TARIK SANTIAGO ESMIN

DESENVOLVIMENTO DE UM APLICATIVO MÓVEL PARA INTEGRAÇÃO COM REPOSITÓRIOS DA BASE DE CONHECIMENTO DOS EQUIPAMENTOS ELETROELETRÔNICOS DE UMA ORGANIZAÇÃO

LAVRAS – MG

2020

TARIK SANTIAGO ESMIN

DESENVOLVIMENTO DE UM APLICATIVO MÓVEL PARA INTEGRAÇÃO COM REPOSITÓRIOS DA BASE DE CONHECIMENTO DOS EQUIPAMENTOS ELETROELETRÔNICOS DE UMA ORGANIZAÇÃO

Relatório de estágio supervisionado apresentado à Universidade Federal de Lavras, como parte das exigências do Curso de Ciência da Computação, para a obtenção do título de Bacharel.

Prof. Dr. Ahmed Ali Abdalla Esmin

Orientador

LAVRAS – MG

2020

RESUMO

Na atual era da informação, é crucial as empresas adotarem métodos de organização e simplificação de acesso a dados e informações de diversas origens e que compõem seus processos organizacionais. Neste relatório, foi proposto a descrição e discussão das etapas de desenvolvimento de uma sistema que atua para satisfazer as demandas da organização LCS Link Engenharia em facilitar o acesso e organização das bases de conhecimento dos seus equipamentos eletrônicos, formado por fotos, vídeos, manuais de uso, documentos de serviço, entre outros arquivos. Com o sistema sendo constituído pela implementação de um aplicativo móvel Android em conjunto com a implementação de microsserviços, na qual realizam a integração dos repositórios remotos onde se encontram armazenadas as bases de conhecimento, com o uso de tecnologias como Ionic, código QR, Google Drive e NodeJs.

Palavras-chave: Bases de conhecimento. Aplicativo móvel. Código QR.

ABSTRACT

In the current information age, it is crucial for companies to adopt methods of organizing and simplifying access to data and information from different sources that make up their organizational processes. In this report, it was proposed to describe and discuss the stages of development of a system that acts to satisfy the demands of the organization LCS Link Engenharia in facilitating the access and organization of the knowledge bases of its electronic equipment, formed by photos, videos, user manuals, service documents and other types of files. With the system being constituted by the implementation of an Android mobile application in conjunction with the implementation of micro services, in which they perform the integration of the remote repositories where the knowledge bases are stored, with the employ of technologies such as Ionic, QR code, Google Drive and NodeJs.

Keywords: Ionic. Mobile Application. QR Code.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Na era da informação, empresas tem apostado no bom gerenciamento e organização dos dados e informações que integram o ambiente empresarial. Referenciados como capitais intelectuais de uma empresa, as informações, e os dados que os formam, são utilizadas como insumos, por exemplo, para as tomadas de decisões, aos planejamentos estratégicos e na otimização de processos internos, fatores de suma importância para a sobrevivência e competitividade da organização no mercado atual.

> A cada dia que passa, a informação vem desempenhando um papel vital para as organizações. Gerencia-lá de modo eficaz, usando-a como um recurso estratégico, contribui de maneira fundamental para um bom planejamento.(REGINATO; GRACIOLI, 2012, p. 2)

Nesse contexto, foi proposto o desenvolvimento de um aplicativo móvel que integrasse dados e informação de equipamentos eletroeletrônicos da empresa Lcs Link Engenharia, tais como manuais, vídeos e documentações de serviço, aos quais compõem as chamadas bases de conhecimentos. Objetivando tornar prático e organizado o acesso as bases pelos funcionários da empresa, para o desenvolvimento do sistema, algumas funcionalidades e restrições discutidas foram:

- O aplicativo deve funcionar primeiramente na plataforma Android.
- O aplicativo deve possuir um uso intuitivo e prático e apresentar bom desempenho.
- Devem ser utilizados códigos *QR* como interface de comunicação entre o aplicativo e a base de conhecimento.
- É necessário a utilização de, pelo menos, uma tecnologia de armazenamento em nuvem para composição das bases de conhecimento, como o Google Drive ou o Microsoft OneDrive.

- O aplicativo deve possibilitar a criação de *tickets* de serviços ou relatórios de problemas encontrados sobre os equipamentos cadastrados no sistema.

1.1 Objetivos

Este relatório possui o objetivo de apresentar a experiência profissional obtida durante o período de estágio na empresa LCS Link Engenharia. Para isso, é realizada a descrição das principais etapas de desenvolvimento de um aplicação móvel voltado a integração das bases do conhecimento dos equipamentos eletroeletrônicos, discutida as tecnologias, métodos e técnicas empregadas durante o processo de desenvolvimento, e manifestado, por fim, o resultado obtido do processo de desenvolvimento e as dificuldades encontradas.

1.2 Estrutura de trabalho

O relatório se divide da seguinte forma: o Capítulo 2 é destinado a apresentação geral da empresa onde ocorreu o período de estágio. o Capítulo 3 é voltado para descrição do referencial teórico, responsável por auxiliar o entendimento do trabalho, no Capítulo 4 são apresentados os métodos e materiais empregadas no trabalho. Já o Capítulo 5 descreve as atividades decorrentes do processo de desenvolvimento do aplicativo, e, por fim, o Capítulo 6 é voltado a conclusão e a discussão de trabalhos futuros.

2 DESCRIÇÃO DA ORGANIZAÇÃO

Neste capítulo são descritas as informações gerais sobre a organização onde ocorreu o estágio, apresentando seu histórico, objetivos, importância de mercado, área de atuação e principais atividades desenvolvidas.

2.1 LCS Link Engenharia

A LCS Link Egenharia é uma empresa sediada na cidade de São Paulo, fundada em 1996, com foco em prover soluções na área de sistemas de automação e geração de energia elétrica. Possuindo como clientes bancos, concessionárias de energia e telefonia, siderúrgica, entre outros, a empresa atua atendendo as demandas dos seguintes segmentos:

2.1.1 Geração e transmissão de energia

Oferece suporte a infraestrutura elétrica de edifícios comerciais ou indústrias, serviços de automação para grupos geradores e usinas hidroelétricas, através, por exemplo, no fornecimento e instalação de painéis de controle, serviços de integração e desenvolvimento de softwares na área de PLC,*Power line connection*, e de supervisórios.

2.1.2 Missão crítica

A missão crítica é definida pela necessidade dos negócios em manter a disponibilidade de serviços, aplicações e processos tidos como essenciais, aos quais sua perda ou paralisação ocasionaria em considerável prejuízos financeiros ou sociais. Nesse contexto, a LCS Link oferece variadas soluções como a instalação de centros de controle, criação e implementação de softwares SGO, sistemas gerenciadores de operadores, e o desenvolvimento de BMS, sistema de gerenciamento de prédios.

Recentemente, buscando atender as demandas internas e de clientes, a empresa LCS Link tem diversificado sua área de atuação com a realização da parceria com a empresa MokaMind Software, que atua na área de desenvolvimento de software (TI), inteligência artificial(IA) e internet das coisas(IoT), com localização na cidade de Lavras. Sendo a MokaMind Software correspondendo ao local onde o presente trabalho obteve maior colaboração durante seu desenvolvimento.

3 REFERENCIAL TEÓRICO

Nesse capítulo são apresentados os principais conceitos utilizados durante o desenvolvimento do sistema.

3.1 Processo de software

Com base na literatura, pode-se afirmar que "Um processo de software é um conjunto de atividades que leva à produção de um produto de software. Essas atividades podem envolver o desenvolvimento do software propriamente dito"(SOMMERVILLE, 2007, p.42).

Há, na engenharia de software, uma grande número de processos de softwares diferentes, com cada processo de software se adaptando melhor a características específicas dos sistemas em desenvolvimento e das pessoas que os desenvolvem (SOMMERVILLE, 2007). Apesar de existirem variados processo de software, pode-se citar algumas atividades comuns a eles: (SOMMERVILLE, 2007)

1. Especificação de software: Atividade de definição das funcionalidades e restrições do software nos chamados requisitos funcionais e não-funcionais.

2. Projeto e implementação de software: Atividade relacionada a implementação do software em si, segundo as demandas levantadas.

3. Validação do software: Atividade que garante que o software cumpra as especificações do cliente.

4. Evolução do software: Atividade que determina as mudanças necessárias para que o software atenda às necessidades mutáveis do cliente.

3.1.1 Modelos de processo de software

O modelo de processo de software é considerado como uma forma de representação abstrata de um determinado processo de software (SOMMERVILLE, 2007). Entre os diferentes modelos de processos de software, são destacados os seguintes (PRESSMAN, 2011) :

1. Modelo cascata: modelo empregado em casos onde os requisitos do software são bem conhecidos e definidos. Também denominado de ciclo de vida clássico, o modelo cascata adota uma abordagem sequencial e sistemática para as atividades de desenvolvimento de software, como exemplo, pode-se iniciar com o levantamento de requisitos, seguido por fases de planejamento, modelagem, construção e implantação do software.

2. Modelo de processo incremental: modelo utilizado em casos em que os requisitos do software são razoavelmente conhecidos, mas que ainda podem ser modificados no futuro, na qual, devido o escopo do trabalho de desenvolvimento, o uso de um processo puramente linear não é adequado. Sua característica está na divisão do processo em incrementos, com cada incremento composto um conjunto de atividades de desenvolvimento, desenvolvido de maneira linear e apresentando uma nova validação ou entrega de software no final do incremento.

3. Modelo de processo evolucionário: baseando-se no fato que software, considerado um sistema complexo, evolui ao longo do tempo, o que torna inadequado o uso de um planejamento de desenvolvimento intrinsecamente linear. Nesse contexto o modelo de processo evolucionário se propõe em separar as atividades de desenvolvimento em ciclos ou iterações, de maneira a permitir a entrega de versões cada vez mais completas do software.

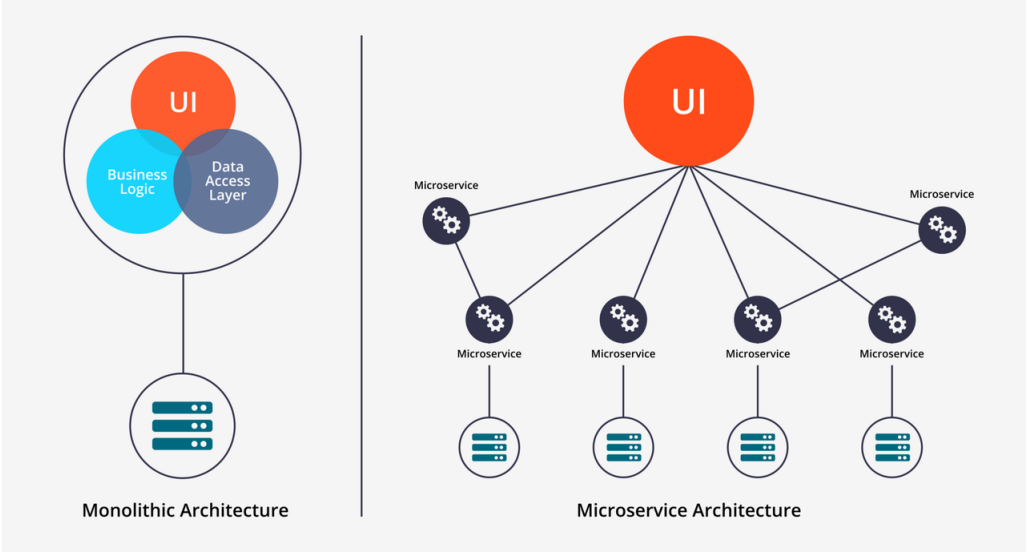
3.2 Arquitetura de microsserviços

Sobre a arquitetura de microsserviços, uma definição válida é dada como:

> Uma arquitetura de microsserviços é uma arquitetura nativa da nuvem que visa executar sistemas de software como um pacote de pequenos serviços. Cada serviço é implementável independentemente em uma potencial diferente plataforma e pilha tecnológica. Pode ser executado em seu próprio processo enquanto se comunica através de mecanismos leves como RESTful ou APIs baseadas em RPC. (Balalaie; Heydarnoori; Jamshidi, 2016, p.1)

Na figura 3.1 está disponível uma representação de uma arquitetura de microsserviço comparada a arquitetura monolítica de 3 camadas, sendo elas a camada de *UI*, ou camada de apresentação, camada lógica e de acesso a dados.

Figura 3.1 – Diagrama de uma arquitetura de microsserviços em comparação a monolítica



Fonte: (MALAV, 2018)

A diferença fundamental entre a arquitetura monolítica e a baseada em microsserviços está que a arquitetura monolítica implementa todas as regras de negócios para serem executadas em um único processo, enquanto que o micros-

serviço implementa as funcionalidades através de um conjunto de serviços que executam de maneira independente em processos separados.

As vantagens do uso da arquitetura baseada em microsserviços, em contraste de uma arquitetura monolítica, estão na maior adaptabilidade ao sistema quanto a adoção de novas tecnologias, na redução do custo de desenvolvimento e de manutenção, melhor capacidade de escalabilidade, resiliência do sistema, e no estrutura mento de equipes de desenvolvimento com a divisão do componente em um conjunto de serviços, impactando na diminuição do *time to-market* do software (Balalaie; Heydarnoori; Jamshidi, 2016).

4 MATERIAIS E MÉTODOS

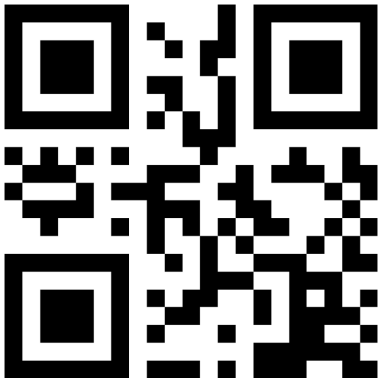
Para desenvolver o sistema proposto, foram empregadas diferentes materiais e métodos aos quais serão descritos a seguir.

4.1 Materiais

4.1.1 Código QR

O código QR é uma imagem caracterizada como uma barra de códigos bidimensional, ao qual possibilita codificar e armazenar mil vezes mais dados que a barra de código tradicional, com os dados podendo ser facilmente acessados ao escanear a imagem pela câmera do celular (KIESEBERG et al., 2010).

Figura 4.1 – Exemplo de um código QR



Fonte: (PEREZ, 2019)

A adoção código QR foi ampla e popular, com a utilização do código em diferentes aplicações como meios de armazenamento de URLs, emails, número de telefones e endereços físicos, além de representar diferentes formatos de dados como cartões de visita, pôsters e letreiros (KIESEBERG et al., 2010).

No contexto do sistema desenvolvido, o código QR foi utilizado para facilitar a rotulação e a organização das bases de conhecimento dos equipamentos eletroeletrônicos, ou de seus relacionados ambientes, funcionando como uma interface de comunicação entre as bases e o aplicativo.

4.1.2 Ambiente de desenvolvimento

Ao ambiente de desenvolvimento foi empregado o sistema operacional Ubuntu 18.04 em conjunto com o editor de código-fonte Visual Studio Code, que são ferramentas de fácil uso, de ampla adoção na área de desenvolvimento de software e de código aberto, possuindo licenças de uso gratuitas. Ademais, foram utilizadas algumas ferramentas de apoio durante o desenvolvimento, como o Postman, software que simula requisições HTTP, empregado para realização de testes no microsserviço, e o draw.io, ferramenta online para o desenvolvimento de diagramas que foi utilizada durante as atividades de construção dos fluxos do sistema.

4.1.3 Banco de Dados

Como software de banco de dados, foi utilizado o MongoDB. 0 MongoDB é classificado como um banco de dados do tipo NoSQL, sendo baseado em documentos auto-descritíveis para representar dados em uma sintaxe similar ao JSON, sem fazer o uso de esquemas e nem do modelo relacional dos bancos SQL. Entre as vantagens que o MongoDB entrega ao mercado estão facilidade e flexibilidade de escalonabilidade horizontal, possibilidade de indexação similar a bancos relacionais, utilização de mecanismo de réplica de dados Mestre-Escravo, entre outros (ABRAMOVA; BERNARDINO, 2013).

Em conjunto ao MongoDB, foi utilizado o NoSqlBooster, software de interface gráfica para o MongoDB que oferece algumas funcionalidades como editor de código para linguagem de consulta de dados, agendamento de tarefas, monitoramento do uso de recursos do computador pelo banco, entre outras funcionalidades.

4.1.4 Linguaguens de programação e Frameworks

Para o desenvolvimento do sistema foi utilizado o Ionic, *framework* de código aberto voltado ao desenvolvimento híbrido de aplicativos móveis, em con-

junto com o Angular, *framework* para criação de interfaces de aplicações web, e com tecnologias comumente utilizadas no *frontend* como HTML, CSS e a linguagem de programação javascript (IONIC, 2020). Ademais, fez-se o uso do Capacitor que é uma tecnologia de conversão do um aplicativo web para o mobile, habilitando o uso de funcionalidades nativos do dispositivo pelo aplicativo.

As vantagens de utilização do Ionic está na disponibilização de diversos componentes, pré-estilados e adaptados para uso em interfaces móveis, de práticas ferramentas de depuração, e um cliente de linha de comando (CLI) para auxiliar no gerenciamento do projeto. Somado a isso está seu baixo custo de desenvolvimento, com a possibilidade de implementação de um aplicativo mobile com o uso de tecnologias já consolidadas no desenvolvimento web, como Javascript e HTML, e, como o Ionic possui uma abordagem híbrida, uma mesma base de código pode ser executada nas plataformas iOS e Android (IONIC, 2020).

Já na implementação do *backend* do sistema, foi empregada o NodeJS, caracterizado como um ambiente de execução Javascript assíncrono e baseado em eventos. Com o NodeJS apresentando benefícios como, alto desempenho em situações de processamento de várias requisições concorrentes, eficiente utilização de recursos de hardware, baixa curva de aprendizado e a presença de uma grande e ativa comunidade, com a disponibilização de diversas bibliotecas. (Tilkov; Vinoski, 2010).

4.1.5 Google Drive

Como repositório de armazenamento das bases de conhecimento dos equipamentos, foi utilizado o Google Drive, o serviço de armazenamento e sincronização de arquivos em nuvem oferecido pela Google. Como vantagens, o Google Drive oferece um plano gratuito com até 15 GB de espaço disponível, integração de outros serviços como Google Sheet e Google Slides, uma excelente UI e

UX com o serviço apresentando um expressivo número de 1 bilhão de usuários cadastrados em 2018.

Ademais, o Google Drive oferece um conjunto de APIs REST próprio, ao qual permite que terceiros integre as diversas funcionalidades de armazenamento do Drive em suas aplicações. Com a possibilidade de executar operações como criar,excluir, renomear, compartilhar e alterar permissões de acesso a pastas e arquivos em um determinado repositório. [1]

4.2 Métodos

4.2.1 Metodologia de desenvolvimento

Para este trabalho, foi empregue a metodologia incremental, já que o projeto foi executado de maneira individual, possui os requisitos funcionais relativamente bem delimitados, existindo a possibilidade de mudança, e pouca comunicação com o cliente do que torna o planejamento das atividades em incrementos simples e interessante.

4.2.2 Etapa de definição de projeto

Fazem parte da etapa de definição do projeto, as seguintes atividades:

- Definição e listagem dos requisitos funcionais e não funcionais da aplicação.
- Utilização dos requisitos levantados para a construção de diagramas e escrita de documentos voltados para o suporte no desenvolvimento do software.
- Levantamento das tecnologias disponíveis e escolha das quais serão empregadas no desenvolvimento do projeto

[1] Link para documentação sobre as APIs REST do Drive: <https://developers.google.com/drive>

4.2.3 Etapa de desenvolvimento do projeto

Fazem parte da etapa de desenvolvimento do *software*, as seguintes atividades separadas por incrementos:

Primeiro incremento:

- Preparação do ambiente de desenvolvimento, com as instalação das ferramentas e dependências necessárias.
- Codificação inicial do aplicativo com a definição das primeiras telas a serem apresentadas como protótipo.
- Validação do protótipo implementado.

Segundo incremento:

- Realizar a modelagem do banco de dados
- Popular o banco de dados com os dados a serem utilizados pelo sistema.
- Implementação do *backend*, seus serviços e APIs.
- *Deploy* do *backend* no servidor.
- Realizar integração entre o *backend* e o aplicativo.
- Validação final do aplicativo.

5 DESENVOLVIMENTO

Nesse capítulo é apresentada a descrição das atividades de desenvolvimento do sistema através dos documentos, diagramas e técnicas empregadas durante o processo de construção do software.

5.1 Levantamento de requisitos

Inicialmente, foi obtido e listado os requisitos funcionais e não funcionais, que orientaram o restante das atividades de desenvolvimento do sistema.

Tabela 5.1 – Requesitos funcionais

#	Descrição
1	O usuário pode logar no aplicativo utilizando um login, ou email, e senha.
2	O sistema deve emitir um token de acesso quando o usuário realiza login.
3	O sistema deve se conectar a, pelo menos, um serviço de armazenamento em nuvem na qual estão armazenadas as bases de conhecimento dos equipamentos eletrônicos.
4	As bases de conhecimentos são agrupadas em uma hierarquia de acesso, na qual o usuário pode acessar a base de conhecimento de um único equipamento ou de um ambiente que contém um conjunto de equipamentos.
5	O código qr pode esta relacionado tanto uma base de conhecimento de um equipamento como de um ambiente.
6	O aplicativo deve possuir capacidade de escanear um código QR e obter informações sobre os equipamentos eletroeletrônicos.
7	Cada código QR deve ter seu acesso restrito a uma determinada lista de usuários.
8	O aplicativo deve fornecer um modo de acesso direto ao repositório em nuvem dos equipamentos, sem que haja necessidade de escanear um código QR.
9	O aplicativo deve disponibilizar um formulário para editar as informações dos equipamentos e enviar em formato de *ticket* de serviço para o servidor.
10	O aplicativo deve permitir que os usuários anexem fotos ao formulários de edição das informações, com as fotos sendo salvas no repositório remoto do equipamento.

Fonte: Do autor (2020)

Tabela 5.2 – Requesitos não funcionais

#	Descrição
1	O aplicativo deve funcionar na plataforma Android.
2	O aplicativo deve possuir um design agradável e intuitivo, com a escolha de cores seguindo ao apresentado pelos outros softwares da empresa.
3	A lógica de autentificação e os dados dos usuários devem estar totalmente contidos no sistema.

Fonte: Do autor (2020)

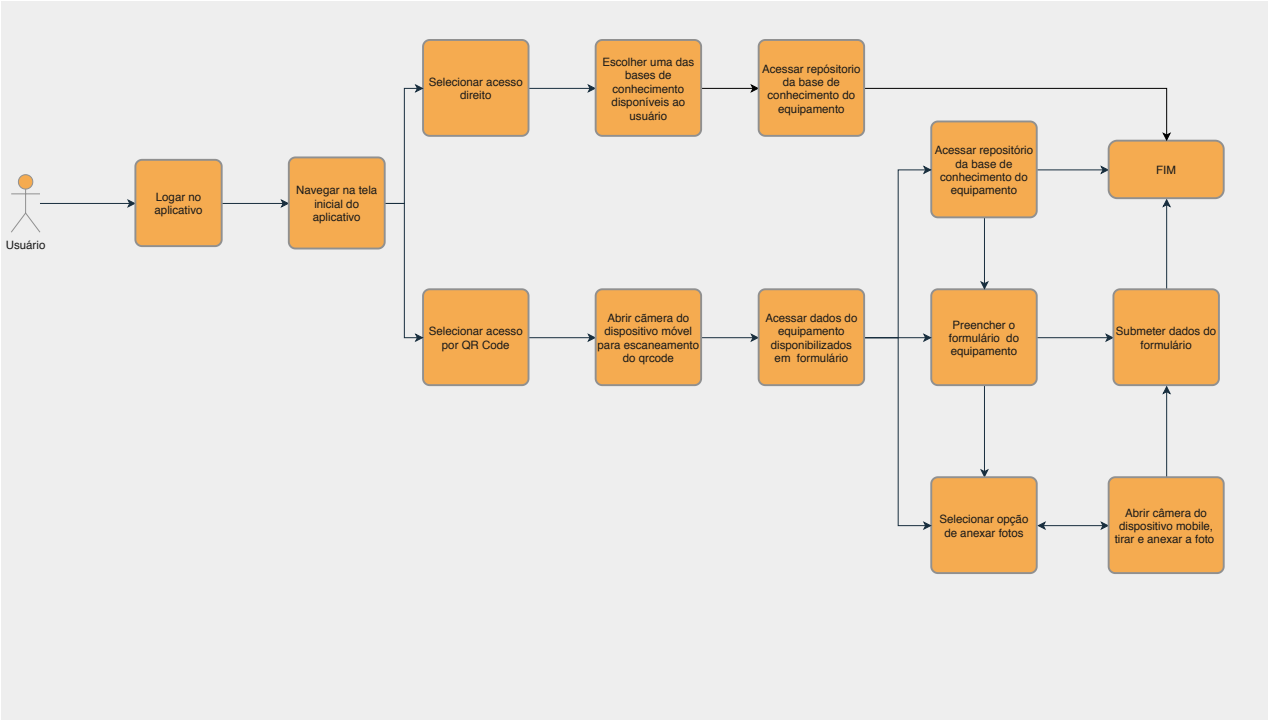
5.2 Implementação inicial do aplicativo

Nessa seção, estarão descritas as atividades que constituíram o desenvolvimento da uma versão inicial do aplicativo móvel.

5.2.1 Projeto

Inicialmente, foi proposto a criação de um fluxo de interação do usuário com o aplicativo, na qual busca-se entender as etapas que o usuário executa para realizar a funcionalidade principal do aplicativo, descrito pelo acesso aos dados dos equipamentos eletroeletrônicos e da sua base de conhecimento em um repositório em nuvem.

Figura 5.1 – Fluxo de interação do usuário com os modos de acesso



Fonte: Do autor(2020)

A partir do fluxo, é possível caracterizar dois modos de acesso as bases de conhecimento descritos, anteriormente, nos requisitos funcionais, são eles:

- Modo de acesso por código QR: também denominado de acesso indireto, é realizado pelo escaneamento do código QR que é associado a um determinado equipamento eletroeletrônico. A partir do escaneamento, o aplicativo consegue receber dados dos equipamento armazenados pelo sistema, sendo eles o nome, local de instalação, data e horário de ultima manutenção e dados de acesso aos repositórios remotos. Com os dados sendo disponibilizados em um formulário, permitindo que os usuários editem e enviem novos dados e submetam as alterações em formato de *ticket* de serviço. Ademais, é permitido ao usuário anexar fotos para submissão ao repositório, disponi-

bilizando, também, a possibilidade de navegar ao repositório que contém a base de conhecimento do equipamento escaneado.

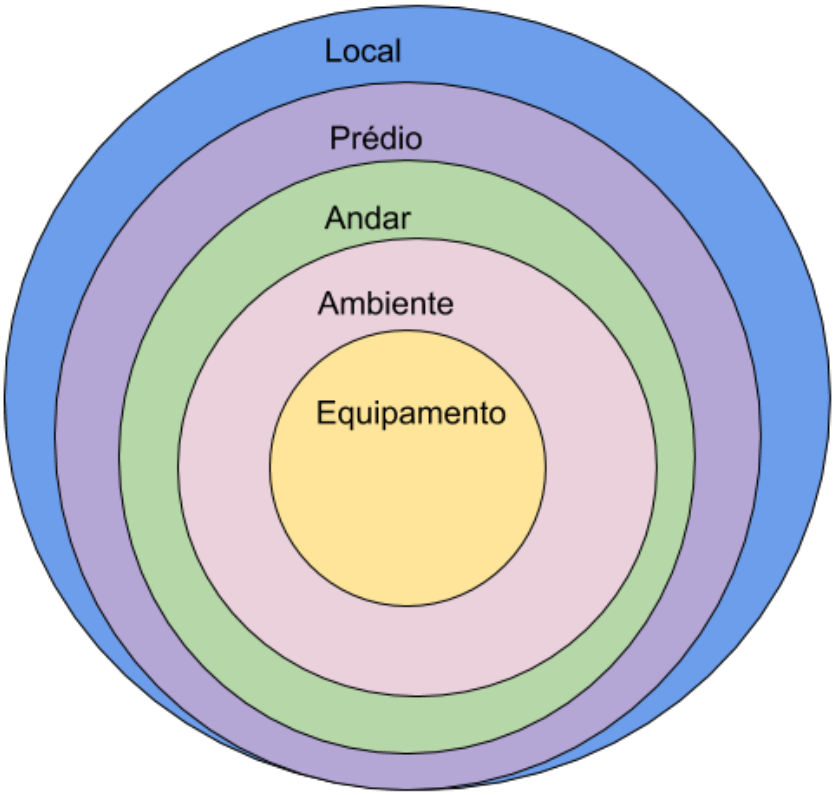
- Modo de acesso indireto: caracteriza o acesso as bases de conhecimento feito sem intermédio do código QR, na qual são listadas alguma das bases de conhecimento que permitem o acesso direto ao usuário. Seu objetivo, logo, é fornecer um acesso imediato e prático ao repositório, despesando o esforço de criar o código QR, associar ao repositório das bases de conhecimento de um equipamento, disponibilizar e escanear o código. Contudo, este modo de acesso é mais limitado, já que impossibilita o aplicativo de buscar os dados do equipamento no sistema.

O fluxo de interação do usuário, contudo, apenas representa uma visão geral sobre os modos de acesso, sendo ainda necessário adicionar mais fatores a cada modo de acesso para se adequar aos requisitos funcionais listados, sendo eles a restrição de acesso as bases de conhecimento e a inserção de uma hierarquia ao acesso.

A restrição de acesso impede que os usuários sem permissão prévia tenham a possibilidade de navegar até as bases de conhecimentos usando o aplicativo. No contexto da organização, esta funcionalidade permite melhor controle, e, consequentemente, mais consistência e segurança as informações contidas nas bases de conhecimento, evitando, por exemplo, o acesso indevido de um funcionário, alocada na área de transmissão de energia, a equipamentos de missão crítica.

Pode-se criar uma organização lógica de vários componentes ou níveis de uma organização com base em sua localização física. Logo, um local pode conter um ou mais prédios, um prédio é constituído de andares que, por sua vez, é composto por ambientes tais como as salas de reunião, de servidores e recepção, com cada ambiente contendo um ou vários equipamentos. Um diagrama que relaciona os níveis de acesso está disponível na figura 5.4

Figura 5.2 – Diagrama dos níveis de acesso



Fonte: Do autor(2020)

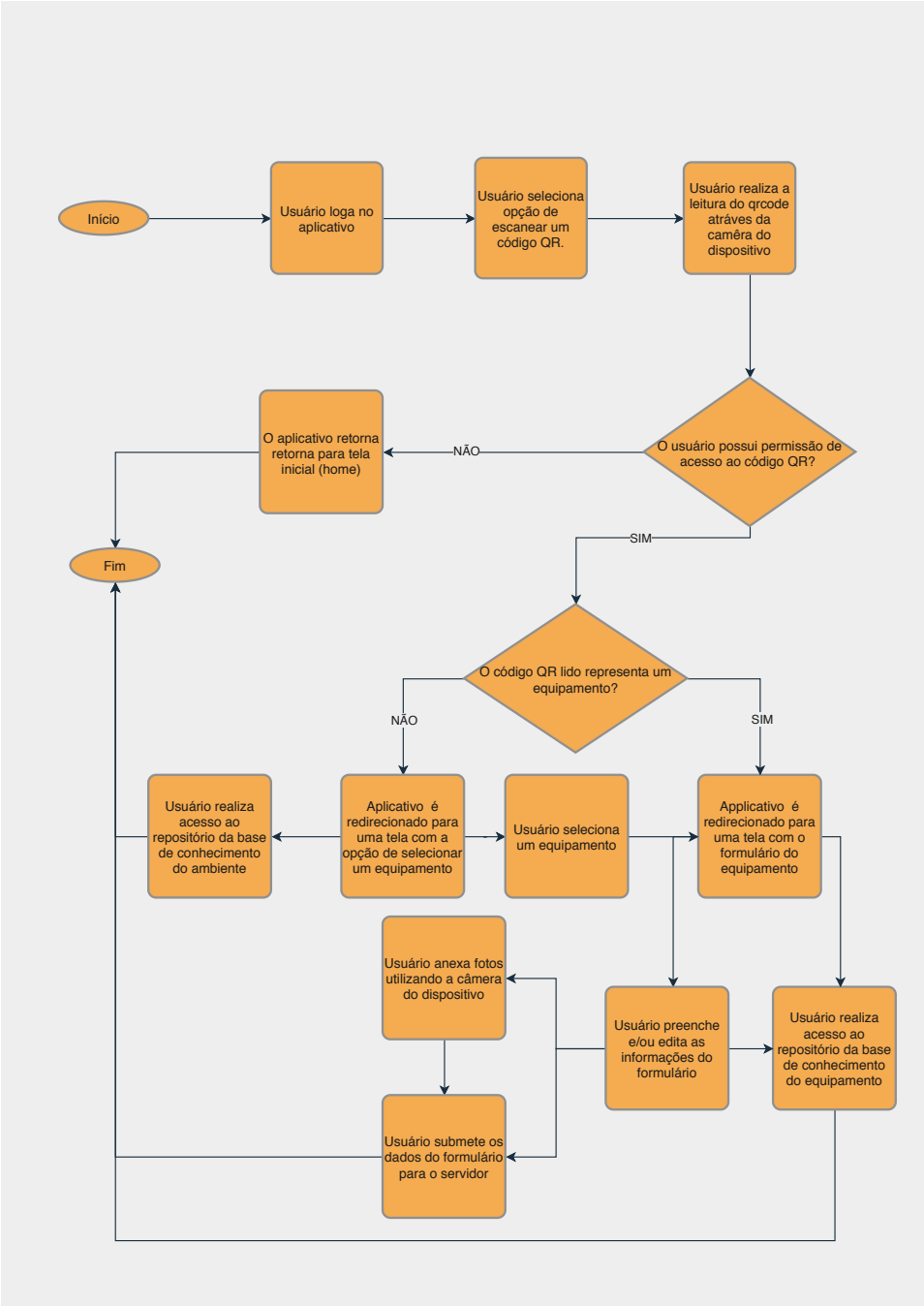
No contexto do trabalho, apenas foi considerado os níveis de ambiente e equipamento, objetivando encurtar o tempo de desenvolvimento do aplicativo, com os níveis superiores podendo ser adicionados futuramente. Com a inserção de uma hierarquia de acesso, o modo de acesso por código QR deve ser modificado de tal forma que um código QR pode também estar relacionado a um ambiente, e consequentemente, à vários equipamentos. Logo, ao escanear um código QR de um ambiente, o usuário deve possuir a opção de selecionar um equipamento

relacionado, ou de navagar até o repositório do ambiente que contêm os arquivos de cada equipamento que está inserido no ambiente.

Já para o modo de acesso direto, os ambientes podem a ser representados como apenas como elementos adicionais listados em conjunto com os equipamentos, com a possibilidade de navegar até o repositório de um determinado ambiente.

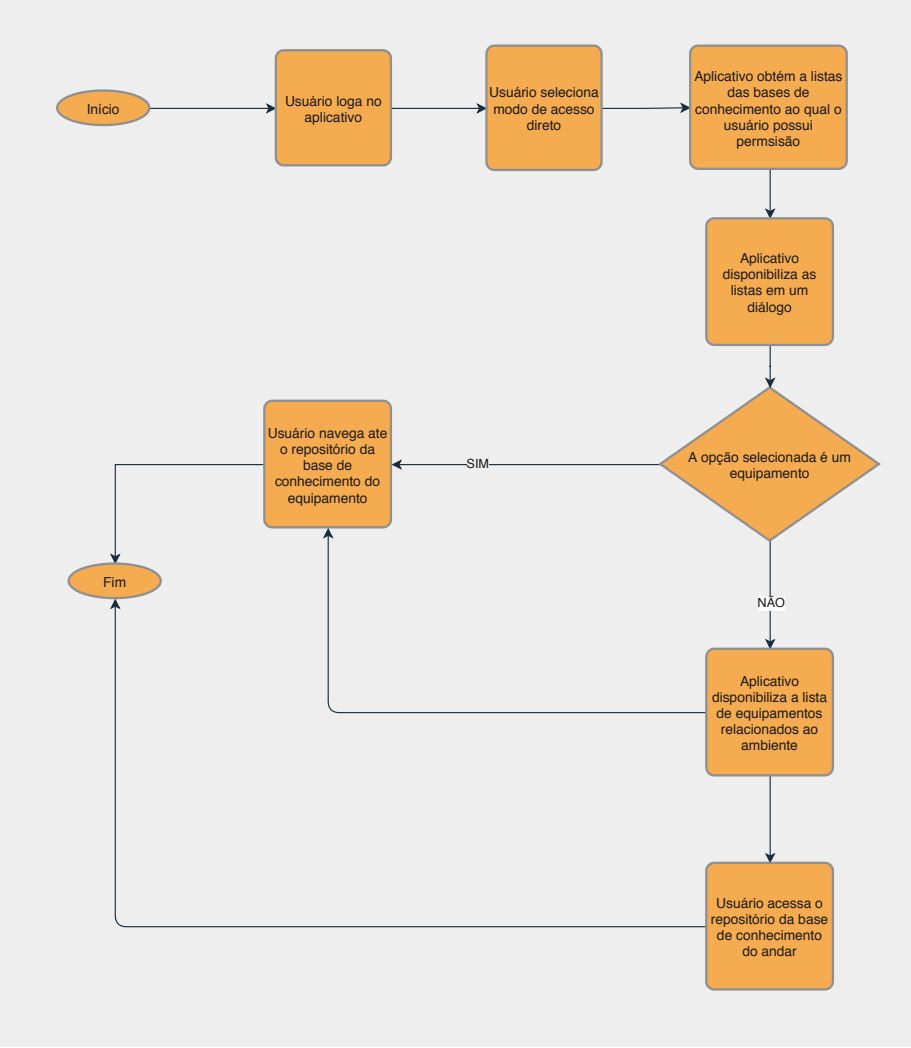
Logo, pode-se refinar o fluxo de interação do usuário em dois diagramas, com cada diagrama descrevendo um modo de acesso incluindo os fatores detalhados anteriormente.

Figura 5.3 – Diagrama do modo de acesso por código QR



Fonte: Do autor(2020)

Figura 5.4 – Diagrama do modo de acesso direto



Fonte: Do autor(2020)

5.2.2 Desenvolvimento das telas iniciais

Após feitas as atividades anteriores, foi iniciado o desenvolvimento das principais telas que irão compor o aplicativo. As telas inicias foram construídas com o *framework* Ionic e com a utilização de apenas dados estáticos, com o propósito de funcionar como um protótipo de alta fidelidade, permitindo a obtenção de

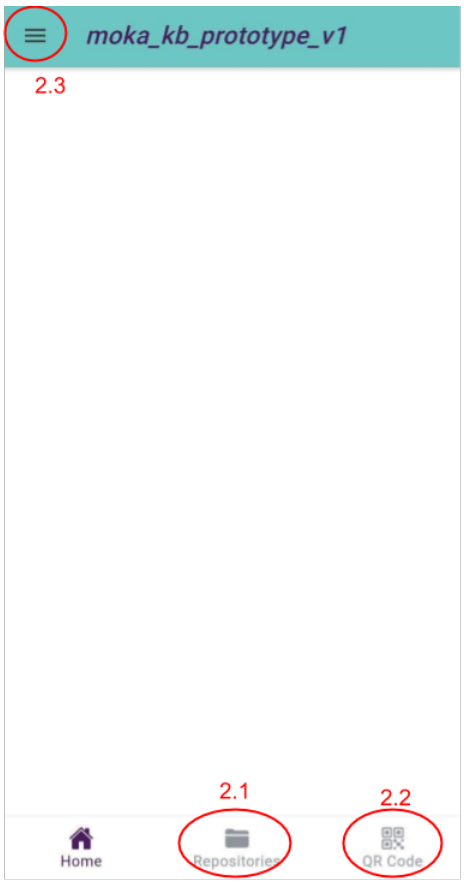
feedback dos usuários-alvo sobre as funcionalidades implementadas. O resultado é demonstrado a seguir:

Figura 5.5 – Tela de login

Fonte: Do autor(2020)

1. Primeira tela disponibilzada para o usuário após inicialização do aplicativo. Neste tela, ocorre o processo da autorização de acesso ao usuário no sistema, a partir dos seus dados de login, ou email, e senha.

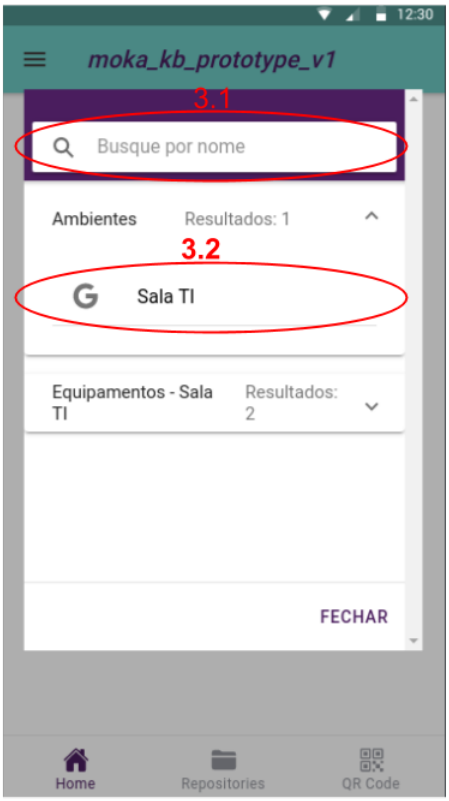
Figura 5.6 – Tela de home



Fonte: Do autor(2020)

2. Tela que procede o login do usuário. Nesta tela é disponibilizado ao usuário as funcionalidades de acesso direto(2.1) ou acesso por qrcode(2.2). Ademais é disponibilizado um menu(2.3),onde o usuário encontra informações do seu perfil, email e login, e o botão de *logout*.

Figura 5.7 – Tela do modo de acesso direto



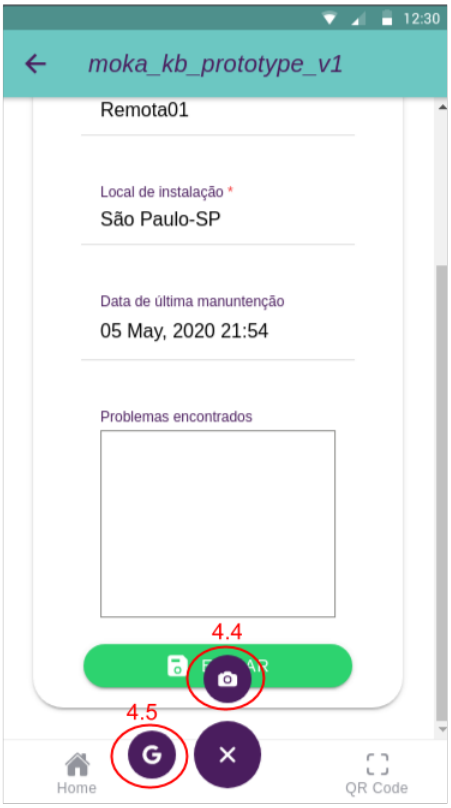
Fonte: Do autor(2020)

3. Tela representada por um dialógo que lista o acesso direto aos repositórios que são permitidas ao usuário. Nesta tela, estão disponíveis um campo de busca(3.1),na qual o usuário pode pesquisar os repositórios por nome, e os elementos clicáveis de acesso aos repositórios (3.2), com cada elemento sendo separado em equipamento ou ambiente.

Figura 5.8 – Tela do modo de acesso por código QR

Fonte: Do autor(2020)

Figura 5.9 – Continuação da tela do modo de acesso por código QR



Fonte: Do autor(2020)

4. Tela exibida após um escaneamento bem-sucedido de um código QR. Neste tela é apresentando um formulário dos dados do equipamentos com os campos editáveis de nome, local de instalação e última manutenção já preenchidos, e uma área de texto destinada para o usuário descrever possíveis problemas encontrados. Ademais, estão disponíveis as funcionalidades de seta de navegação(4.1) e de retorno a tela de escaneamento(4.3), que possuem a mesma função, retorno a tela de home(4.2), anexar fotos(4.4) e acesso aos arquivos do equipamento no Google Drive(4.5). Em caso do código QR escaneado estar relacionado a um determinado am-

Figura 5.10 – Tela do modo de acesso por código QR de ambiente

Fonte: Do autor(2020)

biente, é disponibilizado ao usuário a opção de escolher um equipamento pertencente ao ambiente(4.6) ou de apenas acessar o repositório do ambiente.

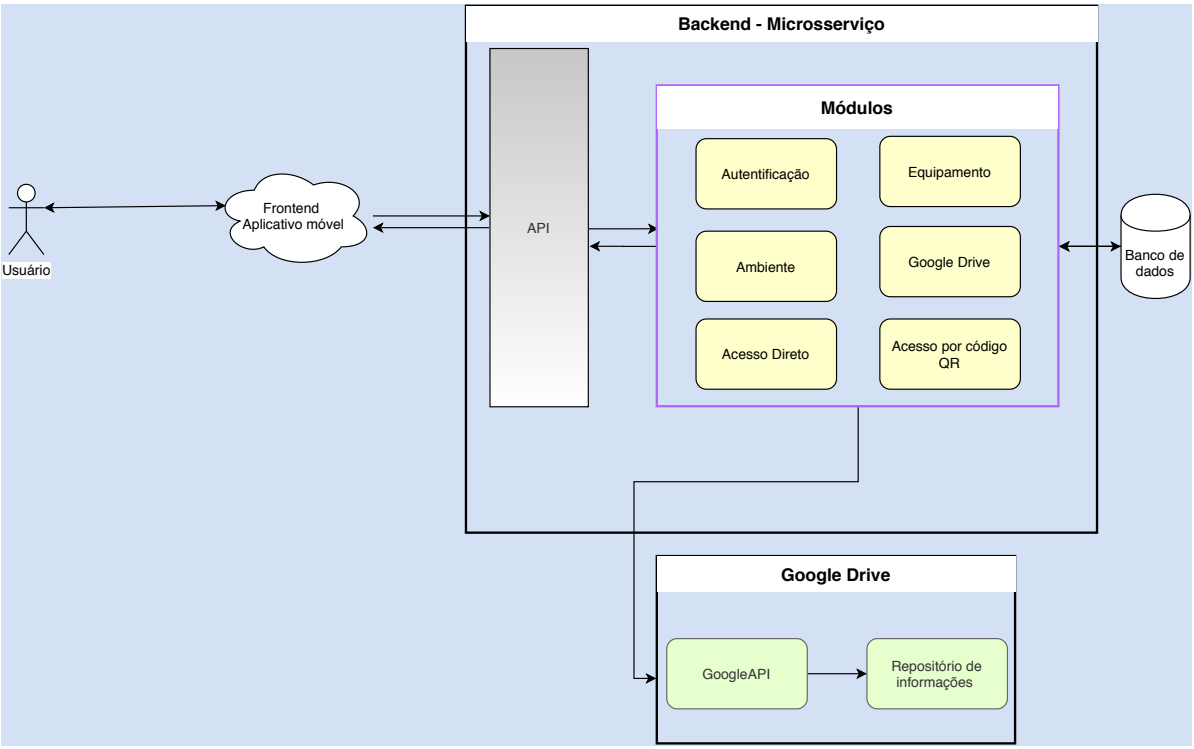
5.3 Implementação do *backend*

Nessa seção é apresentado o desenvolvimento *backend* do sistema.

5.3.1 Visão geral do sistema e módulos

Inicialmente, para o desenvolvimento do *backend* foi construído um digrama com a visão geral do sistema, na qual são descritos, em alto nível, os principais componentes que compõem o *backend* e como se relacionam entre si, ao *frontend* e ao serviço de armazenamento em nuvem Google Drive.

Figura 5.11 – Diagrama geral do sistema



Fonte: Do autor(2020)

Com a decisão de se utilizar uma arquitetura de microsserviço, foi possível estruturar o software em vários componentes menores de funcionamento independente e com domínios definidos de operação, denomina-se esses componentes de serviços ou módulos, que se comunicam com o restante do sistema pela interface de uma API. Abaixo, é descrito cada módulos segundo sua função:

- Módulo de autentificação: responsável por gerenciar a autentificação e carregamento dos dados dos usuários, com a emissão de *tokens* no padrão JWT e na busca pelos credencias de acesso aos repositórios, e autorizar o acesso do usuário as funções do aplicativo, a partir da verificação da tempo de expiração do token, na qual, caso tenha se expirado, obriga ao usuário realizar login novamente,

- Módulo de equipamento: seu funcionamento está ligado a busca e processamento de dados requisitados dos equipamentos, um exemplo de seu uso, está quando o usuário realiza um escaneamento do código QR, na qual o módulos se encarrega de retornar todos os dados formatados que irão compor o formulário de informações do equipamento. Ademais, o módulo também se responsabiliza por tratar os dados recebidos pelos formulários, realizando o processamento e organização nos denominados *tickets* de serviço.

- Módulo de ambiente: de funcionamento similar ao módulo do equipamento, contudo, sua operação é restrita ao contexto dos ambientes. No modo de acesso direto, por exemplo, é o responsável por carregar a listas dos ambientes, e seus dados de acesso, disponíveis ao usuário, além de obter e retornar a lista de equipamentos relacionados ao ambiente.

- Módulo de acesso direto e por Código QR: são módulos responsáveis por realizar o processamento e verificação das restrições e hierarquias de acesso para cada modo de acesso, e, conseguinte, realizar o redirecionamento de fluxo do código para outros os módulos de dados, de equipamento ou ambiente.

- Módulo do Google Drive: é o módulo mais complexo do sistema, será descrito, portanto, em uma seção separada.

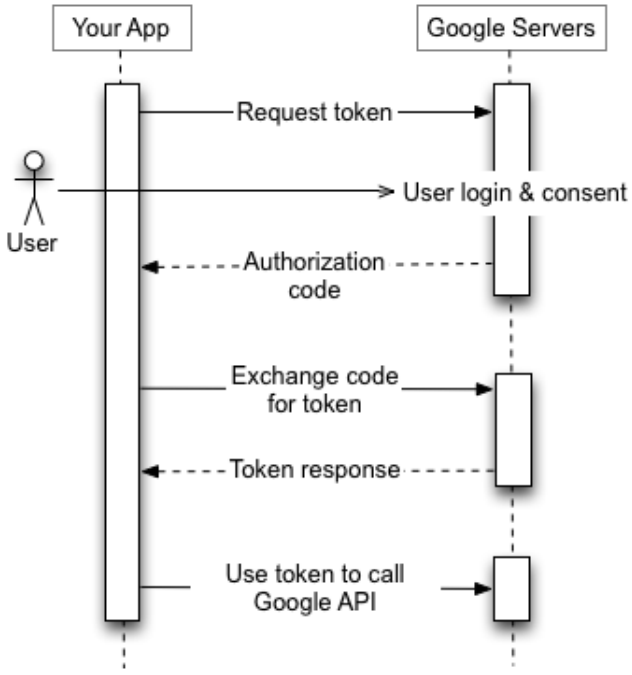
5.3.2 Detalhamento do módulo do Google Drive

O módulo do Google Drive possui a função de executar e gerenciar comunicações remotas do aplicativo com serviço de Drive, através de envio de mensagens HTTP para as APIs disponibilizadas pelo próprio Google, um exemplo do uso deste módulo estaria, portanto, na implementação da funcionalidade de envio de fotos ao repositório de equipamentos. Contudo, para que a aplicação consiga

utilizar a API é necessário obter um *token* de acesso dos servidores de autorização do Google.

Nesse contexto, para a autorização do aplicativo são disponibilizados dois métodos, o emprego do protocolo OAuth2.0 ou da denominada conta serviço. O uso do protocolo OAuth2.0 se baseia em situações onde a aplicação deseja acessar dados específicos de um determinado usuário, sendo necessário à aplicação obter o consenso de uso da Conta Google do usuário. Para aplicativos móveis, o processo de obtenção do consenso é desenvolvido pela utilização do componente padrão *Google Sign In*, na qual é requisitado o login do usuário em sua Conta Google, e, em seguida, a permissão de acesso aos serviços solicitados, também denominado de escopos de acesso, pelo aplicativo.

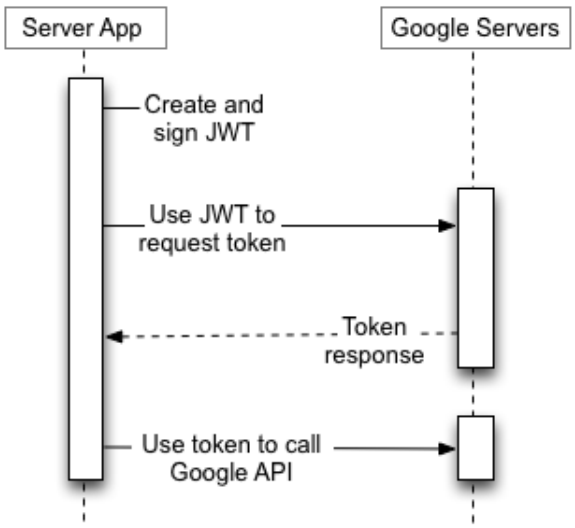
Figura 5.12 – Diagrama de sequência da emprego do protocolo OAuth2.0



Fonte: (GOOGLE, 2020)

Enquanto que as contas serviço são contas próprias da aplicação utilizados para acessar dados baseados em um projeto, logo, sem a necessidade do aplicativo de utilizar contas individuas dos usuários e obter o consenso deste para se comunicar com o Google API.

Figura 5.13 – Diagrama de sequência do emprego da conta serviço



Fonte: (GOOGLE, 2020)

Para o desenvolvimento do sistema, foi escolhido o método de conta serviços, já que permite conter toda a lógica de autorização no lado do servidor, dispensando introduzir mais uma etapa de autorização no aplicativo para obtenção do consenso dos usuários, o que compromete severamente a experiência do usuário no aplicativo.

5.4 Integração e desenvolvimento final do aplicativo

A etapa final do desenvolvimento do sistema compreende todo o processo de integração dos componentes *frontend*, representado pelo aplicativo móvel, e *backend*, representado pelo microsserviço e o banco de dados. No geral, a etapa

de integração foi simples e direta, devido a escolha inicial por utilizar ferramentas como o *framework* Ionic e o NodeJs que reduziram o esforço de desenvolvimento e facilitaram o processo de integração.

Com a integração, o aplicativo passou a utilizar dados totalmente dinâmicos em detrimento aos dados estáticos, logo as funcionalidades implementadas tiverem de ser revistas e aprimoradas em relação a versão inicial apresentada, o que ocasionou, na maioria dos casos, em apenas pequenas mudanças no design do aplicativo. Entretanto, algumas mudanças foram bastante significativas e impuseram desafios de desenvolvimento. Nessa seção, são apresentadas alguns dos problemas que surgiram durante o processo de integração e como eles foram resolvidos.

5.4.1 Limitações do Google Drive API

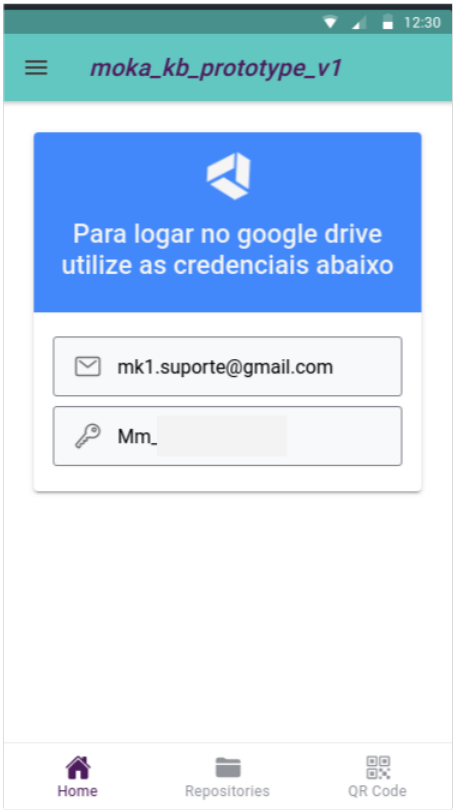
A utilização das contas serviço permitiu que a aplicação acesse e execute ações nos repositórios das bases de conhecimento sem ter de fazer uso da Conta Google do usuário. Contudo, a funcionalidade que determina ao usuário navegar até as bases de conhecimento através da UI do Google Drive não pode ser satisfeita com o uso de contas serviço.

Para acessar a base de conhecimento através da própria IU do Google Drive, requer que a aplicação redirecione o usuário para o browser com o endereço URL do diretório da base de conhecimento selecionado, e que o usuário logue na IU do Drive em uma Conta Google válida e que possua permissão de acesso ao diretório.

Entretanto, é prejudicial a aplicação permitir que os usuários loguem em contas pessoais, pois são maiores os riscos a segurança e integridade das bases de conhecimento e diminui o controle da organização sobre o acesso aos repositórios. Uma solução discutida estaria em criar uma Conta Google para o aplicativo, algo similar a conta serviço, que seria autentificada através da API do Drive, impedindo que usuários acessem o repositório com uma conta pessoal.

Contudo, foi descoberto uma limitação da API do Drive, o Google não permite que aplicações de terceiros realizem o processo de autentificação para acesso a seus serviços. Logo, uma outra solução discutida foi atrelar a cada usuário uma Conta Google mantida pela organização, na qual espera-se que o usuário insira as informação da conta atrelada quando requisitadas pelo Drive. Para auxiliar neste processo, a tela de Home foi reformulado para disponibilizar as informações da Conta Google a ser utilizada pelo usuário.

Figura 5.14 – Tela Home aprimorada



Fonte: do autor (2020)

Entretanto a solução adota não é a ideal, já que a aplicação não possui controle sobre a Conta Google que será utilizada pelo usuário. Sendo assim, uma solução melhor seria uma adoção de outro serviço de armazenamento em nuvem no futuro.

5.4.2 Lentidão no envio das fotos

Com a integração, a funcionalidade de envio de fotos pôde ser implementada no aplicativo, além de submeter um formulário de um dado equipamento, cada usuário poderia anexar um número arbitrário de fotos tiradas da câmera do seu dispositivo, com as fotos sendo submetidas em pastas, padronizadas com o nome de fotos, nos repositórios dos equipamento escaneados. Para enviar os arquivos das imagens por requisição do tipo HTTP POST ao microsserviço, cada imagem foi codificada em Base64 e anexada ao corpo da requisição.

Contudo, o resultado da implementação não foi satisfatória, uma foto anexada levava de 1 a 3 minutos para serem inseridas no Drive, duas a quatro fotos levava até 10 minutos no total. Foi percebido que o gargalho estava na aplicação, que ocupava cerca de 60% do tempo total de submissão das fotos, devido as longas cadeias de caracteres geradas pela codificação da base64 que faziam as mensagens HTTP ficarem pesadas e mais custosas de se processar, aliado, ainda, ao algoritmo ineficiente de decodificação da base64.

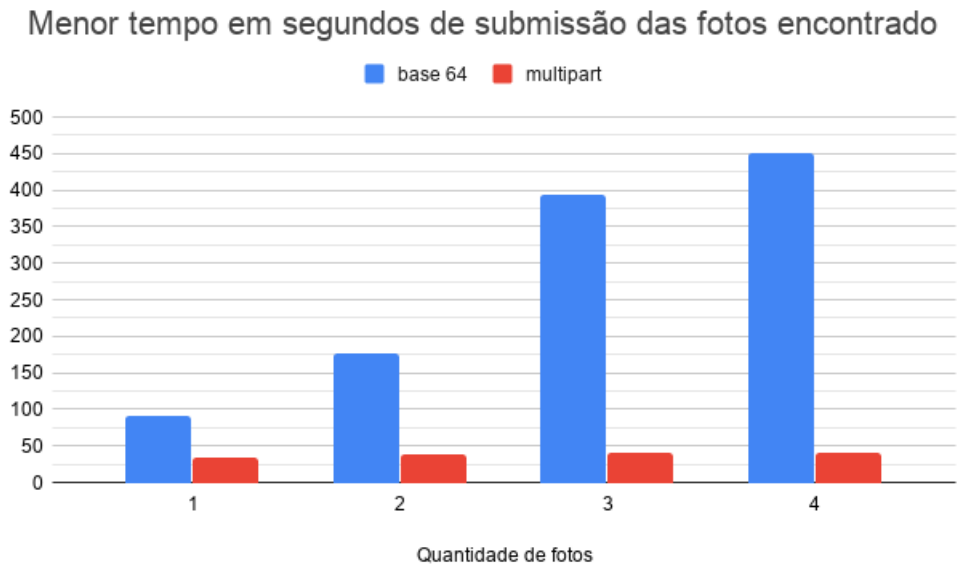
Logo, para solucionar esse problema, foi adotado um limite máximo de quatro fotos anexadas por submissão, e passou a ser utilizado requisições HTTP com o tipo de conteúdo setado como *multipart/form-data*, que possibilita o envio de conteúdo binário diretamente nas requisições HTTP, e, logo, descartado a codificação das fotos por base64. Com a aplicação da solução, houve uma redução considerável do tempo de envio a fotos, como demonstrado na tabela 5.3 e no gráfico 5.15, onde os resultados foram obtidos a partir dos menores tempos arrendondados de submissão das fotos, em segundos, em um total de 5 repetições para cada método.

Tabela 5.3 – Resultados dos tempos de submissão das fotos ao Drive

Quantidade de fotos	Base64(segundos)	Multipart(segundos)
1	90	35
2	176	38
3	393	40
4	451	40

Fonte: Do autor(2020)

Figura 5.15 – Gráfico de comparação dos resultados obtidos



Fonte: do autor (2020)

6 CONCLUSÃO

O presente relatório teve como objetivo apresentar a experiência profissional vivenciada na empresa Lcs Link Engenharia, através da descrição das atividades de desenvolvimento de um aplicação móvel como produto de utilização interna a empresa. A esta aplicação foi definida um conjunto de funcionalidades de acordo com as necessidades da organização em melhorar a organização das bases de conhecimentos dos seus equipamentos eletroeletrônicos, tornando seu acesso acesso mais rápido e prático.

Durante o desenvolvimento da aplicação, foram aplicados e aprimorados diferentes conhecimentos acadêmicos, como os adquiridos pela disciplina de Engenharia de Software, que permitiu modelar fluxogramas. entender e desenvolver a arquitetura do software e metodologias de desenvolvimento. Somado a isso, houve a aplicação de conhecimentos da disciplina de Banco de Dados, no desenvolvimento e gerenciamento dos banco de dados do sistema, programação Web, na utilização de tecnologias de desenvolvimento web e Redes de Computadores, na aplicação do protocolo HTTP.

Ao fim do desenvolvimento, foi obtido um aplicativo móvel para a plataforma Android que apresentava como principais funcionalidades os seguintes pontos:

- O aplicativo pode utilizar códigos QR para acessar dados dos equipamentos.
- O usuário pode preencher e submeter um formulário com novos dados do equipamento, criando um novo *ticket* de serviço.
- O aplicativo fornece meios de acesso mais rápidos aos repositórios.
- O usuário pode enviar fotos do equipamento para o sistema, que ficarão armazenados no repositório no Drive.

- O usuário pode navegar ate os repositórios das bases de conhecimento utilizando a IU do Drive.

No geral, o aplicativo desenvolvido conseguiu atender bem as demandas e expectativas iniciais da empresa e proveu um bom potencial para se tornar um produto de software comercializável. Contudo, ainda é necessário continuar o processo de desenvolvimento, para que o sistema consiga ser utilizado de maneira plena.

Ademais, o desenvolvimento do sistema ocasionou em um considerável amadurecimento profissional do autor, pois requisitou deste um vasto conhecimento sobre diferentes tipos de tecnologias que compõem a área de desenvolvimento de software, envolvendo atividades como desenvolvimento e implantação de serviços, configuração e gerenciamento de servidores e banco de dados, construção e distribuição de aplicativos móveis, manipulação de APIs externas, implementação de algoritmos de autentificação e autorização de usuários, etc. Em resumo, o projeto representou um desafio único sobre a trajetória profissional do autor, resultando no processo de busca de novas informações, obtenção de novos conhecimentos e aperfeiçoamento de habilidades, com muito destes conhecimentos e habilidades, contudo, não sendo adquiridos durante a vivência acadêmica.

6.0.1 Trabalhos Futuros

Para o desenvolvimento futuro, está a necessidade de implementação de um sistema web que consiga criar e gerenciar os dados de cadastro dos usuários do aplicativo, equipamentos e ambientes, além de possibilitar criar e relacionar os códigos QR e dados para acesso direto. Atualmente, todos esses dados se encontram pré-registrados no banco de dados do sistema e não há formas de um colaborador de inserir os dados sem ter que fazer operações diretas no banco.

Adicionalmente, estaria o trabalho de integração com outros serviços de armazenamento de arquivos em nuvem, tais como o Microsoft OneDrive e o Lark,

objetivando contornar a limitação imposta pela API do Drive e aumentar o valor de uso da aplicativo. Além disso, os dados do equipamento salvos na aplicação, através dos *tickets* de serviço e fotos anexadas dos usuários, podem ser utilizados para alimentar outros sistemas da organização, onde poderão ser processados para determinação de informações úteis a empresa.

REFERÊNCIAS

ABRAMOVA, V.; BERNARDINO, J. Nosql databases: Mongodb vs cassandra. In: **Proceedings of the International C* Conference on Computer Science and Software Engineering**. New York, NY, USA: Association for Computing Machinery, 2013. p. 14–22. ISBN 9781450319768. Disponível em: <https://doi.org/10.1145/2494444.2494447>.

Balalaie, A.; Heydarnoori, A.; Jamshidi, P. Microservices architecture enables devops: Migration to a cloud-native architecture. **IEEE Software**, v. 33, n. 3, p. 42–52, 2016.

GOOGLE. **Using OAuth 2.0 to Access Google APIs**. 2020. Disponível em: <https://developers.google.com/identity/protocols/oauth2>. Acesso em: 5 Aug. 2016.

IONIC. **Ionic - Cross-Platform Mobile App Development**. 2020. Disponível em: <https://ionicframework.com>. Acesso em: 26 Jul. 2016.

KIESEBERG, P. et al. Qr code security. In: **Proceedings of the 8th International Conference on Advances in Mobile Computing and Multimedia**. New York, NY, USA: Association for Computing Machinery, 2010. (MoMM '10), p. 430–435. ISBN 9781450304405. Disponível em: <https://doi.org/10.1145/1971519.1971593>.

MALAV, B. Microservices vs Monolithic architecture - Hash#Include - Medium. **Medium**, Hash#Include, Jun 2018. Disponível em: <https://medium.com/startlovingyourself/microservices-vs-monolithic-architecture-c8df91f16bb4>.

PEREZ, A. Etqr: A qr code generator and scanner written using salesforce's lightning web components. **Medium**, Medium, Dec 2019. Disponível em: <https://medium.com/@ElToroIT/etqr-code-scanner-47ef5ea67d90>.

PRESSMAN, R. S. **Engenharia de Software: uma abordagem profssional**. 7. ed. Porto Alegre, RS: Amgh Editora, 2011. ISBN 9788563308337.

REGINATO, C. E. R.; GRACIOLI, O. D. Gerenciamento estratégico da informação por meio da utilização da inteligência competitiva e da gestão do conhecimento: um estudo aplicado à industria moveleira do rs. **Gestão Produção**, Scielo Brasil, v. 19, p. 705 – 716, 2012. ISSN 0104-530X. Disponível em: <http://www.scielo.br/scielo.php?script=sci_arttext&pid=S0104-530X2012000400004&nrm=iso>.

SOMMERVILLE, I. **Engenharia de software**. 8. ed. São Paulo: Pearson Addison Wesley, 2007. ISBN 9788588639287.

Tilkov, S.; Vinoski, S. Node.js: Using javascript to build high-performance network programs. **IEEE Internet Computing**, v. 14, n. 6, p. 80–83, 2010.