno,
- Como usuário, desejo efetuar login para ter meus dados acessíveis apenas para mim.
- Como usuário, desejo visualizar um relatório geral dentro da aplicação e ter acesso a um pdf gerado pela aplicação para poder visualizar detalhadamente os dados e tomar decisões.

A partir disso, as tecnologias foram definidas através de uma conversa com o cliente, pois ele já possui projetos em parceria com o *Software Practices Laboratory*, na UFCG, e tem contato com o desenvolvimento mobile. A linguagem escolhida para o desenvolvimento foi Kotlin, pois além de ser uma ferramenta moderna como Java, é a linguagem recomendada pelo Google para desenvolvimento nativo Android. Além disso, optou-se por usar os serviços de banco de dados e autenticação providos pelo *Firebase*. A decisão foi tomada com base na sua simplicidade de uso, seus planos custo-benefício e também por ter sido utilizado em outros projetos.

Então, antes de seguir para a implementação das atividades, foi utilizado o Kanban fornecido pelo Trello. O Trello é uma aplicação conhecida no que diz respeito a gerenciamento de projetos e permite acompanhar suas atividades em etapas. Conforme o software foi evoluindo, para manter um alinhamento entre as funcionalidades implementadas e o cliente, foram realizadas algumas entregas que ocorriam a cada duas semanas, como geralmente é feito no *framework* Scrum.

Por fim, para garantir o funcionamento da aplicação, foram realizados testes unitários utilizando Junit 4.1 e Mockito, bem como testes exploratórios ao longo de cada funcionalidade desenvolvida na aplicação.

3.2 Desafios e Aprendizados

Desenvolver uma aplicação mobile apresentou um desafio para mim, uma vez que toda minha experiência estava voltada ao desenvolvimento web, especialmente no back-end utilizando Java. Durante esse processo, percebi a necessidade de compreender conceitos que são exclusivos do desenvolvimento móvel e não são encontrados em sistemas web.

Entre esses conceitos, destaco a importância das “atividades” [8] e “fragmentos” [9] para modularizar o sistema. Atividades representam as diferentes telas do aplicativo, enquanto os fragmentos permitem a criação de interfaces flexíveis e reutilizáveis, dependendo de uma atividade para existir. A abordagem de fragmentos foi fundamental para garantir a organização da aplicação juntamente às facilidades de navegação trazidas pelo Android.

Além disso, foi necessário compreender o ciclo de vida [10] de atividades e seus fragmentos. Incluindo eventos de criação (*onCreate*), iniciação (*onStart*) e retorno da tela (*onResume*), por exemplo, entre outros. Conhecer esses ciclos foi essencial para otimizar o uso dos recursos do dispositivos, bem como proporcionar uma experiência de usuário mais suave, pois conhecer esses fundamentos auxiliou no controle de navegação entre as telas.

A adaptação ao desenvolvimento móvel também envolveu a exploração de novas arquiteturas de desenvolvimento, apesar de a mais difundida atualmente ser *MVVM*[11] *(model-view-view-model)*, não tive tempo o suficiente para me aprofundar nesta arquitetura, portanto, optei pela *MVP (model-view-presenter)*, por aparentar ser um pouco mais simples na questão de desenvolvimento.

Devido às restrições de tempo no projeto, recorri ao uso de uma biblioteca externa chamada MPAndroidChart[12] para a manipulação dos gráficos relacionados aos registros de temperaturas, isto proporcionou economia de tempo e permitiu que eu me concentrasse nos aspectos essenciais da aplicação.

Apesar de desafiadora, essa transição do desenvolvimento *web* para o *mobile* proporcionou uma oportunidade valiosa de aprendizado. O conhecimento prévio em desenvolvimento *web* contribuiu no entendimento das boas práticas independentemente da plataforma.

O curso como um todo contribuiu para o desenvolvimento do projeto, destaco as disciplinas que focaram em paradigmas de linguagens de programação, pois além de seu aspecto orientado a objetos, Kotlin possui características de linguagens funcionais. Adicionalmente, as disciplinas de Projeto de Software e Engenharia de Software agregaram no conhecimento de padrões de projetos arquiteturais comportamentais, estruturais e criacionais, fundamentais para basicamente quaisquer sistemas a serem desenvolvidos, assim como ajudaram no processo de gerenciamento do projeto fora do ambiente de desenvolvimento. Ademais, o conhecimento adquirido na disciplina de Programação Concorrente foi fundamental para entender os processos que ocorrem por trás de uma aplicação Android, importante para se ter uma aplicação fluida.

4. CONCLUSÃO E TRABALHOS FUTUROS

Além de o próprio professor pesquisador Rennan Gusmão utilizar o sistema e alunos do curso de Engenharia de Alimentos instruídos por ele, o aplicativo foi e continua sendo utilizado por funcionários da empresa, onde eles demonstraram interesse no aplicativo, bem como aprovaram o projeto.

Previamente, ainda não existia a funcionalidade de gerar o relatório dos registros em modelo de pdf, foi uma sugestão de funcionalidade após o uso da aplicação. Assim, essa foi a última funcionalidade implementada.

Como trabalhos futuros, espera-se que o *TermoApp* possa crescer e servir como base para outros projetos em outros segmentos da indústria de alimentos, como de massas em geral, e possa alcançar empresas de portes maiores. Ainda, espera-se que ele possa acompanhar a evolução tecnológica, como fazer uso de dispositivos embarcados, como sensores para a realização da coleta de temperaturas.

Além disso, o aplicativo foi submetido ao Instituto Nacional de Propriedade Industrial com a supervisão do professor

pesquisador Rennan Gusmão, e espera-se que ele cresça e sirva de base para outros projetos de outros segmentos na indústria de alimentos, como de massas, ou empresas de porte maiores.

REFERÊNCIAS

[1] Estatísticas de mercado ABIMAPI. Disponível em: ABIMAPI - Associação Brasileira das Indústrias de Biscoitos, Massas Alimentícias e Pães & Bolos Industrializados - Ublicacoes estatisticas

[2] Arquitetura MVP. Disponível em: https://acervolima.com/mvp-model-view-presenter-padrao-de-arquitetura-no-android-com-exemplo/

[3] Baas. Disponível em: https://telusdigital-marketplace-production.s3.amazonaws.com/iot/user-content/product/818d-o.pdf

[4] Autenticação com o firebase. Disponível em: https://firebase.google.com/docs/auth/android/firebaseui?hl=pt-br

[5] Mocks. Disponível em: https://www.telerik.com/products/mocking/unit-testing.aspx#:~:text=What%20is%20mocking%3F,or%20state%20of%20external%20dependencies.

[6] Callback function in Kotlin. Disponível em: https://www.baeldung.com/kotlin/callback-functions#:~:text=In%20Kotlin%2C%20a%20callback%20function,invoked%20at%20the%20appropriate%20time.

[7] Mockito. Disponível em: https://site.mockito.org/

[8] Atividades, Android. Disponível em: https://developer.android.com/guide/components/activities/intro-activities?hl=pt-br

[9] Fragmentos, Android. Disponível em: https://developer.android.com/guide/fragments?hl=pt-br

[10] Ciclos de uma aplicação Android. Disponível em: https://developer.android.com/guide/components/activities/activity-lifecycle?hl=pt-br

[11] MVVM Architecture. Disponível em: https://www.digitalocean.com/community/tutorials/android-mvvm-design-pattern

[12] MPAndroidChart. Disponível em: https://github.com/PhilJay/MPAndroidChart