S.A.G.E - Personalized AI-Powered Smartglass

CSD 415 - PROJECT PHASE 1

MDL22CS036	22CSB08 ANANYA P
MDL22CS126	22CSB36 M GAYATHRI
MDL22CS138	22CSB41 NAVNEET RANJISH PILLAI
MDL22CS144	22CSB44 NIKHIL M

B. Tech. Computer Science Engineering

Department of Computer Engineering
Model Engineering College, Ernakulam
Thrikkakara, Kochi - 682021
Phone: 91.484.2575370
http://www.mec.ac.in
hodcs@mec.ac.in

October 2025

Model Engineering College, Ernakulam

Dept. of Computer Engineering

C E R T I F I C A T E

This is to certify that, this report titled ***"S.A.G.E - Personalized AI-Powered Smartglass"*** is a bonafide record of the work done by:

MDL22CS036	22CSB08	**ANANYA P**
MDL22CS126	22CSB36	**M GAYATHRI**
MDL22CS138	22CSB41	**NAVNEET RANJISH PILLAI**
MDL22CS144	22CSB44	**NIKHIL M**

Seventh Semester B. Tech. Computer Science & Engineering students, for the course work in **CSD 415 - Project Phase 1** which is the Main Project Work, under our guidance and supervision, in partial fulfillment of the requirements for the award of the degree, B. Tech. Computer Science and Engineering of **APJ Abdul Kalam University**.

Guide

Mrs. Arya Mary K J
Assistant Professor
Computer Engineering

Coordinator

Dr. Priya S
Professor
Computer Engineering

Head of the Department

Dr. Binu V. P
Professor
Computer Engineering

October, 2025

Acknowledgements

We are profoundly grateful to **Mrs. Arya Mary K J** for expert guidance and continuous encouragement throughout to see that this project rights its target since its commencement to its completion.

We would like to express deepest appreciation towards **Dr. Jayachandran E S**, Principal, Govt. Model Engineering College, Kochi, **Dr. Binu V P**, Head of Department of Computer Engineering and **Dr. Priya S**, Project Coordinator whose invaluable guidance supported us in completing this project.

At last we must express our sincere heartfelt gratitude to all the staff of Computer Engineering Department who helped us directly or indirectly during this course of work.

22CSB08	ANANYA P
22CSB36	M GAYATHRI
22CSB41	NAVNEET RANJISH PILLAI
22CSB44	NIKHIL M

Abstract

The objective of this project is to develop a next-generation AI-powered smartglass system that seamlessly integrates Machine Learning (ML), Deep Learning (DL), and Internet of Things (IoT) technologies to deliver an intelligent, hands-free wearable experience. The proposed prototype consolidates multiple smart functionalities within a compact and ergonomic form factor, enabling users to interact with their surroundings naturally and intuitively.

Designed with accessibility, convenience, and personalization at its core, the system features a dynamic heads-up display (HUD) for real-time navigation feedback, music service integration for playback control and media information, and a personalized AI assistant powered by Google Gemini. To enhance usability - particularly for visually impaired individuals. The smartglass incorporates object recognition capabilities that identify and describe elements within the environment.

The device further supports seamless hands-free interaction, allowing users to manage voice calls, execute contextual commands (e.g., “What’s in front of me?” or “Set an alarm”), and translate live conversations into English in real time. A built-in text-to-speech (TTS) module ensures that AI-generated responses are conveyed audibly, creating an engaging and accessible user experience. This integration of AI-driven perception, natural interaction, and wearable design establishes the system as a step toward practical, context-aware assistive technology for everyday environments.

Contents

List of Figures

List of Tables

1 Introduction

1.1 Overview

The evolution of wearable technology has steadily transformed the boundary between the digital and physical worlds. Devices once limited to activity tracking have matured into context-aware companions capable of intelligent perception and real-time interaction. In this rapidly advancing ecosystem, S.A.G.E (Situational Awareness and Guidance Engine) emerges as an innovative smartglass platform that integrates artificial intelligence, computer vision, and natural language understanding to deliver seamless human–machine collaboration.

S.A.G.E is designed around the principle of contextual assistance - enabling users to perceive, interpret, and interact with their environment through intelligent, hands-free operation. The system fuses visual recognition, audio feedback, and a dynamic heads-up display (HUD) to enhance situational awareness in daily life. Built on the compact Raspberry Pi Zero 2 W, the device achieves a delicate balance between affordability, portability, and performance, making AI-powered assistance accessible to a wider audience.

1.2 System Vision

S.A.G.E represents the evolution of assistive wearables into adaptive, AI-driven companions. It provides personalized environmental understanding, allowing users to access contextual insights in real time. The integration of machine learning with embedded hardware transforms everyday experiences - bridging human intent with computational intelligence through intuitive interaction modes such as speech, vision, and gesture.

A distinguishing aspect of S.A.G.E is its distributed architecture, which delegates complex tasks to a mobile companion application and cloud-based backend. This approach ensures real-time responsiveness without overburdening the onboard hardware. Intensive inference workloads, including facial recognition and translation, are offloaded to the backend, while lightweight processes such as input capture and HUD rendering occur on the device. This division of responsibilities allows efficient multitasking and low latency performance.

1.3 Design Philosophy

The proposed framework follows a software-centric design approach where the hardware merely acts as an interface layer for multimodal inputs and outputs. The Raspberry Pi manages real-time display rendering, and voice activation, while the backend - powered by Python-based APIs - handles high-level AI reasoning. The companion app developed in Flutter synchronizes both systems, ensuring continuous connectivity and an intuitive user experience.

This modular design simplifies hardware dependencies and encourages scalability. Additional features such as gesture control, extended object memory, or contextual scene understanding can be incorporated with minimal architectural changes. Each feature operates as a distinct module under a unified communication protocol, ensuring consistency and extensibility in future iterations.

1.4 Objective

The primary objective of S.A.G.E is to create an accessible and intelligent wearable platform that enhances user autonomy and situational awareness. By combining embedded computing with AI-driven perception, the system aims to:

- Provide hands-free interaction through voice recognition and natural language understanding.
- Translate visual information into meaningful feedback for accessibility and navigation.
- Enable real-time environment understanding via object and facial recognition.
- Support modular software architecture for rapid feature development and deployment.
- Deliver affordable AI integration suitable for both assistive and general-purpose applications.

1.5 Scope

S.A.G.E demonstrates the fusion of artificial intelligence, Internet of Things, and human-computer interaction into a single cohesive wearable framework. It serves as a foundation for future research in context-aware computing, real-time AI systems, and accessibility-oriented design. The project not only illustrates the technical feasibility of embedding machine learning into constrained hardware but also emphasizes the social impact of enabling inclusive and intelligent technologies.

The scope extends beyond functional demonstration - it envisions a future where wearable systems evolve into proactive, adaptive assistants that understand and augment human perception. Through continuous development and integration of advanced AI models, S.A.G.E stands as a milestone toward redefining how humans interact with intelligent machines.

2 System Motivation and Key Features

2.1 Motivation for S.A.G.E

Traditional wearable technologies, though increasingly prevalent, often fall short in offering true contextual intelligence and real-time adaptability. Most existing devices function as passive data collectors-tracking metrics like movement, heart rate, or ambient conditions - without interpreting or responding to the user's environment in a meaningful way. Moreover, these systems tend to rely on proprietary ecosystems, high-cost hardware, or limited accessibility features, which restricts their usability among diverse audiences.

S.A.G.E addresses these limitations by merging artificial intelligence, computer vision, and natural language processing into a cohesive and adaptive wearable platform. The device is conceived not merely as a smartglass but as an intelligent assistant capable of perceiving, understanding, and responding to its surroundings. It facilitates a seamless interaction loop-where user intent, environmental context, and computational reasoning converge to enhance autonomy and awareness.

The motivation behind S.A.G.E is deeply rooted in accessibility and inclusivity. Visually impaired users, for instance, face continuous challenges in navigation and interaction with unfamiliar environments. While high-end AR systems exist, their prohibitive cost and hardware dependencies make them inaccessible to most individuals. S.A.G.E bridges this gap by offering a lightweight, affordable, and open-source platform capable of delivering real-time object recognition, facial identification, and translation through intuitive voice commands.

By offloading computationally heavy processes to a companion mobile application and hosted backend, the system achieves high responsiveness without compromising portability. It exemplifies a paradigm where wearables evolve from task-specific gadgets to intelligent companions that understand context, intent, and emotion-fundamentally redefining personal assistive technology.

2.2 Core Features and Functional Design

S.A.G.E integrates multiple AI-driven modules under a unified system architecture, enabling a wide range of assistive and utility functions. Each feature is carefully engineered to optimize usability, performance, and accessibility.

2.2.1 Heads-Up Display (HUD) Interaction

The embedded heads-up display acts as a visual feedback interface, reflecting real-time information such as navigation cues, object labels, or translation results. Its context-aware behavior ensures that only relevant data is displayed, reducing visual clutter and improving focus. The reflective display assembly provides visual output while maintaining lightweight ergonomics suitable for continuous wear.

2.2.2 Visual Recognition and Environment Awareness

S.A.G.E incorporates dual machine vision modules for object and facial recognition. Using deep learning models optimized through TensorFlow Lite and OpenCV, the system identifies and classifies visual entities in real time. Recognized objects are displayed on the HUD and announced through audio output, allowing users to perceive their surroundings through multimodal feedback. The facial recognition feature matches detected individuals with stored profiles, offering personalized interactions and memory recall functions.

2.2.3 Voice-Activated AI Assistant

The integrated voice assistant enables natural interaction through hands-free commands. Triggered by the wake phrase “Hey S.A.G.E,” it interprets spoken instructions using speech recognition and processes them through the Gemini API. This module provides conversational responses, executes feature control (such as enabling translation or navigation), and ensures fluid dialogue-based operation without the need for manual input.

2.2.4 Real-Time Translation

Through its translation pipeline, S.A.G.E converts captured text or speech from one language to another. The camera captures the visual text, which is processed via Google Vision for OCR and translated using LibreTranslate. The translated content is then either displayed on the HUD or delivered as speech output. This feature enhances accessibility for travelers, students, and professionals engaging in multilingual contexts.

2.2.5 Navigation Assistance

The navigation module integrates with the Google Maps API through the mobile application to deliver real-time routing and guidance. Voice commands initiate navigation, after which the HUD displays route details while the system provides turn-by-turn auditory instructions. This dual-mode guidance ensures safety and convenience during movement, especially for visually impaired users.

2.2.6 Performance and Customization

To ensure efficiency, the device utilizes a hybrid processing approach. Lightweight operations such as image capture and voice activation are performed locally, while high-computation tasks like inference and translation are offloaded to the mobile app or backend. This design minimizes latency, prevents overheating, and extends battery life. Users can also customize system behavior-enabling or disabling modules such as object scanning, translation, or music control-based on personal requirements.

3 Literature Survey

3.1 Overview of Related Work

The development of assistive wearable technology has rapidly advanced with the integration of Artificial Intelligence (AI), Internet of Things (IoT), and embedded systems. Smart glasses have evolved from simple augmented reality devices to complex AI-driven systems capable of real-time visual understanding and interaction. Existing research has primarily focused on enhancing accessibility for visually impaired individuals through object recognition, facial identification, and environment awareness. However, these systems often encounter challenges related to hardware cost, processing limitations, latency, and real-time adaptability.

A comprehensive review of existing works reveals the transition from hardware-intensive to software-optimized designs, where computation is distributed between the wearable unit and companion applications. Several key research contributions that have influenced the design of S.A.G.E are explored, emphasizing methods, findings, and technological innovations in the domain of AI-powered smart glasses.

3.2 Smart Navigation Glasses for Independent Living [1]

Sathya et al. developed a smart wearable system integrating ESP32-CAM, ultrasonic sensors, and Raspberry Pi for real-time obstacle detection and facial recognition. The system utilized OpenCV and TensorFlow Lite for machine learning inference, offering both online and offline object detection capabilities. It provided features such as text-to-speech output, SOS messaging, and Bluetooth connectivity. The hybrid approach of combining sensor-based navigation with AI inference enhanced user independence and reliability in various lighting conditions. Despite these strengths, the reliance on multiple sensors increased power consumption and limited portability.

3.3 Smart Glasses for Visually Impaired People Using Machine Learning [2]

Siva Priyanka et al. proposed a Raspberry Pi-based smart glass integrating object detection using the YOLOv3 model with OpenCV for real-time vision processing. The device converted visual data into audio prompts using Google's Text-to-Speech API, significantly improving user mobility. The design demonstrated high recognition accuracy and reduced cost compared to commercial alternatives. However, the system was restricted to a fixed set of detectable classes and struggled in complex or cluttered environments, highlighting the need for more adaptive AI models.

3.4 Visual Information Translator Using Smart Glasses for the Blind [3]

Rani et al. presented a system capable of translating printed text into speech for blind users. The device used Optical Character Recognition (OCR) combined with text-to-speech synthesis, running on Raspberry Pi and integrated with mobile connectivity. This approach was highly effective for reading printed material and signage, promoting accessibility in public spaces. The limitation of the model was its dependency on printed text; it could not interpret handwritten or complex visual scenes, restricting its applicability in dynamic environments.

3.5 Google Pi: A Raspberry Pi Based Smart Assistant [4]

Katkar et al. introduced Google Pi, a deep learning-powered smart assistant that combined voice commands, object detection, and currency recognition for visually impaired users. Using Raspberry Pi, YOLOv3, and convolutional neural networks (CNNs), the system achieved 92.4% accuracy in object and currency identification. It effectively merged affordability with intelligent interaction by integrating Google APIs for cloud-based inference. However, the system required substantial computational resources, and its dependence on stable internet connectivity limited standalone usability.

3.6 Smart Glass for Visually Impaired Person [5]

Sabitha et al. designed IoT-enabled smart glasses with GPS, ultrasonic sensors, ESP32-CAM, and Bluetooth communication for navigation and emergency support. The system could alert emergency contacts with live video and geolocation during critical situations. It effectively differentiated familiar from unfamiliar environments, offering safety and independence. The design, though comprehensive, was limited by Bluetooth range and high dependency on external mobile devices.

3.7 Facial Recognition Smart Glasses for Visually Impaired People [6]

Ghodake et al. implemented a smart glass that integrated Raspberry Pi, ESP32-CAM, and Google's Text-to-Speech API to perform facial recognition and obstacle detection. The system employed Haar Cascade classifiers through OpenCV to identify individuals and provide auditory feedback. It demonstrated high recognition precision in controlled conditions but suffered from reduced accuracy in low light and increased latency when handling multiple recognition tasks concurrently.

3.8 Smart Glasses Embedded with Facial Recognition Technique [7]

Kumar et al. presented an advanced multi-functional smart glass utilizing Raspberry Pi 4, ultrasonic sensors, and ESP32 cameras. It incorporated both facial and object detection along with entertainment features such as music playback and voice assistance. The design achieved multifunctionality but introduced power management challenges due to increased component integration. The bulkiness of the hardware setup also affected the comfort and wearability of the device.

3.9 Indoor Navigation Glasses Using Deep Learning and Audio Guidance[8]

Loresco et al. developed an indoor navigation system employing MobileNetV2 and MobileNet-SSD on a Raspberry Pi 4B coupled with a Coral Edge TPU accelerator. The glasses provided precise navigation through deep learning models optimized for obstacle and room detection. Audio cues guided users in real-time, achieving 98.96% navigation accuracy without external markers. Although the system excelled in accuracy, it required specific training datasets and dedicated hardware accelerators, increasing deployment costs.

3.10 Single-Shot Image Recognition Using Siamese Neural Networks [9]

Malhotra proposed a Siamese Neural Network (SNN) model for one-shot image recognition capable of identifying objects and faces using minimal training data. The SNN approach demonstrated efficiency in memory-constrained environments and provided high similarity-based recognition accuracy. However, generalization to unseen data was limited, and performance degraded under noise or poor illumination. The study's insights contributed significantly to the development of lightweight, embedded facial recognition systems.

3.11 Comparative Analysis of Reviewed Works

The reviewed studies demonstrate significant progress in wearable assistive systems. Table 3.1 summarizes key features, advantages, and limitations of the referenced systems.

Table 3.1: Comparison of Literature on Smart Glasses for the Visually Impaired

Year	Paper	Advantages	Disadvantages
2023	*Visual Information Translator Using Smart Glasses for Blind*	- Comprehensive features (TTS, object/face detection, SOS). - Cost-effective design with a user-friendly Flutter app.	- Fully dependent on a paired smartphone for processing. - Potential for network latency and battery life issues.
2023	*Smart Glasses Embedded with Facial Recognition Technique*	- Multifunctional design with facial recognition and obstacle detection. - Includes digital assistant features like web browsing.	- Relies on a static database requiring manual code updates. - Requires an internet connection to function.
2023	*Facial Recognition Smart Glasses for Visually Impaired People*	- Wide feature set (YOLO, OCR, currency recognition). - Robust hybrid sensing using a camera and ultrasonic sensor.	- Device is noted to be fragile and easily damaged. - High power consumption and risk of malfunction are key concerns.
2023	*Single-Shot Image Recognition Using Siamese Neural Networks*	- Achieves high accuracy (92%) in one-shot learning tasks. - Does not require retraining to recognize new classes.	- Complex training process that learns a similarity metric. - Performance is slightly lower than state-of-the-art models.
2023	*Smart Glasses for Visually Impaired People using Machine Learning*	- High-precision (99%) real-time multi-object detection using YOLOv3. - Detects multiple objects within a single frame.	- Lacks distance measurement (no ultrasonic sensor). - Requires internet for Google's Text-to-Speech API.
2024	*Indoor Navigation Glasses for the Visually Impaired*	- High real-time performance using a Google Coral Edge TPU. - Achieved a high user usability score (SUS of 85).	- Obstacle detection is limited (trained only on chairs). - The design is a bulky two-part system.
2024	*Google Pi Using Raspberry Pi for Visually Impaired*	- Integrates Google Assistant with custom recognition models. - Cost-effective and accessible Raspberry Pi foundation.	- Lacks navigational or obstacle avoidance features. - Requires constant internet for Google Assistant to work.
2025	*Smart Glass for Visually Impaired Person*	- Context-aware alerts to reduce false alarms in different locations. - Sends emergency data (live video and GPS) to contacts.	- Limited on-board processing (Arduino) requires a smartphone. - Does not include OCR or facial recognition features.

3.11.1 Inferences and Research Gaps

The literature highlights a continuous effort to integrate AI with assistive wearables for improved accessibility. Most systems achieve reliable recognition accuracy but face challenges such as latency, limited adaptability, and constrained hardware performance. Several works rely heavily on external servers or proprietary APIs, which restrict scalability and increase dependency. The transition from sensor-heavy designs to AI-optimized architectures signifies a major step toward lightweight, modular solutions.

S.A.G.E advances this domain by introducing a hybrid, modular design that distributes computational tasks across the wearable unit, mobile companion application, and cloud-based AI backend. It eliminates excessive sensor reliance while maintaining real-time interaction and accuracy. By combining voice activation, HUD feedback, and cloud-assisted inference, it bridges the gap between affordability, intelligence, and accessibility in modern assistive wearables.

4 Problem Statement and Solution

4.1 Problem Statement

4.1.1 Problem Statement

Modern wearable technologies, though advanced, remain limited in addressing real-world accessibility needs. Existing smart glasses often rely on proprietary ecosystems, expensive hardware, and lack the intelligence required for real-time situational awareness. Users, particularly those with visual impairments, struggle with environmental perception, object recognition, and independent navigation. The core challenge lies in developing an affordable, open-source, and context-aware smartglass system capable of processing visual, auditory, and contextual information in real time without compromising usability or performance.

4.1.2 Challenges in Existing Systems

The current generation of assistive smart glasses presents several critical limitations:

- **High Cost and Proprietary Designs:** Commercial solutions are often expensive, restricting accessibility for general users and research adaptation.
- **Limited AI Integration:** Many systems rely on pre-defined tasks without adaptive learning or contextual awareness.
- **Hardware Constraints:** Excessive sensor usage increases power consumption and heat, reducing wearable comfort.
- **Lack of Real-Time Processing:** Visual and auditory feedback often suffers from latency, hindering responsiveness.
- **Minimal User Customization:** Users cannot modify or tailor features based on specific use cases, such as accessibility or travel assistance.

Addressing these challenges requires an intelligent, modular, and cost-efficient system that can balance computational efficiency, comfort, and real-time performance.

4.2 Proposed Solution

4.2.1 System Overview

S.A.G.E - **Situational Awareness and Guidance Engine** - is a wearable smartglass designed to enhance real-world perception through artificial intelligence, embedded systems, and seamless human–machine interaction. The proposed system integrates a Raspberry Pi-based hardware platform with a companion mobile application, enabling cloud-assisted AI computation and real-time feedback through a Heads-Up Display (HUD) and voice interface.

4.2.2 Core Objectives

- **Affordable Accessibility:** Design a low-cost, open-source platform that supports visually impaired and general users alike.
- **Real-Time Intelligence:** Incorporate AI-driven modules for object detection, facial recognition, and language translation with minimal latency.
- **Modular Architecture:** Separate visual, auditory, and communication modules for easier scalability and system upgrades.
- **Voice-Driven Interaction:** Enable fully hands-free operation using voice activation and feedback.
- **Seamless Communication:** Employ a client–server architecture where the Raspberry Pi communicates with a Flutter-based mobile app for AI inference and user configuration.

4.2.3 Functional Modules

The system integrates multiple AI-powered modules, each designed for a specific function:

- **Voice Assistant Module:** Listens for the wake phrase “Hey S.A.G.E” and processes commands using AI language models to interpret and execute user requests.

- **Object Detection Module:** Employs YOLO-based models to identify real-world objects and convey their identities through HUD visuals and audio feedback.
- **Facial Recognition Module:** Detects and identifies known individuals in real time, storing new faces for future reference.
- **Language Translation Module:** Converts captured text into translated speech using APIs like Google Vision and LibreTranslate.
- **Navigation Module:** Integrates with Google Maps APIs to guide users safely through spoken directions and HUD-based route visualization.

4.2.4 System Architecture and Workflow

The S.A.G.E ecosystem distributes computational tasks between the wearable device, the mobile companion application, and cloud-based AI backends:

1. The **Raspberry Pi Zero 2 W** handles lightweight on-device processing, voice activation, and HUD rendering.
2. The **mobile application** manages heavy AI workloads, network configuration, and API-based communication.
3. The **cloud backend** performs advanced inference tasks such as facial recognition and translation using high-compute models.

This distributed processing model ensures optimal performance, reduced latency, and extended battery life while maintaining consistent communication across all components.

4.2.5 Key Advantages

- Real-time environmental understanding through integrated vision and voice processing.
- Lightweight and modular design ensuring comfort and thermal efficiency.
- Open-source architecture for scalability and research adaptability.
- Enhanced accessibility through multimodal (visual + auditory) feedback.
- Efficient workload distribution between local and cloud resources.

5 Software Requirements Specification (SRS)

5.1 Introduction

5.1.1 Purpose

The purpose is to outline the complete functional and non-functional requirements of the **S.A.G.E (Situational Awareness and Guidance Engine)** - a personalized, AI-powered smartglass designed to enhance accessibility and real-world interaction. The system integrates embedded computing, computer vision, natural language processing, and mobile-based AI assistance to provide users with contextual awareness through both visual and auditory feedback. It aims to deliver real-time assistance, object recognition, translation, and navigation support in a compact, wearable form.

5.1.2 Intended Audience and Reading Suggestions

It is intended for engineers, developers, and researchers involved in the design, development, and testing of the S.A.G.E smartglass system. It provides detailed insights into hardware and software requirements, interfaces, and design constraints.

Types of audience:

- **Developers:** To implement, integrate, and test individual modules of the S.A.G.E ecosystem.
- **Researchers:** To extend S.A.G.E capabilities in computer vision and accessibility domains.
- **Users and Evaluators:** To understand feature functionality and system operation.
- **Design Engineers:** To ensure alignment with ergonomic, thermal, and hardware design specifications.

5.1.3 Project Scope

S.A.G.E focuses on bridging the gap between affordability, accessibility, and intelligence in wearable assistive technology. It enhances real-world interaction by providing on-demand AI-powered features such as voice assistance, object and facial recognition, translation, and navigation. The system uses a Raspberry Pi Zero 2 W as its core processor, integrated with a TFT HUD and camera module, while delegating advanced AI computations to a mobile companion app and cloud backend. This design ensures high performance, modularity, and extended usability without compromising portability.

5.1.4 Overview of Developer's Responsibilities

Developers are responsible for:

1. Designing modular and lightweight software for embedded deployment.
2. Implementing secure communication between the wearable device and mobile app.
3. Integrating AI APIs for visual recognition, translation, and conversational interaction.
4. Optimizing latency, thermal performance, and battery usage.
5. Ensuring user-friendly setup and configuration processes.
6. Maintaining compliance with safety and data privacy standards.

5.2 Overall Description

5.2.1 Product Perspective

S.A.G.E integrates hardware, embedded software, and mobile AI systems to provide an intelligent wearable platform. The Raspberry Pi serves as the control unit, HUD rendering, and communication. The mobile companion app acts as an AI hub, handling heavy processing, API communication, and real-time data exchange. Together, they form a hybrid client–server architecture capable of delivering contextual insights with minimal latency.

5.2.2 Product Functions

- Recognize objects, faces, and text using AI-driven computer vision.
- Translate captured text and speech between languages in real time.
- Display navigation and recognition data through the HUD.
- Accept and execute voice commands for hands-free interaction.
- Synchronize seamlessly with the mobile application for cloud-based inference.
- Maintain user profiles and preferences for customized experiences.

5.2.3 User Classes and Characteristics

- **End Users:** Visually impaired or general users seeking intelligent, hands-free assistance.
- **Developers:** Engineers integrating AI and IoT components.
- **Administrators:** Responsible for maintaining backend servers, APIs, and system updates.

5.2.4 Technological Stack

Operating System: Python based FastAPI server hosted on device

Programming Languages: Python 3.9+, Dart (Flutter)

Development Environments: Visual Studio Code

AI Frameworks and APIs:

- **TensorFlow Lite / OpenCV:** For real-time object and facial recognition.
- **Google Vision API:** For text extraction from images.
- **LibreTranslate API:** For multilingual translation services.
- **Gemini API:** For voice understanding and natural conversation.

Backend: Python-based FastAPI server hosted on Raspberry Pi handling local processing and communication with the mobile app.

Frontend: Flutter mobile application for Android handling AI API calls, preferences, and device management.

Database: SQLite for local storage; Firebase or PostgreSQL for cloud-based logs and preferences.

Version Control: GitHub for maintaining source code and version history.

5.2.5 General Constraints

- Limited processing capacity of the Raspberry Pi Zero 2 W.
- Power constraints due to battery-based operation.
- Dependence on stable Wi-Fi connectivity for cloud interaction.
- Need for thermal management in continuous AI operation.

5.2.6 Operating Environment

- **Hardware:** Raspberry Pi Zero 2 W, camera module, microphone, speaker, and TFT HUD.
- **Software:** Python server for embedded processing, Flutter app for mobile interface.
- **Network:** Operates through Wi-Fi hotspot connection from mobile device.

5.2.7 Design and Implementation Constraints

- AI APIs require internet connectivity.
- Power consumption must not exceed safe wearable thresholds.
- Codebase must remain modular to accommodate updates.
- UI must ensure accessibility for visually impaired users.

5.2.8 User Documentation

- Step-by-step setup and calibration guide.
- Mobile app usage manual and configuration instructions.

- Troubleshooting and FAQ section for connectivity and recognition issues.
- Maintenance guide for hardware and firmware updates.

5.2.9 Assumptions and Dependencies

Assumptions:

- The device will have stable power supply via battery or power bank.
- Users will pair the wearable device with the companion mobile app.
- Cloud APIs remain accessible and functional.

Dependencies:

- Dependence on third-party AI APIs for vision and translation.
- Wi-Fi connectivity for backend communication.
- Hardware reliability of display units.

5.3 External Interface Requirements

5.3.1 User Interfaces

The S.A.G.E interface combines auditory and visual interaction for a seamless user experience.

- **HUD Display:** Presents live object labels, translation outputs, and navigation cues.
- **Mobile App:** Offers controls for feature toggling, preferences, and AI model management.
- **Voice Interface:** Allows natural conversation with the assistant via microphone and speaker.

5.3.2 Hardware Interfaces

- Raspberry Pi interfaces with camera via CSI, HUD via SPI, and microphone via USB.

- External speaker provides real-time auditory feedback.
- Mobile device connects wirelessly over Wi-Fi or BLE for synchronization.

5.3.3 Software Interfaces

- REST APIs for communication between Raspberry Pi and mobile app.
- Integration with Google Vision, Gemini, and LibreTranslate APIs.
- Local Python scripts for image processing and system control.
- Flutter modules for user interface and cloud integration.

5.3.4 Communications Interfaces

- Communication occurs via TCP/IP over Wi-Fi hotspot.
- HTTPS ensures secure transmission of visual and voice data.
- Bluetooth Low Energy supports initial pairing and device discovery.

5.4 Hardware and Software Requirements

5.4.1 Hardware Requirements

Wearable Device:

- Raspberry Pi Zero 2 W as primary processor.
- 5MP or higher Pi camera module for object and face recognition.
- TFT HUD for visual feedback.
- USB microphone and mini speaker for voice I/O.
- Power bank (5000 mAh) for continuous operation.

Mobile Device:

- Android smartphone with Wi-Fi hotspot and BLE capability.
- Minimum 4GB RAM and quad-core processor.

5.4.2 Software Requirements

Device Software:

- Required libraries: OpenCV, FastAPI, PyAudio, TensorFlow Lite.
- Custom firmware for camera, display, and power management.

Mobile Application:

- Flutter SDK 3.0+ for Android.
- Integration with Google APIs, Gemini, and LibreTranslate.
- Firebase for data synchronization and logging.

5.5 Functional Requirements

5.5.1 Functional Requirement 1 - Voice Activation and Command Handling

1. The system continuously listens for the wake phrase "Hey S.A.G.E".
2. On activation, it captures and interprets commands for object detection, translation, or navigation.
3. Responses are provided both through voice output and HUD visualization.

5.5.2 Functional Requirement 2 - Object and Facial Recognition

1. Captures live frames through the camera and processes them using AI models.
2. Displays recognized objects or faces on the HUD.
3. Announces recognized entities through audio feedback.

5.5.3 Functional Requirement 3 - Language Translation

1. Extracts text from captured images using Google Vision.
2. Translates the extracted text through LibreTranslate API.
3. Outputs translated text visually and audibly.

5.5.4 Functional Requirement 4 - Navigation Assistance

1. Accepts voice-based navigation requests.
2. Utilizes Google Maps API to generate routes.
3. Displays real-time directions on HUD and provides voice instructions.

5.6 Non-Functional Requirements

5.6.1 Performance Requirements

- Average response time below 2 seconds for AI-based queries.
- System operational for 3-5 hours on a full charge.
- Minimal heat dissipation for safe wearable usage.

5.6.2 Safety Requirements

- Operates within safe temperature thresholds for skin contact.
- Power circuits designed for short-circuit and overvoltage protection.
- Data transmission follows privacy and security protocols.

5.6.3 Security Requirements

- Implements encrypted communication between device and mobile app.
- User credentials securely stored using hashed tokens.
- Access to facial and object recognition data is privacy-compliant.

5.6.4 Software Quality Attributes

- **Reliability:** Consistent performance across long durations of use.
- **Scalability:** Allows integration of new AI models or hardware modules.
- **Maintainability:** Modular design ensures easy updates and debugging.

- **Usability:** Hands-free operation for accessibility and safety.
- **Efficiency:** Optimized task distribution minimizes power and latency.

6 Software Design Document (SDD)

6.1 Introduction

The Software Design Document (SDD) provides a detailed description of how the S.A.G.E system is structured, organized, and implemented. It bridges the requirements defined in the SRS and the actual system realization by elaborating on architecture, data design, module interactions, and algorithmic logic. The SDD acts as a technical guide for developers, integrators, and testers, ensuring consistent understanding of each subsystem and its interactions.

6.1.1 Purpose

The main objectives of the SDD are:

- To define the internal architecture and software workflow of the S.A.G.E system.
- To describe the functional division of modules and their interactions across device, mobile, and cloud layers.
- To document algorithms governing AI modules including object detection, facial recognition, translation, and navigation.

6.1.2 Overview

The system design adheres to the following software engineering principles:

- **Modularity** – Independent modules for vision, translation, voice, and display.
- **Scalability** – Ability to integrate new AI services and features seamlessly.
- **Efficiency** – Distributed processing between wearable device and mobile backend.

- **Reliability** – Continuous operation with minimal latency and high responsiveness.
- **Maintainability** – Simplified debugging and future upgrades through clear modular boundaries.

6.2 System Architecture

The S.A.G.E system is designed using a distributed client–server model. The Raspberry Pi-based smartglass acts as the client, while a companion mobile application serves as the server and AI-processing hub. Cloud APIs extend computational support for advanced inference tasks like translation and face recognition.

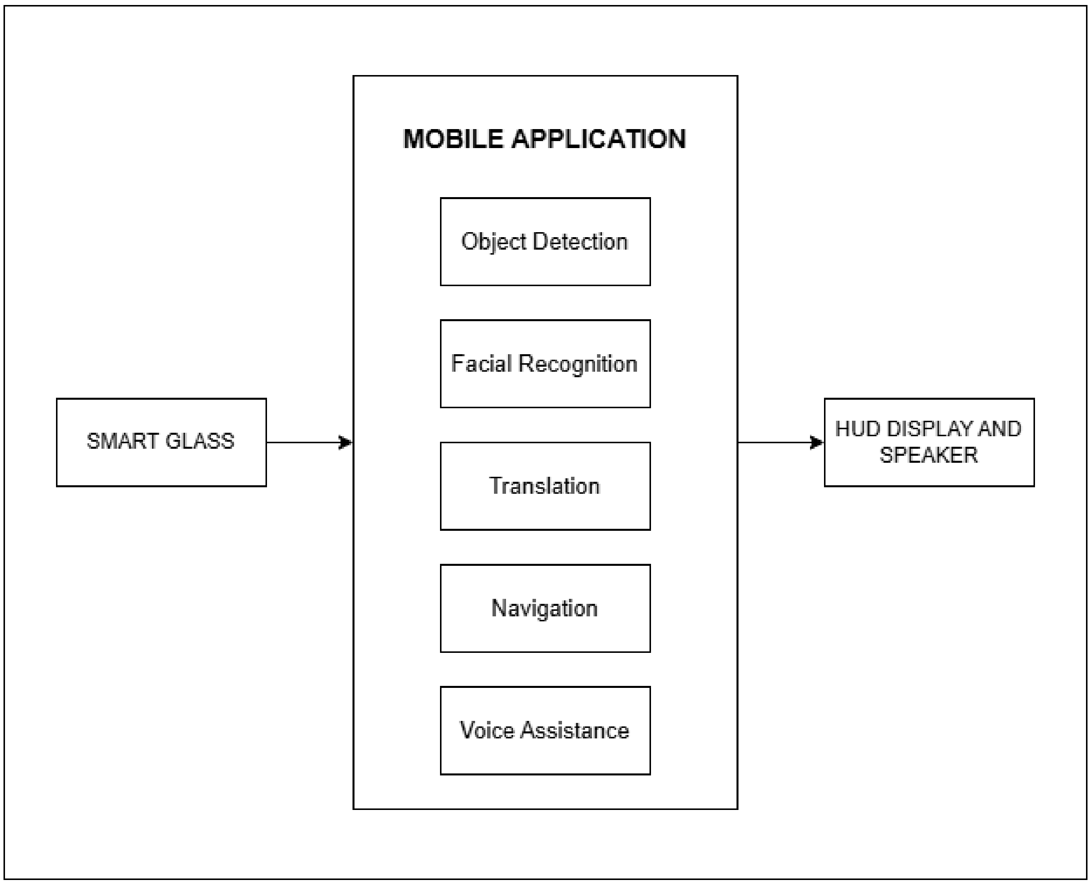

Figure 6.1: System Architecture of S.A.G.E

6.3 Use Case diagram

The Use Case Diagram of the S.A.G.E system depicts interactions between the user, smartglass, mobile application, and cloud services. The Raspberry Pi–based smartglass captures environmental data and sends it to the mobile app for processing and control.

Advanced tasks like translation and face recognition are handled through cloud APIs, ensuring smooth coordination and intelligent assistance.

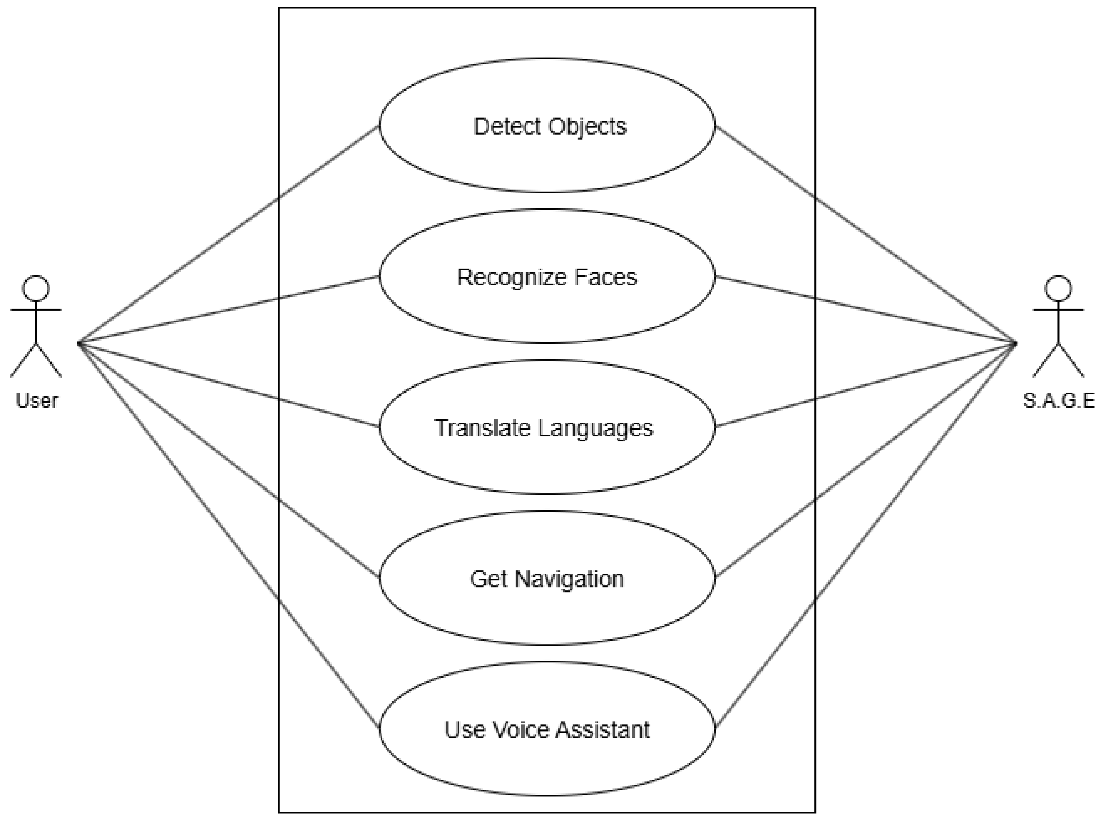

Figure 6.2: Use Case diagram of S.A.G.E

6.4 Data Flow Diagrams

6.4.1 Level 0

At Level 0, S.A.G.E is represented as a single unified process that interacts with external entities such as the **User**, **Environment**, and **Cloud AI Services**. The system receives inputs in the form of user commands (voice prompts) and real-world sensory data (camera feed, GPS, and microphone). These inputs are processed internally by S.A.G.E and result in outputs such as Heads-Up Display (HUD) visuals, audio responses, and real-time navigation guidance.

Inputs:

- Voice commands from the user.
- Environmental visuals captured by the camera.

- GPS coordinates for location awareness.

Outputs:

- Object identification and location alerts through audio.
- Navigation and direction guidance via voice prompts and HUD overlay.
- Translated text displayed on the HUD or read aloud through the audio module.

External Entities:

- **User:** Provides commands and receives real-time feedback.
- **Environment:** Supplies visual and spatial data captured.
- **Cloud AI Services:** Performs advanced inference tasks such as facial recognition and language translation.

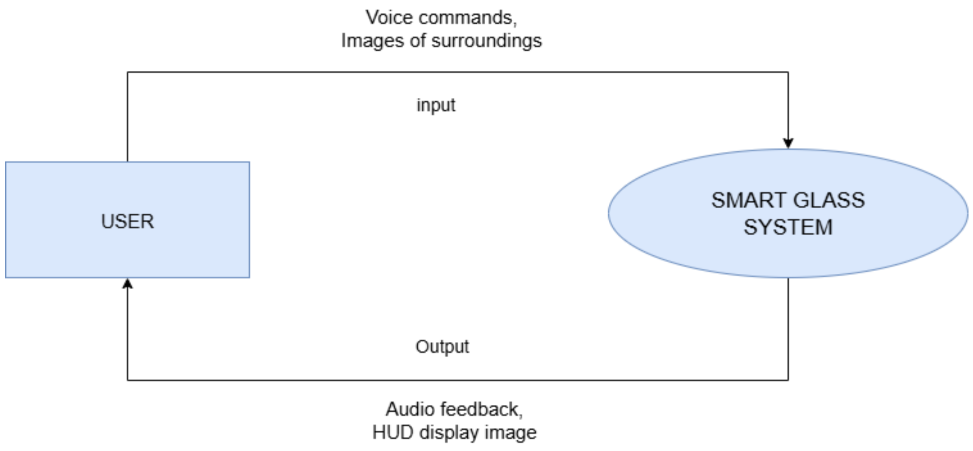

Figure 6.3: Level 0 DFD – Overview of S.A.G.E

6.4.2 Level 1

The Level 1 DFD decomposes the S.A.G.E system into its major functional modules: **Voice Assistant**, **Object Detection**, **Facial Recognition**, **Translation**, and **Navigation**. Each module communicates through the local server and interacts with both the mobile companion app and cloud-based AI services.

Data Flow Description:

- The **User** initiates commands through the voice assistant interface.
- Audio commands are interpreted by the **Voice Recognition Module**, which classifies intents and triggers the relevant subsystem.
- The **Object Detection Module** analyzes the camera feed and provides identified objects as tagged metadata.
- The **Facial Recognition Module** sends visual data to the cloud API for recognition, returning identified names or roles.
- The **Translation Module** handles text or speech conversion and communicates the output to the user through audio or visual channels.
- The **Navigation Module** uses GPS data to fetch real-time route information from Google Maps APIs and provides stepwise instructions.
- All modules share processed data with the **Local Python Server**, which synchronizes responses with the **Mobile Application**.

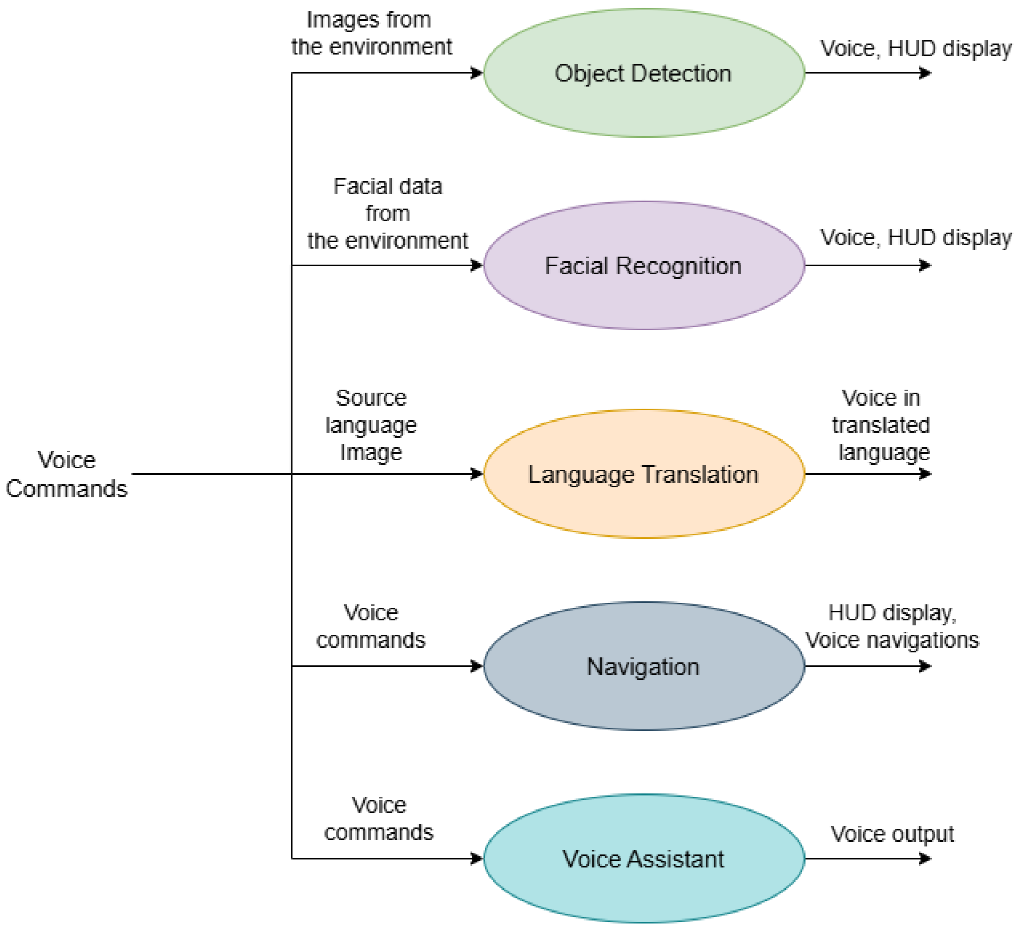

Figure 6.4: Level 1 DFD – Core Functional Modules of S.A.G.E

6.4.3 Level 1.1 – Object Detection Module

The Object Detection Module identifies and labels objects within the camera's field of view using the YOLO (You Only Look Once) deep learning model. Captured frames are transmitted to the connected mobile application for real-time processing. Detected objects are displayed on the S.A.G.E. Head-Up Display (HUD) and announced through audio feedback, ensuring accessibility for visually impaired users. This module enhances situational awareness and helps users recognize and interact safely with their surroundings.

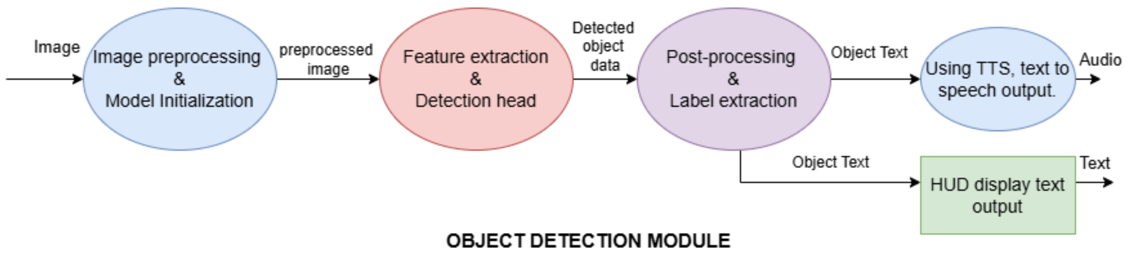
Image
Image preprocessing & Model Initialization
preprocessed image
Feature extraction & Detection head
Detected object data
Post-processing & Label extraction
Object Text
Using TTS, text to speech output.
Audio
Object Text
HUD display text output
Text
OBJECT DETECTION MODULE

Figure 6.5: Level 1.1 DFD – Object Detection Module

6.4.4 Level 1.2 – Facial Recognition Module

The Face Recognition Module is responsible for identifying and recognizing individuals in real time. It captures facial images through the camera, converts them into unique feature vectors, and compares these vectors with stored entries in the database to determine identity matches. When a known individual is recognized, the system announces their name through the audio output, providing immediate feedback to the user. In cases where a face is not recognized, the module securely

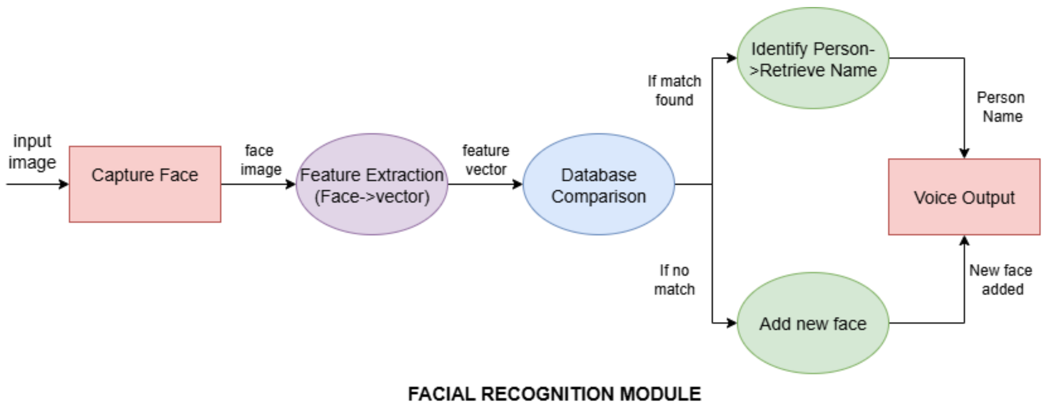

Figure 6.6: Level 1.2 DFD – Facial Recognition Module

6.4.5 Level 1.3 – Translation Module

The Translation Module enables real-time translation of textual or visual content captured from the environment. It detects and extracts text from the camera feed using the Google Vision API, accurately identifying written information from signboards, labels, or documents. The extracted text is then sent to the LibreTranslate API for efficient language translation into the user's preferred language. To ensure seamless accessibility, the translated text is converted into voice output, allowing users to comprehend the information audibly. This module greatly assists users in understanding signboards, instructions, and other written materials in unfamiliar languages, thereby enhancing situational awareness and communication in diverse environments.

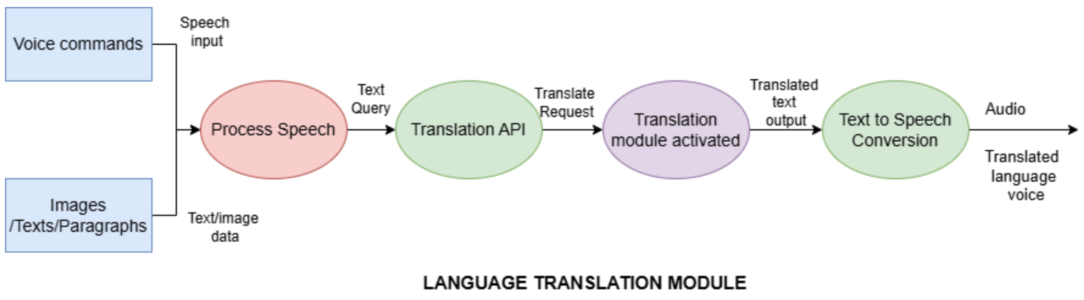

Figure 6.7: Level 1.3 DFD – Translation Module

6.4.6 Level 1.4 – Navigation Module

The Navigation Module provides real-time route guidance and navigation assistance to the user. It integrates seamlessly with the Google Maps API to generate accurate and optimized routes based on the user's current location and destination. The live directions are displayed on the S.A.G.E. Head-Up Display (HUD), ensuring that users can follow routes without distractions. Additionally, the module offers hands-free voice guidance, allowing users to receive turn-by-turn instructions audibly for safe and convenient navigation, particularly useful during movement or when visual attention is limited.

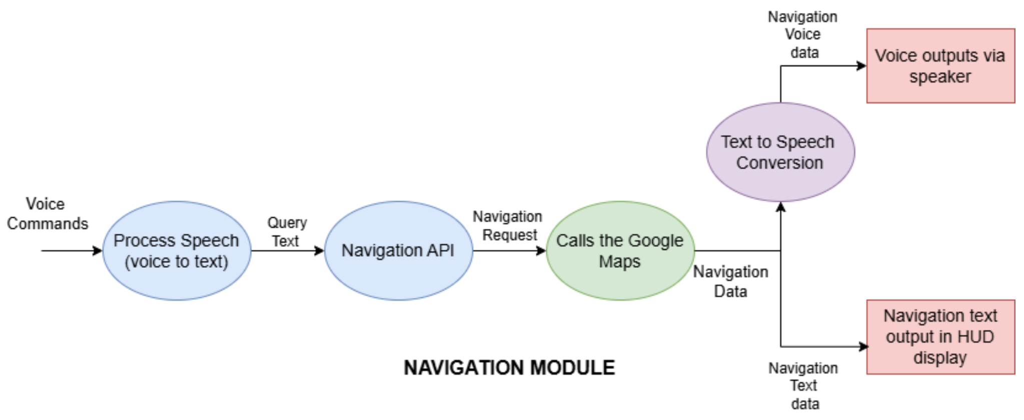

Figure 6.8: Level 1.4 DFD – Navigation Module

6.4.7 Level 1.5 – Voice Assistant Module

The Voice Assistant Module enables voice-based interaction between the user and S.A.G.E., allowing seamless communication through natural speech. It activates using a wake phrase such as "Hey S.A.G.E." and processes voice commands using the Gemini API. This module provides conversational responses, executes control actions, and supports hands-free operation to enhance accessibility for users. It continuously listens for user input, interprets spoken commands through speech-to-text conversion, and performs intent recognition using natural language processing to route them to the appropriate subsystem.

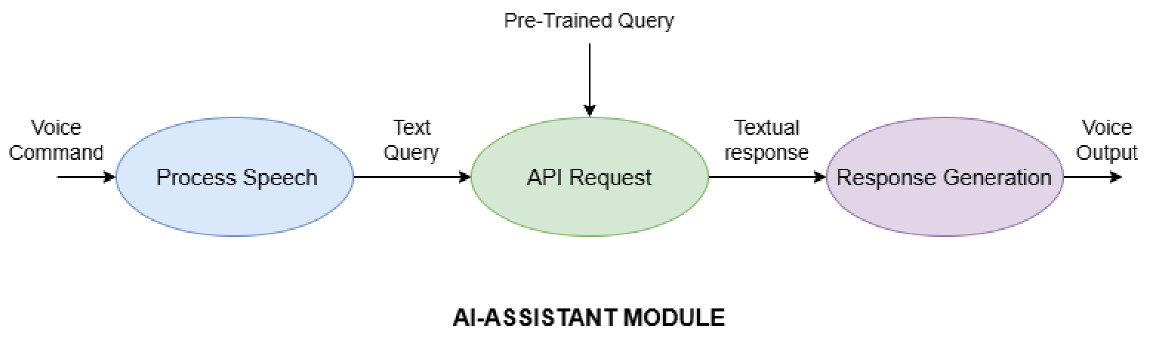

Figure 6.9: Level 1.5 DFD – Voice Assistant Module

6.5 Modules

The S.A.G.E software architecture consists of five main functional modules, all managed by the embedded controller on the Raspberry Pi. Each module operates autonomously but communicates through a shared data bus and message queue system.

6.5.1 Voice Assistant Module

Description: Enables hands-free interaction through voice commands. Detects the wake phrase "Hey S.A.G.E" and processes user queries or control requests using the Gemini API for intelligent responses.

Algorithm 1: Voice Assistant Module Algorithm

```
Input: Voice commands captured through the microphone
Output: Conversational response or system control action
Initialize microphone and wake phrase detector;
while voice assistant is active do
    if wake phrase "Hey S.A.G.E" is detected then
        Record and process the user command;
        Send the command to the Gemini API for NLP processing;
        Receive the generated response or control instruction;
        Provide conversational voice feedback or execute control action;
    end
end
return AssistantResponse;
```

6.5.2 Object Detection Module

Description: Identifies and classifies real-world objects in real time using the camera feed. Utilizes YOLOv11 or TensorFlow Lite for detection and provides results via audio and visual feedback.

Algorithm 2: Object Detection Module Algorithm

```
Input: Real-time visual feed from the camera
Output: Detected objects with audio and visual feedback
Initialize camera and mobile application connection;
while camera captures frames do
    Transmit image frame to the connected mobile application;
    Load YOLOv11 model for real-time object detection;
    Perform detection on each frame and enhance image quality;
    if objects are detected then
        Display object labels on the HUD;
        Convert detection results to audio feedback using TTS;
    end
end
return DetectedObjectsResponse;
```

6.5.3 Facial Recognition Module

Description: Detects and recognizes individuals by comparing live facial data with stored embeddings. Provides instant identity feedback through voice output for user awareness.

Algorithm 3: Facial Recognition Module Algorithm

```
Input: Live facial images captured through the camera
Output: Identified or newly registered face with audio feedback
Initialize camera and face recognition system;
while camera captures a face do
    Convert captured face to a feature vector;
    Compare the feature vector with stored database entries;
    if match found then
        Announce the person's name and description via audio output;
    end
    else
        Store the new face as a new database entry;
    end
end
return RecognitionResponse;
```

6.5.4 Language Translation Module

Description: Translates captured text or speech into the user's preferred language. Uses Google Vision API for OCR and LibreTranslate for real-time multilingual translation with audio feedback.

Algorithm 4: Language Translation Module Algorithm

```
Input: Visual or text-based content captured from the environment
Output: Translated text delivered via audio output
Initialize modules for voice and image input;
while text is detected do
    if voice input then
        Apply Speech-To-Text processing on the captured audio;
    end
    else if image input then
        Capture image snapshot;
        Send image to the mobile application for processing;
        Extract textual data using Google Vision API;
    end
    Send extracted text to LibreTranslate API for translation;
    Receive translated output (e.g., French to English);
    Deliver the translation through voice output on the speaker;
end
return TranslatedResponse;
```

6.5.5 Navigation Module

Description: Offers real-time navigation assistance using GPS and Google Maps API. Displays route details on the HUD and delivers step-by-step audio directions for accessibility.

Algorithm 5: Navigation Module Algorithm

```
Input: GPS data and map requests from the mobile application
Output: Route details displayed on HUD and audio navigation output
Initialize GPS and map communication module;
while navigation is active do
    Apply Speech-To-Text processing on user voice input;
    Pre-process the query and interpret destination command;
    Initialize navigation request through Google Maps API;
    Display real-time route and direction details on HUD;
    Output navigation guidance via audio for accessibility;
end
return NavigationStatus;
```

7 Future Scope

The S.A.G.E project provides a foundation for intelligent, accessible, and context-aware wearable systems. Its modular architecture enables extensive future enhancements that can further elevate user experience and operational efficiency.

7.1 Hardware Optimization

- Integration of lighter and more efficient hardware modules such as custom PCBs and AI accelerators like Coral TPU.
- Implementation of low-power chipsets for prolonged battery life and sustainable operation.
- Incorporation of advanced camera module to enhance recognition accuracy under diverse lighting conditions.

7.2 Enhanced AI Capabilities

- Deployment of edge-based deep learning models for on-device object and facial recognition.
- Real-time adaptation to user context using reinforcement learning techniques.
- Integration of predictive AI for environment anticipation and proactive assistance.

7.3 Augmented User Interaction

- Development of gesture and gaze-based control systems for touchless interaction.
- Integration with visual overlays for enhanced navigation and contextual awareness.

- Support for multilingual voice recognition to improve inclusivity across different demographics.

7.4 Software and Network Advancements

- Migration to 5G and edge-computing infrastructure to reduce latency in cloud-based AI tasks.
- Implementation of containerized microservices for modular deployment and easier scalability.
- Integration with mobile ecosystems for cross-platform compatibility and remote firmware updates.

7.5 Expanded Application Domains

- Adoption in industries such as healthcare, defense, logistics, and tourism.
- Development of specialized modules for emergency response, hazard detection, and indoor navigation.
- Enhancement of accessibility for differently-abled individuals through emotion recognition and intent prediction.

7.6 Research and Development Opportunities

- Exploration of brain-computer interfaces for direct cognitive command integration.
- Advancement in federated learning to maintain privacy in distributed AI model training.
- Collaboration with AI research institutions for dataset curation and model fine-tuning.

8 Appendix A - Glossary

AI – Artificial Intelligence, the field focused on simulating human intelligence processes through computational systems.

API – Application Programming Interface, a structured protocol allowing communication between software applications.

BLE – Bluetooth Low Energy, a wireless communication standard designed for low-power data exchange.

CNN – Convolutional Neural Network, a deep learning architecture used for image and visual recognition tasks.

DFD – Data Flow Diagram, a graphical representation showing data movement within a system.

ESP32 – A microcontroller module with Wi-Fi and Bluetooth capabilities used in IoT and embedded projects.

GPS – Global Positioning System, a satellite-based system that provides geolocation and time information.

HUD – Heads-Up Display, a transparent visual interface that projects data directly within the user's line of sight.

IoT – Internet of Things, a network of interconnected devices capable of exchanging data over the Internet.

ML – Machine Learning, a subset of AI that enables systems to learn and improve automatically from experience.

OCR – Optical Character Recognition, a process of converting printed or handwritten text into machine-readable data.

Pi Camera – A camera module designed for the Raspberry Pi platform, enabling real-time image and video capture.

REST API – Representational State Transfer API, a standardized architectural style for communication between client and server.

Raspberry Pi – A compact, single-board computer used for prototyping and embedded computing applications.

SAGE – Situational Awareness and Guidance Engine, a personalized AI-powered smartglass designed for accessibility and contextual assistance.

TTS – Text-To-Speech, a technology that converts written text into spoken words.

YOLO – You Only Look Once, a real-time object detection algorithm used in computer vision systems.

TensorFlow Lite – A lightweight version of TensorFlow optimized for edge and embedded devices.

Gemini API – A cloud-based AI service used for natural language understanding and contextual processing.

9 Appendix B - Acronyms

AI – Artificial Intelligence
API – Application Programming Interface
BLE – Bluetooth Low Energy
CNN – Convolutional Neural Network
DFD – Data Flow Diagram
ESP32 – Embedded System Processor 32-bit
GPS – Global Positioning System
HUD – Heads-Up Display
IoT – Internet of Things
ML – Machine Learning
OCR – Optical Character Recognition
Pi – Raspberry Pi
REST – Representational State Transfer
SAGE – Situational Awareness and Guidance Engine
SDD – Software Design Document
SRS – Software Requirements Specification
TTS – Text-To-Speech
YOLO – You Only Look Once
API – Application Programming Interface
ML Kit – Google's Machine Learning Kit for mobile AI tasks.
FTP – File Transfer Protocol
HTTPS – Hypertext Transfer Protocol Secure.
AIoT – Artificial Intelligence of Things.
GUI – Graphical User Interface.
Wi-Fi – Wireless Fidelity.

Bibliography

[1] V. Sathya, H. Preetha, and K. Gopika, *Smart Navigation Glasses for Independent Living*, IEEE, 2023. doi:10.1109/ICCCT63501.2025.11020288.

[2] S. Priyanka, A. S. Kumar, M. V. Nagabhushanam, D. Vennela, and P. D. Tulasi, *Smart Glasses for Visually Impaired People using Machine Learning*, IEEE, 2023. doi:10.1109/ICCCNT56998.2023.10307374.

[3] T. P. Rani, V. T. Vignesh, S. Susila, and P. Kalaichelvi, *Visual Information Translator Using Smart Glasses for Blind*, IEEE, 2023. doi:10.1109/ICCEBS58601.2023.10449230.

[4] G. Vinod Katkar, K. Abirami, A. Mary Posonia, B. Ankayarkanni, and D. Ushanandini, *Google Pi using Raspberry Pi for Visually Impaired*, IEEE, 2024. doi:10.1109/AIIoT58432.2024.10574549.

[5] R. Sabitha, S. Pandi, J. Rathi Devi, and G. Jegan, *Smart Glass for Visually Impaired Person*, IEEE, 2023. doi:10.1109/ICCCT63501.2025.11019089.

[6] A. Ghodake, H. Sale, A. Kamble, K. Mankari, O. Lad, and V. Mishra, *Facial Recognition Smart Glasses for Visually Impaired People*, IEEE, 2023. doi:10.1109/ICCUBEA58933.2023.10391977.

[7] M. Kumar, B. Bharti, and U. Chauhan, *Smart Glasses Embedded with Facial Recognition Technique*, IEEE, 2023. doi:10.1109/AISC56616.2023.10085337.

[8] P. J. Loresco, R. J. C. Cruz, K. Z. Ramones, J. A. D. Zafra, and K. R. G. Ramirez, *Indoor Navigation Glasses for the Visually Impaired with Deep Learning and Audio Guidance Using Google Coral Edge TPU*, IEEE, 2024. doi:10.1109/TENCON61640.2024.10902929.

[9] A. Malhotra, *Single-Shot Image Recognition Using Siamese Neural Networks*, IEEE, 2023. doi:10.1109/ICACITE57410.2023.10182466.