Centro Estadual De Educação Tecnológica Paula Souza

Etec Zona Leste

Desenvolvimento De Sistemas

Ana Beatriz Martins Batista

Bruno Santos Portugal

Eliel Andrade Matos Godoy

B.A.E GUARD: IOT para detecção de incêndio

São Paulo

2023

Ana Beatriz Martins Batista

Bruno Santos Portugal

Eliel Andrade Matos Godoy

B.A.E GUARD: IOT para detecção de incêndio

Trabalho de conclusão de curso apresentado ao Curso Técnico em Desenvolvimento de Sistemas da ETEC Zona Leste, orientado pelo Prof. Esp. Jeferson Roberto de Lima, como requisito parcial para obtenção do título de técnico em Desenvolvimento de Sistemas.

São Paulo

2023

Com gratidão, dedicamos este Trabalho de Conclusão de Curso aos nossos amados familiares, que sempre nos acompanharam incansavelmente ao longo desta jornada acadêmica.

Agradecimentos

Aos professores, pelas correções e ensinamentos que nos permitiram apresentar um melhor desempenho em nosso processo de formação profissional ao longo do curso.

“Assim, lembre-se de olhar para as estrelas, não para os próprios pés. Tente compreender o que vê e questione o que faz o universo existir. Seja curioso. E por mais que a vida pareça difícil, sempre há algo que você pode e consegue fazer. Nunca desista. Deixe sua imaginação correr solta. Molde o futuro”.

Stephen Hawking

Resumo

Este estudo aborda a problemática do aumento abundante da ocorrência de incêndios ao longo do tempo, causada principalmente pela falta de prevenção e combate de incêndio, em parte devido ao alto custo dos sistemas contra incêndio existentes. Diante desse cenário, busca-se reduzir os danos causados por incêndios estruturais por meio do desenvolvimento de um sistema de detecção de princípios de incêndio em residências, com ênfase na acessibilidade financeira. O sistema proposto é baseado na tecnologia Internet das Coisas (IoT) e permite aos usuários evitar acidentes ao fornecer informações detalhadas sobre o ambiente monitorado. Para alcançar esse objetivo, foram realizados estudos sobre sistemas embarcados e ambiente mobile. A pesquisa envolveu a análise de requisitos, elaboração de diagramas e prototipação de interfaces gráficas e esquemas elétricos. Além disso, foi desenvolvida uma aplicação para dispositivos móveis, que integra sensores e uma câmera com o sistema de alarme. O sistema foi testado em condições reais para garantir sua eficiência e conformidade com os regulamentos de segurança. Espera-se que esse sistema contribua para a rápida detecção de incêndios e a mitigação de seus efeitos, proporcionando maior segurança residencial.

Palavras-chave: Internet das Coisas. Detecção de incêndio. Sistemas embarcados. Aplicativo mobile. Segurança residencial.

Abstract

This study addresses the significant increase in the occurrence of fires over time, mainly due to the lack of fire prevention and firefighting, partly caused by the high cost of existing fire safety systems. In this context, the aim is to reduce the damages caused by structural fires by developing a low-cost system for detecting fire incidents in residential buildings, with a focus on financial accessibility. The proposed system is based on the Internet of Things (IoT) technology, enabling users to prevent accidents by providing detailed information about the monitored environment. To achieve this goal, studies on embedded systems and mobile environments were conducted. The research involved requirements analysis, diagram development, and prototyping of graphical interfaces and electrical schemes. Additionally, a mobile application was developed to integrate sensors with the alarm system. The system was tested under realistic conditions to ensure its efficiency and compliance with safety regulations. It is expected that this system will contribute to the early detection of fires and the mitigation of their effects, enhancing residential safety.

Keywords: Internet of Things. Fire detection. Embedded systems. Mobile application. Residential safety.

Lista de ilustrações

Lista de Tabelas

Lista de abreviaturas e siglas

Android Open System Platform (AOSP)

Aplication Programing Platform (API)

Android Runtime (ART)

C with Classes (C++)

Data Access Object (DAO)

ESP32 IoT Development Framework (ESP-IDF)

Real-time operating system for microcontrollers (FreeRTOS)

Hardware Abstraction Layer (HAL)

Integrated Development Environment (IDE)

Internet of Things (IoT)

Java Virtual Machine (JVM)

Limited Liability Company (LLC)

Long Range (LoRA)

National Fire Protection Association (NFPA)

Native Development Kit (NDK)

Organic Light-Emitting Diode (OLED)

Regulamento de Segurança contra Incêndios das Edificações (RSCI)

User Interface (UI)

Unified Modeling Language (UML)

Universal Serial Bus (USB)

Extensible Markup Language (XML)

Lista de símbolos

± Sinal de mais ou menos, que representa tanto uma adição quanto subtração.

® Símbolo que representa uma marca registrada ou protegida por direitos autorais (copyright)

SUMÁRIO

1. INTRODUÇÃO

Os incêndios em residências representam um risco significativo para a segurança e a integridade das pessoas, além de causarem danos materiais e econômicos consideráveis (National Fire Protection Association, 2023). Diante dessa problemática, o presente estudo tem como objetivo desenvolver um sistema de detecção de princípios de incêndio em residências, utilizando Internet das Coisas (IoT) e um aplicativo mobile para controle e monitoramento.

O aumento do número de incêndios estruturais nos últimos anos tem despertado a necessidade de soluções mais acessíveis e eficazes para a detecção precoce desses eventos. Dados do Corpo de Bombeiros de São Paulo (2022): 6.291 incêndios em edificações, 2.161 em áreas reguladas pelo RSCI, danos de 265.000 m². Nesse contexto, a hipótese levantada é que um dispositivo de detecção de princípios de incêndio, aliado a um aplicativo mobile, pode auxiliar os moradores na detecção precoce e no combate aos incêndios, proporcionando uma resposta mais rápida e precisa, independentemente da presença física no local. Além disso, a disponibilidade de informações detalhadas sobre o ambiente monitorado contribui para a tomada de decisões preventivas e mitigação de danos.

Este trabalho inclui a pesquisa e seleção dos melhores sensores de gás, fumaça e temperatura para a detecção de incêndios, a elaboração de requisitos e diagramas de casos de uso, estado, sequência e atividades, a prototipação de interfaces gráficas e esquemas elétricos, o desenvolvimento de uma aplicação mobile, a integração dos sensores com o sistema de alarme e a garantia da conformidade com os regulamentos de segurança.

A abordagem adotada envolve a realização de estudos bibliográficos e documentais para embasar o projeto, a testagem para verificar a confiabilidade do dispositivo e do aplicativo desenvolvidos, e a utilização de tecnologias como Kotlin, JetpackCompose, C++, Firebase e outras tecnologias.

A motivação para este trabalho está na necessidade de reduzir os danos causados pelos incêndios residenciais, que têm apresentado um crescimento significativo, principalmente durante a quarentena. A busca por soluções mais acessíveis e eficientes se faz necessária diante do alto custo dos sistemas tradicionais de detecção de incêndio. Além disso, a capacidade de fornecer informações precisas sobre a

ocorrência de incêndios possibilita uma resposta mais rápida e efetiva, contribuindo para a segurança dos moradores.

Ao final deste trabalho, espera-se obter um sistema de detecção de princípios de incêndio em residências que seja economicamente viável e que proporcione maior acesso a medidas preventivas e monitoramento. O sistema desenvolvido tem o potencial de reduzir os danos causados pelos incêndios, bem como incentivar a adoção de medidas de segurança e ações de combate eficazes.

2. REFERENCIAL TEÓRICO

Nesta seção, serão apresentados os principais conceitos e teorias relacionados à Internet das Coisas (IoT) e à detecção e prevenção de incêndios, que constituem a base teórica deste estudo. Serão abordadas obras e pesquisas relevantes, além de autores renomados na área, a fim de fornecer embasamento teórico consistente para o desenvolvimento do projeto.

2.1. Incêndios

Em 2022, o Corpo de Bombeiros de São Paulo atendeu a 6.291 casos de incêndios em edificações, sendo que 2.161 ocorreram em áreas reguladas pelo RSCI e atingiram 265.246 metros quadrados. Não há informações sobre a área afetada em edificações não reguladas pelo RSCI. (CORPO DE BOMBEIROS DO ESTADO DE SÃO PAULO, 2022)

A quarentena provocou um aumento de 60% nos incêndios em 2020, com São Paulo registrando 2.560 ocorrências em março de 2019 e 4.089 no mesmo período de 2020. Em abril, foram 2.589 ocorrências, um acréscimo de 18%. (BBC NEWS BRASIL – FELIPE SOUZA, 2020). Segundo a Cronoshare, as centrais de incêndios convencionais têm um custo médio entre R$1.500 e R$5.800. Já os dispositivos para sistemas mais avançados podem ter preços acima dessa faixa. (Cronoshare - Isabella Barbosa, 2023).

Em outubro de 2022, as ocorrências de incêndios estruturais aumentaram. O Instituto Sprinkler registrou 165 casos, um aumento de 8,6% em relação ao mesmo mês do ano anterior, que teve 152 notícias. (Instituto Sprinkler Brasil, 2022). Neste cenário, fica evidente o crescente aumento de casos de incêndio oriundos da ausência de um projeto de prevenção e combate a incêndio adequado, geralmente pelo alto custo dos mesmos. Abordaremos um tema relevante visando diminuir os danos causados por casos de incêndio em diferentes situações, além de reduzir o custo de um sistema de detecção de princípios de incêndios.

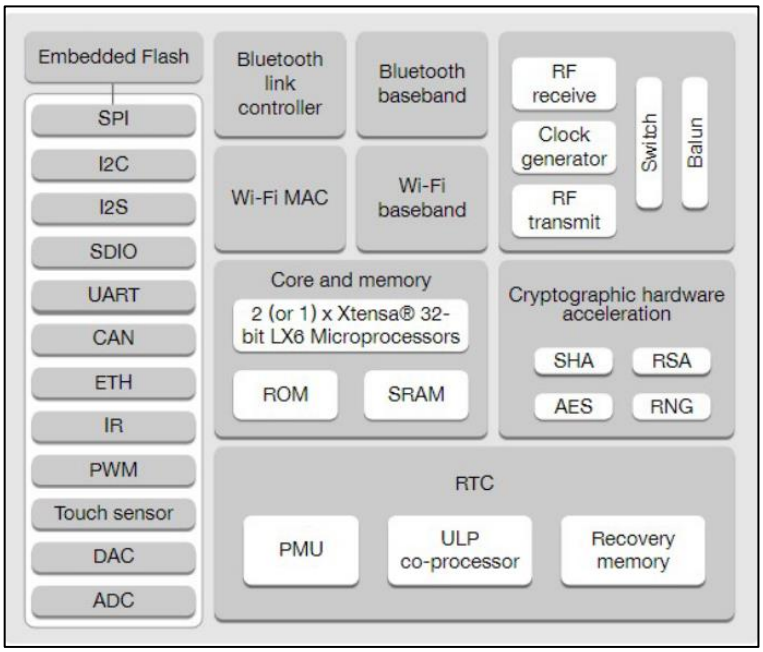
2.2. Materiais Utilizados

Confira os dispositivos físicos que foram utilizados no desenvolvimento do B. A. E. Guard:

2.2.1. ESP32

No contexto dos microcontroladores, há uma variedade de dispositivos que podem ser mencionados. Entre eles, um que tem ganhado destaque recentemente é o ESP32. Esse microcontrolador é frequentemente utilizado em implementações que requerem transferência de dados, tanto remotamente quanto localmente, além de oferecer um bom poder de processamento. Uma das vantagens do ESP32 é sua versatilidade, pois pode ser usado tanto como uma plataforma para prototipagem rápida quanto para aplicação em produtos (Bertoleti, 2019). O ESP32 se destaca por incluir em seu encapsulamento módulos que oferecem suporte a diversos protocolos de comunicação, como I²C, I²S, Wi-Fi, Bluetooth, UART, SPI, entre outros. Essa ampla gama de módulos é uma vantagem em relação a outros microcontroladores disponíveis no mercado, pois não é necessário adaptar módulos externos para essas funcionalidades específicas (Bertoleti, 2019)

Figura 1 - Diagrama de Blocos ESP32

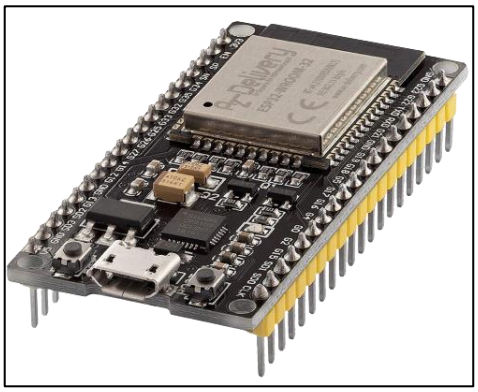


Fonte: (ESPRESSIF, 2019)

Além dos módulos de comunicação internos, o ESP32 possui outros módulos incorporados em seu encapsulamento, que facilitam seu uso e oferecem funcionalidades adicionais. Por exemplo, há módulos de sensor de toque, sensor de temperatura (em alguns modelos específicos) e relógio em tempo real. Outros módulos presentes nesse encapsulamento podem ser observados no diagrama de blocos do microcontrolador mostrado na Figura tal.

Quanto à programação do microcontrolador, existem duas formas diferentes de fazê-lo: usando o SDK oficial da Espressif com a linguagem de programação C ou através do software Arduino IDE com as linguagens C ou C++. Devido à facilidade de uso, a segunda opção acaba sendo a mais popular entre a comunidade de desenvolvedores independentes e para fins didáticos. (Bertoleti, 2019)

Figura 2 - ESP32



Fonte: (Robocore, 2023)

Informações Técnicas:

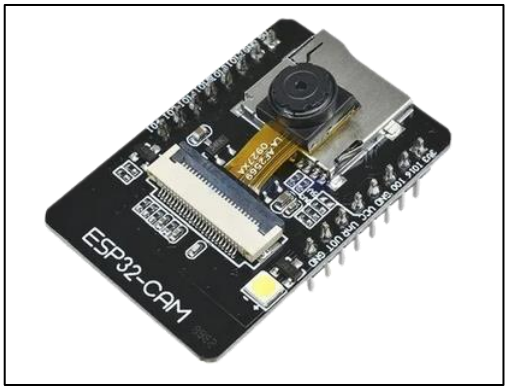
- Processador: Xtensa® Dual-Core 32-bit LX6
- Memória Flash programável: 4 MB
- Memória RAM: 520 KBytes
- Memória ROM: 448 KBytes
- Clock máximo: 240 MHz
- Pinos Digitais GPIO: 25 (todos com PWM)
  - Os pinos GPIO0, GPIO2, GPIO5, GPIO12 e GPIO15 são usados para funções internas, se respeitados alguns cuidados.
- Resolução do PWM: até 16 bits (ajustável via código)
- Wireless 802.11 b/g/n - 2.4GHz (antena integrada)
- Modos de operação: Access Point, Estação, Access Point e Estação

- Bluetooth Low Energy padrão 4.2 integrado
- Tensão de alimentação externa: 4,5 V a 9 V (o módulo possui regulador integrado para 3,3 V)

2.2.2. ESP32Cam

O ESP32 CAM é uma versão especial da placa de desenvolvimento baseada no módulo ESP32 WiFi, mostrando-se um módulo eletrônico altamente tecnológico desenvolvido especialmente para conectar projetos robóticos *maker* ou de automação residencial à Internet com maior facilidade e baixo custo.

Figura 3 - ESP32 Com Câmera



Fonte: (Robocore, 2023)

Informações Técnicas:

- Modelo: ESP32-CAM;
- Tensão de operação: 5V;
- CPU: Xtensa® Dual-Core 32-bit LX6;
- ROM: 448 Kbytes;
- RAM: 520 Kbytes SRAM;
- Flash: 4 MB PSRAM;
- Resolução da foto: 2 Megapixels;
- Clock máximo: 240MHz;
- Wireless padrão 802.11 b/g/n;
- Conexão: Wifi 2.4Ghz (máximo de 150 Mbps);
- Wi-Fi Direct (P2P), P2P Discovery, P2P Group Owner mode e P2P Power

2.2.3. Buzzer

O Buzzer Ativo 5V é um pequeno alto-falante destinado a emitir sinais sonoros a partir do oferecimento de energia DC ao módulo, não variando a frequência de emissão.

Figura 4 - Buzzer Ativa 5v

Fonte: (Robocore, 2023)

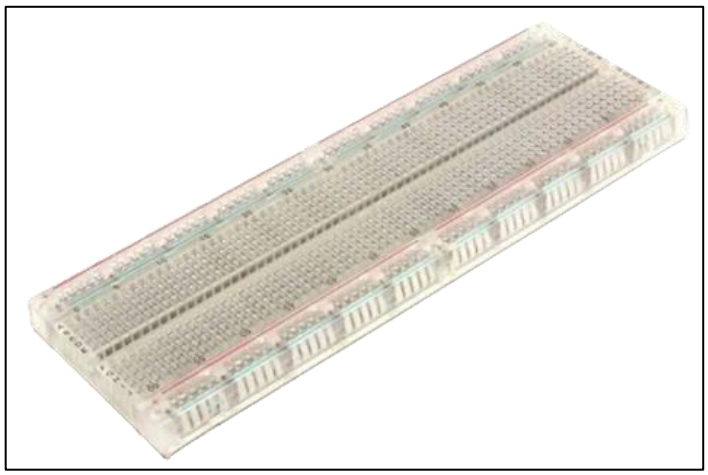
Informações Técnicas:

- Buzzer do tipo ativo (possui oscilador interno)
- Tensão de operação: 3.3V a 5V
- Furo para fixação
- Dimensões: 30 x 13 x 10 mm
- Pinagem: VCC - GND – SINAL

2.2.4. Protoboard

Uma *Protoboard/ Breadboard,* também conhecida como Placa de Ensaio ou Matriz de Contato**,** é uma placa com furos e conexões condutoras para montagem de circuitos elétricos experimentais.

Figura 5 - Protoboard



Fonte: (Robocore, 2023)

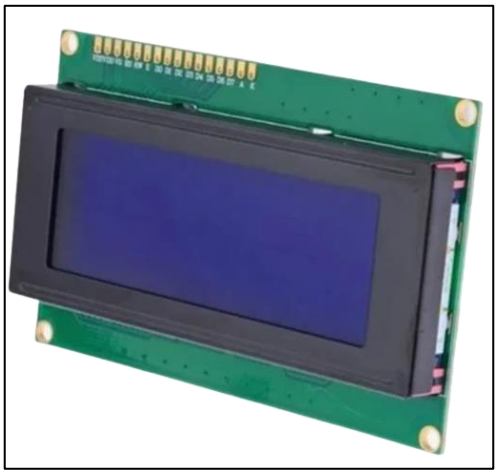
Informações Técnicas:

- Quantidade de contatos: 830;
- Dimensões (CxLxA): 16,5 x 5,5 x 1cm;
- Peso: 83g.

2.2.5. Display LCD 20x4

O Display LCD 20x4 é um equipamento com fundo azul que possui capacidade de suportar a exibição de até 20 caracteres por linha em uma tela de 4 linhas, sendo especialmente indicado para aqueles que desenvolvem projetos com ESP32.

Figura 6 - Display LCD 20x4



Fonte: (Robocore, 2023)

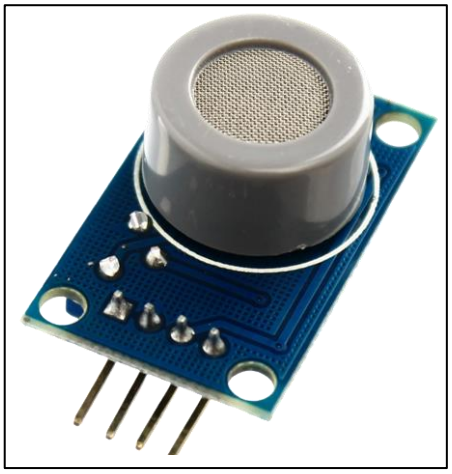
Informações Técnicas:

- Tensão de trabalho: 5V;
- Dimensões da tela (CxL): 76x25mm;
- Dimensões do módulo (CxLxE): 98x60x11mm;
- Peso: 68g;

2.2.6. Sensor de Gás MQ-7

O Detector de Gás/Sensor de Gás MQ-7 Monóxido de Carbono é um dispositivo de segurança amplamente utilizado em projetos domésticos (automação residencial), que são frequentemente implementados em ferramentas de prototipagem como Arduino ou ESP32.

Figura 7 - Sensor de Gás MQ-7



Fonte: (Robocore, 2023)

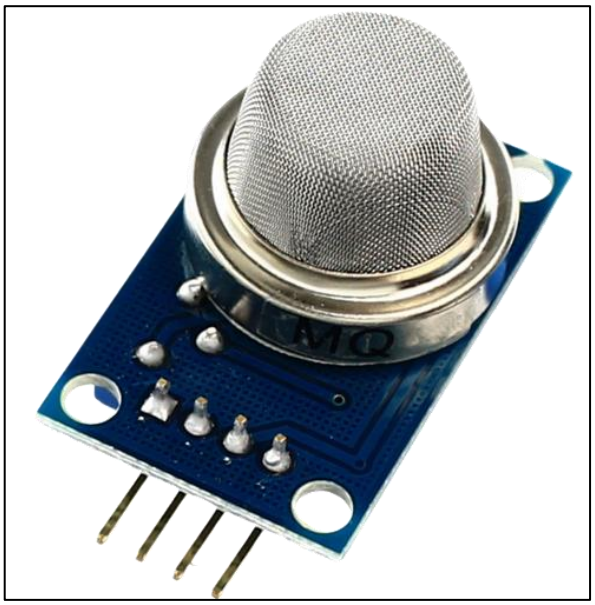
Informações Técnicas:

- Chip: LM393;
- Tensão de funcionamento: DC 5V;
- Sinal de Estimulação: gás Monóxido de Carbono;
- Tipo de interface: Analógico;
- Dimensões (CxLxA): ~32x20x16mm;
- Peso com embalagem: 5g.

2.2.7. Sensor de Gás GLP (Gás de Cozinha) e Gás Natural MQ-5

O Detector de Gás **/** Sensor de Gás GLP (Gás de Cozinha) e Gás Natural MQ-5 é um dispositivo de segurança utilizado principalmente no desenvolvimento de projetos eletrônicos, possuindo alta sensibilidade para detecção de Gás GLP (Gás de Cozinha) e Gás Natural, isso, desde que trabalhe em conjunto com plataformas de prototipagem, como, por exemplo, Arduino ou ESP32.

Figura 8 - Sensor de Gás GLP (Gás de Cozinha) e Gás Natural MQ-5



Fonte: (Robocore, 2023)

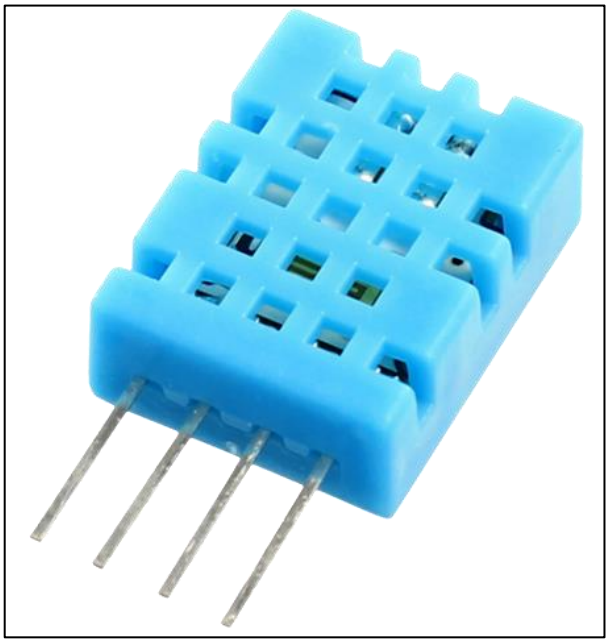
Informações Técnicas:

- Chip: LM393/AZ393;
- Tensão de funcionamento: DC 5V;
- Sinal de Estimulação: Gás GLP (Gás de Cozinha) / Gás Natural;
- Dimensões (CxLxA): 32x20x21mm;
- Peso com embalagem: 8g.

2.2.8. Sensor de Umidade e Temperatura DHT11

O Sensor de Umidade e Temperatura DHT11 como o próprio nome sugere é utilizado para medir a temperatura nas escalas de 0 a 50° graus celsius e a umidade do ar nas faixas de 20 a 90%.

Figura 9 - Sensor de Umidade e Temperatura DHT11



Fonte: (Robocore, 2023)

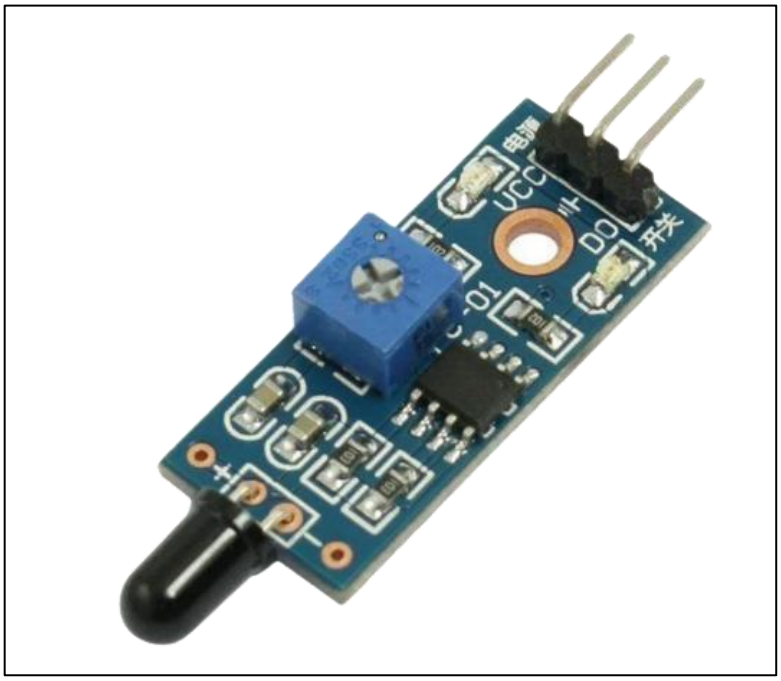
Informações Técnicas:

- Modelo: DHT11;
- Tensão de alimentação: 3 ~ 5,5 VDC;
- Faixa de Corrente: 0,5 ~ 2,5 mA;
- Faixa de operação de umidade: 20 ~ 90%;
- Faixa de operação da temperatura: 0 ~ 50° celsius;
- Margem de Erro: ±4%RH / ±2°C;
- Dimensões (CxLxA): 23x12x5,5mm (com pinos);
- Peso: 1,5g.

2.2.9. Sensor de chama infravermelho

O sensor de chama infravermelho é um equipamento eletrônico desenvolvido para alertar o sistema microcontrolador sobre a presença de fogo em determinado ambiente, para que o mesmo possa tomar as ações necessárias.

Figura 10 - Sensor de chama infravermelho



Fonte: (Robocore, 2023)

Informações Técnicas:

- Sensibilidade da onda: entre 760-1100nm (emitida pelo fogo);
- Faixa de detecção: cerca de 60 graus;
- Temperatura de operação: -25ºC a 85ºC;
- Alimentação: 3 - 5,5VDC;
- Dimensões (CxL): 36x16mm;
- Peso: 2,7g;

2.3. Tecnologias Utilizadas

Confira as ferramentas de software utilizadas no desenvolvimento do B. A. E. Guard:

2.3.1. Unified Modeling Language (UML)

Em conformidade com Guedes (2018), a UML (Unified Modeling Language) é uma linguagem visual amplamente adotada na indústria de engenharia de software para modelar sistemas baseados no paradigma de orientação a objetos. Ela se tornou a linguagem-padrão de modelagem internacionalmente reconhecida, utilizada para definir características de um sistema, como requisitos, comportamento, estrutura lógica, dinâmica de processos e necessidades físicas.

De acordo com Booch (2006), a UML não é uma linguagem de programação, mas sim uma linguagem de modelagem, uma notação que auxilia engenheiros de software na concepção e planejamento de sistemas.

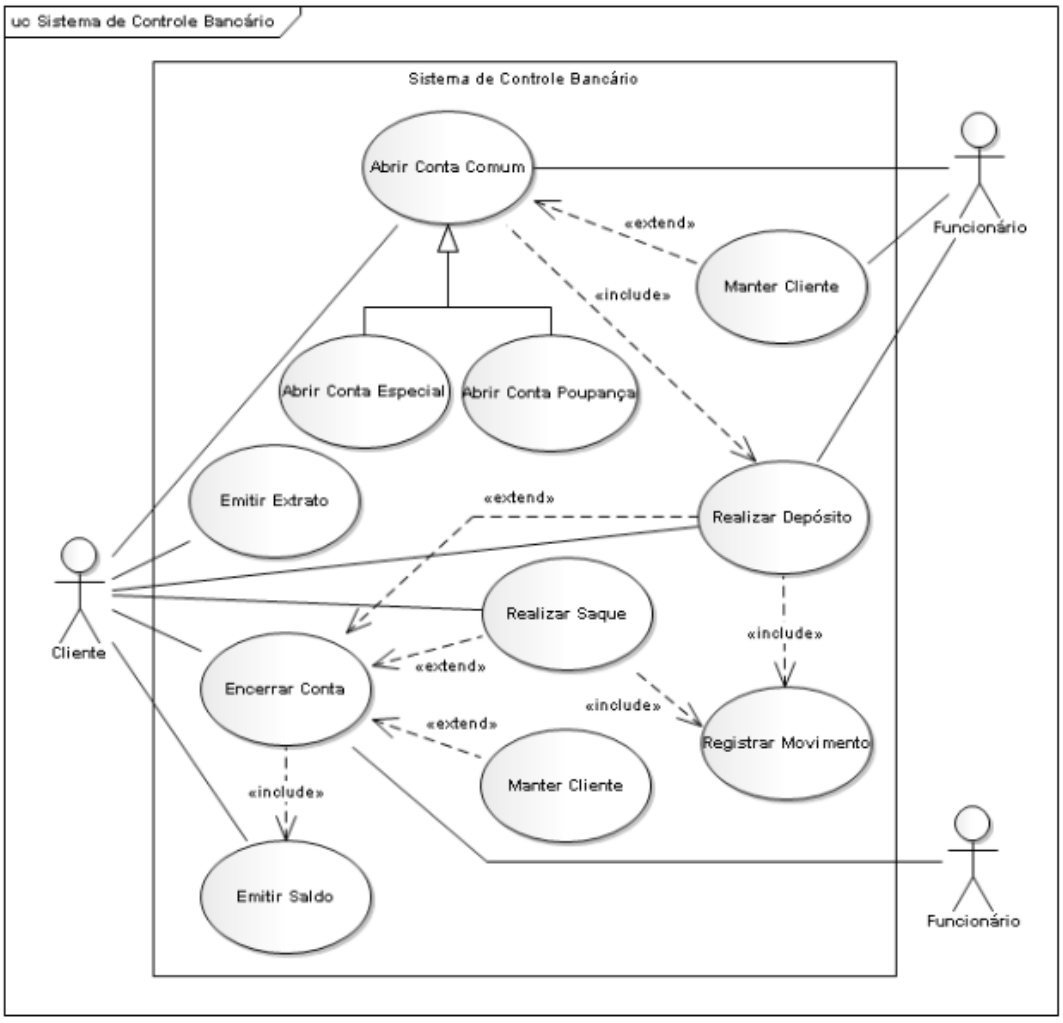
Figura 11 - Logotipo UML

Fonte: (UML, 2023)

2.3.1.1. Diagrama de Caso de Uso

De acordo com Rumbaugh (2006), o diagrama de casos de uso é amplamente utilizado na fase de levantamento e análise de requisitos do sistema, fornecendo uma representação geral e informal do comportamento do sistema. Ele é uma linguagem simples e compreensível, permitindo que os usuários tenham uma visão geral de como o sistema irá se comportar. O objetivo principal desse diagrama é identificar os atores, que podem ser usuários, outros sistemas ou até mesmo hardware especial, que interagem com o sistema. Além disso, o diagrama de caso de uso identifica os casos de uso, as funcionalidades e serviços disponibilizados pelo sistema para os atores, como pode ser visto na figura 11 a seguir.

Figura 12 - Exemplo de Diagrama de Caso de Uso

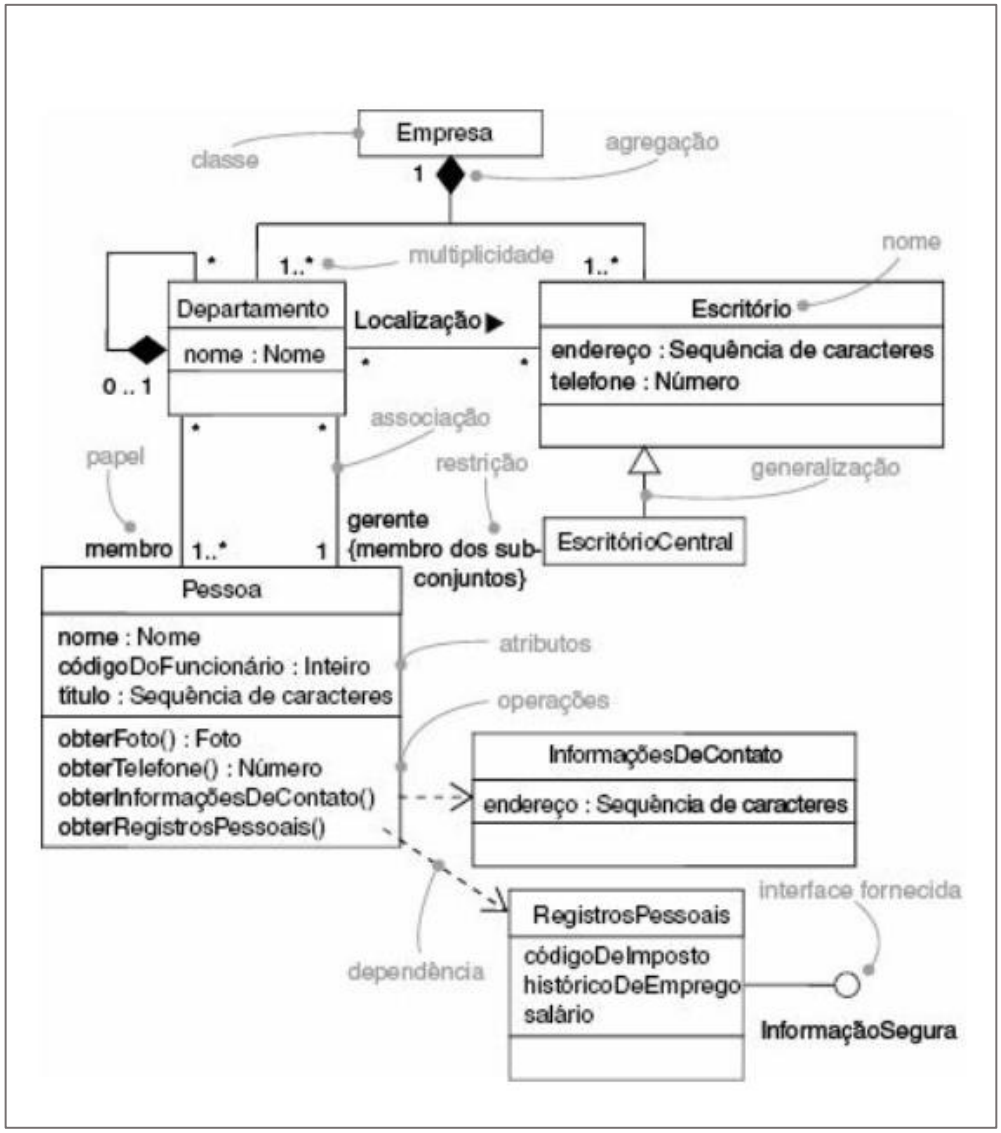


Fonte: (GUEDES, 2011)

2.3.1.2. Diagrama de Classe

Segundo Booch (2006), o diagrama de classes é uma forma de visualizar as entidades do sistema, suas propriedades (atributos) e seus comportamentos (métodos). Ele permite que os desenvolvedores identifiquem as classes principais do sistema, bem como suas associações, heranças e dependências.

Figura 13 - Exemplo de Diagrama de Classe



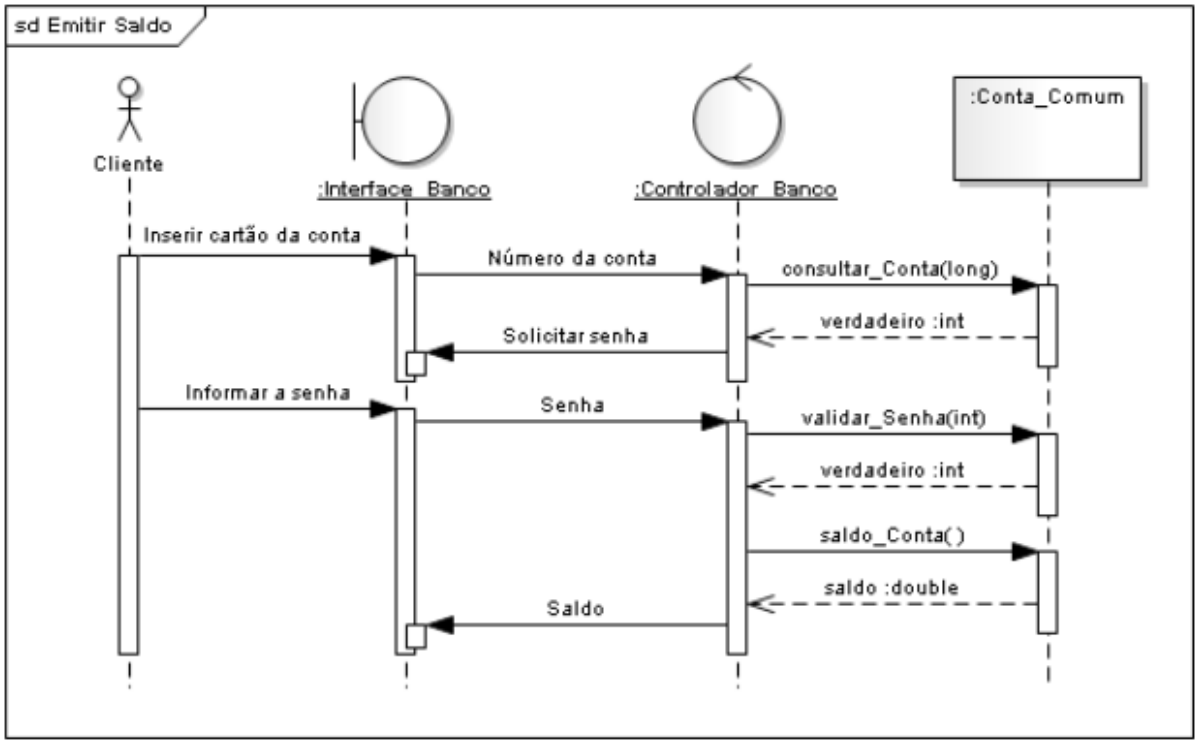
Fonte: (BOOCH, 2012)

Além disso, Rumbaugh e Jacobson (2012) também enfatizaram a importância dos diagramas de classes na modelagem de sistemas, destacando sua clareza e essencialidade no desenvolvimento de software.

2.3.1.3. Diagrama de Sequência

Conforme descrito por Guedes (2011), o diagrama de sequência é uma forma de visualizar o comportamento dinâmico de um sistema, mostrando como os objetos colaboram entre si para realizar uma determinada funcionalidade. Ele destaca as mensagens trocadas entre os objetos ao longo do tempo, permitindo entender as interações.

Figura 14 - Exemplo de Diagrama de Sequência



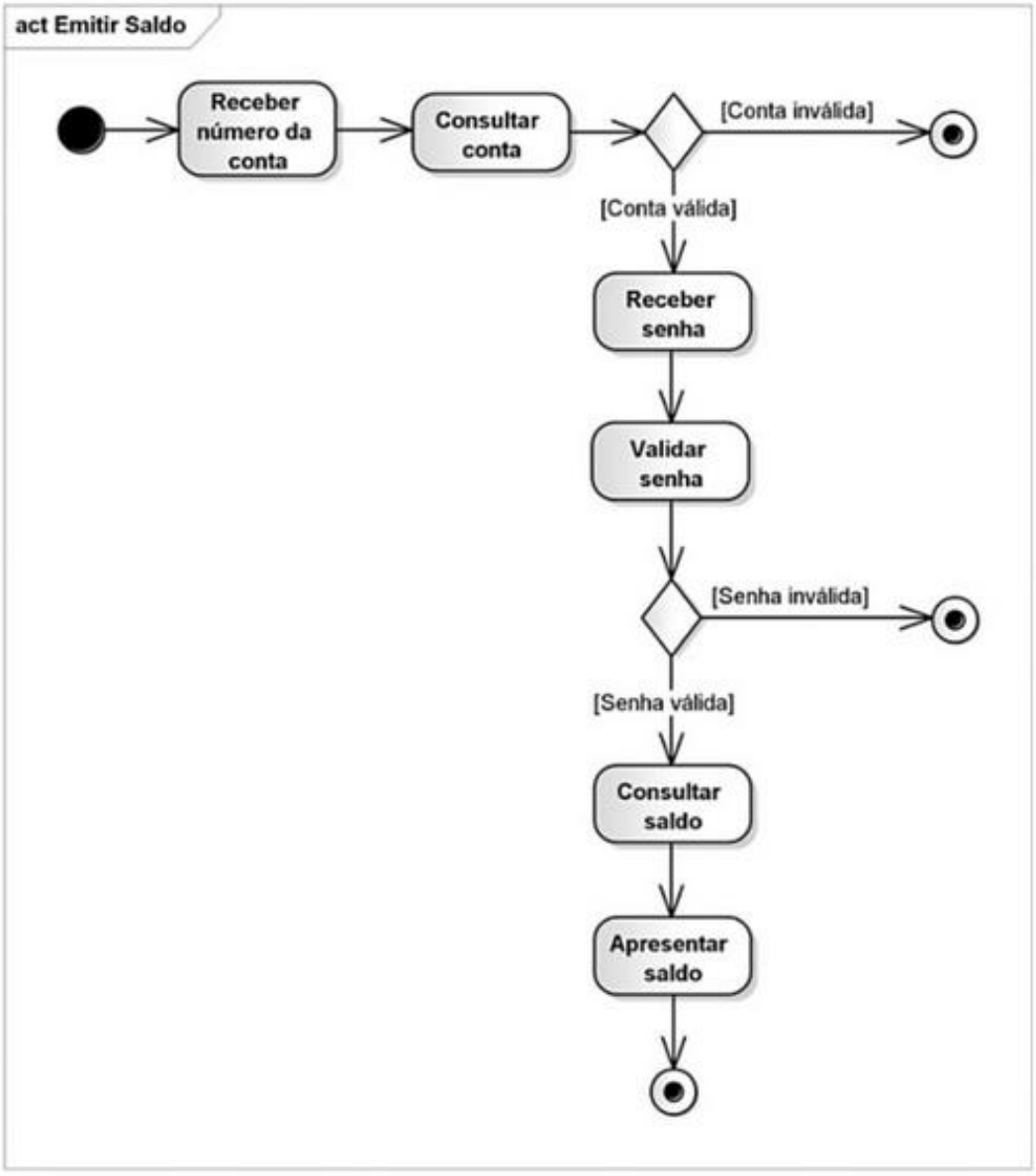
Fonte: (GUEDES, 2011)

Conforme a figura 13, no contexto da emissão de saldo bancário, o diagrama de sequência representaria a interação entre o cliente, o caixa eletrônico e o sistema bancário. Inicialmente, o cliente solicita o saldo ao caixa eletrônico, que envia uma mensagem ao sistema bancário requisitando as informações. O sistema bancário, por sua vez, responde ao caixa eletrônico com o saldo atualizado, que é então exibido ao cliente. Ainda constatado por Martin Fowler (2004), essa representação gráfica permite compreender a ordem e a natureza das interações, facilitando a análise e o entendimento do comportamento do sistema.

2.3.2. Diagrama de Atividades

Congruente a Fowler (2003), o diagrama de atividade é particularmente útil para descrever o fluxo de trabalho, os procedimentos e as interações entre diferentes elementos de um sistema. Ele permite capturar as etapas de um processo, as decisões tomadas durante a execução e as atividades realizadas pelos atores ou componentes do sistema, conforme a figura 14.

Figura 15 - Exemplo de Diagrama de Atividade



Fonte: (GUEDES, 2011)

2.3.3. Internet of Things (IoT)

Nos últimos tempos, o progresso da computação e das tecnologias de comunicação digital tem impulsionado a necessidade de soluções e interações mais ágeis em todos os setores da sociedade. Essa demanda deu origem a um campo emergente na tecnologia conhecido como Internet of Things (IoT), também chamada de Internet das Coisas em português. De acordo com Snataella (2013), a IoT representa o estado atual da Internet, onde objetos, incluindo seres humanos e animais, são transformados em portadores de dispositivos computacionais capazes de se conectar e comunicar.

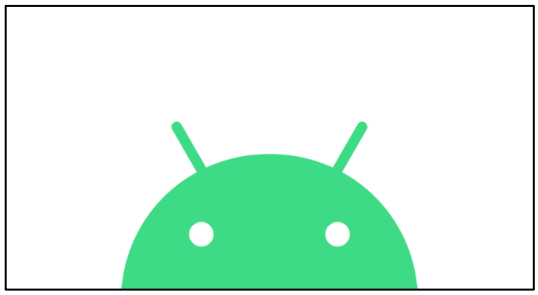
Devido à sua adaptabilidade, as implementações de IoT têm despertado interesse tanto na comunidade acadêmica quanto na indústria (Santos, 2016). Isso ocorre devido à ampla aplicabilidade da IoT, uma vez que praticamente não há restrições em relação aos objetos que podem ser conectados.

O acesso dos dispositivos inteligentes a uma rede pode se dar de diversas formas, sendo os mais populares radiotransmissores de baixa energia, que utilizam os protocolos Wi-Fi ou Bluetooth, além de também ser altamente difundido a comunicação via etiquetas utilizando a tecnologia Radio Frequency Identification Tags (RFID).

2.3.4. Sistema Android

O sistema operacional Android é uma plataforma móvel desenvolvida pela Google, que se tornou extremamente popular em dispositivos móveis, como smartphones e tablets. Lançado em 2008, foi projetado com base no kernel do Linux e se destaca pela sua flexibilidade, usabilidade e vasta biblioteca de aplicativos disponíveis.

Figura 16 - Logotipo do Android

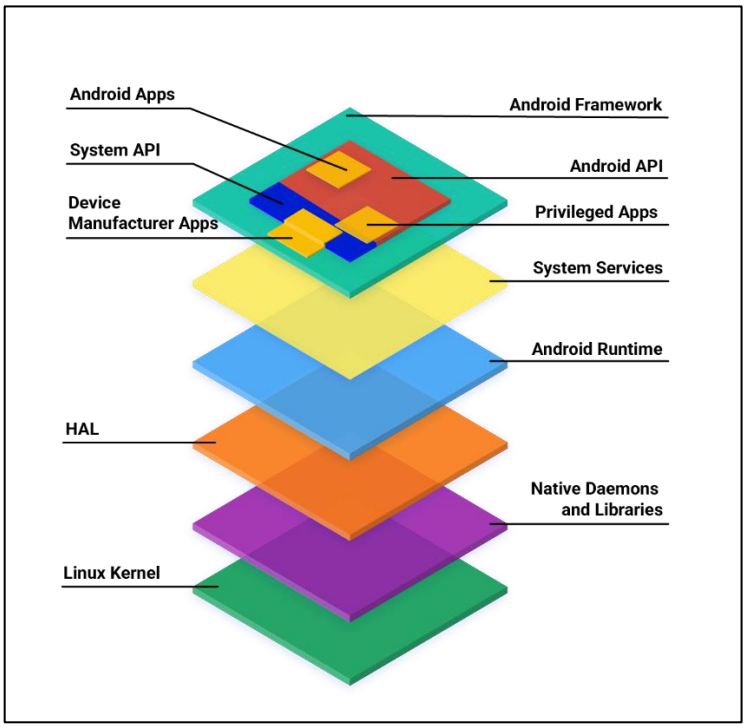


Fonte: (Google LLC, 2023)

2.3.4.1. Arquitetura do Android

De acordo com a documentação oficial (Google LLC, 2023), o Android Open System Platform (AOSP) é um código-fonte Android publicamente disponível e modificável. Qualquer um pode baixar e modificar o AOSP para seu dispositivo. AOSP fornece uma implementação completa e totalmente funcional da plataforma móvel Android. Conforme a seguinte figura 16, essa arquitetura possui estas camadas:

Figura 17 - Camadas Da AOSP



Fonte: (Google LLC, 2023)

- Aplicativos Android (Android Apps): um aplicativo criado exclusivamente por meio da API do Android. A Google Play Store é extensivamente utilizada para localizar e baixar aplicativos, embora existam várias alternativas disponíveis. Em algumas circunstâncias, um fabricante de dispositivos pode optar por fazer uma pré-instalação um aplicativo Android para auxiliar na funcionalidade principal do dispositivo.
- Aplicativos Privilegiados (Privileged Apps): um aplicativo desenvolvido utilizando uma combinação das interfaces de programação de aplicativos (APIs) do Android e do sistema. Esses aplicativos devem ser instalados antecipadamente como aplicativos com privilégios em um dispositivo.

- Aplicativos do fabricante do dispositivo (Device Manufacturer Apps): um aplicativo desenvolvido utilizando uma combinação das interfaces de programação de aplicativos (APIs) do Android, API do sistema e acesso direto à implementação da estrutura do Android. À medida que um produtor de dispositivo pode fazer uso direto de APIs instáveis na estrutura do Android, esses aplicativos devem ser instalados previamente no dispositivo e apenas podem ser atualizados quando o software do sistema do dispositivo for atualizado.
- API do Sistema (System API): a API de Sistema representa as APIs disponíveis exclusivamente para parceiros e fabricantes de equipamentos originais para serem incluídas em aplicativos agrupados.
- API do Android (Android API): permite que os desenvolvedores criem aplicativos para dispositivos Android, interagindo com as funcionalidades e recursos do sistema. A API do Android abrange uma ampla variedade de áreas, como recursos de interface do usuário, comunicação de rede, acesso a banco de dados, gerenciamento de arquivos, reprodução de mídia, serviços de localização, sensores, permissões de segurança, entre outros. Ela fornece aos desenvolvedores as ferramentas necessárias para criar aplicativos eficientes e compatíveis com os dispositivos.
- Frameworks de Android (Android Frameworks): um grupo de códigos pré-compilados sobre os quais os aplicativos são construídos. Fornece um conjunto de componentes e serviços de terceiros que os desenvolvedores podem utilizar ao criar aplicativos. A título exemplificado, é citado a própria API do Android.
- Serviços do Sistema (System Services): camada responsável por fornecer uma variedade de serviços e recursos que os aplicativos podem utilizar para executar tarefas específicas. Esses serviços são implementados como componentes do sistema e oferecem funcionalidades avançadas para os desenvolvedores utilizarem em seus aplicativos. Os serviços do sistema Android abrangem diversas áreas, como serviços de localização, serviços de notificação, serviços de acesso à internet, serviços de armazenamento, serviços de autenticação, entre outros. Esses serviços são projetados para simplificar o desenvolvimento de aplicativos, permitindo que os

desenvolvedores aproveitem as funcionalidades do sistema sem precisar desenvolver esses recursos do zero.

- Android Runtime: o Android Runtime (ART) é o ambiente de execução que melhora o desempenho e a eficiência dos aplicativos. Ele é responsável por interpretar e executar o código-fonte no sistema operacional Android, proporcionando uma experiência de uso mais rápida e eficiente para os usuários.
- HAL: o Hardware Abstraction Layer (HAL) atua como uma interface entre o sistema operacional e o hardware de um dispositivo Android, permitindo que o sistema operacional acesse e controle os recursos de hardware de maneira uniforme, independentemente de suas especificações.
- Camada de Bibliotecas: composta por bibliotecas C/C++ fornecidas pelo Android NDK (Native Development Kit). Oferece funcionalidades de baixo nível, como suporte a gráficos 2D e 3D, reprodução de mídia e manipulação de dados.
- Camada do Kernel do Linux: é a camada mais baixa do sistema Android e é responsável pela interação direta com o hardware do dispositivo. O kernel do Linux fornece os drivers de dispositivo necessários para a comunicação com o processador, memória, câmera, entre outros componentes.

2.3.4.2. Recursos do Android

O sistema Android oferece uma variedade de recursos que fazem com que seja popular e versátil. Entre eles, é possível citar os que foram motivos para a escolha desse sistema operacional para o desenvolvimento do projeto.

O Android permite a execução simultânea de vários aplicativos, permitindo que os usuários alternem rapidamente entre eles e realizem várias tarefas ao mesmo tempo.

O sistema Android fornece um sistema abrangente de notificações, permitindo que os aplicativos enviem mensagens e alertas aos usuários, mesmo quando os aplicativos não estão em primeiro plano.

Os usuários do Android podem personalizar seus dispositivos de várias maneiras, incluindo a escolha de papéis de parede, widgets, ícones e aplicativos de terceiros para personalizar a aparência e o comportamento de seus dispositivos.

O Android oferece uma integração perfeita com os serviços do Google, como Gmail, Google Maps, Google Drive e Google Calendar, entre outros. Isso se deve por ter a própria empresa como sua desenvolvedora e permite que os usuários acessem e sincronizem seus dados em diferentes dispositivos.

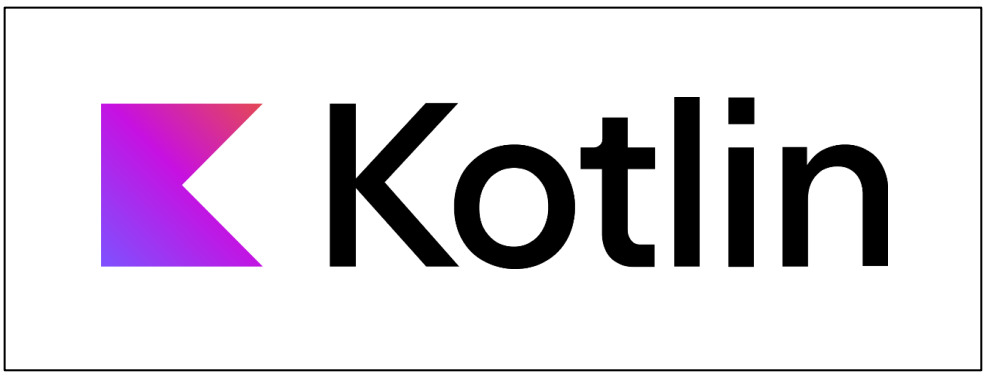
2.3.4.3. Android Studio

O Android Studio é o ambiente de desenvolvimento integrado (IDE) oficial para o desenvolvimento Android e foi utilizado no desenvolvimento do projeto. Ele fornece ferramentas poderosas, como o Android SDK (Software Development Kit) e emuladores de dispositivos virtuais, para facilitar a criação, depuração e testes de aplicativos.

2.3.5. Kotlin

Segundo a documentação oficial (JetBrains, 2023), o Kotlin é uma linguagem de programação projetada para a plataforma Java Virtual Machine (JVM) e desenvolvida pela JetBrains. Ela foi apresentada ao público em 2011 e, desde então, tem ganhado popularidade rapidamente devido à sua interoperabilidade com o Java e sua capacidade de combinar orientação a objetos e programação funcional.

Figura 18 - Logotipo do Kotlin



Fonte: (Kotlin, 2023)

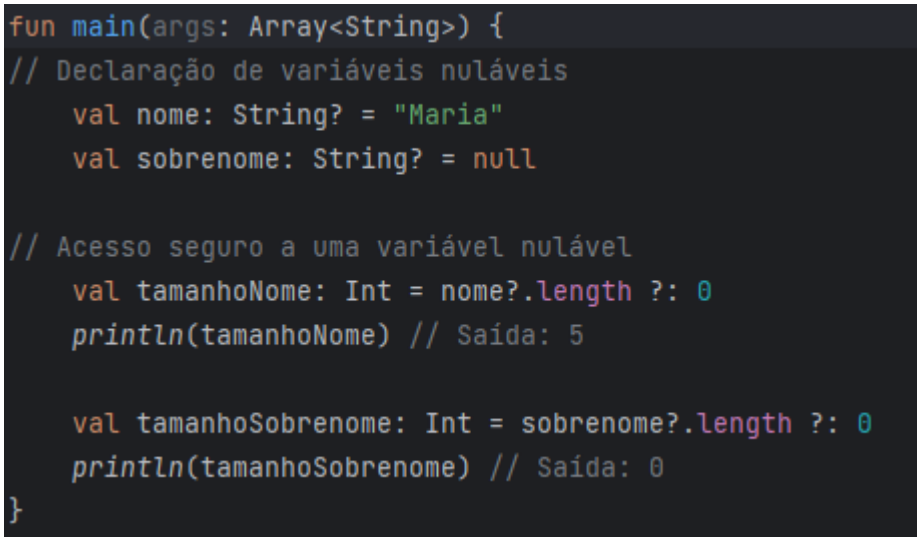
2.3.5.1. Funcionalidade e Recursos

Quando um código em Kotlin é escrito, ele precisa ser traduzido para uma forma que o computador entenda, chamada de "bytecode". O código pode ser compilado em diferentes dispositivos, como computadores, smartphones e tablets. Isso é possível porque ele pode ser convertido para diferentes formatos, dependendo do dispositivo em que será executado. Por exemplo, a linguagem pode ser convertida para bytecode

na JVM, que é uma máquina virtual que permite a execução de programas Java. Dessa forma, o Kotlin é compatível com a vasta quantidade de bibliotecas e códigos existentes em Java, o que facilita muito o desenvolvimento de aplicativos.

Outro recurso importante é a nulidade segura. Diferentemente do Java, onde qualquer referência de objeto pode ser nula, o Kotlin distingue entre referências que podem ser nulas e as que não podem. Isso ajuda a evitar o temido "NullPointerException" durante o tempo de execução, que é um problema comum em muitos sistemas Java e indica que a aplicação tentou usar uma referência de um objeto que estava com valor nulo. Com os sistemas nessa linguagem, é possível declarar explicitamente se uma variável pode ou não ser nula, oferecendo maior segurança e confiabilidade no código.

Figura 19 - Exemplo de código com nulidade segura



```
fun main(args: Array<String>) {
// Declaração de variáveis nuláveis
    val nome: String? = "Maria"
    val sobrenome: String? = null

// Acesso seguro a uma variável nulável
    val tamanhoNome: Int = nome?.length ?: 0
    println(tamanhoNome) // Saída: 5

    val tamanhoSobrenome: Int = sobrenome?.length ?: 0
    println(tamanhoSobrenome) // Saída: 0
}
```

Fonte: (Elaborado pelo próprio autor, 2023)

O Kotlin permite a criação de funções de extensão, que são funções que podem ser chamadas como se fossem métodos de uma classe, mesmo sem modificar a classe original. Esse recurso proporciona uma maior flexibilidade na adição de funcionalidades às classes existentes, sem a necessidade de herança ou alterações no código fonte original. As funções de extensão são muito úteis para estender funcionalidades de bibliotecas e APIs.

Figura 20 - Exemplo de código de função de extensão

```
// Função de extensão para a classe Int que verifica se o número é par
fun Int.isEven(): Boolean {
    // Verifica se o número é divisível por 2 (ou seja, é par)
    return this % 2 == 0
}

fun main(args: Array<String>) {
    // Número de exemplo
    val number = 10

    // Chamando a função de extensão isEven() na variável number
    val isNumberEven = number.isEven()

    // Imprimindo o resultado no console
    if (isNumberEven) {
        println("$number é um número par.")
    } else {
        println("$number é um número ímpar.")
    }
}
```

Fonte: (Elaborado pelo próprio autor, 2023)

Neste exemplo, é criada uma função de extensão chamada *"isEven()"* para a classe *"Int"*, que verifica se um número é par. A função retorna *"true"* (verdadeiro) se o número for divisível por 2, ou seja, é par e *"false"* (falso) caso contrário.

No bloco *"main()"*, definimos um número de exemplo como 10 e chamamos a função de extensão "isEven()" nessa variável. O resultado é armazenado na variável "isNumberEven". Em seguida, exibimos uma mensagem no console, informando se o número é par ou ímpar.

2.3.6. Jetpack Compose

Conforme a documentação oficial (Google LLC, 2023), O Jetpack Compose é uma biblioteca moderna de IU (Interface de Usuário) desenvolvida pelo Google para a criação de interfaces de usuário nativas do Android. Ele foi introduzido como uma alternativa ao Android XML e às bibliotecas tradicionais de IU, como o Android View, com o objetivo de simplificar o desenvolvimento de aplicativos e oferecer uma experiência mais fluida e produtiva para os desenvolvedores.

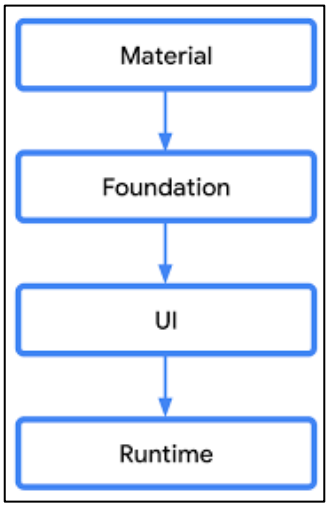
Figura 21 - Logotipo do Jetpack Compose

Fonte: (Google LLC, 2023)

2.3.6.1. Arquitetura

O Jetpack Compose não é um projeto único, mas sim uma combinação de vários módulos que formam uma pilha completa.
As principais camadas do Jetpack Compose são estas:

Figura 22 - Camadas da arquitetura do Jetpack Compose



Fonte: (Google LLC, 2023)

- Camada Material: a camada Material no Jetpack Compose é baseada no Material Design, que é uma linguagem de design desenvolvida pelo Google. Essa camada fornece componentes pré-definidos, como botões, campos de texto, cartões e muito mais, seguindo as diretrizes e padrões do Material Design. Ela permite a criação de interfaces do usuário modernas e consistentes, com a aparência e o comportamento esperados pelos usuários do Android.
- Camada Foundation: a camada Foundation é responsável por fornecer as bases e blocos de construção para o desenvolvimento de interfaces do usuário no Jetpack Compose. Ela inclui APIs essenciais para lidar com temas, estilos, recursos do sistema, animações, gestos e outros aspectos fundamentais. Essa camada reduz a complexidade, facilitando o desenvolvimento de interfaces do usuário ricas e interativas.
- Camada UI: a camada UI é onde se constrói a interface do usuário usando os componentes do Jetpack Compose. Nessa camada, você utiliza os elementos da camada Material, personaliza-os com base nas necessidades do seu

aplicativo e combina-os para criar a hierarquia visual da sua interface. As interações e comportamentos dos componentes são definidos aqui.

- Camada Runtime: a camada Runtime é responsável por executar e gerenciar o ciclo de vida da interface do usuário criada com o Jetpack Compose. Ela trata da renderização eficiente dos componentes, processamento de eventos, tratamento de estados e sincronização com o sistema operacional Android. A camada Runtime garante que sua interface do usuário seja atualizada de forma eficaz e responsiva, refletindo as interações e alterações de estado do usuário.

2.3.6.2. Princípios e Conceitos do Jetpack Compose

Uma das principais características do Jetpack Compose é a abordagem declarativa para a criação de interfaces de usuário. Isso significa que, em vez de escrever código imperativo para descrever como a interface deve ser construída e atualizada, nele, você especifica o estado desejado da interface e a biblioteca se encarrega de renderizar a IU com base nesse estado.

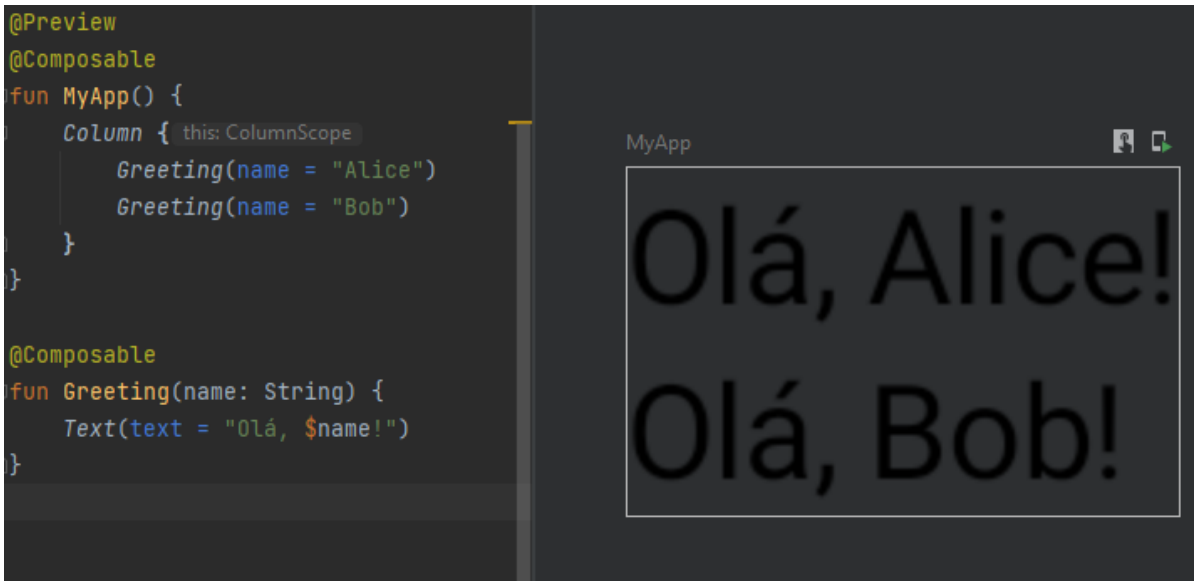
Figura 23 - Exemplo de código declarativo

Fonte: (Elaborado pelo próprio autor, 2023)

Neste exemplo, a função composable "MyCounter" recebe um parâmetro de contagem e exibe esse valor em um componente Text. Ao atualizar o valor de contagem, o Jetpack Compose automaticamente atualiza a interface de usuário com o novo valor.

O Jetpack Compose utiliza funções composable, que são funções puras e independentes, para descrever cada parte da interface de usuário. Cada função composable representa um componente reutilizável que pode ser combinado para construir uma interface complexa. Essas funções composable recebem parâmetros de entrada e retornam um objeto que representa a interface gráfica.

Figura 24 - Exemplo de código de Composable

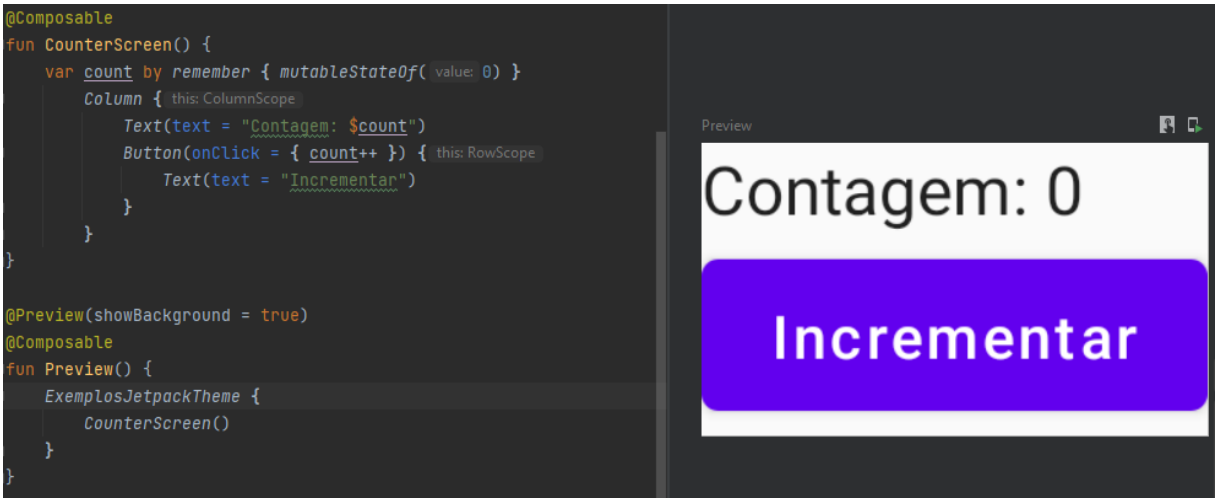


Fonte: (Elaborado pelo próprio autor, 2023)

Neste exemplo, a função composable "MyApp" cria uma coluna que contém dois componentes "Greeting", cada um exibindo uma saudação com um nome diferente. A função "Greeting" é uma função composable separada que recebe o nome como parâmetro e retorna um componente Text.

O Jetpack Compose simplifica o gerenciamento de estado em comparação com as abordagens tradicionais. Ele oferece um conjunto de APIs para declarar e atualizar o estado dos componentes de maneira eficiente. Além disso, o Jetpack Compose possui um sistema de reativação embutido, o que significa que ele atualiza automaticamente a IU sempre que o estado subjacente de um componente é alterado.\

Figura 25 - Exemplo de código com *state management*



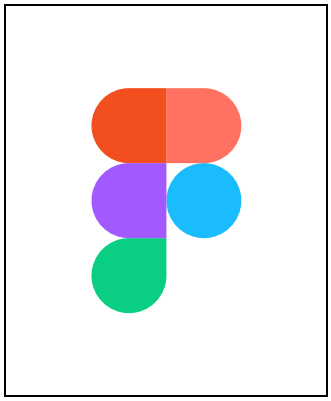
Fonte: (Elaborado pelo próprio autor, 2023)

Neste exemplo, a função composable "CounterScreen" possui uma variável de estado "count" que é gerenciada pela função "remember { mutableStateOf(0) }". A interface exibe o valor atual de contagem e um botão que incrementa o valor quando clicado. O Jetpack Compose se encarrega de atualizar automaticamente a interface sempre que o valor de contagem é alterado.

2.3.7. Figma

O Figma é uma ferramenta de design colaborativo baseada na web, mas que também pode ser utilizada em desktop, e tem ganhado popularidade significativa nos últimos anos. Ela permite que equipes de design trabalhem juntas em tempo real, independentemente da localização geográfica, o que torna o processo de desenvolvimento de interfaces mais eficiente e eficaz.

Figura 26 - Logotipo do Figma



Fonte: (Figma, 2023)

O Figma oferece uma ampla gama de recursos e funcionalidades que tornam o design colaborativo mais eficiente e produtivo.

A ferramenta permite que várias pessoas editem o mesmo arquivo simultaneamente, possibilitando que os membros da equipe visualizem as atualizações em tempo real. Isso elimina a necessidade de usar outras formas de comunicação para compartilhar alterações.

Por ser baseado na web, é possível armazenar todos os arquivos na nuvem, o que permite que os designers acessem e editem seus projetos de qualquer lugar, a qualquer momento. Também oferece recursos de criação de componentes reutilizáveis, o que permite que os usuários criem bibliotecas de componentes e os compartilhem entre diferentes projetos.

Além disso, o programa permite a criação de protótipos interativos para testar a usabilidade e a experiência do usuário. Isso facilita a validação de ideias e a iteração rápida do design.

2.3.8. Firebase

Conforme a documentação disponibilizada pela Google LLC (2022), o Firebase é uma plataforma, desenvolvida pelo Google, que fornece uma variedade de serviços e ferramentas para simplificar e acelerar o processo de desenvolvimento de aplicativos. Com uma ampla gama de recursos, a plataforma tornou-se uma opção popular para desenvolvedores que desejam criar aplicativos escaláveis, confiáveis e em tempo real.

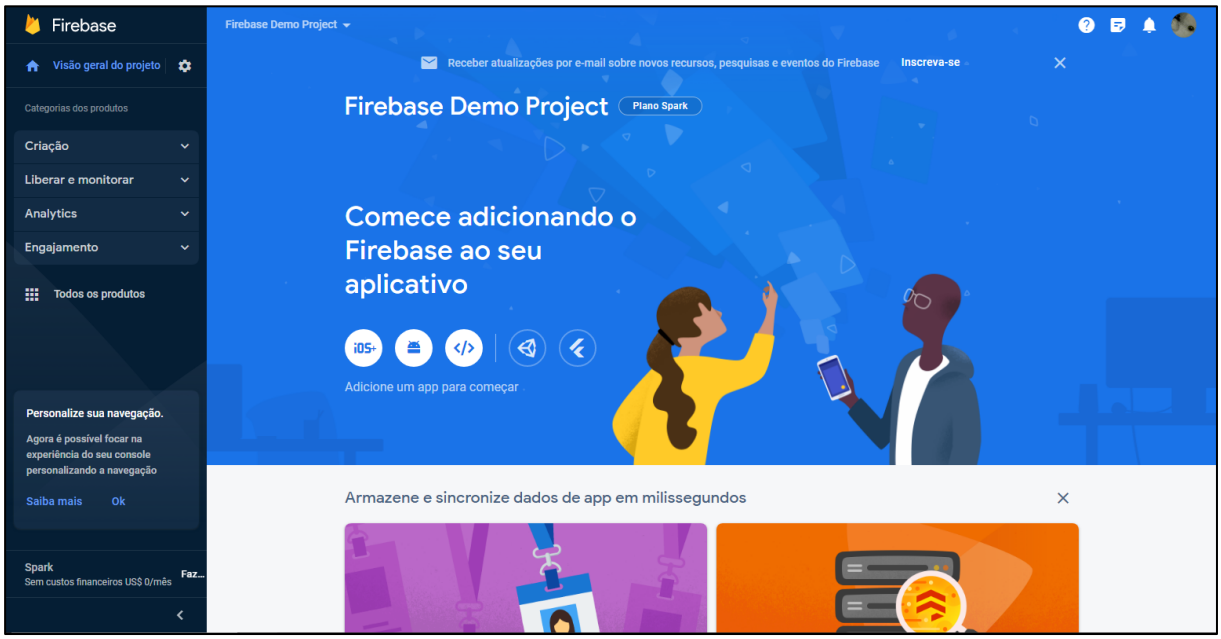
Figura 27 - Logotipo do Firebase

Fonte: (Google LLC, 2023)

2.3.8.1. Visão geral

O Firebase é uma plataforma de desenvolvimento de aplicativos baseada em nuvem que oferece uma ampla gama de serviços e recursos para desenvolvedores. Ele abrange várias áreas-chave do desenvolvimento de aplicativos, como armazenamento de dados, autenticação de usuários, hospedagem de aplicativos, mensagens em tempo real, análise de dados e muito mais. Com o Firebase, os desenvolvedores podem construir aplicativos de alta qualidade, escaláveis e eficientes, sem precisar se preocupar com a infraestrutura de backend complexa.

Figura 28 - Console do Firebase



Fonte: (Elaborado pelo próprio autor, 2023)

2.3.8.2. Principais recursos

Um dos principais recursos do Firebase é o Realtime Database, um banco de dados NoSQL hospedado em nuvem, que permite armazenar e sincronizar dados em tempo real entre clientes e servidores. Essa funcionalidade é particularmente útil para aplicativos colaborativos, como aplicativos de chat, jogos multiplayer e ferramentas de colaboração em tempo real.

O Firebase fornece recursos abrangentes de autenticação de usuários, permitindo que os desenvolvedores adicionem facilmente recursos de login e registro aos seus aplicativos. Com suporte para autenticação por e-mail/senha, autenticação social (por meio de provedores como Google, Facebook, Twitter) e autenticação anônima, os

desenvolvedores podem implementar um sistema seguro e flexível de gerenciamento de usuários.

Com o Firebase Hosting, os desenvolvedores podem implantar e hospedar seus aplicativos da web de forma rápida e fácil. Ele fornece um ambiente de hospedagem seguro, escalável e de alto desempenho, permitindo que os aplicativos sejam entregues aos usuários finais com rapidez e confiabilidade.

Por meio do Firebase Cloud Messaging, os desenvolvedores podem enviar notificações push para seus aplicativos em dispositivos móveis e da web. Essa funcionalidade é essencial para manter os usuários engajados e informados sobre eventos importantes, como atualizações de conteúdo, mensagens diretas e outras interações relevantes.

A plataforma também fornece uma ampla gama de ferramentas para análise e monitoramento de desempenho de aplicativos. Com recursos como o Firebase Analytics e o Firebase Performance Monitoring, os desenvolvedores podem coletar dados sobre o uso do aplicativo, entender o comportamento do usuário, rastrear conversões e identificar gargalos de desempenho, permitindo otimizar continuamente a experiência do usuário.

2.3.8.3. Aplicações do Firebase

O Firebase é amplamente utilizado para o desenvolvimento de aplicativos móveis em plataformas iOS e Android. Com recursos como autenticação de usuários, armazenamento de dados em tempo real e notificações push, os desenvolvedores podem criar aplicativos móveis altamente interativos e escaláveis.

Além de aplicativos móveis, o Firebase também é uma excelente opção para o desenvolvimento de aplicativos da web. Com recursos como hospedagem de aplicativos, autenticação de usuários e banco de dados em tempo real, os desenvolvedores podem criar aplicativos web responsivos e eficientes.

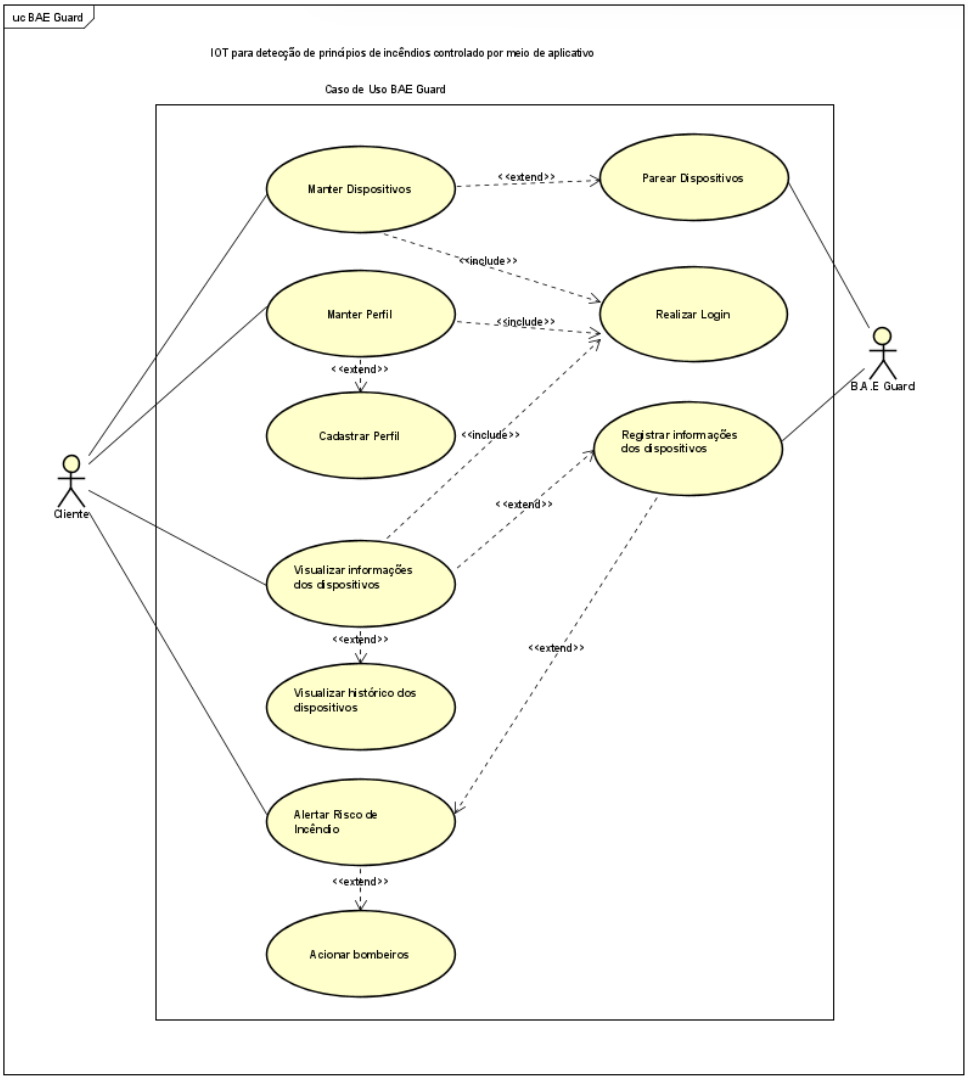
3. DESENVOLVIMENTO

Neste capítulo, serão apresentados e descritos todos os processos de criação do B. A. E. Guard. A documentação dos processos é iniciada por meio dos métodos de estudo UML, ilustrações de códigos, ilustrações das telas da aplicação e fotografias do dispositivo.

3.1. Diagrama de Caso de Uso

Pode-se observar o caso de uso do aplicativo com a figura abaixo, que expressa como o cliente e o dispositivo B. A. E. Guard interagem com o sistema.

Figura 29 - Diagrama de Caso de Uso Geral da Aplicação



Fonte: (Elaborado pelo próprio autor, 2023)

3.1.1. Documentação dos Casos de Uso

Em conformidade com as instruções dos parâmetros UML, segue a documentação dos casos de uso anteriormente apresentados:

Requisitos funcionais do Cliente:

- RF01 – O sistema deverá permitir que o cliente efetue cadastro e/ou login;
- RF02 – O sistema deverá permitir que o cliente possa cadastrar um novo dispositivo;
- RF03 – O cliente poderá visualizar um dispositivo cadastrado;
- RF04 – O cliente poderá editar um dispositivo cadastrado;
- RF05 – O cliente poderá excluir um dispositivo cadastrado;
- RF06 – O cliente poderá visualizar as informações de seu perfil;
- RF07 – O cliente poderá editar as informações de seu perfil.
- RF08 – O cliente poderá excluir seu perfil;
- RF09 – O cliente poderá visualizar as informações de monitoramento em tempo real;
- RF10 – O cliente poderá visualizar o vídeo da câmera do dispositivo em tempo real;
- RF11 – O cliente poderá visualizar o histórico de monitoramento; e
- RF12 – O cliente poderá acionar os bombeiros quando as informações estiverem fora do padrão de segurança.

Requisitos funcionais do dispositivo B. A. E. Guard:

- RF13 – O sistema deverá permitir o pareamento do dispositivo com o perfil do aplicativo; e
- RF14 – O dispositivo deverá cadastrar as informações em tempo real.

Requisitos não-funcionais do sistema geral:

- RNF1 – O sistema deve apresentar medida de temperatura em graus Celsius;

- RNF2 – O aplicativo deve ser compatível com sistema Android igual ou superior a versão 7.0 (Nougat);
- RNF3 – O aplicativo deverá ser desenvolvido utilizando linguagem Kotlin e o kit de ferramentas Jetpack Compose;
- RNF4 – O aplicativo deverá ter interface amigável;
- RNF5 – O dispositivo deverá apresentar, em uma tela LCD: data, hora, temperatura, umidade e status de fumaça e vazamento de gás;
- RNF6 – O aplicativo deverá apresentar temperatura, umidade e status de fumaça e vazamento de gás;
- RNF7 – Deverá ser possível efetuar o login a partir de uma conta Google;
- RNF8 – O sistema deverá utilizar o serviço em nuvem Firebase para a conexão entre o aplicativo e o dispositivo;
- RNF9 – A lógica do dispositivo deverá ser desenvolvida em C++;
- RNF10 – O dispositivo deverá apresentar sensores de temperatura, umidade, chama, gás natural e GLP e gás monóxido de carbono; e

Regra de negócio do sistema em geral:

- RN1 – O dispositivo deverá cumprir o Regulamento de Segurança Contra Incêndio das Edificações e Áreas de Risco no Estado de São Paulo.

Tabelas descritivos dos casos de uso:

Tabela 1 - Descrição do caso de uso a partir da perspectiva do cliente

Nome do Caso de Uso	B. A. E. Guard - IOT para detecção de incêndio.
Ator Principal	Cliente
Ator Secundário	B. A. E. Guard
Resumo	A descrição do caso de uso geral apresenta o fluxo de ações executadas pelos atores, representando o cenário ideal em relação às interações do cliente.
Ações do Ator	Ações do Sistema
1. Realizar cadastro	
2. Realizar login	
	3. Listar dispositivos cadastrados
4. Manter Dispositivos	
	5. Validar informações de monitoramento em tempo real
	6. Alertar risco de incêndio
7. Acionar os bombeiros	
	8. Listar informações de monitoramento em tempo real
9. Visualizar informações de monitoramento em tempo real	
	10. Disponibilizar vídeo da câmera do dispositivo em tempo real
11. Visualizar vídeo da câmera do dispositivo em tempo real	

	12. Listar histórico de monitoramento
13. Visualizar histórico de monitoramento	
	14. Listar informações do perfil
15. Manter perfil	

Fonte: (Elaborado pelo próprio autor, 2023)

Tabela 2 - Descrição do caso de uso a partir da perspectiva do dispositivo

Nome do Caso de Uso	B. A. E. Guard - IOT para detecção de incêndio.
Ator Principal	B. A. E. Guard
Ator Secundário	Cliente
Resumo	A descrição do caso de uso geral apresenta o fluxo de ações executadas pelos atores, representando o cenário ideal em relação às interações do dispositivo B. A. E. Guard.
Ações do Ator	Ações do Sistema
	1. Solicitar pareamento com o perfil
2. Realizar pareamento com o perfil	
3. Cadastrar informações em tempo real	

Fonte: (Elaborado pelo próprio autor, 2023)

Tabela 3 - Descrição do caso de uso “Manter Dispositivos” (Cliente)

Nome do Caso de Uso	Manter Dispositivos
Ator Principal	Cliente
Ator Secundário	B. A. E. Guard
Resumo	Este caso de uso descreve o processo de cadastro, visualização, edição e exclusão de um dispositivo B. A. E. Guard.
Ações do Ator	Ações do Sistema
1. Realizar login	
2. Adicionar dispositivo	
	3. Solicitar nome do dispositivo e ambiente
4. Informar o nome do dispositivo e o ambiente	
	5. Exibir tutorial de configuração
6. Confirmar entendimento do tutorial	
	7. Gerar um QR - Code para o pareamento
8. Apresentar o QR - Code para a câmera do dispositivo	
	9. Validar as informações
	10. Realizar o cadastro do dispositivo
	11. Listar dispositivos cadastrados
12. Visualizar dispositivos	
13. Selecionar dispositivo	

	14. Exibir opções de edição ou exclusão
15. Solicitar edição do dispositivo	
	16. Solicitar novo nome e ambiente
17. Informar novo nome e ambiente	
	18. Validar alteração das informações
	19. Realizar alteração das informações
20. Solicitar exclusão do dispositivo	
	21. Solicitar confirmação da exclusão
22. Confirmar exclusão	
	23. Excluir dispositivo

Fonte: (Elaborado pelo próprio autor, 2023)

Tabela 4 - Descrição do caso de uso “Parear Dispositivos” (B. A. E. Guard)

Nome do Caso de Uso	Parear Dispositivos
Ator Principal	B. A. E. Guard
Ator Secundário	Cliente
Resumo	Este caso de uso descreve o processo de pareamento do dispositivo B. A. E. Guard com o perfil do cliente.
Ações do Ator	Ações do Sistema
1. Realizar a leitura do QR - Code	
	2. Validar as informações
3. Realizar o pareamento com o perfil correto	

Fonte: (Elaborado pelo próprio autor, 2023)

Tabela 5 - Descrição do caso de uso “Manter Perfil” (Cliente)

Nome do Caso de Uso	Manter Perfil
Ator Principal	Cliente
Resumo	Este caso de uso descreve o processo de visualização, edição e exclusão do perfil do cliente.
Ações do Ator	Ações do Sistema
1. Realizar login	
2. Solicitar visualização do perfil	
	3. Listar informações do perfil
4. Visualizar informações do perfil	
	5. Exibir opções de edição e exclusão
6. Solicitar edição de nome	
	7. Solicitar senha atual
8. Informar senha atual	
	9. Validar senha
	10. Solicitar novo nome
11. Informar novo nome	
	12. Validar as informações
	13. Realizar a alteração do nome
14. Solicitar edição de e-mail	
	15. Solicitar senha atual
16. Informar senha atual	
	17. Validar senha
	18. Solicitar novo e-mail

19. Informar novo de e-mail	
	20. Validar as informações
	21. Realizar a alteração do e-mail
22. Solicitar alteração de senha	
	23. Solicitar senha atual
24. Informar senha atual	
	25. Solicitar nova senha
26. Informar nova senha	
	27. Validar nova senha
	28. Realizar alteração da senha
29. Solicitar exclusão do perfil	
	30. Exibe confirmação de exclusão
31. Confirmar exclusão	
	32. Realizar exclusão do perfil

Fonte: (Elaborado pelo próprio autor, 2023)

Tabela 6 - Descrição do caso de uso “Cadastrar Perfil” (Cliente)

Nome do Caso de Uso	Cadastrar Perfil
Ator Principal	Cliente
Resumo	Este caso de uso descreve o processo de cadastro do perfil do cliente.
Ações do Ator	Ações do Sistema
1. Solicitar cadastro de perfil	
	2. Solicitar informações de cadastro do perfil
3. Informar nome, e-mail e senha	
	4. Validar informações
	5. Efetuar cadastro do perfil
	6. Redirecionar o cliente para realizar o login

Fonte: (Elaborado pelo próprio autor, 2023)

Tabela 7 - Descrição do caso de uso “Realizar Login” (Cliente)

Nome do Caso de Uso	Realizar Login
Ator Principal	Cliente
Resumo	Este caso de uso descreve o processo da realização do login na aplicação.
Ações do Ator	Ações do Sistema
1. Solicitar realização login	
	2. Solicitar e-mail e senha
3. Informar e-mail e senha	
	4. Validar informações
	5. Realizar login

Fonte: (Elaborado pelo próprio autor, 2023)

Tabela 8 - Descrição do caso de uso “Visualizar Informações em Tempo Real”

Nome do Caso de Uso	Visualizar Informações em Tempo Real
Ator Principal	Cliente
Ator Secundário	B. A. E. Guard
Resumo	Este caso de uso descreve o processo da visualização das informações em tempo real na perspectiva do cliente.
Ações do Ator	Ações do Sistema
1. Realizar login	
	2. Listar informações em tempo real
3. Visualizar informações em tempo real	
4. Solicitar informações mais detalhadas	
	5. Listar detalhamento das informações
6. Solicitar visualização do vídeo em tempo real da câmera	
	7. Exibir vídeo em tempo real da câmera

Fonte: (Elaborado pelo próprio autor, 2023)

Tabela 9 - Descrição do caso de uso “Cadastrar Informações” (B. A. E. Guard)

Nome do Caso de Uso	Cadastrar Informações
Ator Principal	B. A. E. Guard
Ator Secundário	Cliente
Resumo	Este caso de uso descreve o processo de cadastro das informações coletadas pelo dispositivo B. A. E. Guard.
Ações do Ator	Ações do Sistema
1. Enviar informações coletadas em tempo real	
	2. Receber informações enviadas
	3. Cadastrar informações recebidas no perfil pareado
4. Enviar informações do vídeo da câmera em tempo real	
	5. Receber vídeo da câmera em tempo real
	6. Exibir vídeo da câmera em tempo real no perfil pareado

Fonte: (Elaborado pelo próprio autor, 2023)

Tabela 10 - Descrição do caso de uso “Visualizar histórico” (Cliente)

Nome do Caso de Uso	Visualizar histórico
Ator Principal	Cliente
Ator Secundário	B. A. E. Guard
Resumo	Este caso de uso descreve o processo de visualização do histórico de monitoramento das informações coletadas pelo dispositivo B. A. E. Guard.
Ações do Ator	Ações do Sistema
1. Realizar login	
2. Solicitar visualização do histórico	
	3. Listar histórico de monitoramento
4. Visualizar histórico de monitoramento	

Fonte: (Elaborado pelo próprio autor, 2023)

Tabela 11 - Descrição do caso de uso “Alertar Risco de Incêndio” (Cliente)

Nome do Caso de Uso	Alertar Risco de Incêndio
Ator Principal	Cliente
Ator Secundário	B. A. E. Guard
Resumo	Este caso de uso descreve o processo de alertar o cliente sobre risco de incêndio quando as informações de monitoramento estiverem fora do padrão seguro.
Ações do Ator	Ações do Sistema
	1. Validar informações de monitoramento
	2. Alertar risco de incêndio, caso as informações estiverem fora do padrão de segurança
3. Receber alerta de risco de incêndio	

Fonte: (Elaborado pelo próprio autor, 2023)

Tabela 12 - Descrição do caso de uso “Acionar bombeiros” (Cliente)

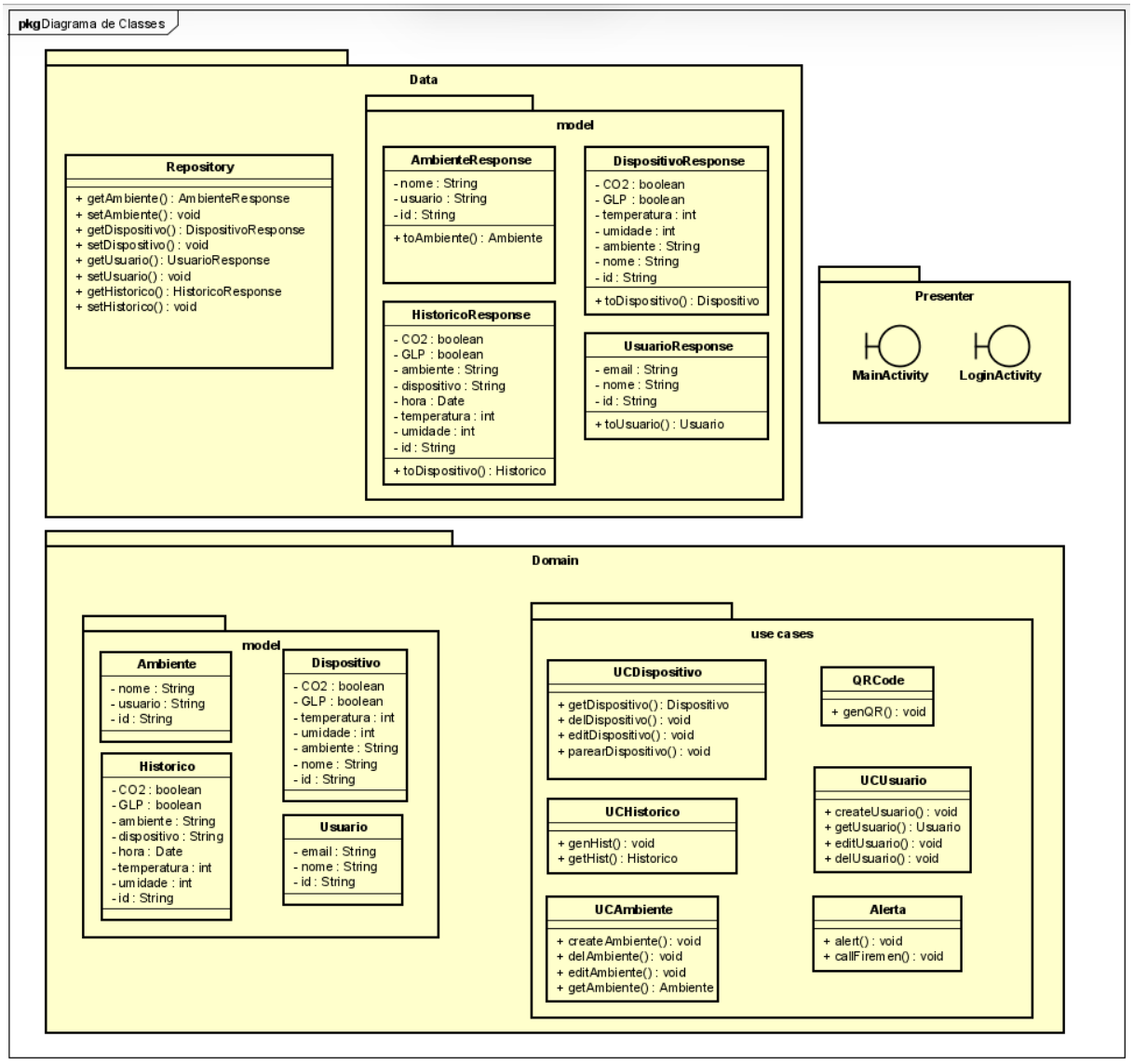
Nome do Caso de Uso	Acionar bombeiros
Ator Principal	Cliente
Resumo	Este caso de uso descreve o processo da ação de acionar os bombeiros pelo cliente.
Ações do Ator	Ações do Sistema
1. Receber alerta de risco de incêndio	
	2. Exibir opção para acionar os bombeiros
3. Selecionar opção para acionar os bombeiros	
	4. Redirecionar o cliente para a tela “Telefone” com o número dos bombeiros
5. Acionar os bombeiros	

Fonte: (Elaborado pelo próprio autor, 2023)

3.2. Diagrama de Classes

Na figura a seguir, é possível observar o diagrama de classe, que representa a estrutura da aplicação forma de classes, suas propriedades e os relacionamentos entre elas. O diagrama de classe mostra as entidades envolvidas no sistema, seus atributos e como elas se relacionam entre si, proporcionando uma visão clara da estrutura e organização do sistema.

Figura 30 - Diagrama de classes de aplicação

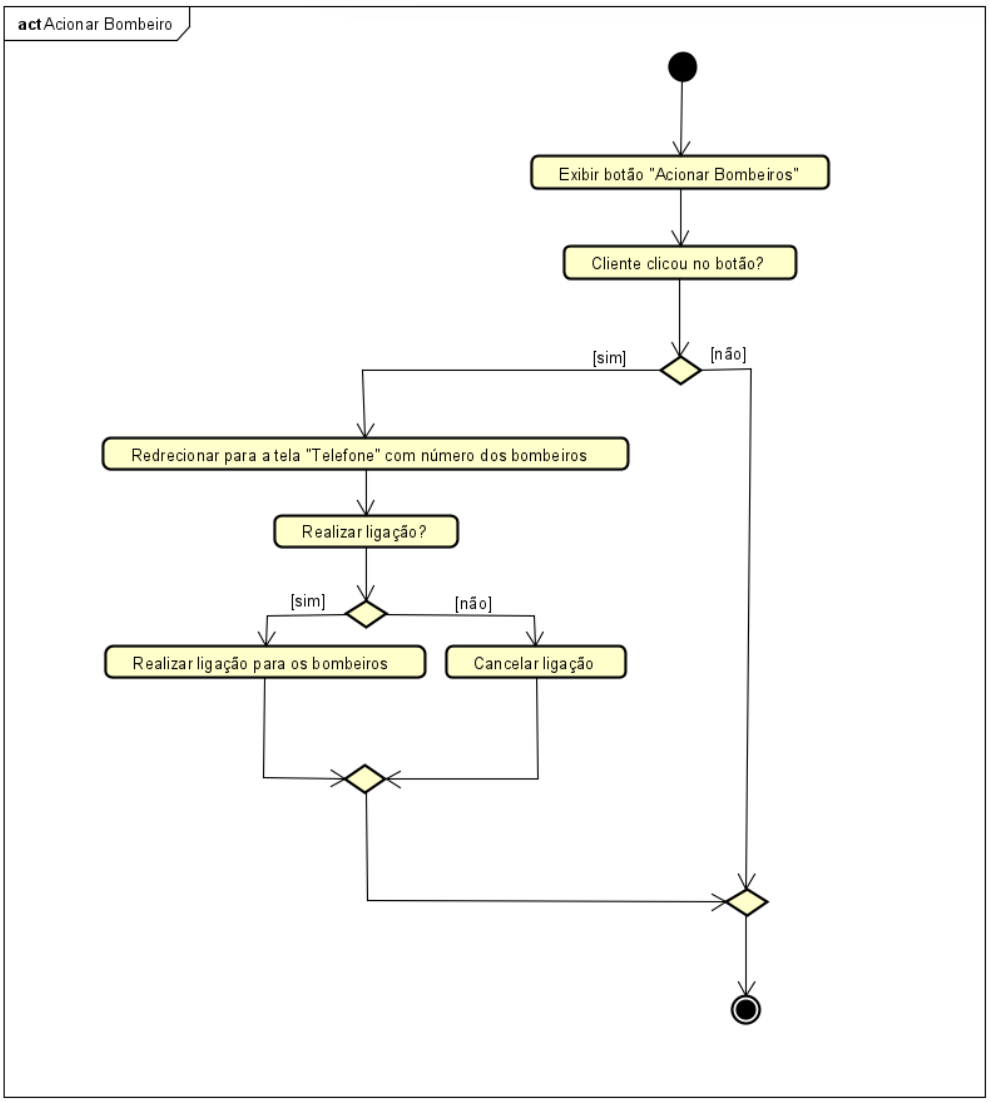


Fonte: (Elaborado pelo próprio autor, 2023)

3.3. Diagramas de Atividade

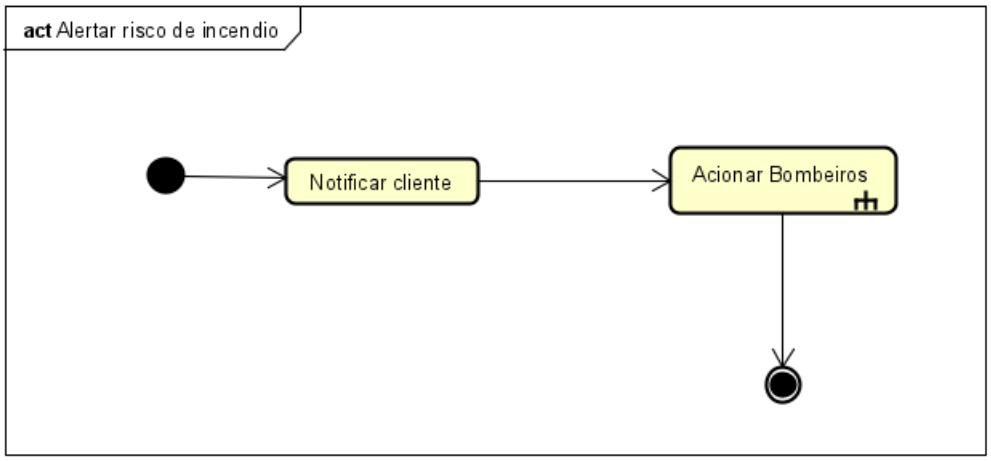
O fluxo de controle de cada funcionalidade do B. A. E. Guard pode ser analisado com os seguintes diagramas de atividade que demonstram seu passo a passo.

Figura 31 - Diagrama de Atividade "Acionar Bombeiros"



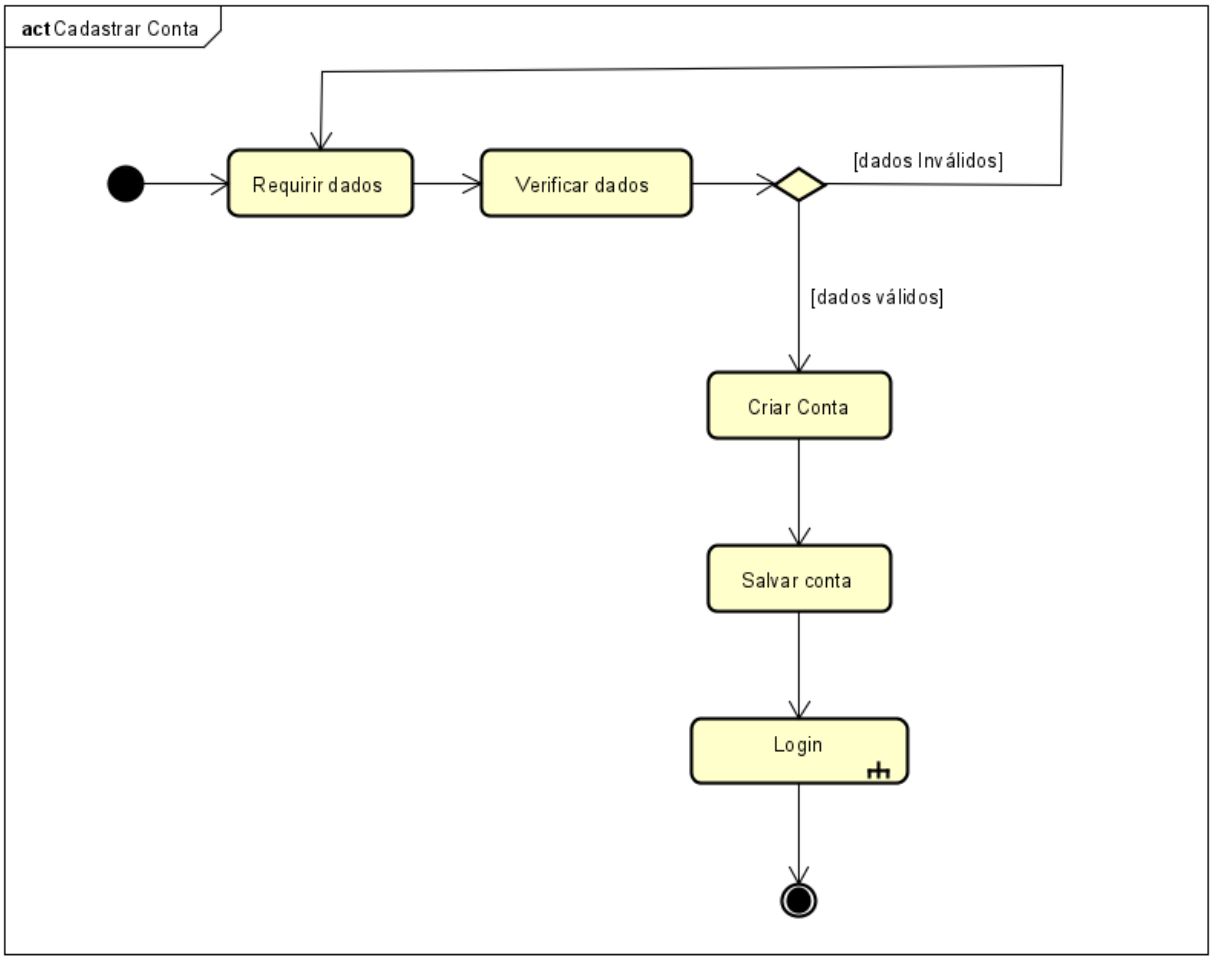
Fonte: (Elaborado pelo próprio autor, 2023)

Figura 32 - Diagrama de Atividade "Alertar risco de incêndio"



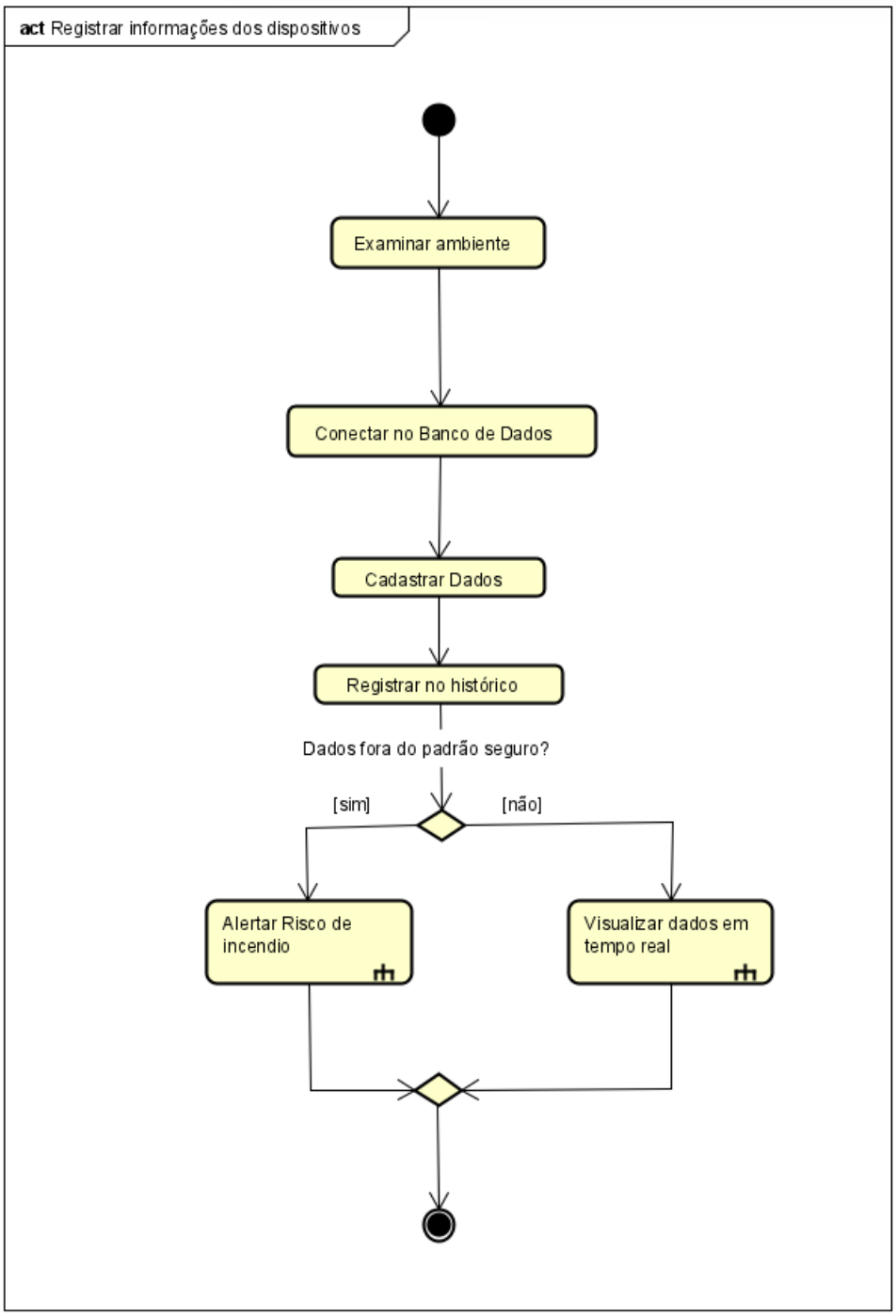
Fonte: (Elaborado pelo próprio autor, 2023)

Figura 33 - Diagrama de atividade “Cadastrar Perfil”



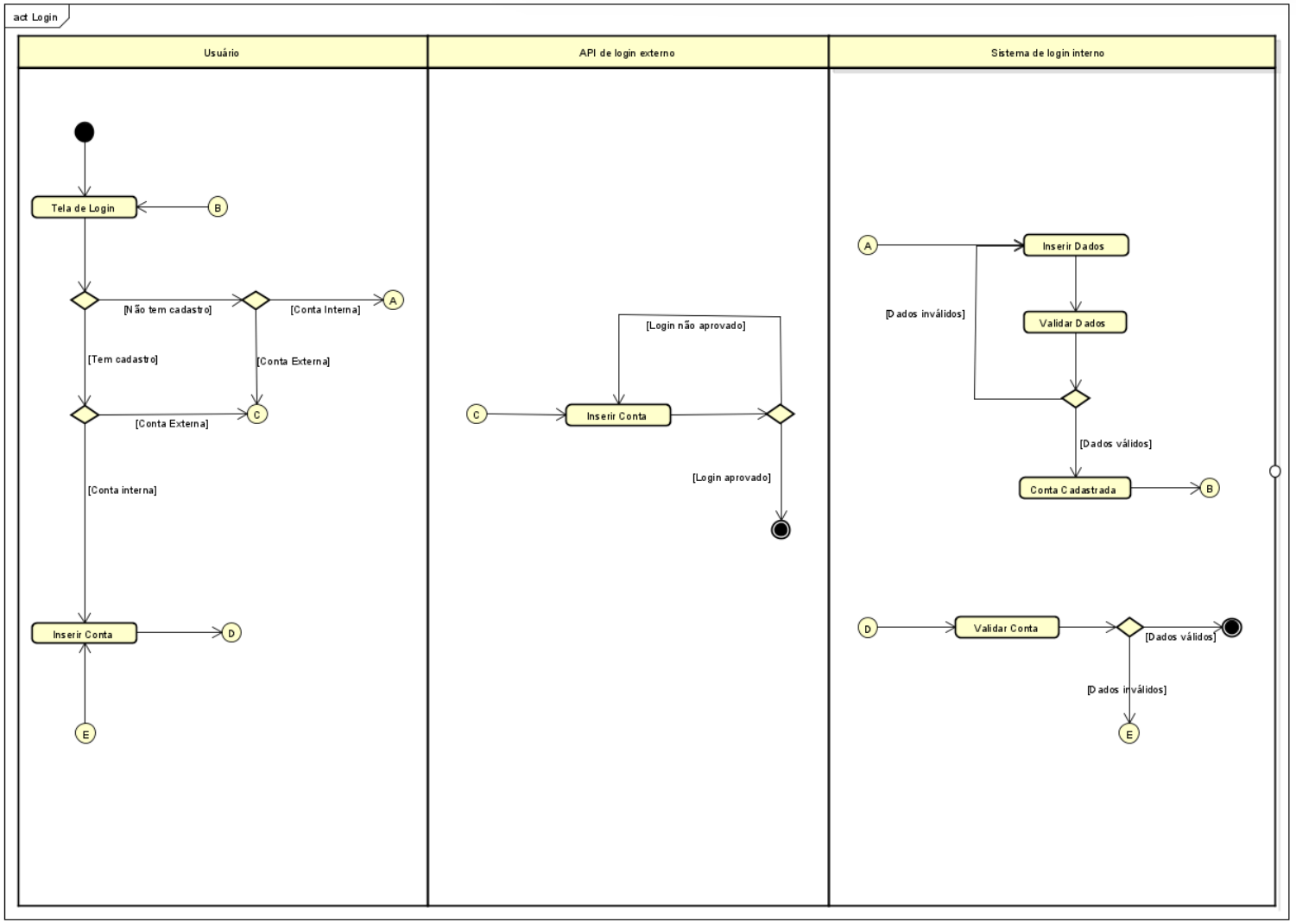
Fonte: (Elaborado pelo próprio autor, 2023)

Figura 34 - Registrar informações dos dispositivos



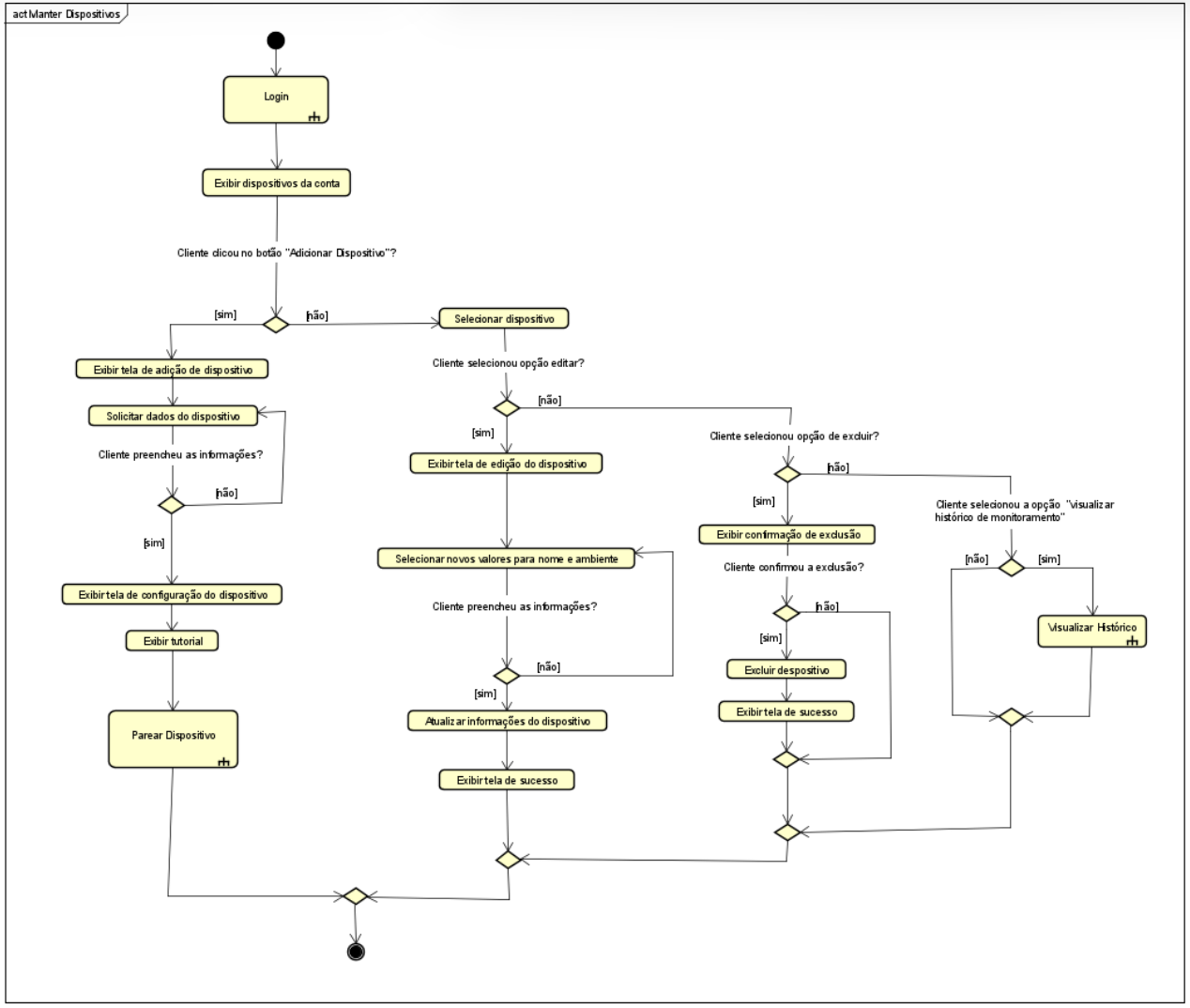
Fonte: (Elaborado pelo próprio autor, 2023)

Figura 35 - Diagrama de atividade “Login”



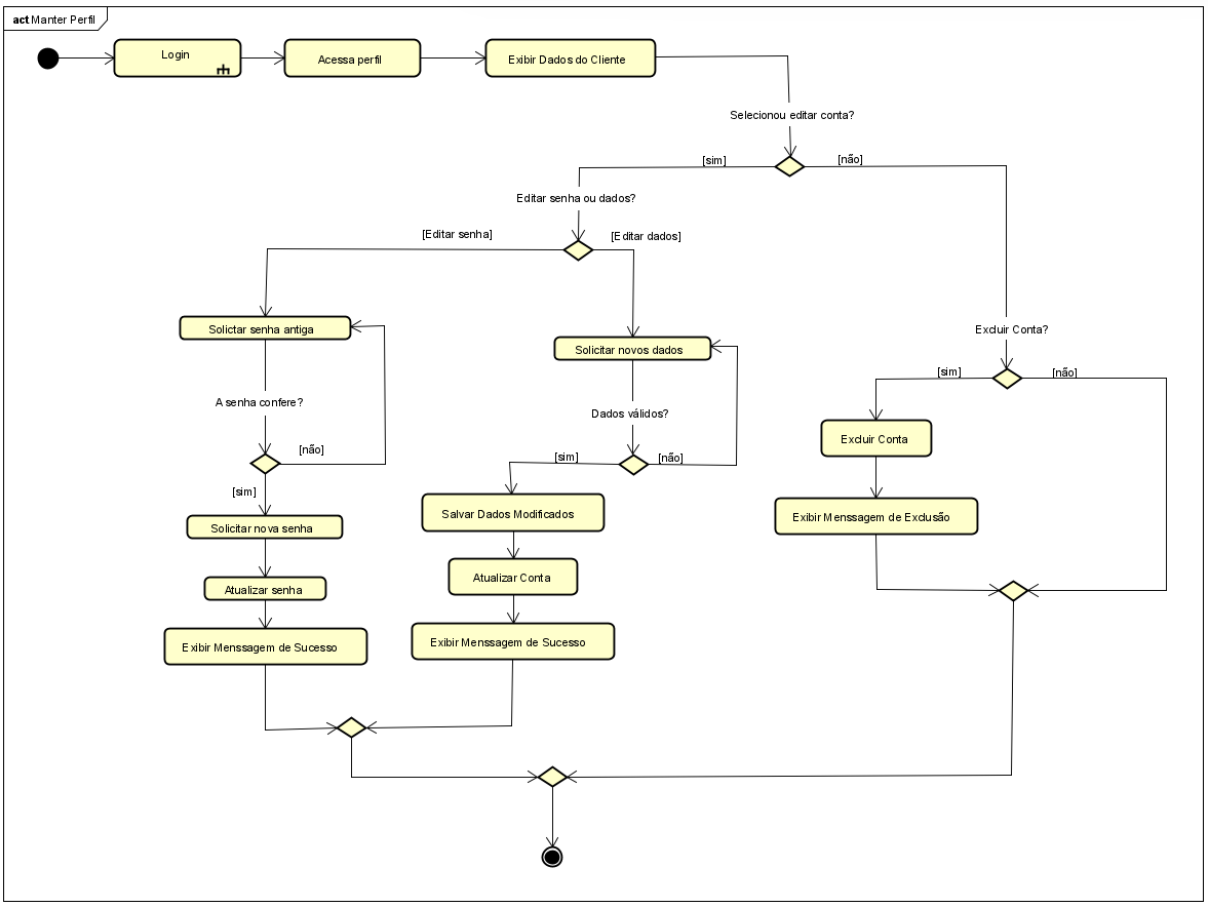
Fonte: (Elaborado pelo próprio autor, 2023)

Figura 36 - Diagrama de Atividade "Manter Dispositivos"



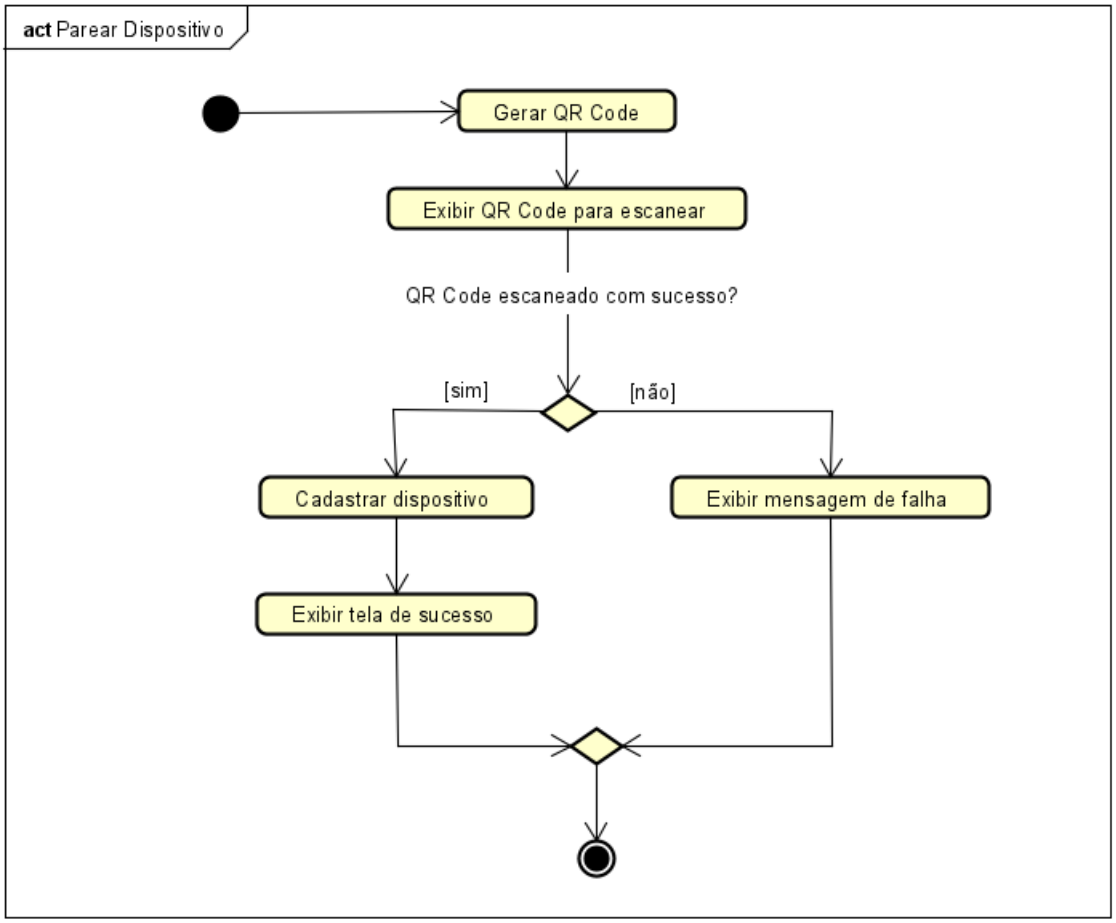
Fonte: (Elaborado pelo próprio autor, 2023)

Figura 37 - Diagrama de atividade “Manter Perfil”



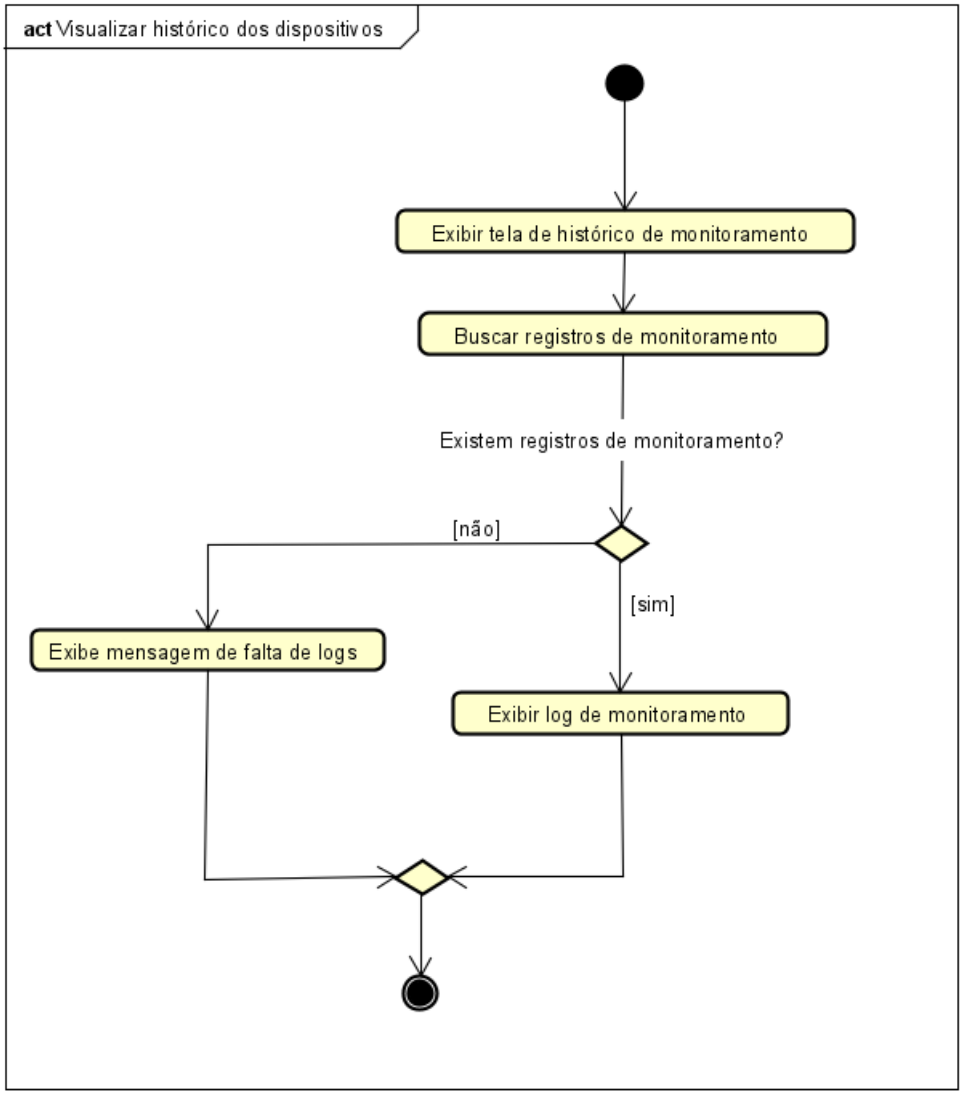
Fonte: (Elaborado pelo próprio autor, 2023)

Figura 38 - Diagrama de atividade “Parear Dispositivos”



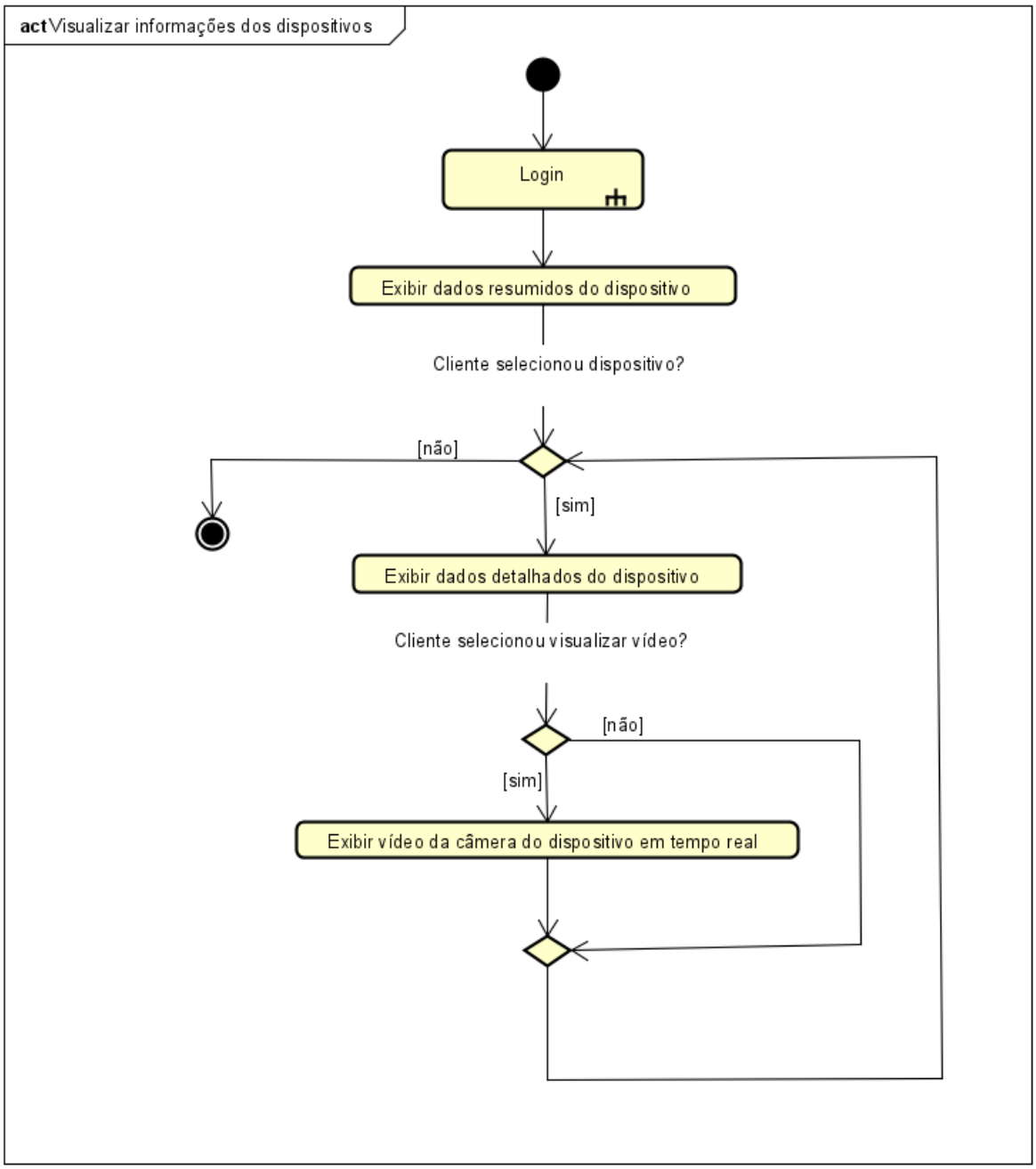
Fonte: (Elaborado pelo próprio autor, 2023)

Figura 39 - Diagrama de atividade “Visualizar histórico dos dispositivos”



Fonte: (Elaborado pelo próprio autor, 2023)

Figura 40 - Diagrama de atividade "Visualizar informações dos dispositivos"

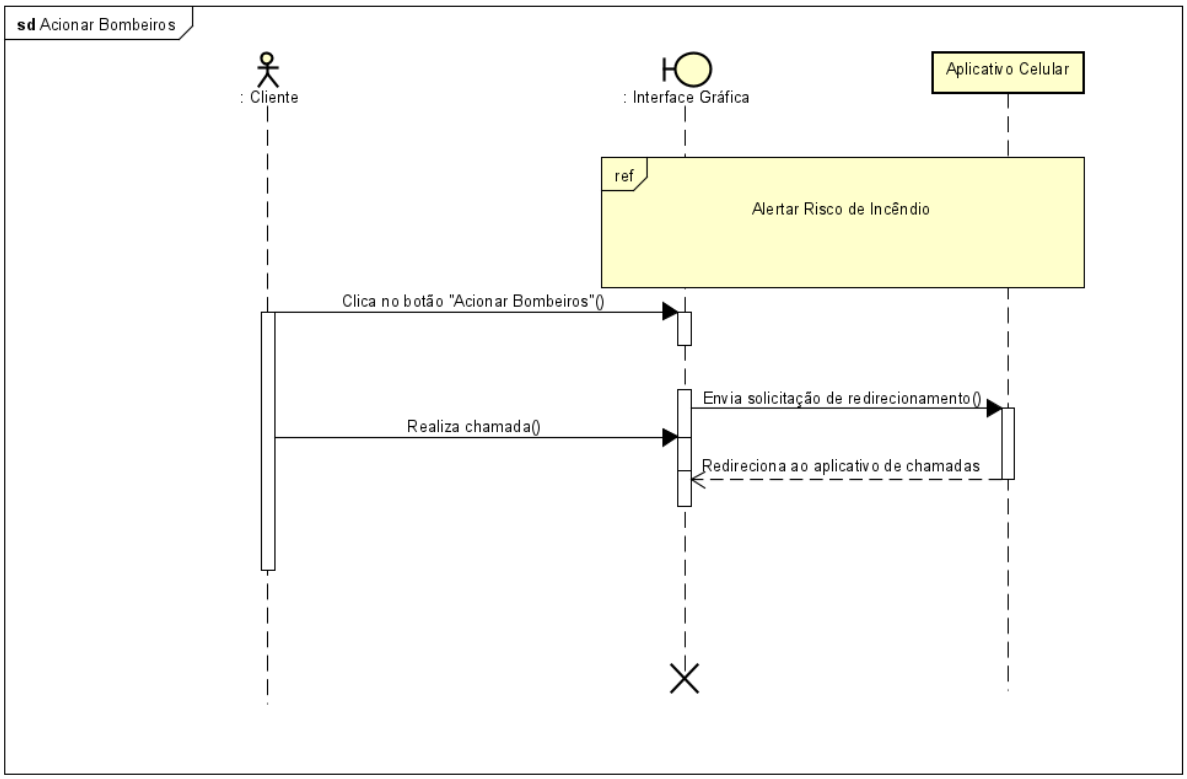


Fonte: (Elaborado pelo próprio autor, 2023)

3.4. Diagramas de Sequência

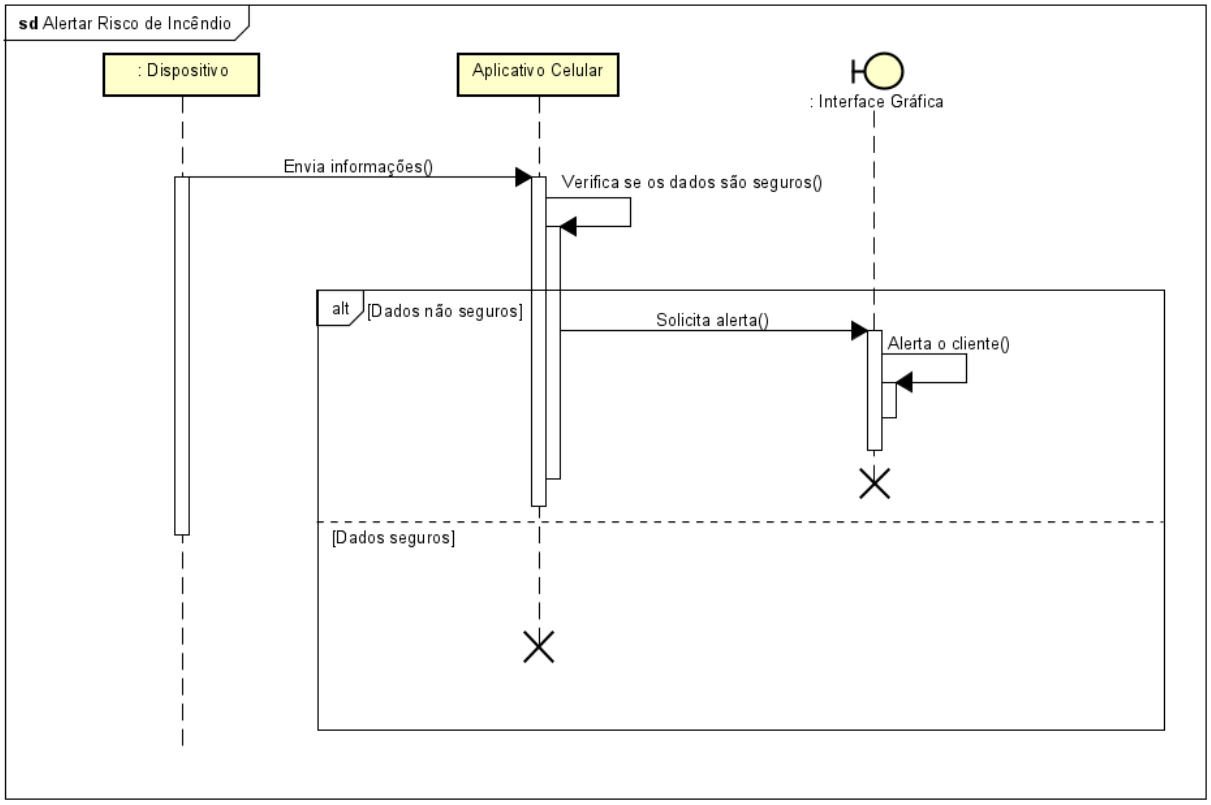
Como os componentes se organizam e interagem entre si ao longo de um determinado processo pode ser analisado através dos seguintes diagramas de sequência.

Figura 41 - Diagrama de sequência “Acionar Bombeiros”



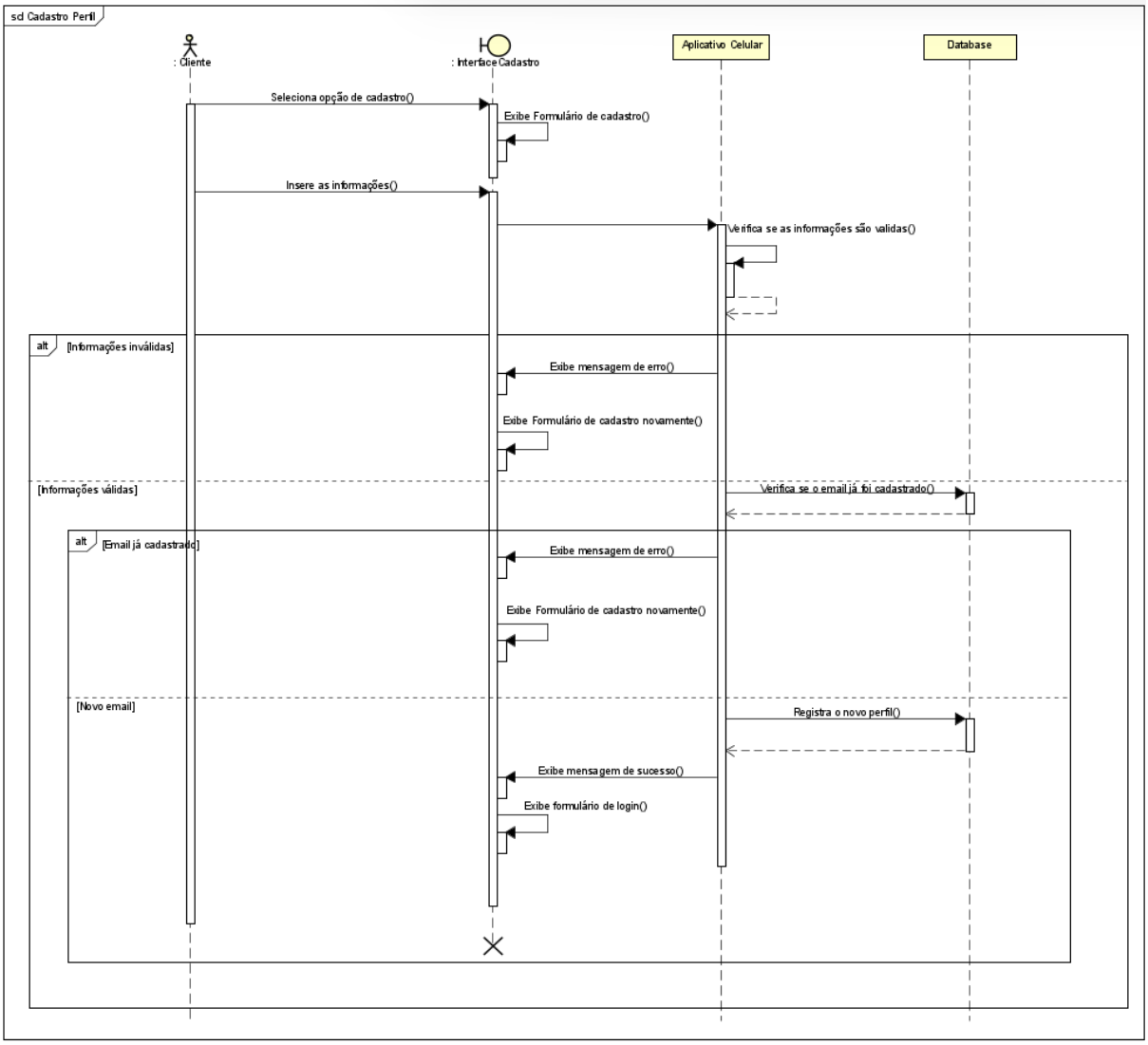
Fonte: (Elaborado pelo próprio autor, 2023)

Figura 42 - Diagrama de sequência “Alertar Risco de Incêndio”



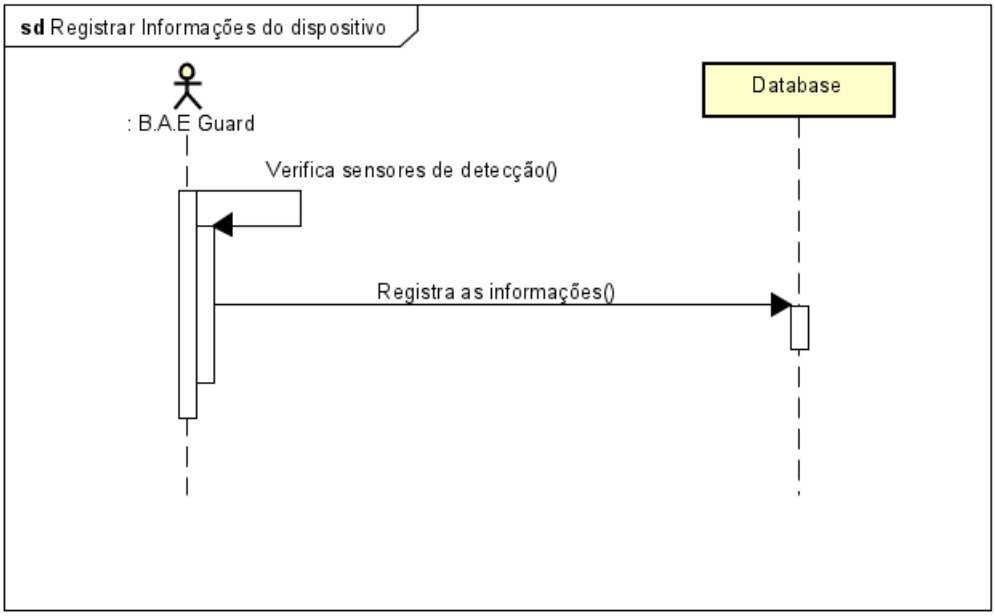
Fonte: (Elaborado pelo próprio autor, 2023)

Figura 43 - Diagrama de sequência “Cadastrar Perfil”



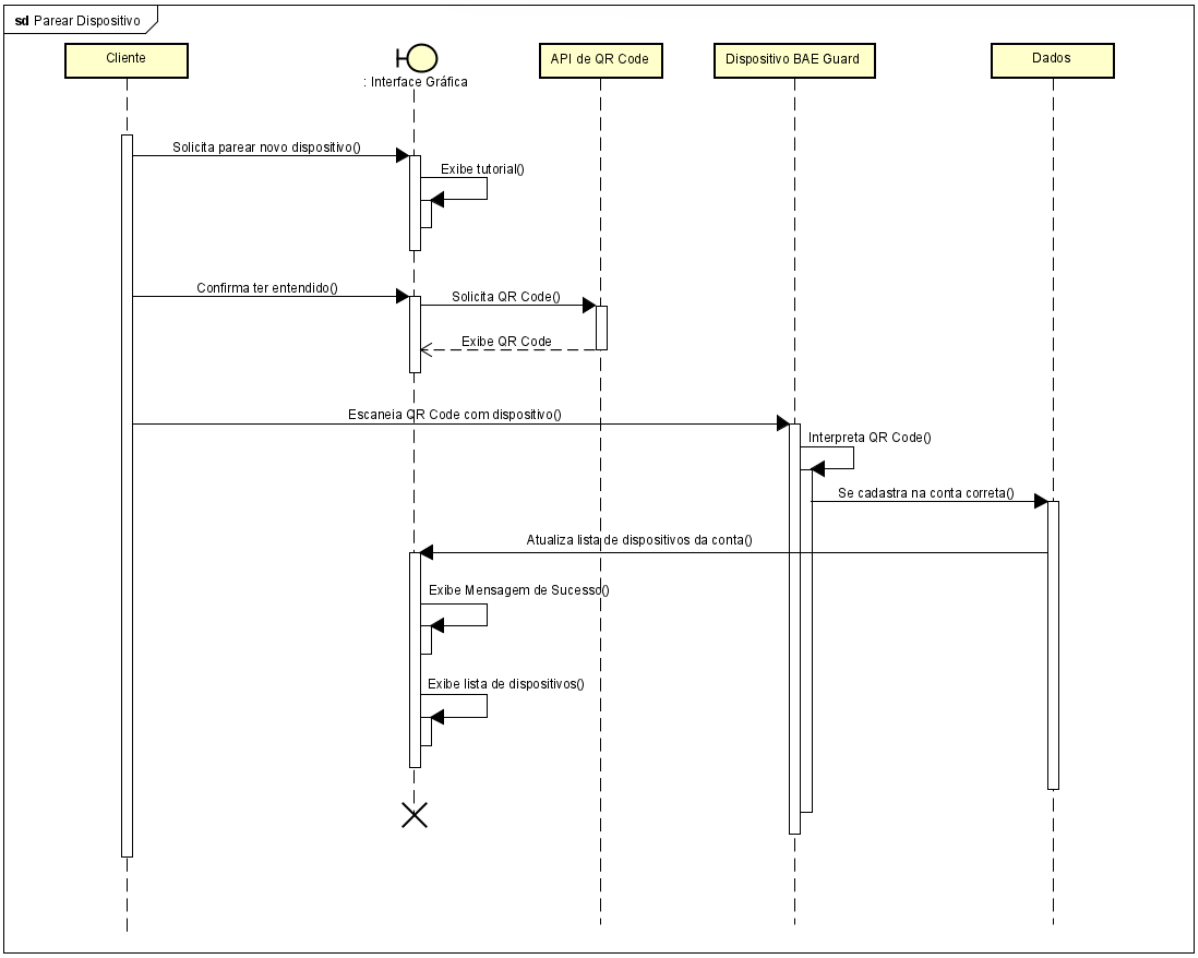
Fonte: (Elaborado pelo próprio autor, 2023)

Figura 44 - Diagrama de sequência “Registrar Informações do dispositivo”



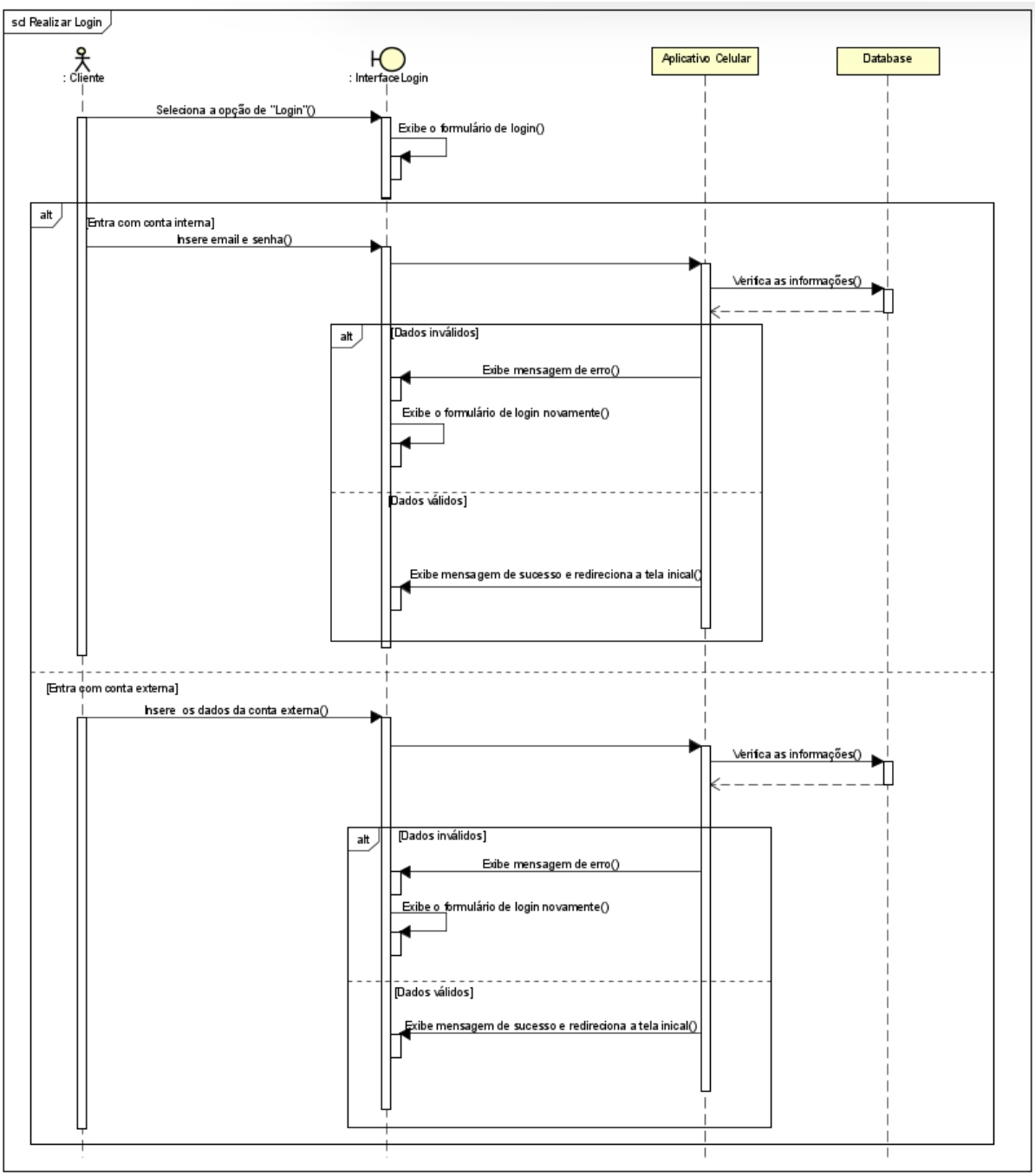
Fonte: (Elaborado pelo próprio autor, 2023)

Figura 45 - Diagrama de sequência “Parear Dispositivo”



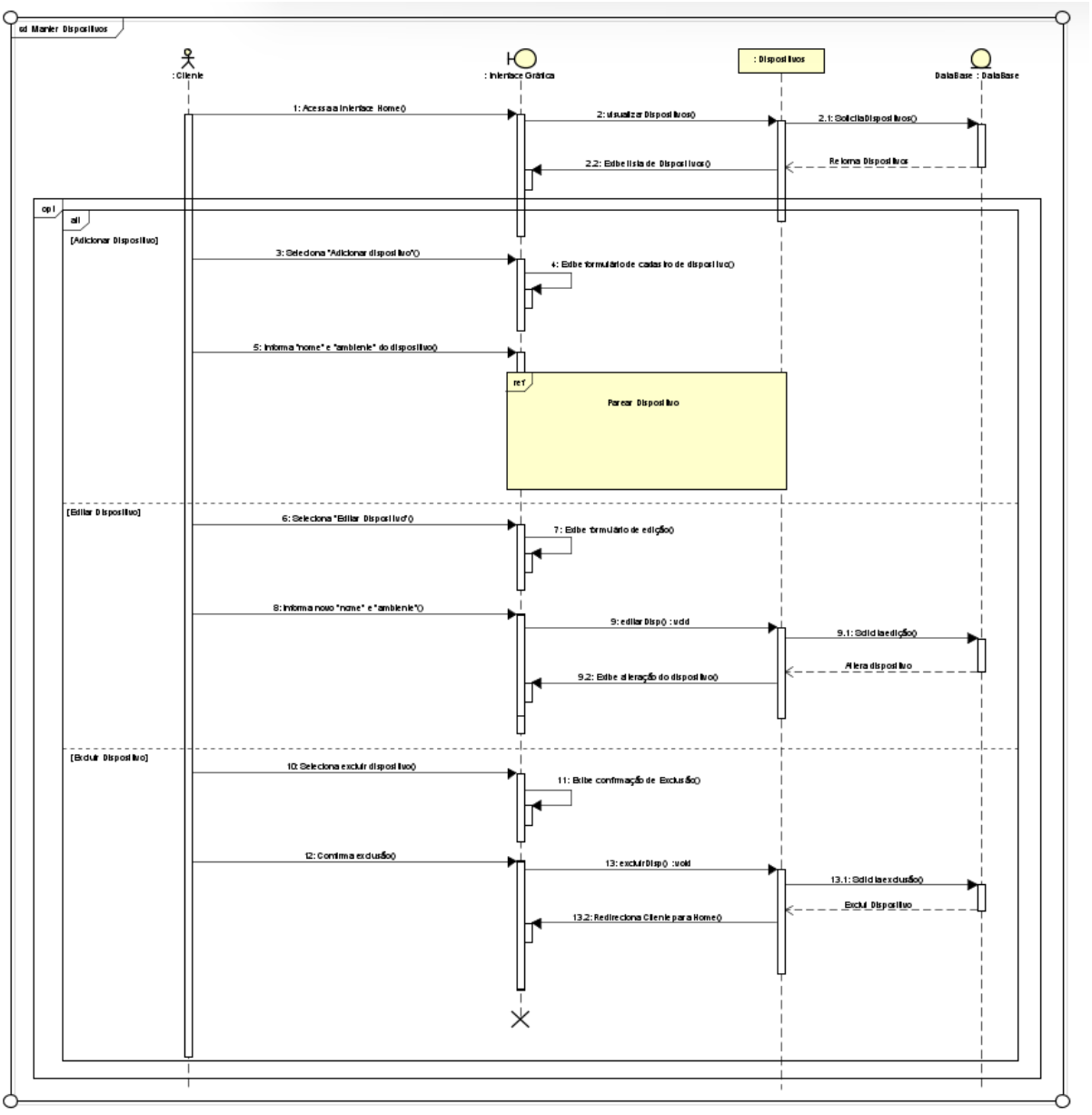
Fonte: (Elaborado pelo próprio autor, 2023)

Figura 46 - Diagrama de sequência “Realizar Login”



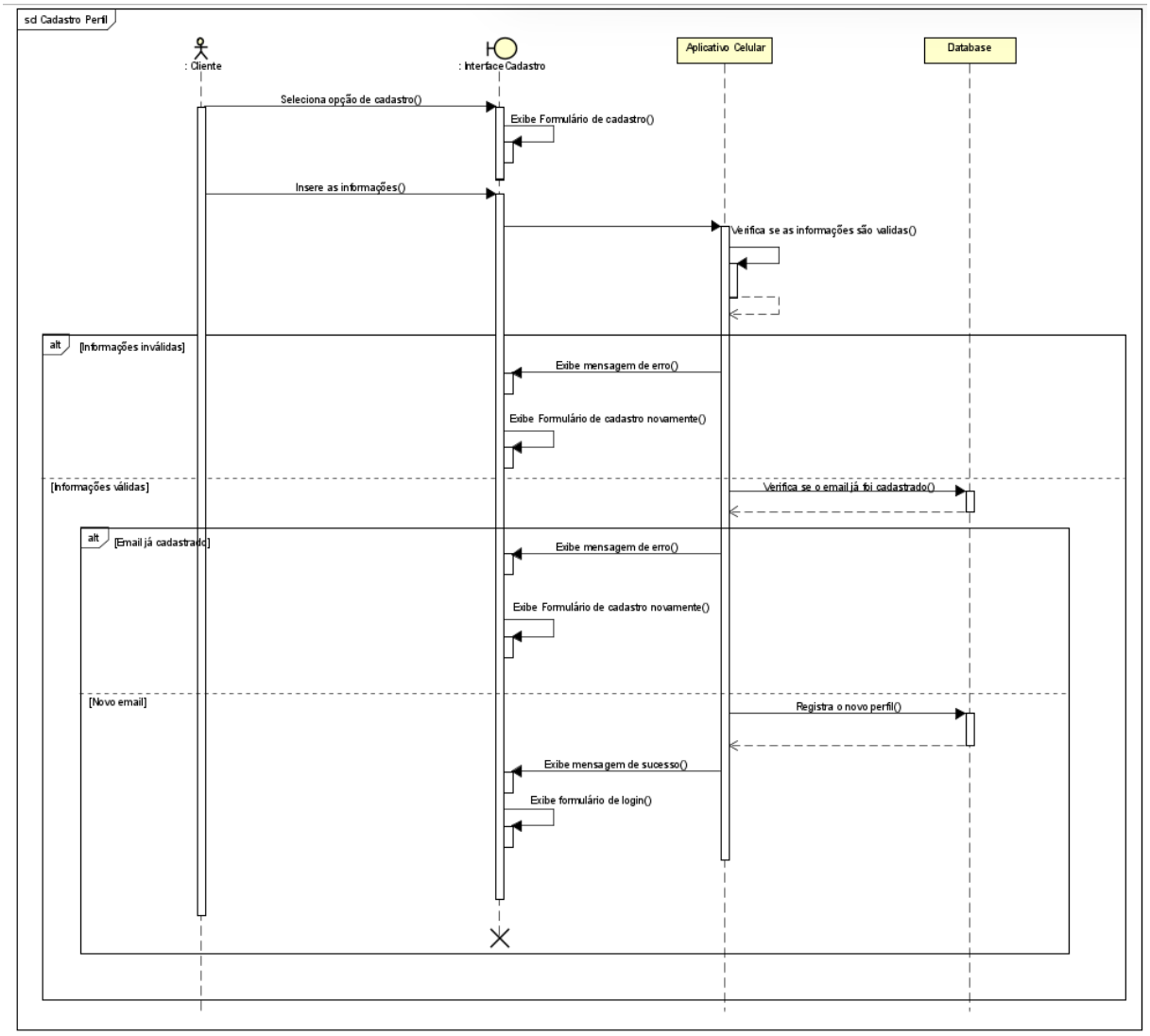
Fonte: (Elaborado pelo próprio autor, 2023)

Figura 47 - Diagrama de sequência “Manter Dispositivos”



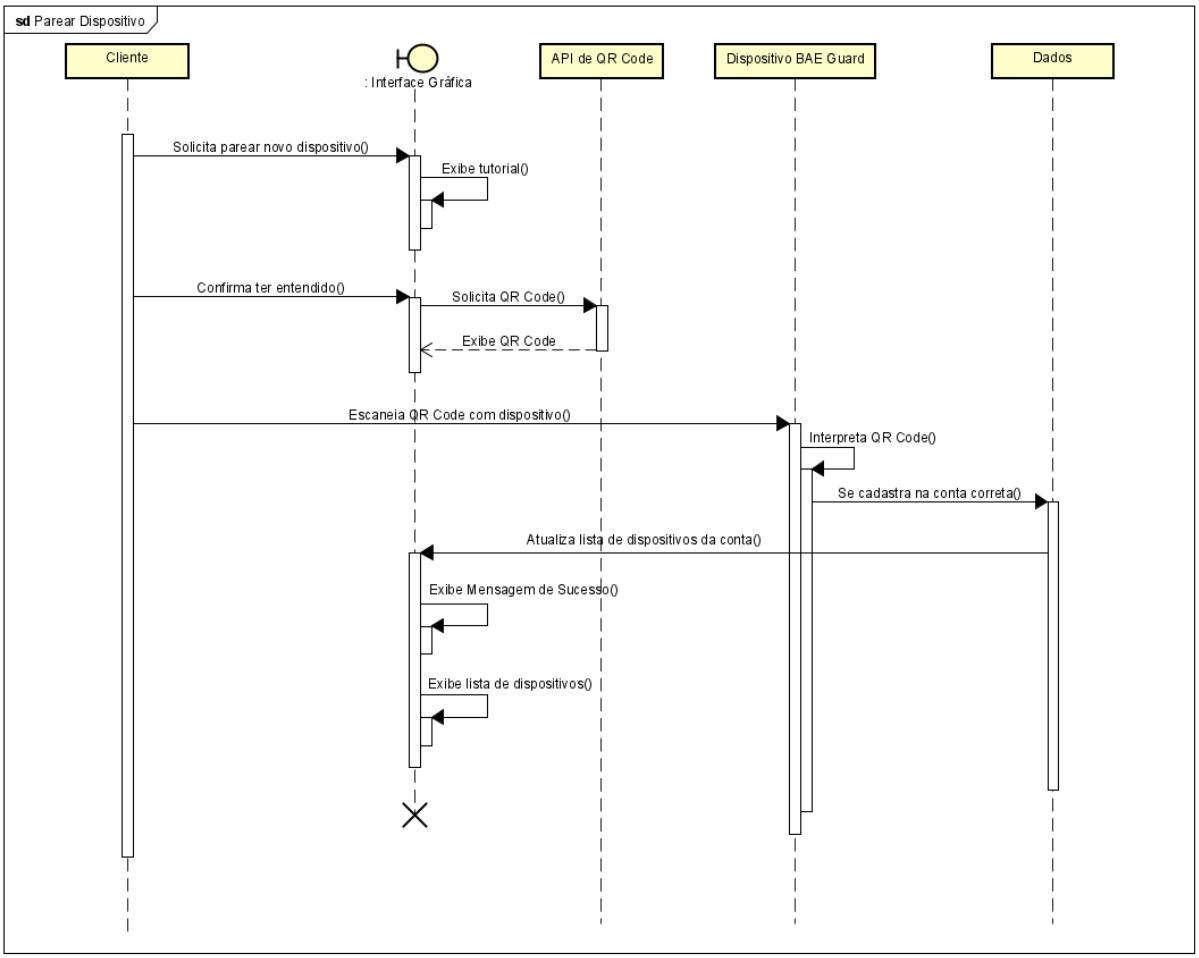
Fonte: (Elaborado pelo próprio autor, 2023)

Figura 48 - Diagrama de sequência “Cadastro do Perfil”



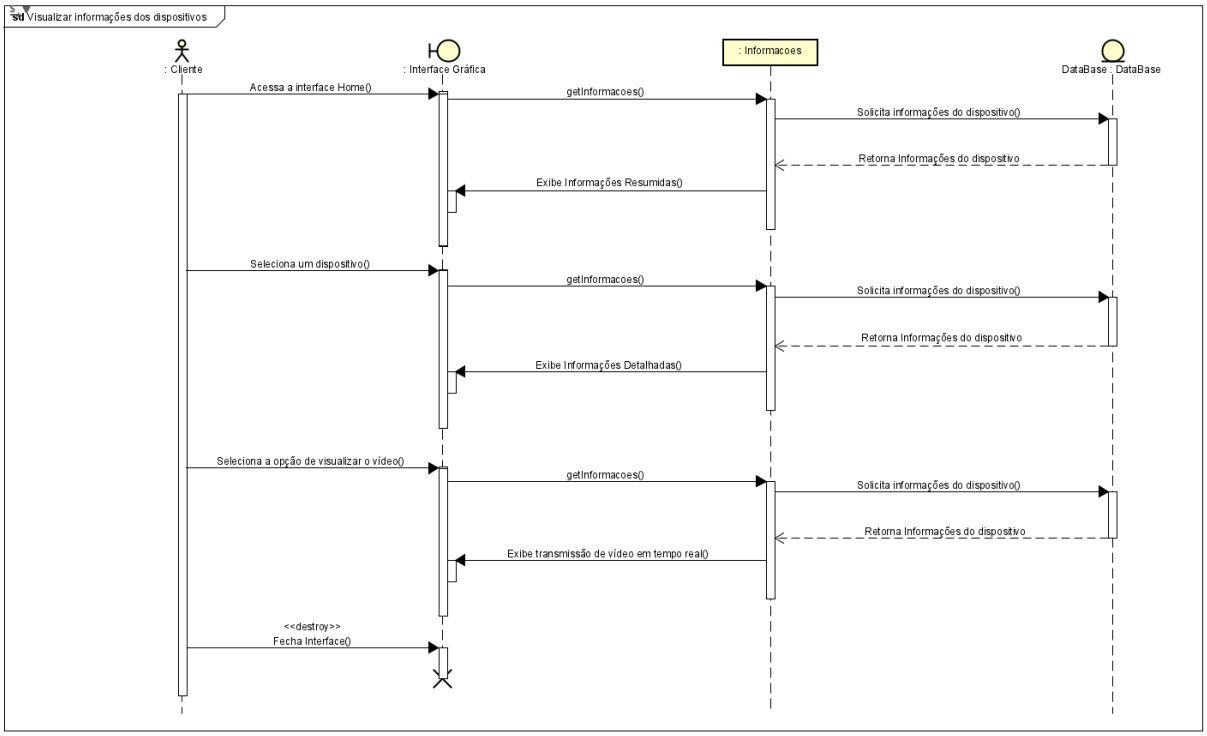
Fonte: (Elaborado pelo próprio autor, 2023)

Figura 49 - Diagrama de sequência “Parear Dispositivo”



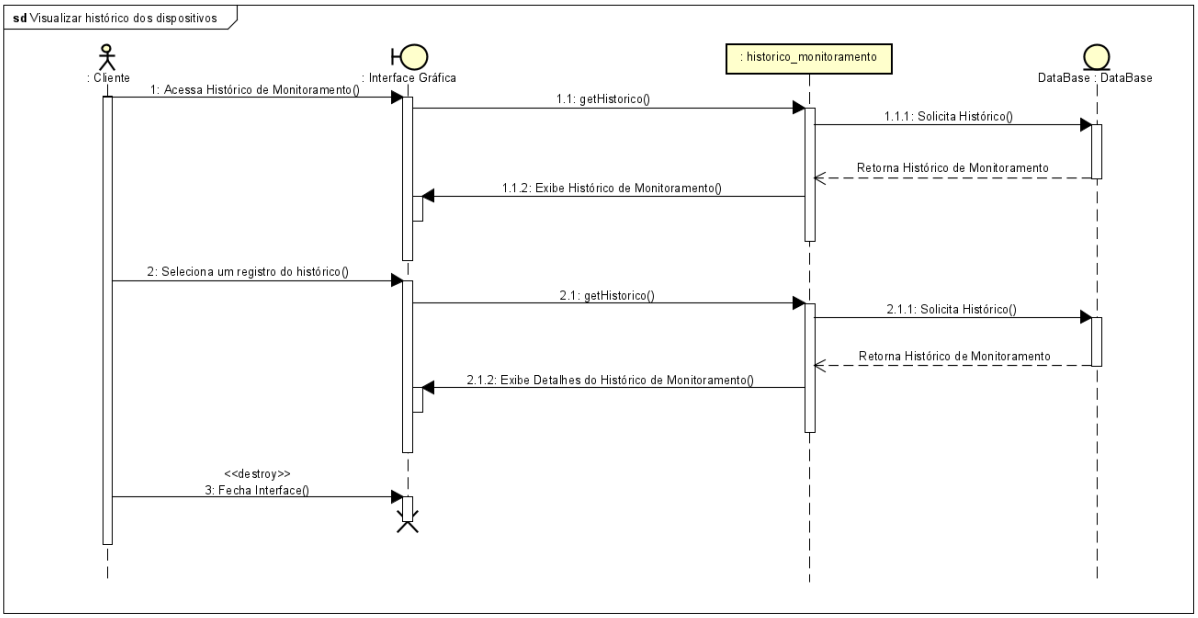
Fonte: (Elaborado pelo próprio autor, 2023)

Figura 50 - Diagrama de sequência “Visualizar informações dos dispositivos”



Fonte: (Elaborado pelo próprio autor, 2023)

Figura 51 - Diagrama de sequência “Visualizar histórico dos dispositivos ”



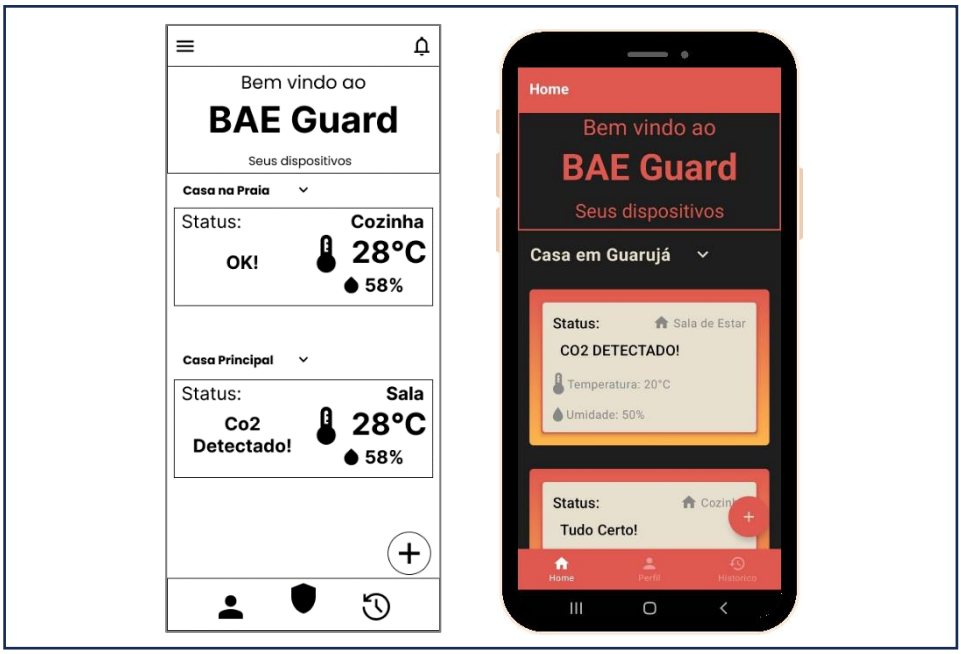
Fonte: (Elaborado pelo próprio autor, 2023)

3.5. Wireframes e Prototipagem das interfaces

De acordo com Lucidchart (2023), um wireframe é uma representação visual em tons de cinza que mostra a estrutura de uma página web ou aplicativo. Existem três categorias principais de wireframes: baixa, média e alta fidelidade (Mind Consulting, 2022). O software Figma foi usado para criar interfaces. A prototipagem de interface, segundo Yvonne Rogers (2013), é crucial para visualizar e testar a usabilidade antes da implementação. Cláudio Tavares (2013) afirma que protótipos em diferentes níveis de fidelidade melhoram o design e facilitam a comunicação. O protótipo da interface principal exibe dispositivos de monitoramento organizados em botões por ambiente, com informações como temperatura, umidade e status.

A primeira interface projetada foi a principal. Nela, cartões reúnem informações cruciais fornecidas pelo dispositivo de prevenção, garantindo uma visão rápida e suscinta do ambiente monitorado, tendo como resultado da figura 52.

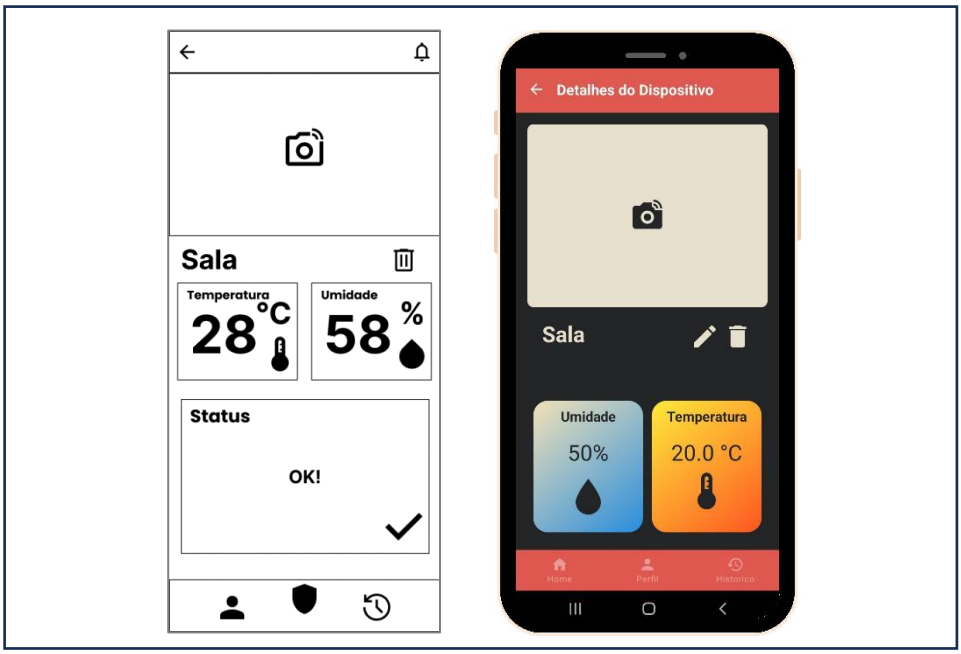
Figura 52 - Wireframe de baixa e alta fidelidade da interface “Home”



Fonte: (Elaborado pelo próprio autor, 2023)

Posteriormente, foi necessário desenvolver uma exibição detalhada do dispositivo com acesso a câmera, assim como a figura 53.

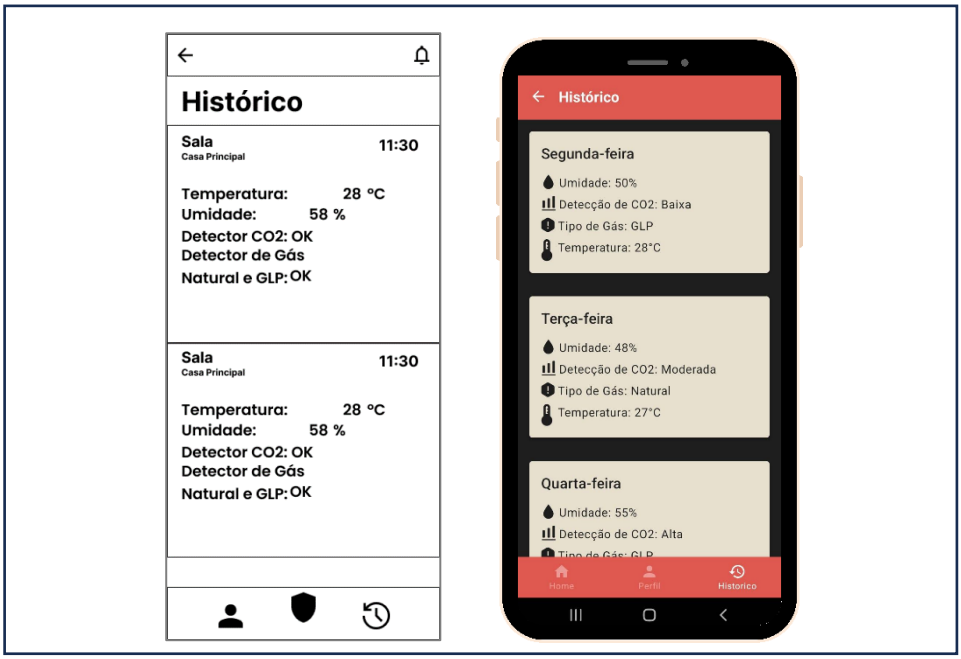
Figura 53 - Wireframe de baixa e alta fidelidade da interface “Dispositivo”



Fonte: (Elaborado pelo próprio autor, 2023)

Em seguida foi elaborada uma visualização para o histórico de detecção dos dispositivos onde se apresenta uma lista cronológica de ocorrências, como representado na figura 54.

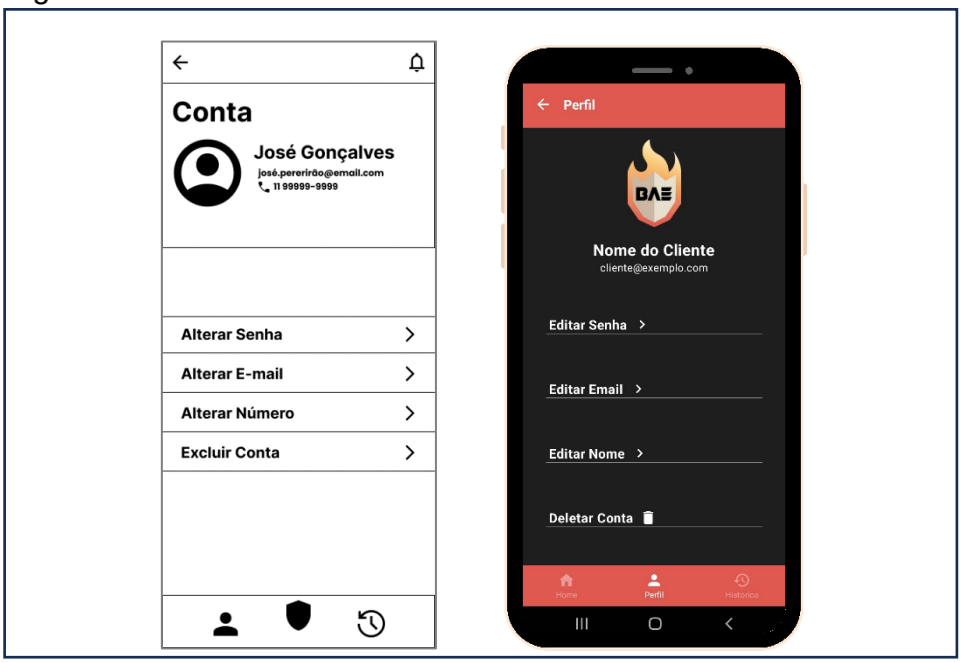
Figura 54 - Wireframe de baixa e alta fidelidade da interface “Histórico”



Fonte: (Elaborado pelo próprio autor, 2023)

Foi também desenvolvida uma visualização para os dados do perfil do usuário logado, comportando tanto perfis internos quanto das contas do Google.

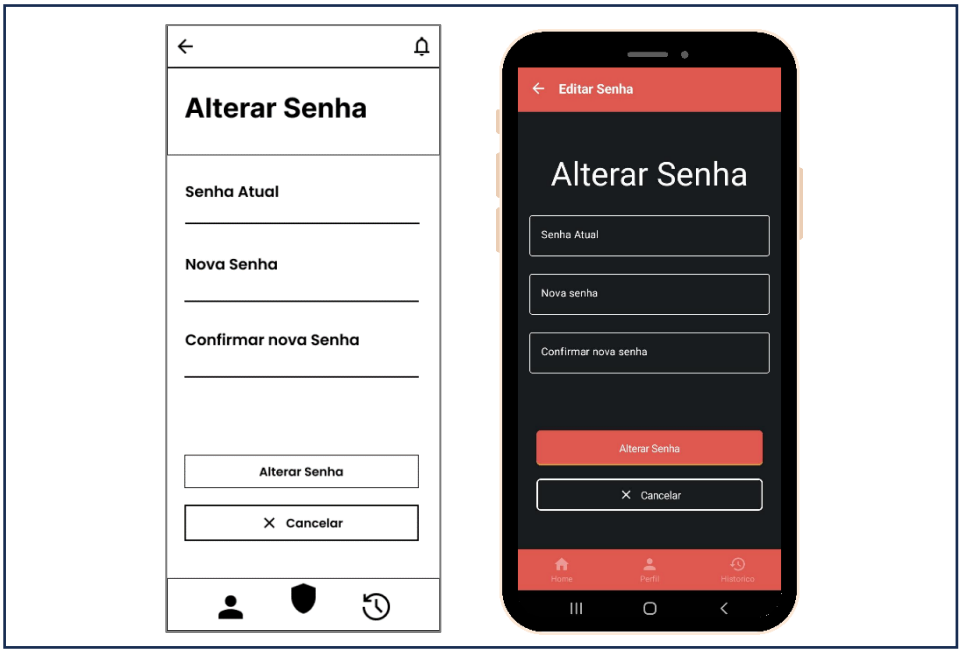
Figura 55 - Wireframe de baixa e alta fidelidade da interface “Perfil”



Fonte: (Elaborado pelo próprio autor, 2023)

Após isso, foram esboçadas as páginas de edição dos dados das contas. A primeira opção da visualização da conta leva ao seguinte formulário de alteração de senha.

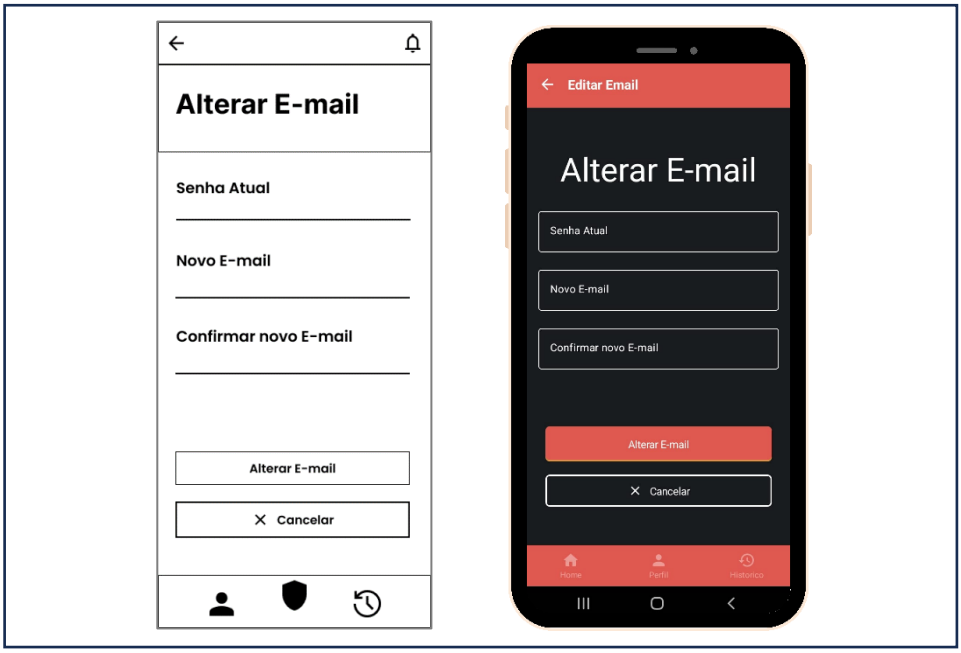
Figura 56 - Wireframe de baixa e alta fidelidade da interface “Alterar Senha”



Fonte: (Elaborado pelo próprio autor, 2023)

A segunda opção leva o usuário a um formulário de edição de e-mail que oferece aos usuários a possibilidade de atualizar o endereço de e-mail associado à sua conta.

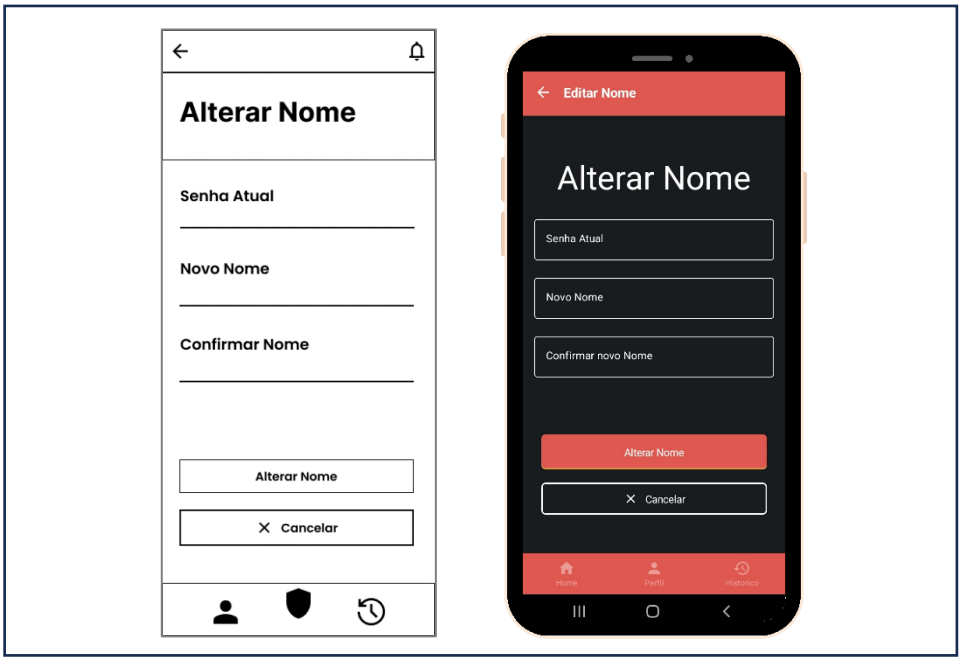
Figura 57 - Wireframe de baixa e alta fidelidade da interface “Alterar E-mail”



Fonte: (Elaborado pelo próprio autor, 2023)

Também é possível alterar o nome atrelado a conta por meio da terceira opção, seguindo o formulário da figura 58.

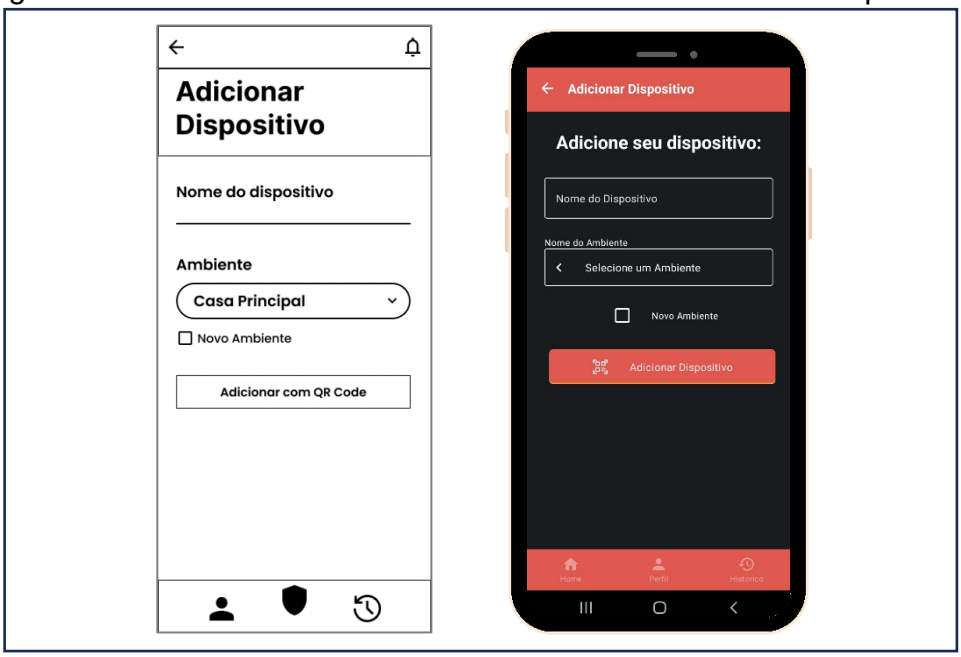
Figura 58 - Wireframe de baixa e alta fidelidade da interface “Alterar Nome”



Fonte: (Elaborado pelo próprio autor, 2023)

Prosseguindo com a interface para outras funcionalidades, foi desenvolvido um formulário para a adição de novos dispositivos à conta.

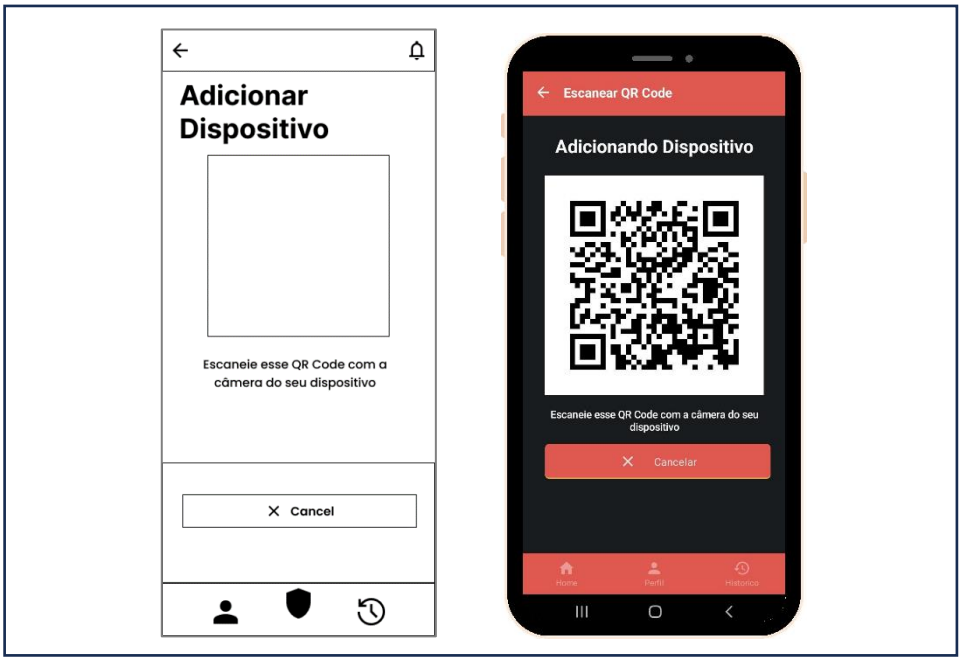
Figura 59 - Wireframe de baixa e alta fidelidade “Adicionar Dispositivo”



Fonte: (Elaborado pelo próprio autor, 2023)

Logo após o formulário, um código QR aparecerá em outra página, representada na figura 60.

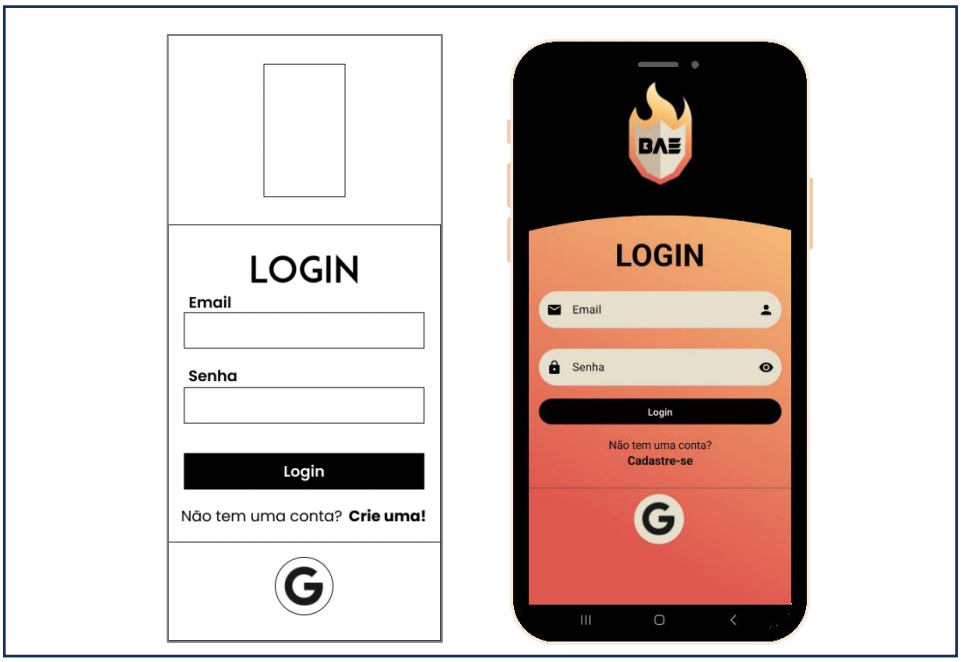
Figura 60 - Wireframe de baixa e alta fidelidade da interface “QR Code”



Fonte: (Elaborado pelo próprio autor, 2023)

Por fim foram elaborados os formulários de Login e criação de conta. Também há a opção de entrar com uma conta Google.

Figura 61 - Wireframe de baixa e alta fidelidade da interface “Login”



Fonte: (Elaborado pelo próprio autor, 2023)

Também é possível criar uma conta interna se o usuário não possuir uma, fornecendo um endereço de e-mail e uma senha no formulário da figura 62.

Figura 62 - Wireframe de baixa e alta fidelidade da interface “Cadastro”

Fonte: (Elaborado pelo próprio autor, 2023)

REFERÊNCIAS

BERTOLETI, P. (2019). Projetos com ESP-32 e LoRa. 1. ed. [S.l.]:. São Paulo: Instituto NCB.

BOOCH, Grady; RUMBAUGH, James; JACOBSON, Ivar. (2006). UML Guia do Usuário. 2.ed. Rio de Janeiro: Elsevier Editora Ltda.

BOOCH, G., RUMBAUGH, J., JACOBSON, I. (2012). UML: Guia do Usuário. 12ª.ed. Rio de Janeiro: Elsevier Editora Ltda.

FOWLER, Martin. (2003). UML Distilled. 3.ed. Sydney: Addison-Wesley.

GUEDES, Gilleanes T. A. (2011). UML 2. 2.ed. São Paulo: Novatec Editora Ltda.

GUEDES, Gilleanes T. A. (2018). UML 2. 3.ed. São Paulo: Novatec Editora Ltda.

LARMAN, Craig. (2006).Utilizando UML e Padrões. 3.ed. Porto Alegre: ARTMED Editora S. A.

ROGERS, Yvonne; SHARP, Helen; PREECE, Jenny. (2013). Design de Interação: Além da Interação Humano-Computador. 2. ed. Porto Alegre: Bookman.

TAVARES, Cláudio Rocha. (2004). Tipografia & Design Gráfico: Design e Produção de Impressos e Livros. São Paulo: Editora Rosari.

ESPRESSIF. (2019). ESP32 Series: Datasheet. Fonte: <https://pdf1.alldatasheet.com/datasheet-pdf/view/1148023/ESPRESSIF/ESP32.html>

SANTOS, B. P. (2016). Internet das coisas: da teoria à prática. Sociedade Brasileira de Computação (SBC).

SNATAELLA, L. (2013). Revista GeMinis ano 4 - n. 2 - v. 1 | p. 19 - 32. Revista GEMInIS.

BARBOSA, I. (9 de Janeiro de 2023). CronoShare. Fonte: : https://www.cronoshare.com.br/quanto-custa/sistema-incendios#:~:text=O%20preço%20de%20instalação%20de%20sistemas%20de%20combate%20a%20incêndio,mais%20de%20R%24%2030.000%20reais.

Corpo de Bombeiros do Estado de São Paulo. (2022). Portal do Governo. Fonte: Secretaria de Segurança Pública do Estado de São Paulo: http://www.ssp.sp.gov.br/Estatistica/CorpoBombeiro.aspx

Instituto Sprinkler Brasil. (25 de Novembro de 2022). Instituto Sprinkler Brasil. Fonte: https://sprinklerbrasil.org.br/imprensa/noticias-de-incendios-estruturais-aumentam-86-em-outubro/

LOCATELLI, C. (13 de Janeiro de 2023). Curtocircuito. Fonte: Curtocircuito: https://curtocircuito.com.br/blog/Categoria%20IoT/conhecendo-esp32

Lucid Software Inc. (s.d.). *O que é wireframe?* Fonte: Lucidchart: https://www.lucidchart.com/pages/pt/o-que-e-wireframe

Mind Consulting. (26 de 11 de 2022). *O que exatamente é um wireframe? Um guia completo para 2022*. Fonte: Mind Group Technologies: https://mindconsulting.com.br/2021/11/o-que-exatamente-e-wireframe-um-guia-completo-para-2022/

Nacional Fire Protection Association (2023). Acesso em 18 de Junho de 2023, disponível em: https://www.nfpa.org

OLIVEIRA, J. (10 de Abril de 2020). xprojetos. Fonte: xprojetos.: https://xprojetos.net/esp32-e-suas-versoes/

ROBOCORE. Buzzer Ativo. [2020]. Disponível em: https://www.robocore.net/atuador-rele/modulo-buzzer-5v-ativo. Acesso em: 18 junho de 2023.

ROBOCORE. Display. [2020]. Disponível em: https://www.robocore.net/display/lcd-20x4-5v-branco-no-azul. Acesso em: 18 junho de 2023.

ROBOCORE. ESP32 - WiFi + Bluetooth. [2020]. Disponível em:

https://www.robocore.net/wifi/esp32-wifi-bluetooth. Acesso em: 18 junho de 2023.

ROBOCORE. ESP32-CAM - ESP32 com Câmera. [2020]. Disponível em:

https://www.robocore.net/wifi/esp32-cam-esp32-com-camera. Acesso em: 18 junho de 2023.

ROBOCORE. Protoboard. [2020]. Disponível em:

https://www.robocore.net/protoboard/protoboard-830-pontos. Acesso em: 18 junho de 2023.

ROBOCORE. Sensor de Gás GLP. [2020]. Disponível em:

https://www.robocore.net/sensor-gas/sensor-de-glp-mq-5. Acesso em: 18 junho de 2023.

ROBOCORE. Sensor de Gás MQ-7. [2020]. Disponível em:

https://www.robocore.net/sensor-gas/modulo-mq-7-sensor-de-monoxido-de-carbono. Acesso em: 18 junho de 2023.

ROBOCORE. Sensor de Temperatura DHT11. [2020]. Disponível em:

https://www.robocore.net/sensor-ambiente/sensor-de-temperatura-dht11. Acesso em: 18 junho de 2023.

ROBOCORE. Sensor de chama infravermelho. [2020]. Disponível em:

https://www.robocore.net/sensor-ambiente/sensor-de-chama. Acesso em: 18 junho de 2023.

Santos, B. P. (2016). Internet das coisas: da teoria à prática. Sociedade Brasileira de Computação (SBC).

SOUZA, F. (21 de Abril de 2020). BBC. Fonte: BBC News Brasil: https://www.bbc.com/portuguese/brasil-52363220

SOUZA, F. (28 de Junho de 2016). Embarcados. Fonte: Embarcados: https://embarcados.com.br/esp32-o-sucessor-do-esp8266/

UML. (01 de Setembro 2023). Unified Modeling Language. Disponível em: http://www.uml.org/. Acesso em: 01 setembro de 2023.