Grenze International Journal of Engineering and Technology, June Issue

VisionMate: A Comprehensive Mobile Assistive Application for Visually Impaired Individuals

Sheela N[1], VaniAshok[2], Anusuya M A[3] and Mohammed Faraz[4]
[1]SJCE, JSS STU/CS, Mysore, India
Email: sheela_cse@sjce.ac.in
[2-4]SJCE, JSS STU/CS, Mysore, India
Email: vanisj@sjce.ac.in, anusuya_ma@sjce.ac.in, mohammedfarazmandya@gmail.com

Abstract*— This paper presents VisionMate, a comprehensive mobile assistive application developed to support visually impaired individuals in performing everyday tasks independently. The primary objective of this work is to develop a unified, accessible solution that addresses the fragmentation and limitations of existing assistive tools. The application combines a wide array of accessibility focused features powered by the Gemini API, which serves as the core engine behind real-time image recognition, voice-enabled personal Chatbot interaction, multilingual text-to-speech (TTS), currency recognition, and smart navigation assistance. The image recognition module enables users to capture photos of their surroundings and receive meaningful voice responses based on spoken prompts. The Chatbot provides conversational support and assistance for personalized queries. The TTS module not only extracts text from images but also detects the language and delivers both the full content and a summarized version through speech. Currency recognition supports multiple international currencies, enhancing financial independence for users' globally. The application also offers live content magnification, allowing users with partial vision to enlarge camera feed visuals in real time according to their preferred text size. An SOS module facilitates emergency communication by sending messages and current location links via SMS and WhatsApp. For mobility assistance, the application utilizes Gemini to generate navigation URLs based on source and destination in walking mode, which are then opened directly in Google Maps for guided navigation. Additionally, the system provides contextual awareness by identifying the user's nearby address, notable landmarks, and current weather conditions. The prototype demonstrates the feasibility of integrating multiple assistive features into a single mobile platform and shows promising functionality during internal validation. With its voice guided interface, large on-screen buttons and multi feature integration, VisionMate demonstrates the potential of mobile AI to deliver inclusive, real-time support for the visually impaired.

***Index Terms*— Assistive Technology, Visually Impaired, Mobile Application, Gemini API, Voice Interaction, Context-Aware Services.**

I. INTRODUCTION

Visual impairment significantly limits an individual's ability to interact with their surroundings and access information.

Grenze ID: 01.GIJET.12.2.294_21

According to the World Health Organization, over 2.2 billion people globally suffer from some form of vision impairment with a large portion facing daily challenges related to mobility, reading, object identification, financial transactions and emergency response. Existing tools like screen readers or Braille devices offer isolated support, often needing special hardware and lacking real-world adaptability. Advances in mobile technology and AI now enable cost effective, integrated assistive systems via smartphones. Mobile devices are now equipped with advanced hardware and are capable of running AI-powered services, making them ideal platforms for delivering inclusive solutions to visually impaired individuals. Recent advances in mobile AI have enabled more comprehensive assistive solutions. Despite the growing work in this domain, most existing applications tend to be fragmented, targeting isolated functionalities such as object detection, text-to-speech conversion or digital magnifiers. Though helpful, many lack interoperability, forcing users to juggle multiple apps for varied support. Additionally, many tools rely on proprietary hardware or external devices, increasing their cost and reducing portability, especially in low resource settings. Another limitation commonly observed is the absence of contextual intelligence. Most applications provide raw outputs without interpreting or summarizing information based on the environment or user queries. Voice control remains basic, lacking the conversational depth needed for intuitive use. Moreover, critical use cases such as emergency alerts, real-time scene understanding or navigation are frequently overlooked or implemented in isolation. As a result, visually impaired users are forced to rely on multiple applications, each with its own learning curve and accessibility challenges. There is a critical need for a unified, intelligent, and affordable assistive mobile platform that integrates vision, voice and contextual understanding to support visually impaired users in navigating diverse real-world scenarios independently.

Recognizing these limitations, there is a growing need for unified platforms that provide intelligent multimodal support through a single interface. Such platforms should combine visual recognition, conversational AI, environmental awareness and voice-guided interaction while remaining lightweight, cost-effective and easy to use. This paper proposes Vision- Mate as a step towards that direction, integrating multiple AI powered capabilities into a single mobile application tailored specifically for the visually impaired community. To address these challenges, this paper presents VisionMate, a feature rich mobile application specifically designed to support visually impaired individuals in navigating a wide range of daily activities through an integrated, voice-interactive platform. Unlike single function tools, VisionMate unifies many features for seamless, all-in-one interaction. The application leverages the Gemini API [10], a break- through multimodal foundation model that goes beyond traditional CNN-based recognition techniques. Unlike conventional models that rely on handcrafted pipelines for image classification or text extraction, Gemini processes both visual and textual data together to generate contextual and intelligent responses. This allows VisionMate to understand user prompts more accurately and deliver richer, situation aware outputs. Powered by Gemini, the application supports real-time image recognition, natural language-based Chatbot interaction, multilingual text-to-speech with summarization, currency identification across international denominations and face recognition using stored image data for identifying known individuals. The system also incorporates essential mobility and safety features, including SOS messaging with live location links shared via SMS and WhatsApp and the ability to retrieve nearby landmarks, addresses and weather details for situational awareness. It also integrates Google Maps navigation by generating walking routes via Gemini, allowing a smooth transition from scene understanding to guided movement. The accessible interface includes large buttons, voice prompts and minimal touch to support low or no vision. By offering a wide spectrum of support from reading printed material to locating landmarks and responding to emergencies, VisionMate serves as a unified, intelligent assistant that enhances autonomy, convenience and confidence for visually impaired users in real-world environments.

II. Related Work

Assistive technologies aimed at supporting the visually impaired have advanced significantly in recent years, largely due to progress in computer vision, speech technologies and mobile computing. Numerous systems have attempted to provide environmental awareness, reading support and object detection. However, most existing solutions are single-purpose, lack deep contextual understanding or rely on external hardware and limited AI capabilities. Several commercial applications have been introduced to help visually impaired individuals to navigate daily life. Microsoft's Seeing AI provides scene and text reading capabilities but is largely rule-based and lacks the flexibility to understand user-specific queries. Google Lookout offers object and text recognition but is task-specific and lacks integration across different use cases. Be My Eyes [20] connects users with sighted volunteers via live video calls, which introduces dependency on third-party human support and active internet connectivity. Sree Sindhura and Jaiswal (2019) proposed a YOLO-based object detection system offering voice

feedback for blind users [1]. While the model provided real-time recognition using the MS COCO dataset, it lacks contextual awareness and do not relate objects meaningfully within a scene. Shifa Shaikh et al. (2020) developed an Android-based object recognition system with basic auditory cues [2]. However, the solution was limited to individual object identification and lacked integration with other assistive functions. D Ghule et al. (2022) presented a multilingual object- to-audio converter using Microsoft Cognitive Services and Amazon DynamoDB [3]. Though it enabled scene narration, the system depended heavily on internet access and failed to incorporate features like face recognition or currency detection. K. Singh et al. (2024) introduced a CNN-based system combining OCR and currency recognition to assist visually impaired users by identifying denominations and extracting printed or handwritten text [8]. Joshi et al. (2021) introduced a real-time mobile platform for object detection but noted difficulties maintaining accuracy across distances and in noisy environments [4]. Ketkale et al. (2021) proposed a cost-effective mobile solution using CNN and android for object detection, it offered no support for personalization or complex scene descriptions [5]. Dileep Reddy Bolla et al. (2023) introduced a Mobile Net based object detection and localization system with family tracking. While optimized for low-power devices, the system avoided advanced AI models and did not support multi- functional interaction or scene-level understanding [6]. Ganesh et al. (2019) used Tesseract OCR on a Raspberry Pi to convert text into speech, but their solution was hardware-dependent and sensitive to image quality, making it unsuitable for mobile applications in uncontrolled environments [7]. Across these works, a key limitation is the lack of integration between multiple assistive features and the absence of multimodal AI, capable of understanding relationships between objects, summarizing scenes and dynamically responding to user voice prompts. While some solutions rely on datasets for object detection, they do not provide conversational, personalized feedback or features like SOS messaging, face recognition or currency identification. VisionMate, by contrast, utilizes the Gemini API's multi- modal capabilities to go beyond static object detection. It generates detailed environmental interpretations based on image context and user queries. The system integrates scene understanding, multilingual TTS with summarization, personalized Chatbot, face and currency recognition and safety features such as emergency alerts to their known ones, all under a single voice-driven mobile platform, setting a new benchmark for intelligent mobile-based assistive systems.

III. METHODOLOGY

The architecture of the VisionMate application is structured into three primary layers as depicted in Fig.1.

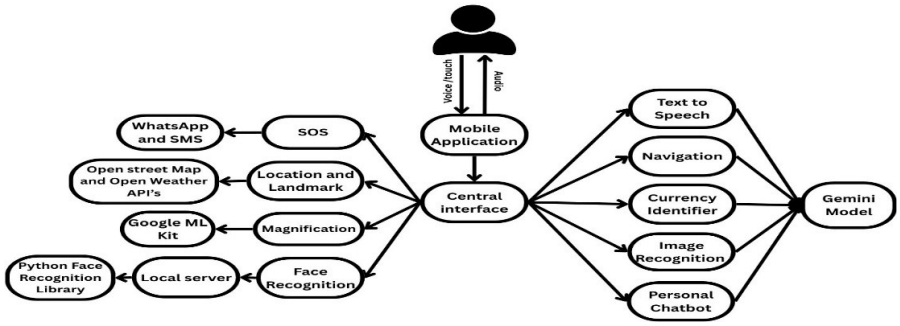

Fig. 1. System Architecture of VisionMate Application

The User Interaction Layer, Mobile Application Layer and Cloud Service and API Layer.

The layered design provides the ability to integrate assistive capabilities in a modular manner without compromising real-time responsiveness and scalability of the platform. The layers have unique functions, with each capable of processing the inputs, communicating with the cloud intelligence, and providing contextual and voice based output to the visually impaired user.

The mathematical representation of the data flow between the Cloud Service Layer, Mobile Application Layer, and User Interaction Layer may be presented as follows.

$$O=f(G(V,U,S)) \quad (1)$$

O is the synthesised output (voice or visual feedback), G is the processing function within the Gemini API, V is the visual input (image captured), U is the user input (speech or text) and S is the system context information (GPS location or weather). Function f (.) represents the post-processing conversion of the multimodal output of Gemini into a response that is readily available.

A. User Interaction Layer

This is the layer that creates an interface between the user and the system. It has an easier visual design, huge, high contrast buttons and is optimized to the voice and gesture interaction. Issuing prompts is handled by voice input, confirmation/questioning of the system, and yes/no responses can be made through gestures like the double tap to repeat audio or a single and a double tap, though some users may not be able to use touch or voice as the primary input methods. The interaction layer will be the entry point of all the core modules, such as image recognition, Chatbot, TTS, magnification, currency and face recognition, SOS alerts, navigation and location-aware services.

B. Mobile Application Layer

It is a layer which is applied on the Flutter framework and focuses on operations on a device level, rendering, input and local data routing. It is linked to in-board sensors and services such as the camera, microphone, GPS and text-to- speech engine. Certain most essential tasks include:
Multimodal Input Capture: It enables the users to capture pictures, voice input or gestures. Text-to-Speech Module: Converts text and summaries are presented in terms of audios. Magnification Module: This module broadcasts and opens the zoom on the camera in real time based on the user preferences. Emergency Module: Dispatches messages that contain a location option either through SMS or WhatsApp. Navigation Handler: Opens the Google map in walk-mode depending on source-destination URLs which are generated on source-destination prompts. Location and Weather Fetcher: With the help of GPS positions, the application finds the places close by, address estimates, and live weather data. The mobile application is a middleware, which wraps the user inputs and state of the system and is communicated to the third-party AI services.To address data privacy concerns, the application is designed to minimize external data retention

C. Cloud Service and API Layer

This layer provides the intelligence that is powered by artificial intelligence that drives the assistive feature of the app. It consists of its essence the Gemini API [10], multimodal interpretation with user voice/text prompts, the images are mixed with them. The most important functions that are processed in this layer are:
Contextual Image Understanding: It identifies objects and their geometrical location in a scene. Conversational Chatbot: Responds to natural language queries. Language-Sensitive Text Summarization: Summarizes and extracts text. Navigation URL Generation: Voice-specified source and destination are transformed to walk-mode Google Maps URLs. It is also possible to interconnect with other external APIs to fetch some of the environmental information such as weather based on location and other landmarks. The feedback of these results is sent back to the mobile layer and processed into voice responses and sent to the user through audio feedback. The strategy in the design of the visual aid app was geared towards the foundation of the modular development which was motivated by the accessibility and refined with repeat usage. Each of the components was to operate independently and also contribute to the development of a complete system, which is a real-time support system that the visually impaired users can use. Each of the items in the system such as the navigation, face recognition, SOS alerts, and text reading was brought in, brought together, and tested in that order. The modular structure offered scalability, simplified the debugging process as well as feature specific improvements.

D. Technology Stack for the application

Frontend: Built on the Flutter platform and the Dart programming language that allows it to be deployed on multiple platforms. Backend: This is written in Python, to handle face recognition and data processing. The app and the backend communicated through the Web Socket and HTTP protocol. APIs and Libraries: Face recognition and text recognition with Google ML Kit [11] [9]. Image understanding, contextual scene interpretation, and intelligent Chatbot interactions are performed on a multimodal foundation model called Gemini API [10]. Openweathermap [13] and openstreetmap APIs [12] of real time weather and geolocation of landmarks based on location. Flutter TTS [14] and STT [15] packages of speech- based interaction. Data Storage: Local: Emergency contacts and user preferences stored using shared preferences [19].

E. Voice and Gesture Interaction

The interaction model is based on voice-based input and feedback. Speech to text feature identifies user query and commands, whereas text to speech feature allows audio response to the system. Gesture interactions are provided with the use of a double-tap to activate commands and customizable neuromorphic buttons with physical feedback. Voice fallback and error prompts increase the level of accessibility. The performance of the speech recognition module was evaluated using the following expression:

$$A_{stt} = \left(\frac{N_c}{N_t}\right) \times 100 \quad (2)$$

A_{stt} Represents the accuracy of the Speech-to-Text module, N_c is the number of correctly recognized voice commands, and N_t is the total number of commands issued by the user.

F. User Interface Design

The interface has been made user-friendly to the visually impaired by: Themes of high contrast and dark backgrounds and light text. Fonts adjusted to a better size. A tactile cue neuromorphic button. Visual dependency minimization by voice-first navigation model. Alerts, error and confirmations speech feedback synthesized.

IV. System Implementation

A. Face Recognition

Image is automatically generated and forwarded through HTTP to a local backend server to compare to faces already stored in it. If a match is found, the identified person's name is displayed on screen and announced using the flutter tts package [14]. New faces can be added by tapping the '+' button, which initiates a guided process: the user captures an image, and the system checks for the presence of whether none, multiple or a single face is detected using the Google ML Kit's Face Detection API [11]. If a single face is identified, the system asks for the person's name and stores the labelled face data via HTTP on the local server.

B. Currency Recognition

Captured images of currency notes are processed through the Gemini API [10] to identify denominations. Responses are converted into audio using the flutter tts package [14]. Voice prompts explain the process to users, and low quality images are handled in terms of errors.

C. Image Recognition

The Gemini API [10] is an integration into this module to analyze the images captured by users. The voice prompts are given by the users through the speech to text package to the API where both the image and the prompt are sent. Synthesized speech is used to provide the description, and the neuromorphic buttons are used to improve the interaction with the tactile sense and errors in order to use it smoothly. The Gemini API image perception process is a probabilistic model as shown by the following:

$$C = arg\ max_{\{i\}} P\left(L_{\{i\}} | I\right) \quad (3)$$

'C' represents the last predicted object or the label of the final label in the prediction, $L_{\{i\}}$ represents all the possible labels in the trained data, and I is the captured image. To make sure the contextual interpretation is correct, the API chooses the highest probability label probability P ($L_{\{i\}}$ |I)).

D. Text Read

Text in captured images is extracted using Google ML Kit's Text Recognition API and displayed with magnifiable fonts. The user can navigate to a text-to-speech screen, where the content is read aloud using flutter tts [14]. Large buttons and voice instructions ensure accessibility.

E. Text-to-Speech

Images containing text are analyzed using the Gemini API [10] to extract and summarize content. The detected text and summary are read out using flutter tts [14], language detection is also available. Interaction is supported through tactile inputs.

The overall process of text extraction, summarization, and speech generation can be expressed as

$$S_{out} = TTS(SUM(T_d)) \quad (4)$$

Where T_d is the detected text, SUM (·) denotes the summarization process applied to T_d , and TTS (·) represents the text-to-speech synthesis producing the final audio output S_{out}.

F. Navigation Assistance

Using the Geolocator [16], this module fetches the user's location. Voice input is handled through speech to text [15] for capturing destination and navigation is launched via a Google Maps URL which is given by the Gemini

API [10] after analyzing source and destination. Audio prompts assist the user to reach destination after it is redirected to Google Maps in walk mode. Using the Geolocator [16], this module fetches the user's location. Voice input is handled through speech to text [15] for capturing destination and navigation is launched via a Google Maps URL which is given by the Gemini API [10] after analyzing source and destination. Audio prompts assist the user to reach destination after it is redirected to Google Maps in walk mode. The route optimization is done using the following distance measure:

$$D = \sqrt{\left\{\left(x_{\{2\}} - x_{\{1\}}\right)^2 + \left(y_{\{2\}} - y_{\{1\}}\right)^2\right\}} \quad (5)$$

Where $(x_{\{1\}},y_{\{1\}})$ and $(x_{\{2\}},y_{\{2\}})$ represent the latitude and longitude of the user's location and destination respectively, and D denotes the straight-line distance used to initialize route optimization.

G. SOS Emergency System

The SOS feature enables users to send emergency messages via SMS or WhatsApp along with their current GPS location. Contact data is managed using shared preferences [19]. Voice based interaction is powered by speech to text [15] and messages are triggered using URL launcher and permission handler [17] [18].

H. Current Location Information

This module retrieves the user's current location, nearby landmarks and weather using Geolocator [16], Open StreetMap [12] and OpenWeatherMap APIs [13]. The aggregated information is read out using flutter tts [14], providing real-time situational awareness.

I. Personal Chatbot

The Chatbot integrates the Gemini API [10] for natural language conversation. Voice input via speech to text [15] is processed and responded to flutter tts [14]. The Chatbot also features messaging interface for typing as well and voice replay, with custom handling for permission and API errors to ensure robust, accessible interaction.

V. Results and Discussions

Since the majority of the application's output is delivered through speech-based interaction, not all functionalities can be fully represented through screenshots alone. However, to demonstrate the system's effectiveness, key visual interfaces and interaction points are illustrated with representative screenshots. These screenshots reflect the functioning of each module, confirming the successful implementation of the system as proof of concept. Performance observations during the prototype phase indicate a minimum response latency of approximately 2 seconds per query.

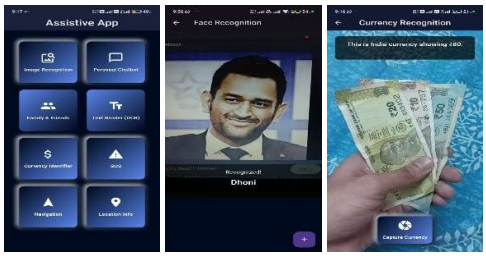

Fig. 2. a. Central Interface with Neuromorphic Buttons b. Face Recognition Interface c. Currency Recognition Interface

Fig. 3. a. Image Recognition Interface b. Text Reade Interface c. Text-to- Speech Interface

Fig. 4. a. Navigation Assistance Interface b. SOS Emergency System Interface c. Personal Chatbot Interface

The Central Interface (Fig. 2a) displays large, accessible neuromorphic buttons for intuitive navigation between modules, designed to accommodate the needs of visually impaired users. The Face Recognition feature (Fig. 2b) captures and identifies pre-registered faces using real-time input and provides auditory feedback for recognition status. The Currency Recognition module (Fig. 2c) accurately identifies Indian currency denominations and speaks the results aloud. The Image Recognition module (Fig. 3a) sends a captured image along with a user's voice prompt to the Gemini API, which returns a descriptive response spoken aloud by the system. The Text Reader (Fig. 3b) extracts and magnifies text from printed documents, supports font size changes, and can also freeze the frame to read particular extracted contents. In the Text-to-Speech module (Fig. 3c), the content obtained by the extraction and the summarized content is read by the Gemini API that supports multilingual content. Navigation Assistance (Fig. 4a) allows voice input of destination and walking directions to use Google Maps which is read out as voice to assist in navigation. SOS Emergency System (Fig. 4b) enables the user to send live position and a pre-determined emergency message both through SMS and WhatsApp. Voice interaction takes place throughout the process. The Current Location Information module is a combination of the location, nearby landmarks, and the weather information which is then spoken up in one, therefore it is not displayed in the form of an image. Finally, the Personal Chatbot (Fig. 4c) is a Chatbot which interacts with users by talking to them and gives them verbal replies through the Gemini API. The Chatbot interface also has message bubbles and voice replay, which are useful in making it easier to use.

VI. CONCLUSION

This paper outlines the design and implementation of a mobile based assistive application to assist the visually impaired people in their day to day lives. The application integrates some key features such as real-time image recognition, currency recognition, text-to-speech reader, magnification, face recognition, present position, emergency SOS notifications, navigation assistance, and a Chatbot chat in a single interface that is easy to operate. The app makes the use of APIs like Gemini and Google ML Kit and provides voice response and touch interaction, which makes it accessible and convenient to the users with different degrees of visual impairment. This system integrates various critical functions into a single mobile platform unlike many of the available options, which only perform a single task or need extra hardware, which makes this system affordable and convenient. The effectiveness of creating this prototype and testing positive results prove that it is possible to develop a full-scale assistive technology based on the commonly purchased mobile development platforms, such as Flutter.

Altogether, this work is a good demonstration of the concept, as it demonstrates the ability of modern AI and mobile technologies to enhance independence, safety and confidence to visually impaired users. As this system is tested further and receives user input, it can help become an excellent assistant in the day-to-day life of blind users. The app will be fully tested on edge cases to make sure that it is stable and reliable in the various circumstances. The use of gestures as a means of interaction will be emphasized to minimize the use of voice input that can be inaccurate in noisy places. The features will be provided with indoor navigation to supplement the current outdoor navigation. Face Recognition module will be transferred to the cloud-based server that will allow scaling and accessing it remotely. Also, the system will be improved to take advantage of the new Gemini models and use more precise data sources regarding weather and location than OpenWeatherMap and OpenStreetMap APIs. The purpose of these improvements is to increase system performance, reliability and user experience. While the current prototype demonstrates functional feasibility, a comprehensive user study with visually impaired participants is scheduled for the next phase. Future work will focus on gathering rigorous quantitative metrics for accuracy and latency across diverse datasets, to improve reliability in low-connectivity areas, future updates will implement offline caching for currency recognition and lightweight on-device object detection models. To introduce offline fallbacks for reliability, we plan to migrate face recognition to a secure private cloud to enable scalable, synchronized data management.

REFERENCES

[1] K. A. S. Sree Sindhura and J. Jaiswal, ”Real Time Object Detection and Recognition with a Voice Feedback for Blind,” Journal of Emerging Technologies and Innovative Research, vol. 6, no. 3, Mar. 2019.

[2] S. Shaikh, V. Karale, and G. Tawde, ”Assistive Object Recognition System for Visually Impaired,” International Journal of Engineering Research & Technology, vol. 9, no. 9, pp. 651–654, Sept. 2020.

[3] D. Ghule, S. Gaikwad, A. Jadhav, and S. N. Mhatre, ”Smart Vision: Vision For The Visually Impaired,” International Journal of Advanced Research in Computer and Communication Engineering, vol. 11, no. 4, pp. 67–70, Apr. 2022.

[4] N. Joshi, S. Maurya, and S. Jain, ”Real-Time Object Detection And Identification For Visually Challenged People Using Mobile Platform,” in Workshop on Advances in Computational Intelligence at ISIC, Feb. 2021, pp. 60–67.

[5] S. Ketkale, S. Londhe, M. Mujawar, and Sneha, ”Proposed System on Object Detection for Visually Impaired People,” International Journal of Research Publication and Reviews, vol. 3, no. 6, pp. 3474–3482, Jun. 2022.

[6] D. R. Bolla, H. Rauniyar, S. DN, R. T. P., and N. Rasool, ”Object Detection and Localization for Visually Impaired,” ITM Web of Conferences, vol. 57, 2023

[7] A. S. Ganesh, R. Sakthivel, and A. Ba, ”OCR based Image Processing with Audio Output for Visually Challenged People,” International Journal for Research in Applied Science & Engineering Technology, vol. 7, no. 3, Mar. 2019.

[8] K. Singh and M. Iyer,”An Integrated Assistive App for Visually Impaired using On-device Intelligence,” International Conference on Emerging Trends in AI and Assistive Technologies, pp. 100–107, Aug. 2024.

[9] Google Developers,”ML Kit: Text Recognition API,” [Online]. Avail- able: https://developers.google.com/ml-kit/vision/text-recognition. [Accessed: Nov. 05, 2024].

[10] Google,”Generative AI with Gemini API,” [Online]. Available: https://ai.google.dev. [Accessed: May. 20, 2024].

[11] Google Developers,”ML Kit: Face Detection API,” [Online]. Avail- able: https://developers.google.com/ml-kit/vision/face-detection. [Accessed: Jan. 16, 2025].

[12] OpenStreetMap Foundation, “Nominatim API,” [Online]. Available: https://nominatim.openstreetmap.org. [Accessed: Nov. 16, 2024].

[13] OpenWeather,”OpenWeatherMap API,” [Online]. Available: https:// openweathermap.org/api. [Accessed: Nov. 14, 2024].

[14] flutter.dev, “flutter tts – Text to Speech plugin,” [Online]. Available: https://pub.dev/packages/flutter tts. [Accessed: Jan. 10, 2025].

[15] flutter.dev, ”speech to text – Speech recognition plugin,” [Online]. Available: https://pub.dev/packages/speech to text. [Accessed: Jan. 12, 2025].

[16] flutter.dev, ”geolocator – Location plugin,” [Online]. Available: https://pub.dev/packages/geolocator. [Accessed: Nov. 14, 2024].

[17] flutter.dev,”url launcher – Launch URLs and apps,” [Online]. Available: https://pub.dev/packages/url launcher. [Accessed: Feb. 06, 2025].

[18] flutter.dev,”permission handler – Permission management plugin,” [Online]. Available: https://pub.dev/packages/permission handler. [Jan. 30, 2025].

[19] flutter.dev,”shared preferences – Persistent storage plugin,” [Online]. Available: https://pub.dev/packages/shared preferences. [Accessed: Jan. 23, 2025].

[20] Be My Eyes,”Be My Eyes – Virtual Volunteer for the Blind,” [Mobile App], [Online]. Available: https://www.bemyeyes.com/. [Accessed: Jun. 12, 2024].