Gyalpozhing College of Information Technology

Royal University of Bhutan

Kabjisa, Chamjekha, Thimphu

AI and Data Science

PRJ303 3rd Year Project

Spring 2024

AI Vision Assistant for Drivers (Third Eye System)

Submitted by:

Chimi Rinzin 12210002

Deepak Ghalley 12210004

Karma Deki 12200014

Sagar Lepcha 12200029

Sonam Choden 12200030

Supervisor: Ms.Ugyen Choden

Table of Contents

1. Project Background

Driver safety is a significant global concern, with hundreds of accidents occurring daily due to factors like fatigue, poor vision, and distracted driving. Traditional driver assistance systems offer only partial solutions to these issues. However, over the past couple of decades, significant advancements have been made in AI-driven driving assistance systems. Initially focused on basic functions like adaptive cruise control and lane-keeping assistance, these systems have evolved dramatically with the development of AI and computer vision technology. This evolution has culminated in the creation of the AI Vision Assistant for Drivers, commonly known as the Third Eye System.

The Third Eye System represents a major leap in driving safety technology by integrating various advanced features into a single, cohesive system that aims to help drivers navigate roads more safely and efficiently. This comprehensive solution has gained considerable traction in the automotive industry in recent years due to its potential to significantly reduce accidents and enhance road safety. Automakers and tech companies are increasingly incorporating AI Vision Assistant technology into their vehicles, prioritizing driver convenience and safety.

The Third Eye System utilizes cameras mounted inside and outside the car, along with sophisticated computer vision algorithms, to monitor both the driver and the surrounding environment. These cameras connect to an application installed on the driver's phone, which processes the visual data in real-time. The system can detect traffic signs, lane deviations, and potential forward collisions, providing the driver with immediate alerts to increase their awareness and help prevent accidents.

By offering real-time notifications and enhancing the driver's situational awareness, the Third Eye System aims to address common causes of accidents and improve overall driving safety. As technology continues to advance, the impact of the Third Eye System on driving safety is expected to grow, heralding a new era in the evolution of driver assistance systems.

2. Problem Statement

In Bhutan, road safety is a big concern due to more vehicles and violations. Addressing Bhutan's pressing road safety issues, this research focuses on making a Traffic Sign Detection App for Bhutanese drivers. Using deep learning, the app detects signs in real-time, alerts to drivers and promotes traffic law adherence. By using AI, it aims to reduce accidents and casualties, improving road safety and quality of life for Bhutanese citizens.

3. Aim

- To develop a user-friendly application for drivers to enhance road safety and real-time assistance.

4. Objectives

- Develop a deep learning model to accurately detect and interpret Bhutanese traffic signs in real-time, aiming for a minimum accuracy rate of 80%.
- Implement robust security measures to protect user data and prevent unauthorized access, ensuring compliance with privacy regulations.
- Integrate an audio alert system into the app to offer real-time auditory cues and warnings, improving drivers' awareness and responsiveness to traffic conditions.

5. Features

1. User Authentication:

- Sign-Up and Sign-In: Users must sign up and sign in to use the system. This ensures secure access and personalized user experience.
- Email Verification: During the sign-up process, an OTP (One-Time Password) is sent to the user's email for verification. This step ensures the authenticity of the user’s email address and secures the account.

2. Traffic Signs Detections:

- Camera Integration: The system uses cameras to detect traffic signs such as horn prohibited, stop signs, no entry signs.

- Real-Time Detection: Advanced computer vision algorithms process visual data in real-time to identify traffic signs accurately.

3. **Audio Alerts:**

- Immediate audio notifications are provided to inform the driver of detected traffic signs, helping them stay aware without diverting their attention from the road.

4. **Profile Viewing:**

- View Profile: Users can view their profiles through the application, allowing them to see their personal information and system usage statistics.
- User-Friendly Interface: The profile section is designed to be intuitive and easy to navigate, ensuring a seamless user experience.

5. **Integration with Mobile Device:**

- Application-Based System: The Third Eye System integrates with the driver's smartphone, utilizing its cameras and processing power.
- Real-Time Data Processing: The application processes data from the cameras in real-time, running sophisticated computer vision algorithms to provide accurate alerts and notifications.

6. Scope

6.1. User Scope

- The end users of the Traffic Sign Detection App are Bhutanese drivers.
- The app will be designed to cater to the needs and preferences of Bhutanese drivers,considering factors such as language, culture, and driving habits.

6.2. System scope

- The AI Vision Assistant within the app would provide real-time assistance to drivers by detecting and interpreting Bhutanese traffic signs.
- Drivers would receive audio alerts regarding detected traffic signs, enhancing their awareness of road conditions.

7. Feasibility

7.1. Technical Feasibility

- Infrastructure: Assessed the availability of internet connectivity and mobile network coverage across Bhutan, especially in rural areas.
- Technology: Determined if the necessary AI technology (such as natural language processing, speech recognition, and machine learning) can be developed or adapted for languages and driving conditions.
- Hardware: Ensured that drivers have access to smartphones or in-car devices that can run the AI assistant smoothly.

7.2. Operational Feasibility :

- Development Costs: Estimated the costs associated with developing, testing, and deploying the AI assistant, including software development, localization, and hardware integration.
- Funding and Investment: Identified potential sources of funding, such as government grants, private investors, or partnerships with tech companies.
- Market Demand: Conducted market research to understand the demand among Bhutanese drivers and their willingness to pay for such a service.

7.3. Legal and Ethical Feasibility:

- Regulations: Reviewed local laws and regulations related to data privacy, AI usage, and road safety to ensure the AI assistant complies with all legal requirements.
- Liability: Addressed potential liability issues in case of accidents or errors caused by the AI assistant.

7.4. Schedule Feasibility:

- Timeline: Developed a realistic timeline for the project, from initial research and development to pilot testing and full-scale deployment.
- Milestones: Set clear milestones and deliverables to track progress and ensure the project stays on schedule.

8. Requirements

8.1. Functional Requirements

As show below are no

8.1.1. User Authentication

- Users could login and register into the app practicing data security and safety.

8.1.2. User Profile Management

- Users could view their own profile in the app.

8.1.3. Traffic Sign Detection

- Used advanced tools of deep learning to get real-time traffic sign detection.

8.1.4. Audio Alert System

- Used Audio to signal the drivers about the upcoming traffic sign.

8.2. Non- Functional Requirement

8.2.1. Portability

- The app would be used from anywhere as long as it has an internet connection.

8.2.2. Availability

- The service would be available to all users anytime.

8.2.3. Reliability

- The system would be trustworthy and fault-tolerant.

8.2.4. User-Friendly UI

- The UI of the app would have application of color theory and interactive designs to let users indulge themselves. The app UI would cater to a diverse audience.

8.2.5. Security

- Relevant user data should be encrypted. User passwords must be encrypted to protect the user information from attackers and hackers. The application should employ mechanisms to prevent data being accessed by unauthorized users.

8.2.6. Scalability

- The application would be able to accommodate any number of users, i.e., any number of users should be able to register and use the application without any hindrance.

8.2.7. Usability

- The system would be tested before deploying to ensure that the application is usable and works as required.

8.3. System Requirement

Tools and Technologies:

8.3.1. Software Specifications

8.3.1.1. For Developing the Model:

- Anaconda (Jupyter notebook)
- Google Colab
- Python
- PyTorch
- TensorFlow
- YOLO
- scikit-learn (sklearn)
- Matplotlib
- Pandas

8.3.1.2. For Database:

- SQLite

8.3.1.3. For Integrating the Model with Android App:

- Front-end (Android Studio)
- Back-end (Java, Android Studio)
- Programming languages: Java
- Version control systems: GitLab

8.3.2. Hardware specifications

- Laptop: Ram = 8GB
- Core processors = i3

9. Data Sources

The dataset utilized in this project is a combination of two reputable sources, each contributing a wealth of data to enhance the training and testing of autonomous vehicle algorithms.

1. **Source 1: Roboflow Universe Self-Driving Cars Dataset**

   The Self-Driving Cars Dataset from Roboflow Universe provides a valuable resource for training and testing autonomous vehicle algorithms. It consists of 4,969 samples of traffic sign images, spanning 15 distinct classes. These classes include Green Light, Red Light, various Speed Limit categories (10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110, 120), and the Stop sign. The dataset encompasses diverse scenarios and environments, offering a comprehensive collection of images that depict various driving conditions.

- **URL: Roboflow Universe Self-Driving Cars Dataset**

2. **Source 2: Chinese Traffic Sign Recognition Database**

   The Self-Driving Cars Dataset, derived from the Chinese Traffic Sign Recognition Database and further explored by members of the Riga Data Science Club, presents an exciting opportunity for researchers and enthusiasts in computer vision and autonomous driving technology. This dataset comprises 5,998 traffic sign images spanning 58 categories, providing a rich resource for training and fine-tuning convolutional neural networks (CNNs).

- **URL**: Chinese Traffic Signs Dataset

10. Methodology

10.1. Methodology of the study

We want to use/follow Agile for the development of both the systems; “AI Vision Assistant for Drivers (Third Eye System)” Mobile Application. Agile project management is not a singular framework; it can be used as an umbrella term to include many different frameworks. For these systems, the SCRUM framework was used as it is most suited for software development. Scrum framework consists of meetings, roles, and tools to help teams working on complex projects collaborate, better structure and manage the workload.

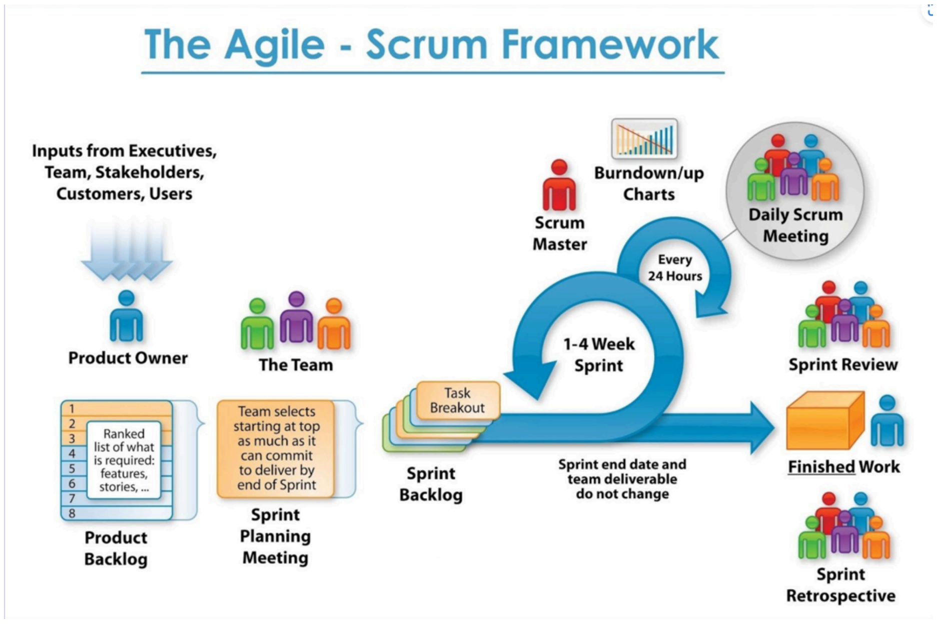

The Agile - Scrum Framework
Inputs from Executives, Team, Stakeholders, Customers, Users
Product Owner
The Team
Scrum Master
Burndown/up Charts
Daily Scrum Meeting
Every 24 Hours
1-4 Week Sprint
Sprint Review
1 2 3 4 5 6 7 8
Ranked list of what is required: features, stories, ...
Product Backlog
Team selects starting at top as much as it can commit to deliver by end of Sprint
Sprint Planning Meeting
Task Breakout
Sprint Backlog
Sprint end date and team deliverable do not change
Finished Work
Sprint Retrospective

Figure 1 Agile methodology

The stages of agile methodology that were followed are:

1. **Project initiation**

The product owner, scrum master, and development team compose the scrum team during this phase. The team held discussions to determine the project's requirements and came to a thorough knowledge of its goals, by having a meeting with Mr. Adwin, who is like a client of our project.

2. **Design**

The project's design is created based on the requirements gathered, and this includes developing a prototype as well as the architectural, user interface, and database designs.

3. **Planning and Estimate**

The team decided which user stories from the product backlog to work on during the sprint during this phase. The team decided on the sprint target, listed the tasks needed to finish the chosen backlog items, and calculated the amount of time needed for each work.

4. **Implementation**

The team began work on the tasks listed in the sprint backlog as soon as the sprint planning was finished. Also assessed simultaneously were other tasks. Daily stand-up meetings, often known as daily scrums, are quick gatherings held throughout the sprint to coordinate work, talk about progress, and identify and remove any obstacles. Each team member discussed the tasks they completed the day before, the tasks they planned to complete today, and any challenges they were currently facing.

10.2. Overall training of the model

- **Data Collection:** Gathered images from the combined datasets (Roboflow Universe Self-Driving Cars Dataset and Chinese Traffic Sign Recognition Database), as well as additional images of Bhutanese traffic signs.

- **Preprocessing:** Resized the size of all images to ensure uniform input dimensions for the model and applied data augmentation techniques (e.g., rotation, flipping, scaling) to increase the diversity of the training dataset and improve model robustness.

- **Model Training:** During YOLOv8 model training, preprocessed image data is optimized with suitable loss functions and regularization techniques. Dataset splitting into training, validation, and test sets ensures comprehensive evaluation. Hyperparameters are configured for optimal performance, aiming to enhance object detection accuracy for the AI Vision Assistant for Drivers (Third Eye System).

- **Evaluation:** During evaluation, validation metrics such as accuracy, precision, recall, F1-score, and mean Average Precision (mAP) are utilized to assess the model's performance. Hyperparameter tuning is conducted based on validation metrics to optimize performance. Additionally, the model's generalization is validated on the test set to ensure robust performance on unseen data.

- **Prediction:** For prediction, the trained YOLOv8 model is employed to process new images of traffic signs, enabling real-time detection and recognition. Subsequently, audio alerts are generated based on the detected traffic signs, providing immediate feedback to the driver to enhance situational awareness and promote safer driving practices.

- **Iteration:** During iteration, continuous improvement is pursued by collecting feedback from model performance and user interactions. The model is iterated upon by refining data collection, preprocessing steps, and training processes. Incremental improvements are implemented based on performance metrics and user feedback to enhance model accuracy and reliability over time.

11. Design

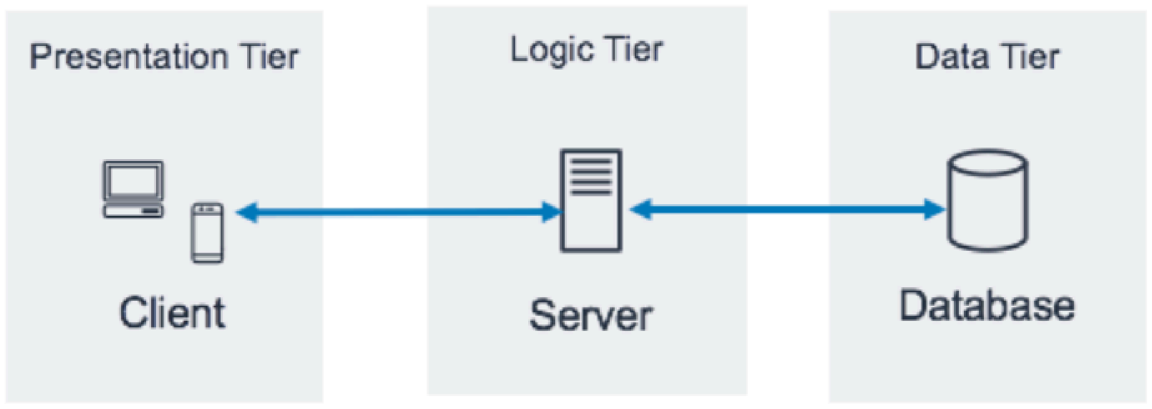


Figure 2 : 3-tier architecture

11.1. Architecture: 3-tier architecture

11.1.1. Presentation tier

The presentation tier is the user interface and communication layer of the application, where the end user interacts with the application. Its main purpose is to display information to and collect information from the user. This top-level tier can run on a web browser, as a desktop application, or a graphical user interface (GUI), for example. Web presentation tiers are usually developed using HTML, CSS and JavaScript. Desktop applications can be written in a variety of languages depending on the platform.

11.1.2. Application tier

The application tier, also known as the logic tier or middle tier, is the heart of the application. In this tier, information collected in the presentation tier is processed - sometimes against other information in the data tier - using business logic, a specific set of business rules. The application tier can also add, delete or modify data in the data tier. The application tier is typically developed using Python, Java, Perl, PHP or Ruby, and communicates with the data tier using API calls.

11.1.3. Data tier

The data tier, sometimes called database tier, data access tier or back-end, is where the information processed by the application is stored and managed. This can be a relational database management system such as PostgreSQL, MySQL,, or in a NoSQL Database server such as MongoDB.

In a three-tier application, all communication goes through the application tier. The presentation tier and the data tier cannot communicate directly with one another.

11.2. Workflow based on model training

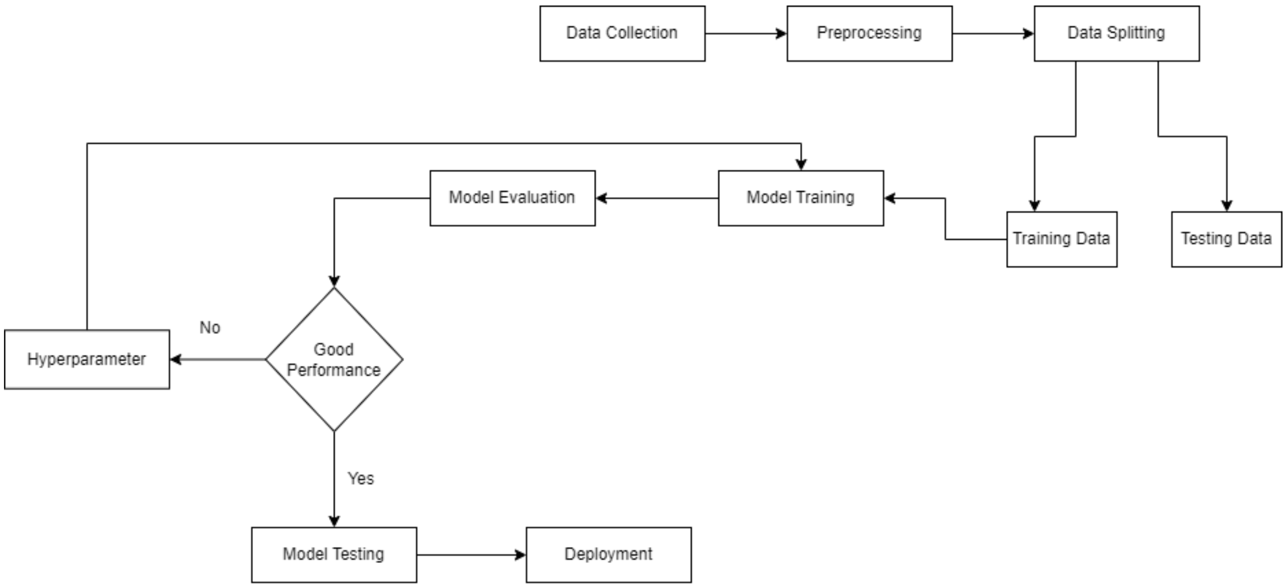


Figure 3 workflow for model training

11.3. Workflow based on user interaction

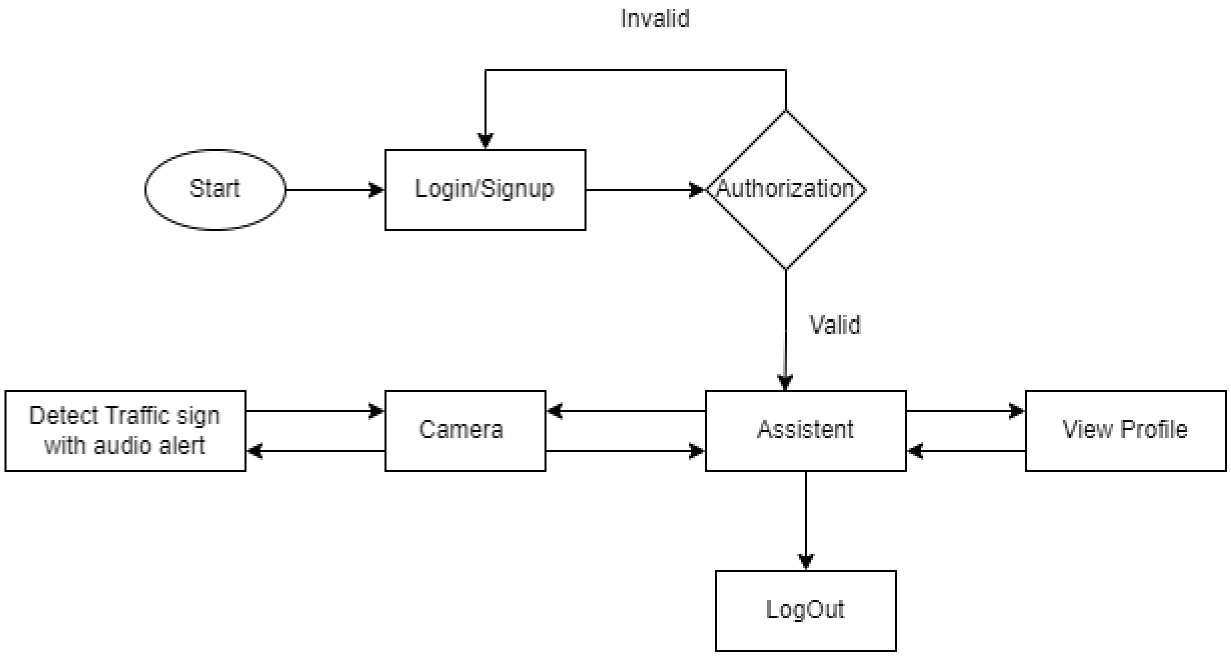


Figure 4 workflow based on user

11.4. Use Case Diagram

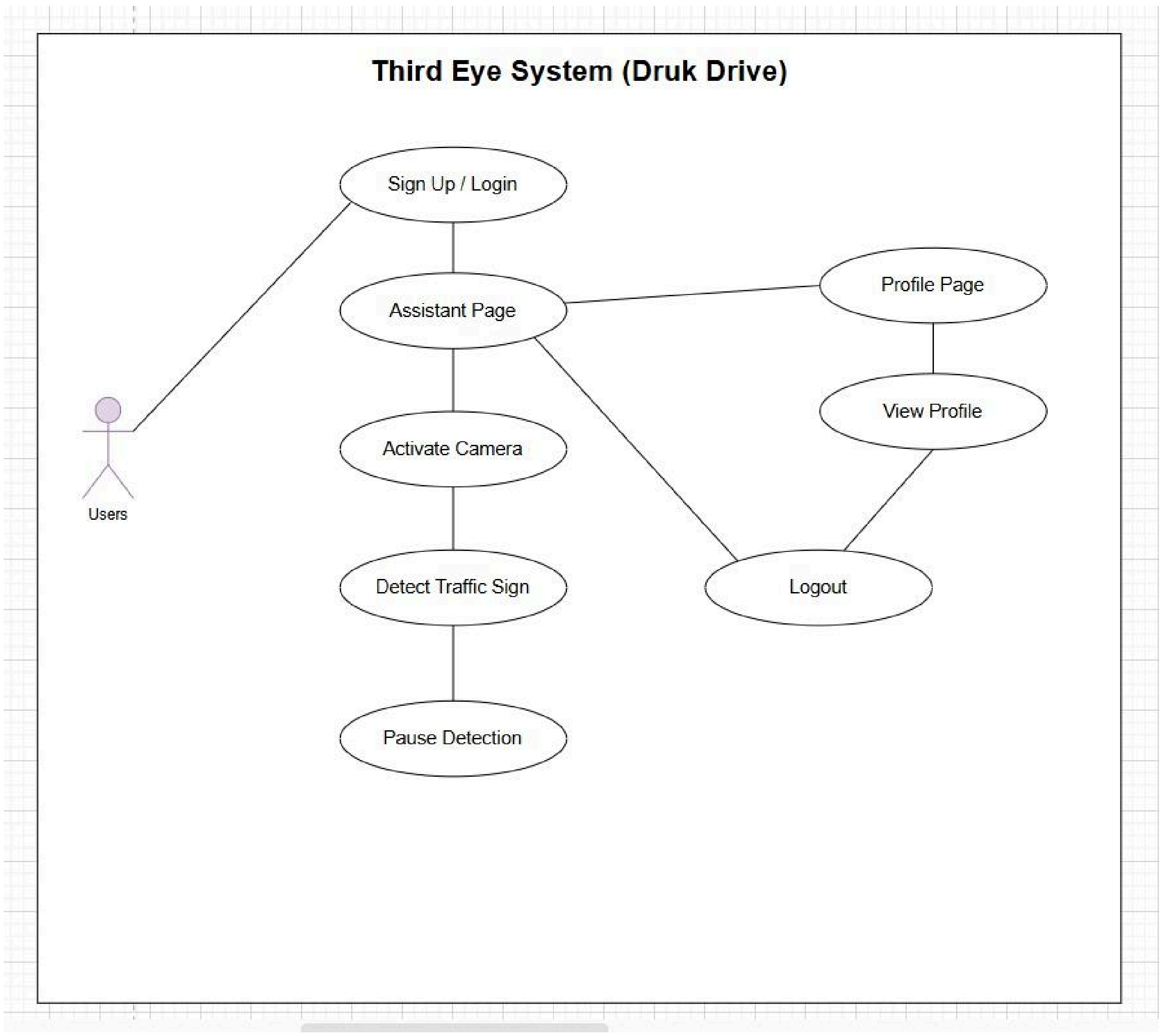


Figure 6 use case diagram

12. Literature Review

(Cherng et al., 2009) discuss the dynamic visual model (DVM), a computer model proposed to identify crucial motions of nearby vehicles while driving on a highway. The DVM comprises three analyzers: sensory analyzers, perceptual analyzers, and conceptual analyzers, inspired by the human visual system. Additionally, an episodic memory system was integrated, enabling functionalities such as adaptive response, configurability, hierarchical processing, and selective attention. Experiments demonstrating single and multiple critical movements validated the viability of the suggested system.

Dwivedi et al. (2014) explore to detect driver fatigue they developed an intelligent system based on vision(. Previous methods often relied on manually designed facial traits such as eyebrow form, yawning, blink rate, and eye closure. The suggested technique used convolutional neural network-learned features to explicitly capture complicated non-linear feature interactions and a variety of latent face traits. The motorist was classified as either sleepy or not using a softmax layer, with the system designed to alert drivers to signs of fatigue or inattention.

Omerustaoglu et al. (2020) investigate to enhance the generalization capacity of a vision-based distracted driver detection model, proposed incorporating sensor data into the model. They utilized sensor data and driver photos from actual drives to develop a new dataset, employing vision-based Convolutional Neural Network (CNN) models and Long-Short Term Memory-Recurrent Neural Network (LSTM-RNN) models to create a two-stage system to identify nine distracted behaviors. The study found that combining sensor data significantly enhanced overall performance, further boosting the accuracy of the vision-based model by optimizing the pre-trained CNN model using a relevant public dataset.

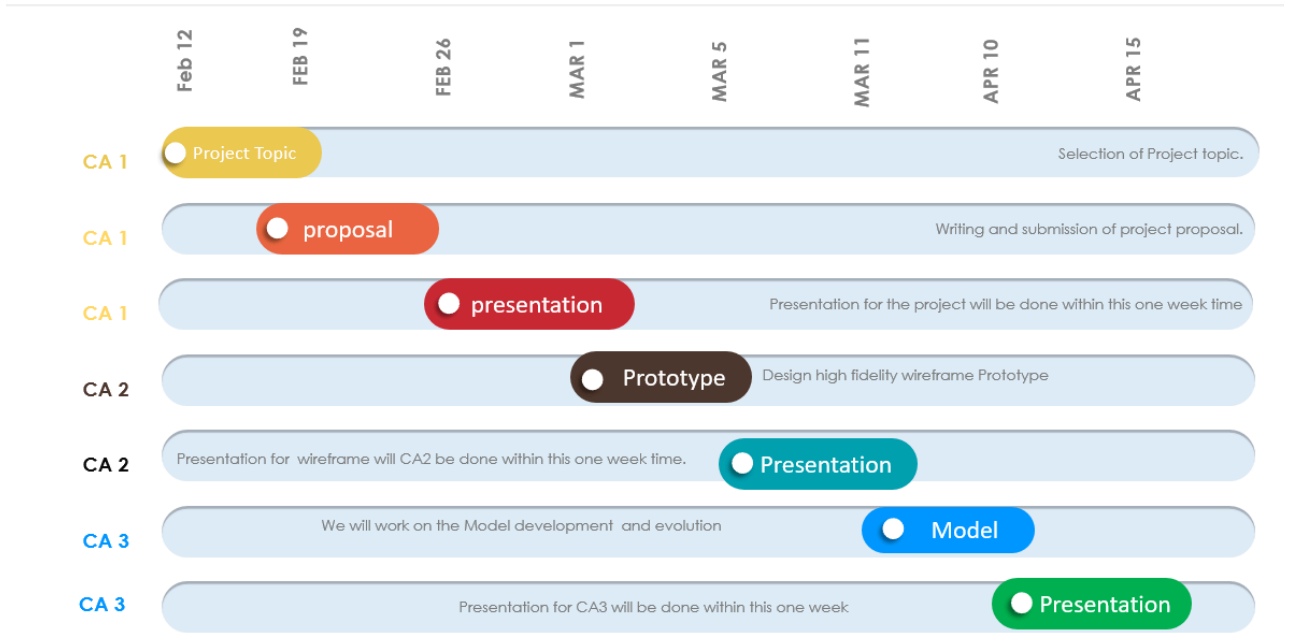
Shirpour et al. (2023) examine on identifying and detecting four categories of traffic objects: road cars, pedestrians, traffic lights, and traffic signs, using a vision-based frame-work. Their method identified and detected these objects both inside and outside of the driver's attentional visual field, utilizing the driver's 3D absolute gaze point coordinates obtained using a non-contact 3D gaze tracker and a front-view stereo imaging system. They achieved encouraging results, with a 91 percent correct object detection rate using a combination of models based on Faster R-CNN and multi-scale HOG-SVM, verified using a ResNet-101 network on real data captured during drives in an urban setting.

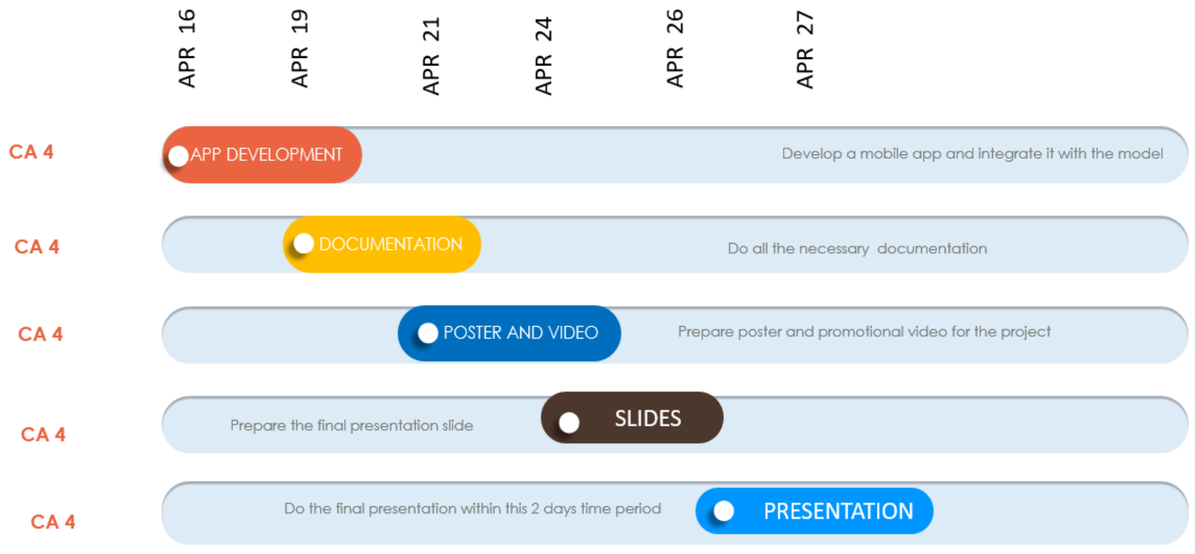
Liu et al. (2016) focus on semi-supervised techniques for driver distraction detection in actual driving situations to reduce the expense of labeling training data. They utilized eye and head movements to assess two driving states—alert and cognitively distracted—employing the Laplacian support vector machine and the semi-supervised extreme learning machine. The semi-supervised learning methods increased detection performance (G-mean) by an average of

0.0245 across all participants with the additional unlabeled data, showing potential for improving model building efficiency in terms of time and cost.

Rahman et al. (2021) classify a vision-based approach to automatically extract features using deep learning architectures and manually extract features based on domain expertise from a driver's eye movement signals to categorize a driver's cognitive burden. Three deep learning architectures and five machine learning models were created, with the support vector machine model with linear kernel function and the convolutional neural network model achieving maximum classification accuracies of 92 percent and 91 percent, respectively, highlighting the potential of this non-contact technology in the development of sophisticated driver assistance systems.

13. Project Milestone





CA 4 APP DEVELOPMENT Develop a mobile app and integrate it with the model

CA 4 DOCUMENTATION Do all the necessary documentation

CA 4 POSTER AND VIDEO Prepare poster and promotional video for the project

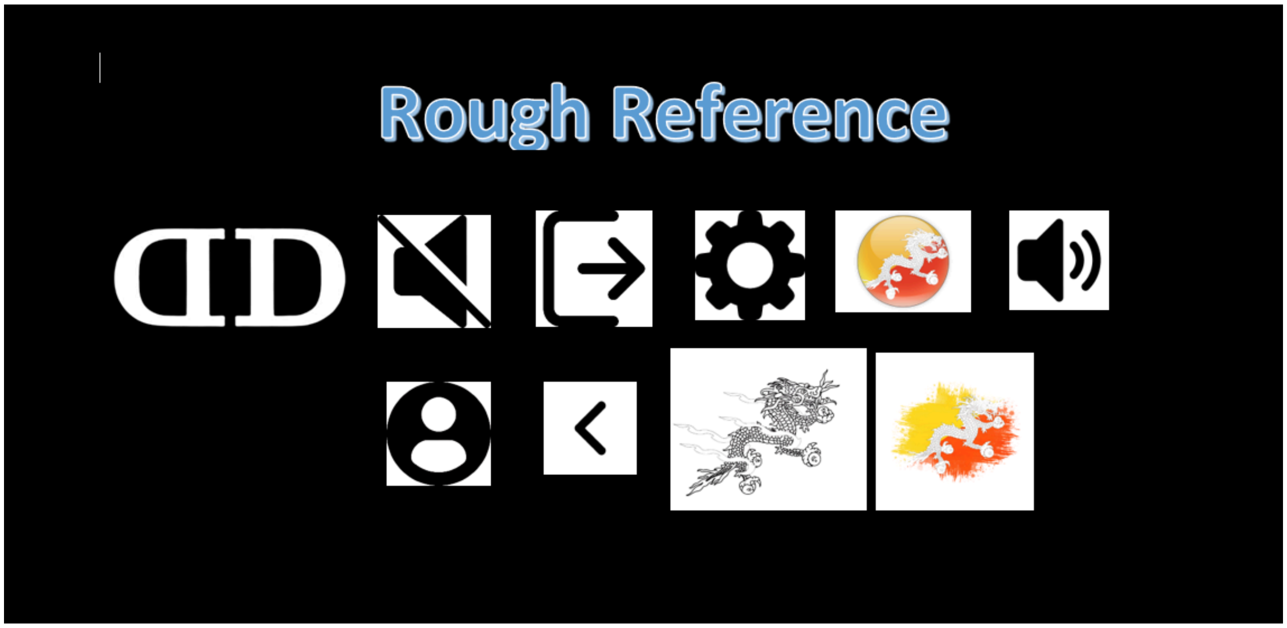
CA 4 Prepare the final presentation slide SLIDES

CA 4 Do the final presentation within this 2 days time period PRESENTATION

14. Prototype

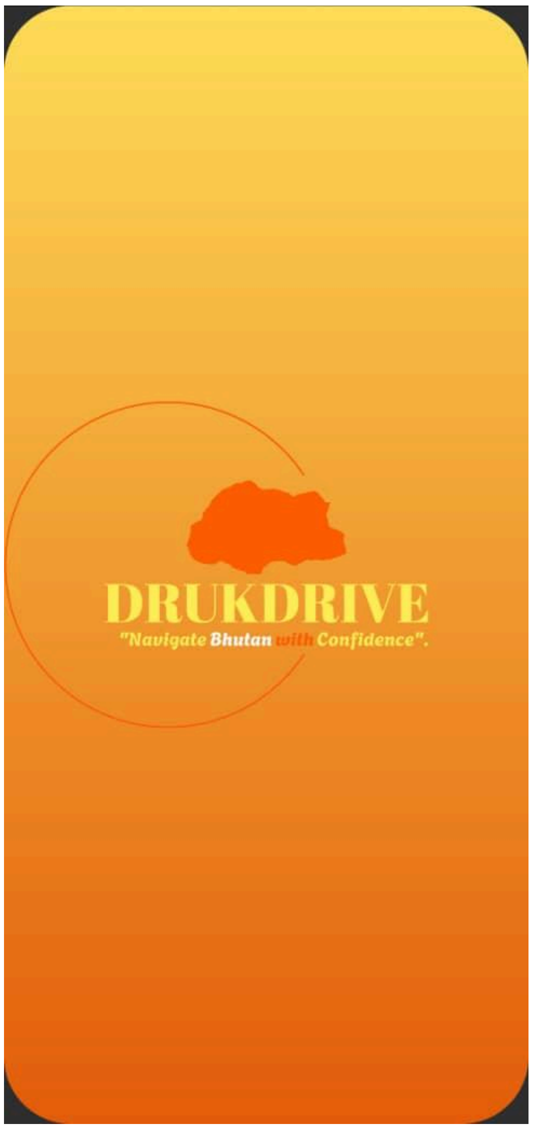
14.1 Rough Reference

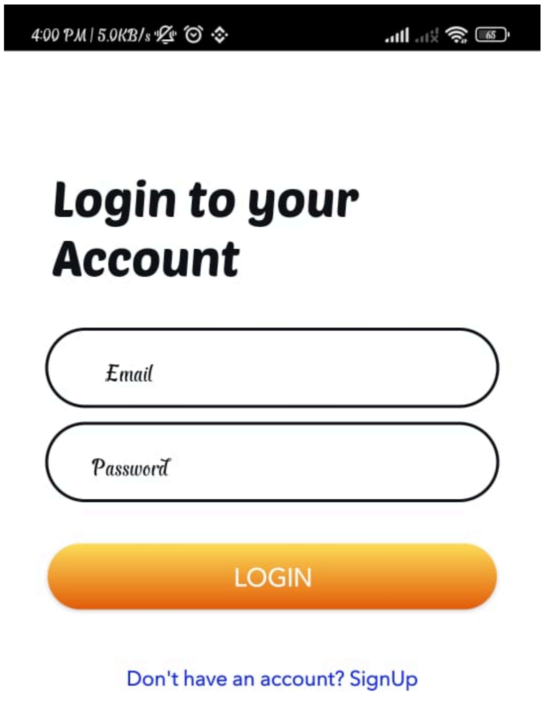
Rough reference of the pictures, icons, logo and colors used in the application.



14.2 User Interface

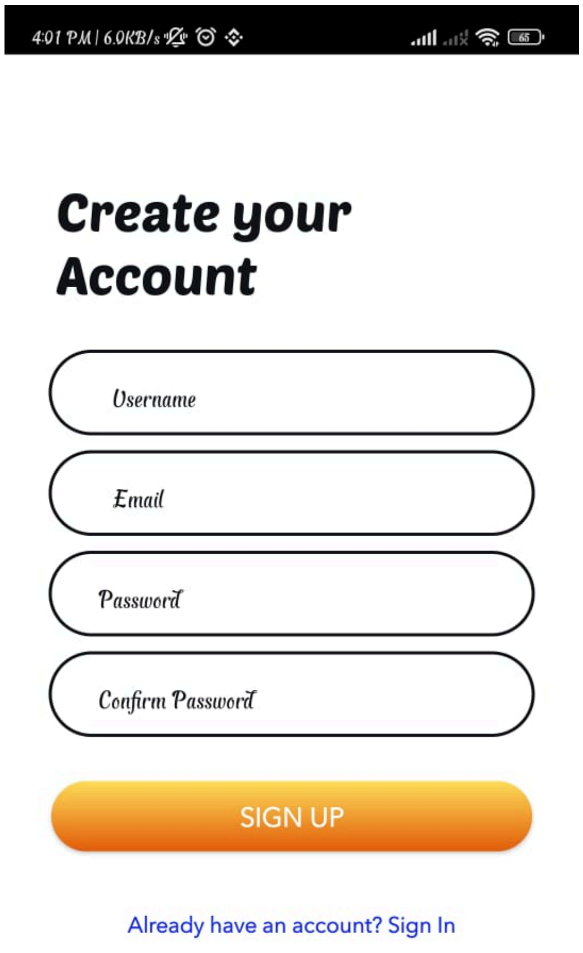
Upon accessing the application, users are greeted with the logo, creating a visually engaging introduction. Subsequently, they seamlessly transition to the landing page, where they can explore the features and functionalities of the AI Vision Assistant for Drivers (Third Eye System).

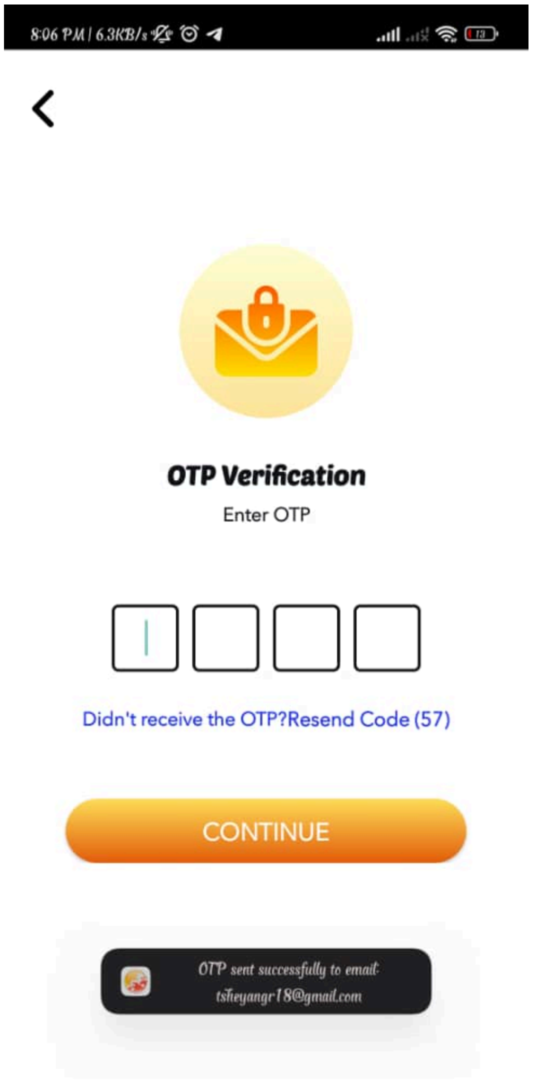




After encountering the logo and landing page, users are redirected to the login page, where they can securely access their accounts if they already have one. This streamlined process ensures seamless navigation and enhances user experience within the AI Vision Assistant for Drivers (Third Eye System).

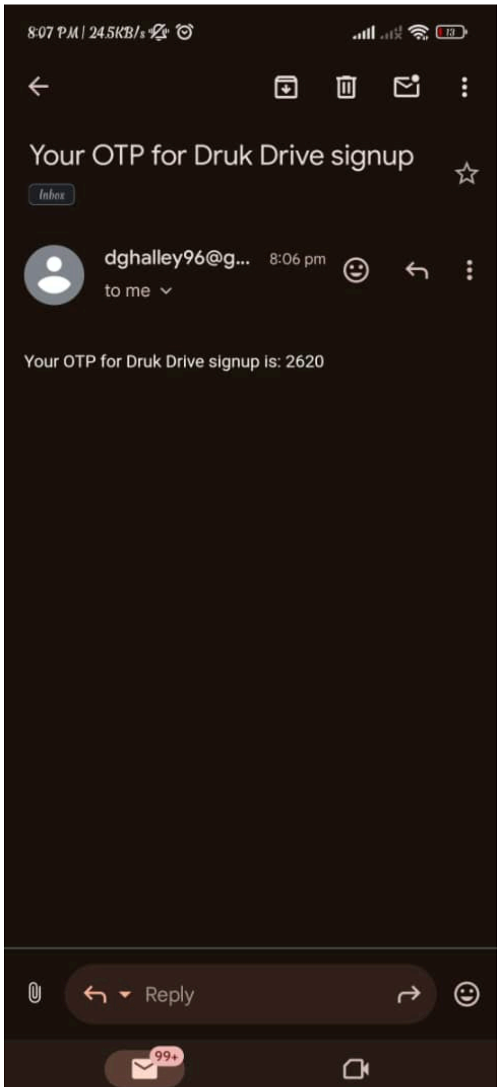
If users do not have an account, they are provided with the option to create one by clicking on the signup button. This intuitive design allows new users to easily register and access the full features of the AI Vision Assistant for Drivers (Third Eye System) without any hassle.





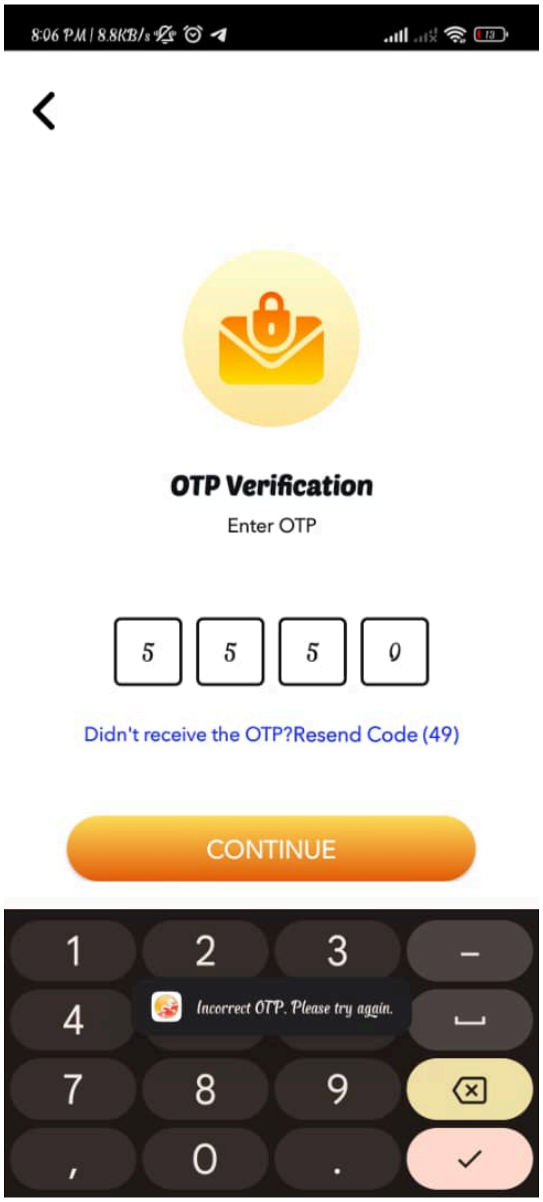
OTP sent successfully to email: tsheyangr18@gmail.com

Upon signing up an OTP (One-Time Password) is sent to the provided email address for verification purposes. This added layer of security ensures that the account creation process is both seamless and secure, enhancing user trust and confidence in the AI Vision Assistant for Drivers (Third Eye System).



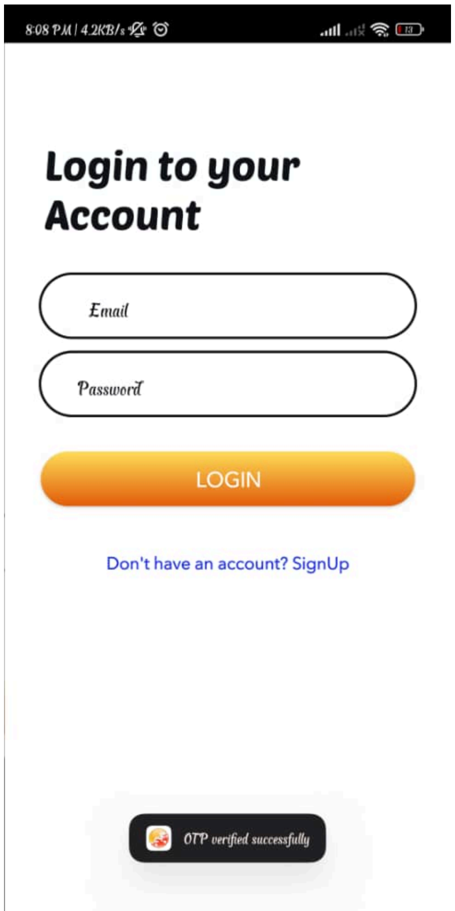
Your OTP for Druk Drive signup

Your OTP for Druk Drive signup is: 2620



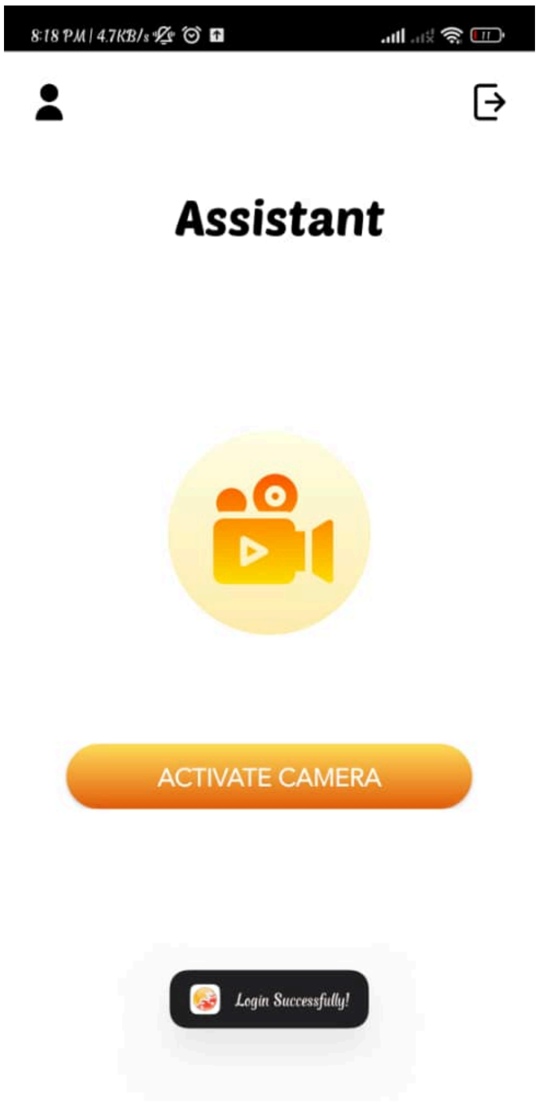
Incorrect OTP. Please try again.

If the entered OTP does not match the one sent to the user's email, an incorrect message prompt is displayed.

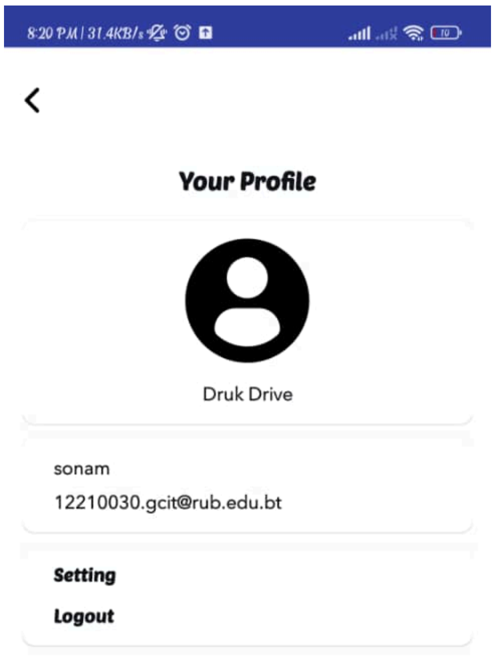


OTP verified successfully

After successfully verifying the OTP, users are directed to the assistant page, where they can activate the camera, view their profile, and log out, providing a seamless and convenient user experience within the AI Vision Assistant for Drivers (Third Eye System).

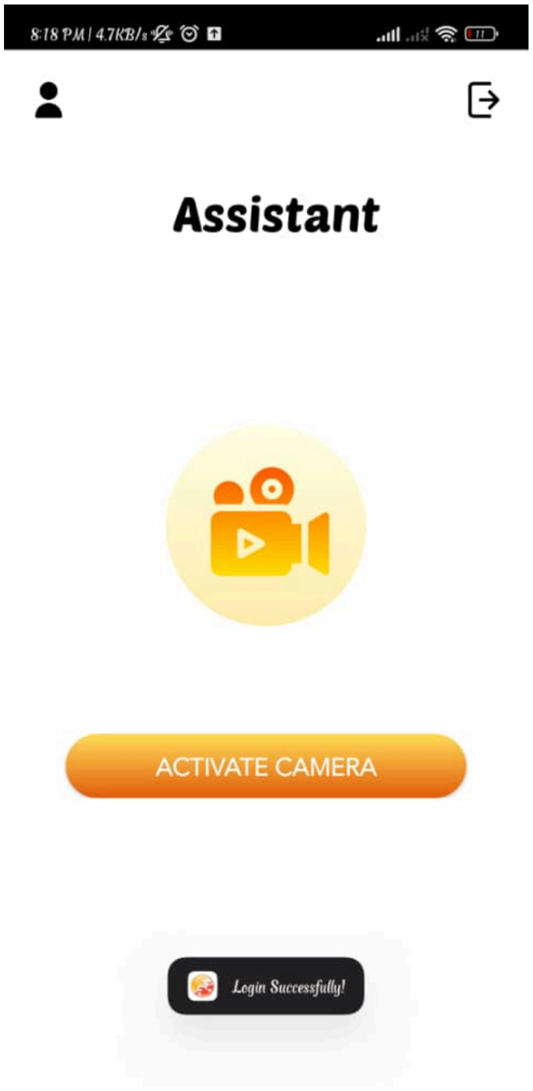


When the user activates the camera, the AI Vision Assistant for Drivers (Third Eye System) detects traffic signs in real-time and provides corresponding audio alerts for each sign, enhancing driver awareness and safety on the road.

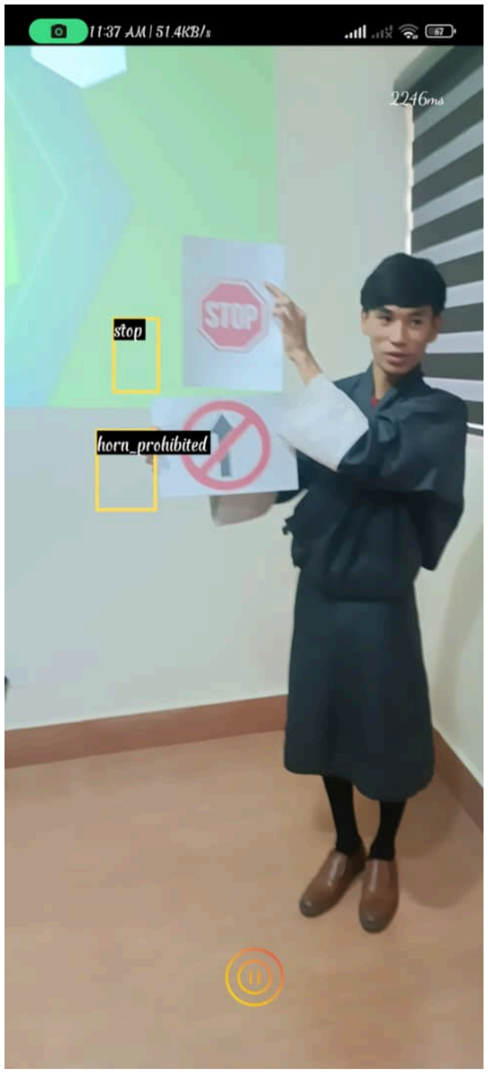


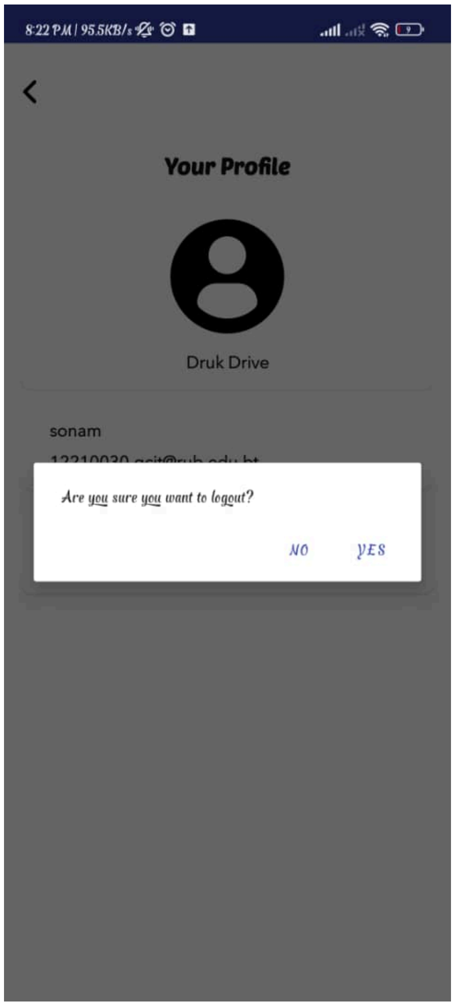
sonam
12210030.gcit@rub.edu.bt

Upon clicking on the profile, users can view their username and email, providing them with access to their account information and enhancing personalization within the AI Vision Assistant for Drivers (Third Eye System).



When the authenticated user activates the camera, the AI Vision Assistant for Drivers (Third Eye System) detects traffic signs in real-time and provides corresponding audio alerts for each sign, enhancing driver awareness and safety on the road.

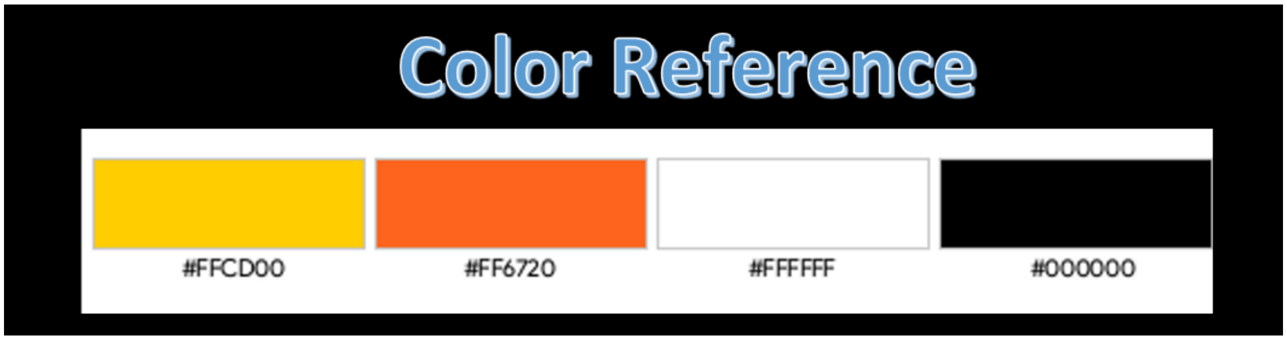




When users click on the logout button, they are prompted to confirm their action. Upon confirmation, users can successfully log out from their account, ensuring a secure and seamless logout process within the AI Vision Assistant for Drivers (Third Eye System).

14.3 Color Reference

The color reference used in the prototype and will be employed in the application development is crucial for maintaining consistency and visual coherence throughout the



15. Model Trained

15.1 Data preprocessing for YOLO

For the YOLO (You Only Look Once) model, the data preprocessing was meticulously handled using RoboFlow, a powerful tool for annotating and augmenting datasets. RoboFlow facilitated the annotation process by allowing precise bounding box labeling for each object within the images, ensuring that the model could learn to detect and classify objects accurately. Additionally, a series of data augmentation techniques were applied to the dataset to enhance its diversity and robustness. These augmentations included operations such as random cropping, horizontal flipping, scaling, and color adjustments. By incorporating these augmentations, the dataset became more varied, which helps in reducing overfitting and improving the model's ability to generalize to new, unseen data. This comprehensive preprocessing step in RoboFlow was crucial in preparing a high-quality dataset that could effectively train the YOLO model, leading to better performance in object detection tasks.

A. YOLOV9

- **Architecture Overview:**

YOLOv9 continues the evolution of the YOLO series, incorporating cutting-edge techniques for enhanced accuracy and speed in object detection.

a. **Input:** Flexible input sizes, typically resized to a standard dimension for processing, allowing the model to handle various image resolutions effectively.

b. **Backbone**: Utilizes an enhanced backbone with deeper layers and more complex structures for improved feature extraction, allowing the model to capture more detailed and abstract features from the input images. This deeper network structure enables better representation and understanding of the image content.

c. **Neck:** Employs advanced feature pyramid networks (FPN) for better multi-scale feature representation, enabling the model to detect objects of varying sizes more effectively. The FPN enhances the model's ability to integrate features from different layers, improving its sensitivity to objects at multiple scales.

d. **Head:** Features refined detection heads with improved bounding box regression and classification capabilities, allowing for more accurate and reliable object detection. These refined heads ensure precise localization of objects and robust classification, even in complex scenes.

e. **Anchor Boxes**: Uses dynamic anchor boxes that adapt based on the dataset to improve detection accuracy, allowing the model to better fit the shapes and sizes of objects in the training data. This adaptability ensures that the anchor boxes are more representative of the objects, leading to higher detection performance.

f. **Classification and Localization:** Integrates advanced techniques for accurate object classification and precise localization, combining these tasks to provide detailed and reliable predictions. This integrated approach ensures that both the category and position of objects are accurately predicted.

- **Preprocessing:**

  a. Data Preparation:

  - We used data from Roboflow, where we performed preprocessing, annotation, and augmentation. Images were resized and augmented using techniques like flipping, scaling, and color adjustments to enhance the robustness and variability of the training data. Roboflow provided a streamlined process for annotating the data, ensuring high-quality labels, and applying consistent augmentation techniques.

b. **Normalization:**

   - Pixel values were normalized to the range [0, 1] to ensure consistent input scaling. This normalization process helps stabilize the training process and improve convergence.

c. **Training Results:**

   - **Training Accuracy:**

     - Achieved 95.2% mean Average Precision (mAP) on the training dataset, reflecting the high accuracy and robustness of the model. This high mAP indicates that the model has effectively learned to recognize and localize objects within the training data.

   - **Testing Accuracy:**

     - Recorded 80.7% mAP on the testing dataset, reflecting state-of-the-art performance in object detection tasks and demonstrating the model's ability to generalize well to new and unseen data. This robust testing performance confirms the model's effectiveness in real-world scenarios, ensuring reliable object detection across various applications.

B. YOLOv8

- **Architecture Overview:**

  YOLOv8 (You Only Look Once) is an advanced object detection model designed for real-time performance, building upon the strengths of its predecessors with improved accuracy and speed.

  a. **Input:** Takes images of various sizes, often resized to a standard dimension (e.g., 416x416 pixels). This flexibility in input size allows the model to adapt to different image resolutions while maintaining performance.

  b. **Backbone:** Uses a convolutional neural network backbone for feature extraction, often based on architectures like CSPDarknet. This backbone is designed to capture rich and hierarchical features from the input images, crucial for accurate object detection.

c. **Neck:** Incorporates a path aggregation network (PANet) to enhance feature fusion and pyramid levels for multi-scale detection. The PANet structure allows the model to combine features from different layers, improving its ability to detect objects at various scales.

d. **Head**: Consists of multiple detection heads, each responsible for predicting bounding boxes, objectness scores, and class probabilities at different scales. These detection heads ensure that objects of different sizes and shapes can be accurately localized and classified.

e. **Anchor Boxes:** Uses predefined anchor boxes to predict bounding boxes for detected objects. These anchor boxes are strategically sized and shaped to match the objects in the dataset, aiding in precise localization.

f. **Classification and Localization:** Combines classification and bounding box regression to identify and localize objects in images. This integrated approach ensures that both the category and position of objects are accurately determined.

- **Preprocessing:**
  - **a. Data Preparation:**

    - We used data from Roboflow, where we performed preprocessing, annotation, and augmentation. Images were resized and augmented using techniques like flipping, scaling, and color adjustments to enhance the robustness and variability of the training data. Roboflow provided a streamlined process for annotating the data, ensuring high-quality labels, and applying consistent augmentation techniques.

  - **b. Normalization:**

- Pixel values were normalized to the range [0, 1] to ensure consistent input scaling. This normalization process is essential for stabilizing the training process and ensuring the model converges efficiently.

c. **Training Results:**

- **Training Accuracy:**

  - Achieved 95.8% mean Average Precision (mAP) on the training dataset, indicating the model's high accuracy and ability to learn from the training data effectively. This high mAP shows that the model can accurately detect and classify objects in the training images.

- **Testing Accuracy:**

  - Recorded 85% mAP on the testing dataset, showcasing excellent detection performance on unseen data. This robust testing performance demonstrates the model's ability to generalize well, ensuring reliable object detection in real-world scenarios and across various applications.

Sure, let's break down the architecture, preprocessing steps, and training results for MobileNetV1, MobileNetV2, and MobileNetV3:

15.2. Model Training for MobileNet

A. MobileNetV1

- **Architecture Overview:**

  a. **Input:** Images are fed into the MobileNetV1 model, each representing a single frame containing spatial information.

  b. **Feature Extraction:** Utilizes depthwise separable convolutions to reduce computational complexity while capturing essential image features.

c. **Classification:** Employs global average pooling followed by a fully connected layer for classification. MobileNetV1 focuses on efficiently extracting features from input images and making accurate predictions.

- **Preprocessing:**

  - Images are resized to the required input size (e.g., 224x224 pixels) and normalized to the range [0, 1].

  - Minimal data augmentation techniques are applied, such as horizontal flipping and random cropping, to enhance model robustness.

- **Training Results:**
  - **Training accuracy:** Achieved a training accuracy of 85.3%, demonstrating the model's ability to learn from the provided training data effectively.
  - **Testing accuracy:** Achieved 80.1%, indicating strong generalization performance on unseen data.

B. MobileNetV2

- **Architecture Overview:**

  a. **Input:** Images are fed into the MobileNetV2 model, each representing a single frame containing spatial information.

  b. **Feature Extraction**: Introduces inverted residual blocks with linear bottlenecks to improve feature representation and efficiency. MobileNetV2 enhances feature extraction capabilities while maintaining computational efficiency.

c. **Classification:** Utilizes a similar classification head as MobileNetV1, consisting of global average pooling followed by fully connected layers.

- **Preprocessing:**
  - Follows similar preprocessing steps as MobileNetV1, including resizing and normalization and data augmentation techniques are applied to further enhance model robustness.

- **Training Results:**
  - **Training accuracy:** Attains a training accuracy of 89.7%, showcasing the model's effectiveness in learning intricate features from the training data.
  - **Testing accuracy:** Achieved 85.2%, indicating strong generalization performance on unseen data.

C. MobileNetV3:

- **Architecture Overview:**

  a. **Input:** Images are fed into the MobileNetV1 model, each representing a single frame containing spatial information.

  b. **Feature Extraction:** Introduces efficient attention mechanisms and improved activation functions to enhance feature representation and model efficiency further.

  c. **Classification:** Retains the global average pooling and fully connected layers for classification, focusing on accurate prediction of object categories.

- **Preprocessing:**
  - Adheres to similar preprocessing steps as MobileNetV1 and MobileNetV2, ensuring consistency in input data preparation.

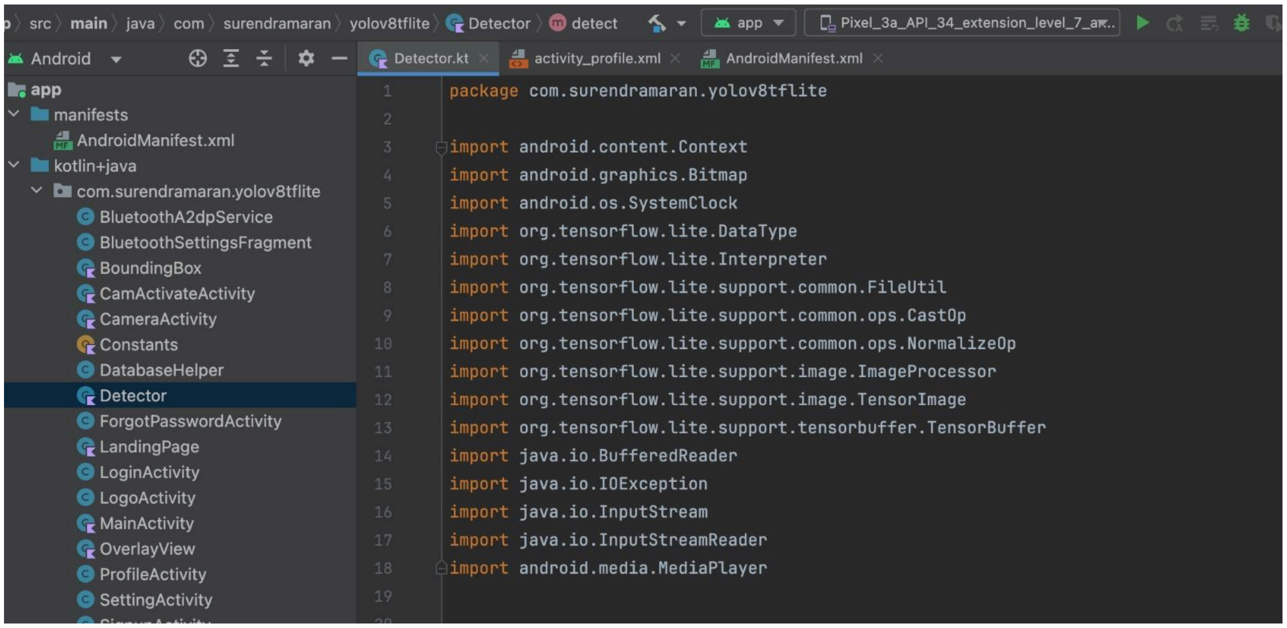
- **Training Results:**

- **Training Accuracy:** Achieves a training accuracy of 92.1%, demonstrating the model's capability to effectively capture intricate patterns in the training data.
- **Testing accuracy:** Achieved 88.5%, indicating robust generalization performance on unseen data.

After thorough evaluation, we determined that the YOLOv8 model exhibited promising performance on our dataset. To further optimize its capabilities, we conducted hyperparameter tuning, a critical step in refining model accuracy and reliability. Initially, we tested different optimizers, including SGD, RMSProp, and AdamW, which yielded accuracies of 92.1%, 92.1%, and 93.9%, respectively. Subsequently, we explored various image sizes, ranging from 640 to 800, with 769 producing the highest accuracy of 94.1%. Additionally, we experimented with multiple dropout values, finding that a dropout of 0 resulted in the highest accuracy of 94.1%. Before hyperparameter tuning, the model achieved a training accuracy of 95.8% with a testing accuracy of 85%. After hyperparameter tuning, while the training accuracy remained unchanged, the testing accuracy significantly improved to 93.8%. These optimizations reflect our dedication to enhancing the AI Vision Assistant for Drivers (Third Eye System), ensuring it delivers reliable and accurate traffic sign detection to enhance driver safety on the road.

17. Model Integration

For model integration, we utilized an APK file (Android Package Kit) and implemented TensorFlow Lite, ensuring efficient deployment of the YOLOv8 model on mobile devices. This approach leverages the lightweight and optimized nature of TensorFlow Lite, facilitating real-time traffic sign detection and audio alerts within the AI Vision Assistant for Drivers (Third Eye System). The APK enables easy deployment and installation of the application on Android devices.



```
package com.surendramaran.yolov8tflite

import android.content.Context
import android.graphics.Bitmap
import android.os.SystemClock
import org.tensorflow.lite.DataType
import org.tensorflow.lite.Interpreter
import org.tensorflow.lite.support.common.FileUtil
import org.tensorflow.lite.support.common.ops.CastOp
import org.tensorflow.lite.support.common.ops.NormalizeOp
import org.tensorflow.lite.support.image.ImageProcessor
import org.tensorflow.lite.support.image.TensorImage
import org.tensorflow.lite.support.tensorbuffer.TensorBuffer
import java.io.BufferedReader
import java.io.IOException
import java.io.InputStream
import java.io.InputStreamReader
import android.media.MediaPlayer
```

GitLab Link: https://gitlab.com/prj3031/drukdrive

18. Challenges and Limitations

The overview of the challenges and limitations faced in developing a traffic sign detection system for a mobile app, covering both data-related issues and performance constraints as shown below:

- **Datasets:**
  - **Image Collection:** A substantial number of images of different traffic signs under various conditions were required to train the model effectively.
  - **Labelling:** Proper labelling of each image with the correct traffic sign was a time-consuming process, ensuring accurate training data for the model.
- **Real-Time Performance:**
  - **Efficient Detection:** Ensuring the app could detect traffic signs quickly and accurately without slowing down posed a significant challenge.
  - **Limited Processing Power:** The constrained processing power of mobile devices compounded the difficulty, making it challenging to run complex models smoothly.

18. Conclusion

In conclusion, the development of DrukDrive which is AI Vision Assistant for Drivers also known as Third Eye System to detect traffic sign detection requires addressing key challenges such as dataset acquisition and accurate labeling. Gathering a diverse and extensive dataset is crucial for training the system effectively, while meticulous labeling ensures the model learns to recognize traffic signs accurately. Moreover, achieving real-time performance on mobile devices presents another hurdle, given their limited processing power. Balancing the need for accuracy with the demand for efficiency is essential in ensuring the system can detect signs quickly and reliably.

However, despite these challenges, advancements in machine learning and optimization techniques offer promising avenues for improvement. Techniques such as data augmentation and model compression can enhance the efficiency of the system, making it more suitable for deployment on mobile platforms. By leveraging these advancements, developers can overcome the hurdles posed by dataset acquisition and real-time performance, ultimately leading to the creation of more effective and reliable traffic sign detection systems.

20. References

Cherng, S., Fang, C., Chen, C., & Chen, S. (2009). Critical motion detection of nearby moving vehicles in a vision-based driver-assistance system. IEEE Transactions on Intelligent Transportation Systems, 10(1), 70–82. https://doi.org/10.1109/tits.2008.2011694

Dwivedi, K., Biswaranjan, K., & Sethi, A. (2014). Drowsy driver detection using representation learning. Institute of Electrical and Electronics Engineers. https://doi.org/10.1109/iadcc.2014.6779459

Omerustaoglu, F., Şakar, C. O., & Kar, G. (2020). Distracted driver detection by combining in-vehicle and image data using deep learning. Applied Soft Computing, 96, 106657. https://doi.org/10.1016/j.asoc.2020.106657

Shirpour, M., Khairdoost, N., Bauer, M., & Beauchemin, S. S. (2023). Traffic object detection and recognition based on the attentional visual field of drivers. IEEE Transactions on Intelligent Vehicles, 8(1), 594–604. https://doi.org/10.1109/tiv.2021.3133849

Liu, T., Yan, Y., Huang, G., Yeo, Y. K., & Lin, Z. (2016). Driver distraction detection using semi-supervised machine learning. IEEE Transactions on Intelligent Transportation Systems, 17(4), 1108–1120. https://doi.org/10.1109/tits.2015.2496157

Rahman, H., Ahmed, M. U., Barua, S., Funk, P., & Begum, S. (2021). Vision-based driver's cognitive load classification considering eye movement using machine learning and deep learning. Sensors, 21(23), 8019. https://doi.org/10.3390/s21238019

THANK YOU!