Software Design Specification

Project Title :

Environment Detection System for Blind Users

Project Code :

EDBP-2025

Internal Advisor :

Prof. Muhammad Fahad

Project Manager :

Dr. Muhammad Ilyas

Project Team :

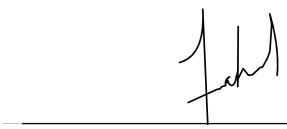
Ali Raza Ansari
BSCS51F22S008 (Team Lead)

Muhammad Abdullah Zammad
BSCS51F22S036

Muhammad Hassan Javed
BSCS51F22S040

Submission Date :

December 15, 2025



Project Manager's Signature

Document Information

Category	Information
Customer	Computer Science, University of Sargodha
Project	Environment Detection For Blind Person
Document	Software Design Specification
Document Version	1.0
Identifier	PGBHO1-2025-DS
Status	Draft
Arthor(s)	Ali Raza Ansari M. Abdullah Zammad M.Hassan Javed
Approver(s)	PM
Issue Date	Dec 17, 2025
Distribution	1. Advisor 2. PM 3. Project Office

Definition of Terms, Acronyms and Abbreviations

Terms	Description
UCD	Use Case diagrame
TTS	TextToSpeech
GUI	Graphical user interface

Table of Contents

1. Introduction

1.1 Purpose of the Document

The Software Design Specification (SDS) document provides a comprehensive description of the architectural, structural, and behavioral design of the Environment Detection System for Blind Users. It presents an overview of the system, its scope, and a detailed explanation of the system design, functionality, and operational workflow.

This document is intended for project stakeholders, software developers, system architects testers, the project advisor, and the project management office. It serves as a reference throughout the development, testing, and deployment phases to ensure that the final product meets its intended objectives. The system design follows the Object-Oriented Design (OOD) methodology.

1.2 Project Overview

The Environment Detection System for Blind Users is an AI-powered application designed to assist visually impaired individuals by detecting objects using a smartphone camera and providing real-time audio feedback. The system processes video frames in real time to identify obstacles in the user's surroundings and communicates the detected information through spoken output using a Text-to-Speech (TTS) engine. This enables users to navigate their environment safely and with increased confidence.

1.3 Scope

Included Features

- Real-time object detection using a smartphone camera
- Image preprocessing techniques such as resizing, normalization, and noise reduction
- Audio feedback generation using an offline Text-to-Speech (TTS) engine
- Direction identification of detected objects (Left, Center, Right)
- Continuous real-time system operation
- A simple, accessible, and user-friendly mobile interface.

Excluded Features

- Distance estimation between the user and detected objects
- Object size measurement
- Navigation or path planning capabilities
- Integration with IoT devices or external hardware
- Facial recognition or optical character recognition (OCR) features.

Design Considerartion

2.1 Assumptions and Dependencies

Assumptions

- The system will be used in real-time environments with adequate lighting conditions suitable for object detection.
- Users will possess Android smartphones equipped with a functional camera and a minimum of 4 GB RAM.
- A lightweight YOLO-based object detection model will be used and will be compatible with supported devices.
- Internet connectivity will be required only for the initial model download and for periodic system updates.
- Hardware components such as the camera and earphones or speakers will function correctly and meet minimum performance requirements.
- Cloud services or local storage used for updates and data logs will remain accessible without significant interruptions.
- Users will operate the system under normal environmental conditions and adhere to basic usage guidelines.

Dependencies

- External components, including Text-to-Speech (TTS) engines and computer vision models, may affect system performance if their updates, configurations, or licensing terms change.
- Successful system operation depends on proper integration with compatible smartphone hardware.
- A stable network connection is required for online features, system updates, and data synchronization.
- Development progress relies on the availability of trained AI models and high-quality datasets to ensure accurate object recognition.
- The system depends on third-party frameworks and APIs, such as TensorFlow and OpenCV, for object detection and system functionality.

2.2 Risks and Volatile Areas

- Object detection accuracy may vary under different lighting and environmental conditions.
- Application performance is highly dependent on the processing power and memory of the user's smartphone.
- Real-time detection performance may degrade under heavy computational load or multitasking.
- Updates to the object detection model may require retraining or re-optimization.

3. System Architecture

This section presents a high-level architectural overview of the Environment Detection System for Blind Users. The system architecture defines how overall functionality is divided into logical components and how these components interact to provide real-time environmental awareness to visually impaired users.

The proposed system adopts a modular and layered architecture, in which each component is responsible for a specific function, such as image acquisition, object detection, decision processing, and audio feedback. This design approach enhances maintainability, scalability, and performance while ensuring the system remains lightweight and suitable for smartphone deployment.

The system primarily utilizes the smartphone's built-in hardware components, including the camera, processor, and speaker or headphones, along with software components such as computer vision models and text-to-speech engines. No external IoT sensors or dedicated hardware devices are required, ensuring portability and cost-effectiveness.

At a high level, the system operates by capturing real-time video input from the smartphone camera, processing video frames through an object detection model, interpreting detected objects, and delivering relevant information to the user via audio output. All processing is optimized to minimize latency, as timely feedback is critical for the safety and confidence of visually impaired users.

3.1 System-Level Architecture

The system-level architecture decomposes the Environment Detection System into the following major subsystems:

Image Acquisition Subsystem

This subsystem is responsible for capturing real-time visual data from the smartphone camera. Continuous video input is obtained and divided into individual frames. These serve as the raw input for further processing.

Responsibilities:

- Access smartphone camera hardware
- Capture live video stream
- Extract frames at an appropriate frame rate.

Pre-Processing Subsystem

Before object detection, the captured frames are pre-processed to improve detection efficiency and reduce computational overhead.

- Resize frames to model-compatible dimensions
- Convert RGB images to grayscale or optimized formats
- Normalize noise and improve contrast if required

Object Detection Subsystem

This is the core component of the system. It uses a custom-trained deep learning model to detect and classify objects present in the environment in real time.

Responsibilities:

- Apply YOLOv3 object detection model
- Identify objects such as people, vehicles, furniture, doors, etc.
- Generate confidence scores and bounding boxes
- Classify detected objects using trained datasets

Decision & Interpretation Subsystem

This subsystem interprets detection results and decides which information should be conveyed to the user.It filters detections based on relevance and confidence to avoid overwhelming the user with excessive audio feedback.

Responsibilities:

- Filter low-confidence detections
- Prioritize critical obstacles
- Convert detection results into meaningful textual descriptions

Audio Feedback Subsystem

This system converts textual information into audible instructions using a Text-to-Speech (TTS) engine, enabling blind users to receive environmental awareness through sound.

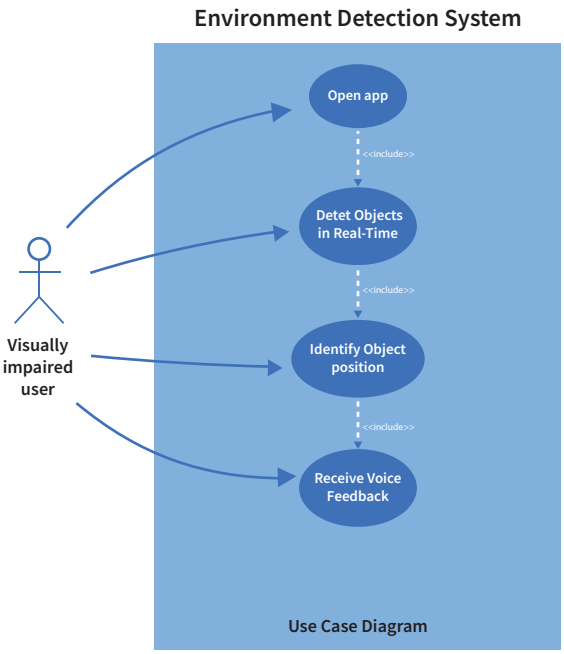
Responsibilities:

- Convert detected object labels into speech
- Deliver audio output via smartphone speaker or headphone
- Ensure clarity, minimal delay, and user friendly speech output

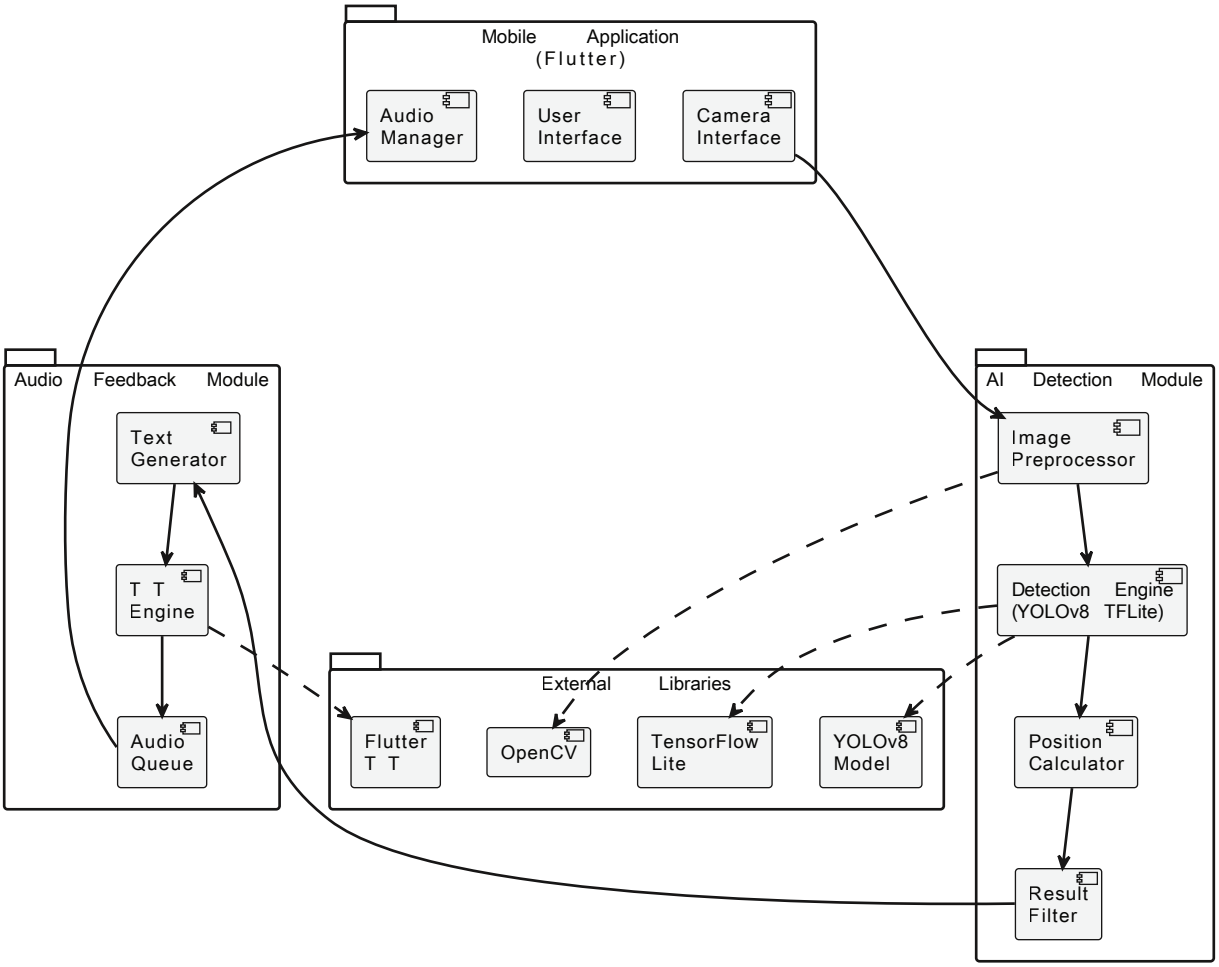
User Interaction Subsystem

This System allows the user ti interact between system and application, ensuring ease of use for visually impaired individuals.

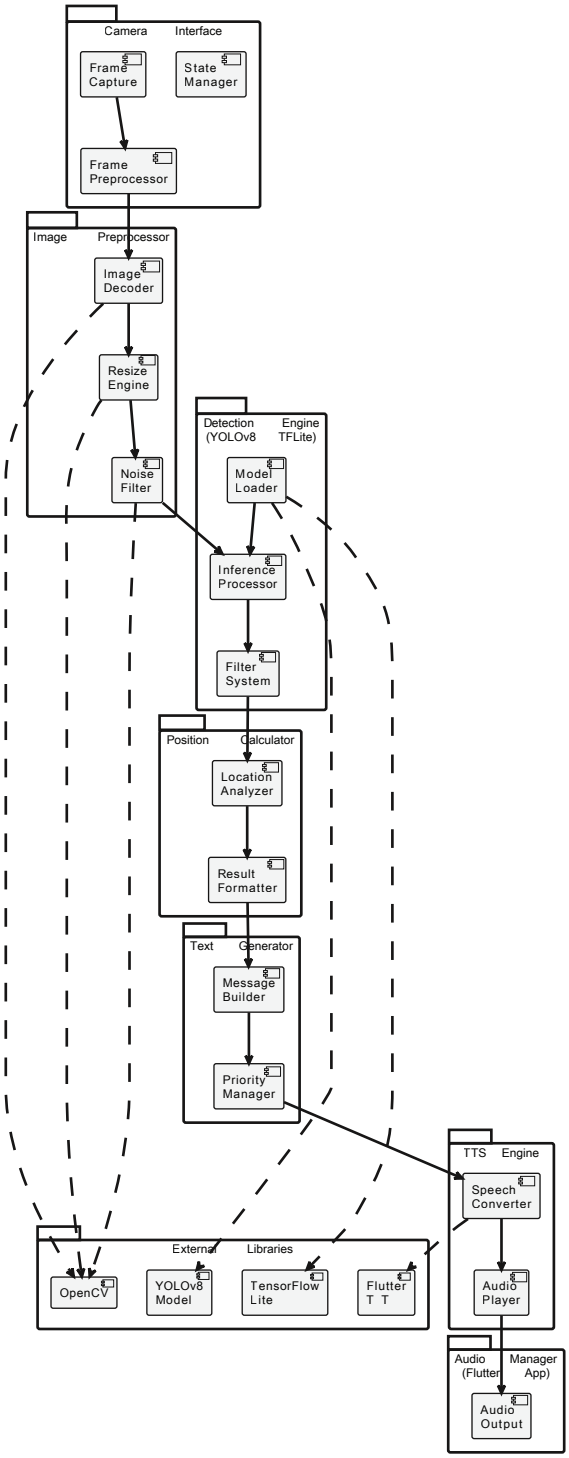
- Maintain a user accessible interface
- Start/stop detection
- Control audio output setting



3.2 Sub-System / Component / Module Level Architectures



3.3 Sub-Component / Sub-Module Level Architecture (1…n)



4. Design Strategies

The design strategies for the Environment Detection System for Blind Users aim for real- time performance, usability for visually impaired users, scalability, and maintain -ability while working within the limits of mobile hardware.

These strategies directly impact the system's overall structure, component inter -action, and design framework.

The system uses a modular, layered, and object-oriented design approach. This allows for independent development, testing, and improvement of components like image acquisition, object detection, decision-making, and audio feedback. All choices focus on the main goals of accessibility, low latency, portability, and cost-effectiveness..

4.1 Strategy 1…n

Strategy 1: Modular and Layered Architectural Design

The proposed system uses a modular and layered architectural approach. Its overall functionality is split into distinct modules: Image Acquisition, Preprocessing, Object Detection, Decision and Interpretation, Audio Feedback, and User Interaction. Each module handles a specific task and interacts with others through clearly defined interfaces.

Description:

The system assesses detected objects based on confidence levels and relevance before giving audio feedback. Only important and critical information is shared with the user.

Reasoning:

Too many audio notifications can overwhelm users and diminish system effectiveness. Filtering ensures users receive clear and practical information, especially about immediate obstacles.

Trade-offs:

Some low-priority or non-essential objects may not be announced, and confidence thresholds need careful adjustment.

Impact Areas:

- Information handling
- User experience
- Operational safety

Strategy 2: Smartphone-Based and Hardware-Independent Design

Description:

The system is designed to work entirely on a standard smartphone. It uses the built-in camera, processing unit, and audio output. No extra hardware or IoT sensors are needed for the system to operate.

Reasoning:

This choice cuts overall system costs and makes it more portable and easier to set up. It also fits within the project scope and helps visually impaired users who already have smartphones.

Trade-offs:

The performance of the system depends on the smartphone's hardware. Without extra sensors, there is limited access to additional environmental data.

Impact Areas:

- System reuse across devices
- Scalable deployment
- Cost-effectiveness

Strategy 3: Real-Time and Low-Latency Processing Design

Description:

To achieve real-time performance, the system uses a lightweight YOLOv3 object detection model, optimized frame processing techniques, and minimal preprocessing operations.

Reasoning:

Timely detection and feedback are crucial for the safety of visually impaired users.YOLO(CV model) strikes a good balance between detection accuracy and processing speed, which makes it a good fit for mobile real-time applications.

Trade-offs:

In challenging environmental conditions, detection accuracy may drop slightly, and continuous processing might increase the load on the device’s CPU or GPU.

Impact Areas:

- Concurrent execution
- Performance optimization
- User safety

- User interface design
- Overall user experience
- Accessibility compliance

Strategy 5: Selective and Intelligent Decision Filtering

Description:

The system assesses detected objects based on confidence levels and relevance before giving audio feedback. Only important and critical information is shared with the user.

Reasoning:

Too many audio notifications can overwhelm users and diminish system effectiveness.Filtering ensures users receive clear and practical information, especially about immediate obstacles.

Trade-offs:

Some low-priority or non-essential objects may not be announced, and confidence thres-holds need careful adjustment.

Strategy 6: Offline-Capable Audio Feedback Mechanism

Description:

Text-to-Speech functionality uses offline-capable engines to maintain system operation no matter the network availability.

Reasoning:

Relying on internet connectivity can lower system reliability, especially outdoors. Offline TTS reduces delays and ensures consistent audio feedback.

Trade-offs:

Offline speech engines might provide slightly lower voice quality and fewer language options compared to cloud-based services.

Impact Areas:

- Data handling
- System reliability
- Real-time response

Strategy 7: Minimal Data Storage and Privacy-Focused Design

Description:

The system processes visual data in real time and avoids storing images or videos unless necessary for testing or debugging. Most data stays in temporary memory.

Reasoning:

This method protects user privacy, reduces storage needs, and follows ethical software design practices, especially for assistive technologies.

Trade-offs:

Limited historical data is available for analysis, and debugging might need manual logging.

Impact Areas:

- Data management
- Privacy and security
- Performance efficiency

Strategy 8: Extensible and Scalable System Structure

Description:

The system's design can support future features like distance estimation, optical character recognition, navigation help, and improved detection models without major changes.

Reasoning:

This design ensures long-term use, supports academic research growth, and allows for future industrial adaptation while keeping the core system functionality intact.

Trade-offs:

Initial planning for the structure takes extra effort to ensure flexibility.

Impact Areas:

- Future enhancements
- Component reuse
- Maintainability

Description:

Concurrent execution is achieved by separating video capture, object detection, and audio output into asynchronous or threaded processes.

Reasoning:

This prevents system blocking, maintains a smooth real-time workflow, and i mproves responsiveness during continuous operation.

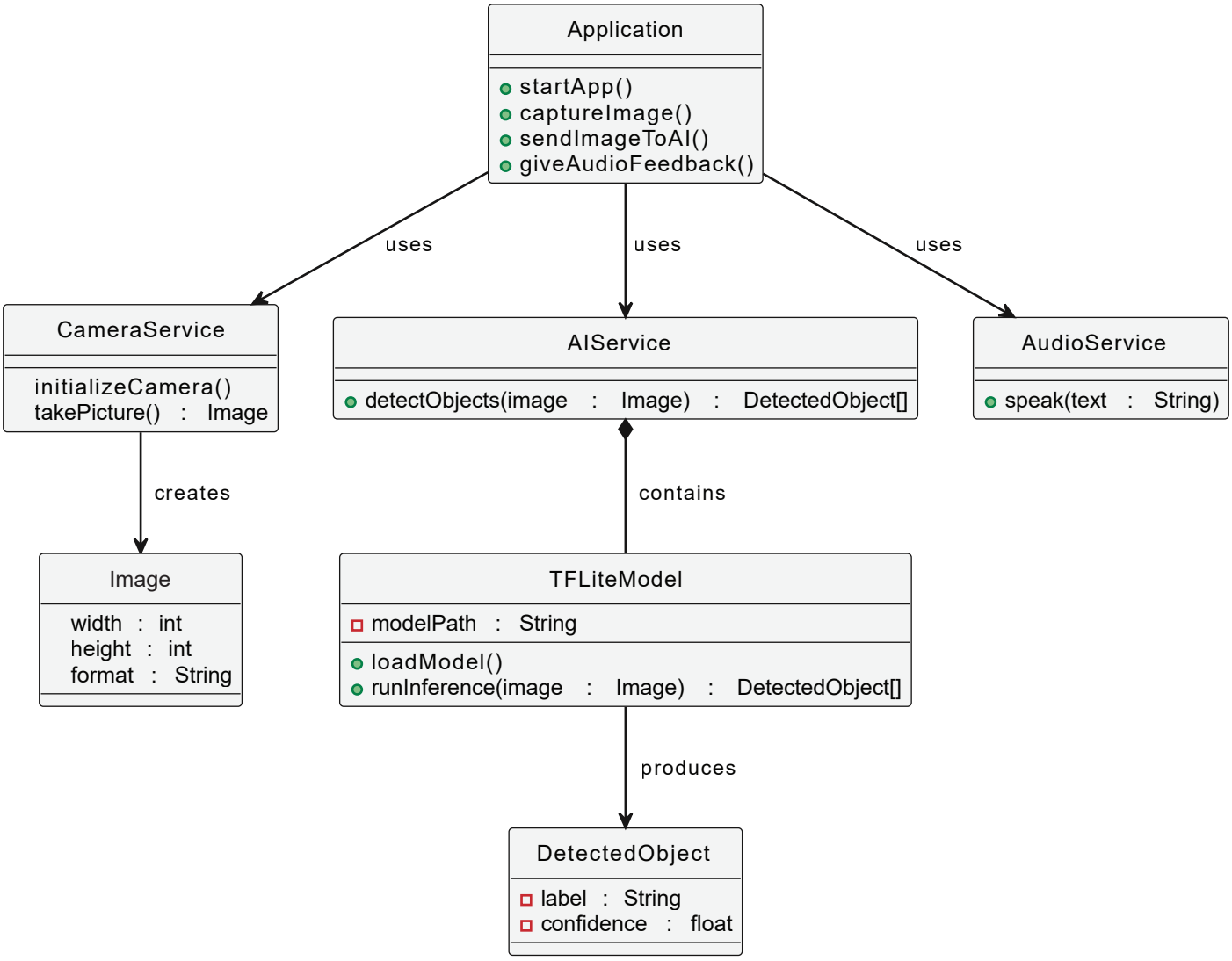
Trade-offs:

Managing threads adds complexity and requires careful synchronization to avoid race conditions.

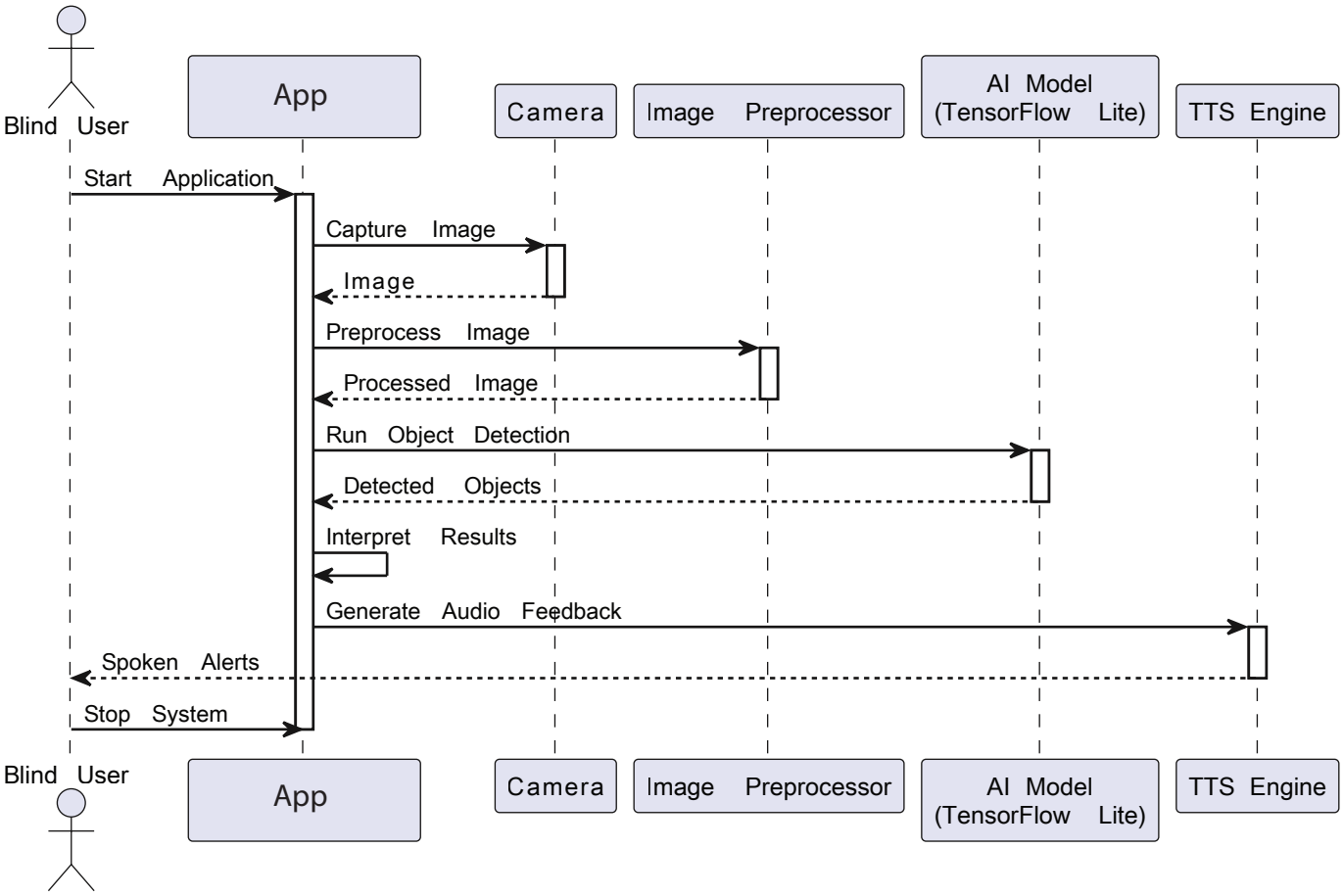
Impact Areas:

- Concurrency handling
- System performance
- Operational stability

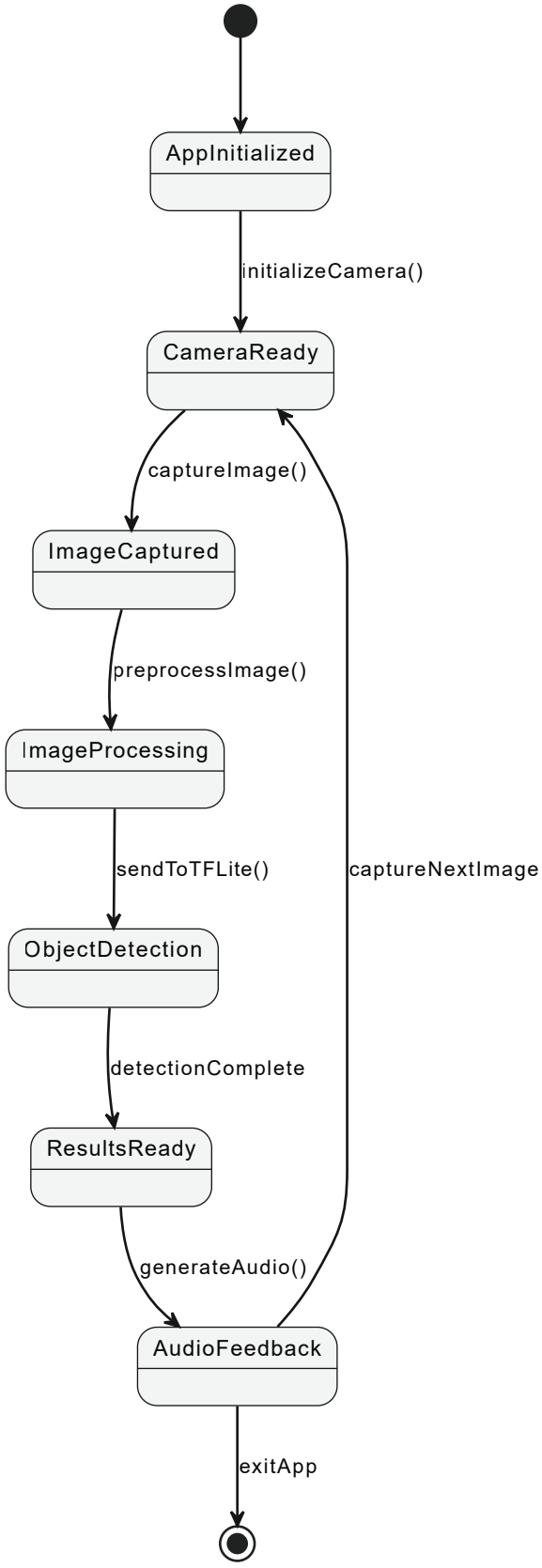
Detailed System Design



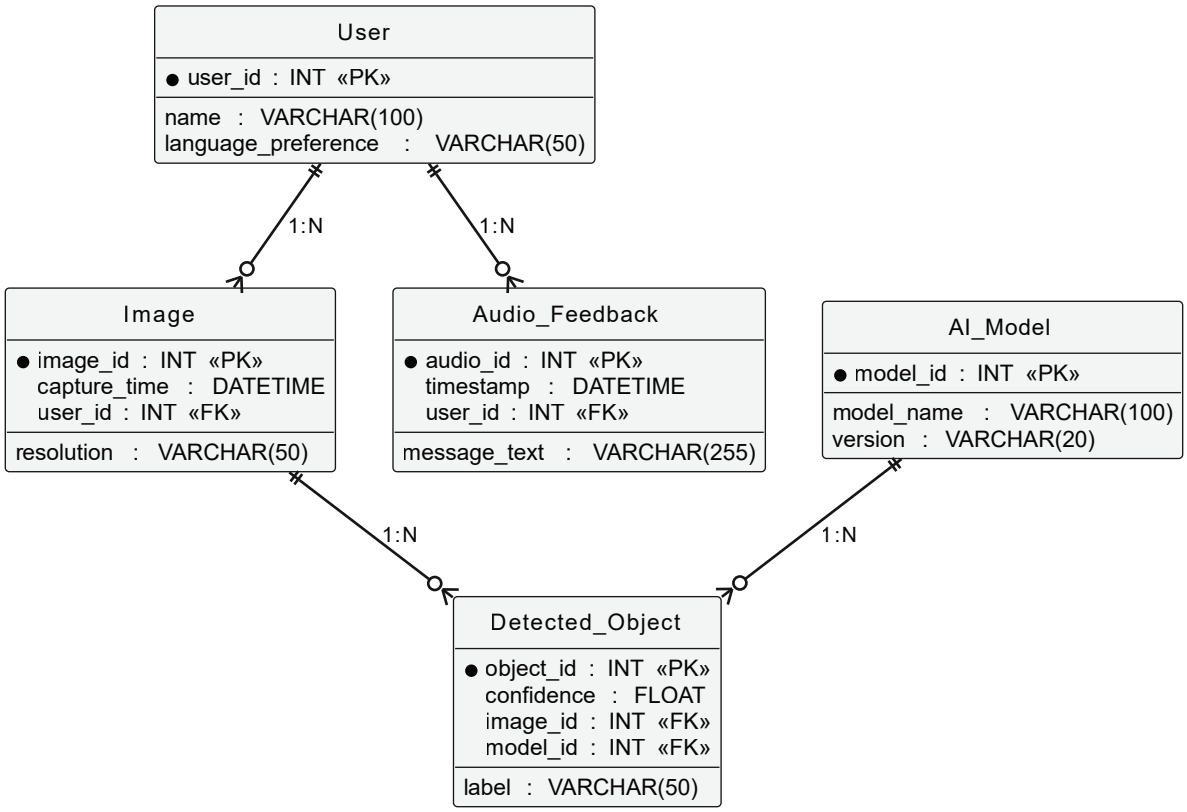
Sequence diagram



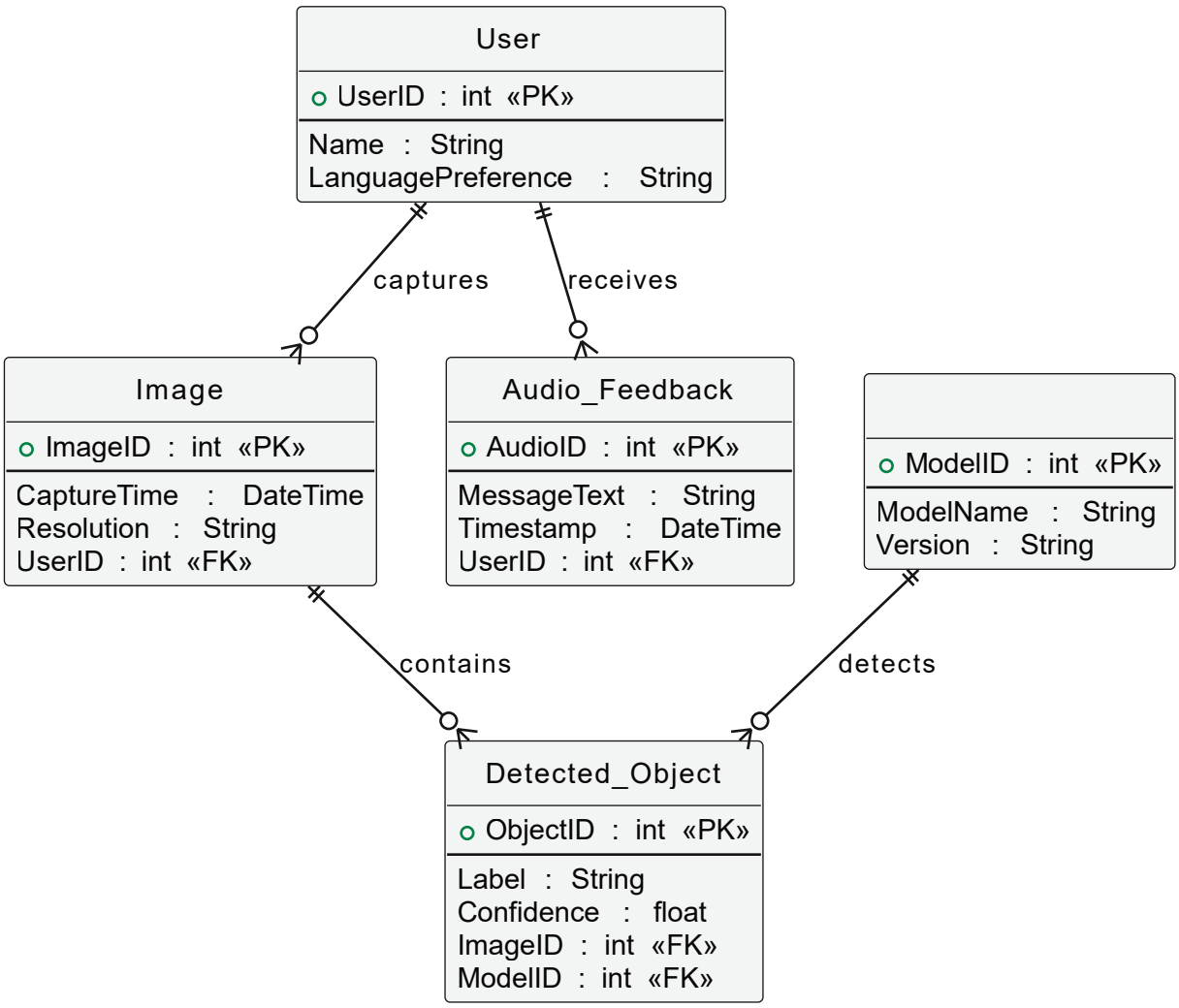
State Transition diagram



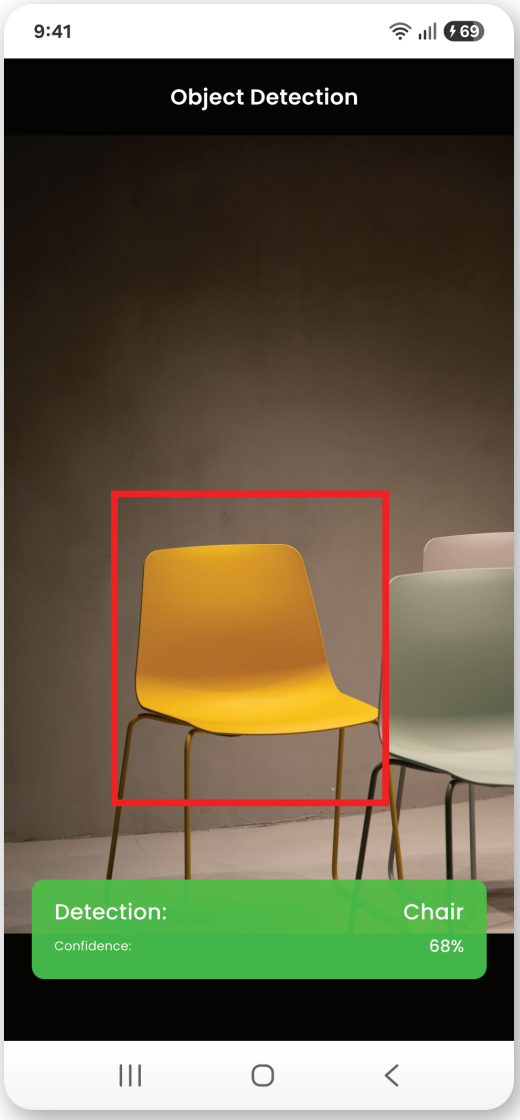
Physical data Model



Logical Model(E/R Model)



Detailed GUI



GUI features:

- Display a live camera preview
- Detected objects using bounding boxes
- Shows the detected object name(label) clearly
- Dispalys confidence percentage
- Provides feeback without user interaction

5. References

Ref. No	Document Title	Date of release	Doc Source
PGBH01 2025- Proposal	Proposal	Oct 20, 2025	https://github.com/ali-raza-91/FYP/tree/main/CP-1/Project_Docs/Proposal
PGBH01 2025-SRS	Software Requirement Specification	Oct 20, 2025	https://github.com/ali-raza-91/FYP/tree/main/CP-1/Project_Docs/SRS
PGBH01 2025-[1[	google lookout	Mar 13, 2019	https://support.google.com/accessibility/android/answer/9031274?hl=en
PGBH01 2025-[2]	Be my eyes	Oct, 2017	https://www.bemyeyes.com/download-app/