CMPE 491

Senior Project 1

Coffee Machine Assistance for the Visually Impaired: A Computer Vision-Based Mobile Application

High-Level Design Report

Project Supervisor

Prof. Dr. Hakkı Gökhan İlk

Project Team Members

Murad Huseynov

Bedir Esen

Hazem Moustafa

Alperen Dalgıç

Table of Contents

1. Introduction

The daily routine of brewing coffee, a simple pleasure for many, presents a significant obstacle for individuals with visual impairments. Modern coffee machines, with their visually dependent interfaces and controls, create barriers to independent operation. This thesis addresses the need for assistive technologies that can bridge this accessibility gap, focusing on empowering visually impaired individuals to confidently use everyday appliances like coffee machines and enhance their daily independence and quality of life. This thesis explores an innovative and accessible approach: utilizing computer vision (CV) and smartphone technology. We focus on developing a mobile application that uses the smartphone camera to "see" and interpret coffee machine interfaces. This project aims to create a specific, real-time guidance system for a chosen coffee machine model, providing precise voice instructions for its operation. The scope of this research is the design, development, and evaluation of a CV-based mobile application to assist visually impaired individuals in independently operating a defined coffee machine model through voice guidance.

Assistive technology research has explored various methods for appliance accessibility, from tactile aids to auditory feedback and voice control. However, many solutions are limited in adaptability, integration ease, or real-world usability. This thesis builds upon this foundation by investigating CV – a rapidly advancing field – to address coffee machine operation specifically. By using visual understanding and real-time guidance, this project aims to contribute a novel, potentially more universally applicable, and context-aware approach to appliance accessibility. This research shows that a computer vision-based mobile application can offer a practical solution for coffee machine assistance, enhancing independence for visually impaired users.

1.1 Purpose of the system

The objective of this project is to design, implement, and evaluate a mobile application that enables visually impaired users to operate a specific coffee machine model independently. The system leverages computer vision via a smartphone camera to "see" and interpret coffee machine interfaces, providing real-time voice guidance through Text-to-Speech engines. The research problem lies in the inaccessibility of modern coffee machines for visually impaired individuals.

1.2 Design goals

1. Accurate Coffee Machine Recognition: To design and implement an EfficientDet [1] based Computer Vision system capable of accurately recognizing a specific coffee machine model and its components in real-time.

2. Effective Voice Guidance: To develop a voice guidance system using platform-specific Text-to-Speech engines (Android TTS [5] and iOS VoiceOver [6]) that provides clear, sequential, and context-aware instructions for operating the recognized coffee machine.

3. Accessible Mobile Application: To integrate the computer vision and voice guidance systems into a user-friendly and accessible mobile application developed using JavaScript and React Native.

4. Usability and Performance Validation: To thoroughly test and evaluate the application's technical performance (e.g., recognition accuracy, guidance latency) and its usability with visually impaired individuals to ensure it provides an effective and accessible method for operating the target coffee machine.

5. Enhanced User Independence: To provide a practical solution that enhances the independence of visually impaired users in performing a daily living task, specifically the operation of a coffee machine.

1.3 Definitions, acronyms, and abbreviations

- AI: Artificial Intelligence
- API: Application Programming Interface
- AT: Assistive Technology
- CPU: Central Processing Unit
- CV: Computer Vision
- GDPR: General Data Protection Regulation
- GPU: Graphics Processing Unit
- HTTPS: Hypertext Transfer Protocol Secure
- iOS: Mobile Operating System by Apple
- JavaScript: Programming Language
- MobileNet: A family of convolutional neural networks for mobile vision applications
- NPU: Neural Processing Unit
- OpenCV: Open-Source Computer Vision Library
- PII: Personally Identifiable Information
- React Native: Framework for building native mobile applications using React
- SQLite: Embedded SQL database engine
- TFLite: TensorFlow Lite
- TTS: Text-to-Speech
- UI: User Interface
- WCAG: Web Content Accessibility Guidelines

1.4 Overview

This project addresses the challenge of independent coffee machine operation for visually impaired individuals by developing a computer vision-based mobile application. The core research problem lies in the inaccessibility of modern coffee machines. The objective is to design, implement, and evaluate a mobile application, developed using JavaScript and React Native, that enables visually impaired users to operate a specific coffee machine model independently. This is achieved by integrating an EfficientDet object detection model, trained for real-time coffee machine recognition, with platform-specific Text-to-Speech engines (Android TTS and iOS VoiceOver) for effective voice guidance. Key results aim to demonstrate the feasibility of accurate coffee machine recognition. Usability testing with visually impaired individuals will be conducted to indicate that the application provides a user-friendly and accessible method for operating the target coffee machine. Ultimately, this research seeks to show that such a mobile application can offer a practical solution for coffee machine assistance, enhancing independence for visually impaired users.

2. Proposed software architecture

2.1 Overview

The proposed software architecture for the Coffee Machine Assistance application adopts a modular, component-based approach designed to optimize performance, maintainability, and cross-platform functionality. This architecture is centered around efficient deep learning for computer vision and accessible mobile application development. The system consists of four primary modules working together to deliver an accessible coffee machine operation experience: an Input Module, a Computer Vision Module, a Voice Guidance Module, and a User Interface Module.

This architecture is highly compatible with the chosen technology stack. The Input Module handles camera input using React Native's capabilities, optimized for interval-based image capture on both Android and iOS. The Computer Vision Module processes these captured images using an EfficientDet object detection model (with a MobileNet [2] backbone), trained with TensorFlow/Keras [3] and deployed via TensorFlow Lite [7] for on-device inference. Image preprocessing and data augmentation leverage OpenCV [4]. The Voice Guidance Module converts the detection results from the CV module into contextual voice instructions using platform-specific native Text-to-Speech (TTS) engines (Android TTS and iOS VoiceOver), ensuring accurate, clear and fast guidance. Finally, the User Interface Module, developed with React Native and JavaScript, provides an accessible interface optimized for screen readers and minimal visual dependency.

The application's data flow is streamlined: camera capture, image preprocessing (resizing, normalization, format conversion), CV processing by the TensorFlow Lite-optimized EfficientDet model to identify coffee machine components and their spatial relationships, feeding results into a contextual processing unit that determines the next instruction, conversion of this instruction into spoken output by the Voice Guidance Module, and processing of user interactions (via accessible touch gestures or voice commands) by the UI module. Local storage using SQLite and file storage handles user settings and the TFLite model, ensuring offline functionality and data privacy. This modular design allows for development efficiency, easier maintenance and updates, cross-platform compatibility with platform-specific optimizations, scalability for future coffee machine models, and critical offline functionality for an assistive technology application.

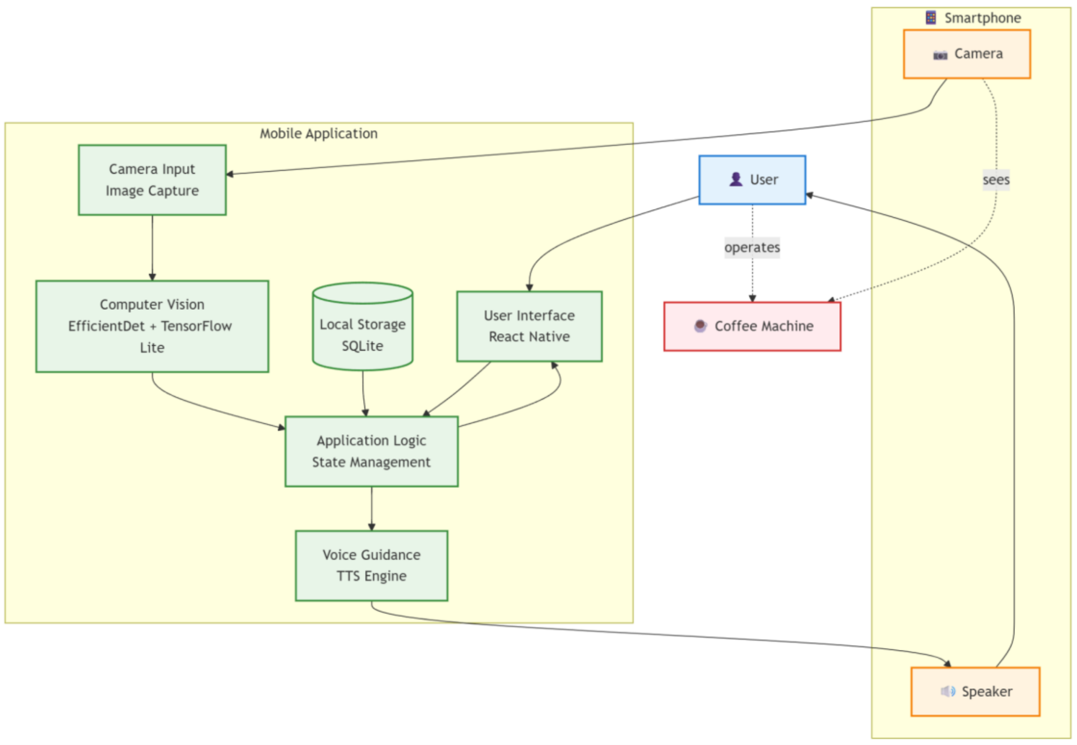

Figure 1: System Architecture Overview

2.2 Subsystem decomposition

The proposed system is structured into four primary subsystems, each responsible for a specific functionality within the mobile application and collectively supporting the goal of accessible coffee machine operation:

Input Module: Manages camera input using React Native’s capabilities. It captures a continuous camera preview for alignment and takes discrete high-resolution frames at regular intervals for analysis.

Computer Vision (CV) Module: Processes captured images using the EfficientDet object detection model with a MobileNet backbone. This module identifies the components and operational states of the coffee machine, forming the basis for the next guidance step.

Voice Guidance Module: Converts detection results into clear, context-aware spoken instructions using the platform-native Text-to-Speech (TTS) engines: Android TTS and iOS VoiceOver. This module ensures the user receives sequential and understandable audio cues.

User Interface (UI) Module: Provides an accessible and screen-reader-compatible interface built with React Native. It supports intuitive gesture-based interactions and is optimized for visually impaired users.

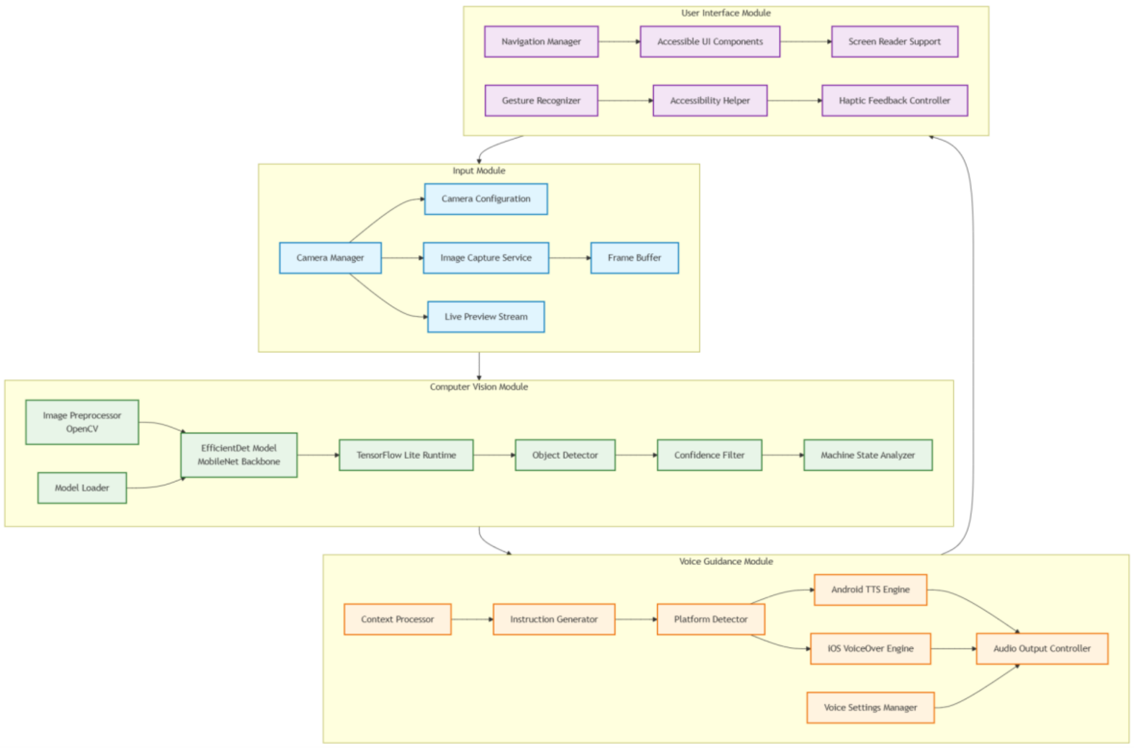

Figure 2: Subsystem Decomposition

2.3 Hardware/software mapping

The system is designed as a mobile application deployed on smartphones. Below is the mapping between hardware and software components:

Hardware:
Smartphone (Android/iOS)
Built-in camera (used for capturing images of the coffee machine)
CPU (primary processing), GPU/NPU (for acceleration if available)
Speaker (for TTS output)
Vibration motor (for haptic feedback)

Software:
React Native: Cross-platform mobile development for UI and input handling.
TensorFlow Lite: Runs the optimized EfficientDet model on-device for object detection.
OpenCV: Performs image preprocessing (e.g., resizing, normalization).
Android TTS / iOS VoiceOver: Converts textual instructions into speech.
SQLite: Stores user settings and preferences locally.
Native Modules: Handle device-specific functionality like camera access and system integration.

This mapping ensures fully on-device execution, enabling offline usage and fast, reliable performance without depending on cloud services.

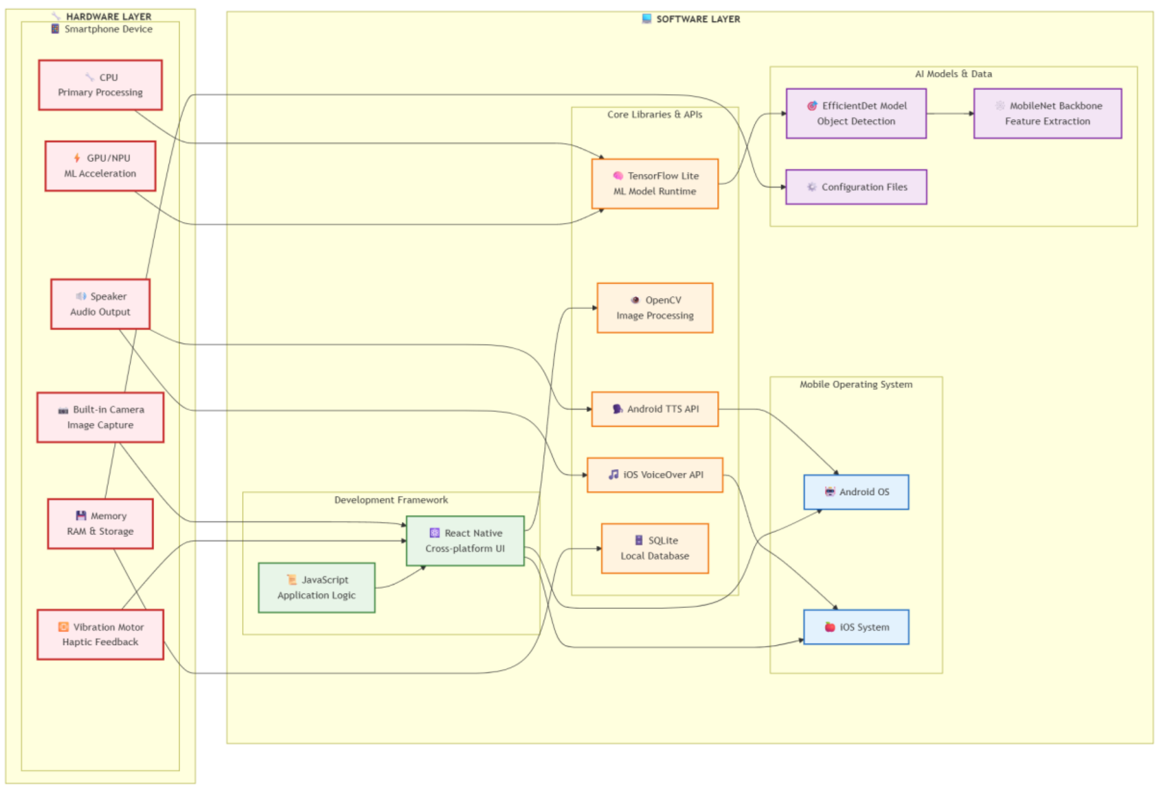

Figure 3: Hardware/Software Component Mapping

2.4 Persistent data management

The application is built with user privacy and data persistence in mind, employing local storage solutions:

Local SQLite Database: Stores user preferences such as voice speed, volume level, and tutorial completion status. These settings ensure a personalized experience and are preserved between sessions.
Model and Configuration Files: The TensorFlow Lite model and configuration data are securely stored in the app's internal storage. These files are read-only during runtime to prevent tampering and ensure model integrity.
Offline Functionality: Since all processing and data storage occur locally, the application functions entirely offline. This eliminates the need for external servers or data transmission, ensuring both privacy and reliability in environments with limited connectivity.
Future-Proofing: If future versions introduce features like cloud-based logging or updates, appropriate privacy-preserving mechanisms (e.g., encrypted communication, user consent dialogs) will be implemented.

2.5 Access control and security

Given the nature of the application as an assistive tool for visually impaired individuals operating a specific coffee machine, no user authentication or login system is required. The application is designed for single-user, device-local usage, and it does not differentiate between users or offer personalized content beyond local preferences.

Data Security & Privacy by Design:

The application adheres to privacy-by-design principles, ensuring that no personally identifiable information (PII) is collected, stored, or transmitted. All user settings and preferences—such as voice speed, volume level, and tutorial completion status—are stored locally in the SQLite database within the app's secure internal storage. These files are sandboxed by the mobile operating system, making them inaccessible to other apps or external services.

Offline Operation: Because the program runs completely offline, there is no requirement for cloud synchronization or external data storage, greatly lowering the surface area for possible data breaches. Additionally, this design decision guarantees usability in settings with spotty or nonexistent internet connectivity.

Security of CV Model and Resources: The program comes with the EfficientDet TFLite model and related configuration files, which are kept in secure internal directories. Because these resources are read-only during runtime, the model's integrity is guaranteed and tampering or unwanted access is avoided.

External Communication: No network requests, analytics platform integrations, or user data transmissions to distant servers are made by the program. This further improves user privacy and removes the dangers of exchanging data with third parties.

Future Considerations: While strictly adhering to accessibility and privacy standards (e.g., WCAG [8] and GDPR compliance), suitable security mechanisms like encrypted data transmission (HTTPS), user consent dialogs, and optional login functionality will be taken into consideration if future iterations of the application include features like remote updates, cloud-based logging, or feedback submission.

In conclusion, the current implementation adheres to the fundamental principles of autonomy, accessibility, and trust for assistive technologies by utilizing on-device processing, local storage, and non-networked architecture to maintain a private and secure environment for users.

2.6 Global software control

The Coffee Machine Assistance mobile application operates as an event-driven system, where the overall software control and application flow are dictated by user interactions, computer vision outputs, and the current state of the coffee-making process. The system is designed to guide the user sequentially through the steps required to operate a coffee machine.

Global software control can be conceptualized as a state machine. The application transitions between various states based on a combination of:

1. **User Input:** The user initiates actions primarily through accessible touch gestures (interpreted by screen readers) or potentially simple voice commands (if implemented). These inputs trigger events like starting the assistance process, confirming a step, or requesting help.
2. **Computer Vision Module Output:** The CV module continuously (at intervals) analyzes the camera feed. Detection of specific coffee machine components (e.g., power button, correct coffee selection button, cup presence) or changes in their state (e.g., water reservoir filled) triggers events that drive the state transitions. Confidence scores from the object detection model play a crucial role in validating these visual cues.
3. **Sequential Guidance Logic:** A predefined decision tree or state machine logic (as detailed in the Voice Guidance Module) determines the appropriate next instruction and the expected subsequent state based on the current state and the inputs received.
4. **Internal Timers/Events:** Certain states might involve timed operations (e.g., waiting for the machine to heat up, brewing time) or internal application events (e.g., camera initialization complete).

The typical flow starts with the user launching the application. The system initializes the camera and the TensorFlow Lite model. The user is then guided to position the phone correctly towards the coffee machine. Once the machine is sufficiently in view, the CV module starts identifying key components. Based on the initial state of the machine (e.g., off, on, needs water) and the user's goal (e.g., make an espresso), the Voice Guidance Module provides step-by-step instructions. Each successful detection of a relevant component or user confirmation of an action (implicitly through CV or explicitly through input) transitions the application to the next logical state in the coffee-making sequence. Error states (e.g., wrong button detected, component not found) will trigger corrective guidance or prompts for the user to readjust the camera or their action. Global control ensures that the user is guided through a coherent and logical sequence, ultimately leading to a successfully brewed coffee or the completion of the desired interaction with the machine.

2.7 Boundary conditions

The Coffee Machine Assistance mobile application incorporates three critical boundary conditions: Initialization, Termination, and Failure Handling, each addressing a distinct phase in the user interaction lifecycle to ensure a seamless and resilient user experience, especially for visually impaired users.

Initialization:

When the application is launched, the user is presented with an accessible home screen that provides two main options through voice direction and a user interface that is compatible with screen readers:

Launch Tutorial Mode: A thorough voice-guided tutorial is launched for new users to show them how to hold the phone, line up the camera with the coffee maker, and use gestures or voice input to engage.

Skip to Operation: Using previously remembered options (such as language or TTS speed), the application can start object detection and guidance right away for users who are returning.

The application loads necessary resources from local storage during initialization, including configuration files, cached settings, and the EfficientDet object identification model. Before going into active mode, the system also quickly checks the camera's condition and the TTS engine's availability to make sure everything is ready.

Termination:

The user can terminate the application at any point using a designated gesture command (e.g., triple-tap) or by closing the app through standard OS mechanisms. Upon termination:

- The app automatically saves user preferences and session logs (e.g., recent successful detection, tutorial completion status) to the local SQLite database.
- All temporary resources (e.g., image buffers) are released, and any background inference threads are gracefully shut down.
- The system reads out a closing message to confirm the session ended successfully and ensures no residual processes consume device resources.

Failure Handling:

The application includes robust mechanisms to gracefully handle unexpected interruptions or failures:

- If the app crashes or is forcefully closed (e.g., due to low memory or user force stop), the system ensures that no user preferences are lost, thanks to periodic writes and atomic saves to the local database.
- If the object detection model fails to load or the camera feed is unavailable, the app switches to fallback mode with explanatory voice instructions and prompts the user to retry or restart the app.
- In rare cases of hardware incompatibility (e.g., missing TTS engine or unsupported camera resolution), appropriate voice error messages are delivered, and the issue is logged locally for future debugging.

These boundary conditions ensure a resilient, user-friendly, and accessible experience aligned with assistive technology standards and user expectations.

3. Subsystem services

The "Coffee Machine Assistance" mobile application provides a range of services to enable visually impaired users to operate coffee machines independently. These services are delivered through the coordinated efforts of the architectural subsystems and their components detailed in Section 2.2 (Figure 2).

User Interaction and Interface Services:

- **Service Description:** Manages all user interactions, including accessible touch gestures and screen reader compatibility. Provides clear, accessible UI elements.
- **Responsible Components:** Primarily the User Interface (UI) Subsystem, including its Navigation Manager, Accessible UI Components, Screen Reader Support, Gesture Recognizer, and Accessibility Helper.

Camera Input and Management Services:

- **Service Description:** Handles access to the device camera, manages the live preview stream for alignment, and captures discrete, high-resolution image frames for analysis.
- **Responsible Components:** The Input Module, including its Camera Manager, Image Capture Service, and Frame Buffer.

Computer Vision Processing Services:

- **Service Description:** Performs real-time image analysis to recognize coffee machine components and their operational states. This includes image preprocessing, model inference, and confidence filtering.
- **Responsible Components:** The Computer Vision (CV) Subsystem, utilizing its Image Preprocessor (OpenCV), Model Loader, EfficientDet/MobileNet model via TensorFlow Lite Runtime, Object Detector, Confidence Filter, and Machine State Analyzer.

Voice Guidance and Audio Feedback Services:

- **Service Description:** Converts detected machine states and contextual information into clear, sequential, and context-aware spoken instructions using native Text-to-Speech engines. Manages audio output and voice settings.
- **Responsible Components:** The Voice Guidance Module, including its Context Processor, Instruction Generator, Platform Detector, native TTS Engines (Android/iOS), Audio Output Controller, and Voice Settings Manager.

Application Logic and Control Services:

- **Service Description:** Orchestrates the overall application flow, sequences user guidance, manages system states (based on user input and CV detections), handles errors, and adapts instructions dynamically.
- **Responsible Components:** This is a core logical function, supported by elements within the Voice Guidance Module (Context Processor, Instruction Generator), CV Module (Machine State Analyzer), and coordinated interaction logic.

Visual Rendering Services:

- **Service Description:** Manages the display of visual feedback, such as the live camera feed and detected object bounding boxes. This is primarily for users with partial sight or for development/debugging. Ensures smooth UI rendering.
- **Responsible Components:** Aspects of the User Interface (UI) Subsystem responsible for displaying visual elements.

Interaction Feedback (Haptic) Services:

- Service Description: Provides tactile feedback (vibrations) in response to user gestures and key interaction points, enhancing the non-visual usability and natural feel of the application.
- Responsible Components: The User Interface (UI) Subsystem's Haptic Feedback Controller and Gesture Recognizer.

Persistent Data Management Services:

- Service Description: Manages the storage and retrieval of user preferences (e.g., voice speed, volume), cached CV models, and configuration files, ensuring data persistence and offline functionality.
- Responsible Components: Utilizes SQLite and system file storage.

4. Glossary

Computer Vision (CV): A field of artificial intelligence that enables computers and systems to derive meaningful information from digital images, videos, and other visual inputs. In this project, it's used to interpret coffee machine interfaces.

EfficientDet: An object detection model architecture known for its balance of accuracy and computational efficiency, making it suitable for deployment on mobile devices. It is the core model used for recognizing coffee machine components.

JavaScript: A high-level programming language primarily known as the scripting language for Web pages, but also used in many non-browser environments, including mobile app development with frameworks like React Native.

MobileNet: A class of lightweight deep convolutional neural networks designed for mobile and embedded vision applications. It serves as the backbone for the EfficientDet model in this project.

Native Modules: In the context of React Native, these are components written in the native programming language of the platform (e.g., Java/Kotlin for Android, Objective-C/Swift for iOS) that bridge JavaScript code with native device capabilities like camera access or specific system integrations.

OpenCV (Open-Source Computer Vision Library): An open-source library of programming functions mainly aimed at real-time computer vision. Used in this project for image preprocessing tasks such as resizing and normalization.

React Native: An open-source UI software framework created by Meta Platforms, Inc. It is used to develop applications for Android, Android TV, iOS, macOS, tvOS, Web, Windows, and UWP by enabling developers to use the React framework along with native platform capabilities.

SQLite: A C-language library that implements a small, fast, self-contained, high-reliability, full-featured, SQL database engine. It is used in this application for local storage of user settings and preferences.

TensorFlow Lite (TFLite): A version of TensorFlow, an open-source machine learning framework, that is optimized for on-device inference on mobile and embedded devices. It enables running the EfficientDet model directly on the smartphone.

Text-to-Speech (TTS): A type of assistive technology that reads digital text aloud. It is used in this application to provide voice guidance to visually impaired users.

5. References

[1] M. Tan, R. Pang, and Q. V. Le, "EfficientDet: Scalable and Efficient Object Detection," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020. arXiv:1911.09070.

[2] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," arXiv:1704.04861, 2017.

[3] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, et al., "TensorFlow: A System for Large-Scale Machine Learning," in 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI), 2016, pp. 265-283.

[4] OpenCV team, "OpenCV (Open Source Computer Vision Library)". [Online]. Available: https://opencv.org/. [Accessed: March 28, 2025].

[5] Android Developers, "TextToSpeech | Android Developers". [Online]. Available: https://developer.android.com/reference/android/speech/tts/TextToSpeech. [Accessed: March 29, 2025].

[6] Apple Inc., "VoiceOver | Apple Developer Documentation," [Online]. Available: https://developer.apple.com/documentation/accessibility/voiceover. [Accessed: March 29, 2025].

[7] TensorFlow, "TensorFlow Lite Guide". [Online]. Available: https://www.tensorflow.org/lite/guide. [Accessed: March 30, 2025].

[8] World Wide Web Consortium, "Web Content Accessibility Guidelines (WCAG) 2.1," W3C Recommendation, 2018. [Online]. Available: https://www.w3.org/TR/WCAG21/. [Accessed: April 1, 2025].