**CENTRO ESTADUAL DE EDUCAÇÃO TECNOLÓGICA PAULA SOUZA DA ETEC DA ZONA LESTE**

**Mtec Desenvolvimento de Sistemas AMS**

**Murillo Castro De Jesus**

**Rafael Ferreira Lopes**

**Vinicius Rafael Rodrigues**

**Vitor Daisuke Iwamoto**

**EYE TRUCK: Sistema de Prevenção de Colisão com Viadutos e Pontes**

**São Paulo**

**2025**

**Murillo Castro De Jesus**

**Rafael Ferreira Lopes**

**Vinicius Rafael Rodrigues**

**Vitor Daisuke Iwamoto**

**EYE TRUCK: Sistema de Prevenção de Colisão com Viadutos e Pontes**

Trabalho de Conclusão de Curso apresentado ao Curso Ensino Médio com Habilitação Profissional de Técnico em Desenvolvimento de Sistemas AMS da Etec Zona Leste, orientado pelo Professor Jeferson Roberto de Lima, como requisito parcial para obtenção do título de técnico em Desenvolvimento de Sistemas.

**São Paulo**

**2025**

**Resumo**

O projeto propõe desenvolver um dispositivo IoT com a finalidade de evitar colisões de veículos de transporte de cargas em viadutos. O sistema é formado por um dispositivo com hardware e uma aplicação web. O software será utilizado para o usuário informar a altura do caminhão. O hardware compara a altura do caminhão com a altura máxima permitida, indicada na placa do viaduto captado por uma câmera. Com essa comparação será possível emitir sinais alertando o condutor do caminhão sobre a altura do veículo em relação à altura do viaduto, podendo evitar uma possível colisão por falta de atenção do motorista.

**Palavras-chave**: Internet das Coisas (IoT); Altura máxima permitida; Reconhecimento de placas; Prevenção de colisões; Veículos de carga.

**Abstract**

The project proposes to develop an IoT device to prevent collisions between cargo vehicles on overpasses. The system consists of a hardware device and a web application. The software will be used to inform the user of the height of the truck. The hardware compares the height of the truck with the maximum permitted height, indicated on the overpass sign captured by a camera. With this comparison, it will be possible to emit signals alerting the truck driver about the height of the vehicle in relation to the height of the overpass, thus avoiding a possible collision due to the driver's lack of attention.

**Keywords**: Internet of Things (IoT); Maximum allowed height; Sign recognition; Collision prevention; Cargo vehicles.

**LISTA DE ILUSTRAÇÕES**

[Figura 1 - Raspberry PI 3, Modelo B 13](#_Toc210851549)

[Figura 2 - Fluxo de um sistema baseado em visão computacional 14](#_Toc210851550)

[Figura 3 - Trecho do código que realiza a captura da face 15](#_Toc210851551)

[Figura 4 - Imagem da face alinhada com marcação nos olhos 16](#_Toc210851552)

[Figura 5 - Imagem da região de interesse 16](#_Toc210851553)

[Figura 6 - Exemplo da tecnologia YOLO em ambiente urbano 17](#_Toc210851554)

[Figura 7 - Abordagem tradicional 20](#_Toc210851555)

[Figura 8 - Abordagem do aprendizado de máquina 20](#_Toc210851556)

[Figura 9 - Adaptando-se automaticamente à mudança 21](#_Toc210851557)

[Figura 10 - Exemplo do código em Python 22](#_Toc210851558)

[Figura 11 - Resultado do código em Python 23](#_Toc210851559)

[Figura 12 - Outro resultado do código em Python 23](#_Toc210851560)

[Figura 13 - Funcionamento do Firebase 25](#_Toc210851561)

[Figura 14 - Explicação de um Dado 26](#_Toc210851562)

[Figura 15 - Instalação da biblioteca EasyOCR 28](#_Toc210851563)

[Figura 16 - Exemplo de código em JavaScript 31](#_Toc210851564)

[Figura 17 – Exemplo de Caso de Uso 33](#_Toc210851565)

[Figura 18 - Diagrama de atividade Fazer Login 34](#_Toc210851566)

[Figura 19 – Diagrama de sequência Fazer Login 35](#_Toc210851567)

[Figura 20 - Diagrama de máquina de estado Fazer Login 36](#_Toc210851568)

[Figura 21 - Diagrama de Casos de Uso 39](#_Toc210851569)

[Figura 22 - Diagrama de Atividade: Cadastrar Conta 56](#_Toc210851570)

[Figura 23 - Diagrama de Atividade: Fazer Login 57](#_Toc210851571)

[Figura 24 - Diagrama de Atividade: Consultar Perfil 58](#_Toc210851572)

[Figura 25 - Diagrama de Atividade: Alterar Informações Do Perfil 59](#_Toc210851573)

[Figura 26 - Diagrama de Atividade: Enviar Altura Do Caminhão 60](#_Toc210851574)

[Figura 27 - Diagrama de Atividade: Reconhecer Placa 61](#_Toc210851575)

[Figura 28 - Diagrama de Atividade: Retirar Altura 62](#_Toc210851576)

[Figura 29 - Diagrama de Atividade: Comparar Altura 63](#_Toc210851577)

[Figura 30 - Diagrama de Atividade: Exibir Alerta 64](#_Toc210851578)

[Figura 31 - Diagrama de Atividade: Conectar Com o Bando de Dados 65](#_Toc210851579)

**LISTA DE QUADROS**

[Tabela 1 - Documentação do Caso de Uso: Fazer Login 43](#_Toc210763179)

[Tabela 2 - Documentação do Caso de Uso: Cadastrar Conta 44](#_Toc210763180)

[Tabela 3 - Documentação do Caso de Uso: Enviar Altura do Caminhão 46](#_Toc210763181)

[Tabela 4 - Documentação do Caso de Uso: Consultar Perfil 47](#_Toc210763182)

[Tabela 5 - Documentação do Caso de Uso: Alterar informações do Perfil 48](#_Toc210763183)

[Tabela 6 - Documentação do Caso de Uso: Conectar com o Banco de Dados 50](#_Toc210763184)

[Tabela 7 - Documentação do Caso de Uso: Exibir Alerta 51](#_Toc210763185)

[Tabela 8 - Documentação do Caso de Uso: Comparar Altura 52](#_Toc210763186)

[Tabela 9 - Documentação do Caso de Uso: Retirar Altura 53](#_Toc210763187)

[Tabela 10 - Documentação do Caso de Uso: Reconhecer Placa 54](#_Toc210763188)

**LISTA DE ABREVIATURAS E SIGLAS**

Cascading Style Sheet (CSS)

Create, Read, Update e Delete (CRUD).

Deep Learning (DL)

HyperText Markup Language (HTML)

Internet of Things (IoT)

Node Package Manager (NPM)

Open Source Computer Vision Library (OpenCV)

Optical Character Recognition (OCR)

Package Installer for Python (PIP)

Rede Neural Artificial (RNA)

Regra de Negócio (RN)

Requisito Funcional (RF)

Requisito Não Funcional (RNF)

Unified Modeling Language (UML)

Universal Serial Bus (USB)

You Only Look Once (YOLO)

Sumário

[1 INTRODUÇÃO 8](#_Toc210851782)

[2 REFERENCIAL TEÓRICO 10](#_Toc210851783)

[2.1 Acidentes envolvendo caminhões e viadutos 10](#_Toc210851784)

[2.2 Tecnologias Utilizadas 10](#_Toc210851785)

[2.2.1 Internet of Things (IoT) 11](#_Toc210851786)

[2.2.2 Raspberry PI 12](#_Toc210851787)

[2.2.3 Visão Computacional 14](#_Toc210851788)

[2.2.4 OpenCV (Open Source Computer Vision Library) 15](#_Toc210851789)

[2.2.5 YOLO (You Only Look Once) 17](#_Toc210851790)

[2.2.6 Aprendizado de Máquina 19](#_Toc210851791)

[2.2.7 Python 22](#_Toc210851792)

[2.2.8 Firebase 24](#_Toc210851793)

[2.2.9 EasyOCR 27](#_Toc210851794)

[2.2.10 Package Installer for Python (PIP) 28](#_Toc210851795)

[2.2.11 React Native 29](#_Toc210851796)

[2.2.12 CSS (Cascading Style Sheet) 30](#_Toc210851797)

[2.2.13 JavaScript 31](#_Toc210851798)

[2.2.14 Node.JS 32](#_Toc210851799)

[2.2.15 UML (Unified Modeling Language) 33](#_Toc210851800)

[2.2.16 Wireframe 37](#_Toc210851801)

[3 DESENVOLVIMENTO 38](#_Toc210851802)

[3.1 Diagrama de Caso de Uso 38](#_Toc210851803)

[3.1.1 Documentação dos Casos de Uso 40](#_Toc210851804)

[3.2 Diagrama de Atividade 56](#_Toc210851805)

[3.3 Diagrama de Sequência 66](#_Toc210851806)

[4 CONCLUSÃO 67](#_Toc210851807)

[REFERÊNCIAS 68](#_Toc210851808)

# INTRODUÇÃO

O Eyetruck é um dispositivo para prevenção de colisões de veículos de carga com viadutos e pontes, por meio do reconhecimento das placas R-15, que indicam a altura máxima permitida. O sistema é composto por hardware, câmera e uma aplicação móvel, sendo capaz de comparar a altura máxima informada na sinalização com a altura do caminhão, inserida previamente pelo motorista. Caso tenha risco de colisão, alertas serão emitidos ao condutor, contribuindo para segurança viária.

A problemática que motivou a fazer este trabalho está relacionada à recorrência de acidentes e bloqueios de vias provocados por veículos de carga que ultrapassam a altura máxima permitida em trechos específicos. Embora exista uma lei que obriga a existência de sinalização indicando a altura máxima em viadutos e pontes, esses métodos não têm sido eficientes o bastante para evitar os acontecimentos. Pode se observar que as estruturas citadas apresentam variações nas alturas, o que leva a situações em que o veículo consegue passar por uma via, mas não por outra.

Incialmente, acreditava-se que a principal causa do problema fosse a falta de sinalização para alertar o motorista, no entanto foi descoberto que é obrigatório as sinalizações especificando a altura máxima permitida de veículos, mas certas variáveis como a falta de padronização no tamanho das estruturas e a desatenção ou a falta de familiaridade dos motoristas nas vias contribuem para que os acidentes continuem ocorrendo.

O desenvolvimento do EyeTruck justifica-se pela necessidade de um outro método de prevenção de acidentes, redução de custos relacionados a danos na infraestrutura viária e segurança tanto para o motorista do caminhão quanto para os motoristas alheios, assim, contribuindo para uma viagem mais segura.

Este projeto está delimitado à cidade de São Paulo, capital e região metropolitana, devido ao tráfego intenso de veículos de grande porte e elevado de número de infraestruturas como viadutos e pontes, visando atender cidades com acesso à internet e grande concentração de estruturas que contenham restrições de altura.

Por fim, utilizamos o livro Internet das Coisas de Eduardo Magrani (2018) para falar sobre IoT e o livro Introdução à Visão Computacional: Uma abordagem prática com Python e OpenCV de Felipe Barelli (2018) para utilizar a câmera como os olhos do nosso dispositivo.

# REFERENCIAL TEÓRICO

Neste capítulo serão apresentados os principais conceitos, tecnologias e uma visão geral da problemática, tendo em vista melhor entendimento do contexto em que este trabalho foi desenvolvido.

## Acidentes envolvendo caminhões e viadutos

Segundo notícia publicada pela Folha de S. Paulo (2019), ao menos 15 caminhões por mês ficam bloqueados por ultrapassarem a altura permitida em pontes e viadutos. Entre janeiro de 2017 e outubro de 2018, foram registrados 213 casos desse tipo de ocorrência, que geraram riscos a outros condutores de veículos e danos às infraestruturas de transportes.

Com relação a isso, de acordo com a R7 (2019), a cada dois dias acontecia um acidente envolvendo caminhões com excesso de altura que não conseguiram passar por viadutos e pontes na capital paulista, sendo que esses dados foram tirados de um levantamento feito pela CET (Companhia de Engenharia de Tráfego) revelando que de janeiro a setembro foram 157 ocorrências. Apenas o mês de setembro foram 22 casos, enfatizando que além dos danos à estrutura, geravam problemas no tráfego com congestionamento e enormes filas de carros.

## Tecnologias Utilizadas

A seguir, serão explicadas as tecnologias utilizadas no desenvolvimento do projeto Eye Truck, abordando linguagens de programação, componentes eletrônicos e conceitos utilizados.

### Internet of Things (IoT)

De acordo com Santos (2019), a Internet das Coisas é uma conexão de dispositivos que se comunicam entre si com a finalidade de realizar tarefas de forma autônoma, sem a necessidade de interação humana.

Carrion e Quaresma (2019) destacam que o cientista da computação e inventor John Romkey criou uma torradeira que podia ser ligada e desligada pela internet, sendo considerada o primeiro de muitos dispositivos IoT.

Diante disso, percebe-se que a Internet das Coisas é muito mais do que apenas conectar um dispositivo à internet. Seu objetivo é tornar as “coisas” inteligentes, sendo capazes de aproveitar a conexão e utilizar os dados coletados do ambiente ou das redes às quais estão conectadas (OLIVEIRA, 2021).

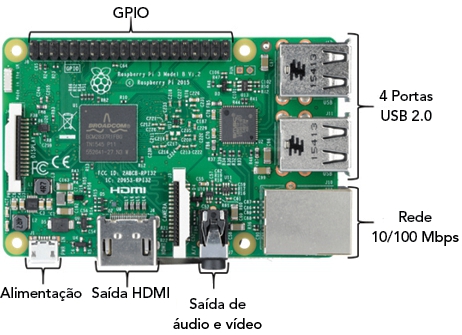
### Raspberry PI

Em 2006, Eben Upton, Rob Mullins, Jack Lang e Alan Mycroft criaram um computador portátil e acessível para crianças na Universidade de Cambridge, na Inglaterra. A iniciativa surgiu após a percepção de que os estudantes que se inscreviam para participar do laboratório de Ciências da Computação apresentavam pouco conhecimento prévio sobre linguagens de programação e arquitetura dos computadores quando comparado à dos alunos de 1990. Assim, o projeto teve como objetivo ajudar jovens a adquirirem conhecimentos de *software* e *hardware*. (EBERMAM et al. 2017).

Em 2012, a empresa Raspberry PI, fundada por Eben e seus colegas, criou um microcomputador de baixo custo com o mesmo nome. Esse pequeno dispositivo é um ótimo meio para a introdução aos conceitos de Eletrônica, podendo ser utilizado tanto por novatos na área da tecnologia quanto por especialistas nas áreas de Computação. (OLIVEIRA; NABARRO; ZANETTI, 2018).

Conforme Silva, Dias e Escudero (2022), o Raspberry PI possui diferentes modelos disponíveis. A principal diferença entre eles está na sua capacidade computacional. Alguns exemplos para melhor entendimento seriam a potência e o tipo do processador, quantidade de entradas USB (*Universal Serial Bus*), tecnologias implementadas, entre outros.

Figura - Raspberry PI 3, Modelo B



Fonte: Oliveira, Nabarro, Zanetti, 2018.

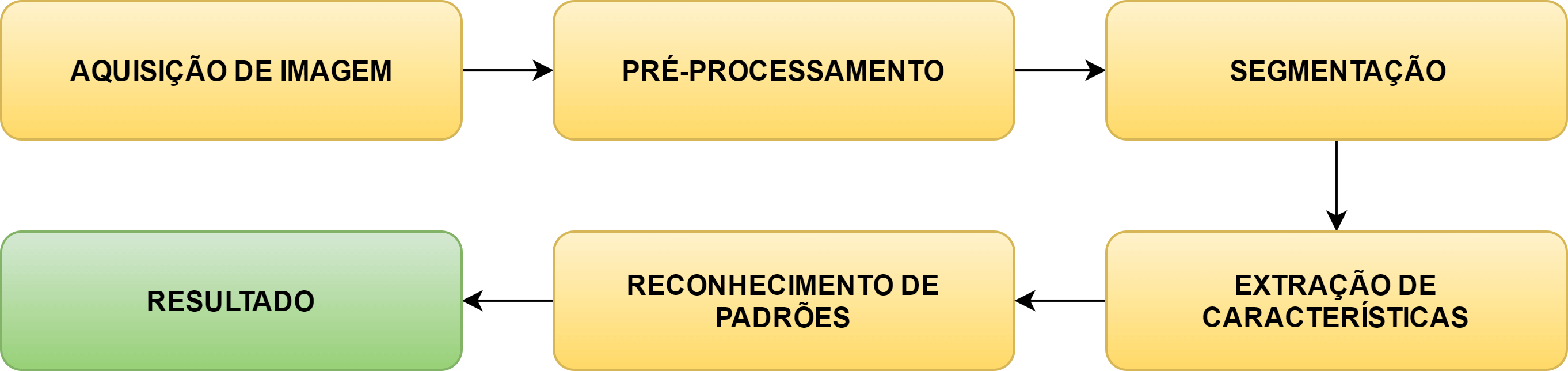
### Visão Computacional

Segundo Barelli (2018) Visão Computacional é o estudo de máquinas que possam reconhecer elementos de um ambiente, através de imagens capturadas, sensores e outras tecnologias, essas informações podem ser extraídas e manipuladas pelo desenvolvedor conforme seus objetivos.

A visão computacional é um tema muito recente no mundo tecnológico, com sua primeira aparição em 1955, com Selfridge. Na década de 70 começaram as pesquisas de visão computacional com a inteligência artificial, os cientistas acreditavam que em breve teriam o sentido completo da visão das máquinas, atrasando-se em decorrência da complexidade que o tema possuía na época para a ciência, surgindo pela falta de informações biológicas de como nosso cérebro consegue interpretar um ambiente ou objeto (MILANO; HONORATO, 2014).

De acordo com Barelli (2018) os sistemas de visão computacional, por mais que pareçam complexos, seguem um fluxo em comum, como mostrado no exemplo a seguir.

Figura - Fluxo de um sistema baseado em visão computacional



Fonte: Barelli, 2018, p. 18.

Visão Computacional não é somente limitado ao controle de qualidade, esta tecnologia pode ser usada em diversas áreas do mundo, desde agricultura até sistemas de reconhecimento autônomo, um desses exemplos é o "Alvinn" um robô produzido por Dean Pormeleau, um carro que podia adquirir conhecimento através da experiencia, ou seja, ele reconhecia as ruas e "memorizava" elas. (MILANO; HONORATO, 2014).

### OpenCV (Open Source Computer Vision Library)

OpenCV é uma biblioteca multiplataforma feita pela Intel que divulgou sua criação por volta dos anos 2000, desenvolvida especialmente para o carregamento de imagens e estruturação de dados em aplicações, considerado um grande aliado da visão computacional o OpenCV, por ser de código aberto (Open Source), ou seja, é disponível para uso do público. (BARELLI, 2018).

Consoante a Marengoni e Stringhini (2010), o OpenCV é uma biblioteca gigante com diversas funções disponíveis de acesso gratuito, é dividido em cinco principais grupos, que são: Manipulação e tratamento de imagens, análise da estrutura de cenas, monitoramento de movimento e rastreamento de objetos, identificação de padrões e ajuste de câmera com reconstrução em 3D. A ideia do projeto foi criar um projeto que facilitasse o acesso dos desenvolvedores a tecnologias de visão computacional.

Agora iremos exibir um exemplo que consiga identificar, capturar e salvar uma imagem de um rosto.

Figura - Trecho do código que realiza a captura da face



Fonte: Sette, Carvalho, Basile, 2021, p. 5.

No exemplo do código, dois métodos principais da biblioteca OpenCV são usados para executar o reconhecimento facial. O primeiro, cv2.VideoCapture(), ativa a câmera do dispositivo, permitindo a captura de imagens em tempo real. O segundo, cv2.CascadeClassifier, aplica modelos de detecção baseados em classificadores em cascata já treinados, para identificar os rostos nas imagens capturadas. Aí também estão algumas outras funções, como cv2.cvtColor, para converter as imagens em tons de cinza, e cv2.imwrite, que salva os arquivos gerados, que também são frequentes ao longo do código (SETTE; CARVALHO; BASILE, 2021).

Figura - Imagem da face alinhada com marcação nos olhos



Fonte: Sette, Carvalho, Basile, 2021, p. 6.

Nesta imagem o OpenCV está destacando o ponto de interesse, ou seja, ele descarta o que não é relevante para o desenvolvedor e para o sistema para facilitar a extração de dados e informações úteis com mais facilidade.

Figura - Imagem da região de interesse



Fonte: Sette, Carvalho, Basile, 2021, p. 6.

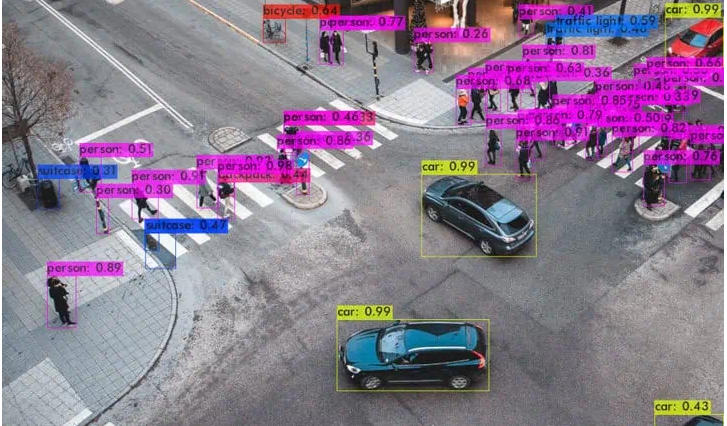
Por fim ele separou a região de interesse do resto da imagem utilizando as bibliotecas do OpenCV para o desenvolvedor poder manipulá-la da forma que quiser, como por exemplo, reconhecimento por íris ou photoshop. (SETTE; CARVALHO; BASILE, 2021).

### YOLO (You Only Look Once)

YOLO (You Only Look Once) é conhecido por suas práticas com redes neurais, é um modelo de detecção de objetos criado por Joseph Readmon e Ali Farhadi na universidade de Washington em 2015. Seu diferencial dentre as outras tecnologias de detecção de objetos é sua velocidade para a detecção, porém pelo YOLO olhar somente uma vez para a imagem acontece uma perda de acurácia de acordo com Araújo (2022).

Conforme Silva (2024), YOLO cria grids, que são divisões na imagem, para que dentro de cada grid exista uma caixa delimitadora (bounding box) para armazenar a probabilidade de existir um objeto naquela região.

Figura - Exemplo da tecnologia YOLO em ambiente urbano



Fonte: Ultralytics, 2023.

Cada Bounding box tem um ponto de confiança que vai indicar se existe ou não a chance de existir um objeto na região, caso a probabilidade seja muito baixa a bounding box vai desaparecer consoante ao Leocádio et al. (2021).

Concomitante ao Pacheco (2018), o YOLO vem de uma Rede Neural Artificial (RNA), o RNA surgiu por meio de um modelo de matemática do neurônio biológico realizado por MCulloch e Pitts na data de 1943. No RNA é feito diversas camadas que agem de forma analógica a um neurônio realizando um processamento de uma parte da informação total, essas camadas são: camada de entrada, camadas intermediárias ou ocultas, e camada de saída.

A camada de entrada irá inicialmente receber os primeiros dados da rede, logo depois na camada intermediária irá acontecer o maior número de processamento e por último, dentro da camada de saída, acontecerá a conclusão que é exibida de acordo com Kovács (2023).

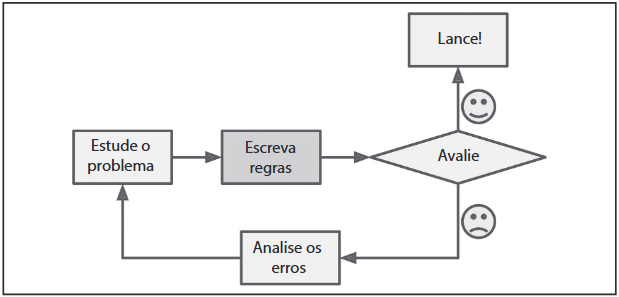
Segundo Pereira (2017), Deep Learning (DL) também chamado de Aprendizado Profundo envolve RNA, o DL foi criado como exemplo para analisar e assimilar arquiteturas que têm quantidades maiores de dados, com isso ele é capaz de criar exemplos parciais que são assimilados durante o treinamento, assim ocupando a necessidade do pré-processamento e extração de recursos manualmente designados.

### Aprendizado de Máquina

Segundo Almeida (2019), quando pensamos no tópico “aprendizado de máquina” pensamos em um robô que irá dominar o mundo ou irá fazer todas as nossas tarefas do dia a dia, porém isso já existe desde 1990 quando foi criado o “filtro de spam”, se pararmos para notar os pequenos detalhes como “recomendados” entendemos que o aprendizado de máquina já existe em diversas tecnologias. Para resumir em poucas palavras, o aprendizado de máquina é uma arte ou ciência que permite uma máquina ser treinada e possa aprender com dados.

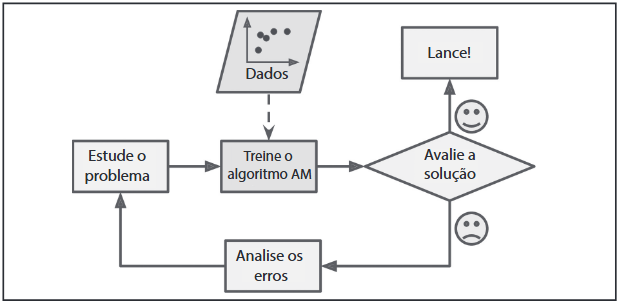
Uma das principais vantagens de se utilizar o Aprendizado de Máquina é sua tecnologia que permite se adaptar, por exemplo, vamos pensar que você esteja criando um programa de “filtro de spam”, caso você fosse criar em uma abordagem comum (sem o Aprendizado de Máquina), você teria que criar diversas regras de forma que quando chegasse um e-mail “Oferta imperdível” ele fosse para a caixa de spam pelas palavras, até o momento seu programa funcionaria, porém se o e-mail passasse a chegar como “Esse desconto só para você”, seria inviável conforme Santos (2019).

Figura - Abordagem tradicional



Fonte: Géron, 2020, p. 4.

Figura - Abordagem do aprendizado de máquina



Fonte: Géron, 2020, p. 5.

Consoante ao Géron (2020), notamos a diferença entre os tipos de abordagem, na (Figura 2) o caso do spam seria diferente, caso o aprendizado de máquina percebesse que “Esse desconto só para você” se tornou um email frequente na caixa de spam dos outros usuários, automaticamente ele começa a redirecionar todos os próximos e-mails para a caixa de spam. Aqui está um exemplo de como funcionaria essa atualização de dados para que fosse funcional:

Figura - Adaptando-se automaticamente à mudança



Fonte: Géron, 2020, p. 5.

### Python

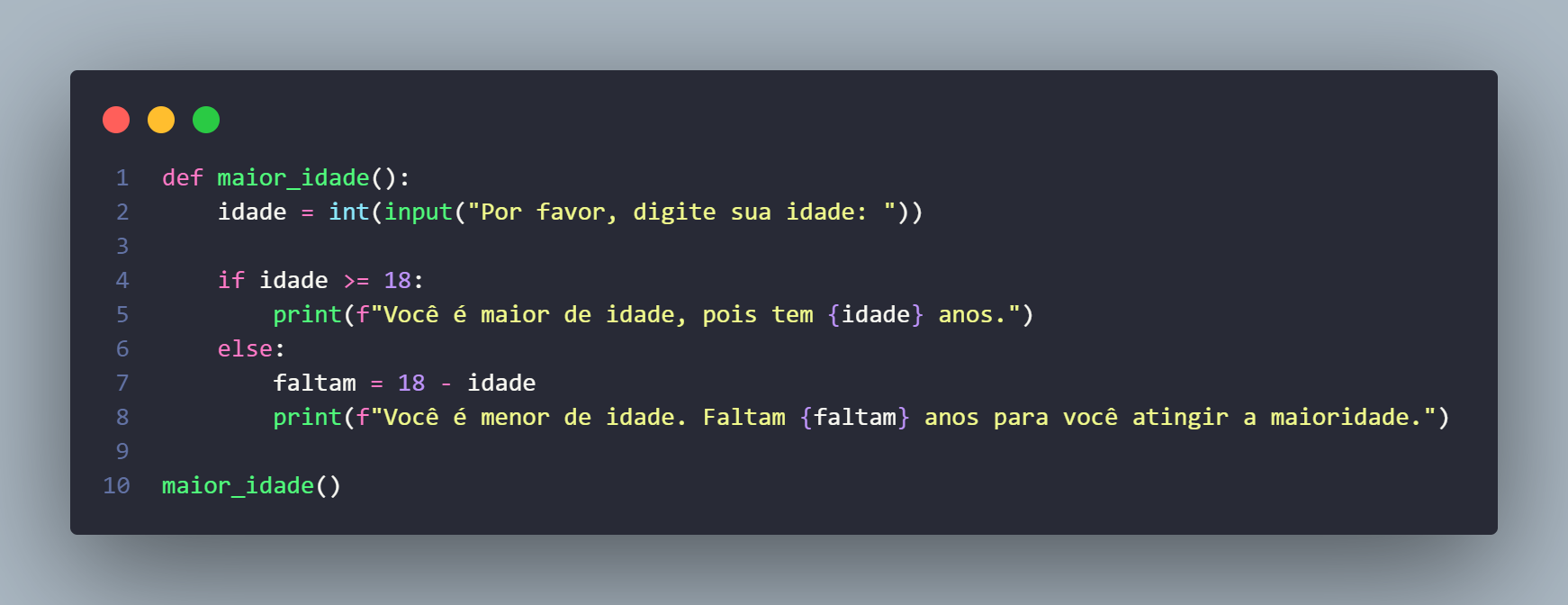
O Python é uma linguagem de fácil acesso e consolidada no mercado, isso graças as empresas envolvidas em sua criação, podendo ser usada em diversos sistemas operacionais como Linux, Windows, Mac OS; e por sua alta flexibilidade podendo ser utilizada em outras áreas da programação como por exemplo: Inteligência Artificial, aplicações mobile e até mesmo animações 3D como destaca Nilo Menezes (2010).

Segundo Matthes (2016), Python tem uma grande performance diante de sua simplicidade, com uma sintaxe objetiva e legível, grande versatilidade em uso geral, fácil manutenção e alta concentração de pacotes para o auxílio de desenvolvedores, sendo capaz de realizar desde pequenos códigos até grandes aplicações.

Nos últimos 20 anos, ele passou de uma linguagem de computação científica inovadora [...] para uma das mais importantes linguagens para ciência de dados, machine learning e desenvolvimento de softwares em geral no ambiente acadêmico e no setor empresarial. (McKinney, 2023, p.18)

Python consolidou seu espaço na indústria global, sendo aplicada em grandes empresas, como Google, Yahoo, Microsoft, Nokia e Disney para aplicações web, desenvolvimento de softwares para celulares e animações 3D, como destaca Borges (2014).

Figura - Exemplo do código em Python



Fonte: Autoria Própria, 2025.

O código acima é um exemplo de uma função básica em Python que realiza uma verificação para saber se a idade digitada pelo usuário indica se ele é maior de idade ou não. A seguir, uma breve explicação do código:

Linha 1: Cria uma função denominada maior\_idade.

Linha 2: Utiliza a função input() para o usuário digitar sua idade e retornar como string. Através da função int(), ele transforma esse valor em um número inteiro, que será armazenado dentro da variável idade.

Linha 4: Usa o comando de decisão if para verificar se o valor armazenado na variável idade é maior ou igual a 18 por meio do operador de comparação “maior ou igual a” (>=) .

Linha 5: Aplica a função print() para aparecer a mensagem no console. O F fora das aspas aponta que é uma f-string, que possibilita adicionar variáveis no texto, como a variável idade adicionada na mensagem.

Linha 6: Caso a condição idade >= 18 for falsa, emprega-se outro comando de decisão, o else, que executará outro código.

Linha 7: Calcula quantos anos faltam para o usuário completar 18 anos por meio de uma subtração, 18 – idade, que é armazenada na variável faltam.

Linha 8: Exibe uma mensagem no console informando que o usuário ainda é menor de idade e mostrando quantos anos faltam para atingir a maior idade, através da variável faltam.

Linha 10: Chama a função maior\_idade para que o código seja executado.

Figura - Resultado do código em Python



Fonte: Autoria Própria, 2025.

Figura - Outro resultado do código em Python



Fonte: Autoria Própria.

Acima, há uma demonstração de dois exemplos do retorno do código em Python, nos quais foram necessários fornecer a idade do usuário. Foi possível fazer uma verificação por meio de um comando de decisão e do uso de um operador de comparação para indicar se ele era maior de idade ou não.

### Firebase

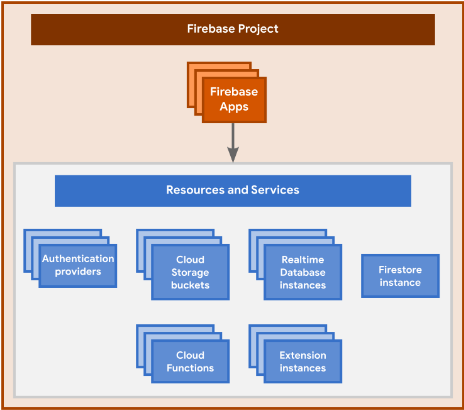
Gonçalves (2014) explica que um banco de dados pode ser entendido como um conjunto de informações relacionadas, que podem ser armazenadas tanto em um computador quanto, atualmente, em serviços de nuvem. Esses dados são organizados em linhas e colunas, estruturados em tabelas criadas dentro do banco, existe neste meio de dados uma forma de descrever as 4 principais formas de manipular as informações chamada CRUD (Create, Read, Update e Delete), o Create como o próprio nome diz, cria/insere uma nova informação no banco, seguido do Read que é listar esses dados inseridos, em seguido o Update é poder atualizar/alterar essas informações a qualquer momento e por fim o Delete é a função que exclui os dados inseridos no banco.

De acordo com Carvalho (2015), o banco de dados influencia em todas as etapas de um projeto: desde a fase inicial de criação até o desempenho do sistema e sua manutenção ao longo do tempo. Por esse motivo, contar com uma base de dados bem estruturada e confiável é essencial para o sucesso de qualquer aplicação.

Machado (2021) reforça essa ideia ao afirmar que todo software moderno depende de uma boa organização e de uma manutenção adequada dos dados. Nesse contexto, o Google disponibiliza o Firebase, uma plataforma robusta que oferece diversas ferramentas, escalabilidade, respostas rápidas às consultas, flexibilidade de armazenamento e alta disponibilidade.

Dentro de um projeto no Firebase, é possível registrar um ou mais aplicativos, todos com os mesmos acessos e recursos compartilhados, funcionando de forma integrada. A plataforma disponibiliza diferentes planos de uso, como o Spark e o Blaze, que são planos pagos da plataforma e variam conforme as limitações e necessidades de cada projeto (GOOGLE, 2025).

Figura - Funcionamento do Firebase



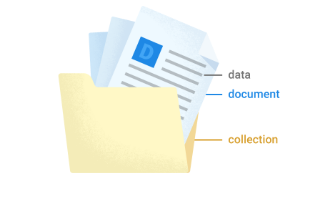
Fonte: Google, 2025.

#### Cloud Firestore

O Firestore é um banco de dados NoSQL que possibilita aos usuários armazenar e sincronizar informações tanto no servidor quanto no cliente. Ele pode ser aplicado em aplicativos da web, móveis e de servidor, garantindo a integração e a sincronização entre diferentes aplicativos. As informações são organizadas em grupos de documentos que se alocam no banco. (MATIOLA, 2023).

Conforme a Google (2025), o Cloud Firestore é hospedado na nuvem, assim podendo ser acessado juntamente com os aplicativos da Apple, Android e da Web com um conjunto de ferramentas disponibilizado pela empresa chamado SDKs, trabalhando com o NoSQL do Cloud Firestore que armazena dados em blocos de documentos, assim mapeando valores para ajudar o usuário a se organizar e desenvolver consultas.

Figura - Explicação de um Dado



Fonte: GOOGLE (2025).

Em suma, o Firestore é um banco de dados flexível e escalável que estrutura dados em coleções e documentos de forma hierárquica. Ele permite consultas rápidas e eficazes, além de sincronização em tempo real entre cliente e servidor, assegurando desempenho e integração constante (FREITAS, 2023).

### EasyOCR

Com o objetivo de facilitar o processo de desenvolvimento de modelos de aprendizado de máquina, o Facebook lançou em 2016 o PyTorch, uma biblioteca de código aberto baseada na linguagem de programação Python, conforme Florencio (2020).

De acordo com Souza (2024), a sua eficácia em construir modelos dinâmicos, possibilita um processo de desenvolvimento eficiente, flexível e altamente otimizado para o uso com GPUs, permitindo que o usuário tenha uma melhor performace utilizando a potência maxima do hardware para treinar seus modelos de forma rapida.

Através dessa tecnologia, foi possível desenvolver a biblioteca EasyOCR, responsável pelo reconhecimento óptico de caracteres, empregada na extração de textos presentes em imagens ou documentos, atualmente conseguindo reconhecer mais de 80 idiomas, segundo Cordeiro (2024).

### Package Installer for Python (PIP)

Em conformidade com o escritor Matthes (2016), o Package Installer for Python também conhecido como (PIP), é uma ferramenta crucial para o gerenciamento de pacotes que possui as principais funcionalidades de instalar, atualizar e desinstalar bibliotecas no Python.

A instrução abaixo indica como realizar a instalação da biblioteca EasyOCR e suas dependências para a linguagem de programação Python:

Figura - Instalação da biblioteca EasyOCR



Fonte: Autoria Própria, 2025.

### React Native

O React Native foi desenvolvido em 2013 durante um Hackathon interno do Facebook, evento conhecido como uma maratona de programação que envolve equipes de diversos setores para solucionar um problema em um determinado período de tempo. Após o evento, a empresa identificou o potencial da ideia e começou a desenvolvê-la. A primeira aparição pública ocorreu em 2015, na React.js Conference. Meses depois, em uma conferência realizada pelo Facebook, conhecida como F8, foi anunciado que o React Native seria um projeto de código aberto, disponível por meio do repositório no GitHub (Escudelario e Pinho, 2020).

De acordo com Galvão (2018), o React Native é uma tecnologia flexível, podendo ser utilizada em dispositivos Android e iOS. Ele funciona em uma arquitetura com três camadas, a principal executada no dispositivo da aplicação, a segunda, executada em uma máquina virtual com o JavaScript e a terceira, servindo de ponte entre as outras duas, as quais nunca se comunicam diretamente.

Para a criação, implementação e testes em dispositivos móveis utilizaremos o Expo, um conjunto de serviços e ferramentas para auxiliar no desenvolvimento de aplicações em React, com foco principal voltado para React Native, sendo possível empregar JavaScript em seu ambiente, tirando a necessidade de apenas utilizar os recursos da plataforma nativa, conforme Falcão (2022).

De acordo com Coelho e Antunes (2023), o framework Expo simplifica a configuração do ambiente e oferece diversas funcionalidades, com recursos que colaboram para a eficiência do processo de desenvolvimento, possibilitando testar e depurar a aplicação em um dispositivo móvel em tempo real por meio do recurso Expo Go.

### CSS (Cascading Style Sheet)

Conforme Eis (2015), a linguagem de estilo CSS, foi criada em 1994, para o auxílio de formatar documentos em HTML (HyperText Markup Language), desenvolvidas por Bert Boss e Hákon Wium Lie, sua principal diferença é a aptidão da herança de estilos, sendo capaz de reutilizar o mesmo estilo em diversos elementos do código.

Também conhecida como Cascading Style Sheets, sua utilização serve para especificar a formatação e aparência do documento, ganhando destaque por sua manutenção de fácil atualização, com a separação do conteúdo de HTML para apresentação CSS, conforme relato por Albino (2023).

Como destaca Silva (2007), com a finalidade de tirar qualquer formatação do documento HTML para apresentação web, o CSS veio para ajustar as fontes, cores, posicionamento, tamanhos e todo o visual da página, são inúmeras vantagens, entre elas, a redução de carga da página web, agilidade na manutenção e controle completo da apresentação do site através de um arquivo central.

### JavaScript

Nas palavras de Iepsen (2018), o Javascript foi inventado em meados de 1995 pela Netscape para ser uma linguagem que fosse universal nos navegadores da época, pois mais ou menos neste tempo havia um conflito de linguagens que funcionavam em certos navegadores e em outros não, o Javascript chegou para apaziguar a situação

Figura - Exemplo de código em JavaScript



Fonte: Autoria Própria, 2025.

A história do JavaScript é atual comparado as outras linguagens, de acordo com Silva e Sobral (2017), a linguagem nasceu na década de 90 com a intenção de ser uma nova forma de codificar Front-end (aparência) de páginas web. Com sua rápida ascensão no mercado competitivo, popularidade e sua alta adaptabilidade os desenvolvedores da época começaram a utilizar a linguagem na área de back-end (Funcionalidades), pois era mais prático o programador utilizar somente 1 linguagem para as 2 funções do que várias para chegarem no mesmo objetivo.

Conforme afirma o autor Flanagan (2012), o JavaScript é a linguagem mais utilizada nos sites e dispositivos modernos de hoje em dia, tornando-a uma das linguagens mais usadas da história, esse tipo de código pertence a uma base de linguagens que todo programador deve conhecer, que são: HTML, CSS e JS.

### Node.JS

Na década de 2000 os computadores estavam com cada vez mais falhas de sistema e frequentes travamentos, era cada vez mais necessário um upgrade nas peças dos servidores. Como era algo com um alto custo, Ryan Dahl teve uma ideia que amenizasse este problema sem gastos exacerbados, com ajuda inicial de 14 colaboradores eles criaram o Node.js (PEREIRA, 2014),

Segundo Ihrig (2020), o Node.js foi desenvolvido principalmente como uma ferramenta para programar aplicações web que possam suportar muitos usuários ao mesmo tempo. Sua alta velocidade de execução faz com que o Node.js seja particularmente apropriado para sistemas que exigem respostas rápidas e processamento em tempo real.

Silva (2025) evidencia que o node.js é um grande fator para que o JavaScript esteja entre as linguagens mais aplicadas e confiáveis pelos programadores do mundo hoje em dia, possibilitando o uso flexível do front-end e o back-end de maneira eficiente.

Juntamente com o Node.js, acompanha o gerenciador de pacotes conhecido como Node Package Manager (NPM), que permite instalar, atualizar e remover módulos de forma simples, além de armazenar um grande acervo de pacotes e bibliotecas de código aberto em JavaScript, facilitando o gerenciamento de dependências e otimizando o desenvolvimento de projetos ao integrar funcionalidades fornecidas pela comunidade, segundo Santos (2023).

### UML (Unified Modeling Language)

Segundo Guedes (2018), a UML é uma linguagem visual de modelagem que auxilia na compreensão do sistema, definindo os requisitos fundamentais para seu funcionamento e manutenção, sendo adotada como norma-padrão internacionalmente.

Conforme Booch (2006), a UML, assim como todas as outras linguagens, exerce a função de estabelecer um vocabulário e regras para a comunicação entre os envolvidos no desenvolvimento do projeto, possibilitando a representação conceitual e física dos elementos que compõem o sistema.

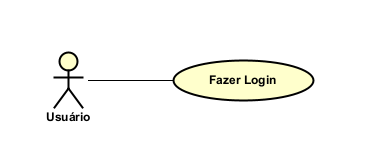
Fowler (2005) destaca que, mesmo a UML não possuindo o mesmo rigor dos métodos matemáticos, sua aplicabilidade a torna valiosa no cotidiano dos programadores. Ela é relevante mesmo sem ser totalmente formal ou detalhada.

Uma casa não se constrói com apenas uma planta, ela é dividida em várias plantas que mostram diversas partes da casa para uma melhor visualização. Devido a isso, existem 13 diagramas no total que podem ser utilizados para descrever o funcionamento tanto estrutural quanto lógico de um sistema (GUEDES, 2018).

Com isso em mente, utilizaremos o Diagrama de Casos de Uso, Diagrama de Atividade, Diagrama de Sequência e, por fim, Diagrama de Máquina de estados.

O diagrama de casos de uso nos mostra uma visão geral dos requisitos do sistema, sem se aprofundar nas especificações das funcionalidades. Através dele podemos entender as funcionalidades do sistema e quais funcionalidades o usuário tem acesso (GUEDES, 2018).

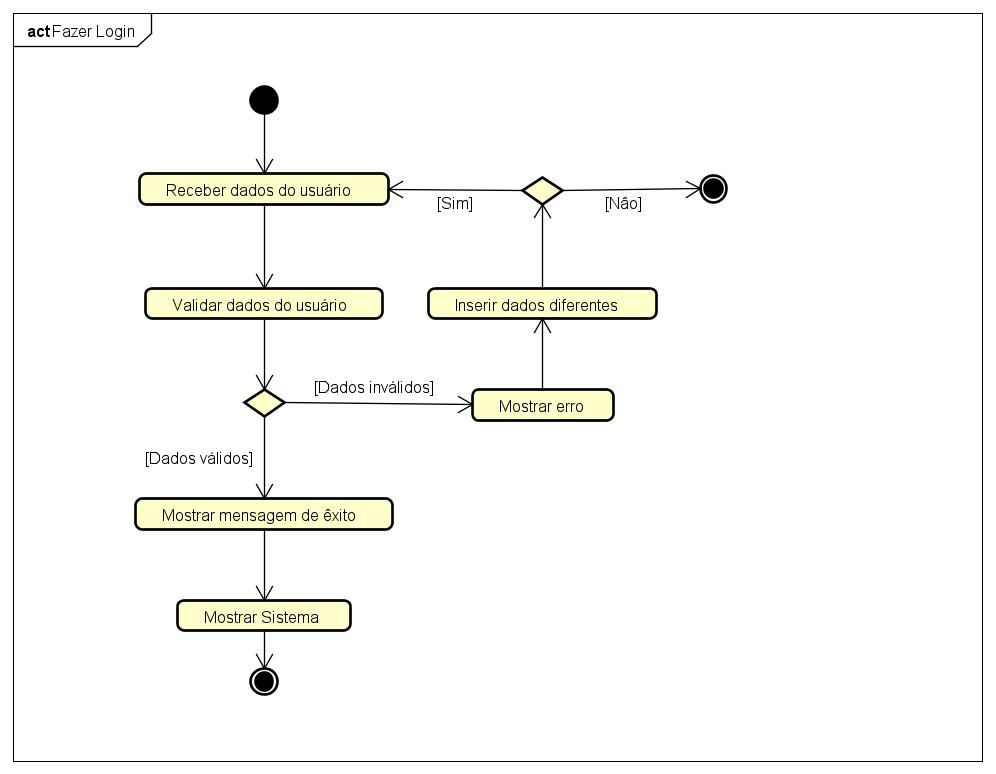
Figura – Exemplo de Caso de Uso



Fonte: Autoria Própria, 2025.

O diagrama de atividade demonstra o fluxo de ações e decisões, representando a execução das atividades de forma clara e visual, desde o início até a conclusão do processo. (BOOCH, 2006).

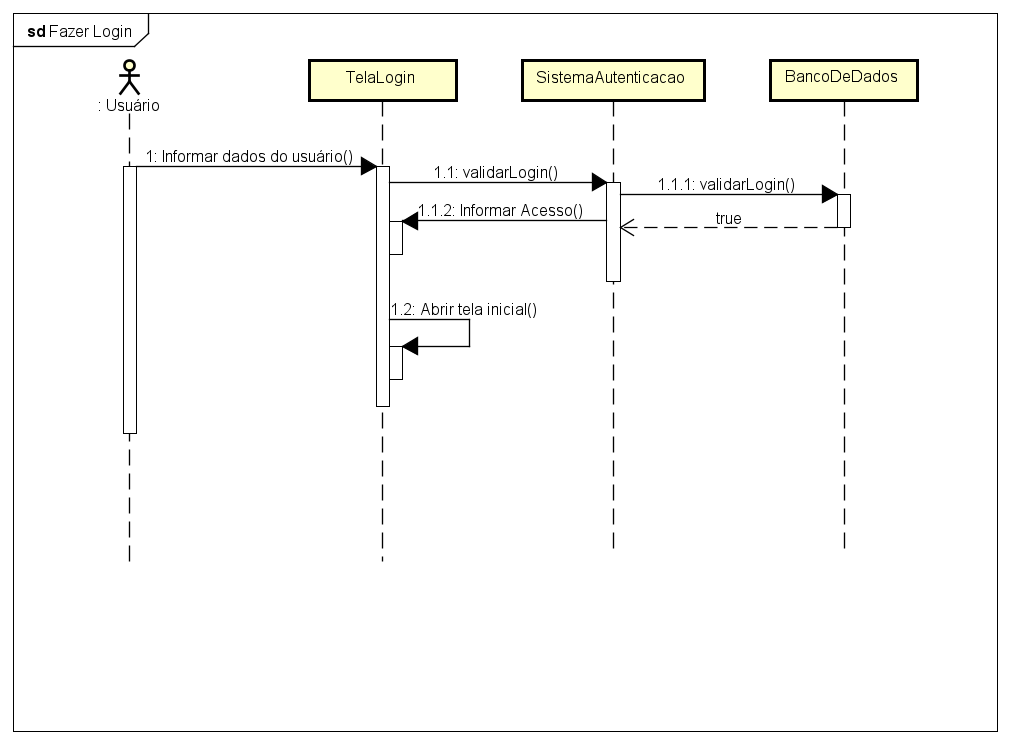
Figura - Diagrama de atividade Fazer Login



Fonte: Autoria Própria, 2025.

O diagrama de sequência é uma representação das interações de um objeto durante a execução de um processo, deixando visível as mensagens trocadas entre os objetos, a ordem em que ocorrem e quais métodos são chamados. Para ser construído, ele utiliza casos de uso e o diagrama de classes (GUEDES, 2018).

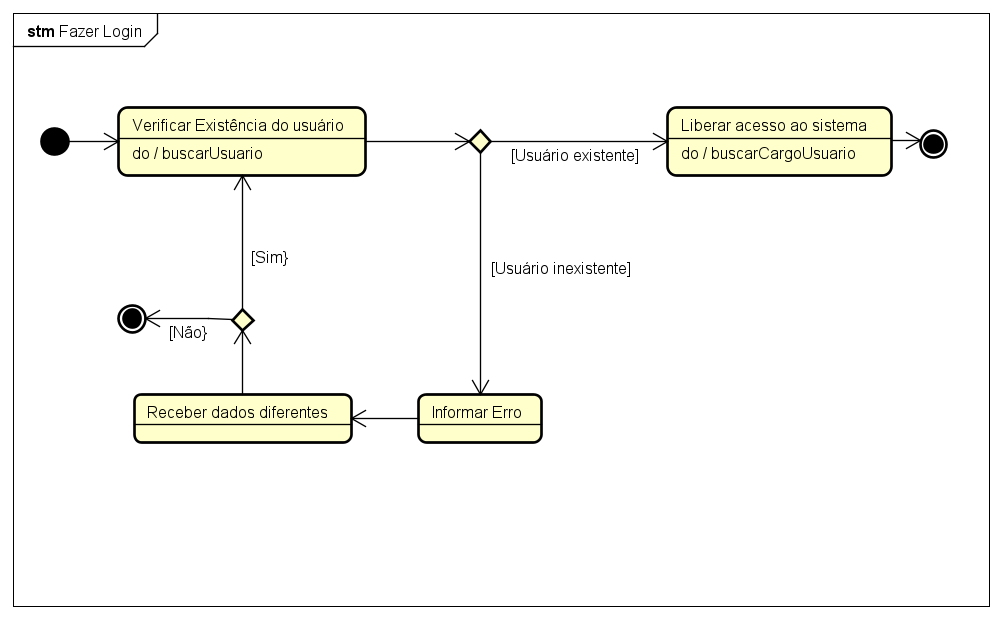
Figura – Diagrama de sequência Fazer Login



Fonte: Autoria Própria, 2025.

O diagrama de máquina de estado é utilizado para representar o comportamento de um objeto ou sistema em resposta a eventos, mostrando os estados pelos quais um objeto pode passar ao longo de sua vida (FOWLER, 2005).

Figura - Diagrama de máquina de estado Fazer Login



Fonte: Autoria Própria, 2025.

### Wireframe

Em conformidade com Osório (2021), os wireframes são criados para elaborar protótipos visuais da estrutura e finalidade da aplicação, auxiliando na usabilidade ao mostrar os elementos essenciais que deverão estar presentes no layout do software antes de ser desenvolvido, fornecendo uma perspectiva visual de como ficará.

Em continuidade sobre prototipação, o Figma destaca-se como uma das principais ferramentas de design do mercado, sobressaindo-se pela sua alta performance e pela aptidão para uso simultâneo de usuários, oferecendo uma ampla variedade de recursos tecnológicos que facilitam o desenvolvimento de esboços visuais, segundo Rosa (2023).

O objetivo dessa ferramenta é facilitar, para os desenvolvedores, a criação de uma interface digital, oferecendo serviços como criação de componentes, protótipos interativos, realização de testes e colaboração do time em tempo real de forma remota e simultânea, conforme Silva (2024).

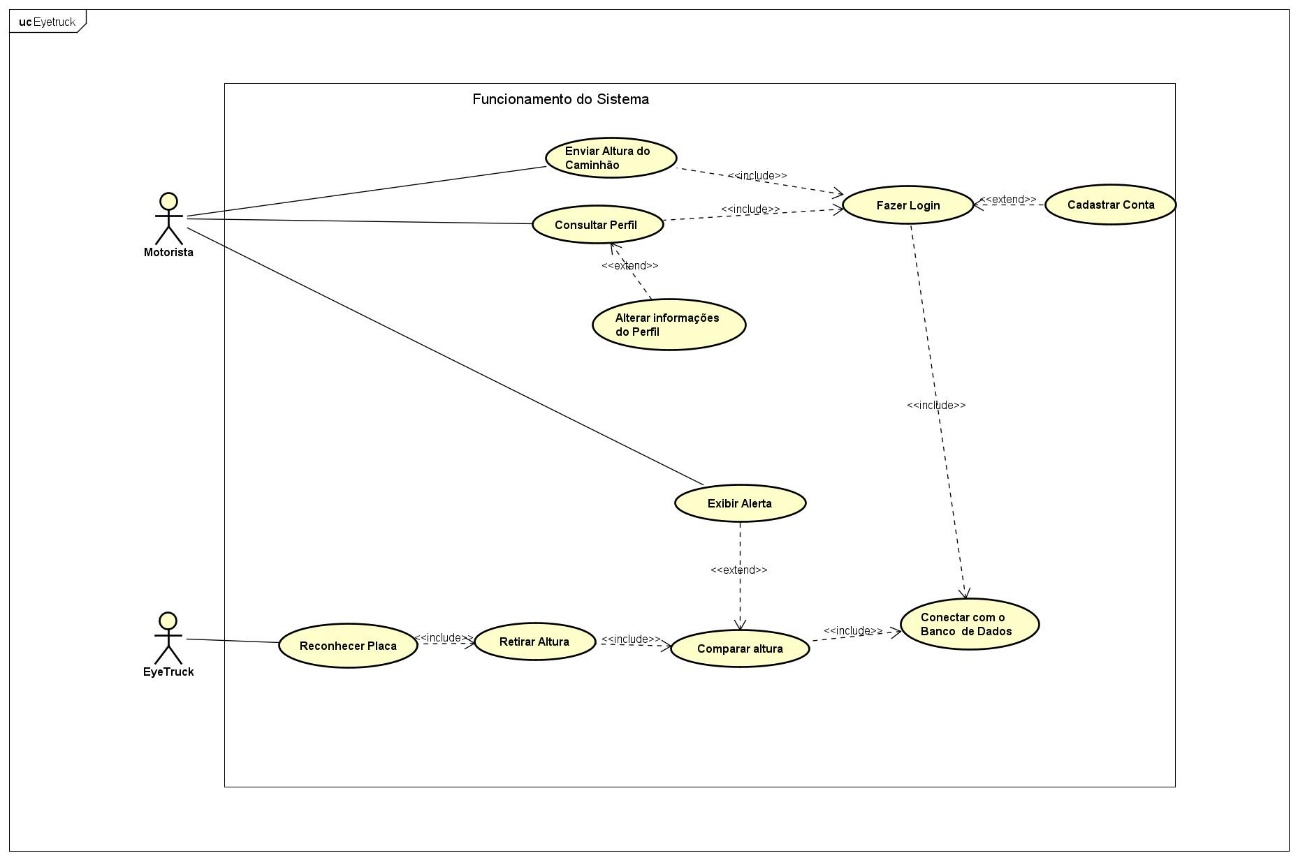
# DESENVOLVIMENTO

Neste capítulo, serão apresentadas as etapas utilizadas para o desenvolvimento do EyeTruck. O processo inicia-se pela documentação, elaborada com base nos métodos da UML, seguida pelo processo de configuração do dispositivo Raspberry PI 5, pelo desenvolvimento das telas e pelo método utilizado para a compilação do aplicativo no dispositivo móvel. No decorrer do capítulo, serão descritos os diagramas UML de caso de uso, atividade, sequência e máquina de estado, indispensáveis para a construção do projeto. Posteriormente, é exibido o processo de criação da aplicação mobile e as suas respectivas interfaces. Essa base sólida possibilitou o desenvolvimento tanto do aplicativo quanto do dispositivo embarcado, que se complementam para entregar a solução proposta.

## Diagrama de Caso de Uso

Nesta seção, será realizada uma detalhada análise do diagrama de caso de uso do EyeTruck, considerado o primeiro e principal diagrama da UML, servindo de base para todos os diagramas posteriores. O diagrama de caso de uso apresenta as interações entre os atores externos, como os motoristas, e as funcionalidades do sistema. Além disso, ele indica quais funcionalidades o usuário tem acesso, oferecendo uma visão geral do funcionamento do projeto. Essas funcionalidades estão representadas por 10 elipses, conforme ilustrado na figura abaixo.

Figura - Diagrama de Casos de Uso



Fonte: Autoria Própria, 2025.

### Documentação dos Casos de Uso

A documentação aprofunda-se nas etapas percorridas pelo usuário e pelo sistema durante a execução de um determinado caso de uso, incluindo os possíveis cenários alternativos. Complementarmente, são apresentadas as regras de negócio, os requisitos funcionais e não funcionais do EyeTruck, com a finalidade de proporcionar melhor compreensão do sistema para um indivíduo externo.

Na sequência, são elucidados os termos citados anteriormente. As regras de negócio estabelecem condições e restrições que independem da tecnologia utilizada. Os requisitos funcionais representam as funções que o sistema deve executar, enquanto os requisitos não funcionais abordam aspectos de qualidade, desempenho, segurança e usabilidade, assegurando que a solução seja eficiente e confiável.

Regras de Negócio:

* RN01: O acesso ao aplicativo requer autenticação por login.
* RN02: O suporte físico para o display é de responsabilidade do usuário.
* RN03: Apenas usuários com idade igual ou superior a 21 anos poderão utilizar o sistema.
* RN04: As senhas cadastradas devem conter no mínimo seis caracteres.
* RN05: O sistema deve garantir boa usabilidade, principalmente em dispositivos móveis.
* RN06: A empresa não se responsabiliza por acidentes decorrentes de instalação incorreta do dispositivo.
* RN07: A manutenção do hardware não é de responsabilidade da empresa.
* RN08: O sistema deve oferecer instalação simples e utilização intuitiva.

Requisitos Funcionais do Aplicativo:

* RF01: O sistema deve validar se o e-mail já está cadastrado.
* RF02: O sistema deve validar e conectar-se ao banco de dados.
* RF03: O sistema deve validar se o número do dispositivo já está cadastrado no banco de dados.
* RF04: O sistema deve exibir as informações do usuário cadastradas no banco.

Requisitos Funcionais do Motorista:

* RF01: O motorista deve poder cadastrar-se com dados pessoais e do veículo (e-mail, senha e altura do caminhão).
* RF02: O motorista deve acessar o sistema por meio de login (e-mail e senha).
* RF03: O motorista deve poder alterar sua senha.
* RF04: O motorista deve poder encerrar a sessão manualmente.
* RF05: O motorista deve poder editar suas informações pessoais.
* RF06: O motorista deve visualizar suas informações pessoais.
* RF07: O motorista deve atualizar a altura do caminhão quando necessário.

Requisitos Funcionais do EyeTruck:

* RF01: O dispositivo deve processar os dados recebidos da câmera.
* RF02: O dispositivo deve interpretar caracteres de placas de sinalização R-15.
* RF03: O dispositivo deve enviar os dados processados para o sistema, possibilitando a comparação com as medidas do veículo.
* RF04: O dispositivo deve comparar os valores processados com a altura do caminhão registrada.
* RF05: O dispositivo deve exibir um alerta ao motorista caso os valores da altura do veículo sejam maiores que os indicados na placa.

Requisitos Não Funcionais do EyeTruck:

* RNF01: O aplicativo será desenvolvido em React Native e terá design responsivo para tablets e celulares.
* RNF02: O sistema deve ser de uso simples e intuitivo.
* RNF03: A atualização dos dados enviados pelo dispositivo deve ocorrer com a menor latência possível.
* RNF04: O banco de dados será implementado utilizando os serviços do Firebase.
* RNF05: A modelagem do sistema deve seguir os padrões UML.

Abaixo estão as tabelas descritivas dos casos de uso presentes no diagrama:

Tabela - Documentação do Caso de Uso: Fazer Login

|  |  |
| --- | --- |
| **Nome do Caso de Uso** | **Fazer Login** |
| Caso de uso Geral |  |
| Ator Principal | Motorista |
| Ator Secundário |  |
| Resumo | Esse caso de uso descreve as etapas que o motorista deve seguir para acessar as funcionalidades principais do sistema. |
| Pré-Condições | Ter cadastro ativo. |
| Pós-Condições | Acesso autorizado à tela inicial. |
| **Cenário Principal** | |
| **Ações do Ator** | **Ações do Sistema** |
| 1. O motorista informa e-mail e senha |  |
|  | 2.O sistema valida as credenciais |
|  | 3.O sistema libera o acesso e redireciona para a tela inicial |
| Restrições/Validações |  |
| **Cenário Alternativo – Veficação de erros** | |
| **Ações do Ator** | **Ações do Sistema** |
| 1. O motorista insere e-mail ou senha incorretos | 2. O sistema exibe mensagem de erro |

Fonte: Autoria Própria, 2025.

Tabela - Documentação do Caso de Uso: Cadastrar Conta

|  |  |
| --- | --- |
| **Nome do Caso de Uso** | **Cadastrar Conta** |
| Caso de uso Geral |  |
| Ator Principal | Motorista |
| Ator Secundário |  |
| Resumo | Define como o motorista cria sua conta no sistema. |
| Pré-Condições | Possuir o código do dispositivo. |
| Pós-Condições | Usuário cadastrado com sucesso. |
| **Cenário Principal** | |
| **Ações do Ator** | **Ações do Sistema** |
| 1. O motorista informa dados pessoais (nome completo, e-mail, altura do caminhão e senha) |  |
|  | 1. O sistema valida a senha e registra as informações |
|  | 1. O sistema confirma o cadastro e direciona para a tela Login |
| Restrições/Validações |  |
| **Cenário Alternativo - Email Inválido** | |
| **Ações do Ator** | **Ações do Sistema** |
| 1. O motorista insere um e-mail existente |  |
|  | 1. O sistema informa que o e-mail já está registrado |

Fonte: Autoria Própria, 2025.

Tabela - Documentação do Caso de Uso: Enviar Altura do Caminhão

|  |  |
| --- | --- |
| **Nome do Caso de Uso** | **Enviar Altura do Caminhão** |
| Caso de uso Geral |  |
| Ator Principal | Motorista. |
| Ator Secundário | Dispositivos. |
| Resumo | O motorista informa ou atualiza a altura do caminhão |
| Pré-Condições | Estar logado. |
| Pós-Condições | Altura atualizada no sistema. |
| **Cenário Principal** | |
| **Ações do Ator** | **Ações do Sistema** |
| 1. O motorista acessa o menu de configurações |  |
| 1. O motorista informa a altura do caminhão |  |
|  | 1. O sistema armazena a informação no Banco de Dados |
| Restrições/Validações |  |

Fonte: Autoria Própria, 2025.

Tabela - Documentação do Caso de Uso: Consultar Perfil

|  |  |
| --- | --- |
| **Nome do Caso de Uso** | **Consultar Perfil** |
| Caso de uso Geral |  |
| Ator Principal | Motorista. |
| Ator Secundário |  |
| Resumo | Permite ao motorista alterar informações pessoais. |
| Pré-Condições | O usuário deve estar logado. |
| Pós-Condições |  |
| **Cenário Principal** | |
| **Ações do Ator** | **Ações do Sistema** |
| 1. O motorista acessa a página de perfil |  |
|  | 1. O sistema exibe os dados atuais |
| Restrições/Validações |  |
| **Cenário Alternativo - Alterar informações do Perfil** | |
| **Ações do Ator** | **Ações do Sistema** |
|  | 1. Executar caso de uso Alterar informações do Perfil |

Fonte: Autoria Própria, 2025.

Tabela - Documentação do Caso de Uso: Alterar informações do Perfil

|  |  |
| --- | --- |
| **Nome do Caso de Uso** | **Alterar informações do Perfil** |
| Caso de uso Geral |  |
| Ator Principal | Motorista. |
| Ator Secundário |  |
| Resumo | Esse caso de uso descreve os passos que o usuário deve seguir para editar o seu perfil. |
| Pré-Condições | O usuário deve estar logado. |
| Pós-Condições | Perfil Atualizado |
| **Cenário Principal** | |
| **Ações do Ator** | **Ações do Sistema** |
| 1. Acessar a página de Perfil |  |
|  | 1. Exibir os dados cadastrados |
| 1. Preencher os campos de texto com as novas informações |  |
| 1. Clicar no botão de Editar |  |
|  | 1. O sistema faz a validação dos dados inseridos |
|  | 1. As informações do usuário são atualizadas |
| Restrições/Validações |  |
| **Cenário Alternativo - Alterar Senha** | |
| **Ações do Ator** | **Ações do Sistema** |
| 1. Selecionar a opção modificar senha |  |
|  | 1. Exibir o formulário de troca de senha |
| 1. Informar a senha atual |  |
| 1. Definir uma nova senha |  |
| 1. Repetir a nova senha para confirmação |  |
|  | 1. O sistema valida se a senha atual está correta |
|  | 1. O sistema verifica se a confirmação da nova senha coincide |
|  | 1. Em caso de sucesso, atualizar a senha no cadastro |
| **Cenário Alternativo - Informações Inválidas** | |
| **Ações do Ator** | **Ações do Sistema** |
|  | 1. Exibir mensagem de erro |
| **Cenário Exceção - Excluir Conta** | |
| **Ações do Ator** | **Ações do Sistema** |
| 1. Selecionar opção de excluir conta |  |
|  | 1. Exibir modal de confirmação de exclusão |
| 1. Confirmar a exclusão da conta. |  |

Fonte: Autoria Própria, 2025.

Tabela - Documentação do Caso de Uso: Conectar com o Banco de Dados

|  |  |
| --- | --- |
| **Nome do Caso de Uso** | **Conectar com o Banco de Dados** |
| Caso de uso Geral |  |
| Ator Principal | EyeTruck |
| Ator Secundário | Motorista |
| Resumo | Permite a conexão do aplicativo com o hardware. |
| Pré-Condições | Dispositivo ligado e conectado ao Wifi |
| Pós-Condições |  |
| **Cenário Principal** | |
| **Ações do Ator** | **Ações do Sistema** |
|  | 1. Receber parâmetros de conexão |
|  | 1. Validar parâmetros |
|  | 1. Conectar com o banco |
|  | 1. Permitir operações |
| Restrições/Validações |  |
| **Cenário Exceção - Erro na conexão** | |
| **Ações do Ator** | **Ações do Sistema** |
|  | 1. Exibir mensagem de erro ao conectar com o banco |

Fonte: Autoria Própria, 2025.

Tabela - Documentação do Caso de Uso: Exibir Alerta

|  |  |
| --- | --- |
| **Nome do Caso de Uso** | **Exibir Alerta** |
| Caso de uso Geral |  |
| Ator Principal | Dispositivo Eyetruck |
| Ator Secundário | Motorista |
| Resumo | Notifica o motorista sobre possíveis colisões com base na altura do veículo. |
| Pré-Condições | Executar Caso de Uso Comparar Altura |
| Pós-Condições | Motorista informado do risco. |
| **Cenário Principal** | |
| **Ações do Ator** | **Ações do Sistema** |
|  | 1. O dispositivo detecta risco de colisão |
|  | 1. O sistema apresenta a mensagem de alerta |
| Restrições/Validações |  |

Fonte: Autoria Própria, 2025.

Tabela - Documentação do Caso de Uso: Comparar Altura

|  |  |
| --- | --- |
| **Nome do Caso de Uso** | **Comparar Altura** |
| Caso de uso Geral |  |
| Ator Principal | Dispositivo EyeTruck |
| Ator Secundário |  |
| Resumo | Garante que a altura registrada do caminhão seja comparada com a altura de obstáculos. |
| Pré-Condições | Executar Caso de Uso Retirar Altura |
| Pós-Condições |  |
| **Cenário Principal** | |
| **Ações do Ator** | **Ações do Sistema** |
|  | 1. Receber Altura do Caminhão |
|  | 1. Comparar com a Altura da placa |
| Restrições/Validações |  |

Fonte: Autoria Própria, 2025.

Tabela - Documentação do Caso de Uso: Retirar Altura

|  |  |
| --- | --- |
| **Nome do Caso de Uso** | **Retirar Altura** |
| Caso de uso Geral |  |
| Ator Principal | Dispositivo EyeTruck |
| Ator Secundário | Motorista |
| Resumo | Reconhece os caracteres na placa e retira o valor para comparação. |
| Pré-Condições | Executar Caso de Uso Reconhecer Placa |
| Pós-Condições |  |
| **Cenário Principal** | |
| **Ações do Ator** | **Ações do Sistema** |
|  | 1. Receber imagem da placa |
|  | 1. Reconhecer caracteres na placa |
|  | 1. Extrair os caracteres. |
| Restrições/Validações |  |

Fonte: Autoria Própria, 2025.

Tabela - Documentação do Caso de Uso: Reconhecer Placa

|  |  |
| --- | --- |
| **Nome do Caso de Uso** | **Reconhecer Placa** |
| Caso de uso Geral |  |
| Ator Principal | Dispositivo EyeTruck |
| Ator Secundário | Motorista |
| Resumo | Reconhece o padrão da placa através do YOLO para tirar uma foto e usar posteriormente. |
| Pré-Condições |  |
| Pós-Condições |  |
| **Cenário Principal** | |
| **Ações do Ator** | **Ações do Sistema** |
|  | 1. Receber a imagem de entrada. |
|  | 1. Dividir a imagem em grades |
|  | 1. Detectar regiões com características compatíveis com placas do tipo R-15. |
|  | 1. Calcular a confiança e classificar cada região detectada. |
|  | 1. Selecionar as detecções mais confiáveis e descartar as irrelevantes. |
|  | 1. Retornar a detecção com maior confiança para a classe R-15. |
| Restrições/Validações |  |

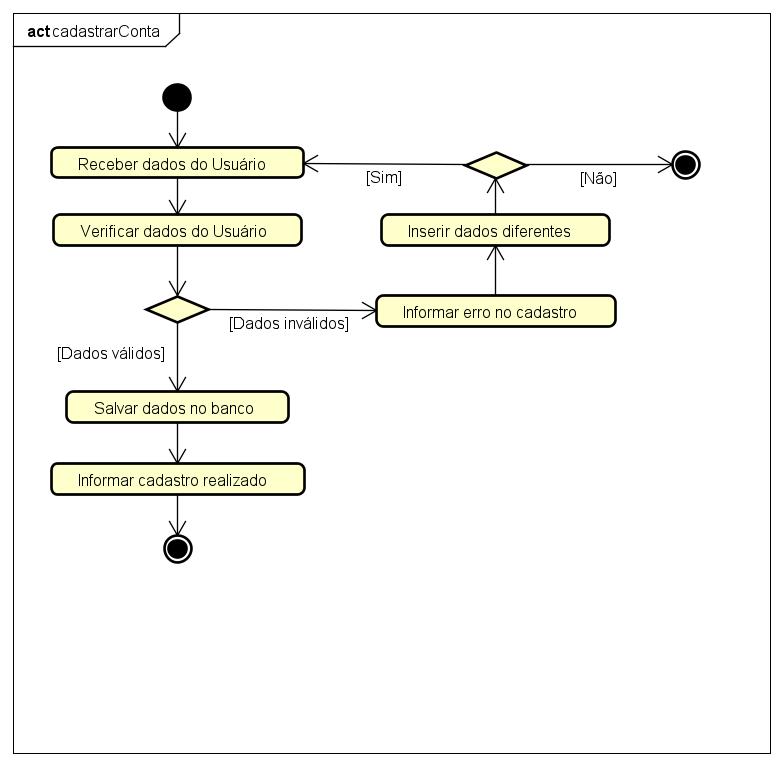
Fonte: Autoria Própria, 2025.

## Diagrama de Atividade

O presente diagrama tem como objetivo representar o fluxo de ações dos casos de uso de forma visual auxiliando no planejamento e na construção das funcionalidades do projeto. Sua principal função é descrever o comportamento do sistema, dividindo em etapas para facilitar a visualização e compreensão.

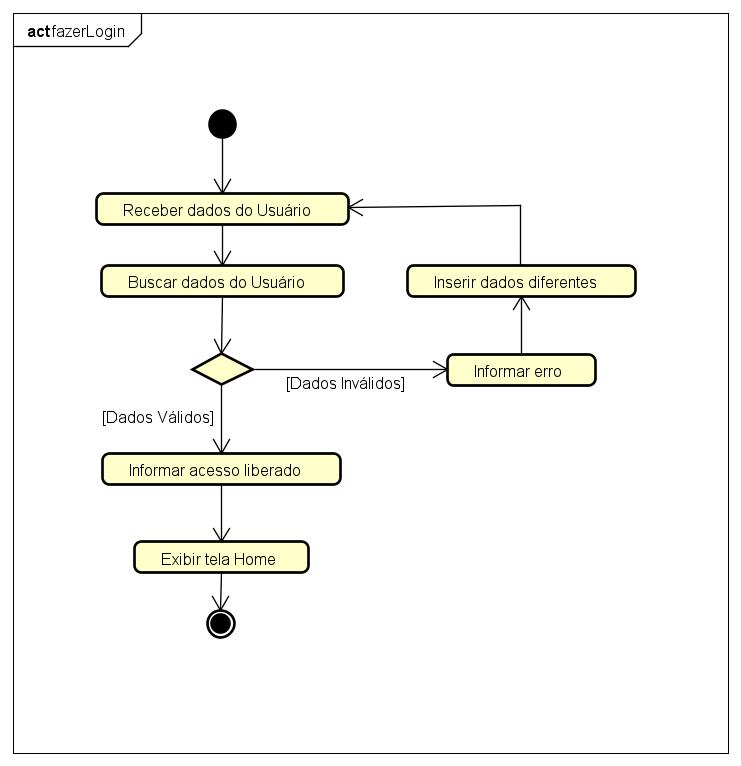
Os diagramas serão organizados com base no ator que interage com o caso de uso. Inicialmente, será apresentado os diagramas relacionados ao ator e, posteriormente, aqueles que representam as interações do sistema.

Figura - Diagrama de Atividade: Cadastrar Conta



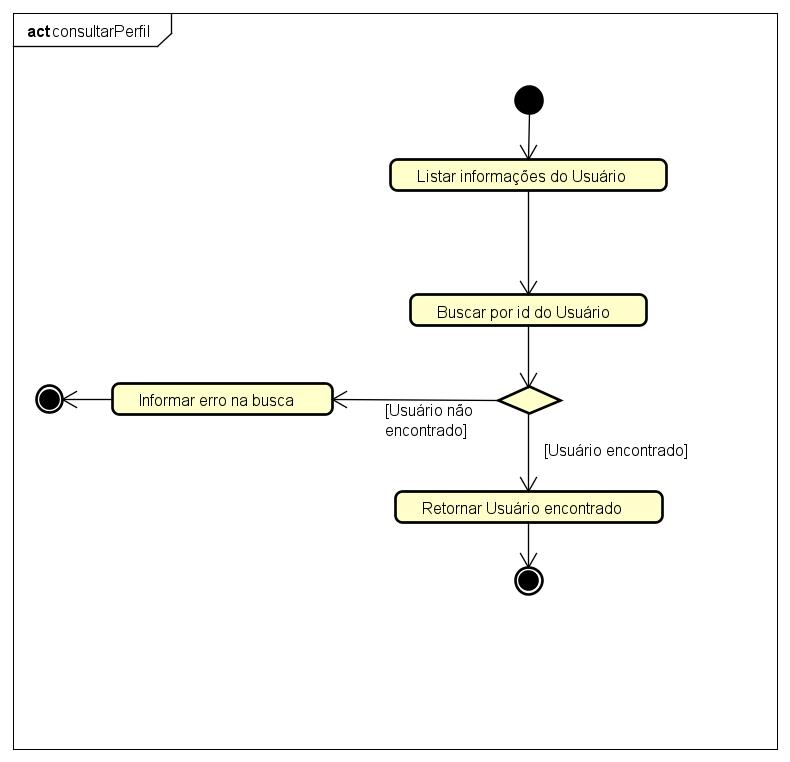
Fonte: Autoria Própria, 2025.

Figura - Diagrama de Atividade: Fazer Login



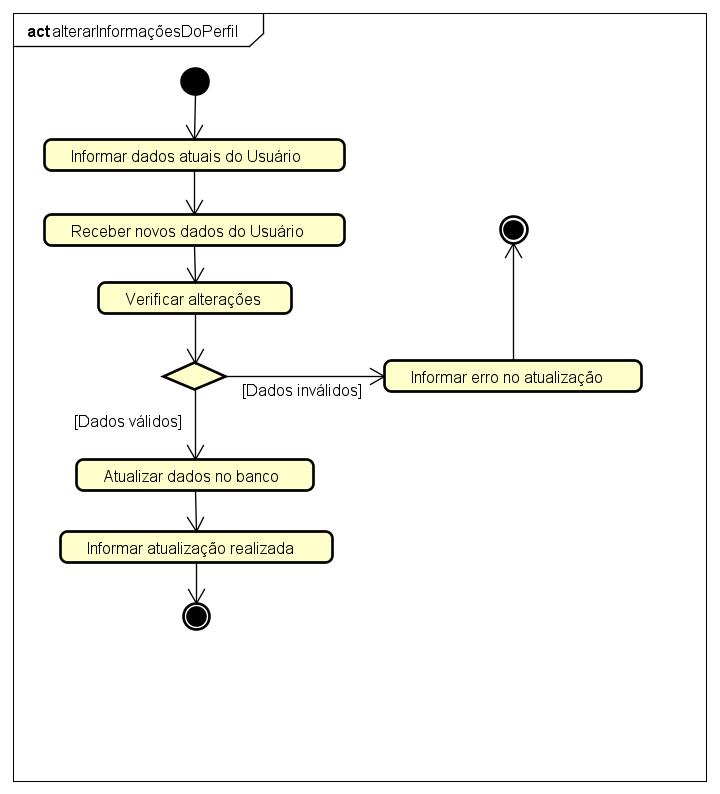
Fonte: Autoria Própria, 2025.

Figura - Diagrama de Atividade: Consultar Perfil



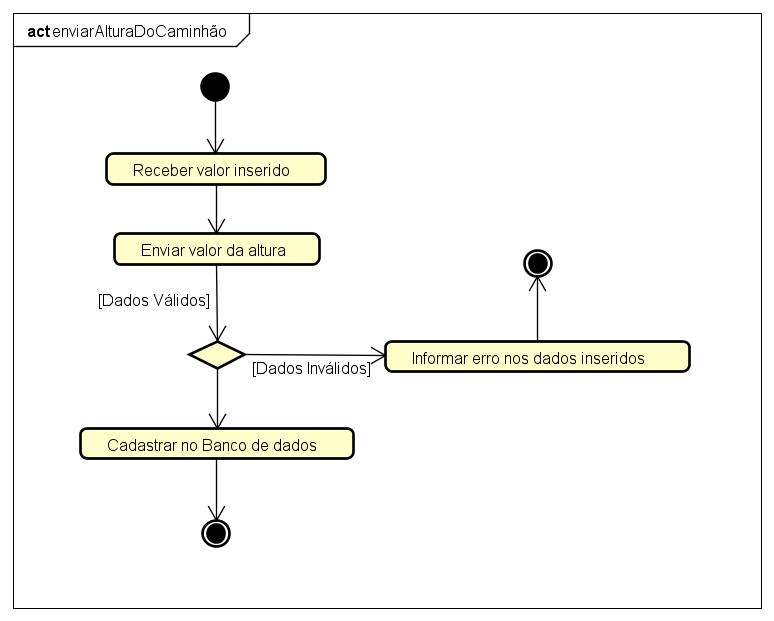
Fonte: Autoria Própria, 2025.

Figura - Diagrama de Atividade: Alterar Informações Do Perfil



Fonte: Autoria Própria, 2025.

Figura - Diagrama de Atividade: Enviar Altura Do Caminhão



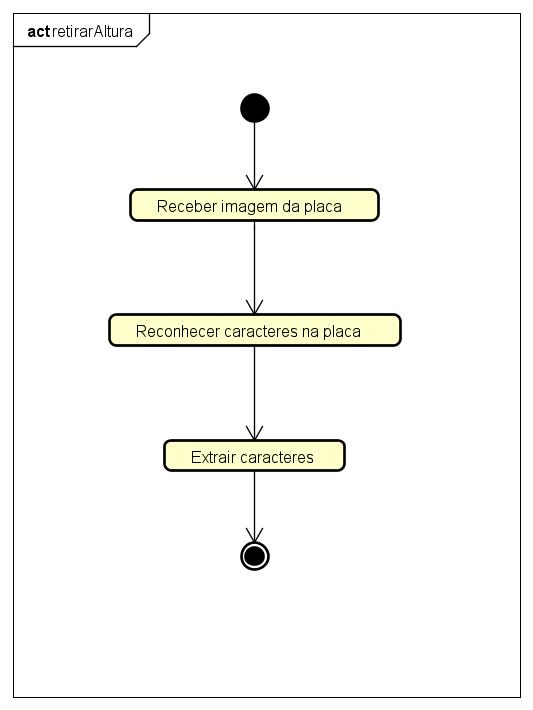
Fonte: Autoria Própria, 2025.

Figura - Diagrama de Atividade: Reconhecer Placa



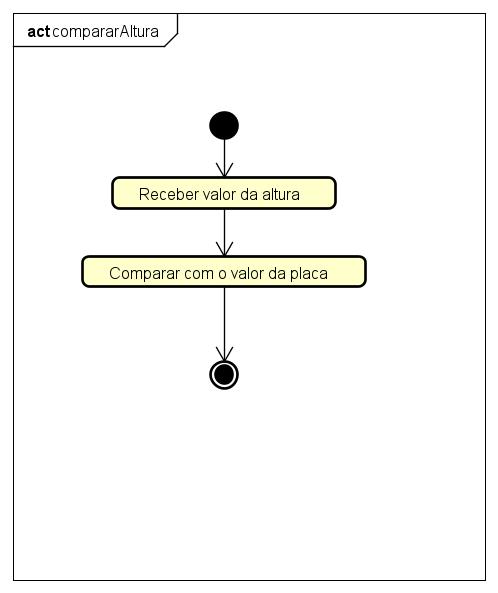
Fonte: Autoria Própria, 2025.

Figura - Diagrama de Atividade: Retirar Altura



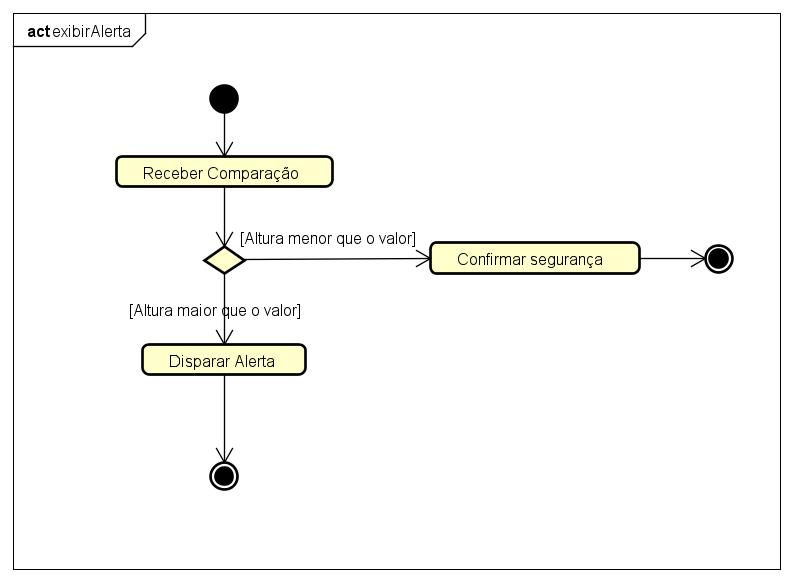
Fonte: Autoria Própria, 2025.

Figura - Diagrama de Atividade: Comparar Altura



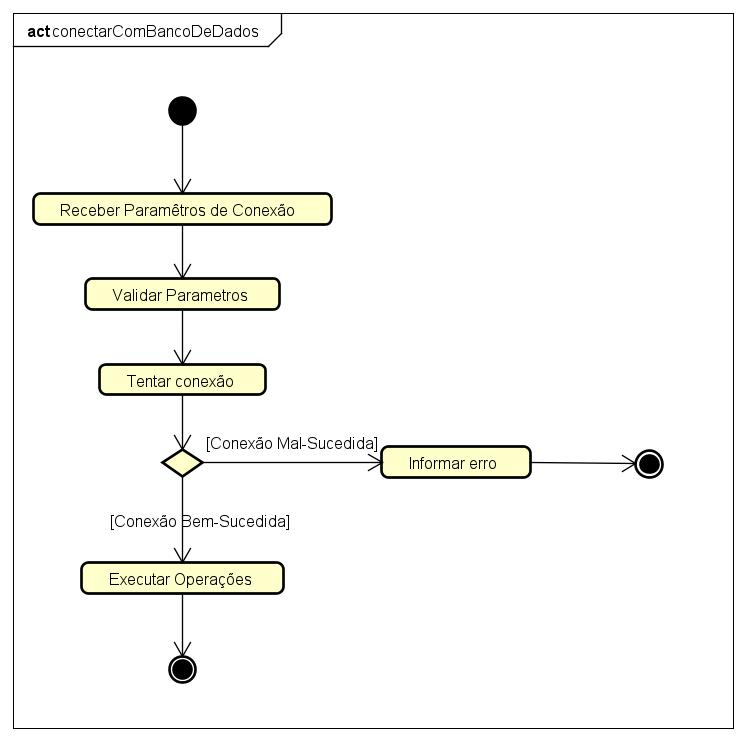
Fonte: Autoria Própria, 2025.

Figura - Diagrama de Atividade: Exibir Alerta



Fonte: Autoria Própria, 2025.

Figura - Diagrama de Atividade: Conectar Com o Bando de Dados

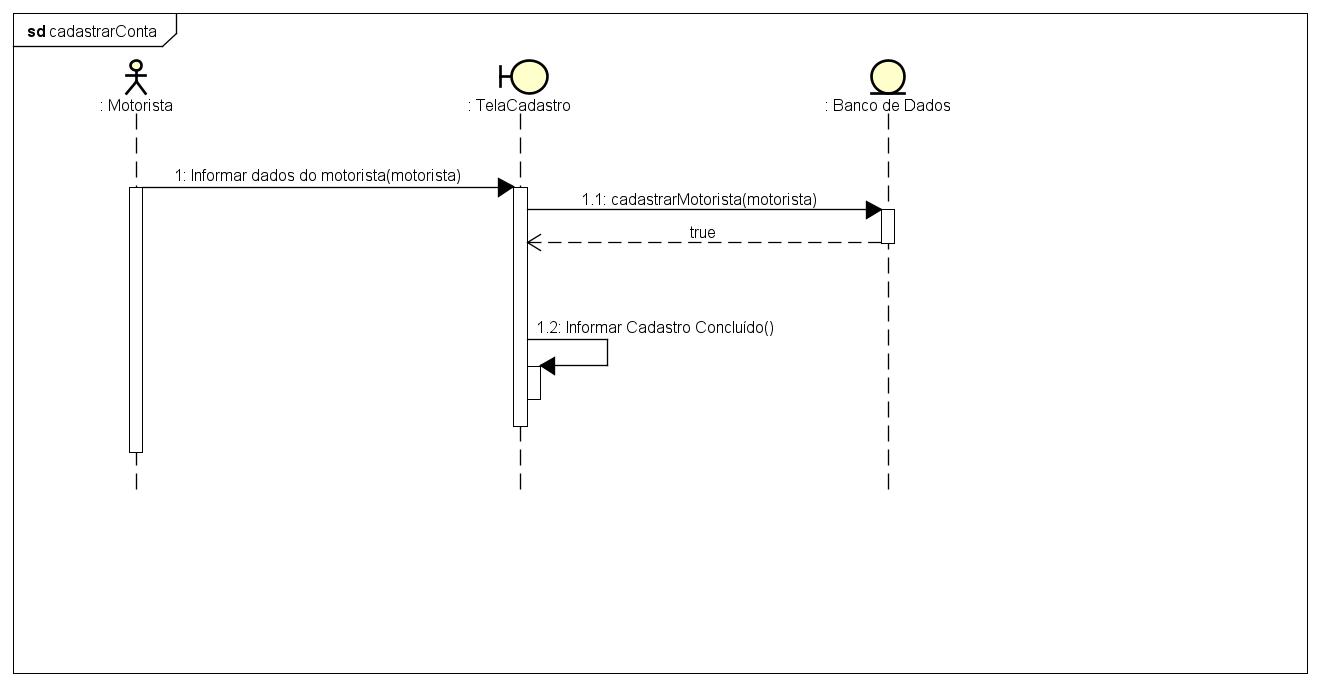


Fonte: Autoria Própria, 2025.

## Diagrama de Sequência

Nesta seção, será apresentado o diagrama de sequência, que enfatiza a ordem temporal das interações, esclarecendo a sequência de ações, os elementos envolvidos, o conteúdo e a ordem em que os processos acontecem. A importância desse diagrama é detalhar a comunicação entre os componentes do sistema e a ordem cronológica dos acontecimentos.

Figura - Diagrama de Sequência: Cadastrar Conta



Fonte: Autoria Própria, 2025.

Figura - Diagrama de Sequência: Fazer Login



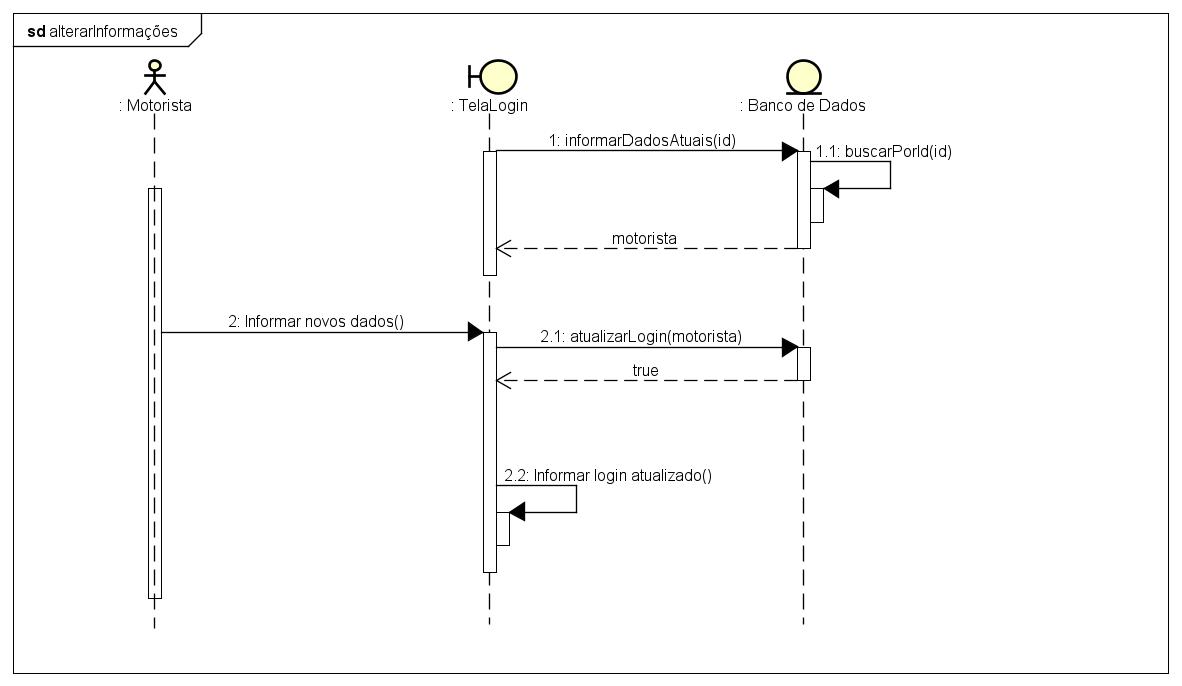
Fonte: Autoria Própria, 2025.

Figura - Diagrama de Sequência: Consultar Perfil



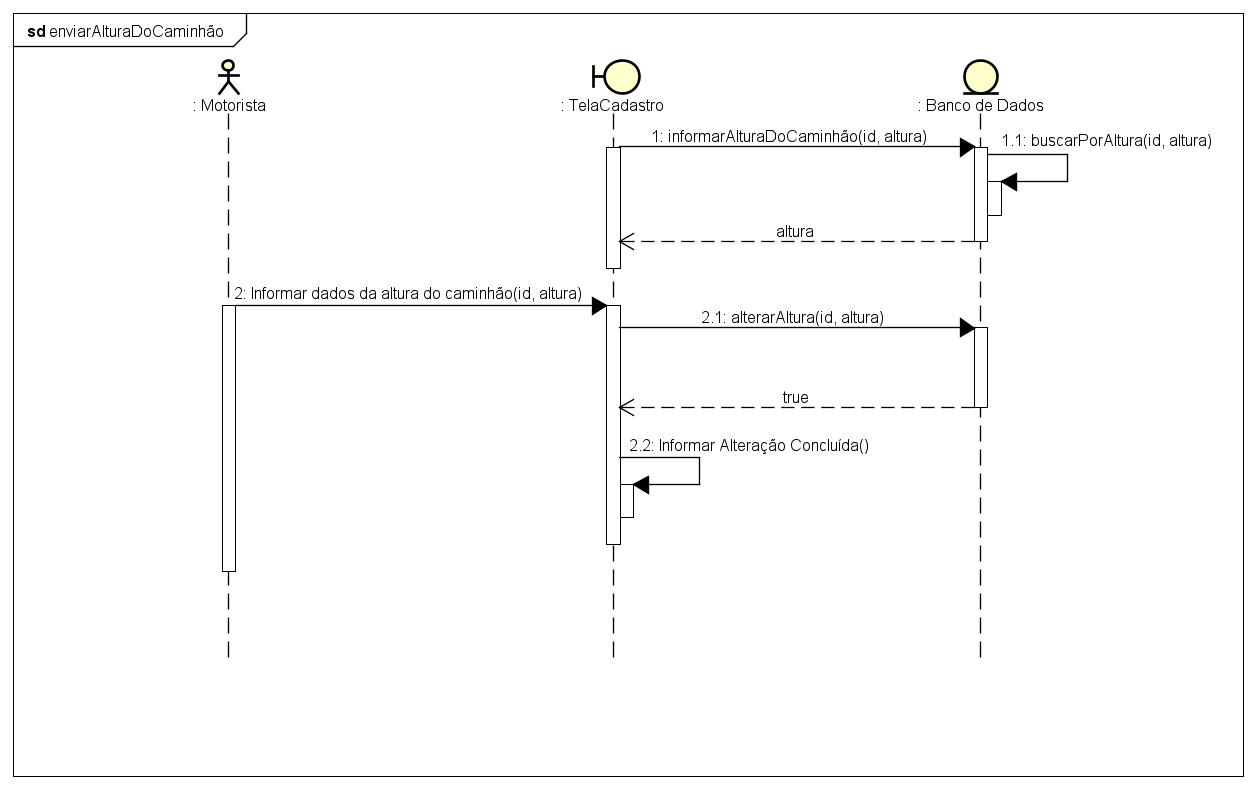
Fonte: Autoria Própria, 2025.

Figura - Diagrama de Sequência: Alterar Informações do Perfil



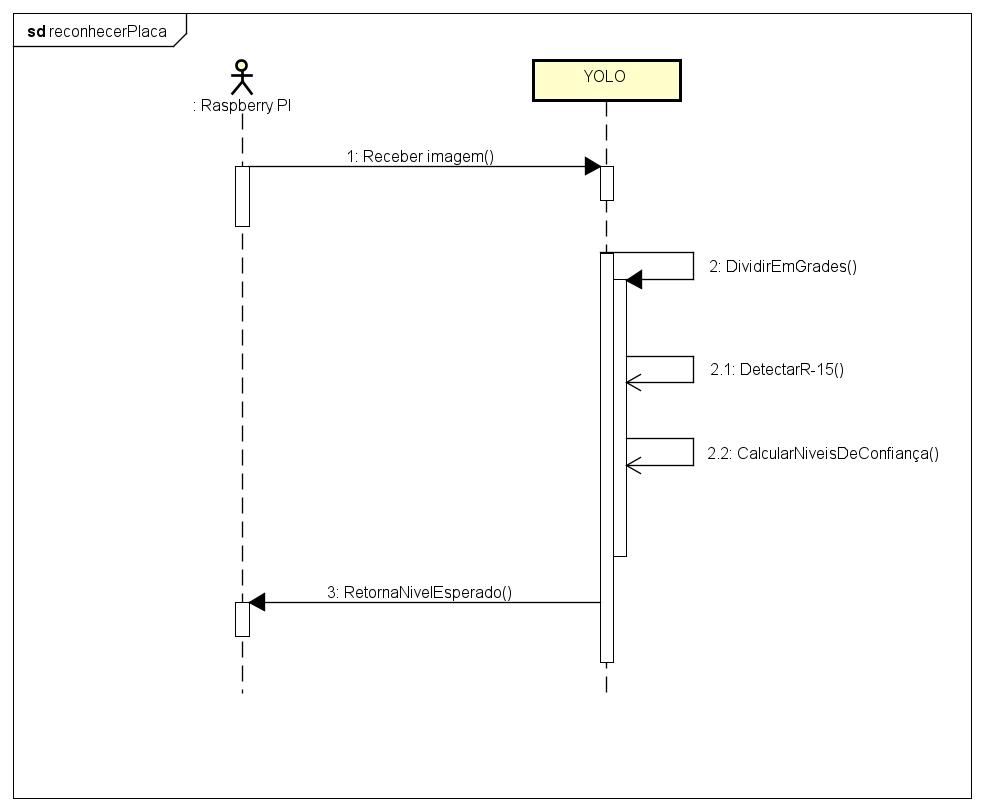
Fonte: Autoria Própria, 2025.

Figura - Diagrama de Sequência: Enviar Altura Do Caminhão



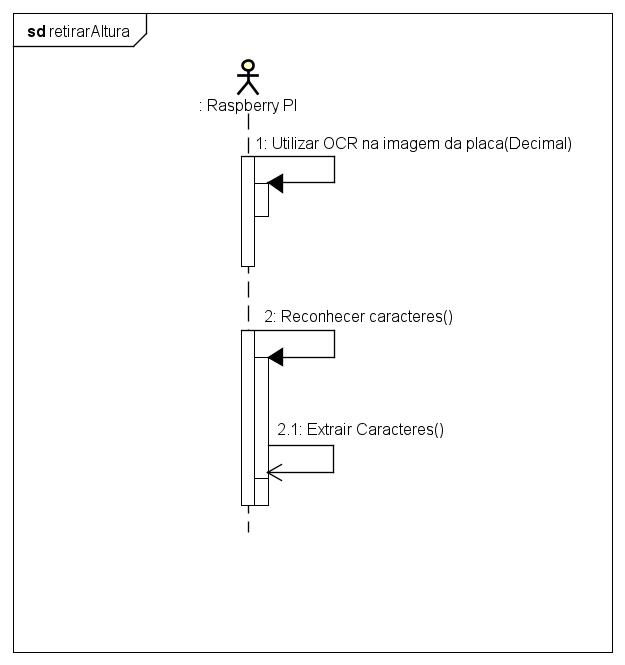
Fonte: Autoria Própria, 2025.

Figura - Diagrama de Sequência: Reconhecer Placa



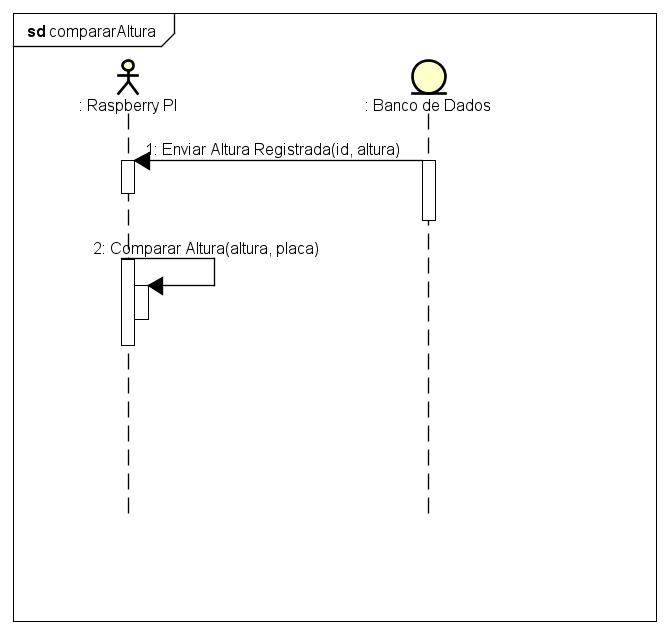
Fonte: Autoria Própria, 2025.

Figura - Diagrama de Sequência: Retirar Altura



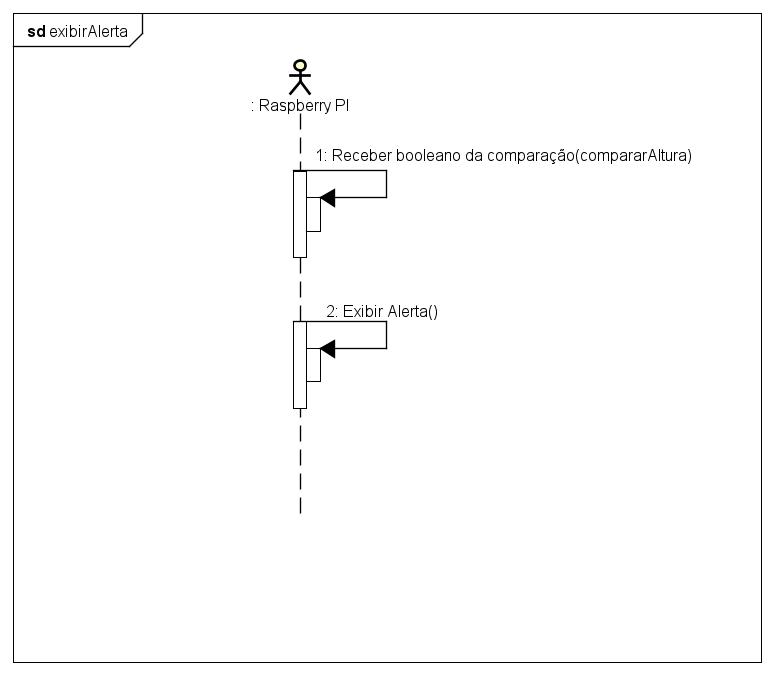
Fonte: Autoria Própria, 2025.

Figura - Diagrama de Sequência: Comparar Altura



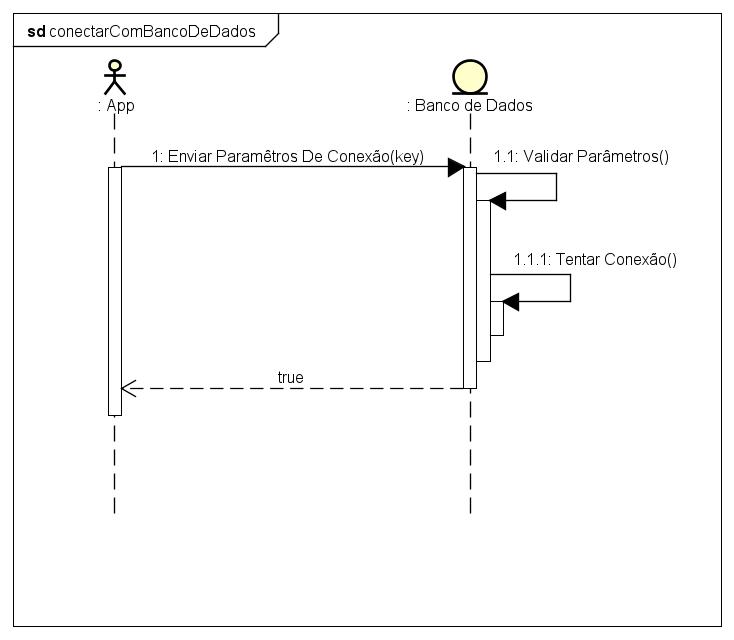
Fonte: Autoria Própria, 2025.

Figura - Diagrama de Sequência: Exibir Alerta



Fonte: Autoria Própria, 2025.

Figura - Diagrama de Sequência: Conectar Com o Banco de Dados



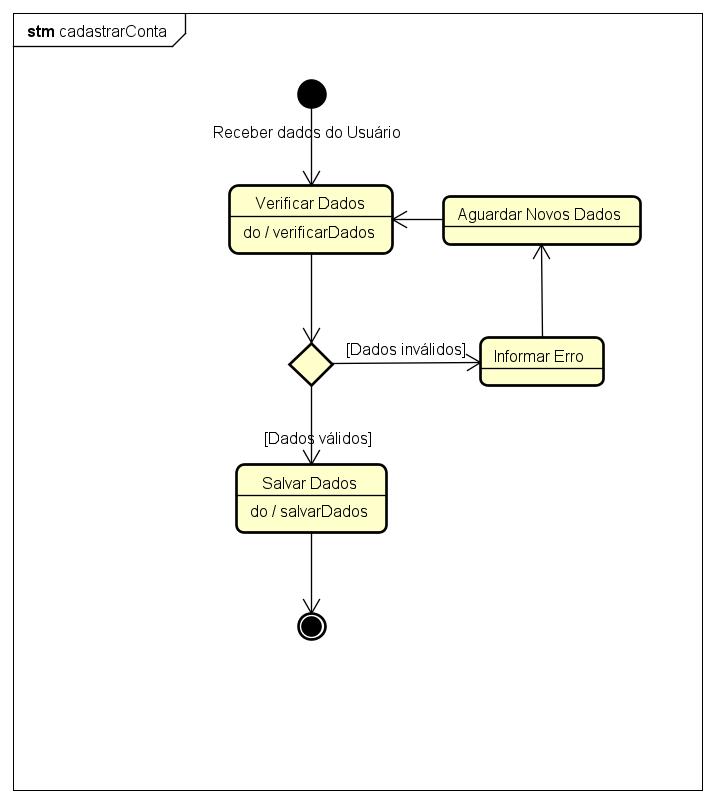
Fonte: Autoria Própria, 2025.

## Diagrama de Máquina de Estado

O diagrama de máquina de estado é empregado para descrever o comportamento do sistema, em função da representação dos diferentes estados que um objeto pode assumir ao longo do tempo diante da ocorrência de um evento.

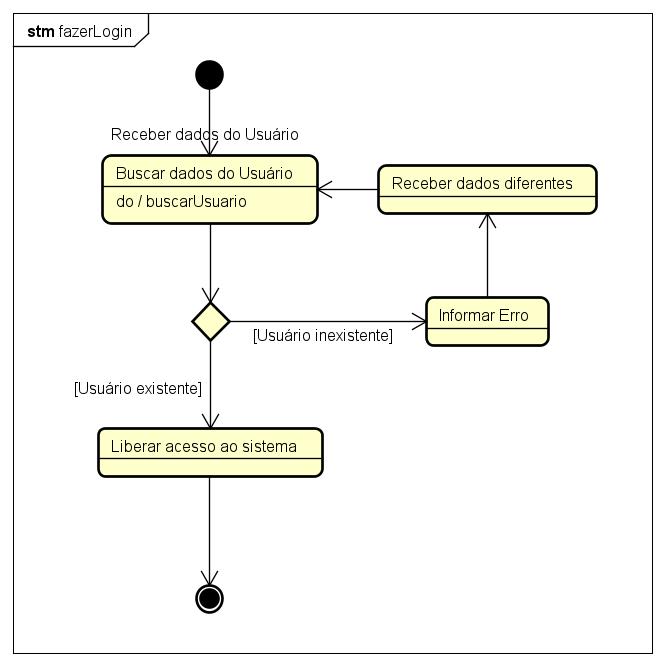
Ademais, este diagrama mostra como o sistema reage a determinadas ações, retratando as transições que acontecem durante a execução dos processos.

Figura - Diagrama de Máquina de Estado: Cadastrar Conta



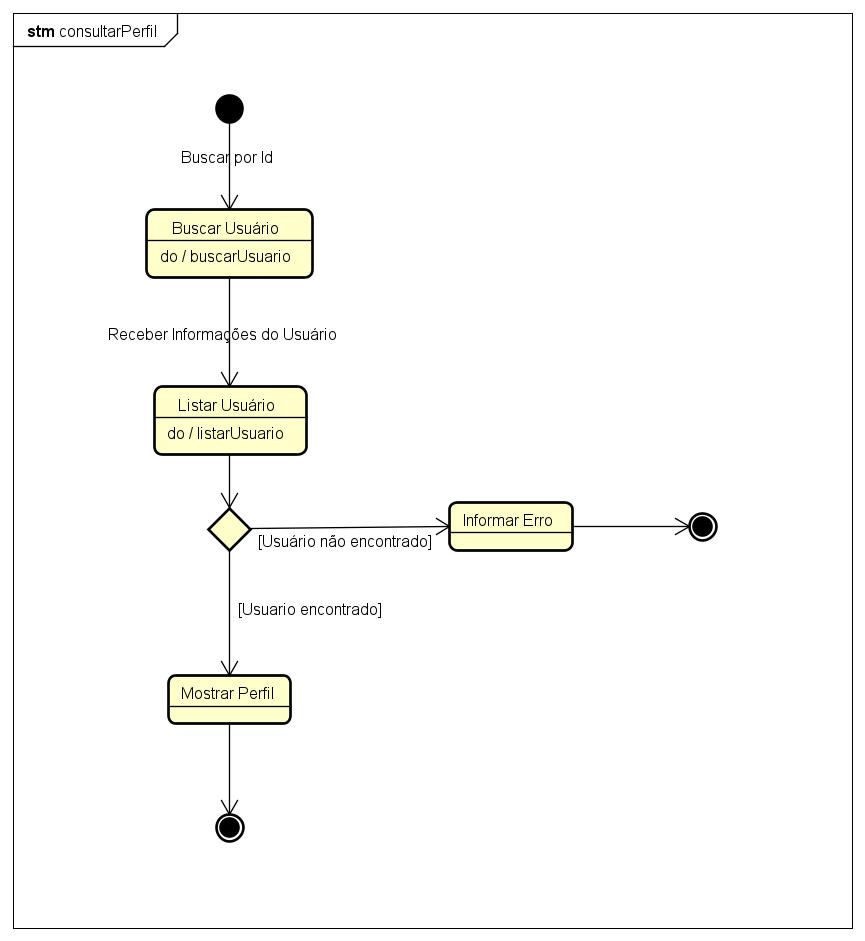
Fonte: Autoria Própria, 2025.

Figura - Diagrama de Máquina de Estado: Fazer Login



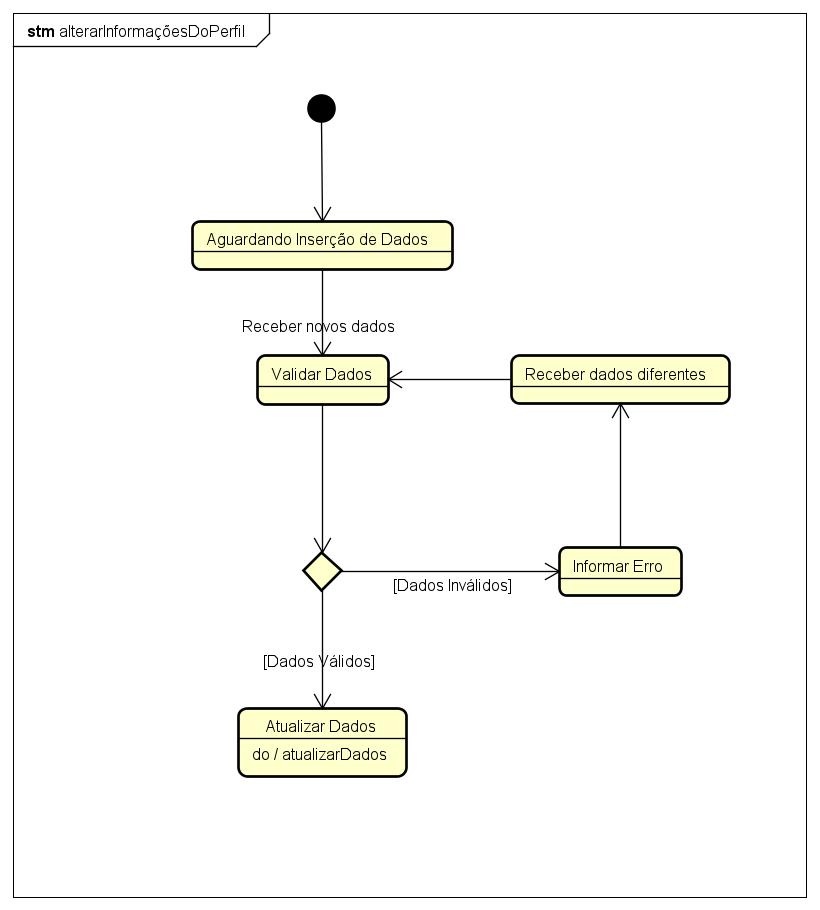
Fonte: Autoria Própria, 2025.

Figura - Diagrama de Máquina de Estado: Consultar Perfil



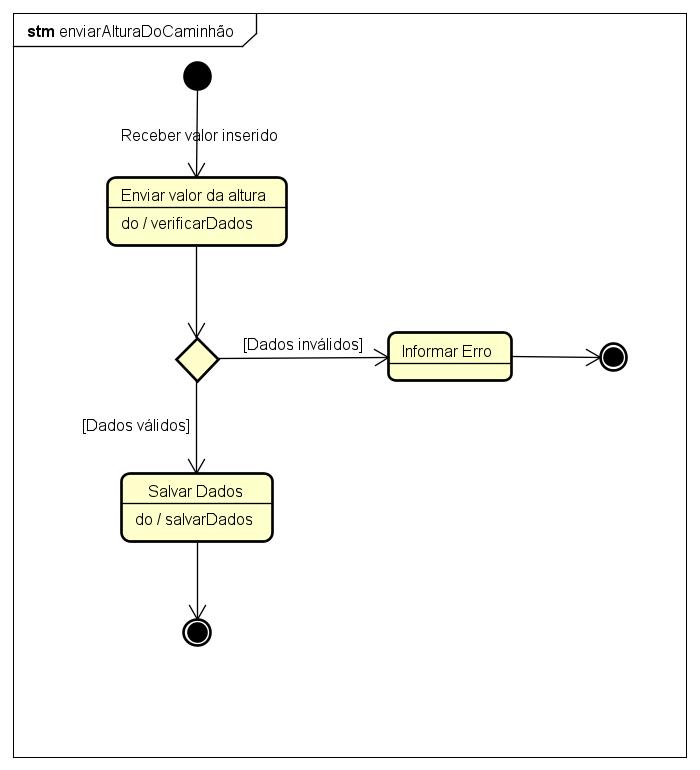
Fonte: Autoria Própria, 2025.

Figura - Diagrama de Máquina de Estado: Alterar Informações do Perfil



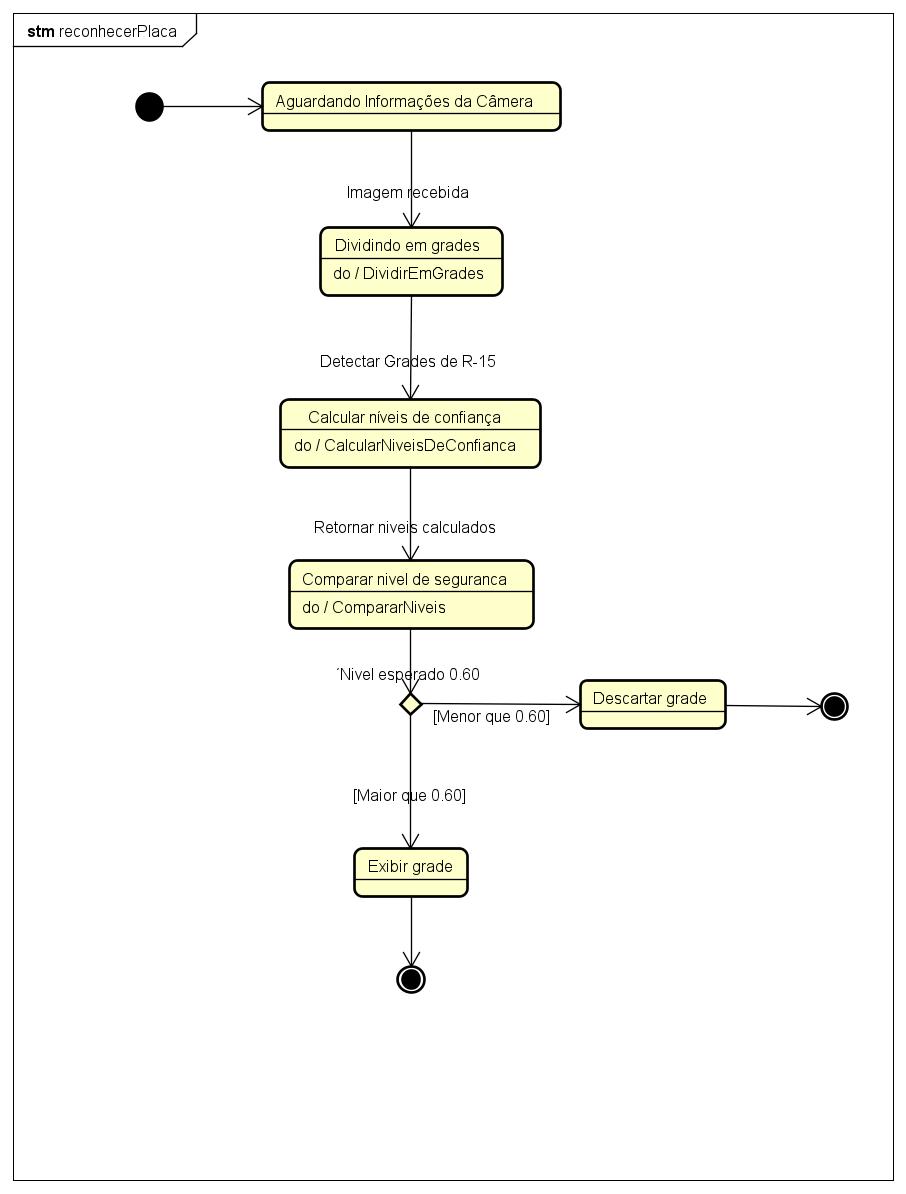
Fonte: Autoria Própria, 2025.

Figura - Diagrama de Máquina de Estado: Enviar Altura do Caminhão



Fonte: Autoria Própria, 2025.

Figura - Diagrama de Máquina de Estado: Reconhecer Placa



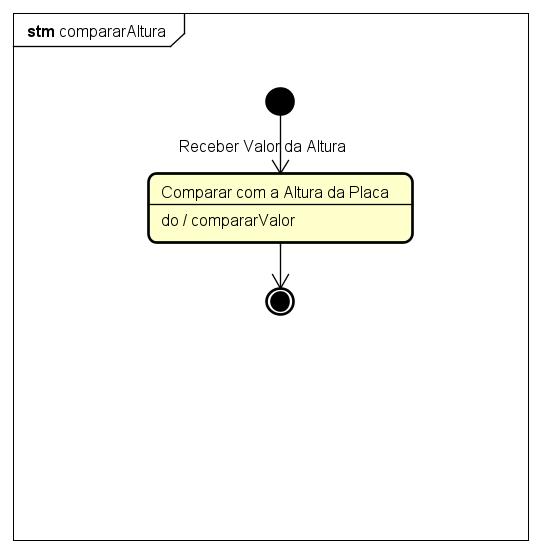
Fonte: Autoria Própria, 2025.

Figura - Diagrama de Máquina de Estado: Retirar Altura



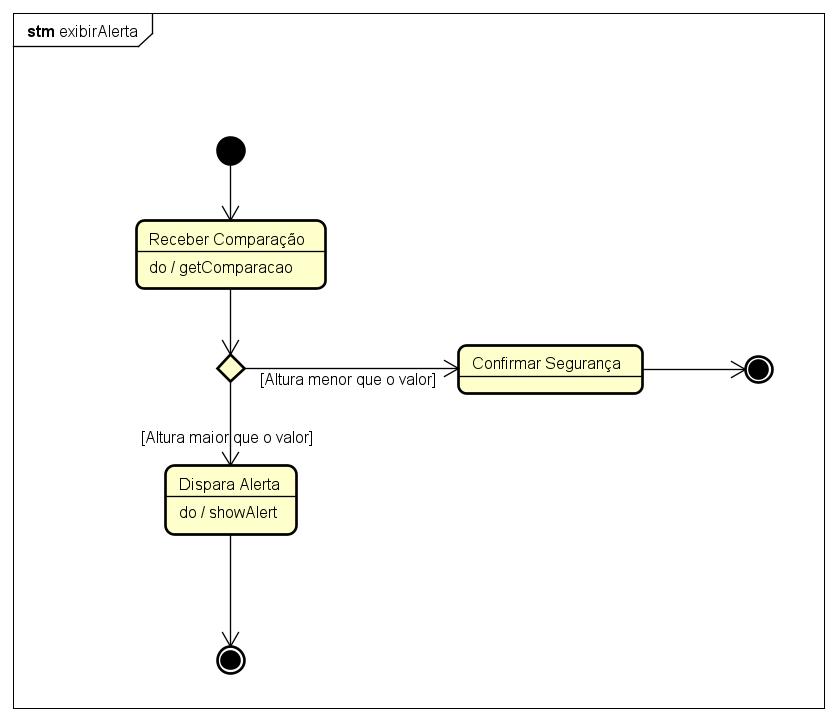
Fonte: Autoria Própria, 2025.

Figura - Diagrama de Máquina de Estado: Comparar Altura



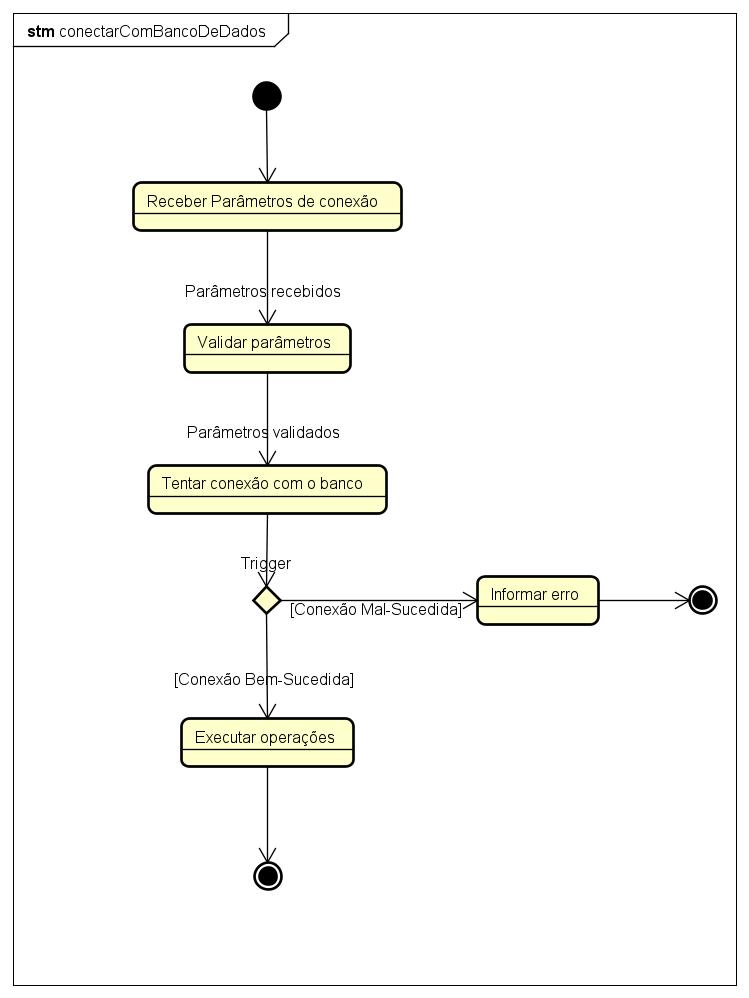
Fonte: Autoria Própria, 2025.

Figura - Diagrama de Máquina de Estado: Exibir Alerta



Fonte: Autoria Própria, 2025.

Figura - Diagrama de Máquina de Estado: Conectar Com o Banco de Dados



Fonte: Autoria Própria, 2025.

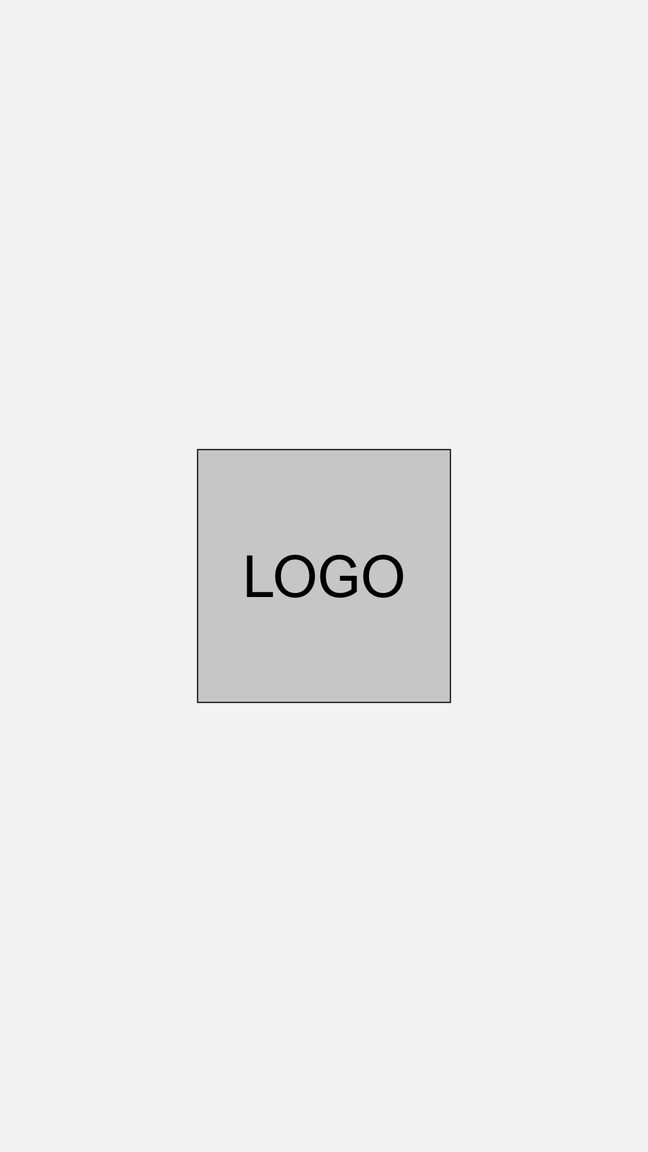
## Prototipação das telas do aplicativo

Esta seção tem como principal objetivo apresentar a prototipação das interfaces gráficas contendo os elementos visuais do aplicativo, as quais foram desenvolvidas por meio de uma técnica de design que consiste em desenhar esboços visuais conhecida como *wireframes* de baixa e alta fidelidade, utilizando recursos visuais para deixar a experiência do usuário mais intuitiva, objetiva, agradável e confortável, que atenda o público-alvo.

O primeiro protótipo de interface desenvolvido foi a tela de carregamento do aplicativo mobile, na qual a logo da marca foi posicionada de forma centralizada, ganhando destaque através de sua cor e de um fundo neutro, que realça ainda mais o logotipo, transmitindo suavidade.

O esboço inicial da tela de carregamento, conhecido como wireframe de baixa fidelidade, não tem a finalidade de foco em cores, estilos ou detalhes gráficos, e sim criar a ideia da estrutura gráfica onde os elementos principais ficaram posicionados:

Figura - Wireframe de Baixa Fidelidade: Tela de carregamento



Fonte: Autoria Própria, 2025.

Em seguida, pode-se observar o wireframe de alta fidelidade, que já contém cores e formas, aproximando-se do design final da tela de carregamento da aplicação mobile:

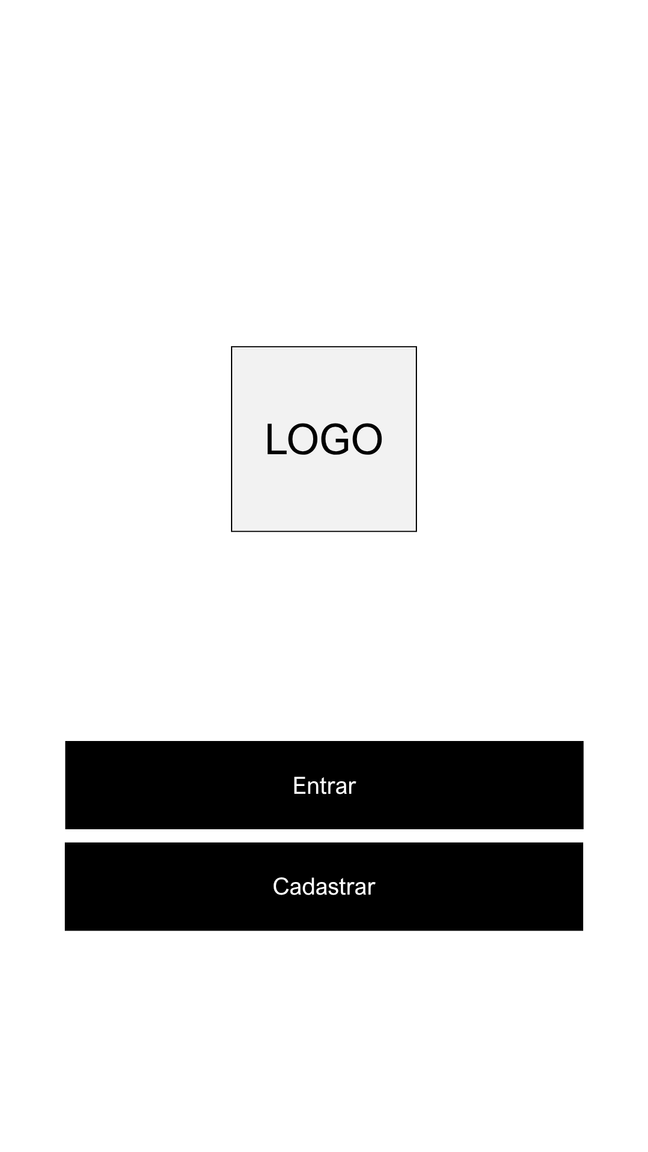
Figura - Wireframe de Alta Fidelidade: Tela de carregamento



Fonte: Autoria Própria, 2025.

O segundo wireframe exibe a interface da tela inicial do aplicativo mobile, que contém como elementos principais dois botões: um que redireciona o usuário para a tela de login e outro para iniciar a etapa de cadastro, levando-o à tela de encontrar dispositivo. Para iniciar a demonstração, utilizamos um wireframe de baixa fidelidade, com o objetivo de esboçar e organizar os elementos principais necessários em um layout básico, sem componentes visuais detalhados:

Figura - Wireframe de Baixa Fidelidade: Tela de início



Fonte: Autoria Própria, 2025.

Na figura abaixo, é possível observar o protótipo de alta fidelidade da tela inicial do aplicativo, com um design moderno e detalhado, aplicando a identidade visual do projeto e assemelhando-se à versão final da interface.

Figura - Wireframe de Alta Fidelidade: Tela de início



Fonte: Autoria Própria, 2025.

O próximo esboço visual apresenta o layout da tela de encontrar dispositivo, que possui uma breve descrição do objetivo da etapa e um campo de texto que deve ser preenchido com um código presente no aparelho, com a finalidade de localizá-lo no banco de dados. Caso o dispositivo não seja encontrado, exibe uma mensagem de erro e bloqueia a próxima etapa para cadastrar os dados do caminhoneiro. O layout é apresentado inicialmente por um wireframe de baixa fidelidade:

Figura - Wireframe de Baixa Fidelidade: Tela de encontrar dispositivo



Fonte: Autoria Própria, 2025.

Na sequência, é possível visualizar uma versão do wireframe de alta fidelidade, com elementos e componentes mais detalhados, proporcionando uma experiência mais agradável ao condutor do veículo:

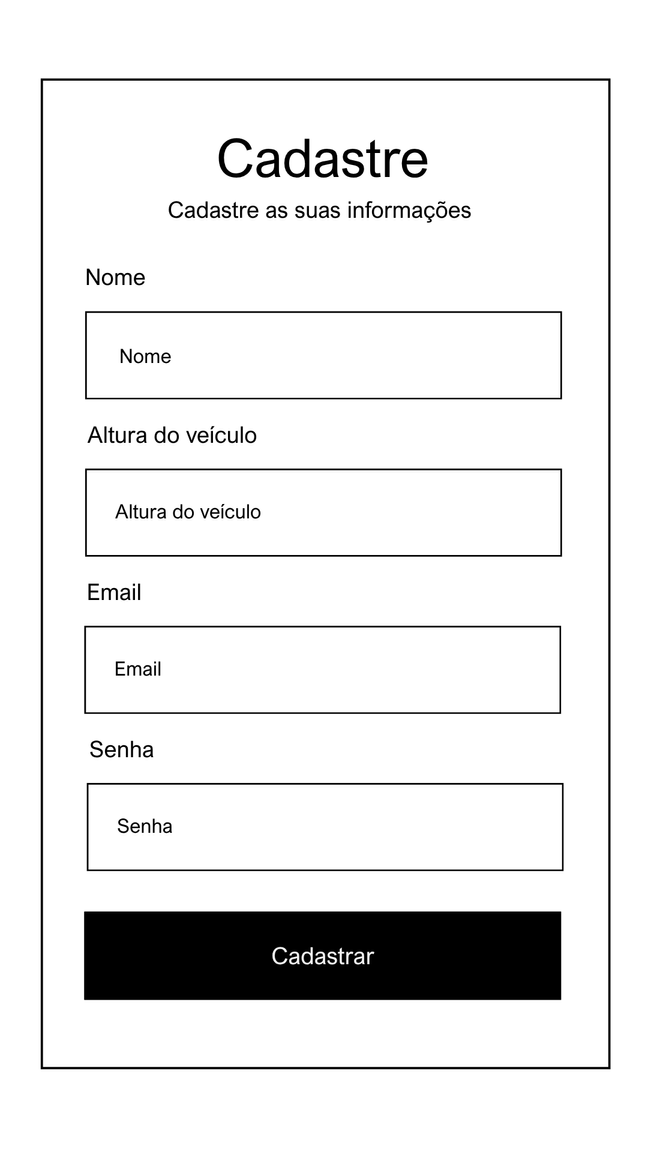
Figura - Wireframe de Alta Fidelidade: Tela de encontrar dispositivo



Fonte: Autoria Própria, 2025.

O quarto protótipo se refere à tela de cadastro, composta por quatro campos de texto: nome, altura do caminhão, e-mail e senha. Esses dados devem ser fornecidos para que o cadastro do usuário seja realizado no sistema associado ao seu dispositivo. No entanto, é crucial que a etapa anterior, encontrar dispositivo, seja concluída com sucesso, possibilitando a efetivação do cadastro e o redirecionamento do usuário à tela de login. A seguir, apresenta-se o wireframe de baixa fidelidade para idealizar esquematização dos componentes na interface:

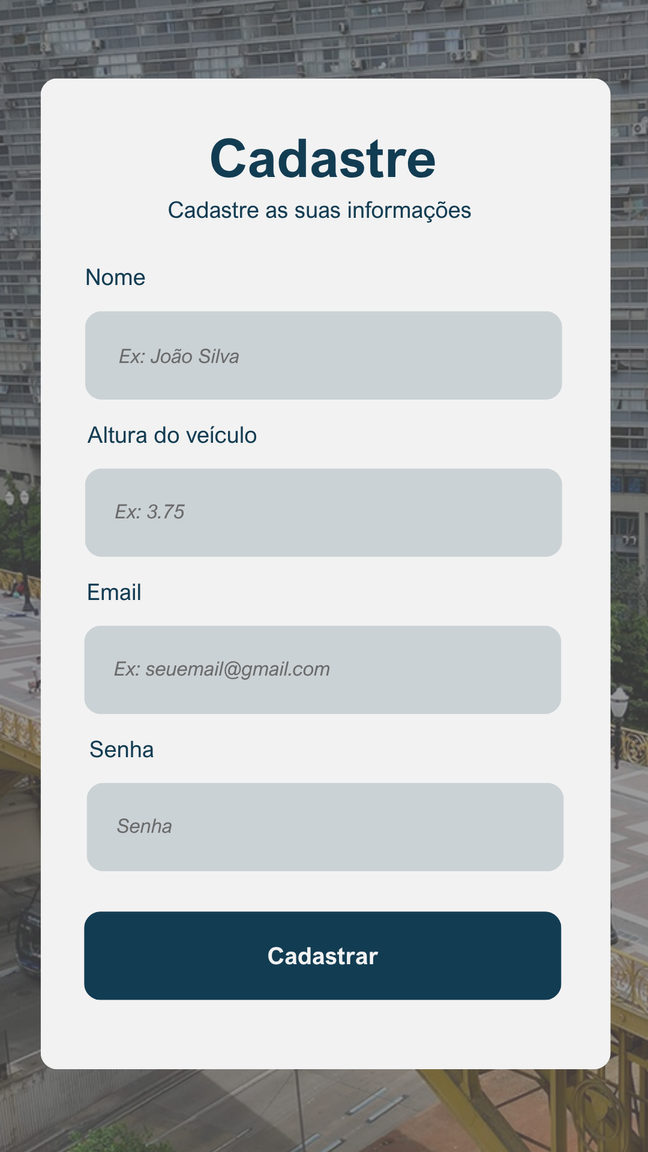
Figura - Wireframe de Baixa Fidelidade: Tela de cadastro



Fonte: Autoria Própria, 2025.

Logo após, é possível observar o wireframe de alta fidelidade, com elementos mais estruturados e melhor apresentados:

Figura - Wireframe de Alta Fidelidade: Tela de cadastro



Fonte: Autoria Própria, 2025.

A seguir, apresentamos o quinto desenho visual, que mostra a interface da tela de login. Ela é composta por dois campos de texto principais, essenciais para a autenticação do usuário: e-mail e senha. Caso os dados não sejam encontrados, será exibida uma mensagem de erro; caso contrário, o usuário será redirecionado a home do aplicativo. Há, também, um link que, se clicado, levará o usuário para a primeira etapa do cadastro. Logo abaixo, é possível observar a esquematização da tela com elementos básicos, utilizando a técnica de design wireframe de baixa fidelidade.

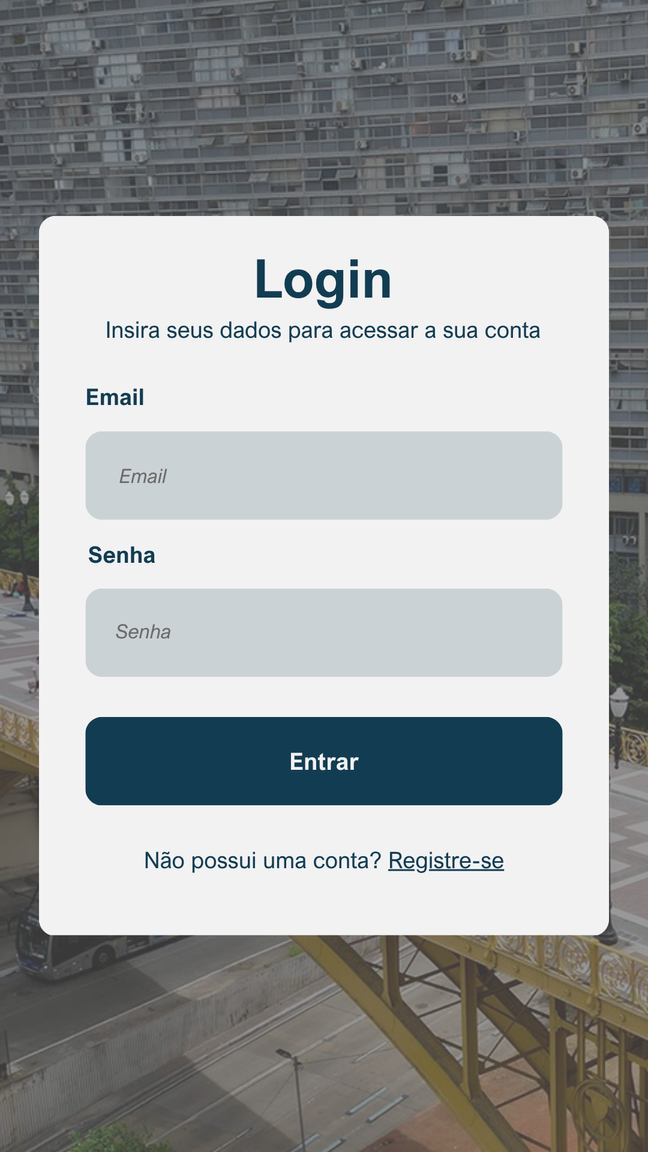
Figura - Wireframe de Baixa Fidelidade: Tela de login



Fonte: Autoria Própria, 2025.

Posteriormente, é apresentado o wireframe de alta fidelidade, com um design confortável e detalhado, visando a melhor experiência do motorista:

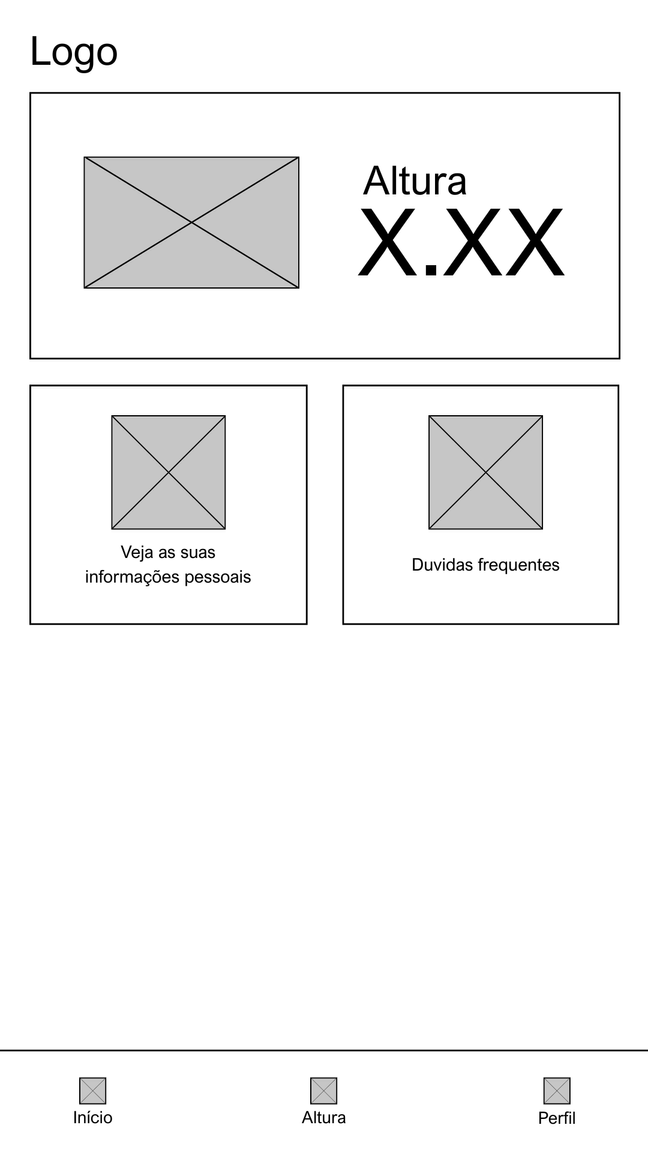
Figura - Wireframe de Alta Fidelidade: Tela de login



Fonte: Autoria Própria, 2025.

O sexto esboço exibe a interface que contém o dashboard da home do aplicativo, cujo acesso exige a autenticação do usuário. A interface possui três caixas de texto: a primeira mostra a altura do caminhão e, ao ser clicada, encaminha para a tela de edição da altura cadastrada; a segunda e a terceira são cards que, ao serem clicados, direcionam para as telas de edição de perfil e dúvidas frequentes, respectivamente. Na parte inferior da tela, encontra-se a barra de menu que facilita a navegação do usuário no aplicativo. Logo a seguir trata-se de um wireframe de baixa fidelidade, para melhor compreensão de onde os elementos são posicionados na tela:

Figura - Wireframe de Baixa Fidelidade: Tela da home



Fonte: Autoria Própria, 2025.

Na figura abaixo, é possível observar o wireframe de alta fidelidade da tela inicial, com um visual mais elaborado, aproximando-se do design final:

Figura - Wireframe de Alta Fidelidade: Tela home



Fonte: Autoria Própria, 2025.

O próximo protótipo tem foco em demonstrar a tela de edição de altura. O elemento principal presente na tela é a caixa de texto, onde é possível visualizar e editar a altura do caminhão previamente cadastrada pelo usuário. A tela também fornece uma breve descrição e uma barra de navegação, permitindo a transição entre as telas. Logo abaixo, é possível visualizar um wireframe de baixa fidelidade da tela:

Figura - Wireframe de Baixa Fidelidade: Tela de editar altura



Fonte: Autoria Própria, 2025.

Na imagem abaixo, apresenta-se o wireframe de alta fidelidade da tela de edição de altura, com um design mais refinado e alinhado à identidade visual do projeto, aproximando-se da interface final:

Figura - Wireframe de Alta Fidelidade: Tela de editar altura



Fonte: Autoria Própria, 2025.

O sétimo esboço visual apresenta a tela de edição de perfil, que tem como principal objetivo fornecer os dados do usuário extraídos do banco de dados e permitir a atualização dessas informações, caso necessário. A interface é composta por campos de texto e um botão para salvar as alterações. Também é possível observar a opção de sair da conta, caso o motorista veja essa necessidade. A seguir é demonstrado o wireframe de baixa fidelidade com apenas os itens principais e sem estilização:

Figura - Wireframe de Baixa Fidelidade: Tela de editar perfil



Fonte: Autoria Própria, 2025.

Em continuidade, é apresentado o wireframe de alta fidelidade, com os principais elementos estilizados e visualmente atrativos, aproximando-se da interface final:

Figura - Wireframe de Alta Fidelidade: Tela de editar perfil



Fonte: Autoria Própria, 2025.

# CONCLUSÃO

# REFERÊNCIAS

ALBINO, Igor Cacerez. **Dominando o CSS:** um guia completo para desenvolvedores web. São Paulo: Clube de Autores, 2023.

ALMEIDA, F. C.; SOUSA, A. L. de; PEREIRA, M. G. A. **Avaliação de técnicas de aprendizado de máquina para classificação de dados meteorológicos em sistemas agrícolas**. Revista Brasileira de Ciência do Solo, v. 43, p. 1-10, 2019. Disponível em: https://apct.sede.embrapa.br/cct/article/view/26381/14242.

ARAÚJO, Aline Moura. **Detecção e Destaque em Vídeo de Objetos utilizando YOLO.** 2022. 69 p. Dissertação (Programa de Pós-Graduação em Informática) - Universidade Federal da Paraíba, João Pessoa, 2022.

BARELLI, Felipe. **Introdução à Visão Computacional**: Uma abordagem prática com Python e OpenCV. 1. São Paulo: Casa do Código, 2018.

BOOCH, Grady. **UML: Guia do Usuário**. 1. São Paulo: Campus, 2006.

BORGES, Luiz Eduardo. **Python para Desenvolvedores:** Aborda Python 3.3. 1. ed. São Paulo: Novatec Editora Ltda, 2014.

CARRION, Patricia; QUARESMA, Manuela. Internet da Coisas (IoT): Definições e aplicabilidade aos usuários finais. **Human Factors in Design**, Florianópolis, v. 8, n. 15, p. 049–066, 2019. DOI: 10.5965/2316796308152019049. Disponível em: https://www.revistas.udesc.br/index.php/hfd/article/view/2316796308152019049.

CARVALHO, Vinícius. **MySQL: Comece com o principal banco de dados open source do mercado**. Editora Casa do Código, 2015.

CORDEIRO, Michelle Sousa. **Alfabetização assistida por visão computacional:** detecção de cartões e reconhecimento de letras. 2024. Trabalho de Conclusão de Curso (Bacharelado em Ciências e Tecnologia, com ênfase em Computação Aplicada) – Universidade Federal do Rio Grande do Norte, Escola de Ciências e Tecnologia, Natal, RN, 2024.

COELHO, Pedro Citadin; ANTUNES, Luciano. **Aplicação móvel voltada ao compartilhamento de informações acerca de praias integrando a ferramenta Google Maps e a Biblioteca React Native**. 2023. Trabalho de Conclusão de Curso (Bacharelado em Ciência da Computação) – Universidade do Extremo Sul Catarinense, UNESC, Criciúma, 2023.

MILANO, Danilo de; HONORATO, Luciano Barrozo. **Visão Computacional**. UNICAMP Universidade Estadual de Campinas FT Faculdade de Tecnologia, 2014.

EBERMAM, Elivelto; PESENTE, Guilherme Moraes; RIOS, Renan Osório; PULINI, Igor Carlos. **Programação para leigos com Raspberry Pi**. Vitória: Edifes; João Pessoa: Editora IFPB, 2017.

EIS, Diego. **Guia Front-End:** O caminho das pedras para ser um dev Front-End. São Paulo: Casa dos Códigos, 2015

ESCUDELARIO, Bruno; PINHO, Diego. **React Native:** Desenvolvimento de aplicativos mobile com React. São Paulo: Casa do Código, 2020.

FALCÃO, Filipe Dourado. **Desenvolvimento do aplicativo Turistando Beberibe utilizando React Native.** 2022. Trabalho de Conclusão de Curso (Bacharelado em Sistemas e Mídias Digitais) – Universidade Federal do Ceará, Instituto Universidade Virtual (UFC Virtual), Fortaleza, 2022.

FIGMA. **O que é o Figma?**. Disponível em https://help.figma.com/hc/pt-br. Acesso em: 09 ago. 2025.

FLANAGAN, David. **JavaScript**: O Guia Definitivo. Porto Alegre: Bookman, 2012.

FLORENCIO, Felipe de Almeida. **Estudo comparativo do desempenho de bibliotecas para redes neurais convolucionais em diferentes microarquiteturas de GPU**. 2020. Dissertação (Mestrado em Ciência da Computação) – Universidade Federal de Sergipe, São Cristóvão, SE, 2020.

FOWLER, Martin. **UML Essencial**: Um Breve Guia para a Linguagem-padrão de Modelagem de Objetos. 3. Porto Alegre: Bookman, 2005.

FREITAS, Pabricio Bonfim. **Projeto e desenvolvimento de um sistema de gerenciamento de chaves (SGC) físicas convencionais para instituições de ensino**. 2023. Trabalho de Conclusão de Curso (graduação em Tecnologia em Análise e Desenvolvimento de Sistemas) - Instituto Federal de Educação, Ciência e Tecnologia da Bahia Campus Valença, Valença, 2023.

GALVÃO, Pedro Sereno. **Comprehensive Repository Analysis of Mobile Projects Built with Reactive Native**. 2018. 28 f. Dissertação (Programa de Graduação em Ciência da Computação) – Universidade Federal de Pernambuco, Recife, 2018.

GÉRON, Aurélien. **Mãos à obra**: aprendizado de máquina com Scikit-Learn, Keras e TensorFlow: conceitos, ferramentas e técnicas para a construção de sistemas inteligentes. 2. ed. Rio de Janeiro: Alta Books, 2020.

GONÇALVES, Eduardo. **SQL:** Uma abordagem para bancos de dados Oracle. 1. ed. São Paulo: Casa do Código, 2014.

GOOGLE. **Firebase**. Disponível em: https://firebase.google.com/. Acesso em: 11 ago. 2025.

GUEDES, Gilleanes. **UML 2: Uma Abordagem Prática**. 3. São Paulo: Novatec Editora Ltda, 2018.

IEPSEN, Edécio Fernando . **Lógica de Programação e Algoritmos com Javascript**. São Paulo: Novatec, 2018.

IHRIG, Colin J. **Pro Node.Js Para Desenvolvedores**. 1. ed. Rio de Janeiro: Ciência Moderna, 2020.

KOVÁCS, Zsolt. **Redes Neurais Artificiais**: Fundamentos e Aplicações. 4. ed. São Paulo: Editora Livraria da Física, 2023.

LEOCÁDIO, Rodolfo R. V. et al. **Detecção de Abelhas Nativas em Colmeias em Campo Utilizando Visão Computacional**. In: WORKSHOP DE COMPUTAÇÃO APLICADA À GESTÃO DO MEIO AMBIENTE E RECURSOS NATURAIS (WCAMA), 12., 2021, Porto Alegre: Sociedade Brasileira de Computação, 2021. p. 59-68.

MACHADO, Kheronn Khennedy. **Angular 11 e Firebase**: Construindo uma aplicação integrada com a plataforma do Google. São Paulo. Casa do Código, 2021.

MAGRANI, Eduardo. **A internet das coisas**. Rio de Janeiro: FGV Editora, 2018.

MARENGONI, M.; STRINGHINI, S. Tutorial: Introdução à Visão Computacional usando OpenCV. **Revista de Informática Teórica e Aplicada**, *[S. l.]*, v. 16, n. 1, p. 125–160, 2010. DOI: 10.22456/2175-2745.11477. Disponível em: https://seer.ufrgs.br/index.php/rita/article/view/rita\_v16\_n1\_p125. Acesso em: 27 maio. 2025.

MATIOLA, Guilherme Schlindwein. **Aplicativo móvel para controle do processamento de dados de amostras de solo com integração da tecnologia RFId**. 2023. Trabalho de Conclusão de Curso (Graduação em Engenharia de Controle e Automação) - Universidade Federal de Santa Catarina Campus Blumenau, Bluemenau, 2023.

MATTHES, Eric. **Curso Intensivo de Python:** uma Introdução Prática e Baseada em Projetos à Programação. São Paulo: Novatec Editora Ltda, 2016.

MCKINNEY, Wes. **Python para Análise de Dados**: Tratamento de dados com pandas, NumPy e Jupyter. 3. ed. São Paulo: Novatec Editora Ltda, 2023.

MENEZES, Nilo Ney Coutinho. **Introdução a programação com Python.** São Paulo: Novatec, p. 13, 2010.

OLIVEIRA, Cláudio Luis Vieira; NABARRO, Cristina Becker Matos; ZANETTI, Humberto Augusto Piovesana. **Raspberry Pi Descomplicado**. 1. ed. São Paulo: Editora Érica., 2018. 224 p.

OLIVEIRA, Sérgio De. **Internet Das Coisas Com Esp8266, Arduino E Raspberry Pi**. 2. ed. São Paulo: Novatec, 2021.

OSÓRIO, Abrão Júnior. **Interface gráfica para busca de profissionais criativos.** 2020. 101 f. Trabalho de Conclusão de Curso (Graduação em Design) – Universidade Federal de Uberlândia, Uberlândia, 2021.

PACHECO, César Augusto Rodrigues. **Deep Learning**: Conceitos e Utilização nas Diversas Áreas do Conhecimento. 2018. 32 f. Trabalho de Conclusão de Curso (Engenharia de Computação) – Centro Universitário de Anápolis, 2018.

PEREIRA, Caio Ribeiro. **Aplicações web real-time com Node.js**. São Paulo: Casa do Código, 2014.

PEREIRA, Matheus de Matos. **Aprendizado Profundo**: Redes LSTM. 2017. 40 f. Trabalho de Conclusão de Curso (Sistemas de Informação) – Universidade Federal da Grande Dourados, 2017.

RIBEIRO, Joyce. **A cada dois dias, um caminhão entala em viadutos de São Paulo**. R7, 2019. Disponível em: https://noticias.r7.com/sao-paulo/a-cada-dois-dias-um-caminhao-entala-em-viadutos-de-sao-paulo-30102019/. Acesso em: 10 set. 2025.

RODRIGUES, Arthur. **Quinze caminhões ficam presos em viadutos todo mês em SP e abalam estruturas**. Folha de S.Paulo, 2019. Disponível em: https://www.folha.uol.com.br/cotidiano/2019/02/quinze-caminhoes-ficam-presos-em-viadutos-todo-mes-em-sp-e-abalam-estruturas. Acesso em: 07 abr. 2025.

ROSA, João Vitor de Moura. **Desenvolvimento de uma aplicação web para simulados preparatórios de vestibular com enfoque no ENEM**. 2023. Trabalho de Conclusão de Curso (Sistemas de Informação) – Universidade Federal de Ouro Preto, João Monlevade, 2023.

SANTOS, Leanderson da Silva**. Gerenciador de Projetos da Agenda 2030 da Organização das Nações Unidas (GPA2030)**. 2023. Trabalho de Conclusão de Curso (Tecnologia em Análise e Desenvolvimento de Sistemas) – Instituto Federal de Educação, Ciências e Tecnologia do Amazonas, Campus Manaus Centro, Manaus, 2023.

SANTOS, José A. dos; SILVA, Felipe L. da. **Um estudo sobre a aplicação de algoritmos de aprendizado de máquina para a detecção de fraudes em sistemas bancários**. Simpósio Brasileiro de Sistemas de Informação (SBSI), v. 1, p. 100-110, 2019. Disponível em: <https://sol.sbc.org.br/index.php/sbsi/article/view/5818/5716>.

SANTOS, Sandro. **Introdução à IoT**: desvendando a Internet das Coisas. Santa Catarina: Clube de Autores, 2019.

SETTE, Guilherme; CARVALHO, Luiz; BASILE, Antonio. ***Reconhecimento facial pela região dos olhos utilizando OpenCV***. 2021. Trabalho de Conclusão de Curso (Graduação em Ciência da Computação) – Faculdade de Computação e Informática, Universidade Presbiteriana Mackenzie, São Paulo, 2021.

SILVA, F. A. S.; IGLESIAS, W. A. N.; BOER, M. T. **Utilização da prototipagem e criação do front-end pelo Figma para um aplicativo móvel de pets**. 2024. Artigo de Graduação (Tecnologia em Sistemas para Internet) – Faculdade de Tecnologia Prof. José Camargo, Jales, 2024.

SILVA, Daniela Rocha; SOBRAL, Luis Felipe Bentin. **Um Estudo em Larga Escala sobre a Estrutura do Código-fonte de Pacotes JavaScript**. 2017. Graduação (Bacharel em Sistemas de Informação)- UNIRIO, 2017.

SILVA, Gabriel de Oliveira; DIAS, Wanderson Roger Azevedo; ESCUDERO, Danilo Pereira. **Análise do Desempenho Computacional da Raspberry Pi Executando o Benchmark SysBench**. Simpósio em Sistemas Computacionais de Alto Desempenho (SSCAD). SBC, 2022.

SILVA, João Victor Negreiros da; CASTRO, Jorge Vinícius Santos. **DOCKERYZER - Desenvolvimento de uma aplicação de linha de comando para criação automática de dockerfiles em projetos NODE.JS com uso de inteligência artificial**. 2025. 28 f. Trabalho de Conclusão de Curso (Bacharelado em Engenharia de Computação) – Instituto Federal de Educação, Ciência e Tecnologia da Paraíba, Campina Grande, 2025.

SILVA, Mauricio Samy. **Construindo Sites com CSS e (x)HTML**. 1. ed. São Paulo: Novatec, 2007.

SILVA, Tayná Cristina Sousa. **Detecção e classificação de derramamento de óleo na superfície oceânica baseada em aprendizagem profunda via algoritmo** **YOLO**. 2024. 75 f. Dissertação (Programa de Pós-Graduação em Engenharia de Eletricidade/CCET) - Universidade Federal do Maranhão, São Luís, 2024.

SOUZA, Caio Adriano Xavier de. **Previsão da demanda de materiais de construção utilizando redes neurais artificiais: uma comparação entre TensorFlow e PyTorch**. 2024. Monografia (Graduação em Engenharia de Produção) - Universidade Federal de Ouro Preto, Escola de Minas, Ouro Preto, MG, 2024.