A PROJECT SYNOPSIS ON

Project Group No: 45

"WeSee – A Helper for the Visually Impaired" SDG 3 – Good Healthand Well-being and SDG 10 - Reduced Inequalities.

SUBMITTED TO THE SAVITRIBAI PHULE PUNE UNIVERSITY, PUNE IN THE PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE

BACHELOR OF TECHNOLOGY (Computer Engineering)
SUBMITTED BY

Pooja Thorat	Exam no: 22410119
Rutik Tupe	Exam no: 22410122
Sarthak Velapure	Exam no: 22410125

DEPARTMENT OF COMPUTER ENGINEERING

AISSMS Institute of Information Technology
Kennedy Road, Near R.T.O., Pune – 411 001, Maharashtra (INDIA).

SAVITRIBAI PHULE PUNE UNIVERSITY, PUNE
2025-26

Abstract

WeSee – A Helper for the Visually Impaired is an innovative web-based assistive system that empowers individuals with low vision to perform daily tasks with enhanced independence. The application integrates **real-time data processing** through a device's **webcam**, leveraging **Computer Vision, Digital Image Processing, and Deep Learning** to provide instant assistance.

With features such as **real-time text recognition, object detection, and dynamic audio output**, *WeSee* enables users to seamlessly interpret their surroundings and interact with both written and digital content. Its adaptive design ensures consistent performance across diverse environmental conditions, making it practical for everyday use.

By reducing accessibility barriers, *WeSee* fosters **inclusion, independence, and equal opportunities** for visually impaired individuals. In alignment with the **United Nations Sustainable Development Goals (SDGs)**, this project directly contributes to **SDG 3 (Good Health and Well-being)**, while also supporting **SDG 10 (Reduced Inequalities)**.

Contents/Index

Sr. No	Title	Page no
1.	Project Title	**1**
2.	Introduction	**1**
3.	Technical Keywords	**2**
4.	Domain of Project	**2**
5.	Problem Statement	**2**
6.	Internal Guide	**2**
7.	Sponsorship Details and External Guide Details	**2**
8.	List of Features	**3**
9.	System Architecture	**4**
10.	List of Modules and Functionality	**7**
11.	Goals and Objectives	**8**
12.	Methodology	**9**
13.	Scope of the Project	**10**
14.	Software and Hardware Requirements	**11**
15.	Input to the Project	**12**
16.	Expected Output/Outcomes	**13**
17.	Algorithms	**14**
18.	Plan of Project Execution	**16**
19.	**References**	**17**

1.0 “WeSee – A Helper for the Visually Impaired”.

2.0 Introduction

For most people, vision is the primary sense that helps them understand and interact with the world. However, for **visually impaired individuals**, everyday tasks such as reading, identifying objects, or moving independently can be very difficult. Visual impairments cannot always be corrected with glasses, surgery, or medication, and this creates barriers in **education, social interaction, and independent living**.

With the advancement of **Artificial Intelligence, Computer Vision, and Deep Learning**, there are now opportunities to develop assistive tools that can make life easier for the visually impaired. Our project, *WeSee – A Helper for the Visually Impaired*, is a step in this direction. It is a **web-based application** that uses a **webcam to capture real-time data** from the user’s surroundings and process it instantly to provide meaningful assistance.

The application is capable of **real-time text recognition, object detection, and immediate audio feedback**. For example, a visually impaired person can use the webcam to point at a signboard, book, or object, and the system will instantly describe it aloud. This feature allows users to **read printed or handwritten text, identify objects, and navigate their surroundings more confidently**. Because the application works in real time, it adapts quickly to changes in the environment, making it suitable for both indoor and outdoor use.

The **scope** of the project is to build an effective, accessible, and reliable assistive tool. By combining **real-time webcam input with AI-driven processing**, *WeSee* bridges the gap between visual information and accessibility. It not only empowers visually impaired individuals to live more independently but also contributes to larger goals such as **inclusive education, improved quality of life, and social integration**.

In essence, *WeSee* is more than just a technical innovation—it is a **real-time companion** that turns visual data from a webcam into spoken words, helping visually impaired individuals **access information, navigate their world, and engage in daily life with confidence**.

3.0 Technical Keywords (ACM Key-words)

Technical Key Words:

- **Natural Language Processing (NLP)**
- **Deep Learning (DL)**
- **Machine Learning (ML)**
- **Optical Character Recognition (OCR)**
- **YOLO (You Only Look Once)**
- **Real-time Data Processing**
- **Webcam-based Assistive Technology**
- **Audio Signal Processing**

4.0 Domain of Project

Artificial Intelligence, Machine Learning, Natural Language Processing, Web Development, Deep Learning.

5.0 Problem Statement

Visually impaired people face many challenges in their daily lives, such as reading printed text, recognizing objects, and moving around independently. While tools like Braille or audio books exist, they do not provide help in **real time** or in every situation. Many modern solutions are either too expensive, too complicated to use, or not suitable for all environments. Because of this, there is still a big gap in technology that can give **instant, simple, and reliable support**. There is a strong need for an easy-to-use system that can use a **webcam to capture real-time data**, read text, detect objects, and provide **immediate audio feedback**. Such a solution would help visually impaired individuals live with more confidence, independence, and inclusion in everyday life. The application is a user-friendly, accessible, and capable of recognizing a wide range of text formats and objects in real-time, allowing visually impaired individuals to navigate their surroundings, read printed materials, and identify objects with ease and independence.

6.0 Internal Guide

Mrs. Shraddha Toney

8.0 List of Features

Core Platform Features:-

- **User-Friendly Interface:**
    - Simple, intuitive, and accessible design for visually impaired users.
    - Large buttons, voice prompts, and minimal navigation for ease of use.
- **Real-time Webcam Integration:**
    - Uses the device's webcam to capture live video data.
    - Processes input instantly for text recognition and object detection.
- **Audio Feedback System:**
    - Converts visual information into speech.
    - Provides immediate voice-based guidance in real time.
- **Image-based Detection:**
    - Identifies and analyzes images or frames captured through the webcam.
    - Extracts meaningful information from images (objects, signs, labels, or surroundings).
    - Provides immediate audio output describing detected elements.

Assistive Capabilities:-

- **Text Recognition (OCR):**
    - Detects and reads printed or handwritten text from the environment.
    - Converts text into audio for instant comprehension.
    - Supports multiple text formats (documents, signboards, books, labels, etc.).
- **Object Detection (YOLO + Deep Learning):**
    - Identifies and locates objects in real time using advanced algorithms.
    - Provides spoken descriptions to help users recognize surroundings.
    - Works across different environments (indoor/outdoor).
- **Context Awareness:**
    - Combines real-time text, object, and image data to give users contextual information.
    - Helps users interpret their environment more effective

Accessibility & Usability:-

- **Customizable Audio Settings:**
    - Adjustable volume, speed, and voice tone for audio feedback.
    - Multi-language support for broader accessibility.
- **Environmental Adaptability:**
    - Works in different lighting conditions and backgrounds.
    - Provides reliable performance in dynamic settings.
- **Low Latency Real-time Processing:**
    - Ensures fast response time for text reading and object detection.
    - Reduces delay between webcam input and audio output.

9.0 System Architecture

1. Core Web Application (Monolithic Core):-

The backbone of the *WeSee* system is a **Flask-based web application**. This core handles the overall structure, routes, and user interactions.

- **Technology:** Python with Flask
- **Database:** SQLite (for simplicity) – stores user profiles, interaction logs, and accessibility settings (e.g., voice speed, preferred language).
- **Responsibilities:**
    - **User Authentication:** Secure login and personalized settings for visually impaired users.
    - **Dashboard:** A simple, accessible interface where users can start live webcam detection, upload images/documents, or trigger text reading.
    - **Routing & URL Management:** Connects the user to different modules (OCR, Object Detection, Text-to-Speech).

2. AI-Powered Modules (Microservices-style)

The advanced features of *WeSee* are built as separate AI-driven modules, integrated into the Flask app.

- **Technology:** Python with Streamlit (for testing modules independently) + Flask (integration).
- **Deployment:** Each module can run as a microservice but is accessible from the main *WeSee* dashboard.
- **List of Modules:**
    - **OCR & Text Reader Module:** Extracts text from images/documents using *Pytesseract* and converts it to speech via *gTTS*.
    - **Real-Time Webcam Object Detection Module:** Uses **YOLOv5 with OpenCV** to identify and describe objects around the user in real time.
    - **Image-Based Detection Module:** Upload image → detect objects/text → provide spoken output.
    - **NLP Module:** Uses Hugging Face Transformers to simplify extracted text and make it more understandable before converting to speech.
    - **Audio Output Module:** Generates human-like speech for all detected objects/texts.

3. AI and Third-Party Service Integration

The intelligence of *WeSee* relies on integrating **Computer Vision, OCR, and NLP models**.

- **Core AI Libraries:**
    - **OpenCV:** For real-time webcam feed processing.
    - **YOLOv5:** For real-time object detection and labeling.
    - **Pytesseract OCR:** For text recognition from images/documents.
    - **Hugging Face Transformers:** For Natural Language Processing (text simplification, summarization).

- **Speech Technology:**
  - **gTTS (Google Text-to-Speech):** Converts detected objects and extracted text into spoken feedback.
  - Optional: Could be extended with *pyttsx3* for offline TTS.

4. Data Flow and User Interaction

Here's how a typical interaction works in *WeSee*:

1. **User Login:** The visually impaired user logs in through the Flask interface.
2. **Choose Action:** From the dashboard, the user selects either *Live Webcam Detection*, *Image Upload*, or *Text Reading*.
3. **Live Webcam Flow:**
   - Flask triggers the real-time webcam module.
   - OpenCV captures frames, YOLO detects objects, and labels are sent to the TTS module.
   - The user hears real-time descriptions of objects around them.

4. **Image Upload Flow:**
   - The uploaded image is processed.
   - Pytesseract extracts text → Hugging Face NLP simplifies → gTTS converts to speech.
   - If objects are present, YOLO detects and describes them.

5. **Output:** User receives **audio feedback** in natural, human-like speech for both objects and text.
6. **Settings:** User can adjust voice speed, language, and output format from their profile.

5. Accessibility Features

- **Keyboard Shortcuts:** For easy navigation without a mouse.
- **Voice Commands (future scope):** Allowing users to control the application hands-free.
- **Screen-Reader Friendly UI:** Simple HTML + Bootstrap with ARIA labels.

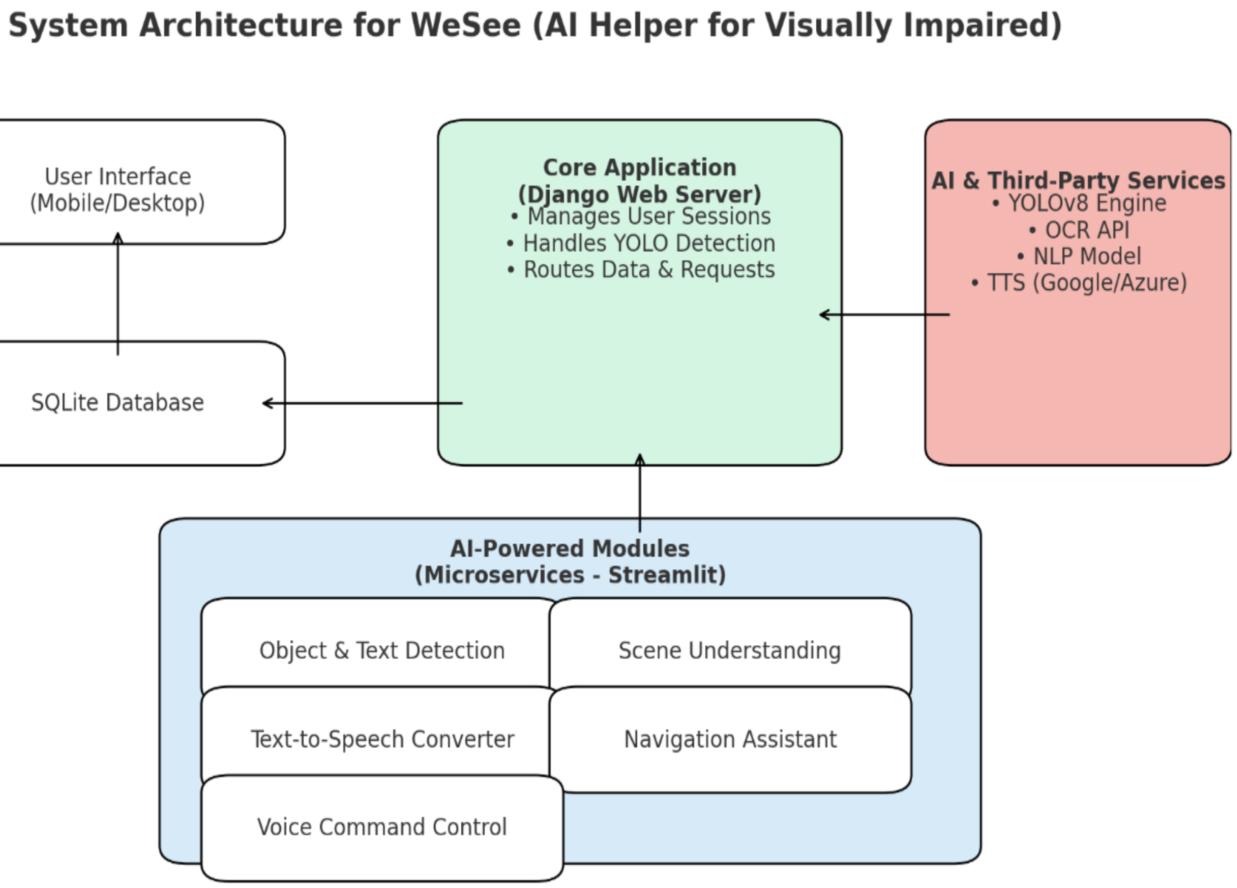

Figure 1: System architecture

10.0 List of Modules and Functionality:

1. Core System

Functionality:

- User Authentication: Provides a secure system for users to register, log in, and log out, ensuring privacy and personalized accessibility settings.
- Main Dashboard: Serves as the central hub of the application, offering a simple and accessible interface for users to navigate seamlessly between modules.

2. Text Recognition Module

Functionality:

- Upload Picture: Allows users to select and upload images containing text. The system extracts and presents the text in user-friendly formats such as enlarged on-screen text or text-to-speech.
- Capture Picture: Enables users to capture images of physical documents using a webcam or mobile camera for instant text recognition and conversion into accessible formats.
- Live Detection: Continuously processes live camera feeds, instantly detecting and narrating text content in real-time, providing dynamic assistance in physical environments.

3. Object Detection Module

Functionality:

- Upload Picture: Users can upload images for analysis, and the module detects and identifies objects within the picture, presenting results in descriptive text and speech form.
- Capture Picture: Captures images in real time through a camera, immediately identifying objects and offering contextual feedback to aid navigation and understanding.
- Live Detection: Provides continuous real-time object detection through a live camera feed, narrating detected objects and spatial cues to support navigation in unfamiliar or dynamic environments.

4. Accessibility Support Module

Functionality:

- Text-to-Speech Conversion: Converts extracted text or object descriptions into natural audio, ensuring seamless information access.
- Customizable Output: Provides options for large-print text, adjustable voice settings, and multiple language support for enhanced accessibility.
- User Feedback Loop: Allows users to give feedback on recognition accuracy, helping improve system performance over time.

5. Resource & Assistance Module

Functionality:

- Help & Tutorials: Provides easy-to-understand guides, video tutorials, and demonstrations for new users.
- Offline Mode Support: Ensures critical text recognition features can be used without an active internet connection.
- Integration with Assistive Devices: Supports compatibility with screen readers and external assistive technologies for visually impaired use.

11.0 Goals and Objectives

Goal:

The primary goal of *WeeSee* is to empower visually impaired individuals by providing them with independence and confidence in navigating their surroundings. The system aims to achieve this through real-time object detection and recognition using computer vision techniques like YOLO, combined with natural scene description capabilities powered by NLP. It also focuses on enabling text reading through OCR with speech output, making both printed and digital content accessible. Additionally, *WeeSee* seeks to support voice-based commands for seamless interaction, ensure a simple and user-friendly interface, and build a scalable framework that can be adapted for mobile or wearable devices in the future.

Objectives:

- Use **real-time webcam input** to detect and recognize objects, people, and environments with **YOLO (You Only Look Once)** and Deep Learning models.
- Provide **image-based detection and text-to-speech support**, so users can “hear” what’s around them.
- Integrate **Natural Language Processing (NLP)** to allow voice commands and interactive feedback.
- Ensure the system runs **in real time** with smooth processing and minimal delays.
- Build a **simple, user-friendly interface** that visually impaired users can easily operate.
- Leverage **AI, deep learning, and computer vision** to make the system scalable and adaptable to multiple real-world situations.

12.0 Methodology

The development of the *WeeSee* prototype followed a structured and modular methodology, ensuring efficient integration of real-time vision, NLP, and assistive audio features. The process was divided into multiple phases, from architectural planning to implementation, testing, and deployment.

1.System Design and ArchitecturalPlanning
The first phase focused on defining a flexible and scalable architecture suitable for real-time processing. A hybrid model was chosen: a central application built with Python and Flask for handling user interaction, system control, and data flow management, combined with modular AI components for computer vision, OCR, and NLP. For object detection, YOLOv8 was selected due to its speed and accuracy in real-time image analysis. Additional modules for text recognition (OCR) and speech synthesis were designed as separate, plug-and-play services. A lightweight database (SQLite) was adopted to manage configuration settings and logs, ensuring simple yet reliable integration.

2.Core Application Development
With the architecture defined, the central application was developed to serve as the backbone of *WeeSee*. This involved creating a user-friendly interface and implementing control over the webcam feed for live video capture. The core system was responsible for routing data between the real-time detection pipeline, the OCR engine, and the text-to-speech module. The dashboard was designed to be minimalistic and accessible, prioritizing smooth navigation for visually impaired users via audio feedback and keyboard shortcuts.

3.AI Module Implementation and Integration
Parallel to the core development, each assistive feature was developed as an independent module and later integrated into the system:

- Real-Time Object Detection Module: Built using YOLOv8 to identify objects and obstacles through live webcam input. Detected objects were processed and translated into audio alerts.
- OCR and Text Reading Module: Implemented with Tesseract OCR to read printed or handwritten text from captured frames, converting it into natural-sounding speech using a text-to-speech engine.
- Scene Description & NLP Module: Leveraged pretrained transformer models (such as BLIP or LLaVA) to generate natural language descriptions of the user's environment, enabling a deeper contextual understanding of surroundings.
- Voice Command Module: Integrated speech recognition APIs to allow users to

interact with the system hands-free, triggering features like object recognition or text reading via simple spoken commands.

4.Testing and Final Integration

Each module was tested individually with real-time webcam data to ensure accuracy, low latency, and robustness under varying lighting and background conditions. Integration testing followed, ensuring smooth communication between modules and consistent output delivery via audio feedback. Special focus was given to latency reduction, as real-time responsiveness was critical for usability. The final system was deployed on a local environment using a laptop webcam, with provisions for scaling to mobile or wearable devices in the future. End-to-end validation confirmed that a visually impaired user could operate *WeeSee* seamlessly to detect objects, read text, and receive scene descriptions in real time.

13.0 Scope of the Project

- **Core Platform Development:**
  - Development of a secure and user-friendly application capable of running on both desktop and mobile.
  - Implementation of a central user interface that integrates live camera feed, text-to-speech output, and real-time notifications for users.
- **AI-Powered Modules:**
  - **Real-Time Object Detection:** Using **YOLO** and **Deep Learning** models to detect objects, obstacles, and surrounding elements through the webcam feed.
  - **Text Recognition (OCR):** Extraction of text from images (e.g., signboards, labels, documents) and conversion into speech using **Text-to-Speech (TTS)**.
  - **Image-Based Navigation Assistance:** Providing guidance on walking paths by detecting obstacles, traffic signals, and directions in real-time.
  - **NLP-Driven Voice Assistant:** Natural language understanding to allow users to interact with the system using voice commands.
- **Technology Implementation:**
  - The project leverages **Python, Django/Flask for backend**, and **Streamlit or Flutter for frontend prototyping**.
  - Integration of **real-time webcam feed, speech synthesis (gTTS or pyttsx3), and audio-based interaction**.
  - Database for prototype: **SQLite** (with scope for migration to cloud databases like Firebase or PostgreSQL).
  - AI modules are built using **YOLO, OpenCV, OCR (Tesseract), NLP models, and Deep Learning frameworks like TensorFlow/PyTorch**.

14.0 Software and Hardware Requirements

Hardware Requirements:-

- **Development Machine / Server:**
  - CPU: Quad-core processor, 2.5 GHz or higher (GPU support recommended for real-time object detection).
  - RAM: Minimum 8 GB (16 GB recommended for running YOLO, OCR, and NLP models efficiently).
  - GPU (Optional but Recommended): NVIDIA GPU with CUDA support (e.g., GTX 1650 or higher) for faster deep learning inference.
  - Storage: At least 50 GB free space for datasets, trained models, libraries, and logs.
- **Client Device (End-User):**
  - Any standard desktop, laptop, or mobile device with a working webcam or external camera.
  - Minimum 2 GB RAM and support for modern browsers / mobile apps.
  - Microphone and speaker/headphones for voice interaction and TTS output.

Software Requirements:-

- **Operating System:**
  - Compatible with Windows 10/11, Ubuntu 20.04+ (Linux), macOS Monterey or later.
- **Server-Side / Backend Software:**
  - Programming Language: Python 3.9+
  - Web Framework: Django / Flask (for user interface & API integration)
  - AI & ML Libraries:
    - YOLOv5 / YOLOv8 (for real-time object detection)
    - OpenCV (for video processing and image-based navigation)
    - Tesseract OCR / EasyOCR (for text recognition from images)
    - TensorFlow / PyTorch (for deep learning models)
    - NLTK / spaCy / Hugging Face Transformers (for NLP-based voice assistant)
    - gTTS / pyttsx3 (for text-to-speech output)
  - Database: SQLite 3 (prototype), scalable to PostgreSQL/Firebase in future.

- **Client-Side Software:**
  - Any modern web browser (Google Chrome, Mozilla Firefox, Safari, Microsoft Edge).
  - Mobile app version (future scope): Flutter / React Native for Android & iOS.
- **Development Tools:**
  - IDE: Visual Studio Code / PyCharm
  - Version Control: Git + GitHub/GitLab
  - Package Manager: pip / conda for Python libraries
  - Virtual Environment: venv / Anaconda

15.0 Input to the Project

The inputs to the WeSee system are designed to make interaction simple and intuitive for visually impaired users. At the core, the system accepts image or video feeds from a connected camera (mobile or wearable device) which serve as the primary input for object detection, scene understanding, and text recognition. Additionally, users can provide voice commands to control the application, such as requesting object identification, navigation guidance, or text-to-speech conversion.

For the Object Detection Module, the input consists of real-time images or video frames, which are analyzed using YOLO to detect and label surrounding objects. The Text Reading Module accepts inputs in the form of printed or handwritten text captured through the camera, which is then processed using OCR techniques and converted into speech. The Navigation Assistance Module requires input from the device's camera and optionally GPS or sensor data to detect obstacles, pathways, and directional cues. Furthermore, the system also takes user preferences as input, such as language selection, speech speed, or type of information to prioritize (e.g., object detection vs. text reading).

16.0 Expected Output/Outcomes

1. Object Detection & Recognition: The system provides real-time identification of surrounding objects and obstacles through the webcam/camera feed, with corresponding voice output describing each object (e.g., “chair ahead,” “person on your left”).
2. Text-to-Speech Reader: Captured printed or handwritten text (books, signboards, documents, menus, etc.) is converted into clear and natural speech output in real time, enabling users to understand written information instantly.
3. Scene Understanding: Generates a brief audio summary of the environment (e.g., “You are in a living room with a sofa, table, and TV”), helping users gain contextual awareness of their surroundings.
4. Navigation Assistance: Provides real-time obstacle alerts, directional guidance, and pathway information using webcam/camera input and, if available, GPS/sensor data. Outputs are given as step-by-step voice instructions for safer mobility.
5. Voice Command Response: Responds to user voice queries (e.g., “Read this text,” “What’s in front of me?”) with immediate voice feedback, ensuring smooth hands-free interaction.
6. Personalization: Outputs are customized to user preferences, including speech speed, level of detail (short vs. descriptive), and language settings, making the system more accessible and user-friendly.

17.0 Algorithms

1. **Object Detection: YOLOv8 Model**

This module is the backbone of WESEE, enabling real-time detection of objects in the user's surroundings.

- Algorithm Steps:
    1. Image Capture: The live video stream from the camera is split into frames.
    2. Preprocessing: Frames are resized and normalized to match the YOLOv8 input format.
    3. Feature Extraction: The model uses convolutional layers to extract spatial features.
    4. Bounding Box Prediction: YOLOv8 divides the frame into a grid and predicts bounding boxes with class probabilities.
    5. Non-Max Suppression: Overlapping boxes are filtered to retain the most accurate detections.
    6. Output: Detected objects (e.g., "chair", "door", "person") are labeled with class and confidence score.

2. Text Recognition: OCR with Tesseract / EasyOCR

This module allows visually impaired users to access written information from their environment.

- Algorithm Steps:
    1. Image Capture: A snapshot of the text region is extracted from the camera feed.
    2. Preprocessing: The text region is converted to grayscale, denoised, and thresholded to improve clarity.
    3. Segmentation: Characters or words are segmented from the background.
    4. Feature Extraction: OCR extracts character-level features using pre-trained models.
    5. Text Output: The recognized text is converted into digital text format.

3. Navigation Assistance: Obstacle Detection & Distance Estimation

This module warns users about nearby obstacles for safe mobility.

- Algorithm Steps:
    1. Depth Estimation: Using stereo vision or monocular depth approximation, distances of objects are estimated.
    2. Object Prioritization: Objects closer than a threshold (e.g., 2 meters) are flagged as potential obstacles.
    3. Risk Analysis: Objects are categorized as "low", "medium", or "high" risk based on speed and distance.

4. Audio Feedback System: Text-to-Speech (TTS)

This module converts all detections and text outputs into natural, spoken audio.

- Algorithm Steps:
    1. Message Preparation: The system prepares a short, meaningful description (e.g., “Person ahead at 2 meters”).
    2. Speech Conversion: A TTS engine (e.g., Google TTS or pyttsx3) converts the text into speech.
    3. Real-Time Output: Audio is played through earphones/speakers instantly.

5 Voice Command & NLP Query Assistant

Enables hands-free interaction with WESEE through natural voice commands.

- Steps:
    1. Capture user’s voice input (e.g., “What’s in front of me?”).
    2. Convert speech to text using ASR (Automatic Speech Recognition).
    3. Apply NLP model to understand intent.
    4. Trigger the relevant module (object detection, text reading, or navigation).
    5. Generate spoken response back to user via TTS.

6. Webcam-Based Text Detection (Live OCR)

Allows users to point the webcam toward any printed/handwritten material (signboards, menus, documents). Extracts and reads text in real time.

Steps:

1 Capture continuous frames from webcam.

2 Preprocess (grayscale, denoise, thresholding).

3 Detect text regions (EAST/CRAFT text detector).

4 Apply OCR (Tesseract/EasyOCR) on detected regions.

5 Stream recognized text directly to Text-to-Speech for immediate audio output.

18.0 Plan of Project Execution

Sr.No.	Month	Tasks
1	August 2025	Topic Finalization and Topic Presentation
2	September 2025	Blueprint Design and Requirement Gathering
3	October 2025	Prototype Building
4	November 2025	UML Designing
5	December 2025	Task Distribution
6	January 2026	Coding and Implementation
7	February 2026	Coding and Implementation and paper publication
8	March 2026	Testing and Report Writing
9	April 2026	Final Submission

19.0 References

[1] R. Mittal and A. Garg, "Text extraction using OCR: A Systematic Review," 2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA), Coimbatore, India, 2020, pp. 357-362, doi:10.1109/ICIRCA48905.2020.9183326.

[2] Chauhan, S., Kumar, P., & Bhatia, T. (2020). Design and Implementation of a Smart Assistive Device for Visually Impaired People. International Journal of Advanced Research in Science, Communication, and Technology (IJARSCT), 2(3), 2456-9267. Retrieved from (https://ijarsct.co.in/Paper11233.pdf)

[3] Prakhar Sisodia and Syed Wajahat Abbas Rizvi, "Optical Character Recognition Development Using Python", J. Infor. Electr. Electron. Eng., vol. 4, no. 3, pp. 1–13, Nov. 2023.

[4] Diwan, T., Anirudh, G. & Tembhurne, J.V. Object detection using YOLO: challenges, architectural successors, datasets and applications. Multimed Tools Appl 82, 9243–9275 (2023). https://doi.org/10.1007/s11042-022-13644-y (https://rdcu.be/dCN2x)

[5] Peiyuan Jiang, Daji Ergu, Fangyao Liu, Ying Cai, Bo Ma, A Review of Yolo Algorithm Developments, Procedia Computer Science, Volume 199, 2022, Pages 1066-1073, ISSN 1877-0509, https://doi.org/10.1016/j.procs.2022.01.135. (https://www.sciencedirect.com/science/article/pii/S1877050922001363)

[6] Koppala Guravaiah, Yarlagadda Sai Bhavadeesh, Peddi Shwejan, Allu Harsha Vardhan, S Lavanya, Third Eye: Object Recognition and Speech Generation for Visually Impaired, Procedia Computer Science, Volume 218, 2023, Pages 1144-1155, ISSN 1877-0509, https://doi.org/10.1016/j.procs.2023.01.093. (https://www.sciencedirect.com/science/article/pii/S1877050923000935)

[7] Petrie, Helen, and Nigel Bevan. "The evaluation of accessibility, usability, and user experience." The universal access handbook 1 (2009): 1-16. (https://www.academia.edu/download/30388638/the_evaluation_of_accessib i lity_usability_and_user_experience.pdf)

[8] Tomlinson, S.M. (2016), Perceptions of accessibility and usability by blind or visually impaired persons: A pilot study. Proc. Assoc. Info. Sci. Tech., 53: 1-4. https://doi.org/10.1002/pra2.2016.14505301120

[9] Smartphone-Based Object Detection Systems
D. Ravi Kumar, Hiren Kumar Thakkar, Suresh Merugu,
Vinit Kumar Gunjan, and Suneet K. Gupta
https://www.researchgate.net/publication/356066523_Object_Detection_System_for_Visually_Impaired_Persons_Using_Smartphone

[10] National Programme for Control of Blindness & Visual Impairment (NPCBVI), Government
of India "The National Blindness & Visual Impairment Survey India https://npcbvi.gov.in/writeReadData/mainlinkFile/File341.pdf

[11] Badave A, Jagtap R, Kaovasia R et al (2020) Android based object detection system for visually
impaired. In: International conference on industry 4.0 technology 2020, pp 34–38. https://doi.org/10.1109/I4Tech48345.2020.9102694

[12] Anitha J, Subalaxmi A, Vijayalakshmi G (2019) Real time object detection for visually
challenged persons. Int. J. Innov. Technol. Explor. Eng. 8:312–314
5. Jadhav PSP, Tomy S, Jayswal SS et al (2016) Object detection in android smartphone
https://doi.org/10.17148/IJARCCE.2016.51171

[13] Khairnar DP, Karad RB, Kapse A et al (2020) PARTHA: A visually impaired assistance system.
https://doi.org/10.1109/CSCITA47329.2020.913779

Project Topic Finalization Committee

Mrs. S.B. Toney	Dr. A. G. Said	Dr. K. S. Wagh
Project Guide	Project Coordinator	HOD Comp. Engg