UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO
DEPARTAMENTO DE AUTOMAÇÃO E SISTEMAS
CURSO DE GRADUAÇÃO EM ENGENHARIA DE CONTROLE E AUTOMAÇÃO

Gabriel Martins de Souza

Bloqueio expresso: uma abordagem de bloqueio elétrico com aplicações *Android*

Florianópolis
2024

Gabriel Martins de Souza

Bloqueio expresso: uma abordagem de bloqueio elétrico com aplicações *Android*

Relatório final da disciplina DAS5511 (Projeto de Fim de Curso) como Trabalho de Conclusão do Curso de Graduação em Engenharia de Controle e Automação da Universidade Federal de Santa Catarina em Florianópolis.
Orientador: Prof. Carlos Barros Montez, Dr.

Florianópolis
2024

Ficha catalográfica gerada por meio de sistema automatizado gerenciado pela BU/UFSC.
Dados inseridos pelo próprio autor.

Souza, Gabriel Martins de
Bloqueio Expresso : uma abordagem de bloqueio elétrico com aplicações Android / Gabriel Martins de Souza ; orientador, Carlos Barros Montez, 2024.
81 p.

Trabalho de Conclusão de Curso (graduação) - Universidade Federal de Santa Catarina, Centro Tecnológico, Graduação em Engenharia de Controle e Automação, Florianópolis, 2024.

Inclui referências.

1. Engenharia de Controle e Automação. 2. Bloqueio elétrico. 3. Aplicações Android. 4. Sistemas Embarcados. 5. Desenvolvimento JAVA. I. Montez, Carlos Barros. II. Universidade Federal de Santa Catarina. Graduação em Engenharia de Controle e Automação. III. Título.

Gabriel Martins de Souza

Bloqueio expresso: uma abordagem de bloqueio elétrico com aplicações *Android*

Esta monografia foi julgada no contexto da disciplina DAS5511 (Projeto de Fim de Curso) e aprovada em sua forma final pelo Curso de Graduação em Engenharia de Controle e Automação

Florianópolis, 11 de julho de 2024.

Prof. Marcelo de Lellis Costa de Oliveira, Dr.
Coordenador do Curso

Banca Examinadora:

Prof. Carlos Barros Montez, Dr.
Orientador
UFSC/CTC/DAS

Documento assinado digitalmente
Carlos Barros Montez
Data: 23/07/2024 16:27:33-0300
CPF: ***.035.027-**
Verifique as assinaturas em https://v.ufsc.br

Enrico Borgonovo Tridapalli, Eng.
Supervisor
8N1 Sistemas Embarcados LTDA

Documento assinado digitalmente
ENRICO BORGONOVO TRIDAPALLI
Data: 23/07/2024 15:31:54-0300
Verifique em https://validar.iti.gov.br

Prof. Leandro Buss Becker, Dr.
Avaliador
UFSC/CTC/DAS

Prof. Hector Bessa Silveira, Dr.
Presidente da Banca
UFSC/CTC/DAS

Dedico este trabalho a meus amados pais, como uma conclusão de todo o esforço e investimento empregados em meus estudos ao longo de todos esses anos, e a minha amada namorada, por todo o incentivo e a compreensão dados durante a execução deste projeto.

AGRADECIMENTOS

Agradeço, primeiramente, a Deus e à espiritualidade, por me conceder a disposição e saúde necessárias para a realização deste trabalho.

Agradeço ao meu chefe, o senhor Rodrigo Neri de Souza, pela oportunidade e confiança, além de todos as sugestões e os conhecimentos partilhados durante a execução deste projeto.

Agradeço ao meu supervisor, o senhor Enrico Borgonovo Tridapalli, por toda a ajuda concedida ao longo do desenvolvimento deste projeto.

Agradeço ao meu orientador, o senhor Carlos Barros Montez, pela sua disponibilidade, orientações e gentileza.

Agradeço a meus pais, minha família e a minha namorada por todo o suporte e incentivo dados, fundamentais para a elaboração e conclusão deste trabalho.

DECLARAÇÃO DE PUBLICIDADE

Florianópolis, 11 de julho de 2024.

Na condição de representante da 8N1 Sistemas Embarcados LTDA na qual o presente trabalho foi realizado, declaro não haver ressalvas quanto ao aspecto de sigilo ou propriedade intelectual sobre as informações contidas neste documento, que impeçam a sua publicação por parte da Universidade Federal de Santa Catarina (UFSC) para acesso pelo público em geral, incluindo a sua disponibilização *online* no Repositório Institucional da Biblioteca Universitária da UFSC. Além disso, declaro ciência de que o autor, na condição de estudante da UFSC, é obrigado a depositar este documento, por se tratar de um Trabalho de Conclusão de Curso, no referido Repositório Institucional, em atendimento à Resolução Normativa n° 126/2019/CUn.

Por estar de acordo com esses termos, subscrevo-me abaixo.

gov.br
Documento assinado digitalmente
ENRICO BORGONOVO TRIDAPALLI
Data: 17/07/2024 18:50:30-0300
Verifique em https://validar.iti.gov.br

Enrico Borgonovo Tridapalli
8N1 Sistemas Embarcados LTDA

RESUMO

Bloqueios elétricos consistem na desenergização de uma máquina e posterior bloqueio da sua fonte de alimentação. Eles fazem parte da realidade da maioria das empresas atualmente e conferem segurança aos processos de manutenção ou reparo de equipamentos elétricos. Esse procedimento é tipicamente realizado com o uso de um cadeado, cuja chave é armazenada dentro de um objeto denominado caixa de bloqueio, na qual os trabalhadores que operarão no equipamento bloqueado inserem cadeados individuais, garantindo que a chave não possa ser recuperada e a máquina reenergizada até que cada um desses trabalhadores retire seu cadeado. Apesar de funcionar adequadamente na maioria dos ramos industriais, na área da mineração, esse procedimento é um grande desafio, devido ao fato do local do bloqueio geralmente encontrar-se distante do equipamento bloqueado, obrigando que haja o transporte da chave de um ponto ao outro, adicionando um relevante tempo ocioso ao processo. Com o intuito de sanar esse problema, desenvolveu-se uma solução que utiliza, no local do bloqueio, uma espécie de armário inteligente controlado por uma IHM (Interface Humano-Máquina) com um sistema operacional *Android* embarcado. Essa IHM utiliza um aplicativo desenvolvido para gerenciar as portas do armário e armazenar adequadamente a chave do bloqueio. No local da manutenção, um aplicativo *Android* desenvolvido para dispositivos *mobile* é responsável por gerenciar a entrada e saída dos executores do serviço, gerando, quando a intervenção à máquina tem fim, um código de desbloqueio para abrir a porta do armário e retirar a chave. Todo esse processo ocorre *offline* e sem comunicação entre as aplicações do armário e *mobile*. A metodologia de desenvolvimento do projeto baseou-se em entender o processo e seus detalhes, a partir de conversas com o representante de uma grande mineradora brasileira, e mapear, então, os requisitos funcionais e não funcionais, com o intuito de modelar a solução a ser desenvolvida. Após ter esse modelo bem definido, empregaram-se iterações curtas e entregas contínuas, garantindo o foco na solução e constante alinhamento com as expectativas do cliente. Os resultados obtidos através da execução simultânea, em campo, do processo realizado da maneira tradicional e o proposto representaram um ganho temporal considerável, reduzindo os prejuízos financeiros oriundos da ociosidade causada pelo problema inicial.

Palavras-chave: Bloqueio elétrico. Aplicações *android*. Armário inteligente. Interface Homem-Máquina.

ABSTRACT

Electrical lockouts consist of de-energizing a machine and subsequently locking its power source. They are part of the reality of most companies today and provide safety for maintenance or repair processes of electrical equipment. This procedure is typically carried out using a padlock, the key to which is stored inside an object called a lockbox. In this box, workers who will operate on the locked-out equipment insert individual padlocks, ensuring that the key cannot be retrieved and the machine cannot be re-energized until each of these workers removes their padlock. Although this works adequately in most industrial sectors, it poses a significant challenge in the mining industry because the lockout location is often far from the locked-out equipment. This distance necessitates the transportation of the key from one point to another, adding considerable idle time to the process. To address this issue, a solution was developed that uses, at the lockout location, a type of smart cabinet controlled by an HMI (Human-Machine Interface) with an embedded Android operating system. This HMI uses an application developed to manage the cabinet doors and properly store the lockout key. At the maintenance site, an Android application developed for mobile devices is responsible for managing the entry and exit of service executors. When the intervention on the machine is complete, the application generates an unlock code to open the cabinet door and retrieve the key. This entire process occurs offline and without communication between the cabinet and mobile applications. The project development methodology was based on understanding the process and its details through discussions with a representative from a large brazilian mining company. This allowed us to map functional and non-functional requirements to model the solution to be developed. After having a well-defined model, short iterations and continuous deliveries were employed, ensuring focus on the solution and constant alignment with the client's expectations. The results obtained through the simultaneous execution, in the field, of the process carried out in the traditional manner and the proposed method represented an extremely considerable time gain, reducing financial losses stemming from the idle time caused by the initial problem.

Keywords: Electrical lockout. Android application. Smart locker. Human-Machine Interface

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

ADB	*Android Debug Bridge*
IDE	*Interface Development Environment*
IHM	Interface Homem-Máquina
JNI	*Java Native Interface*
NDK	*Native Development Kit*
NFC	*Near-Field Communication*
NR-10	Norma Regulamentadora 10
RFID	*Radio-frequency Identification*
RTC	*Real-Time Clock*
RTOS	*Real-Time Operation System*
SDK	*Software Development Kit*
XML	*Extensible Markup Language*

SUMÁRIO

1 INTRODUÇÃO

Bloqueios elétricos são procedimentos de segurança amplamente utilizados com o intuito de evitar acidentes envolvendo equipamentos elétricos durante a execução de manutenções, consertos ou qualquer outro tipo de intervenção no equipamento em questão. Esse processo tipicamente consiste na desenergização do equipamento, seguida de um bloqueio da fonte de alimentação elétrica, garantindo que durante a execução do serviço a ser realizado esse dispositivo não recebe energia elétrica.

Essa técnica é essencial para certificar que durante processos como manutenções ou reparos em equipamentos elétricos os trabalhadores envolvidos estejam em segurança contra choques ou surtos elétricos, bem como protege o próprio equipamento, que pode sofrer graves danos se energizado durante determinado procedimento de reparo ou manutenção.

A importância dessa operação pode ser atestada pela sua menção em algumas normas regulamentadoras (NRs), das quais se destaca a NR-10, que versa sobre a segurança em instalações e serviços em eletricidade.

Nesse contexto, uma grande mineradora brasileira realiza esse processo, segundo o diretor de uma de suas minas, mais de 300 vezes ao mês. Contudo, diferente da maioria das empresas, ela enfrenta um grande problema nesse procedimento: o local onde se encontra o equipamento e o local onde se encontra a central elétrica em que será realizado o bloqueio elétrico são consideravelmente distantes.

Esse fato aliado ao processo como deve ser realizado, que será abordado em capítulos subsequentes, gera um enorme desperdício, no quesito financeiro, que chega na casa dos milhões de reais por hora, devido ao tempo que uma máquina encontra-se inoperante.

Esse tempo inoperante divide-se em dois tipos. O primeiro é devido ao processo de manutenção ou reparo em si, ou seja, o tempo em que os executores do serviço em questão estão trabalhando no equipamento. Este tempo pode ser reduzido otimizando ou modificando as técnicas utilizadas em cada processo, o que não foi objeto de estudo e trabalho no projeto desenvolvido.

Já o segundo tipo de tempo inoperante é dado pelo período entre a desenergização do equipamento e o início da manutenção ou reparo. Portanto, é um tempo em que a máquina está desligada sem que nada relevante esteja acontecendo e justificando a sua ociosidade.

Apesar de parecer em um primeiro momento algo não tão grave, esse tipo de tempo inoperante pode chegar a horas a depender da distância entre o equipamento e a central elétrica e a disponibilidade de transporte entre esses dois ambientes, uma vez que, pela dimensão territorial de muitas minas, toda a locomoção entre os diversos espaços dentro dela é realizada por meio de veículos automotores.

Um outro problema ainda mais grave que acontece nessa mineradora e que pode elevar indefinidamente esse tempo ocioso do tipo 2 é a desatenção de algum dos executores do serviço de manutenção/reparo que, por algum motivo, esqueça de retirar o seu cadeado da caixa onde a chave usada para realizar o bloqueio encontra-se. Como já mencionado, o processo atualmente utilizado será melhor detalhado posteriormente. Por ora, a descrição anterior já é suficiente para entender a gravidade do acontecimento deste fato.

Nesses casos, os regulamentos internos da empresa determinam inúmeras questões burocráticas e de responsabilidade para o caso de, por exemplo, arrombar um desses cadeados, o que torna essa alternativa simples inviável.

Segundo o gerente de projetos, responsável pela proposição deste projeto, já ocorreu uma ocasião em que um trabalhador havia esquecido esse cadeado na caixa. Ao fim daquele processo de manutenção que havia durado alguns dias, ele já se encontrava em férias, em outro estado do país. Então, para que ele retornasse até a mina da maneira mais rápida possível, a empresa custeou passagem aérea e todo o translado apenas para que ele retirasse seu cadeado da caixa, liberando o desbloqueio energético do equipamento. Enquanto isso, a máquina em questão já encontrava-se em condições normais de operação, porém, estando parada, pura e simplesmente, por um descuido, gerando milhões e mais milhões de reais em prejuízo pela ociosidade.

A partir do que foi mencionado nos parágrafos anteriores, é possível entender a importância deste projeto para a empresa em questão. Ainda que este processo de bloqueio elétrico não seja problema para 99% das empresas que o realizam, para empresas do ramo de mineração, como no caso do cliente deste projeto, ele representa um grande gargalo.

A motivação deste projeto consiste na elaboração de uma solução que seja capaz de resolver simultaneamente os dois principais problemas apresentados anteriormente: a necessidade de locomoção entre a central elétrica e o local do equipamento a ser trabalhado e a obrigatoriedade da retirada presencial do cadeado de cada operador da manutenção. Essa solução deverá gerar um grande impacto positivo no quesito financeiro, uma vez que, ela garantirá que assim que a máquina for eletricamente bloqueada, a manutenção ou reparo já poderá iniciar, bem como permitirá que os responsáveis por essa atividade, poderão liberar o desbloqueio ainda que estejão do outro lado do mundo.

1.1 OBJETIVOS

Este projeto tem, como objetivo geral, a resolução da problemática mencionada anteriormente nos procedimentos de bloqueio elétrico realizados dentro da mineradora. Essa resolução baseia-se na elaboração de um novo produto e um consequente novo procedimento para realizar os bloqueios elétricos, em detrimento do produto e proce-

dimento atualmente empregado nessa empresa de extração, ainda que esses sejam amplamente utilizados sem grandes problemas na maioria das empresas.

Entre os objetivos específicos, destacam-se:

- a utilização do sistema operacional *Android*, uma vez que ele é uma das principais expertises da 8N1 Sistemas Embarcados LTDA, através da sua recém lançada plataforma de IHM;
- a integração dessa plataforma com a solução de *smart locker* de uma empresa parceira envolvida no projeto;
- o desenvolvimento de aplicações para *Android* que sejam capazes de substituir totalmente a solução atualmente utilizada;
- o desenvolvimento de interfaces intuitivas que facilitem o entendimento e utilização do sistema por parte dos usuários finais;
- a finalização, dentro do prazo coberto por este documento, de um protótipo funcional, com o intuito de apresentar ao cliente, bem como realizar testes em campo e, posteriormente sugerir melhorias que tornem esse protótipo mais próximo de um produto final.

1.2 METODOLOGIA

Inicialmente foi realizada uma reunião com o cliente, com o intuito de entender e delimitar tanto o problema quanto a solução a ser proposta. Nessa etapa, também foram levantados os principais requisitos do sistema, isto é, aquilo que ele deve conter e fazer e o que não. Como o problema engloba questões legais, também se fez necessário entender as normas por trás do processo, para obter respostas sobre possíveis limitações ou obrigações legais que o sistema devesse adequar-se. Caso isso não fosse observado, todo o projeto poderia tornar-se inválido, uma vez que infringiria normas. Após uma adequada definição de objetivos, requisitos e limitações do projeto, o desenvolvimento técnico foi iniciado. Como o desenvolvimento do projeto, com relação ao que está coberto por este documento, foi realizado majoritariamente por apenas uma pessoa (o próprio autor) sob a supervisão e orientação dos respectivos supervisor e orientador, a metodologia de desenvolvimento empregada foi a de *Agile Solo*, caracterizada pelas seguintes caraterísticas:

- iterações curtas: quebrar o trabalho em pequenas tarefas e ciclos de desenvolvimento curtos, o que permite ajustar e adaptar o projeto conforme necessário;
- entrega contínua: priorizar a entrega de incrementos funcionais em intervalos regulares, obtendo feedback mais rápido, para realizar pequenos ajustes conforme necessário;

- adaptabilidade: responder às mudanças nos requisitos ou no ambiente de desenvolvimento de forma rápida e eficaz;
- alinhamento com o supervisor e com o cliente: mesmo trabalhando sozinho, manter uma comunicação clara e regular com o supervisor e com o cliente, garantindo que o projeto esteja tomando a forma esperada;
- auto-organização e responsabilidade: planejar, executar e revisar o trabalho, garantindo que as tarefas sejam concluídas dentro do prazo e dos requisitos definidos;

A existência de uma metodologia é sempre extremamente importante para garantir o projeto seja desenvolvido de maneira organizada, uniforme e robusta. Com metas intermediárias e prazos bem definidos, podendo focar nos detalhes do desenvolvimento, mas sem perder a conexão com o todo. Ressalta-se também que, durante o desenvolvimento do projeto, surgiram problemas e dificuldades na elaboração de determinadas tarefas, como serão mostradas nos capítulos subsequentes. Esses, caso não fossem sanados de maneira autônoma, foram objeto de pesquisas nas respectivas bibliografias correlatas ao tema em questão, bem como alvo de consultas, tanto junto ao supervisor do projeto quanto junto ao orientador.

1.3 SOLUÇÃO PROPOSTA E RESULTADOS

Como será abordado nos próximos capítulos, atualmente, a mineradora usa uma caixa de bloqueio, na qual insere-se a chave que bloqueia a alimentação energética do equipamento. Essa caixa contém entrada para cadeados, para que outros funcionários possam bloquear a abertura dessa caixa, garantindo que a chave fique la dentro e cada trabalhador que atuará na máquina tenha a segurança de que esse dispositivo não poderá ser reenergizado.

Como já detalhado, essa abordagem traz inúmeros problemas para a empresa, sendo considerada por muitos dentro dela um importante ponto de melhoria, de modo a acabar com esse gargalo.

Com o intuito de superar esse obstáculo, a solução proposta envolve a utilização de uma espécie de armário inteligente, embarcado em sistema operacional *Android*. Nesse armário, uma aplicação desenvolvida para esse processo deve ser capaz de gerenciar o processo de armazenamento da chave responsável pelo bloqueio elétrico, garantindo a segurança dos executores da manutenção/reparo, o que substitui a caixa de bloqueio.

Além dessa aplicação, uma outra, também para *Android*, deve ser responsável por gerar um código capaz de desbloquear o armário inteligente e liberar o acesso a

chave que pode reverter o bloqueio elétrico. Esta outra aplicação substitui a necessidade de locomoção da central elétrica até o local onde o equipamento encontra-se.

Após a finalização de todo o desenvolvimento coberto por este documento, que será descrito ao longo dos próximos capítulos, o primeiro protótipo funcional foi concluido, como fora inicialmente planejado. Este protótipo foi apresentado ao representante da mineradora que propôs o projeto, além de ser levado a campo, em uma mina localizada no estado de Minas Gerais, e testada, com a presença do diretor da respectiva mina, bem como outros funcionários dela que possuem algum interesse no projeto.

Como esperado, alguns pontos de melhorias foram apontados, porém, de maneira geral, o protótipo cumpriu com as expectativas e validou a solução proposta como uma alternativa viável ao que é feito atualmente dentro das minas. O que significa um possível fim para os desperdícios corriqueiros durante os frequentes processos de bloqueio elétrico, bem como uma flexibilização e modernização desse procedimento, com relação a como ele é feito hoje.

1.4 RESPONSÁVEIS PELO PROJETO

A 8N1 Sistemas Embarcados (Figura 1) é uma empresa de sistemas embarcados que está situada em Florianópolis, mais especificamente no Parque Tecnológico Alfa, no bairro João Paulo. Em seu escritório, ela possui duas salas, uma focada *hardware*, responsável pela produção propriamente dita dos dispositivos embarcados confeccionados pela 8N1 e outra focada em *software*, responsável pelo projeto, desenvolvimento, testes, manutenção e programação desses dispositivos embarcados.

Como mencionado no próprio site da empresa, ela desenvolve produtos eletrônicos desde a concepção (idealização e/ou contextualização do problema, proposta de solução junto ao cliente, se houver), passando por todo o processo de prototipagem, estudo de requisitos e seleção e testes de componentes até a produção em massa para diversas organizações (8N1, 2023).

Ela é uma empresa de P&D terceirizada com mais de 10 anos de experiência no desenvolvimento de *hardwares*, *firmwares* e *softwares* em inúmeras áreas. Via de regra, ela produz sob demanda, a partir de projetos propostos pelos clientes que procuram a empresa e constroem os requisitos e desenvolvem tais projetos junto ao cliente. Em casos específicos, a 8N1 desenvolve o produto internamente para posteriormente oferecê-lo como solução a potenciais interessados.

Ainda segundo a própria 8N1: “Os Sistemas Embarcados e o desenvolvimento de Hardware são nossa especialidade. Ao longo dos anos, construímos uma variedade de dispositivos embarcados com C/C++, *Linux*, *Android* e diversos *Real-Time Operation System* (RTOS), ou sistemas operacionais de tempo real. Trabalhamos em todos os níveis do projeto: pesquisa, rascunho, especificação, prova de conceito, plane-

Figura 1 – Logo da 8N1 Sistemas Embarcados LTDA.

Fonte: 8N1 Sistemas Embarcados LTDA (2023).

jamento, gestão, desenvolvimento, prototipagem, testes, certificação, documentação, aquisição, treinamento e fabricação."

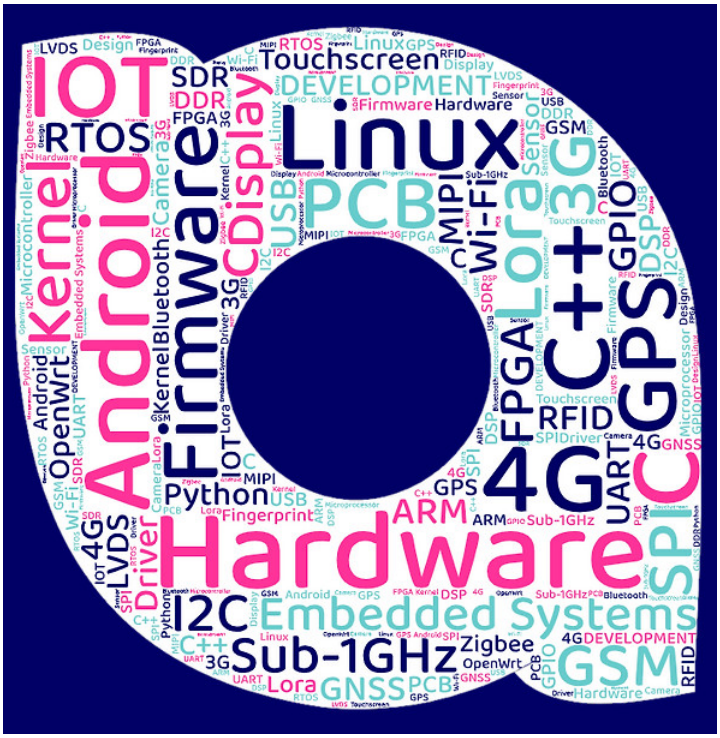
A 8N1 busca estar sempre em sintonia com aquilo que há de mais recente e eficiente no mundo da tecnologia nas mais diversas áreas. Na Figura 2, é possível ver uma imagem disponibilizadas por eles mesmos na qual pode-se observar diferentes tecnologias que já foram trabalhadas de alguma forma pela empresa em algum projeto já executado.

Entre os clientes, ou "parceiros", como são chamados pela companhia, estão algumas empresas como a Orsegups (empresa de segurança), InfoTV (rede de mídia), Nexti (empresa de auxílio de gestão de negócios), Meitech (fabricante de máquinas para indústria alimentícia, especializada na evisceração de aves), TAW (empresa que busca trazer novas tecnologias para ambientes educacionais), entre outras.

Esse projeto foi realizado a partir da cooperação com uma outra empresa parceira da 8N1 Sistemas Embarcados. Essa outra empresa tem como expertise o desenvolvimento de *smart lockers* (armário inteligentes), cada vez mais presentes em condomínios, edifícios comerciais e empresariais, academias, bibliotecas, entre outros ambientes. Essa empresa parceira forneceu um dispositivo portátil com a sua solução em armários inteligentes.

A partir esse dispositivo começou o trabalho por parte do autor desse documento, em nome da 8N1 Sistemas Embarcados. Em algumas questões pontuais, que serão mencionadas, a intervenção de outros desenvolvedores da 8N1 Sistemas Embarcados foi necessária.

Figura 2 – Tecnologias já utilizadas pela 8N1 Sistemas Embarcados LTDA.



Fonte: 8N1 Sistemas Embarcados LTDA (2023).

Do lado do cliente, também ressalta-se o importante papel que o representante da mineradora, que ocupa o cargo de gerente de projetos na empresa de mineração. Através de algumas reuniões, nas quais o desenvolvimento era apresentado a fim de alinhar expectativas, sugerir melhorias e alterações e fornecer valiosas informações sobre a empresa e seus processos, sobretudo no que se refere ao procedimento de bloqueio elétrico, contribuindo imensamente para o sucesso do primeiro protótipo funcional e posterior aprovação à continuidade do desenvolvimento do projeto para tornar-se um produto final e totalmente viável para a mineradora.

1.5 ESTRUTURAÇÃO

Todas as informações mencionadas anteriormente serão mais profundamente abordadas nos capítulos seguintes de acordo com a seguinte estruturação:

- contextualização da solução proposta: solução atualmente utilizada e seus problemas. Detalhamento do problema a ser tratado e sua importância. Outras soluções atualmente disponíveis no mercado e seus problemas. Solução proposta e suas justificativas;
- especificações: requisitos gerais, funcionais e não funcionais do projeto que ba-

lizarão o desenvolvimento do mesmo. Diagramas de casos de uso e diagramas de atividades, relevantes ao desenvolvimento e entendimento do projeto;

- descrição do projeto: apresentação da empresa desenvolvedora da solução. Implementação da solução proposta na prática. Hardwares utilizados. Softwares utilizados. Ferramentas de desenvolvimento empregadas, tais como *Interface Development Environment* (IDE), sistema operacional, plataformas, entre outros. Protocolos de comunicação e integração empregados. Problemas enfrentados durante o desenvolvimento e suas respectivas soluções;
- resultados: resultados obtidos e análises acerca deles. Adequação aos requisitos funcionais e não funcionais especificados. Teste realizado em campo comparando o desempenho da solução atual com a solução desenvolvida, principalmente com relação ao tempo empregado. Vantagens e desvantagens da solução proposta.

2 CONTEXTUALIZAÇÃO DA SOLUÇÃO PROPOSTA

Como já mencionado na introdução deste documento, uma grande mineradora brasileira propôs a 8N1 Sistemas Embarcados o desenvolvimento de um projeto que resolvesse o problema com que essa mineradora lida constantemente, sempre que realiza algum procedimento que demande a execução de um bloqueio elétrico, algo que ocorre mais de 300 vezes no mês, sendo essa informação de apenas uma das várias minas que essa empresa possui ou administra em território brasileiro.

Para melhorar o entendimento do problema e a sua natureza, faz-se necessária a explicação mais detalhada de como o processo de bloqueio elétrico ocorre hoje dentro da empresa, bem como quem são seus participantes e quais os pontos críticos desse procedimento. Na seção 2.1, será apresentada de maneira mais profunda essa questão.

2.1 DESCRIÇÃO DO PROCESSO DE BLOQUEIO ELÉTRICO ATUALMENTE FEITO

Atualmente, a empresa recorre ao procedimento e soluções dominantes do mercado, que são, na verdade, extremamente simples, ainda que eficientes para a maioria dos casos. Contudo, para o ramo de mineração essa eficiência não se comprova.

O procedimento ocorre sempre que uma manutenção, reparo ou qualquer outra intervenção a um equipamento elétrico da mina seja necessário. Para tal, é necessário desenergizar este equipamento e garantir o seu bloqueio elétrico, isto é, impedir que alguém seja capaz de reenergizar essa máquina sem que algumas medidas que garantam a segurança dos trabalhadores envolvidos diretamente no processo sejam tomadas.

Para tal, desliga-se a alimentação elétrica do equipamento desejado na central elétrica da mina. Além de desenergizar, é de extrema importância garantir a continuidade do corte de alimentação, então utiliza-se um cadeado na central elétrica que alimenta aquele equipamento de modo impedir o retorno da alavanca, botoeira, disjuntor ou qualquer outro mecanismo utilizado no sistema elétrico (dispositivos de comando) ao ponto de energização daquele circuito.

Segundo a Norma Regulamentadora 10 (NR-10), esse processo de bloqueio elétrico ainda necessita de uma sinalização apropriada para ser considerado concluído. Essa sinalização é realizada tradicionalmente com a utilização de uma etiqueta junto ao cadeado com dizeres de perigo e ordens para não intervir, conforme mostrado na Figura 3 (MTE, 2004).

Esse processo garante que, enquanto aquele cadeado estiver ali, o circuito ao qual ele alimenta não será, de maneira alguma, energizado. Mantendo assim a segurança dos trabalhadores que entrarão em contato direta com o equipamento, bem como a própria integridade do mesmo.

Figura 3 – Cadeado e etiqueta utilizados no bloqueio elétrico.

Fonte: TAGOUT® (2015).

Portanto, para garantir que esse processo não sofra violações, é necessário manter a chave daquele cadeado segura até o fim de toda a intervenção no dispositivo a ser trabalhado. É justamente nesse aspecto do processo que reside um dos principais problemas.

A mineradora em questão, com o intuito de garantir maior segurança e transparência nesse processo, estabelece que após a realização do bloqueio elétrico e sua devida sinalização, o eletricista que realizou esse procedimento deve entregar a chave desse cadeado a um supervisor da empresa. Este, por sua vez, coloca essa chave dentro de uma caixa e coloca um outro cadeado. Após isso, ele leva a sua chave, que concede acesso a caixa que contém a chave iniciadora do processo, até o local onde ocorrerá a manutenção ou reparo. Lá, esse supervisor apresenta a chave à equipe responsável pela execução da tarefa e a coloca dentro de uma caixa de bloqueio. Essa caixa possui inúmeros espaços destinados à inserção de cadeados (Figura 4), que serão colocados individualmente por cada um dos executores do serviço, o que garante que cada um deles tenha a certeza de que o seu próprio cadeado mantém inviolável a chave que pode reenergizar aquele equipamento e por em risco a sua própria vida.

Após finalizado a atividade a ser executada naquela máquina, cada um dos trabalhadores envolvidos retira o cadeado da caixa de bloqueio e após o último deles o fazer, significa que supostamente, nenhum outro operário tem alguma tarefa pendente naquela máquina e, portanto, ela pode ser reenergizada. Nesse momento então, aquele supervisor que trouxe a chave pode acessar novamente a caixa de bloqueio e retirar a chave dela. Após isso, ele a leva de volta até a central elétrica e abre a primeira caixa, dentro da qual está mantida a chave que desbloqueia o cadeado que evita a possibilidade de reenergizar o equipamento. Ele entrega então essa chave ao eletricista para que ele possa religar essa máquina para continuar a sua operação.

Figura 4 – Caixa de bloqueio utilizada no processo.

Fonte: TAGOUT® (2015).

Este processo apesar de simples, gera diversos problemas para a mineradora. Como mostrado na Figura 4, a solução atualmente utilizada é extremamente simples e totalmente mecânica. Além disso, também ressalta-se que o supervisor fica encarregado de registrar as informações principais de cada processo de bloqueio, como horários de início e conclusão, equipamento envolvido e indivíduos participantes. Esses registros são feitas manualmente em um livro de registros localizado na central elétrica. Posteriormente, esses dados são inseridos, também manualmente, em um *software* da empresa utilizado para armazenar e gerenciar os dados relacionados a diferentes processos da empresa de extração.

Após entender como funciona o processo de bloqueio elétrico dentro da mineradora, além da solução atualmente empregada, é possível compreender os principais problemas dessa abordagem, os quais são tratados na seção 2.2.

2.2 DETALHAMENTO DOS PROBLEMAS

A partir da descrição do processo de bloqueio elétrico é possível elencar quais as principais limitações que esse processo causa para a empresa de mineração em questão e quais as suas principais consequências.

O primeiro problema a ser relatado é o mais imediato de se notar: a necessidade de deslocamento entre a central elétrica e o local de manutenção do equipamento. Devido à grande extensão territorial de minas de extração, a locomoção dentro delas não é algo simples ou ágil, dependendo da utilização e disponibilidade de veículos automotores, o que nem sempre acontecerá de maneira rápida, uma vez que esses veículos são utilizados a todo instante para deslocamentos para os mais variados fins, o que adiciona uma componente extremamente variável ao tempo total ocioso associado a essa característica intrínseca dessa solução. Segundo o representante

da empresa, esse tempo de espera por um veículo automotor disponível pode chegar até a casa de uma hora, fora o próprio tempo do trajeto, além de considerar que esse percurso deve ser realizado tanto na ida da chave ao local da manutenção, quanto na volta à central elétrica, o que duplica o efeito desse aspecto no tempo ocioso do equipamento.

Como fora inicialmente explicado pelo representante da empresa que propôs o projeto, determinados equipamentos chegam a representar desperdícios de valores milionários por hora ociosa. Aliado a isso, considera-se o fato de que em um mês, mais de 300 bloqueios elétricos são realizados em média, é possível chegar na conclusão que vários milhões de reais são perdidos, não apenas pela necessidade de manutenções e reparos, o que já é um problema, apesar de inevitável, mas também pelo baixa eficiência da solução atualmente empregada para o processo de bloqueio elétrico.

Um outro problema de grande relevância dentro da empresa com relação a essa abordagem é o travamento físico e presencial que a caixa de bloqueio introduz no processo. Apesar de importante para garantir que cada trabalhador possa colocar o seu próprio cadeado e se evitar a reenergização do equipamento, também torna o processo completo frágil por ser vulnerável a possíveis esquecimentos do cadeado por parte de algum dos trabalhadores ou até mesmo uma intervenção maldosa de alguma pessoa poder colocar um cadeado nessa caixa e travar o processo, gerando perdas milionárias à empresa.

O segundo exemplo, apesar de possível e representar indubitavelmente uma vulneralibilidade da solução, ainda não possui histórico de ocorrência na empresa. Contudo, o primeiro exemplo foi relatado pelo representante da empresa de extração como algo que já aconteceu e causou um grande tormento para a mina.

Segundo o relato deste membro, houve um caso em que um desses trabalhadores esquecera o seu cadeado na caixa de bloqueio e deixara a mina. Ao fim da manutenção, alguns dias depois, o supervisor se deparou com um cadeado que ninguém havia retirado, impossibilitando a retomada da operação daquela máquina. Ao descobrirem a quem pertencia tal cadeado, aquele trabalhador foi contatado, porém, ele encontrava-se em férias, em outro estado do país.

Como arrombar um cadeado é um procedimento extremamente problemático dentro da empresa, gerando inúmeras investigações e demandando uma estrada de burocracias para concretizá-lo, além de todas as implicações de responsabilizações em consequência de possíveis fatos que possam se seguir a esse ato, evita-se ao máximo optar por esse caminho.

Por fim, a conclusão desse relato foi a empresa ter arcado com os custos de trazer esse indivíduo até a mina para realizar o desbloqueio de seu cadeado, o que, além de todo o transtorno e gastos com a locomoção desse trabalhador até a mina, gerou o prejuízo de a máquina ficar inoperante além do tempo necessário por uma

vulnerabilidade do processo a um simples descuido ou falta de atenção de um dos envolvidos.

Um outro problema, esse sendo um pouco menos grave, uma vez que não interfere diretamente no processo, logo não gera grandes desperdícios, é o registro manual das informações do processo de bloqueio. Ele requer um trabalho e organização adicional por parte do supervisor, além de tornar o registro de informações uma tarefa menos impessoal do que o ideal, afinal, um membro da empresa possui o poder de registrar o que aconteceu, quando e quem o fez, independente do que realmente tenha acontecido, portanto, demonstrando uma vulnerabilidade no ponto de vista de confiabilidade de informações.

Antes de propor uma nova solução alternativa à vigente, é imprescindível analisar e entender outras alternativas disponíveis no mercado. Na seção 2.3, a principal alternativa disponível no mercado será apresentada.

2.3 OUTRA SOLUÇÃO DISPONÍVEL NO MERCADO

Segundo o representante da mineradora, existem pouquíssimas alternativas às caixas de bloqueio, sendo a única relevante um produto da alemã **Bosch**. Esse produto, denominado *Smart Lockout* (Figura 5), é uma solução feita propriamente para o setor de mineração (BOSCH-REXROTH, 2024).

Figura 5 – *Smart Lockout*: solução da **Bosch** para bloqueios elétricos na mineração.

Fonte: **Bosch Rexroth Africa** (2024).

Esse equipamento foi desenvolvido justamente para enfrentar o problema da distância entre o ponto de bloqueio e o ponto de manutenção, como é mencionado no próprio *site* do produto. Lá também é possível encontrar a Figura 6, onde é feita

uma pequena comparação desse produto com a solução tradicional, além de seus principais benefícios.

Figura 6 – Comparação do *Smart Lockout* com a solução tradicional e seus benefícios.

Fonte: **Bosch** (2024).

Contudo, apesar de resolver parte dos problemas, esse representante classifica esse produto como inferior, do ponto de vista prático, às tradicionais caixas de bloqueio, já que, por ter apenas uma cavidade de bloqueio, permite apenas um bloqueio por vez, o que na para a realidade da mineradora, seria ineficaz, pois inúmeros bloqueios podem ocorrer simultaneamente. Além disso, ele também afirmou que, sob a ótica de usabilidade, esse produto não se destaca, demonstrando que a intuitividade é um requisito essencial para o sucesso do projeto.

Por fim, agora que esgotou-se a explicação acerca do processo de bloqueio, a solução atualmente utilizada e seus problemas, bem como outras soluções disponíveis no mercado, a seção 2.4, a última deste capítulo, versará sobre a solução a ser proposta para esse projeto.

2.4 SOLUÇÃO A SER PROPOSTA

A partir dos problemas relatados nas seções anteriores, a solução a ser proposta pode ser elaborada. Ainda que faltem detalhar os requisitos, que serão tema do capítulo 3, já é possível esboçar um caminho a seguir para contornar os aspectos negativos das soluções atual e alternativa (**Bosch**).

Como já mencionado, a solução conta com a participação de uma empresa parceira com expertise em armários inteligentes. A partir de um protótipo inicial dessa empresa, contendo toda a estrutura física e mecânica do produto a ser desenvolvido, bem como a tecnologia proprietária da empresa de portas automatizadas, o projeto foi desenvolvido.

Nesse protótipo inicial foi integrada a IHM da 8N1 Sistemas Embarcados, que utiliza sistema operacional *Android*. Nessa IHM, uma aplicação desenvolvida para essa solução é responsável por gerenciar o processo, bem como manipular as portas do produto, agindo de maneira análoga a um *smart locker*.

A ideia por trás dessa solução é substituir a caixa de bloqueio através desses armários inteligentes gerenciados pela aplicação *Android* que roda na IHM. A chave do bloqueio elétrica ficaria guardada dentro de uma das gavetas do armário, até que um código adequado fosse inserido para abrir a respectiva porta e liberar a chave para desbloqueio do sistema de comando, que impede a reenergização do equipamento que esteja sendo alvo de manutenção.

Enquanto isso ocorre no local do bloqueio, do outro lado, no ambiente em que se encontra a máquina a ser manutenida ou reparada, uma aplicação *mobile* seria utilizada para gerar esse código de desbloqueio necessário para abrir o armário. Tal código apenas poderá ser gerado no caso de nenhum trabalhador estar registrado como em serviço. Ao terminá-lo, esse trabalhador utiliza uma senha para informar ao sistema que a sua contribuição com a atividade está concluída.

Vale ressaltar que essas aplicações operam de maneira independente e sem comunicação entre si, uma vez que, como será visto adiante no capítulo 3, um dos principais requisitos desse projeto é o suporte à operação sem conexão à internet, já que essa é a realidade de algumas minas no Brasil, uma conexão instável ou até inexistente.

A comunicação entre os dois lados é realizada por radiofrequência, com soluções externas ao projeto, que são produtos já amplamente utilizados e que funcionam de maneira satisfatória. Através dessas tecnologias, os códigos necessários dentro do processo são transmitidos de um ponto ao outro. Esse método permite extinguir o problema da necessidade de locomoção entre um ponto ao outro.

O segundo principal problema: a necessidade de o executor da manutenção retirar o seu cadeado presencialmente também é eliminada do processo de bloqueio elétrico, uma vez que agora, o cadeado de outrora que mantinha a chave presa dentro

da caixa de bloqueio agora se tornou uma senha individual a ser inserida dentro do aplicativo *mobile* por parte de cada um dos trabalhadores envolvidos na manutenção. Devido a esse aspecto da nova solução, esse operário, caso esqueça de registrar o término de seu serviço, basta informar a senha ao supervisor para que esse possa tornar a participação desse operário como finalizada. Isso faz com que o trabalhador não precise mais retornar até a mina, podendo, por exemplo, declarar seu serviço como finalizado de qualquer lugar.

Como a entrada desses trabalhadores é inserida e supervisionada diretamente pelo supervisor do processo, a possibilidade de algum terceiro inserir um cadeado de maneira mal-intencionada e gerar um travamento permanente ao processo também é eliminada.

Por fim, as aplicações passam a ser capazes de registrar, de maneira automática, as informações relacionadas àquele processo de bloqueio elétrico. Além da agilidade no processo, a automatização e virtualização do registro desses dados, facilita uma futura integração de dados com algum outro sistema gerencial da empresa, já que esses dados não precisariam mais inseridas manualmente de um livro para um *software*.

2.5 CONSIDERAÇÕES

Neste capítulo foi apresentado em detalhes o processo a ser trabalhado e seus respectivos problemas, bem como a solução atualmente empregada e outra opção disponível no mercado com suas limitações no processo. Por fim, também é relatada a solução proposta, com seus principais diferenciais com relação as suas alternativas.

Este capítulo contém uma importante contextualização da problemática, necessária ao entendimento do desafio e idealização da solução, sem a qual, o projeto poderia perder o foco e direcionamento adequados, gerando, ao fim, um produto irrelevante ou incompleto.

No capítulo 3, são elencados os requisitos funcionais e não funcionais, bem como os diagramas necessários ao entendimento e desenvolvimento do projeto, tais como diagramas de casos de uso e diagrama de atividades.

3 ESPECIFICAÇÕES E REQUISITOS

Neste capítulo, são detalhadas as especificações sobre o projeto que foi desenvolvido, incluindo os requisitos funcionais e não funcionais, além de diagramas importantes à compreensão e ao desenvolvimento da solução, como diagramas de caso de uso.

As especificações são aspectos essenciais para garantir que todos os elementos críticos do projeto sejam levados em consideração e cumpridos, oferecendo uma visão clara e aprofundada do que é necessário à implementação bem-sucedida da solução. Elas fazem o papel de guia para os responsáveis pelo projeto, o que facilita o planejamento e a execução do mesmo, além de estabelecer critérios objetivos para a validação e aprovação do protótipo funcional entregue ao fim do período coberto por esse documento.

Primeiramente, na seção 3.1, serão apresentados os requisitos funcionais, que descrevem as funcionalidades e comportamentos esperados do sistema. Em seguida, na seção 3.2, os requisitos não funcionais abordarão as características e restrições necessárias ao sistema, como desempenho, segurança, limitações e usabilidade. Por fim, na seção 3.3, os diagramas de casos de uso e de atividades serão utilizados para indicar e ilustrar o funcionamento do sistema desenvolvido.

3.1 REQUISITOS FUNCIONAIS

Nesta seção, são abordados os requisitos funcionais que balizaram o desenvolvimento do projeto. Para cada um deles será fornecida uma explicação um pouco mais detalhada daquilo que se espera ou uma motivação para esse requisito. Ressalta-se que alguns requisitos se referem apenas à aplicação controladora do armário, que será denominada de aplicação do armário, outros se referem apenas à aplicação geradora do código de desbloqueio, que será denominada de aplicação *mobile*, enquanto outros se referem a ambas as aplicações.

No Quadro 1, são elencados os requisitos funcionais do sistema. Parte deles tem origem nas especificações do problema, enquanto outros derivam da solução proposta e das expertises e experiência da 8N1 Sistemas Embarcados.

A seguir, cada requisito funcional é detalhado:

- R.F. 01: o gerenciamento de usuários supervisores deve estar presente em ambos os sistemas. Esse gerenciamento equivale à implementação do *CRUD* (*create*, *read*, *update* e *delete*), o que consiste nas quatro funções essenciais em persistência de dados. Portanto, o sistema deve permitir tanto a criação e edição de usuários supervisores quanto a leitura e remoção dos mesmos. Esse requisito

Quadro 1 – Requisitos funcionais.

Requisito	**Nome**	**Prioridade**	**Aplicação**
R.F. 01	Gerenciamento de usuários supervisores	Alta	Ambas
R.F. 02	Configuração do modo quiosque	Alta	Armário
R.F. 03	Inicialização/reinicialização automática da aplicação	Alta	Armário
R.F. 04	Acesso às configurações do *Android*	Média	Armário
R.F. 05	Acesso ao *wi-fi* do *Android*	Média	Armário
R.F. 06	Suporte à autenticação *Radio-frequency Identification* (RFID)	Média	Ambas
R.F. 07	Gerenciamento das portas do armário	Alta	Armário
R.F. 08	Geração de código para bloqueio	Alta	Armário
R.F. 09	Armazenamento das informações de cada processo	Média	Ambas
R.F. 10	*Login* necessário a cada ação	Alta	Ambas
R.F. 11	*Logout* automático após o término de uma ação	Alta	Ambas
R.F. 12	*Logout* automático após ociosidade de 1 minuto	Alta	Ambas
R.F. 13	Acesso às informações do processo em andamento	Média	Ambas
R.F. 14	Sincronização entre a aplicação do armário e *mobile*	Alta	Ambas
R.F. 15	Informações de rede na página principal	Baixa	Armário
R.F. 16	Gerenciamento de usuários executores do serviço	Alta	*Mobile*
R.F. 17	Geração de código para confirmação e desbloqueio	Alta	*Mobile*
R.F. 18	Cadastro prévio dos executores de cada processo	Alta	*Mobile*
R.F. 19	Utilização de senhas para permitir entrada e saída dos executores do processo	Alta	*Mobile*

Fonte: AUTOR (2024).

é considerado essencial ao sistema, afinal, toda ação deve ser executada após uma autenticação.

- R.F. 02: a configuração do modo quiosque é um requisito que deve estar presente na aplicação do armário. Ela consiste no bloqueio do sistema operacional, de modo que o funcionamento do dispositivo fique focado apenas na aplicação do armário, impedindo que o usuário seja capaz de fechar a aplicação ou utilizar funcionalidades do sistema operacional *Android* sem a permissão prévia

da aplicação. Esse requisito é considerado como de alta prioridade, já que ele confere segurança ao sistema ao limitar o acesso do usuário à infinidade de funcionalidades do *Android*.

- R.F. 03: a inicialização/reinicialização automática da aplicação é um requisito que deve estar presente na aplicação do armário. Ela consiste em garantir que o foco do sistema operacional seja subir a aplicação a qual esse sistema deve-se dedicar exclusivamente, seja ao iniciar o sistema, seja ao detectar que a aplicação sofreu algum *crash*. Esse requisito é considerado como de alta prioridade, já que ele, juntamente com o requisito R.F. 02, ele torna-se a única maneira de subir a aplicação do armário.

- R.F. 04: o acesso às configurações do *Android* é um requisito que deve estar presente na aplicação do armário. Ele consiste em conceder ao usuário com credenciais de administrador acesso à tela de configurações padrão do sistema *Android*. Esse requisito é considerado como prioridade média, uma vez que ele não interfere diretamente na operação da aplicação e não impacta o funcionamento geral do sistema.

- R.F. 05: o acesso ao *wi-fi* do *Android* é um requisito que deve estar presente na aplicação do armário. Ele consiste em conceder ao usuário com credenciais de administrador acesso à configuração de *wi-fi* do sistema *Android*. Esse requisito é considerado como prioridade média, uma vez que a aplicação deve operar totalmente *offline* e, portanto, não interfere diretamente na operação da aplicação e não impacta o funcionamento geral do sistema.

- R.F. 06: o suporte à autenticação RFID é um requisito que deve estar presente em ambas aplicações. Ele é importante para agilizar o processo de autenticação, tornando melhor a usabilidade do sistema geral do sistema, já que a todo instante, nas aplicações é necessário realizar *login*. Esse requisito é considerado como de média prioridade, já que, apesar de melhorar a usabilidade do sistema, ele não é algo obrigatório para conseguir realizar de maneira completa os processos executados pelas aplicações.

- R.F. 07: o gerenciamento das portas do armário é um requisito que deve estar presente na aplicação do armário, afinal não há armários a se controlar na aplicação *mobile*. Ele consiste em tornar a aplicação do armário capaz de executar as duas operações suportadas pela solução de *smart locker* empregada no projeto, sendo essas a leitura do estado da porta e o comando de abertura dela. Esse requisito é considerado como de alta prioridade, já que ele constitui a tarefa essencial da aplicação, sem a qual, a solução torna-se totalmente inutilizável para realizar o processo de bloqueio elétrico.

- R.F. 08: a geração de código de bloqueio é um requisito que deve estar presente na aplicação do armário. Ela consiste em criar um código a partir de determinadas informações do processo em questão, utilizando métodos de criptografia, com o intuito de tornar esse código mais seguro e indecifrável. Esse requisito é considerado como prioridade alta, uma vez que, como o R.F. 07, ele constitui uma tarefa essencial ao funcionamento do sistema, além de conferir segurança ao processo.

- R.F. 09: o armazenamento das informações de cada processo é um requisito que deve estar presente em ambas as aplicações. Ele consiste em registrar as informações associadas a cada processo de bloqueio realizado com o sistema. Esse requisito é considerado como prioridade média, uma vez que não representa algo essencial ao funcionamento adequado da solução, porém, representa uma funcionalidade bastante interessante, que, como já mencionado no capítulo 2, automatiza e virtualiza os dados que atualmente são registrados de maneira manual.

- R.F. 10: a necessidade de realizar *login* a cada ação é um requisito que deve estar presente em ambas aplicações. Ela consiste solicitar a autenticação do usuário sempre que for requisitada a execução de alguma ação. Esse requisito é considerado como de alta prioridade, já que ele confere segurança ao sistema burocratizá-lo, impedindo que alguém sem as devidas credenciais realizem alguma ação que não deveria ser permitida.

- R.F. 11: o *logout* automático após o término de uma ação é um requisito que deve estar presente em ambas aplicações. Ele consiste em garantir que, após cada ação realizada no sistema, o usuário perca o seu *login*, limitando cada autenticação a apenas uma ação. Esse requisito é considerado como de alta prioridade, já que ele, juntamente com o requisito R.F. 10, obriga que o usuário tenha que inserir suas credenciais a cada uma das ações.

- R.F. 12: o *logout* automático após ociosidade de 1 minuto é um requisito que deve estar presente em ambas aplicações. Ele consiste em garantir que, após 1 minuto sem que haja alguma interação quando algum usuário tenha realizado *login*, esse sofra *logout*. Esse requisito é considerado como de alta prioridade, já que ele evita que algum usuário esqueça a aplicação logada, permitindo que alguém sem permissão realize alguma ação em seu nome.

- R.F. 13: o acesso às informações do processo em andamento é um requisito que deve estar presente em ambas aplicações. Ele consiste em exibir, em algum lugar da tela principal de cada uma das aplicações, as principais informações do processo de bloqueio que está em andamento. Esse requisito tem prioridade

média, devido ao fato de não representar uma funcionalidade essencial ao funcionamento do processo, configurando-se apenas como uma *feature* de apoio à execução do processo.

- R.F. 14: a sincronização entre a aplicação do armário e a aplicação *mobile* é um requisito que deve estar presente em ambas as aplicações. Ela consiste em implementar um método de sincronizar as duas aplicações antes de realizar qualquer processo de bloqueio elétrico. Essa sincronização realiza o papel de comunicação entre as aplicações, uma que vez, pela necessidade de operar sem conexão de rede, não é possível transmitir informação em tempo real. Esse requisito é considerado como de alta prioridade, pelo fato de ser fundamental para o funcionamento adequado da solução

- R.F. 15: a exibição de informações de rede na página principal é um requisito que deve estar presente na aplicação do armário. Ela consiste em mostrar, na pagina principal da aplicação, as informações principais da conexão de rede atualmente utilizada. Esse requisito é considerado como de baixa prioridade, afinal, além de pouca relevância para o sistema, ele tem um caráter exclusivamente informativo.

- R.F. 16: o gerenciamento de usuários executores de serviço deve estar presente na aplicação *mobile*. Esse gerenciamento equivale à implementação do *CRUD* para os usuários que realizarão a manutenção ou reparo. Portanto, o sistema deve permitir tanto a criação e edição de usuários executores do serviço quanto a leitura e remoção dos mesmos. Esse requisito é considerado essencial ao sistema, afinal, é necessário cadastrar esses trabalhadores para a solução ser capaz de operar adequadamente.

- R.F. 17: a geração de código de confirmação e desbloqueio é um requisito que deve estar presente na aplicação *mobile*. Ela consiste em criar um código para a confirmação do bloqueio e outro para o desbloqueio. As características desses códigos são análogos ao que fora descrito no R.F. 08. Esse requisito é considerado como prioridade alta, uma vez que ele constitui uma tarefa essencial ao funcionamento do sistema, além de conferir segurança ao processo.

- R.F. 18: o cadastro prévio dos executores de cada processo é um requisito que deve estar presente na aplicação *mobile*. Ele consiste permitir a um usuário supervisor selecionar, dentre os usuários executores de serviços já cadastrados, quais estarão envolvidos na manutenção a ser realizada. Esse requisito é considerado como prioridade alta, uma vez que ele constitui uma tarefa essencial ao funcionamento do sistema, sendo um requisito adicional ao R.F. 16.

- R.F. 19: a utilização de senhas para permitir entrada e saída dos executores do processo é um requisito que deve estar presente na aplicação *mobile*. Ela

consiste em permitir que os executores de serviço consigam registrar sua entrada ou sua saída da atividade a ser realizada através de uma senha. Esse requisito é considerado como de alta prioridade, já que ele é essencial ao funcionamento do processo, substituindo o papel dos cadeados na caixa de bloqueio.

Esses requisitos funcionais foram utilizados para direcionar o desenvolvimento das aplicações, tendo foco no atendimento a essas especificações desejadas. Na seção 3.2, serão apresentados os requisitos não funcionais, igualmente importantes no entendimento da solução ideal, bem como no seu desenvolvimento.

3.2 REQUISITOS NÃO FUNCIONAIS

Nesta seção, são abordados os requisitos não funcionais que nortearam o desenvolvimento do projeto. Assim como na seção 3.1, para cada um desses requisitos será fornecida uma explicação um pouco mais detalhada daquilo que se espera ou uma motivação para esse requisito.

Os requisitos não funcionais do sistema são elencados no Quadro 2. Parte deles tem origem nas especificações do problema, enquanto outros derivam da solução proposta e das expertises e experiência da 8N1 Sistemas Embarcados.

Quadro 2 – Requisitos não funcionais.

Requisito	**Nome**	**Prioridade**	**Aplicação**
R.NF. 01	Personalização do projeto	Alta	Ambas
R.NF. 02	Usabilidade	Alta	Ambas
R.NF. 03	Intuitividade	Alta	Ambas
R.NF. 04	Manutenibilidade	Média	Ambas
R.NF. 05	Operação *offline*	Alta	Ambas
R.NF. 06	Segurança no controle das portas	Alta	Armário
R.NF. 07	Segurança na geração e criptografia dos códigos	Alta	Ambas
R.NF. 08	Desenvolvimento em JAVA, para *Android*	Baixa	Ambas
R.NF. 09	Horário do sistema sempre correto	Alta	Armário

Fonte: AUTOR (2024).

A seguir, cada requisito não funcional é detalhado:

- R.NF. 01: a personalização do projeto é um requisito para ambas as aplicações. Essa personalização consiste em tornar esse primeiro protótipo funcional mais próximo a um produto final. Ela será realizada definindo cores, *designs* de elementos, logo, interfaces, tudo isso com o intuito de criar uma identidade visual para esse projeto. Apesar de majoritariamente visual, sem relevância funcional,

esse requisito é importante para conceder um caráter mais moderno e bem elaborado para a solução, ainda que essa esteja em um estágio embrionário, em relação ao que deve ser o produto final. Isso é especialmente importante pelo fato de a continuidade do projeto dentro da mineradora depender da aprovação de diretores, que por serem profissionais especializados nas questões administrativas da empresa, são leigos na área de desenvolvimento e, por essa razão, poderiam não compreender o momento atual do projeto e se oporiam à sua continuidade por julgá-lo mal desenvolvido pela falta de atenção às questões estéticas, afinal, é isso que gerará maior impacto aos olhos desses diretores, por ser a cara do produto. No lado do armário, a personalização deve ir além da aplicação, sendo realizada dentro do sistema operacional *Android* embarcado na IHM utilizada como interface gráfica do protótipo. Entre essas mudanças, estão a *bootlogo* e o *bootanimation* do *Android*. Esse requisito é considerado como de alta prioridade, uma vez que ele pode ser o diferencial que auxiliará o projeto a ter continuidade no seu desenvolvimento.

- R.NF. 02: a usabilidade é um requisito que deve estar presente em ambas aplicações. Ela consiste na facilidade, na simplicidade e na dinamicidade da utilização da aplicação. É importante também ressaltar que a prevenção a erros causados por mau utilização das suas funcionalidades por parte do usuário também é uma característica essencial para garantir uma boa experiência por parte dos diferentes tipos de usuários que utilizarem as aplicações. Esse requisito é considerado como de alta prioridade, já que a sua ausência pode ser, como fora para a outra solução já existente no mercado, uma barreira que inviabilize a sua continuidade e seu potencial sucesso.

- R.NF. 03: a intuitividade é um requisito que deve estar presente em ambas as aplicações. Ela consiste em garantir que que a aplicação seja extremamente simples de se entender e utilizar, permitindo facilmente um aprendizado rápido e efetivo por parte de seus usuários, sem que haja um grande número de instruções ou mesmo a necessidade de um treinamento complexo. Esse requisito é considerado como de alta prioridade, já que ele, juntamente com o requisito R.NF. 02, pode ser o aspecto que torne o projeto inviável ou bem sucedido. Ressalta-se aqui que alguns dos usuários que utilizarão essas aplicações podem ser indivíduos com baixa instrução e/ou familiaridade com novas tecnologias que, por essa razão podem sofrer com soluções pouco intuitivas.

- R.NF. 04: a manutenibilidade é um requisito que deve estar presente em ambas as aplicações. Ela consiste em estruturar, a nível de código, as aplicações de maneira que essas possam ser facilmente modificadas. Essa é uma boa prática de programação que preza pela utilização de trechos reaproveitáveis de código, es-

crita simples e intuitiva, legibilidade, modularidade e flexibilidade. Esse requisito é considerado como prioridade média, uma vez que ele não interfere diretamente na operação das aplicações. Contudo, ele é importante ao se considerar que esse projeto ainda será desenvolvido além do coberto por este documento, possivelmente por outros desenvolvedores que não o próprio autor.

- R.NF. 05: a operação *offline* deve estar presente em ambos os sistemas. Ela consiste em garantir que a solução opere de maneira adequada e completo independente de disponibilidade de conexão à rede. Esse requisito é considerado essencial ao sistema, afinal, como já fora mencionado anteriormente, não é possível garantir a estabilidade ou até mesmo disponibilidade de qualquer tipo de conexão de rede em todas as minas possuídas por essa empresa, o que tornaria esse projeto restrito, o que não era o objetivo inicial da empresa.

- R.NF. 06: a segurança no controle das portas é um requisito que deve estar presente na aplicação do armário. Ela consiste em garantir a abertura das portas do armário apenas nos momentos adequados, não permitindo nenhuma brecha, ou maneira alternativa de comandar tais portas, pois isso seria enquadrado como uma falha gravíssima de segurança. Esse requisito é considerado como de alta prioridade, por ser obrigatório ao sistema, uma vez que o seu não atendimento tornaria a solução inviável, afinal, seria equivalente a existir uma maneira de retirar a chave do bloqueio de dentro da sua caixa sem garantir que nenhum trabalhador esteja envolvido momentaneamente naquele processo de manutenção, o que pode levar a danos irreparáveis tanto ao equipamento quanto a esses indivíduos, incluindo morte, gerando grandes problemas para a empresa.

- R.NF. 07: a segurança na geração e criptografia dos códigos é um requisito que deve estar presente em ambas aplicações, já que ambas geram códigos. Ela consiste em garantir que o código seja indecifrável, sendo obtido apenas a partir das devidas aplicações, impossibilitando fraudes que sejam capazes de realizar etapas do processo que não devessem ocorrer naquele momento. Esse requisito é considerado como de alta prioridade, já o seu não cumprimento poderia acarretar em aberturas inapropriadas da porta, colocando equipamento e vidas em risco, gerando grandes prejuízos para a empresa, configurando uma grande falha de segurança do sistema.

- R.NF. 08: o desenvolvimento em linguagem JAVA, para *Android* é um requisito que deve estar em ambas aplicações. Ele consiste em utilizar as ferramentas de desenvolvimento *Android Software Development Kit* (SDK), utilizando linguagem JAVA para o desenvolvimento das aplicações. Esse requisito é considerado como prioridade baixa, uma vez que a linguagem a ser utilizada não é preponderante

para o sucesso do projeto. Contudo, como todas as soluções implementadas até hoje para essa IHM utilizaram JAVA, incluindo algumas realizadas pelo próprio autor, a escolha dessa linguagem de programação em detrimento de outras como o *Kotlin*, por exemplo, representa um ganho de tempo em desenvolvimento, já que muitas funcionalidades já estariam implementadas ou parcialmente indicadas na documentação da IHM. Além disso, a familiaridade do autor com tal linguagem torna a produtividade e qualidade da solução a ser desenvolvida consideravelmente maior.

- R.NF. 09: o horário do sistema estar sempre correto é um requisito que deve estar presente na aplicação do armário. Ele consiste em garantir que, mesmo quando a IHM onde a aplicação do armário está sendo executada não possua acesso à rede, o horário do sistema operacional esteja correto. Esse requisito é considerado essencial ao sistema, pois esse aspecto faz parte da solução pensada para contornar a independência de conexão de rede para realizar a comunicação entre aplicações, sem o qual, a solução desenvolvida torna-se inoperante.

Esses requisitos não funcionais, assim como os funcionais, foram utilizados para direcionar o desenvolvimento das aplicações, tendo foco no atendimento a essas especificações desejadas. Na seção 3.3, com o intuito de prosseguir no detalhamento das especificações do projeto, serão apresentados diferentes diagramas elaborados para documentar os procedimentos, melhorar o entendimento do processo e suas nuânces e auxiliar no desenvolvimento da solução.

3.3 DIAGRAMAS

Nesta seção são apresentados alguns diagramas UML que descrevem os comportamentos esperados na solução desenvolvida. Primeiramente, na seção 3.3.1, serão exibidos os diagramas de casos de uso. Depois, na seção 3.3.2, os diagramas de atividades associados a esses casos de uso serão mostrados.

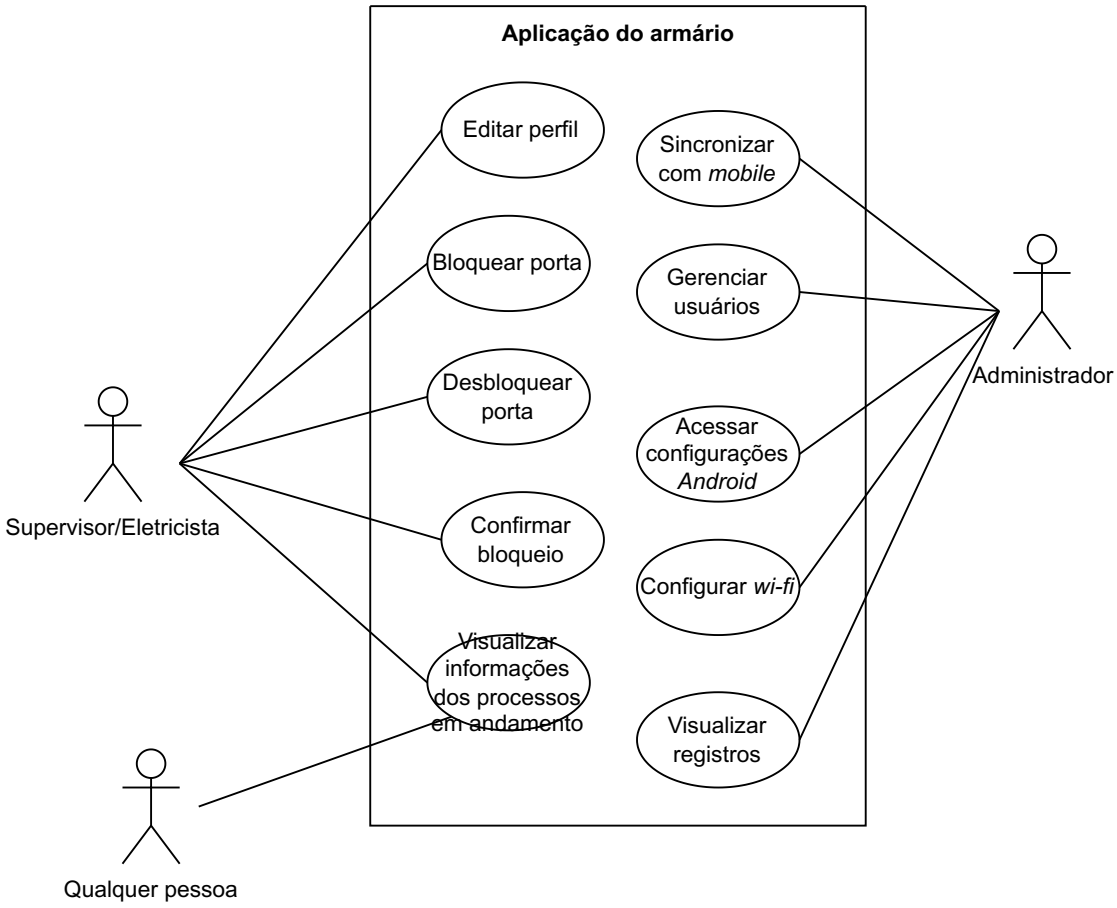
3.3.1 Casos de uso

Foram elaborados dois diagramas de casos de uso para auxiliar no entendimento das funcionalidades presentes na solução. Um deles se refere à aplicação do armário (Figura 7), enquanto o outro está correlacionado à aplicação *mobile* (Figura 8).

O diagrama de casos de uso da Figura 7 mostra a existência de três diferentes usuários: o administrador, o eletricista que em alguns casos pode ser substituído pelo

supervisor do processo, e, por fim, qualquer pessoa que não esteja cadastrada no sistema.

Figura 7 – Casos de uso da aplicação do armário.



Fonte: AUTOR (2024).

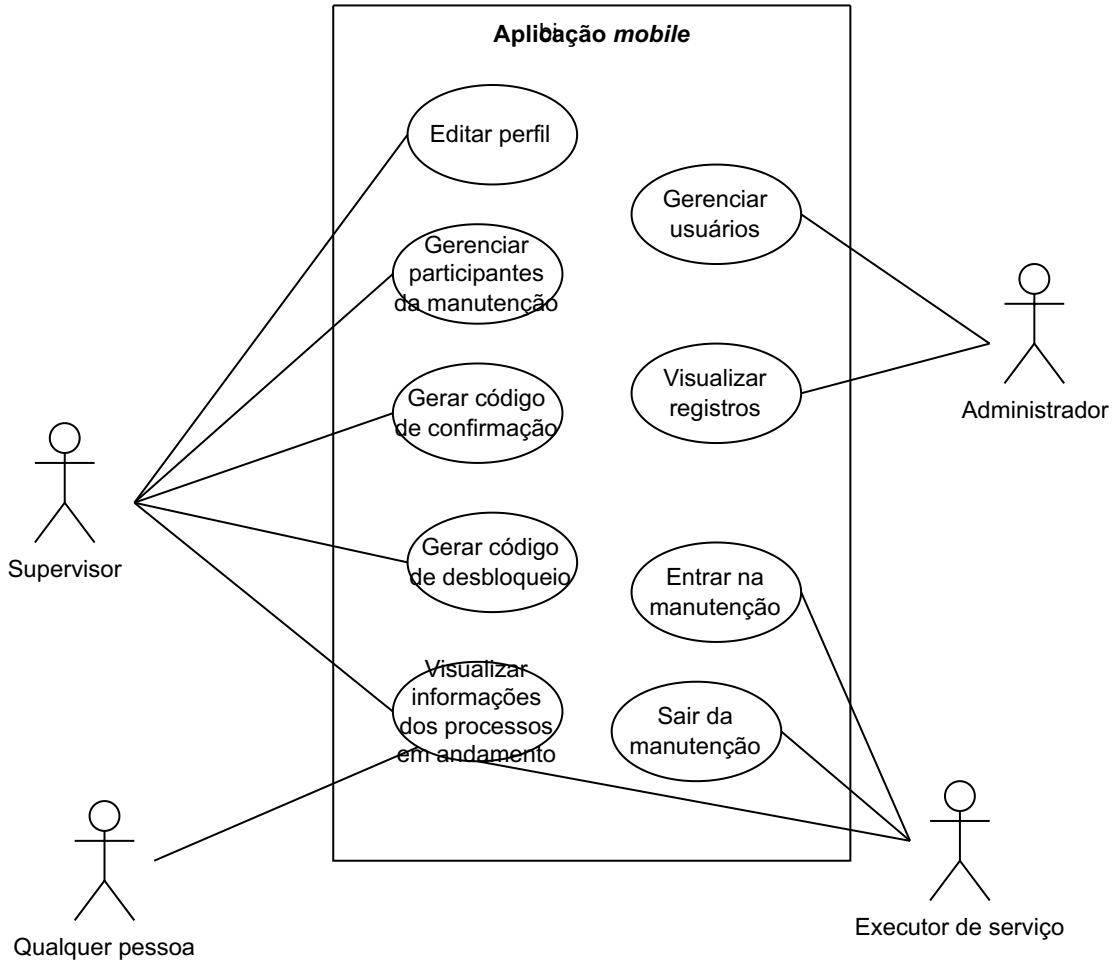
O primeiro é capaz de acessar a tela de sincronização com a aplicação *mobile*, gerenciar usuários (realizando as operações de CRUD), configurar o *wi-fi* do dispositivo, acessar as configurações do sistema operacional *Android* e ainda visualizar os registros dos processos concluídos.

O segundo é capaz de editar seu próprio cadastro, alterando nome e senha, além de realizar o bloqueio e desbloqueio da porta, mediante o código correto de desbloqueio do processo. Além disso, ele é capaz de realizar a confirmação do bloqueio, também mediante um código de confirmação a ser gerada pela aplicação *mobile*.

Por fim, tanto ele quanto qualquer outro usuário são capazes de visualizar as principais informações dos processos de bloqueio que estão em andamento naquele momento.

Já o diagrama de casos de uso da Figura 8 mostra a existência de quatro usuários: o administrador, o supervisor do processo, o executor do serviço de manutenção ou reparo e, por fim, qualquer pessoa que não esteja cadastrada no sistema.

Figura 8 – Casos de uso da aplicação *mobile*.



Fonte: AUTOR (2024).

O primeiro é capaz de gerenciar usuários (realizando as operações de CRUD) tanto supervisores quanto executores de serviço e visualizar os registros dos processos concluídos.

O segundo é capaz de editar seu próprio cadastro, alterando nome e senha, gerenciar os executores participantes de uma manutenção, podendo adicionar ou remover esses usuários desse processo, desde que o administrador os tenha cadastrado previamente. Ele também é capaz de gerar o código de confirmação e o código de desbloqueio.

O terceiro é capaz de entrar e sair do processo de manutenção, mediante a

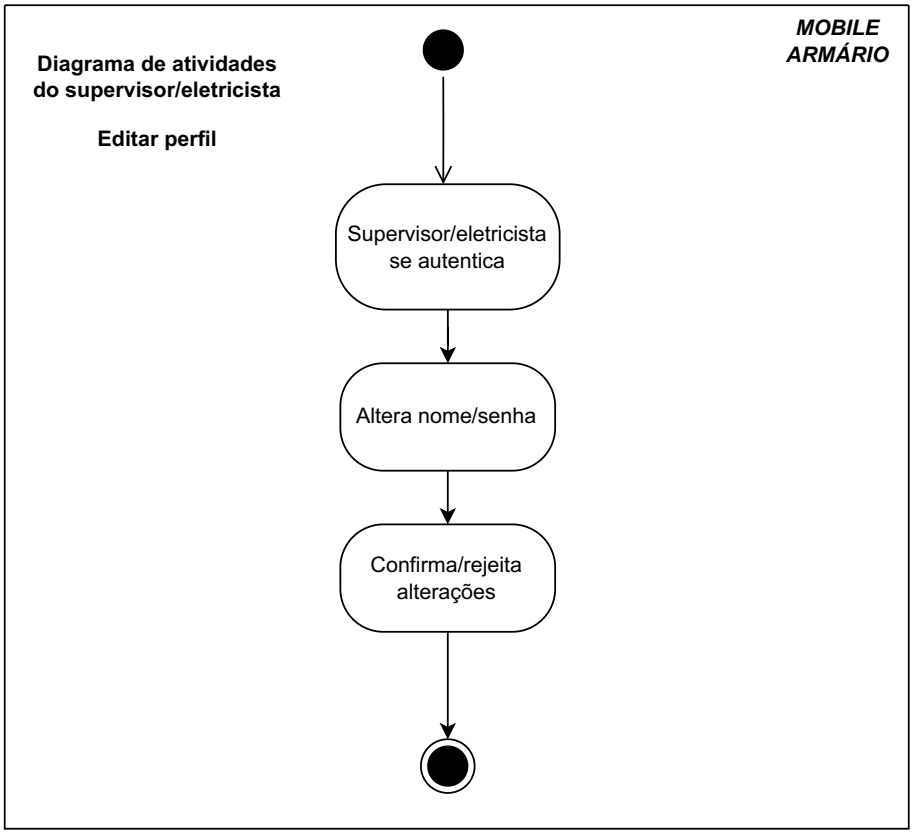
inserção da sua senha. Assim como na aplicação do armário, todos os usuários podem visualizar as informações do processo em andamento.

Com essa exibição das funcionalidades das aplicações através desses diagramas de casos de uso concluída, faz-se necessária a apresentação dos diagramas de atividades referentes a esses casos de uso. Esses diagramas serão assunto da seção 3.3.2.

3.3.2 Diagrama de atividades

Nesta seção são apresentados os diagramas de atividades referentes aos casos de uso representados pelos diagramas das Figuras 7 e 8, na seção 3.3.1. Na Figura 9, é possível observar o diagrama de atividades da edição de perfil de usuário, existente em ambas aplicações, para o supervisor ou eletricista, no caso do armário. Nessa atividade, o supervisor ou eletricista, após inserir suas credenciais, pode alterar nome e/ou senha e então confirma ou desfaz as alterações.

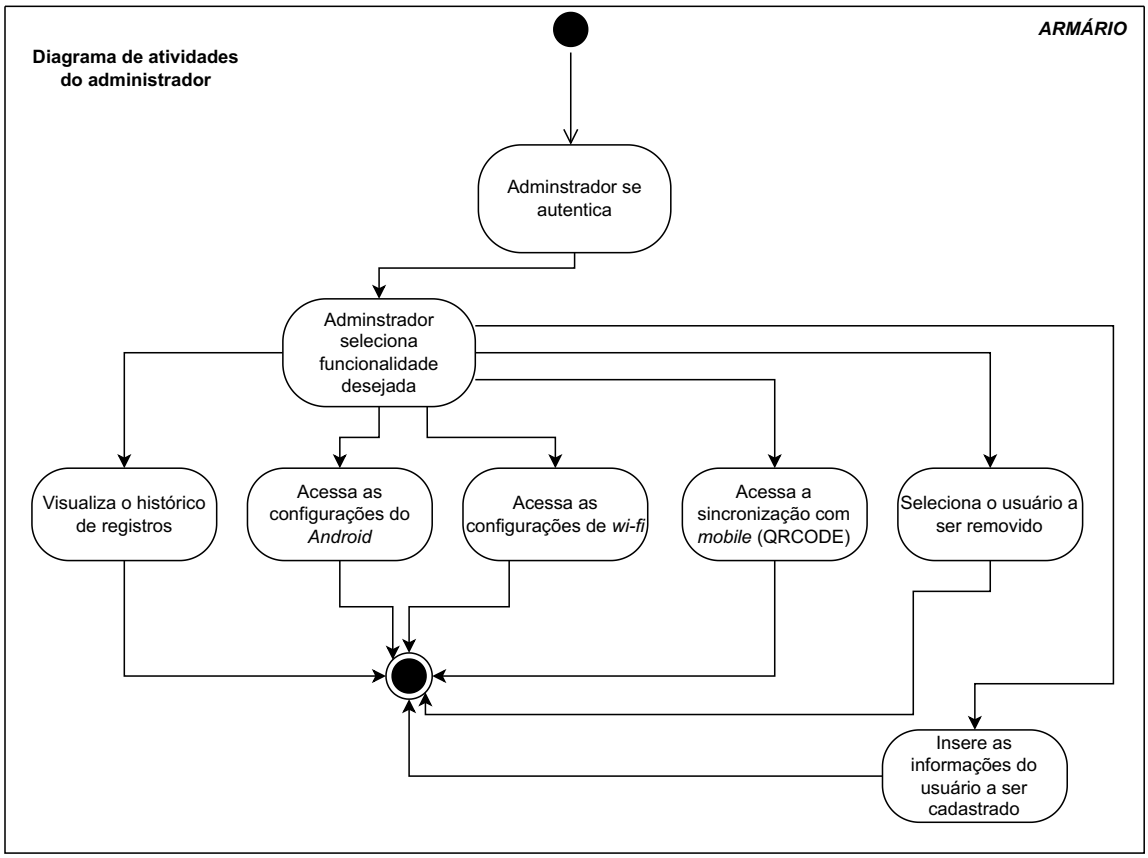
Figura 9 – Diagrama de atividades referente à edição de perfil.



Fonte: AUTOR (2024).

Na Figura 10, observa-se o diagrama de atividades do administrador na aplicação do armário. Nele, encontram-se todas as funcionalidades acessíveis através do credenciamento de um usuário do tipo administrador. Essas são a visualização do histórico de registros, o acesso às configurações do *Android* embarcado, o acesso às configurações do *wi-fi*, acesso ao *QRCode* utilizado para sincronização entre armário e *mobile* e, por fim, a visualização, remoção e adição de usuários.

Figura 10 – Diagrama de atividades do administrador na aplicação do armário.

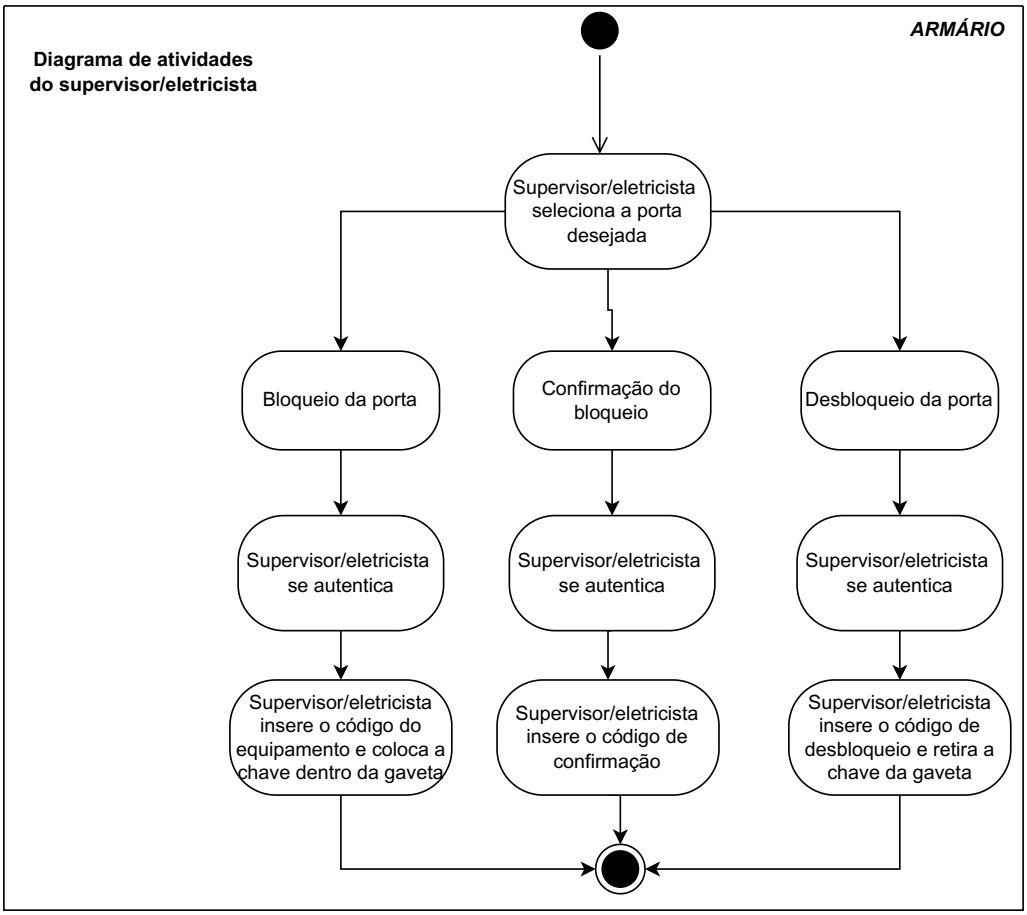


Fonte: AUTOR (2024).

Na Figura 11, observa-se o diagrama de atividades do supervisor/eletricista na aplicação do armário. Além da edição do seu perfil, ele pode realizar mais três ações. A partir da seleção de uma das sete portas disponíveis, esse usuário é capaz de realizar o bloqueio e desbloqueio da porta e a confirmação do bloqueio. Para cada uma dessas ações ele precisa se autenticar e, no caso do bloqueio, também precisa inserir o código do equipamento bloqueado, cuja chave que permite seu desbloqueio será armazenada dentro da porta selecionada.

Na Figura 12, observa-se o diagrama de atividades do administrador na aplicação *mobile*. Nele, após se autenticar, o administrador pode visualizar o histórico ou

Figura 11 – Diagrama de atividades do supervisor/eletricista na aplicação do armário.



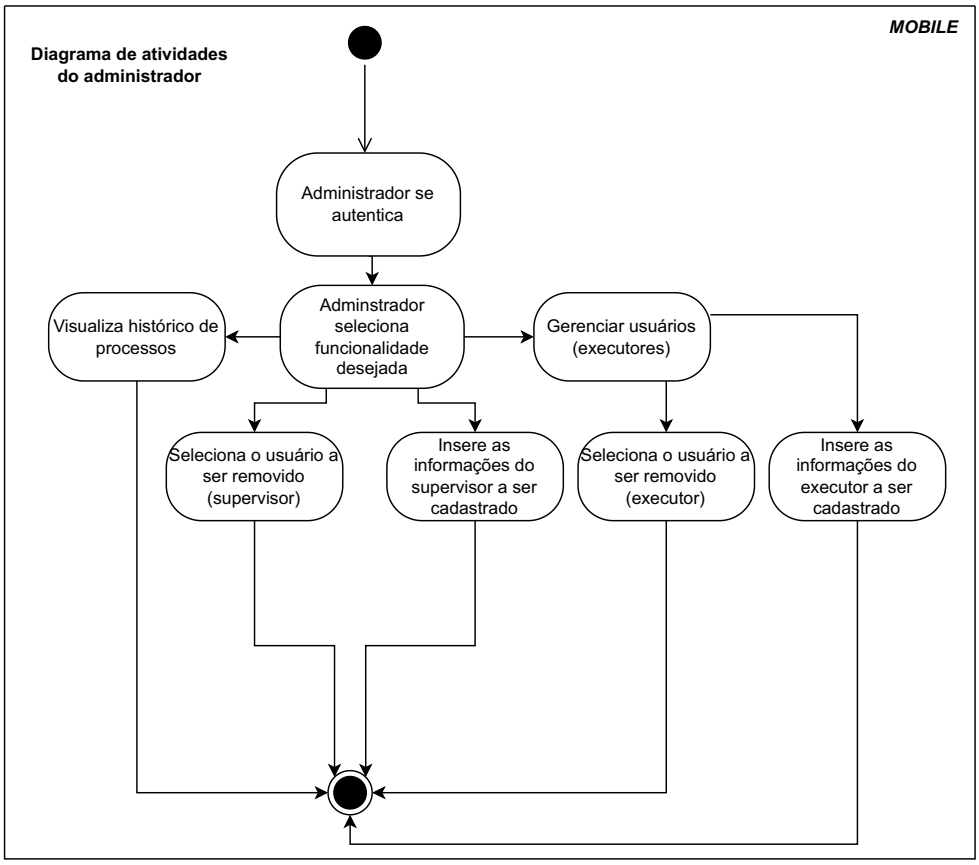
Fonte: AUTOR (2024).

gerenciar os usuários, podendo visualizar, remover e adicionar usuários, tanto supervisor, quanto executor de serviço.

Na Figura 13, observa-se o diagrama de atividades do executor de serviço na aplicação *mobile*. Esse usuário possui apenas duas atividades: entrar e sair do processo de manutenção ou reparo. Para entrar, ele deve selecionar seu nome na lista de usuários pré-selecionados para esse procedimento. Após isso, ele poderá solicitar a entrada no processo. Ele deve então utilizar a senha original, passada pelo responsável pelo seu cadastro no sistema, para definir uma nova senha. Feito isso, a entrada no processo é concluída.

É importante salientar que essa ação é equivalente à colocação do cadeado individual do executor na caixa de bloqueio que guardava a chave capaz de abrir a caixa que continha a chave que desbloquearia eletricamente o equipamento em questão.

Figura 12 – Diagrama de atividades do administrador na aplicação *mobile*.



Fonte: AUTOR (2024).

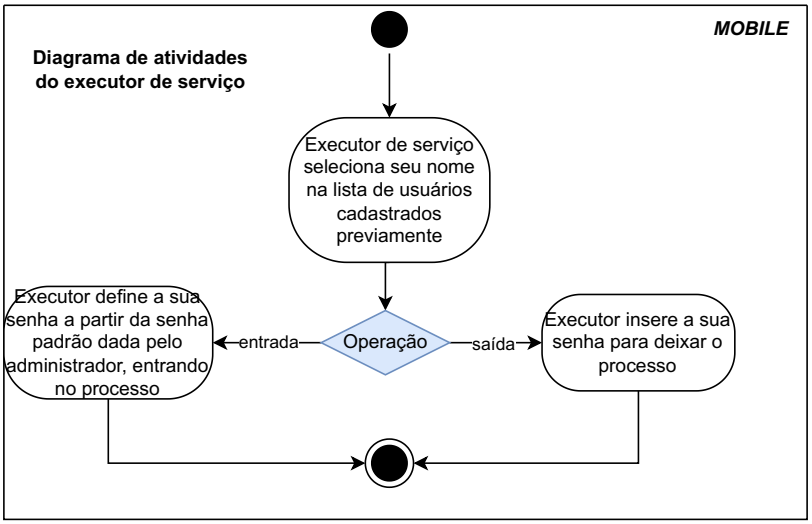
Portanto, ressalta-se que essa atividade cria um travamento no processo, tornando o código de desbloqueio inacessível, até que o executor que entrou na manutenção saia.

Para sair da manutenção, o executor de serviço deve selecionar seu nome na lista de usuários pré-cadastrados e solicitar a saída através do devido comando. Após isso, basta que ele insire a sua senha definida na primeira etapa. Ao fazê-lo, esse executor terá concluído sua participação no processo.

É importante dizer que essa ação é equivalente à retirada do cadeado individual do executor da caixa de bloqueio. Caso esse executor seja o último trabalhador com saída pendente, após sua saída do processo, a aplicação estará habilitada a gerar o código de desbloqueio, assim que o supervisor desejar.

Na Figura 14, observa-se o diagrama de atividades do gerenciamento de participantes na manutenção na aplicação *mobile*, sendo realizada pelo supervisor. Nele o supervisor se autentica e seleciona, a partir da lista de executores cadastrados previamente, o executor desejado e então adiciona ou remove esse usuário do pro-

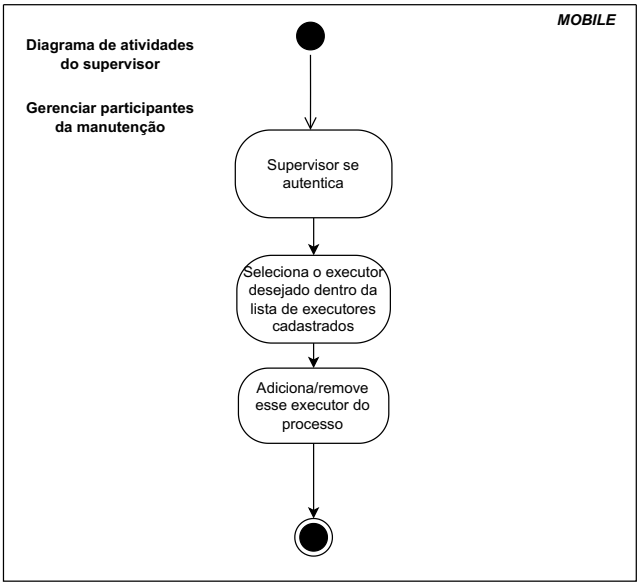
Figura 13 – Diagrama de atividades do executor de serviços na aplicação *mobile*.



Fonte: AUTOR (2024).

cesso. Possivelmente, essa ação também poderá ser implementada como acessível ao administrador.

Figura 14 – Diagrama de atividades do supervisor na aplicação *mobile*.

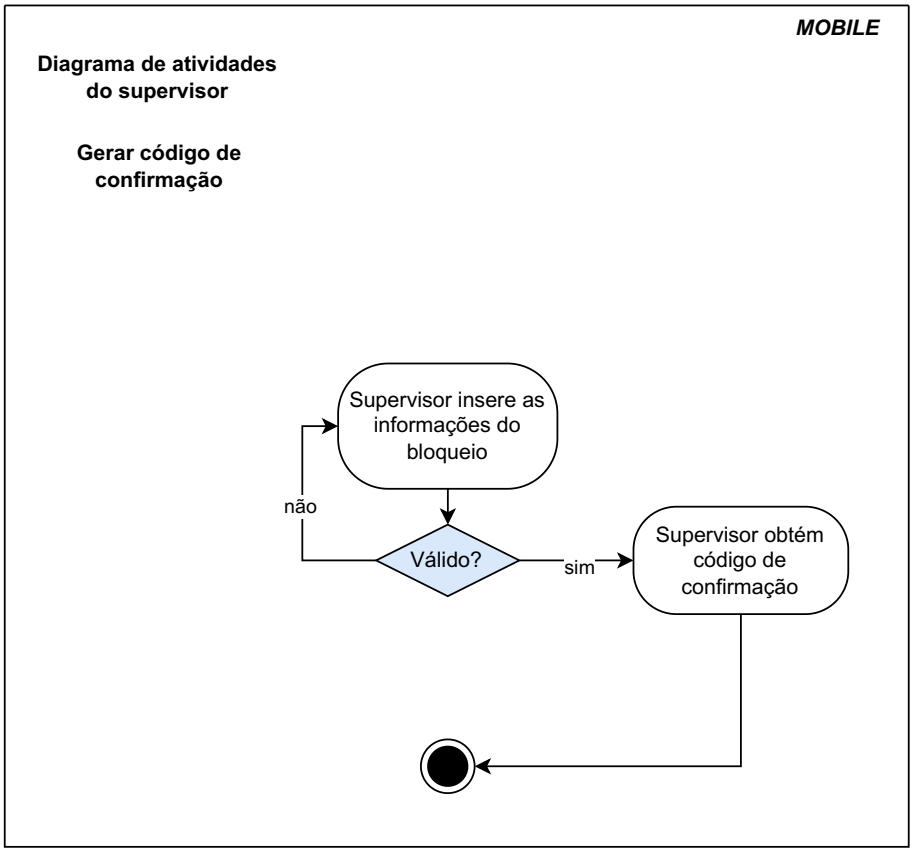


Fonte: AUTOR (2024).

Na Figura 15, observa-se o diagrama de atividades da geração do código de confirmação na aplicação *mobile*, sendo realizada pelo supervisor. Nele, após se au-

tenticar, o supervisor insere as informações do bloqueio. Caso elas sejam válidas, o código de confirmação será gerado e exibido ao supervisor.

Figura 15 – Diagrama de atividades referente à geração do código de confirmação na aplicação *mobile*.

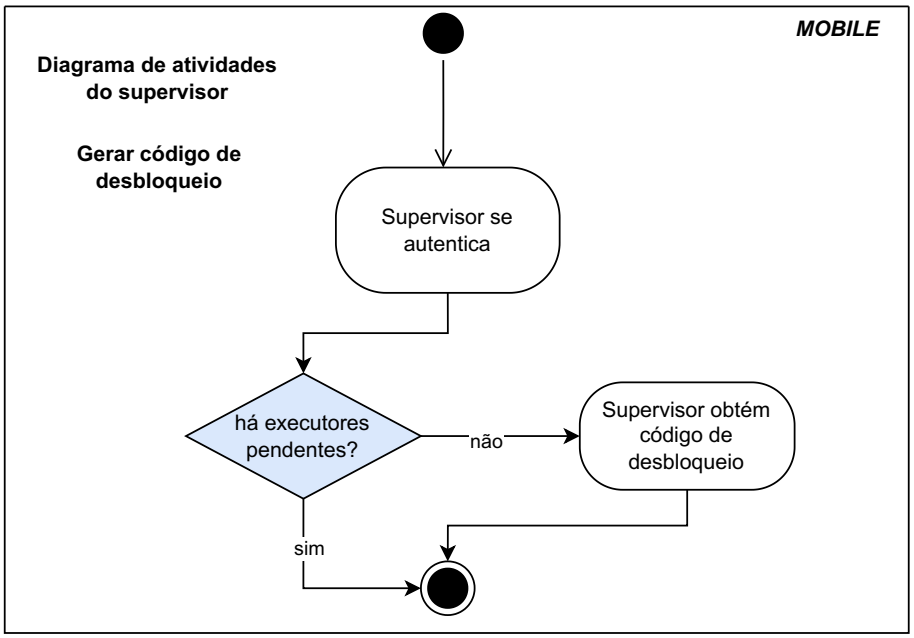


Fonte: AUTOR (2024).

Na Figura 16, observa-se o diagrama de atividades da geração do código de desbloqueio na aplicação *mobile*, sendo realizada pelo supervisor. Nele, após se autenticar, caso não haja executor nenhum cuja saída esteja pendente, o código de desbloqueio será gerado e exibido ao supervisor.

Nesta seção, foram apresentados os diagramas utilizados para compreender e desenvolver a solução proposta. Neste capítulo, também foram exibidos diversos requisitos, funcionais e não funcionais, delimitando e especificando o comportamento desejado para a solução.

Figura 16 – Diagrama de atividades referente à geração do código de confirmação na aplicação *mobile*.



Fonte: AUTOR (2024).

3.4 CONSIDERAÇÕES

Neste capítulo foram elencados os requisitos funcionais e não funcionais do projeto, obtidos a partir das conversas realizadas com os representantes da mineradora. Além disso também foram mostrados os diagramas de casos de uso e de atividades referentes às aplicações desenvolvidas.

Este capítulo possui um conteúdo essencial ao desenvolvimento das duas aplicações e do protótipo como um todo, uma vez que, os requisitos, sejam funcionais ou não funcionais, conferem um norte ao desenvolvimento por determinarem as características que devem estar presentes no produto final.

No capítulo 4, é abordada a descrição do projeto, com detalhamento dos diferentes aspectos do protótipo desenvolvido, desde as ferramentas e materiais utilizados até os métodos empregados no desenvolvimento e suas respectivas conclusões para alcançar o resultado final esperado.

4 DESCRIÇÃO DO PROJETO

Neste capítulo o projeto será descrito, inicialmente com uma apresentação dos *hardwares* e *softwares* empregados, bem como as ferramentas utilizadas para desenvolver, além da implementação da solução. Por fim, os principais desafios enfrentados e suas respectivas resoluções serão abordados.

4.1 *HARDWARES* EMPREGADOS

Nesta seção, serão abordados os principais *hardwares* que compõem o projeto. São dois principais dispositivos: um deles é o equipamento da empresa parceira especializada em *smart lockers*, enquanto o outro é a IHM da 8N1, que já foi mencionada anteriormente neste documento.

4.1.1 Armário inteligente

O armário inteligente é desenvolvido por uma outra empresa, chegando até a 8N1 com muitos segredos de tecnologia. Além de toda a carcaça metálica (Figura 17), o produto possui uma fonte de alimentação e um ATMega 328p, o mesmo microcontrolador do *Arduino*, utilizado para gerenciar o equipamento, comunicando-se tanto com uma placa eletrônica responsável por manipular as portas do armário, quanto com a própria IHM, recebendo comandos dela e retornando informações de estado das portas.

Portanto, os componentes do armário são a carcaça, o microcontrolador, uma placa eletrônica que gerencia as portas, as próprias portas e uma fonte de alimentação. Por razões sigilosas e pelo fato de a tecnologia de gerenciamento das portas serem de uma empresa terceira, não será entrado em maiores detalhes acerca da tecnologia empregada na operação das portas, sendo tratada, daqui em diante, como uma caixa-preta.

4.1.2 IHM

A IHM, ao contrário do armário inteligente, é um produto desenvolvido pela própria 8N1. Ela consiste em um display conectado a uma placa eletrônica com um sistema operacional *Android* embarcado (Figura 18). Ela utiliza a plataforma *Qualcomm Snapdragon* como processador, o que confere um excelente poder computacional para o dispositivo.

O domínio sobre o design eletrônico da IHM permite uma alta personalização do produto, uma vez que a própria empresa é capaz de alterar a sua PCB de modo a satisfazer mudanças desejadas. Além disso, isso confere uma altíssima flexibilidade a esse dispositivo, que pode ser facilmente atualizado para integrar novas *features* em

Figura 17 – Armário inteligente desenvolvido para a solução proposta.

Fonte: AUTOR (2024).

seu projeto. Na Figura 19 é possível observar a PCB da IHM. Ela contém conexões para antena GPS (versão de cerâmica que apresenta maior precisão e velocidade de resposta), *Wi-Fi*, USB, Ethernet, armazenamento SD, SimCard (dados móveis como 2G, 3G, 4G, entre outros), entrada para auto-falantes e microfone, *fingerprint* (leitor de impressão digital), conexão para o display frontal mostrado na Figura, 4 pinos de entrada, 4 pinos de saída, pinos de tensão 24V, 5V, 3.3V e 1.8V, além de pinos para suporte às interfaces de comunicação CAN, SPI, RS485, UART e I2C.

Além da propriedade sobre o quesito eletrônico desse produto, o total domínio sobre o sistema *Android* embarcado também é essencial para conferir o aspecto de altamente customizável para a IHM. Isso permite que ela satisfaça diferentes especificações de projetos, podendo se moldar a diferentes situações e, através da alta customização e personalização, ser utilizada para os mais diversos fins.

Agora que a parte física do projeto (*hardware*) já foi apresentada, faz-se ne-

Figura 18 – Vista frontal da IHM da 8N1 mostrando a sua tela *touchscreen*.

Fonte: AUTOR (2024).

cessário exibir a parte virtual *software*. Na seção 4.2 são abordados os principais *softwares* associados à IHM, bem como as suas principais funções dentro do sistema.

4.2 *SOFTWARES*/*FIRMWARES* EMPREGADOS

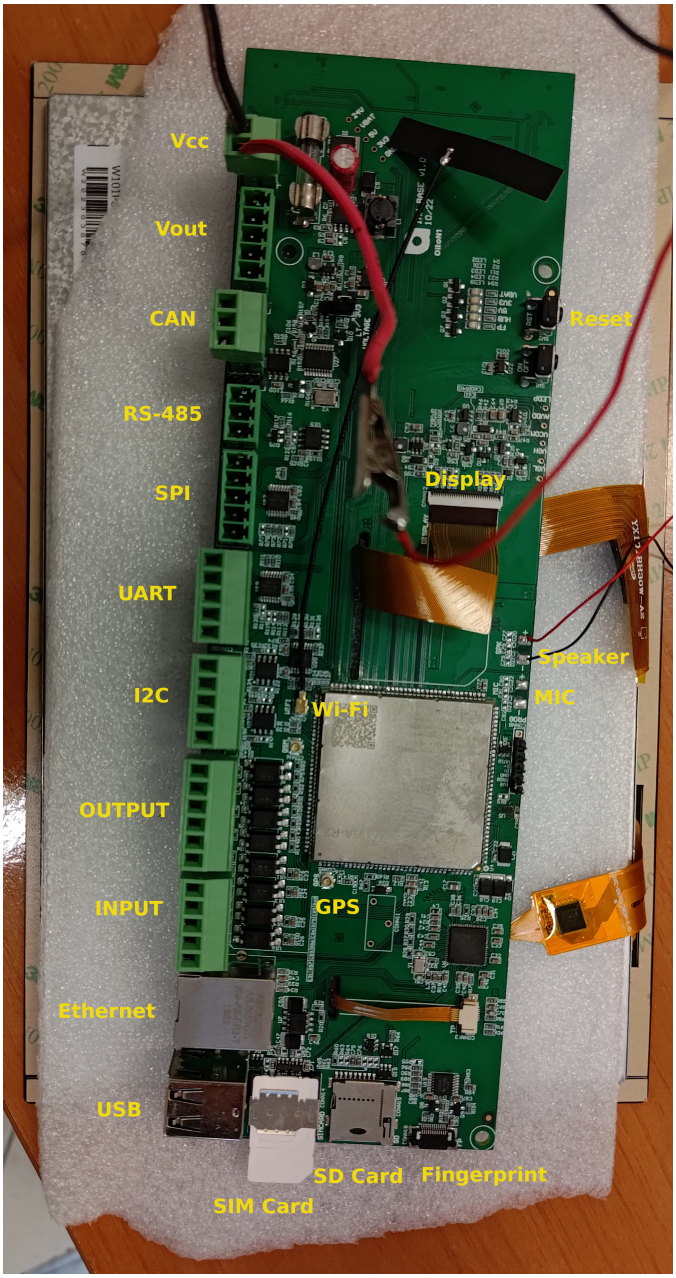
Nesta seção serão vistos e comentados os *softwares*/*firmwares* desenvolvidos para cumprir os requisitos esperados para o sistema final. Ressalta-se que o *firmware* do ATMega 328p, que controla as portas do armário, não será abordado por ser tratado pelo sistema como uma caixa preta. Contudo, ele possui duas *API*'s, uma para abrir alguma porta e outra para ler o estado dela.

Aqui serão tratados o serviço que gerencia a IHM (*manager*) e os *softwares* que realizam a maior parte do trabalho: a aplicação do armário, que roda nesse dispositivo e a aplicação *mobile*, que é projetada para operar em qualquer dispositivo *Android* superior ao *Android* 7.

4.2.1 IHM *Manager*

O IHM *Manager* corresponde um serviço desenvolvido na linguagem C++ com o intuito de gerenciar determinados comportamentos da IHM. Esse programa é exe-

Figura 19 – Vista traseira da IHM da 8N1 mostrando a sua PCB.



Fonte: AUTOR (2024).

cutado de maneira automática e permanente enquanto o dispositivo estiver operando e é essencial para garantir uma série de características tipicamente desejadas nas aplicações nas quais ela normalmente é empregada. Ele é um serviço muito utilizado pela 8N1 em diversos projetos dos variados temas, uma vez que confere uma gama de customizações e comandos do *Android* que podem ser disponibilizados através de *API*'s acessíveis a alguma aplicação, que não conseguiria, de outro modo, realizar tais ações.

Esse *software* possui uma série de *API*'s e rotinas. Entre estas, para a aplicação

a qual versa esse documento, as mais relevantes são as *API*'s de habilitar e desabilitar o *system UI* (*system user interface*), que corresponde um serviço do *Android* que disponibiliza diversos componentes de interface gráfica para facilitar o uso do sistema por parte do usuário. Entre esses elementos estão a barra de *status*, a barra de navegação, os menus do sistema, tela de bloqueio, entre outros.

Salienta-se que a desativação desse serviço representa uma das maneiras de configurar um determinado dispositivo *Android* no modo quiosque, uma vez que ele impossibilita que o usuário feche a aplicação, retorne através do *Android*, navegue por menus do sistema ou mesmo abra alguma aplicação.

Esse serviço do *manager* ainda conta com a implementação de algumas outras rotinas ou tarefas, das quais destacam-se a rotina de verificação de vivacidade da aplicação desejada e a inicialização ou reinicialização automática dela.

Essas tarefas são essenciais ao sistema, uma vez que, como o sistema deve estar no modo quiosque em situações normais, o usuário não seria capaz de abrir aplicação alguma no dispositivo, nem mesmo a aplicação dedicada. Assim sendo, é necessário que exista algum serviço na IHM que o faça, caso contrário, a aplicação nunca seria iniciada e o sistema se tornaria inoperante. De maneira análoga, não há como o usuário reinicializar a aplicação, portanto, é necessário existir uma rotina automática que verifique possíveis *crashes* da aplicação, reiniciando-a quando necessário.

Além deste aspectos, adiciona-se o fato de que, se a IHM corresponde a um dispositivo de uso específico e dedicado a uma determinada aplicação, faz todo sentido que o dispositivo opere com essa finalidade. Isto é, sabe-se que a principal função do sistema operacional é executar essa aplicação, logo, seria muito prático que ele focasse em sempre mantê-la em execução. Além de contribuir com a ocultação do que acontece no dispositivo por trás das cortinas.

Uma outra tarefa importante do *manager* é garantir que a aplicação sempre tenha a prioridade no sistema, isto é, no caso de alguma outra aplicação abrir, colocando a aplicação principal em segundo plano ou minimizando-a, esse serviço possui uma rotina de verificação que ao identificar esse acontecimento retorna a aplicação dedicada ao plano principal e minimiza qualquer outra aplicação cuja execução tenha sido invocada de maneira externa à primeira.

Por fim, a comunicação com esse serviço se dá por meio de um *socket netcat* ou *nc*, através da porta 5022, com as chamadas adequadas das *API*'s implementadas nesse programa.

4.2.2 Aplicação do armário

A aplicação do armário corresponde a um aplicativo *Android*, ou seja, um *Android Package Kit* (*apk*). *Apk* é a extensão de arquivo que o sistema operacional *Android* utiliza para compactar arquivo e instalá-los na forma de um programa, sendo

portanto a extensão de arquivo executável desse sistema operacional, semelhante a *exe* ou ao *bin*.

Essa aplicação foi desenvolvida em JAVA. Essa escolha deve-se ao fato de muitas bibliotecas e classes da IHM estarem todas implementadas nessa linguagem o que facilita a sua utilização e integração com todo o sistema, agilizando todo o desenvolvimento. Outro fator determinante para a escolha dessa linguagem foi a experiência prévia e familiaridade do autor com a linguagem JAVA, em detrimento da total falta de conhecimento acerca da principal alternativa ao JAVA quando se refere ao desenvolvimento para *Android*, o *Kotlin*.

O JAVA foi utilizado explicitamente para construir toda a arquitetura e operação *back-end*, ao passo que, para construir a parte visual (*front-end*) empregaram-se arquivos no formato *Extensible Markup Language* (XML) para auxiliar na elaboração de toda a interface das telas.

Também ressalta-se que a IHM demanda a utilização de uma outra linguagem de maneira implícita no projeto: o C++. A necessidade de uso dessa linguagem surge da existência de uma *Java Native Interface* (JNI), que consiste basicamente em um mapeamento de funções entre a linguagem JAVA e alguma outra, nesse caso o C++.

A JNI é extremamente importante devido ao fato de que todos módulos da IHM foram implementados utilizando C/C++, devido ao seu baixo nível, leveza e velocidade, o que a torna uma linguagem mais ágil e eficiente para sistemas embarcados, cuja capacidade computacional tende a ser limitada. Assim, para que a aplicação possa ter acesso a todas aquelas *features* mostradas na Figura 19, é necessária a inclusão de uma biblioteca de extensão *.so*. Essa, por sua vez, é obtida através da compilação de uma série de classes implementadas em C++, formando essa biblioteca onde os métodos referentes aos módulos disponíveis na IHM podem ser acessados no JAVA, fazendo com que a integração entre aplicação e periféricos da placa acontece de maneira adequada (QIAN *et al.*, 2014).

Aprofundando-se na aplicação do armário, ela consiste em um programa essencial para o funcionamento de todo o protótipo e toda a solução, uma vez que é responsável por executar a atividade central do processo de bloqueio elétrico: o armazenamento da chave e sua posterior liberação. Para tal, ela utiliza o protocolo de comunicação RS-485, por meio do módulo disponível na JNI, para comunicar-se diretamente com o ATMega 328p que controla as solenóides que abrem as portas do armário.

Além dessas operações centrais, a aplicação também executa a importante tarefa de gerar códigos, essenciais ao processo para conferir mais segurança a ele. Outra tarefa importantíssima realizada pela aplicação do armário é a implementação de uma estrutura de persistência de dados, utilizada para armazenar informações dos processos em andamento, bem como uma lista de usuários cadastrados.

Entre outras operações dessa aplicação estão a exibição de determinadas informações do sistema e do processo em andamento, a geração de um *QRCode* extremamente importante para garantir o funcionamento apropriado da aplicação *mobile*, o acesso do usuário a configurações do sistema e a gravação de um registro de atividades, podendo ser consultado dentro da própria aplicação.

Por fim, essa aplicação também é capaz de realizar a leitura de RFID do tipo *Mifare*. Essa leitura utiliza um módulo RFID integrado à IHM, comunicando-se com ela por meio de um protocolo SPI, também disponível na IHM devido à existência da JNI que implementa os métodos que integram o módulo SPI à aplicação.

4.2.3 Aplicação *mobile*

A aplicação *mobile* também é um aplicativo, assim como a do armário, afinal, ela é igualmente desenvolvida para sistemas operacionais *Android*, o que abrange a maior parte dos dispositivos celulares no Brasil, bem como *tablets*. A ideia por trás dessa aplicação é justamente dar muita flexibilidade de dispositivo, uma vez que a aplicação do armário é restrita ao protótipo do armário inteligente, enquanto o lado *mobile* da solução não requer um dispositivo adicional.

Ressalta-se que do ponto de vista de implementação, a aplicação *mobile* herda muitas coisas da aplicação do armário, tanto em design, quanto em funcionalidade e linhas de código propriamente ditas. Ela também é desenvolvida em JAVA por algumas das razões elencadas para o caso da aplicação do armário, adicionando-se àqueles o fato de muitos métodos e classes especificamente iguais aos que seriam necessários no aplicativo *mobile* já estarem implementados de maneira completa e funcional na outra aplicação, precisando executar pequenas alterações para realizar o porte de uma aplicação para outra.

Pelo fato de essa aplicação ser desenvolvida para plataformas variadas, sob as quais a 8N1 e o autor não têm conhecimento e controle de quais periféricos existem, uma das grandes diferenças entre as aplicações é a ausência de uma JNI. Nesse sentido, a gestão e utilização de periféricos fica condicionada às tentativas de utilizá-los através de bibliotecas específicas e nativas do próprio *Android* de cada dispositivo, em que cada fabricante deve ser responsável por configurar e preparar de modo estes periféricos operem adequadamente através dessas *libs* nativas.

Assim como a aplicação do armário, a *mobile* implementa algumas estruturas de persistência de dados, bem como gera códigos, que são extremamente fundamentais ao funcionamento correto e eficiente do sistema como um todo. Além disso, também armazena e exibe informações referentes tanto ao processo em andamento, quanto aos já realizados e concluídos.

Com relação a periféricos do dispositivo, é obrigatório para o funcionamento da solução que o dispositivo em questão possua uma câmera e que seja concedida

permissão ao aplicativo para utilizá-la. Outro periférico utilizado nessa aplicação é o *Near-Field Communication* (NFC), que realiza o papel de leitor RFID, tornando a aplicação *mobile* capaz de interagir com crachás e *tags*. Diferente da câmera, a existência de NFC não é requisito, pois o aplicativo funciona normalmente, ainda que o dispositivo no qual ele opera não possua ou não permita a utilização deste periférico.

Nesta seção foram apresentados os *softwares* desenvolvidos para a implementação da solução proposta, abordando suas características principais e sua contribuição para o funcionamento adequado do protótipo funcional desenvolvido. Na seção 4.3 serão elencadas as principais ferramentas empregadas durante o desenvolvimento do projeto.

4.3 FERRAMENTAS

Nesta seção são abordadas as principais ferramentas utilizadas durante o desenvolvimento do projeto descrito neste documento, bem como a sua respectiva importância. O primeiro deles será a famosa *IDE*: O *Microsoft Visual Studio Code* ou simplesmente *VSCode*.

4.3.1 *VSCode*

O *VSCode* é, segundo uma pesquisa de 2023 do *Stack Overflow*, famoso fórum de programadores, a *IDE* mais popular da atualidade, com mais de 70% da preferência dos entrevistados (STACKOVERFLOW, 2023).

Esse domínio deve-se a muitos diferenciais, dentre os quais estão a leveza e alta perfomance dessa *IDE*, o suporte a inúmeras linguagens de programação, que aliado à infinidade de extensões disponíveis o torna extremamente versátil e customizável, uma funcionalidade de *debugging* integrada à *IDE*, o *Live Share*, que facilita trabalhos colaborativos, entre muitos outros, sendo tudo isso em um *software* gratuito e *open source*.

A sua escolha neste projeto, no entanto, se deu outros diferenciais, além da experiência prévia e da preferência pessoal do autor. Um destes é o recurso *IntelliSense*, que oferece auto-completação de código, agilizando a sua escrita bem como reduzindo probabilidade de erros. O outro principal diferencial dessa ferramenta é a sua integração com versionamento *Git*, algo que será extremamente útil nas próximas etapas do projeto, à medida que ele torna-se mais complexo e cheio de funcionalidades, permitindo controle e acesso a versões mais antigas de maneira fácil e ágil.

4.3.2 *Gradle*

O *Gradle* é uma ferramenta essencial para o desenvolvimento de aplicações *Android*. Ela é uma ferramenta de *build*, importantes para automatizar processos que

sem o seu auxílio deveriam ser feitas manualmente. Ela possui uma integração nativa às ferramentas de desenvolvimento do *Android* e muitos outros recursos, como o *build cache* e *builds* incrementais, que, alidos a toda flexibilidade de projeto oferecida por essa ferramenta, a tornam ideal para o desenvolvimento desse projeto, apresentando tempos relativamente rápidos de *builds* de projeto, o que agiliza os processos de compilação e testes (KOUSEN, 2016).

4.3.3 *Android SDK*

O *Android SDK* é um conjunto de ferramentas fornecidas pelo *Google* para o desenvolvimento de aplicativos para *Android*. Ele fornece o necessário para compilar, depurar e testar aplicativos para Android.

Algumas de suas ferramentas não serão utilizadas, como o *Android Studio*, *IDE* oficial para desenvolvimento *Android*, porém, que apresenta usabilidade e desempenho inferior *VSCode*.

Contudo, outras como o *Native Development Kit* (NDK), que juntamente com a *JNI* permitem a utilização de código nativo C/C++ por aplicações JAVA, o que permite mais eficiência e rapidez (QIAN *et al.*, 2014).

Uma outra ferramenta essencial é o *Android Debug Bridge* (ADB), que opera como uma ferramenta de comunicação entre o desenvolvedor e o dispositivo *Android*, permitindo acesso ao dispositivo, navegação e controle interno dentro dos diretórios, execução de comandos por terminal, manipulação de arquivos, instalação de aplicativos, visualização de *logs* de depuração, entre outros recursos (ANDROID.DEVELOPERS, 2024b).

Além disso, salienta-se que o *ADB* permite a utilização de outras ferramentas externas, das quais será salientada o *scrcpy* (*Screen Copy*), que permite a visualização e gravação da tela do dispositivo *Android*, além da interação com esse dispositivo, o que torna possível ao desenvolvedor realizar todas as etapas do desenvolvimento de maneira remota a partir do computador utilizado, agilizando o processo e tornando-o mais eficiente.

Por fim, uma outra ferramenta essencial, sobretudo nas etapas mais próximas do fim do projeto como um produto final é o *Pro Guard*, que realiza os processos de ofuscação, minimização e otimização de código, o que reduz o tamanho do *APK* (aplicativo *Android*), otimiza sua performance e ofusca o código, dificultando o processo de engenharia reversa. Salienta-se que essa ferramenta será utilizada apenas futuramente. Devido à fase inicial do desenvolvimento deste projeto, não se justifica seu uso no momento.

4.3.4 Ferramentas de elementos *visuais*

As ferramentas de elementos visuais são as ferramentas utilizadas ao longo do desenvolvimento para criar as imagens, ícones e animações inseridas de alguma forma no projeto. A necessidade de personalização do projeto obriga a confecção de imagens e animações que auxiliem na intuitividade do sistema, ao passo que conferem mais identidade ao projeto. Destacam-se três ferramentas utilizadas: o *draw.io*, o *GIMP* e o *Canva*.

O primeiro é uma ferramenta gratuita, *online* e *open source* que permite a criação de inúmeros elementos gráficos, desde ícones a diagramas. Essa ferramenta possui uma longa lista de modelos e formas primitivas, a partir das quais o usuário pode criar diversas imagens. Os ícones das aplicações e os diagramas UML foram criados com tal ferramenta.

Já o *GIMP* (*GNU Image Manipulation Program*) é um programa, também gratuito e de código aberto, que permite a manipulação e edição de imagens de forma profissional. Com uma gama de funcionalidades, e um certo nível de dificuldade, ele fora usado de maneira conjunta ao *draw.io*, permitindo muitas operações mais complexas sobre as imagens que a ferramenta *online*, por sua simplicidade, não permitia ou não apresentava um resultado satisfatório.

A última ferramentada elencada, o *Canva*, é uma ferramenta *online* de edição de imagens e animação, que permite a utilização gratuita, apesar de limitada, de muitas funcionalidades, que permitem a pessoas sem experiência prévia com *design* gráfico e edição de vídeos criar animações visualmente atrativas. Essa ferramenta foi utilizada para produzir a *bootanimation* inserida no dispositivo *Android* e que será detalhada na seção 4.4.1.

Nesta seção foram apresentadas as principais ferramentas utilizadas durante o desenvolvimento deste projeto, bem como suas principais características e motivações de uso. Na seção 4.4 será detalhada a implementação das várias funcionalidades da solução desenvolvida.

4.4 IMPLEMENTAÇÃO DA SOLUÇÃO

Nesta seção é mostrado como a solução proposta e descrita nos capítulos e seções anteriores fora implementada nas aplicações *Android* desenvolvidas, além de modificações relevantes do sistema operacional da IHM.

4.4.1 *Bootanimation* e *bootlogo*

Um dos requisitos do projeto era a personalização das aplicações. Com o intuito de produzir uma identidade visual e customizar o sistema operacional *Android* da IHM

de modo a torná-lo mais dedicado à aplicação do armário, fez-se necessário começar essas alterações pela inicialização do sistema.

Na seção 4.2 fora descrito o importante papel do binário *IHM manager*, que inicializa a aplicação logo que o armário é energizado e o sistema *Android* completa a sua inicialização.

Contudo, desejou-se personalizar a atividade de inicialização do sistema operacional, substituindo as imagens e animações que surgem na tela quando o dispositivo é ligado.

Primeiramente, modificou-se o *bootlogo*. O *bootlogo* corresponde a uma imagem estática exibida pelo *bootloader*, primeiro programa executado pelo dispositivo. Ele serve para indicar visualmente ao usuário que o dispositivo está inicializando normalmente e persiste até que o *kernel* do *Android* esteja em execução e o sistema operacional possa começar a ser carregado.

A alteração do *bootlogo* é um pouco mais complicada, por se tratar de uma atividade atrelada ao *bootloader*, portanto, sem as inúmeras proteções que o sistema *Android* possui contra erros cometidos por usuários ou até desenvolvedores. Por isso, foi necessário muito cuidado ao realizar essa tarefa, através da modificação da imagem *Android*, substituindo a *logo* padrão pela criada para esse projeto na partição de *boot* da imagem, ressaltando que esta *logo* deve estar no formato *RAW* ou *BMP*. Posteriormente, como após em qualquer modificação de imagem, foi necessário empacotar as partições e gravar, na IHM, a nova imagem modificada.

Após a alteração da *bootlogo*, também alterou-se o *bootanimation*. Ele consiste em uma animação carregada pelo *kernel* do *Android* e persiste sua execução em *loop* até que o sistema operacional esteja pronto.

Como já mencionado anteriormente na seção 4.3, foi utilizada a ferramenta *Canva* para criar a animação a ser configurada como *bootanimation*. O vídeo gerado é de curta duração (três segundos) e consiste na *logo* com algumas animações e movimentos de algumas letras e símbolos que a compõem. Apesar de simples, ela reforça a identidade visual criada para esse produto.

A alteração da *bootanimation*, diferentemente da *bootlogo*, não precisa ser realizada através de modificação da imagem do *Android*, podendo ser realizada através do *ADB*, com o comando *adb push bootanimation.zip /system/media/*. Apesar de mais simples de se concretizar, o arquivo a ser enviado deve ser um *.zip* (*bootanimation.zip*), cujo conteúdo deve ser uma pasta contendo os *frames* da animação, além de um conter um arquivo de texto (.txt) denominado *desc.txt*, dentro do qual deve haver informações, como resolução, *fps* (*frames* por segundo), modo de reprodução, *frame inicial*, entre outras configurações relevantes da animação (SHUKLA, 2023).

Após a execução de todo o procedimento descrito, bastou reiniciar o dispositivo para visualizar o novo *bootanimation* em execução de maneira correta, conferindo mais

um nível de personalização e identidade visual ao projeto.

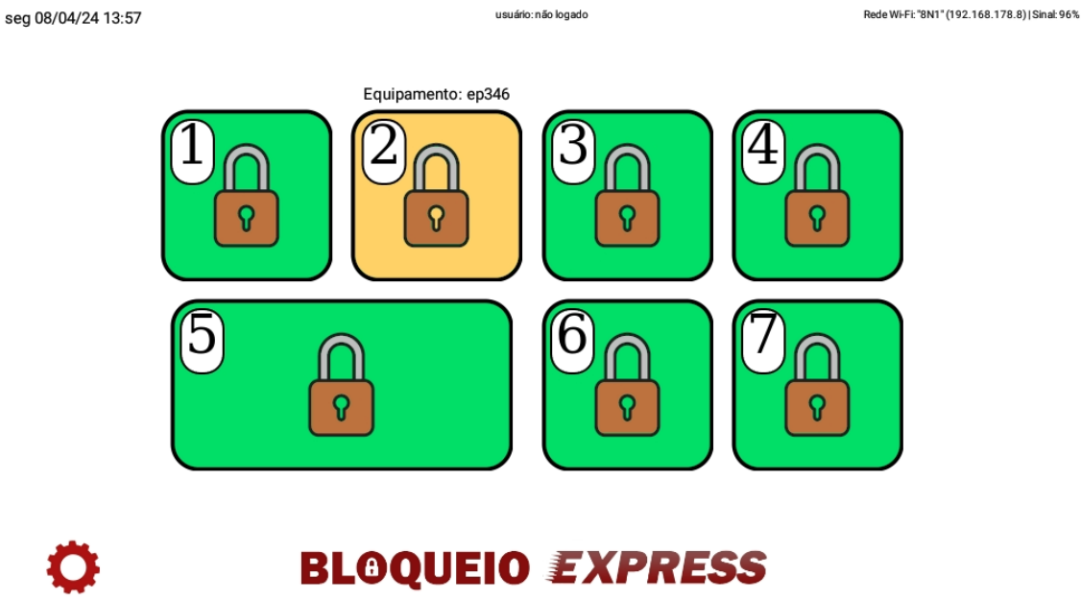
4.4.2 Operação principal

A operação principal ocorre simultaneamente nas duas aplicações, de modo que em determinado momento, uma dependa da outra e vice versa para prosseguir com o processo de bloqueio da chave.

4.4.2.1 Aplicação do armário

A Figura 20 exibe a tela principal da aplicação do armário, na qual é possível ver, além de informações de sistema, como data, hora e conexão de rede, o usuário logado. Também possui um botão que leva à tela de identificação do usuário, o que será visto adiante na seção 4.4.5.

Figura 20 – Tela inicial da aplicação do armário.



Fonte: AUTOR (2024).

Outro componente dessa tela é o painel de portas, composto por sete botões, cada um representando uma das sete portas do armário. Cada porta pode apresentar três estados, representados por cores distintas, tendo, para cada estado, um resultado diferente ao acionar o respectivo botão.

O primeiro estado é **Desbloqueado**, representado pela cor verde. Ao acionar o botão neste estado, o usuário será levado para a tela de identificação e, após inserir credenciais válidas, ele poderá realizar um bloqueio na porta selecionada, que abrirá

após a inserção de algumas informações e gerará um código ao usuário, assim que a porta for fechada pelo usuário e o bloqueio for continuado.

Isso leva essa porta do sistema ao segundo estado: **Aguardando confirmação**, representado pela cor amarela. Nesse estado, qualquer pessoa pode clicar nessa porta e visualizar algumas informações gerais daquele processo, como data de início, responsável pelo bloqueio e equipamento bloqueado.

Junto dessa visualização, existe um botão que leva à confirmação do bloqueio. Ao acioná-lo, o usuário deve autenticar-se e, então inserir o código de confirmação gerado pela aplicação *mobile*. Isso confirma o processo de bloqueio elétrico, levando essa porta ao terceiro e último estado: **Bloqueado**.

Nesse estado, representado pela cor vermelha, qualquer usuário também pode visualizar as informações daquele processo, além de clicar no botão de desbloqueio. Este botão leva novamente à tela de identificação e, posteriormente, o usuário, já credenciado, insere o código de desbloqueio gerado pela aplicação *mobile* e retira a chave de dentro do armário. O que encerra o ciclo, retornando o estado para **Desbloqueado**.

4.4.2.2 Aplicação *mobile*

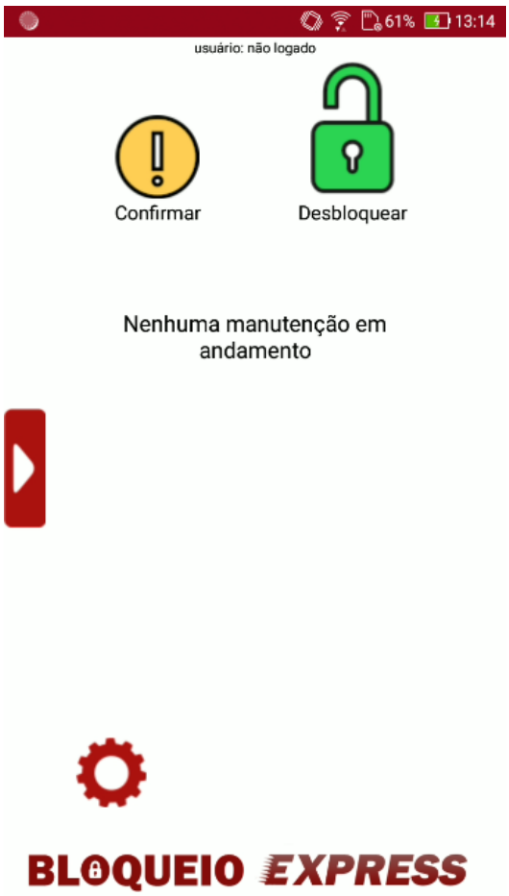
A Figura 21 exibe a tela principal da aplicação *mobile*. Nela nota-se a existência das informações de usuário e de manutenção em andamento. Ainda existe uma engrenagem, que leva à tela de identificação, funcionando de maneira análoga à aplicação do armário, na qual, caso o usuário identificado seja administrador, ele será levado à pagina de administrador, enquanto caso seja apenas um usuário comum, ele será redirecionado para a tela de edição de perfil,

Na tela principal também existem outros dois botões, um para gerar o código de confirmação e o outro para gerar o código de desbloqueio. Ao acioná-los, o usuário se identifica. Após isso, no caso do botão de confirmação, deverá ser inserido certas informações referentes ao processo e então será gerado o código de confirmação. No caso do botão de desbloqueio, simplesmente será informado o código de desbloqueio.

Vale ressaltar ainda que o botão de desbloqueio apenas seguirá esse comportamento quando não houver nenhum trabalhador registrado na manutenção cuja saída esteja pendente, o que seria correspondente, na solução vigente, a ter um cadeado impedindo a abertura da caixa de bloqueio.

Quando houver uma manutenção em andamento, na tela principal também será mostrada, além dos detalhes desse processo, uma lista com os usuários registrados previamente naquela manutenção. A partir dessa lista, é possível selecionar um desses usuários de modo a registrar a sua entrada efetiva ou saída do processo.

Figura 21 – Tela inicial da aplicação *mobile*.



Fonte: AUTOR (2024).

4.4.3 Controle das portas

O controle das portas é realizado através do microcontrolador ATMega 328p, este, por sua vez, comunica-se com a aplicação através da *JNI*, que fornece métodos de acesso às *API's* do microcontrolador utilizando comunicação serial RS-485.

Através desses métodos e desse protocolo, a aplicação consegue enviar comandos de abertura das portas, indicando qual porta deve abrir como parâmetro do método, bem como ler o estado atual de cada uma das portas. Isso permite que o aplicativo JAVA executado pela IHM consiga gerenciar as portas, ainda que esse sistema configure-se como uma caixa-preta.

Como será abordado na seção 4.5, em alguns momentos a comunicação é interrompida, resultando em uma ausência de resposta ao comando ou requisição enviada pela aplicação. Quando isso ocorre, é possível ouvir um estalo no dispositivo, originado da ativação do relê que faz a reinicialização forçada da placa eletrônica do microcontrolador, que, ao reiniciar, reestabelece a comunicação, caso contrário, ela é reiniciada repetidamente, até que a comunicação seja reestabelecida.

Essa abordagem apenas é possível devido a existência da *JNI* que energiza ou desenergiza esse relê através de uma das *GPIO's* da IHM, utilizando o método disponibilizado pela *JNI*, no qual o pino é passado como argumento juntamente do estado a ser forçado.

4.4.4 Geração dos códigos

Uma das funcionalidades mais importantes de todo o sistema é a geração dos códigos. Eles devem ser criptografados, de modo a garantir a segurança de que só serão descobertos caso seja seguido o procedimento adequado, reduzindo a chance de que ele seja adivinhado.

Ao mesmo tempo, ele deve ser igualmente conhecido por duas aplicações que não se comunicam, uma vez que são projetadas para operação *offline*.

Portanto, esses códigos devem ser determinísticos, ainda que não aparentem, e deve seguir regras de criptografia que inviabilizem a sua inteligibilidade.

Por razões de confidencialidade e segurança, o método criptográfico utilizado, bem como a forma como ele é utilizado e como os códigos finais são gerados não serão abordados neste documento.

Com o auxílio desse procedimento, os códigos de bloqueio, confirmação e desbloqueio podem ser gerados de maneira segura, sem que haja grandes riscos de falhas de segurança que permitem alguém abrir a porta do armário de maneira indevida, colocando em risco a vida de alguns indivíduos e a própria integridade do equipamento.

4.4.5 Identificação

A tela de identificação consiste em dois campos de edição de texto, o primeiro pede a inserção de matrícula, enquanto o segundo solicita a respectiva senha desse usuário. Essa tela é acessada sempre que o usuário tenta realizar alguma ação, ou quando ele aciona a engrenagem da tela inicial.

No primeiro caso, o usuário, após se identificar, será sempre redirecionado àquela ação que precedeu essa tela. Na segunda situação, caso ele seja um usuário comum, será redirecionado para a tela de edição de perfil, enquanto será enviado para a página de administrador, caso ele seja esse tipo de usuário.

Nessa tela, ressalta-se a utilização de leitura RFID, que no caso do armário, também fora implementada a partir da *JNI*, através de uma classe de métodos criados para realizar a integração entre o módulo RFID e a IHM por meio do protocolo SPI, suportado pelo dispositivo *Android*, por este conter um módulo SPI.

No caso da aplicação *mobile*, utilizou-se a própria biblioteca nativa de *NFC* do *Android*, que permite a utilização desta funcionalidade no dispositivo, caso o *NFC* exista e esteja disponível (ANDROID.DEVELOPERS, 2024c) (ANDROID.DEVELOPERS, 2024a).

4.4.6 Estrutura de persistência de dados

Como o sistema, tanto no aplicativo do armário, quanto no aplicativo *mobile*, tem sua operação fortemente vinculada ao processo de identificação daquele usuário que deseja realizar a ação, torna-se obrigatório a existência, nessas aplicações, de alguma estrutura de persistência de dados, que sejam capazes de armazenar as principais informações de cada usuário que se cadastra, a fim de poder aceitar ou não as credenciais inseridas na etapa de autenticação,

Futuramente, essas aplicações deverão ser integradas aos bancos de dados unificados dessa mineradora. Contudo, como o projeto encontra-se em uma fase embrionária de desenvolvimento, o foco atual é na funcionalidade central, isto é, o gerenciamento apropriado das portas e adequada geração de códigos. Sendo, no momento, a questão da persistência de dados algo necessário para tudo funcionar adequadamente, porém, ainda assim, secundário.

Portanto, como haverá uma etapa futura de integração de banco de dados, optou-se por implementar as estruturas de persistência de dados não com um banco de dados diretamente, mas através de um mecanismo de persistência de dados nativo do *Android*: a classe *Shared Preferences*.

Essa classe é amplamente utilizada em aplicações *Android* por ser extremamente leve e eficiente, consistindo em uma estrutura de arquivo *XML*, na qual são salvos pares chave-valor. O desempenho temporal dessa biblioteca é superior ao desempenho de banco de dados implementados para *Android*, como o *SQLite*, para pequenas quantidades de dados, que será o caso desse protótipo. Esse desempenho inferior de BD's se deve à necessidade de carregar toda a arquitetura de um banco de dados, o que torna menos eficiente para conjuntos de dados simples e pequenos. Além disso, a sua implementação da *Shared Preferences* é mais simples do que a de uma base de dados, tornando o desenvolvimento mais rápido. Ela possui implementação de métodos de adição, remoção, leitura e edição de valores, portanto satisfazendo todas as operações elementares, os *CRUD's* (WANG; ROUNTEV, 2016).

A estrutura utilizada foi a de selecionar uma das informações, a qual será chamada aqui de ID, como sendo uma chave primária, já que esse dado deve ser único. A partir disso, toda categoria de dado a ser inserido para cada usuário corresponde a um novo arquivo *XML*, no qual a chave é esse ID e o valor correspondente é o próprio dado. Ou seja, se o cadastro solicitasse nome e idade, haveriam dois arquivos *XML*, um para os nomes cadastrados e outro para as idades, nos quais essas duas informações seriam os respectivos valores desses arquivos e o ID cadastrado através da leitura de um crachá, por exemplo, seria a chave (ANDROID.DEVELOPERS, 2024d).

Na aplicação do armário existe apenas uma estrutura de persistência de dados de usuários, referentes aos usuários supervisores, ao passo que, na aplicação *mobile* existem 3, uma para os usuários supervisores, outra para executores de serviço e uma

outra, muito mais volátil para os membros de uma determinada manutenção.

Salienta-se que, devido ao enfoque na operação e na eficiência do sistema como um todo, essa abordagem não gera problemas, uma vez que, além de existir um pequeno número de dados, não serão realizadas consultas complexas e esses dados possuem estruturas extremamente simples como *strings* e inteiros. Todavia, a adoção de um banco de dados, propriamente dito, como um *SQLite*, por exemplo, será de extrema importância futuramente, seja para facilitar a integração com a base de dados existente na empresa, seja para garantir a escalabilidade do sistema.

4.4.7 Configurações

O acesso às configurações, tanto gerais do *Android*, quanto de *wi-fi* eram requisitos do projeto. Ele deveria ser permitido ao usuário administrador do sistema. Como indicado na seção 4.4.5, o usuário administrador se autentica e acessa a sua página (Figura 22). Nessa tela, o administrador será capaz de acessar qualquer uma das duas opções.

Figura 22 – Tela do administrador.

Fonte: AUTOR (2024).

Ao selecionar uma dessas opções, o usuário administrador é redirecionado para a respectiva tela de configuração solicitada ao aplicativo, tornando possível que o usuário altere as configurações como desejar ou modifique/visualize as configurações de *wi-fi*, cumprindo os requisitos funcionais do projeto.

Salienta-se que, o sistema *Android* da IHM mantém o *systemui* desabilitado. Contudo, sem ele o sistema operacional não oferece os elementos da interface visual, dentro os quais se destaca, do ponto de vista funcional, a barra de navegação. É nela que se encontra o botão de retorno do *Android* e sem ela não seria possível permitir ao usuário retornar à aplicação após realizar as ações desejadas, o que obrigaria esse usuário a reinicilizar o dispositivo para poder operá-lo novamente através do aplicativo. Para contornar esse problema, utilizou-se a *JNI* para criar um método capaz de manipular o *systemui*, permitindo habilitar ou desabilitar esse serviço.

Assim sendo, ao acessar alguma das páginas de configurações, o usuário pode visualizar o surgimento dos elementos da interface de usuário do sistema, como a própria barra de navegação, além da barra de status, na parte superior do *display*. Isso permite que o usuário utilize o comando de retorno do *Android*, permitindo a ele voltar ao aplicativo quando desejar. A aplicação, por sua vez, ao recarregar a página do administrador verifica o estado do *systemui* e o desabilita, garantindo que o usuário não possa fechar ou minimizar a aplicação de maneira externa à ela própria, além de garantir a imersão do aplicativo, ocupando toda a tela.

4.4.8 Sincronização

Outra função existente na página do administrador é a sincronização entre aplicações (armário e *mobile*). Ao selecionar essa função, o usuário é redirecionado para outra tela, onde será exibido um *QRCode*. Este, por sua vez, deve ser lido pela aplicação *mobile*. Isso garante a sincronização entre as duas aplicações, sendo essa uma etapa obrigatória para realizar qualquer processo de bloqueio elétrico utilizando essa solução.

Esse *QRCode* contém informações extremamente importantes para o sistema como um todo, sem o qual ele não pode operar corretamente. A sua implementação foi realizada através da biblioteca ZXing (pronuncia-se *zebra crossing*), amplamente empregada em aplicativos *Android* para gerar e ler códigos de barra e *QRCodes*. Para tal, foi necessário editar o módulo */app/build.gradle* do aplicativo a adicionar a dependência, conforme segue:

```
dependencies {
    implementation 'com.google.zxing:core:3.4.1'
}
```

o que permitiu o uso dos métodos associados à geração de *QRCode* (THAPA; TIWARI, 2019).

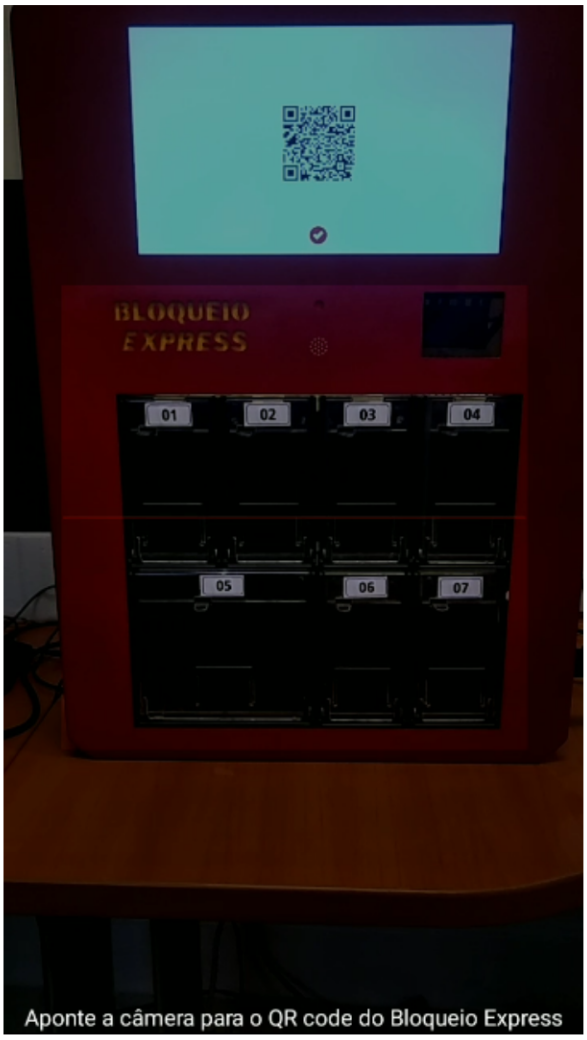
Do outro lado, a aplicação *mobile*, além de solicitar a utilização da câmera através dos das bibliotecas nativas do *Android*, utilizou-se a biblioteca *ZXing Android Embedded*, a seguir:

```
dependencies {
    implementation 'com.journeyapps:zxing-android-embedded:4.2.0'
}
```

uma implementação da *ZXing* focada em *Android*, que permite uma integração mais fácil com a câmera, o que torna todo processo mais rápido e direto.

Após isso, com a funcionalidade de sincronização implementada de ambos os lados, torna-se possível utilizar a solução de maneira eficiente. A Figura 23 exibe a tela da aplicação *mobile* durante o processo de sincronização entre as aplicações conforme descrito anteriormente.

Figura 23 – Sincronização entre aplicações.



Fonte: AUTOR (2024).

4.4.9 Histórico

Ainda na mesma página do administrador, esse usuário pode selecionar a opção de visualizar o histórico de registros dos processos já concluídos. Ao fazê-lo, ele será enviado a outra página, na qual consta uma lista de registros, nos quais constam as principais informações sobre cada processo, como o equipamento bloqueado, horário de início e fim do processo, além do responsável por realizar o bloqueio elétrico e armazenar a chave dentro do armário.

Esse registro foi implementado como um arquivo de texto, localizado no diretório de armazenamento externo do aplicativo do armário, que é totalmente lido e exibido ao usuário quando a página de históricos é acessada pelo administrador.

Enquanto o processo está em andamento, as informações associadas a ele ainda não são gravadas em arquivo e portanto não constam nessa página. Elas ficam salvas temporariamente em uma instância do *Shared Preferences*, como já dito na seção 4.4.2. Apenas ao finalizar esse processo que essas informações são salvas no arquivo, todas de uma vez, e então eliminadas do *Shared Preferences*. Isso é essencial para garantir a integridade desse arquivo, mantendo-o organizado e sem corrompimentos ou mesmo vunerável a algum tipo de *bug* que poderia prejudicar essa funcionalidade.

Nesta seção foram detalhados os métodos de implementação das principais funcionalidades das aplicações que tornam essa solução eficaz. Adiante, na seção 4.5, serão apresentados os principais problemas enfrentados durante o desenvolvimento do protótipo funcional.

4.5 PROBLEMAS ENFRENTADOS

Nesta seção são abordados os principais problemas encontrados durante o desenvolvimento dessa solução, além dos respectivos caminhos seguidos com o intuito de obter a resoluções desses. Foram dois principais problemas encontrados: a IHM, por vezes, estava reiniciando repetidamente durante sua inicialização e a comunicação entre a aplicação e o microcontrolador das portas era encerrada sem razões aparentes.

4.5.1 *Reboot* ao iniciar

O primeiro problema enfrentado foi a ocorrência de repetidas reinicializações durante o processos de inicialização da IHM. Esse problema era algo que acontecia algumas vezes e depois o dispositivo iniciava normalmente.

Devido ao momento do *boot* em que ocorria esse problema, a sua resolução tornou-se muito mais complicada, pois não era possível acessar os *logs* de depuração através do comando *logcat* dentro da IHM, tampouco era possível acessá-la.

Assim, a descoberta da solução desta questão foi realizada de maneira empírica, através de tentativa e erro. Contudo, destaca-se a orientação do supervisor, que através da sua vasta experiência com sistemas *Android*, e que também foi um dos principais desenvolvedores da IHM, sugeriu que o problema estaria na inicialização de algum periférico. Devido a isso, o tempo e tentativas empregados para sanar esse problema não foi tão elevado quanto poderia ser.

Após algumas tentativas, constatou-se que o problema estava na ausência da câmera no dispositivo, enquanto a imagem *Android* que estava gravada na IHM possuia *drivers* de inicialização de câmera, o que causava as repetidas reinicializações no dispositivo no momento em que ele inicializava.

Como a aplicação do armário, até então, não deve fazer uso de câmera, optou-se por modificar a imagem do *Android*, customizando-o de modo a retirar os *drivers* de inicialização da câmera. Após isso, a IHM passou a inicializar da maneira adequada, sem reinicializações, comprovando que o problema era a ausência da câmera quando o sistema operacional continha *drivers* que tentavam inicializar tal periférico, gerando erros que levavam à reinicialização.

4.5.2 Interrupção de comunicação

Outro problema encontrado durante a execução do projeto foi a interrupção repentina e aleatória da comunicação entre a aplicação e as portas. Algumas vezes, sem uma ordem específica de eventos, o microcontrolador parava de responder às solicitações e comandos enviados pela aplicação.

A única maneira de resolver essa questão, na prática, era reiniciar a placa eletrônica na qual o microcontrolador e as portas estavam conectadas. Essa placa é parte do trabalho da empresa parceira, apresentando um desafio um pouco maior para a resolução.

Constatou-se, então, juntamente de um desenvolvedor da empresa parceira que essa placa estava travando, por algum motivo, de modo que nem mesmo o botão de *reset* existente nessa placa respondia ao seu acionamento, o que dificultou ainda mais a solução do problema.

Por fim, a solução pensada e implementada foi inserção de um relê controlado pela IHM através da aplicação entre a fonte e a placa eletrônica que apresentava comportamento indesejado. Desse modo, criou-se uma espécie de botão alternativo de *reset*, permitindo que a aplicação resolva esse problema, reinicializando a placa sempre que notar a interrupção da comunicação.

Esse relê foi soldado a uma das saídas de *GPIO* contidas na IHM, sendo essas saídas possíveis de se controlar através do aplicativo do armário, graças ao papel da JNI na aplicação JAVA. Essa solução demonstra a grande versatilidade permitida pela IHM *Android* da 8N1, possibilitando a resolução de um problema existente dentro de

um outro sistema considerado praticamente uma caixa-preta.

Nesta seção foram abordados os principais problemas enfrentados durante a execução do projeto, além das suas respectivas soluções. Neste capítulo também foi realizada uma detalhada descrição deste projeto, bem com os elementos mais importantes à sua execução e compreensão. No capítulo 5 será realizada uma descrição e análise dos resultados obtidos através das técnicas e procedimentos descritos ao longo deste documento.

4.6 CONSIDERAÇÕES

Neste capítulo foram elencadas as principais características técnicas do projeto, perpassando desde *hardwares* e *softwares* empregados até ferramentas e técnicas utilizadas para desenvolver o primeiro protótipo funcional a ser apresentado à mineradora.

Este capítulo contém o núcleo do projeto desenvolvido, apresentando e detalhando os mais variados aspectos técnicos do projeto. Isso permite um entendimento aprimorado de como o protótipo opera e satisfaz os requisitos desejados, além de como foi possível alcançar tais resultados.

No capítulo 5, são abordados os principais resultados obtidos ao fim projeto, percorrendo todos os requisitos do projeto. Será mencionada uma visita feita a fim de realizar testes em campo. Por fim, serão elencadas vantagens e desvantagens do protótipo desenvolvido.

5 RESULTADOS

Neste capítulo são apresentados os principais resultados. Além disso, serão revisitados os requisitos, tanto funcionais, quanto não funcionais. Também será detalhado um teste em campo realizado ao final do projeto com o protótipo funcional desenvolvido, no qual o seu desempenho foi comparado com a solução tradicional. Ainda neste capítulo também serão elencados vantagens e desvantagens dessa solução.

Após o entendimento do processo de bloqueio elétrico e suas respectivas implicações dentro da realidade da mineradora, além da formulação solução a ser desenvolvida a partir dos requisitos funcionais e não funcionais colhidos junto ao representante dessa empresa, foi possível avançar com o desenvolvimento, alcançando um dos principais objetivos iniciais desse projeto: desenvolver e finalizar um protótipo funcional para ser apresentado aos representantes da empresa cliente.

A solução final consiste nesse protótipo, composto por um armário inteligente gerenciado por um aplicativo JAVA para *Android*, que executa em uma IHM, cujo sistema operacional *Android* fora personalizado de modo a trabalhar de maneira dedicada à essa aplicação. Além do armário, um aplicativo desenvolvido para dispositivos *mobile*, como *smartphones* e *tablets*, com *Android* embarcado. Esse par de aplicações é capaz de implementar e satisfazer os requisitos demandados inicialmente, fazendo com que a solução proposta opere de maneira adequada e seja capaz de substituir integralmente o processo tradicionalmente executado quando bloqueios elétricos são realizados.

Na seção 5.1 será novamente elencado cada um dos requisitos, funcionais e não funcionais, acompanhado de uma breve descrição de como tal requisito fora atendido na solução proposta.

5.1 ADEQUAÇÃO AOS REQUISITOS

Nas seções 3.1 e 3.2, foram elencados e explicados os requisitos funcionais e não funcionais que deveriam estar presentes na solução final. Nesta seção será discorrido acerca destes requisitos e como eles foram atendidos no resultado da solução final. Partindo dos requisitos funcionais, a seguir será detalhado como cada um deles fora satisfeito:

- R.F. 01 (gerenciamento de usuários supervisores): atendido através da estrutura de persistência de dados descrita na seção 4.4.6.
- R.F. 02 (configuração do modo quisque): atendido através do serviço *ihm-manager*, descrito na seção 4.2.

- R.F. 03 (inicialização/reinicialização automática da aplicação): atendido através do serviço *ihm-manager*, descrito na seção 4.2.
- R.F. 04 (acesso às configurações do *Android*): atendido através do botão de configurações, disponível na página do administrador, descrito na subseção 4.4.7.
- R.F. 05 (acesso ao *wi-fi* do *Android*): atendido através do botão de configuração *wi-fi*, disponível na página do administrador, descrito na subseção 4.4.7.
- R.F. 06 (suporte à autenticação *RFID*): atendido através procedimento descrito na subseção 4.4.5.
- R.F. 07 (gerenciamento das portas do armário): atendido através das *API's* disponibilizadas pela empresa parceira integradas pela *JNI*, conforme descrito na subseção 4.4.3.
- R.F. 08 (geração de código para bloqueio): atendido através dos procedimentos descritos na subseção 4.4.4.
- R.F. 09 (armazenamento das informações de cada processo): atendido através dos procedimentos descritos na subseção 4.4.9.
- R.F. 10 (*login* necessário a cada ação): atendido através do redirecionamento do usuário, em ambas aplicações, para a página de identificação
- R.F. 11 (*logout* automático após o término de uma ação): atendido através da implementação de um método de *logout* assim que o usuário finaliza alguma ação.
- R.F. 12 (*logout* automático após ociosidade de 1 minuto): atendido através da implementação de um *timeout*, em ambas aplicações, que basicamente gera um evento de *logout*, quando as aplicações estão em sua tela principal com um usuário logado e passa-se 1 minuto sem qualquer interação por parte desse usuário.
- R.F. 13 (acesso às informações do processo em andamento): atendido através da inserção de um campo de texto, na tela principal de ambas aplicações, contendo as principais informações do processo, caso tenha algum em andamento.
- R.F. 14 (sincronização entre a aplicação do armário e *mobile*): atendido através do procedimento descrito na subseção 4.4.8.
- R.F. 15 (informações de rede na página principal): atendido através da inserção de um campo de texto, na tela principal aplicação do armário, contendo as principais informações de rede da IHM.

- R.F. 16 (gerenciamento de usuários executores do serviço): atendido através da estrutura de persistência de dados descrita na seção 4.4.6.
- R.F. 17 (geração de código para confirmação e desbloqueio): atendido através dos procedimentos descritos na subseção 4.4.4.
- R.F. 18 (cadastro prévio dos executores de cada processo): atendido através da estrutura de persistência de dados descrita na seção 4.4.6.
- R.F. 19 (utilização de senhas para permitir entrada e saída dos executores do processo): atendido através da implementação de uma solicitação de senha pelo usuário para realizar a entrada ou a saída do processo.

Na sequência, é realizado o mesmo detalhamento para os requisitos não funcionais:

- R.NF. 01 (Personalização do projeto): atendido através do uso das ferramentas descritas na subseção 4.3.4 e na subseção 4.4.1.
- R.NF. 02 (Usabilidade): atendido através do desenvolvimento das funcionalidades com foco na dinamicidade, simplicidade e rapidez das funcionalidades das aplicações.
- R.NF. 03 (Intuitividade): atendido através da implementação de uma interface simples e com o uso de ícones auto-explicativos para sua respectiva funcionalidade.
- R.NF. 04 (Manutenibilidade): atendido através da organização do código utilizando as boas práticas de programação e documentação.
- R.NF. 05 (Operação *offline*): atendido através da criação de uma estratégia de sincronização entre as aplicações.
- R.NF. 06 (Segurança no controle das portas): atendido através da inserção de códigos sem os quais não é possível abrir as portas sem seguir o procedimento determinado pela aplicação.
- R.NF. 07 (Segurança na geração e criptografia dos códigos): atendido através de implementação de um algoritmo de criptografia e geraçao dos códigos de bloqueio, confirmação e desbloqueio, garantindo uma aparente aleatoriedade, ainda que seja totalmente determinístico.
- R.NF. 08 (Desenvolvimento em JAVA, para *Android*): atendido através da utilização da linguagem JAVA para desenvolver ambas aplicações.

- R.NF. 09 (Horário do sistema sempre correto): atendido através da existência de um *Real-Time Clock* (RTC) na IHM, que garante a hora corrente mesmo quando o dispositivo está desligado, não precisando obter o horário correto pela conexão à rede.

Na seção 5.2 serão mencionados e detalhados os principais prós e contras referentes ao resultado final obtido no desenvolvimento do projeto e do primeiro protótipo funcional.

5.2 PRÓS E CONTRAS

A solução desenvolvida utilizando aplicações *Android* e uma arquitetura de geração de códigos possui muitas vantagens e algumas desvantagens quando comparada com a solução atualmente empregada na mineradora com a utilização das caixas de bloqueios e cadeados. Contudo, como será visto adiante, os ganhos são muito mais relevantes do que as perdas, fazendo com que a adoção desse sistema se justifique.

Entre os prós desse projeto está o fim da necessidade de os trabalhadores envolvidos de alguma forma com o processo de bloqueio elétrico ou manutenção terem uma chave e um cadeado e precisarem dela para conseguir fazer com que a chave que desbloqueia o equipamento seja recuperada. Agora eles apenas precisam registrar uma senha na aplicação, o que torna tudo mais simples.

Uma outra grande inovação desse sistema é a possibilidade de desbloqueio remoto do armário, isto é, não é mais necessário que o trabalhador esteja no local com a sua chave em mãos para destravar seu cadeado e retirá-lo da caixa, ele pode fazê-lo remotamente informando a senha daquele processo.

Um outro grande ponto positivo dessa solução é a virtualização e automatização das informações, que antes era registrada manualmente em um livro, cujas informações eram posteriormente repassadas para uma planilha. Agora todo o sistema faz isso de maneira automática, registrando as informações em arquivo. Vale ressaltar que, no momento, essas informações apenas estão disponíveis dentro do aplicativo, porém, como estão guardadas em arquivo, uma futura utilização dessas informações ou integração com algum outro sistema externo deverá ser mais simples de se realizar. Como essa questão não era algo essencial para esse momento do projeto, ele deverá ser um ponto de melhoria para o sistema.

Um outro aspecto positivo, e certamente o principal, uma vez que foi o maior motivador desse projeto, é o fim da necessidade de locomoção do ponto de bloqueio até o local da manutenção ou reparo. Isso gerará uma grande economia de tempo e "tempo é dinheiro", portanto, gerando um grande impacto positivo do ponto de vista financeira, pois a máquina ficará menos tempo ociosa.

Por outro lado, como qualquer sistema, ele também possui as suas desvantagens com relação ao sistema vigente. O principal deles é também um dos grandes desafios desse projeto, que corresponde à mudança cultural de um processo tradicionalmente realizado da mesma maneira em todo lugar. Os trabalhadores podem ser a principal resistência à implementação dessa solução, uma vez que estão muito acostumados a realizar o procedimento padrão. E como a problemática tratada neste documento possui um caráter extremamente específico para o ramo da mineração, esses trabalhadores não deverão ver a solução proposta aqui disputar espaço no mercado com os cadeados e caixas de bloqueio utilizadas em praticamente todo lugar, mas sim apenas no ramo da mineração.

Esse aspecto negativo gera um outro problema: a necessidade de realizar treinamentos para capacitar os potenciais usuários desse sistema a operá-lo adequadamente. Ainda que pensado e desenvolvido para ser extremamente intuitivo, é fundamental que os trabalhadores que terão qualquer contato com esse sistema tenham algum conhecimento prévio dele, até para que entendam como esse produto é capaz de garantir a segurança destes trabalhadores.

Um outro contra é o fato de a solução utilizar dispositivos eletrônicos e não mais apenas elementos totalmente mecânicos, como caixas, cadeados e chaves. Esse aspecto é algo a se considerar quando se pensa no tipo de ambiente no qual o produto irá operar. Uma mina é um ambiente extremamente hostil para equipamentos eletrônicos, devido ao excesso de poeira e partículas no ar. Ainda que tenha as proteções adequadas, é natural que a durabilidade dos equipamentos neste tipo de ambiente seja reduzida, o que demandaria manutenções recorrentes ou mesmo substituição desses equipamentos constantemente.

Na seção 5.3 será descrito descrito o teste realizado em campo, em uma mina, que permitiu validar e atestar a utilidade do sistema bem como o seu ganho em eficiência e agilidade quando comparado com o sistema tradicional. Ao fim da seção também será revelado o posicionamento daqueles representantes da mineradora que acompanharam o teste.

5.3 TESTE EM CAMPO

Com o intuito de validar a eficiência e a utilidade da solução desenvolvida, após a conclusão de um primeiro protótipo funcional, foi marcada, juntamente ao representante da mineradora, uma visita a uma das minas para acompanhar um processo tradicional de bloqueio elétrico, bem como comparar com o desempenho do projeto criado e descrito nesse documento. O armário foi levado até a central elétrica onde o bloqueio é realizado (Figura 24).

Neste teste, o procedimento padrão com as inúmeras caixas, chaves e cadeados foi realizado normalmente, tendo o seu desempenho temporal anotado. Simultanea-

Figura 24 – Armário inteligente no local de bloqueio do teste em campo.

Fonte: AUTOR (2024).

mente a isso, foi realizada uma simulação de um processo em paralelo, com uma “chave falsa"sendo armazenada no armário, como se essa fosse a chave que desbloqueia elétricamente o equipamento, tendo também os resultados temporais registrados para comparação.

Constatou-se que enquanto a primeira etapa do processo do lado da nova solução estava concluído, o supervisor ainda se encontrava no local do bloqueio, no aguardo de transporte. A mesma situação ocorreu na segunda etapa do processo. Enquanto o código de desbloqueio já havia sido gerado e informado e a “chave falsa"retirada do armário, portanto, já estando habilitada a reenergizar a máquina, o supervisor ainda estava no ambiente em que a manutenção foi realizada, aguardando transporte.

Por questões confidenciais os resultados desse teste não serão numericamente informados, porém, eles foram suficientemente relevantes para comprovar a efetividade

da solução desenvolvida frente à solução vigente. Isso causou impacto positivo na opinião dos representantes que acompanharam e supervisionaram esse teste.

Devido a esses fatos, o projeto teve um êxito inicial, obtendo a aprovação por parte do gerente de uma das minas, aquela onde o teste foi realizado, e do gerente de projetos que foi o representante da empresa que apresentou inicialmente a problemática e solicitou o projeto. Por inúmeras questões administrativas internas da empresa, o projeto está, no momento, em fase de aprovação por parte da diretoria dessa empresa para que ele possa ser considerado oficialmente como aprovado, podendo, então dar-se continuidade a ele, realizando melhorias e correções, além de pensar e implementar novas funcionalidades.

5.4 CONSIDERAÇÕES

Neste capítulo foram apresentados os principais resultados obtidos, seus prós e contras e alguns detalhes sobre um teste em campo realizado em uma das minas da empresa. Isso possibilitou uma validação do protótipo desenvolvido, além de entender, frente às alternativas existentes, quais os pontos fortes e as fragilidades desse projeto.

No capítulo 6, são relatadas as considerações finais do autor acerca do projeto descrito ao longo de todos os capítulos anteriores. Além disso, também serão elencadas algumas melhorias futuras que poderiam ser implementadas em novas etapas de desenvolvimento ou que representariam um aprimoramento ao produto final.

6 CONCLUSÃO

Neste documento foi apresentada uma nova abordagem a uma etapa obrigatória do processo de bloqueio elétrico. Esse procedimento, como foi explicado nos capítulos anteriores, é uma técnica amplamente realizada em equipamentos elétricos de modo a possibilitar que sejam feitas manutenções, reparos ou qualquer outro tipo de intervenção mais intrusiva ao equipamento.

Esse processo envolve desenergizar o equipamento e mantê-lo nessa condição por meio de uma chave, que deve permanecer inacessível até a conclusão da atividade, oferecendo uma camada de proteção contra acidentes elétricos. A fase de armazenamento seguro da chave é essencial e atualmente é feita usando caixas de bloqueio com cadeados e chaves, inseridos individualmente por cada trabalhador de manutenção. Embora esse método seja simples e eficaz na maioria das empresas, ele apresenta problemas significativos no setor de mineração.

Devido à grande extensão territorial das minas, o transporte da chave entre o local do bloqueio e o ambiente onde a máquina está torna-se um gargalo, por conta do tempo de locomoção e da disponibilidade de veículos para o translado. Esse processo prolonga o tempo ocioso na manutenção, resultando em grandes prejuízos financeiros devido à frequência das operações de manutenção.

A partir desse problema, idealizou-se uma nova solução, como uma alternativa àquilo que é atualmente empregado para realizar esse processo. Esse solução consistiu em empregar uma espécie de armário inteligente para armazenar as chaves e gerenciar a liberação dessa. Para tal, integrou-se a solução desenvolvida por uma empresa terceira, especializada em *smart lockers*, à IHM desenvolvida pela 8N1, que contém o sistema operacional *Android* embarcado. Para essa IHM, desenvolveu-se um aplicativo que seria o responsável por exercer o controle sobre as portas do armário, através de identificações e uso de códigos. Esse armário deveria ficar no local do bloqueio, ao passo que, do outro lado, no local da máquina bloqueada, algum dispositivo *mobile* com sistema *Android*, como um *smartphone* ou um *tablet*, ou até mesmo uma outra IHM, executaria uma outra aplicação, que seria responsável por gerar os códigos utilizados para permitir a abertura da porta do armário que contivesse a chave do bloqueio.

Ao final do desenvolvimento desse projeto, um primeiro protótipo funcional foi concluído. Esse é capaz de executar todas as funcionalidades desejadas através dos requisitos funcionais e satisfazia os requisitos não funcionais. Com o intuito de validar esse protótipo, realizou-se então um teste em campo, em uma mina localizada em Minas Gerais. Esse teste contou com a presença de alguns representantes da mineradora que propusera o projeto. Nele, a solução desenvolvida fora executada simultaneamente ao procedimento vigente na empresa, de modo a permitir uma comparação mais ho-

nesta e eficiente de desempenho do sistema proposto. O resultado desse teste foi extremamente positivo, gerando uma grande expectativa por parte desses representantes de que esse projeto será aprovado pela diretoria da empresa, podendo ter seu desenvolvimento continuado.

Como já mencionado, apesar de demonstrar a sua superioridade com relação ao método tradicional, por questões burocráticas, o andamento e posterior implementação desse projeto depende ainda de uma aprovação de uma diretoria administrativa. Portanto, ainda não é possível contabilizar os ganhos que esta solução trará à empresa. Contudo, é possível estimar que, pelo grande número de processos de bloqueio que são realizados todo mês, além do tempo ocioso médio causado pelo deslocamento entre o ponto de bloqueio e o local da máquina empregado nesses processos e o valor financeiro que isso representa, haverá uma grande economia temporal, que por consequência gerará uma considerável redução de prejuízos associados a procedimentos de bloqueio elétrico.

Além desse fator, também pode-se citar os impactos tecnológicos, uma vez que, com essa nova solução, o procedimento antes realizado todo com produtos puramente mecânicos e registrados manualmente em um livro, sendo repassados posteriormente para planilhas, agora será realizado todo por meio de aplicativos *Android* e todas as informações serão automaticamente gravados em um arquivo que futuramente poderá ser acessado e integrado a outros sistemas da empresa.

Um outro aspecto relevante dessa abordagem é o impacto cultural. Existe uma prática tradicional realizada dentro da empresa quando aborda-se processos de bloqueio elétrico. Essa solução, todavia, propõe uma mudança considerável do modo como faz-se esse processo, demandando um período de aceitação, treinamento e adequação dos membros da empresa e prestadores de serviços. Ainda que tenha esse revés, essa mudança cultural com a utilização dessa proposta trará muitos benefícios do ponto de vista de praticidade e flexibilidade desse processo, permitindo, por exemplo, desbloqueio remotos, ainda que algum dos trabalhadores tenha esquecido de registrar a conclusão da sua atividade, o que atualmente o obriga a retornar até a mina e retirar seu cadeado.

Como esse sistema ainda encontra-se em fases embrionárias, tendo sequer uma aprovação oficial por parte da empresa, existem muitas limitações com relação àquilo que deverá ser o produto final que será enviado a campo e implementado como a solução definitiva substituta aos tradicionais cadeados e caixas de bloqueio.

Entre essas limitações, pode-se ressaltar a impossibilidade de realizar mais de um bloqueio simultâneo no lado da aplicação *mobile*, algo que deverá ser melhor trabalhado futuramente de modo a permitir uma extensão dessa funcionalidade essencial ao sistema.

Uma outra limitação do sistema é o acesso ao arquivo que registra as informa-

ções do processo. O sistema, por ora, apenas permite a sua visualização dentro da aplicação, não sendo possível atualmente exportar ou enviar essas informações para outro lugar ou sistema.

Com relação a essa limitação, o estágio atual do projeto tem como enfoque a validação do sistema quanto ao seu objetivo principal, de bloqueio e desbloqueio das portas, tendo outras questões, como a gestão de informações, como algo não prioritário.

Aliado a isso, muitos aspectos do produto final ainda não foram determinados ou mesmo não são conhecidos pela mineradora, sendo objetos de discussões futuras, em caso de continuidade.

Todavia, é possível elencar melhorias recomendadas a esse sistema desenvolvido, entre os quais a permissividade de mais bloqueios simultâneos na aplicação *mobile* e a possibilidade de enviar as informações coletadas pelo sistema para algum outro lugar onde elas possam ser integradas a algum *software* centralizado de gestão.

A alteração da estrutura de persistência de dados para um banco de dados propriamente dito também será uma melhoria a ser implementada, principalmente pela facilitação de integração com a base de dados principal da mineradora, algo que não foi objeto de trabalho nesta etapa do desenvolvimento.

A utilização de biometria, seja de reconhecimento facial, seja de impressão digital, também seria uma melhoria interessante ao substituir as senhas, atualmente utilizadas, tornando o sistema ainda mais intuitivo e dinâmico.

Por fim, após todas essas ponderações, ressalta-se a grande importância que este projeto possui do posto de vista de virtualização e automação de processos. Isso demonstra a relevância da área de atuação da Engenharia de Controle e Automação, que, em conjunto com outras áreas do conhecimento, é capaz de construir soluções eficientes e inovadoras para os mais variados nichos.

Ela atua como uma cola utilizada para preencher as lacunas existentes entre as demais Engenharias e áreas do saber, de modo a integrar os conhecimentos de naturezas distintas, fazendo com que, como no caso desse projeto, os produtos de um trabalho do campo da Mecânica consigam comunicar-se com os resultados da Eletrônica e ambos estejam integrados a um sistema da Informática, dando vida a um projeto completo, confirmando o caráter multidisciplinar e relevância desse ramo da Engenharia.

REFERÊNCIAS

8N1. **8N1**. [*S.l.*], 2023. Disponível em: https://www.8n1.com.br/. Acesso em: 29 mai. 2024.

ANDROID.DEVELOPERS. **Advanced NFC overview**. [*S.l.*], 2024a. Disponível em: https://developer.android.com/develop/connectivity/nfc/advanced-nfc. Acesso em: 5 jun. 2024.

ANDROID.DEVELOPERS. **Android Debug Bridge (adb)**. [*S.l.*], 2024b. Disponível em: https://developer.android.com/tools/adb. Acesso em: 4 jun. 2024.

ANDROID.DEVELOPERS. **NFC basics**. [*S.l.*], 2024c. Disponível em: https://developer.android.com/develop/connectivity/nfc/nfc. Acesso em: 5 jun. 2024.

ANDROID.DEVELOPERS. **SharedPreferences**. [*S.l.*], 2024d. Disponível em: https://developer.android.com/reference/android/content/SharedPreferences. Acesso em: 6 jun. 2024.

BOSCH-REXROTH. **Smart Lockout System**. [*S.l.*], 2024. Disponível em: https://boschrexroth.africa/en/smart-lockout-system. Acesso em: 26 mai. 2024.

KOUSEN, Ken. **Gradle Recipes for Android: Master the New Build System for Android**. [*S.l.*]: "O'Reilly Media, Inc.", 2016.

MINISTÉRIO DO TRABALHO E EMPREGO. **NR 10**: Segurança em instalações e serviços em eletricidade. Brasília, dez. 2004.

QIAN, Chenxiong; LUO, Xiapu; SHAO, Yuru; CHAN, Alvin TS. On tracking information flows through jni in android applications. *In*: IEEE. 2014 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks. [*S.l.*: *s.n.*], 2014. P. 180–191.

SHUKLA, Anand. **Customising Boot Animation in Android**. [*S.l.*], 2023. Disponível em: https://medium.com/@amshukla1/customising-boot-animation-in-android-24b0e6d78ce4. Acesso em: 3 jun. 2024.

STACKOVERFLOW. **2023 Developer Survey**. [*S.l.*], 2023. Disponível em: `https://survey.stackoverflow.co/2023/#technology-most-popular-technologies`. Acesso em: 2 jun. 2024.

THAPA, Sandeep; TIWARI, Shiva. Android Application, Ours, 2019.

WANG, Yan; ROUNTEV, Atanas. Profiling the responsiveness of android applications via automated resource amplification. *In*: PROCEEDINGS of the International Conference on Mobile Software Engineering and Systems. [*S.l.*: *s.n.*], 2016. P. 48–58.