A Project Report

Submitted in partial fulfillment of the requirements
for the award of the Honors Degree of

BACHELOR OF SCIENCE (COMPUTER SCIENCE)

‘WeSee - A Helper for the Visually Impaired’

By,

Ms. SAACHI GHANSHYAM SHINDE

Under the Esteemed Guidance of,

Prof. ABDUL RASHID PATEL

MALAD KANDIVALI EDUCATION SOCIETY’S

DEPARTMENT OF COMPUTER SCIENCE OF
NAGINDAS KHANDWALA COLLEGE
(AUTONOMOUS)

(Reaccredited ‘A’ Grade by NAAC)
(AFFILIATED TO UNIVERSITY OF MUMBAI)
(ISO 9001:2015)

2023-2024

MALAD KANDIVALI EDUCATION SOCIETY'S

**NAGINDAS KHANDWALA COLLEGE OF COMMERCE,
ARTS & MANAGEMENT STUDIES & SHANTABEN NAGINDAS
KHANDWALA COLLEGE OF SCIENCE
MALAD [W], MUMBAI – 64**

(AUTONOMOUS)

**(Reaccredited 'A' Grade by NAAC)
(AFFILIATED TO UNIVERSITY OF MUMBAI)
(ISO 9001:2015)**

CERTIFICATE

This is to certify that the project entitled, **'WeSee – A Helper for the Visually Impaired'** is bonafide work of **Ms. Saachi Ghanshyam Shinde** bearing the roll no. **12** for the course**, Final Project (Course Code: 2365UHAIPR)** submitted in partial fulfillment of the requirements for the award of degree in **Bachelor of Science (Honors)** in **Computer Science**, specializing in **Artificial Intelligence and Machine Learning** from University of Mumbai.

Internal Guide **Coordinator** **External Examiner**

Date: **College Seal**

DECLARATION

I, **Saachi Ghanshyam Shinde,** bearing roll number **12**, declare that the project entitled **'WeSee - A Helper for the Visually Impaired'** conducted **at Nagindas Khandwala College**, is entirely my original work.

I hereby affirm that this project has not been duplicated for submission to any other university, and to the best of my knowledge, no other individual has submitted it elsewhere.

This project is undertaken in fulfillment of the requirements for the award of the Honors degree of **Bachelor of Science in Computer Science,** specializing in **Artificial Intelligence and Machine Learning** as part of the final semester curriculum.

Any external sources or references utilized in the project have been duly acknowledged through proper citation. Therefore, I understand the importance of academic integrity, and I take full responsibility for the content and originality of this project.

Ms. Saachi Ghanshyam Shinde

_____

ACKNOWLEDGEMENT

The successful completion of this project has been made possible through the resolute support and guidance of many remarkable individuals.

I extend my sincere thanks to the monumental figures of my college, for their great support throughout my graduation years.

Dr. (Mrs.) Ancy Jose, Director of MKES INSTITUTIONS,
Prof. Dr. Moushumi Datta, Principal,
Prof. Dr. Mona Mehta, Vice Principal & IQAC Coordinator,

I take this opportunity to express my profound gratitude and deep regards to **Prof. Rashmi Tiwari**, Head of Department (iNurture) for her continuous support and encouragement.

Further, I am particularly grateful to my project guide, **Prof. Rashid Patel**, whose dedicated guidance played a pivotal role in shaping the project. This project has been shaped by his expert opinions and he has helped me improve this project and achieve the level that it has acquired.

In conclusion, my deepest appreciation goes to all those mentioned above, as well as others who may not be explicitly named but have played a part in this journey. Their collective efforts have been the driving force behind the successful completion of this project.

ABSTRACT

‘**WeSee - A Helper for the Visually Impaired**’ – Simply, a web application for individuals with low vision capabilities to excel in daily tasks.

The project, **‘WeSee’** is aimed to address the challenges faced by visually impaired individuals by leveraging cutting-edge technologies, primarily based in Computer Vision and Digital Image Processing, specifically built upon the foundations of Deep Learning.

It is an innovative assistive tool that empowers visually impaired users to access written and digital content seamlessly, by leveraging a device's camera to provide real-time information about the user's surroundings.

Key structures of ‘**WeSee’** include real-time text recognition, real-time object detection and audio output capabilities, enabling users to effortlessly comprehend and interact with their surroundings. Moreover, the system is designed to adapt to various environmental conditions, providing a robust solution for users in different settings.

In conclusion, this project contributes to the inclusive integration of individuals with visual disability into our society by fostering independent access to information and promoting equal opportunities.

Through the integration of state-of-the-art technologies in Artificial Intelligence and Machine Learning, this project aligns with the overarching goal of creating a more accessible and inclusive world for all individuals.

TABLE OF CONTENTS

LIST OF FIGURES

Chapter 1: Introduction

1.1] PROBLEM DEFINITION

Visually impaired people, also known as visually challenged or visually disabled individuals, refer to individuals who experience limitations or difficulties in their vision that cannot be fully corrected by glasses, contact lenses, medication, or surgery.

Thus, 'Visually Impaired' is an umbrella term that covers a range of visual impairments, from partial sight to total blindness.

Simply, all visually impaired individuals face major challenges in interpreting visual information. They often face a barrier against independent navigation, information gathering, and interaction with the environment in their daily lives.

Therefore, Problem Definition for this project, 'WeSee' is to develop a web application that addresses these challenges by providing efficient text recognition and object detection capabilities with scene recognition through voice output, specifically tailored for visually impaired users.

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.

1.2] OBJECTIVES

The Objective of developing ‘WeSee’, a web application for assisting visually impaired individuals is multifaceted and aims to significantly improve their quality of life and independence.

Simply, ‘WeSee’ is an innovative assistive tool that empowers people with visual impairments to access digital content seamlessly, by using a camera to provide real-time context of the surroundings.

Key structures of ‘WeSee’ include text recognition, object detection and audio output capabilities, enabling users to effortlessly comprehend and interact with their surroundings. One key objective is to enhance accessibility through the creation of a simple, intuitive and easy to navigate user-friendly interface.

Another critical objective is the use of efficient text recognition algorithms that are capable of converting printed or handwritten text into digital formats that can be easily read aloud or presented in accessible formats such as large print.

Additionally, the project implements robust object detection capabilities. This functionality will assist visually impaired users in identifying and locating objects in their surroundings, ensuring that users receive instant information as they move through different environments.

Overall, the primary objective is to control technology to empower visually impaired individuals, enabling them to access information, navigate their surroundings, and engage more independently in various aspects of life, such as social interactions, education, & more.

1.3] SCOPE

The Scope of ‘WeSee’, refers to the specific boundaries, objectives, deliverables, and requirements that are accomplished and included within the project. It outlines the overall goals and constraints of the project, providing a clear understanding of its achievements and its resources.

The completed scope of the ‘WeSee’ project, showcases a comprehensive and successful execution. The application's user-centric design prioritizes accessibility, featuring a seamless interface that integrates with assistive technologies for an intuitive user experience.

Overall, ‘WeSee’ stands as a successful and impactful solution, significantly enhancing the independence and quality of life for visually impaired individuals through advanced text and object detection capabilities.

The following highlight the completed scope of the project:

- User-Centric Interface
- Advanced Text Recognition Algorithms
- Efficient Object Detection Capabilities
- Real-time Functionality
- Immediate Audio Output
- Good Accuracy and Reliability
- Comprehensive Documentation
- Iterative Development

Chapter 2: Project Study

2.1] LITERATURE REVIEW

The development of assistive technologies for visually impaired individuals has garnered significant attention in the research community.

'WeSee' embodies this innovation, integrating text recognition, object detection, and audio output to provide real-time contextual understanding of digital content and surroundings. This Literature Review examines seminal works that underpin the technological advancements incorporated. Thus, it highlights the foundational work that informs and supports the fundamentals of the 'WeSee' project.

Study of Text Recognition Technologies:

The advent of Optical Character Recognition (OCR) technologies has revolutionized accessibility for visually impaired users. The, [1] paper provided an extensive review of OCR technologies, highlighting the evolution to the modern processing of text recognition using OCR.

Similarly, [2] explored the application of Tesseract OCR in real-time text recognition scenarios, demonstrating its effectiveness in converting printed text to speech for visually impaired users. In [3] their findings underscore Tesseract's adaptability and efficiency in Python, which are crucial for the 'WeSee' application.

Study of Object Detection Techniques:

The ability to detect and identify objects in real-time is fundamental for enhancing the autonomy of visually impaired individuals. The implementation of You Only Look Once (YOLO) models, as discussed by [4], represents a breakthrough in real-time object detection.

Even the comparative study of [5] shows the study of all YOLO models and how YOLOv5, with its lightweight architecture and fast processing times, offers flexible control of model size and data enhancement. This study gives YOLO5 to be particularly suited for assistive technologies, offering immediate feedback to users about their surroundings.

Furthermore, [6] detailed the integration of YOLO5 model to identify objects in their feed and the ability to generate speech describing the objects detected in the scene using gTTs, a key feature of 'WeSee'. Their research highlights the importance of intuitive audio responses in conveying object information effectively, enhancing spatial awareness for visually impaired users, a method proposed as 'Third Eye'.

Study of User Interface Design for Accessibility:

Creating user-friendly interfaces for assistive technologies is critical for ensuring their effectiveness. And so, [7] investigated best practices in designing accessible web applications, emphasizing simplicity, intuitiveness, and ease of navigation. Their guidelines inform the 'WeSee' interface design, prioritizing straightforward interactions to accommodate users' needs.

Moreover, [8] explored the limitations of web applications lacking accessibility and the impact of adaptive interfaces for user experience on visually impaired individuals, suggesting that customizable features significantly improve engagement and satisfaction for them.

To conclude, by integrating efficient text recognition algorithms, such as those found in Tesseract, and cutting-edge object detection models like YOLOv5, alongside creating an intuitive user interface, 'WeSee' aims to enhance accessibility and independence for visually impaired individuals.

Chapter 3: Requirement & Analysis

3.1] PROPOSED SYSTEM

The Proposed System is a web-based application designed to offer advanced image processing capabilities through a user-friendly interface to deliver a seamless experience for users seeking help in understanding their content.

The application is structured to provide both flexibility and efficiency in processing, with a focus on usability and performance. In the structure of the system, the application is organized into several key components, each serving a specific function within the larger ecosystem.

Text Recognition: Users can upload images, capture them directly through their device, or engage in live capture to identify and extract text within the images. This feature can be particularly useful for digitizing documents, etc.

Object Detection: Similar to text recognition, this feature allows users to upload or capture images for the purpose of identifying and categorizing objects within the image. This functionality is beneficial for a variety of applications, including information gathering and educational purposes.

Thus, this proposed system of 'WeSee' aims to harness the power of Digital Image Processing, based on Deep Learning to provide accessible text recognition and object detection services through a web interface to deliver a valuable tool for a wide range of visually impaired users of the entire spectrum and perform multiple applications.

3.2] JUSTIFICATION OF PLATFORM

‘WeSee’ stands justified on multiple fronts, addressing a pressing social need for a positive impact on the quality of life for visually impaired individuals.

Additionally, the technical justification for the ‘WeSee’ project is anchored with the application of state-of-the-art technologies and methodologies that align with the specific needs of visually impaired users. The following technical aspects underscore the project's rationale:

Advanced Computer Vision: Employing sophisticated algorithms for object detection and scene recognition, the project harnesses the potential of computer vision to interpret visual data in ways that mimic human sight.

Optical Character Recognition (OCR): The project integrates robust OCR technology to accurately transcribe printed text into spoken words, thereby granting users access to written information that was previously inaccessible.

Machine Learning and AI: Leveraging AI and machine learning models, ‘WeSee’ is equipped to learn from interactions, improve its recognition capabilities over time, and provide personalized experiences to users.

Text-to-Speech (TTS) Engine: Incorporating high-quality TTS engines enables the conversion of text data into clear and natural-sounding audio, making the output comprehensible and user-friendly.

Web Technology Stack: The choice of a modern web technology stack ensures cross-platform functionality, responsiveness, and scalability, essential for a seamless user experience across different devices and platforms.

User Interface (UI): The project prioritizes intuitive UI design, catering to the navigation patterns and interaction preferences of visually impaired users.

3.3] REQUIREMENT SPECIFICATION

The requirement specifications served as a foundation for designing, developing, and testing the 'WeSee' application, ensuring that it met the needs and expectations of visually impaired users while adhering to technical standards and best practices.

Therefore, the specification were broken down into following types:

- **Functional Requirements**

Text Recognition: the system must accurately recognize printed and handwritten text in various languages and formats.

Object Detection: the system should be able to detect and identify common objects in real-time, aiding in interaction for visually impaired users.

User Interface: the interface must be user-friendly, accessible, and compatible with assistive technologies.

Real-time Feedback: provide instant feedback and information to users during navigation and interaction.

- **Non-Functional Requirements**

Accessibility: the application must comply with accessibility standards and guidelines to ensure inclusivity for visually impaired users.

Performance: the system should deliver high levels of accuracy and reliability in text recognition and object detection functionalities.

Scalability: Design the system to be scalable and adaptable for future enhancements and technological advancements.

Usability: ensure the application is easy to learn, use, and navigate for visually impaired users with varying levels of technological expertise.

Reliability: the system should be dependable and stable, with minimal downtime or disruptions during operation.

- **Technical Requirements**

Programming Languages and Frameworks: specify the programming languages, frameworks, and technologies to be used for development.

Testing and Quality Assurance: include requirements for testing methodologies, test cases, and quality assurance processes to ensure system functionality and performance.

Hence, these requirements underwent review, validation, and approval to ensure accuracy, completeness and alignment with project goals. They served as a reference for the planning, development, testing, and evaluation, guiding the entire project lifecycle to successful completion.

3.4] PLANNING AND SCHEDULING

The ‘WeSee’ project was meticulously orchestrated, with distinct checkpoints to ensure progress and quality. A detailed project timeline was developed to outline the various stages of development, from conceptualization of the notion to the final deployment of the system.

The planning and scheduling were articulated around three critical project review milestones, held in January, February, and March to assess progress and thus, formed the vital stages of the timeline.

Review #1 - Project Planning and Initiation (10% Completion):

The project began with a thorough planning stage where the vision for the ‘WeSee’ was translated into an actionable blueprint. The initial phase set the stage, involving extensive research into the needs of visually impaired users and selecting appropriate technologies for text recognition, object detection, and scene recognition.

Review #2 - Project Development & Execution (40% Completion):

At this juncture, the foundation laid during planning was built upon, with the development team prioritizing the creation of a preliminary look. The chosen technologies were implemented to create the core modules, such as the text and object detection, which were essential for early testing and refinement.

Review #3 - Project Integration and Monitoring (80% Completion):

Leading up to the third review, the focus was on integrating voice output capabilities and refining the user interface for ease of use. Rigorous testing was conducted to ensure the application’s responsiveness and reliability, adjusting based on feedback and monitoring the performance.

Moreover, the project was scheduled and programmed in relation to the high priority tasks of ‘WeSee’.

The integral priority tasks referred to the key activities and components within the project that were critical for the success of ‘WeSee’ and timely completion. These tasks were essential, foundational and central to achieving the project's objectives and delivering the desired outcomes.

They were scheduled systematically and given priority in terms of their resource allocation and focus because delays or issues with these tasks significantly impacted the overall project timeline and success.

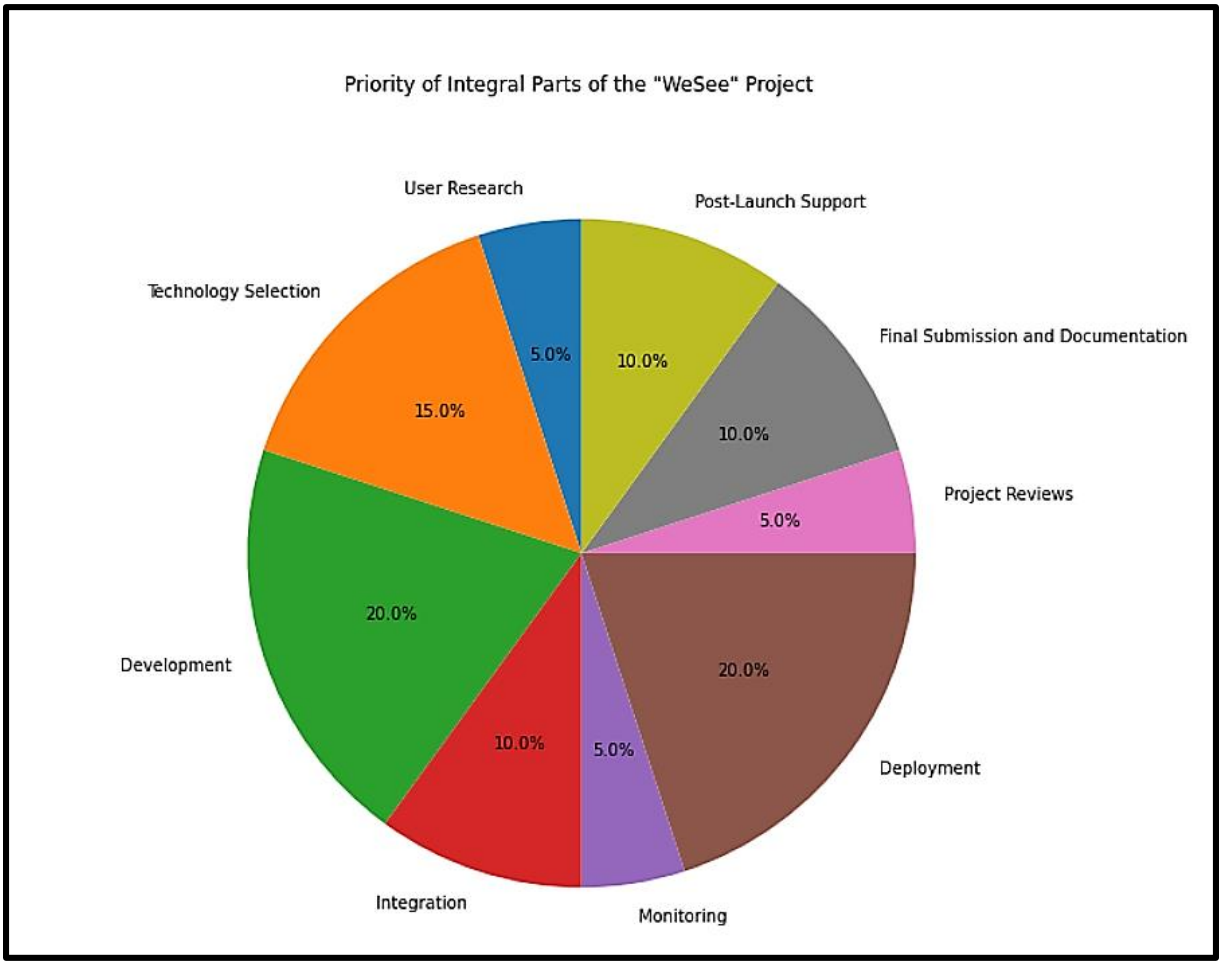

Figure 1: Priority of Integral Tasks

Finally, in the Project closure phase of ‘WeSee’, it involved the systematic and organized steps taken to conclude all activities, deliverables, and resources associated with the project.

It marked the official conclusion of the project and ensures that all project objectives have been met satisfactorily.

Thus, the culmination of the stages led to the final submission, where the project was comprehensively reviewed, documented and formalized the project closure with a complete report.

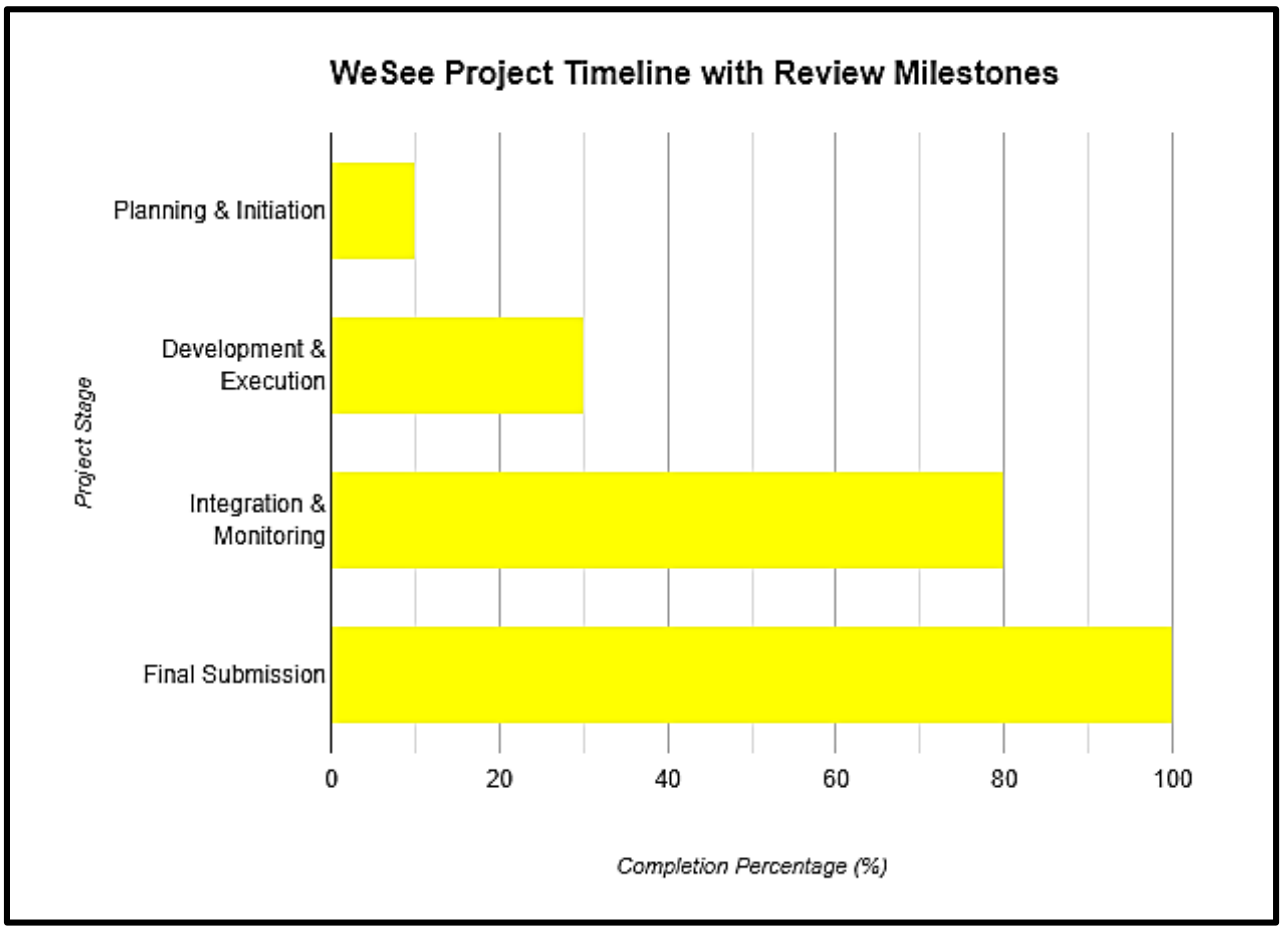

Figure 2: ‘WeSee’ Project Timeline

3.5] LIST OF TECHNOLOGIES

The development of the ‘WeSee’ web application involved the integration of several cutting-edge technologies to deliver advanced text recognition and object detection functionalities for visually impaired users.

The key technologies employed in this project include:

- **Programming Languages**

Python: primary programming language used for backend logic, algorithms, and data processing. Its versatility and extensive libraries make it ideal for complex machine learning tasks in the project.

JavaScript: utilized for frontend scripting and enhancing user interactivity. Script elements were integrated in HTML for dynamic UI components and seamless user experiences.

- **Deployment Framework**

Flask: a lightweight web framework in Python for backend server development and method creations. Flask's simplicity and scalability made it suitable for rapid routes and development iterations.

- **Front-End Frameworks**

HTML: the standard markup language for structuring web pages. It’s used in union with CSS and JavaScript to create structured and appealing interfaces.

CSS: applied for styling and visual design to web pages, ensuring consistency, appeal and aesthetics in the user interface.

Bootstrap: a CSS framework utilized for creating responsive interfaces, speeding up UI development and ensuring cross-browser compatibility.

- **Computer Vision**

OpenCV: a computer vision library used for image processing, analysis, and manipulation. Also, robustly enabled image recognition and object tracking.

- **Text Recognition and Processing**

Pytesseract OCR: integrated for accurate optical character recognition (OCR) to extract text from images and documents. Custom preprocessing techniques were implemented to enhance OCR accuracy for varying input.

- **Machine Learning and AI models**

YOLOv5: a cutting-edge object detection model that enabled real-time detection of objects in images and video streams. YOLOv5's speed and accuracy were leveraged for efficient object detection.

Hugging Face Transformers: leveraged for natural language processing (NLP) tasks in pre-trained models to enhance text-based functionalities.

- **Audio Output**

gTTS (Google Text-to-Speech): utilized for text-to-speech conversion, providing audio feedback to users and enhance the user experience.

- **Version Control**

Git: distributed version control system used for collaboration, managing model versions, branching strategies and version tagging were employed.

3.6] COMPARATIVE STUDY

In essence, the Comparative Study paints a compelling picture of the diverse technological landscape that empowered ‘WeSee’.

It reveals how ‘WeSee’ leveraged a potent blend of general-purpose programming, cutting-edge AI, and web development tools, all underpinned by robust version control practices.

The following bar chart offers a retrospective look into the technological foundation of the ‘WeSee’. Each bar represents a specific technology, and its length corresponds to the estimated level of usage employed within.

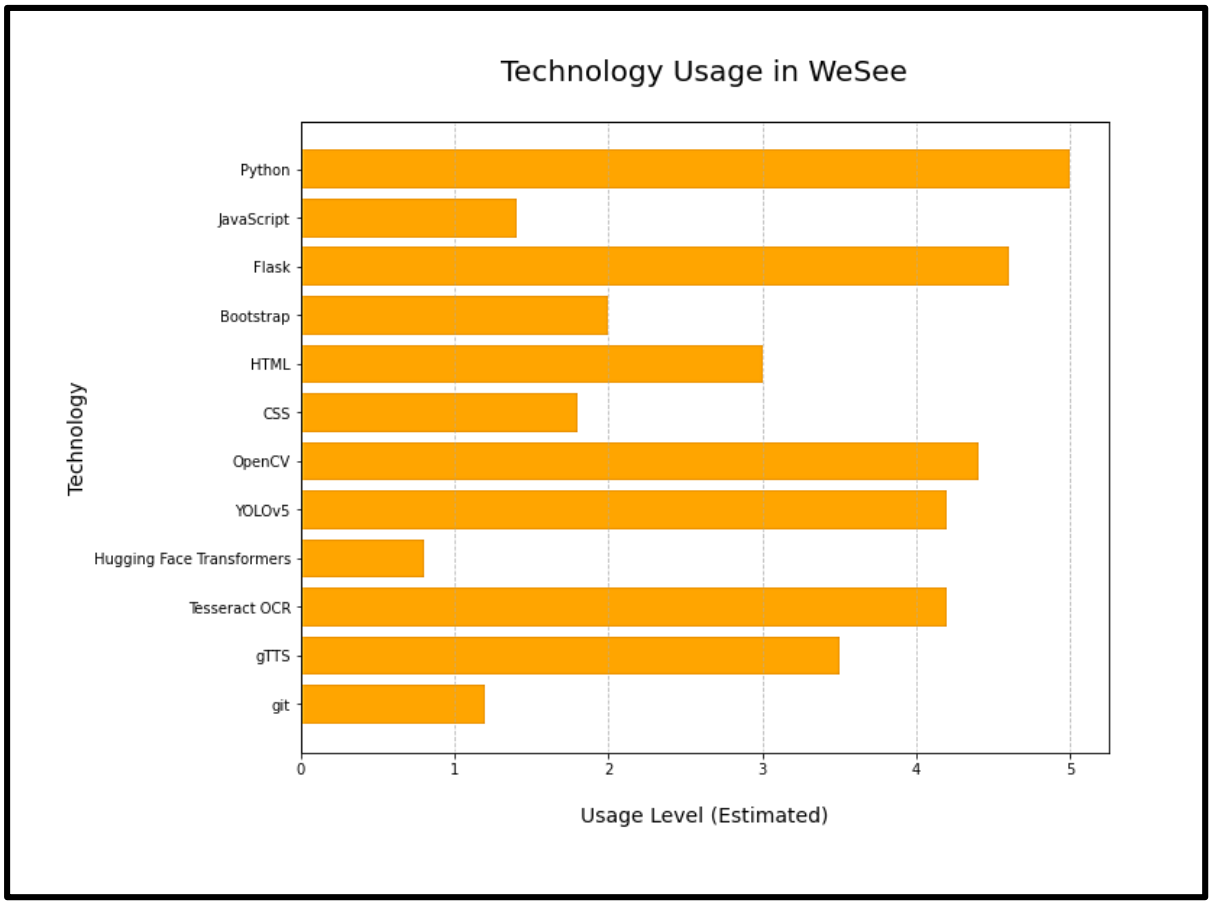

Figure 3: Comparison of Technologies

Python, The Foundation:

Python emerged as the most heavily utilized technology in ‘WeSee’. Its versatility and extensive libraries played a critical role in the project's development. This dominance shows that core functionalities and logic relied heavily on Python's well-established capabilities.

Diving into Artificial Intelligence:

The chart further reveals ‘WeSee's’ significant focus on cutting-edge Artificial Intelligence (AI) technologies. Also, OpenCV, YOLOv5, and Tesseract OCR all held good, prominent positions.

This portrays how computer vision, object detection, and text recognition played vital roles in ‘WeSee's’ overall functionality. Thus, this project utilized these technologies for tasks of image analysis, object identification and automated text extraction efficiently and effectively.

Additionally, gTTS highlights the integration of audio output functionalities, hinting at the integration of audio feedback mechanisms implemented within.

Beyond Core Development, Flask:

The presence of Flask indicates that ‘WeSee’ ventured beyond core development and delved into web application creation. Its lightweight nature had been ideal for building a custom web tailored to the project's needs.

Even shows its construction using HTML, the fundamental building block of web pages, along with CSS for styling the visual appearance and layout.

The inclusion of Bootstrap, a popular CSS framework, proposes how ‘WeSee’ prioritized creating a user-friendly and responsive web interface.

Chapter 4: System Development

4.1] BASIC MODULES

‘WeSee’ leverages a modular design approach to achieve its functionalities, for explicitly targeting accessibility features for disability users.

This structure promotes code reusability, maintainability, and scalability, ensuring efficient development and future project growth, ultimately empowering users with visual impairments.

The 2 Core Modules:

‘WeSee’ is built upon two core modules that address common challenges faced by visually impaired users.

- **Text Recognition Module**

This module focuses on extracting text data from images. It utilizes libraries and techniques like Optical Character Recognition (OCR) to decipher text.

This functionality assists visually impaired users in accessing information presented in physical documents, digital images, or even physical environments through live camera feeds.

- **Object Detection Module**

This module concentrates on identifying and locating objects within images. It employs computer vision algorithms and pre-trained models to recognize and pinpoint specific objects of interest.

This functionality will aids visually impaired users in navigating their surroundings, identifying objects in their environment, or understanding the content of visual information presented to them.

The 3 Sub-Modules within each Core Module:

Each core module is further divided into three sub-modules, catering to specific user interaction methods and accessibility needs:

- **Upload Picture:**

This sub-module allows users to select image files from their devices. The chosen image is then processed by the respective core module (text recognition or object detection) to deliver the desired results (extracted text or identified objects) in a user-friendly format, through large printed text and text-to-speech conversion.

- **Capture Picture:**

This sub-module enables users to capture images directly from their webcams. The captured image is then fed into the relevant core module for text recognition or object detection analysis. This can be particularly helpful for situations where users encounter physical documents or objects they want to understand in real-time.

- **Live Detection:**

This most vital sub-module facilitates real-time processing through a live camera feed. The camera stream is continuously analyzed by the core module, providing users with immediate feedback on text content (text recognition) or detected objects (object detection) within the live video feed, with continuous and immediate audio feedback.

This functionality can be crucial for visually impaired users navigating unfamiliar environments or understanding dynamic visual information.

In conclusion, the modular design implemented in ‘WeSee’ serves as a strong foundation for its text recognition and object detection functionalities.

The clear separation between core modules and sub-modules promotes a well-organized and structured codebase. And, enables efficient development, promotes code maintainability, and paves the way for future scalability as the project grows and incorporates new accessibility features.

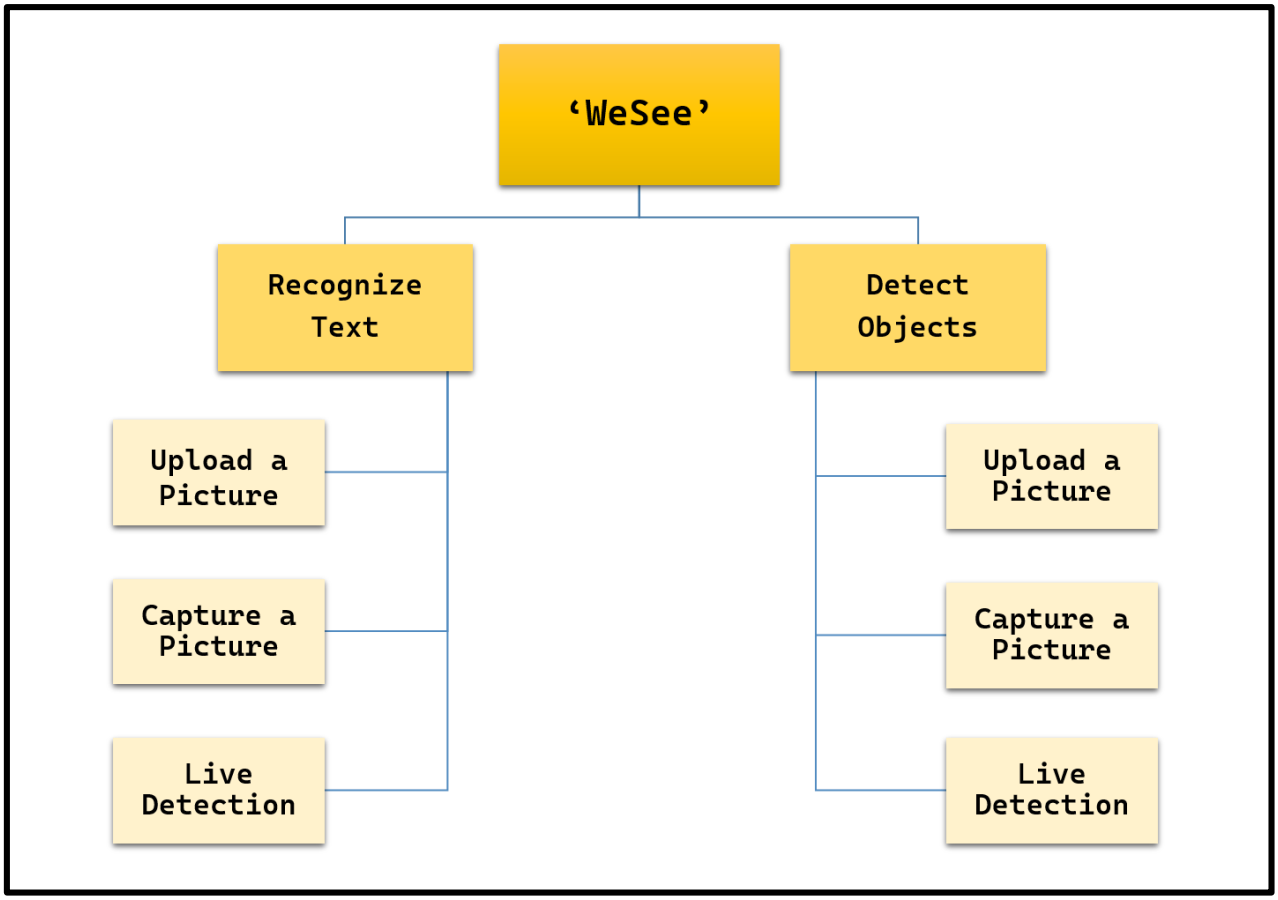

Figure 4: Basic Modules

4.2] CODES

The Backend Codes in ‘WeSee’ are the project's invisible engine, handling the core functionalities and logic that drive the application's functionalities. This section delves into the essential aspects of backend development.

Solid backend code is the backbone of a successful web application. In ‘WeSee’, its backend ensures efficient data management, secure interaction and seamless execution of core functionalities of text recognition and object detection. Hence, promoting acute application performance, scalability, and maintainability in the long run. The central codes of the ‘WeSee’ are:

- **main.py**

This is the main Python script for the web application, serving as the entry point for your web application. It handles routing, configuration, and interaction between the different parts of the project (templates, models).

```
from flask import Flask, render_template, request, send_from_directory,
jsonify

from werkzeug.utils import secure_filename
import os
from datetime import datetime

import cv2
import pytesseract
from gtts import gTTS

from text_models import text_recognition_upload
from text_models import text_recognition_capture
from text_models import text_recognition_live

from object_models import object_detection_upload
from object_models import object_detection_capture
from object_models import object_detection_live
```

```
app = Flask(__name__, static_folder='static')

app.config['UPLOAD_FOLDER'] = 'static/captured'
app.config['AUDIO_FOLDER'] = 'static/audio'


camera = cv2.VideoCapture(0)




@app.route('/')
def home():
    return render_template('home-page.html')


@app.route('/about')
def about():
    return render_template('about.html')


@app.route('/contact')
def contact():
    return render_template('contact.html')

@app.route('/text')
def text():
    return render_template('text-recognition.html')


@app.route('/text1')
def text1():
    return render_template('text-recognition-upload.html')


@app.route('/text-recognition', methods=['POST'])

def text_recognition():
    # Retrieve image data from the request (replace with actual logic)
    image_file = request.files.get('image')

    if image_file:
        # Save the image to a temporary location or use in-memory processing
        image_path = os.path.join(app.config['UPLOAD_FOLDER'],
secure_filename(image_file.filename))
        image_file.save(image_path)
```

```
        # Call the model1 function to perform text recognition
        text, audio_file, adjusted_image_path, timestamp =
text_recognition_upload.perform_ocr_and_audio(image_path)
        # Process the results (e.g., display text, play audio)
        return render_template('text-recognition-upload.html', text=text,
audio_file=audio_file, adjusted_image_path=adjusted_image_path,
timestamp=timestamp)

    else:
        return render_template('text-recognition-upload.html', error="Please
upload an image.")


@app.route('/audio/<path:filename>')

def serve_audio(filename):
    return send_from_directory(app.config['AUDIO_FOLDER'], filename)


@app.route('/text2')

def text2():
      return render_template('text-recognition-capture.html')


@app.route('/text-capture', methods=['POST'])

def text_capture():
    text_output, jitt_img_path, audio_path =
text_recognition_capture.perform_text_capture()
    return render_template('text-recognition-capture.html',
text_output=text_output, jitt_img_path=jitt_img_path, audio_path=audio_path)



@app.route('/text3')
def text3():
    return render_template('text-recognition-live.html')


@app.route('/text-live', methods=['POST'])
def text_live():

    text = text_recognition_live.perform_live_text()

    return render_template('text-recognition-live.html', text=text )
```

```
@app.route('/object')
def object():
    return render_template('object-detection.html')


@app.route('/object1')
def object1():
    return render_template('object-detection-upload.html')


@app.route('/object-upload', methods=['POST'])

def object_upload():

    if 'file' not in request.files:
        return 'No file part'
    file = request.files['file']
    if file.filename == '':
        return 'No selected file'
    if file:
        filename = secure_filename(file.filename)
        file_path = os.path.join(app.config['UPLOAD_FOLDER'], filename)
        file.save(file_path)

        # Call the object detection and generate audio description
        detection_result, audio_path, timestap =
object_detection_upload.detect_objects(file_path)

        # Get paths for displaying results
        audio_file = os.path.basename(audio_path)

        return render_template('object-detection-upload.html',
detection_result=detection_result, audio_file=audio_file, timestap=timestap)


@app.route('/uploads/<filename>')

def uploaded_file(filename):
    return send_from_directory(app.config['AUDIO_FOLDER'], filename)


@app.route('/object2')
def object2():
    return render_template('object-detection-capture.html')
```

```
@app.route('/object-capture')

def object_capture():

    description_text, img_filename, audio_filename =
object_detection_capture.perform_object_detection()

    # Render a template with the detection results and paths to the image and
audio file
    return render_template('object-detection-capture.html',
description_text=description_text, img_filename=img_filename,
audio_filename=audio_filename)


@app.route('/give_image/<filename>')

def give_image(filename):
    return send_from_directory('static/captured', filename)


@app.route('/give_audio/<filename>')

def give_audio(filename):
    return send_from_directory(app.config['AUDIO_FOLDER'], filename)


@app.route('/object3')
def object3():
    return render_template('object-detection-live.html')


@app.route('/object-live', methods=['POST'])

def object_live():

    detected_objects = object_detection_live.object_detection()

    return render_template('object-detection-
live.html',  detected_objects=detected_objects )



if __name__ == '__main__':
    app.run(debug=True)
```

- **home-page.html**

This is the HTML templates that defines the structure and content of the web applications, home page. It creates the main landing page of ‘WeSee’. And so, it’s responsible for briefly introducing the application and its purpose.

Moreover, it helps in providing clear navigation options for users to explore different functionalities of the application, such as separate links for text recognition, object detection, and the "About" page.

It also holds the integration with the CSS style-sheets and Bootstrap scripts.

```
<!-- ---- -->

<!doctype html>

<html lang="en" class="h-100">

<head>

<link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css"
rel="stylesheet"
integrity="sha384-
QWTKZyjpPEjISv5WaRU9OFeRpok6YctnYmDr5pNlyT2bRjXh0JMhjY6hW+ALEwIH"
crossorigin = "anonymous" >


    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <meta name="description" content="">
    <meta name="author" content="Mark Otto, Jacob Thornton, and
Bootstrap contributors">
    <meta name="generator" content="Hugo 0.88.1">

    <title> WeSee - a helper! </title>

    <link rel="icon" href="/static/eyeslogo.png" type="image/x-icon">
```

```
    <link
rel="canonical" href="https://getbootstrap.com/docs/5.1/examples/cover/">
    <!-- Bootstrap core CSS -->

<link href="../assets/dist/css/bootstrap.min.css" rel="stylesheet">

    <style>
      .bd-placeholder-img {
        font-size: 1.125rem;
        text-anchor: middle;
    font-family: 'Georgia', serif;
        -webkit-user-select: none;
        -moz-user-select: none;
        user-select: none;
      }

      @media (min-width: 768px) {
        .bd-placeholder-img-lg {
          font-size: 3.5rem;
        }
      }
    </style>

<!-- ---- -->

<body class=" text-center text-dark bg-warning" >

<!-- ---- -->

<div>

<!-- ---- -->

<header class="cover-container d-flex flex-column text-white py-0"
style="background-color: #1C2833;"  >


<div class="mx-3 mt-3 mb-1" style=" height: 70px;" >

<h2 class="float-md-start mb-0 fw-bold display-5 "> WeSee. </h2>

<img class="d-block mx-auto mb-4 float-md-start" src="/static/image.png" alt=""
width="100" height="60">

<nav class="nav nav-masthead justify-content-center float-md-end">
```

```
<button type="button" class = "btn btn-light px-3 me-md-3 gap-3 fw-bold"
style="background-color: #F7F9F9; height:50px;">
<a class="nav-link" style="color: #17202A;"  href="/about"> About </a>
</button>

<button type="button" class = "btn btn-light px-3 gap-3 fw-bold" style =
"background-color: #F7F9F9; height:50px; ">

<a class="nav-link" style="color: #17202A;" href="/contact"> Contact </a>
</button>

</nav>

</div>

</header>

<!-- ---- -->

<div style="height:80px;" > </div>

<!-- ---- -->

<main>

<!-- ---- -->

<div class="container align-items-center rounded-3 border shadow-lg"
style="background-color: #FCF3CF; width: 950px;  font-family: 'Georgia',
serif;">

<div>

<div style="height:50px;" > </div>

<p class="display-4 fw-bold"> See the World, <br> Hear the Words. </p>

<div style="height:30px;" > </div>

<h2 class="display-7"> A Helper for the Visually Impaired. </h2>

<div class=" mx-5">

<div style="height:20px;" > </div>
```

```
<p class="lead mb-4">
WeSee is an innovative assistive tool that empowers people with visual
impairments <br> to access digital content seamlessly, by leveraging a device's
camera to provide <br> real-time information about the user's surroundings.
</p>

<p class="lead mb-4">
The one of the most important goal of WeSee aligns with the overarching <br>
goal of creating a more accessible and inclusive world for all individuals.
</p>


<div style="height:40px;" > </div>

</div>

</div>

</div>

<!-- ---- -->


<div style="height:50px;" > </div>


<!-- ---- -->

  <div class="container my-5">

    <div class="row p-4 pb-0 pe-lg-0 pt-lg-5 align-items-center rounded-3 border
shadow-lg" style="background-color: #FCF3CF ;  font-family: 'Georgia', serif;">

      <div class="col-lg-7 p-3 p-lg-5 pt-lg-3">

        <h1 class="display-4 fw-bold lh-1"> let us help you with ...</h1>

<div style="height:20px;" > </div>

  <p class="lead"> Key structures of ‘We See’ include text recognition, object
detection and audio output capabilities along with real time help, enabling
users to effortlessly comprehend and interact with their surroundings. </p>

<div style="height:40px;" > </div>
```

```
<!-- ---- -->

<div class="container rounded-3 border shadow-lg align-items-center"
style="background-color: #D68910 ; color: #FDEBD0; width: 360px; height: 80px;"
>

<div class= "mx-3 align-items-center">

<a href="/text" class="list-group-item list-group-item-action d-flex gap-3 py-3"
aria-current="true">

<img src="/static/text.png" alt="twbs" width="38" height="38" class="rounded-
circle flex-shrink-0 mt-1 mb-1" >

<div class= "mx-3">

<h2 class="mb-0"> detect text </h2>

</div>

</a>
</div>
</div>

<!-- ---- -->

<div style="height:30px;" > </div>

<!-- ---- -->

<div class="container rounded-3 border shadow-lg align-items-center"
style="background-color: #D68910 ; color: #FDEBD0; width: 360px; height: 80px;"
>

<div class= "mx-3  align-items-center">

<a href="/object" class="list-group-item list-group-item-action d-flex gap-3 py-
3" aria-current="true">

<img src="/static/object.png" alt="twbs" width="38" height="38" class="rounded-
circle flex-shrink-0 mt-1 mb-1 ">

<div class= "mx-3">

<h2 class="mb-0"> detect images </h2>
```

```
</div>

</a>
</div>

</div>
</div>

<!-- ---- -->

      <div class="col-lg-4 offset-md-0 p-0 mb-5 overflow-hidden">

          <img  src="/static/lady.png" alt="" width="400px" height="400px">

      </div>
    </div>
  </div>

</main>

<!-- ---- -->


<div style="height:50px;" > </div>

<!-- ---- -->

<script src="../assets/dist/js/bootstrap.bundle.min.js"></script>

<!-- ---- -->


</body>

</html>

<!-- ---- -->
```

4.3] INTEGRATION & DEPLOYMENT

The integration and deployment phase took the developed components of ‘WeSee’ and combined them into a functional application ready for users.

Integration, the uniting of the pieces, saw the connection of various parts of the project into a cohesive whole.

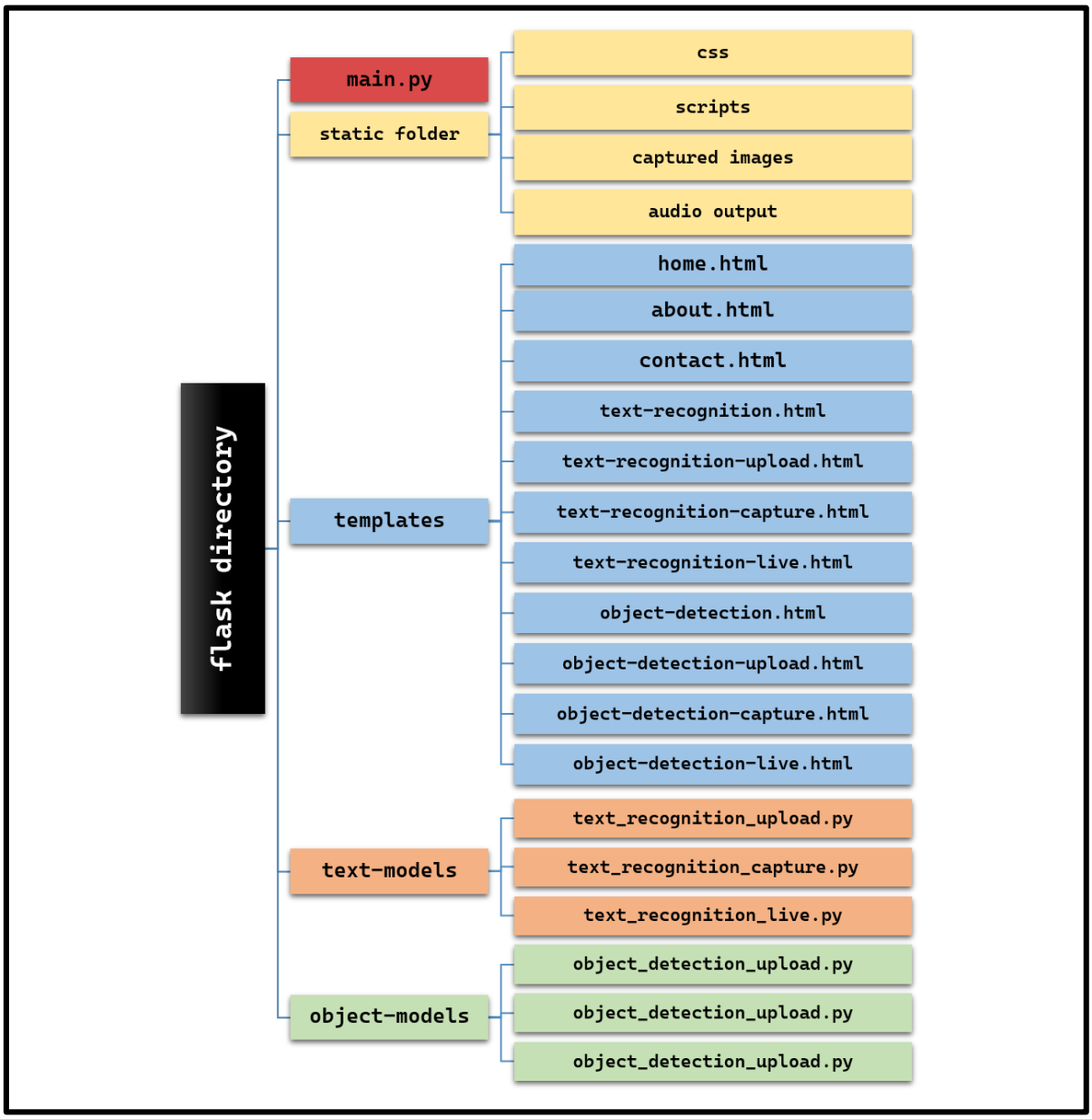

Figure 5: Flask Directory Integration

Here, the diagram above depicts the structure of the projects Flask web application in its core directory that was deployed through its 'main.py' file.

main.py: the main Python script of the Flask application, which serves as the entry point when running the application. And, evidently Flask maintained the routing, handling user requests and directing them to the relevant backend functions as per the functions triggered.

static: the assets like CSS styles (style.css) and JavaScript files (script.js) were referenced and incorporated into the HTML templates using this folder. Additionally, it also formed the primary location for storing the necessary uploaded, captured images and the .mp3 files of the audio output.

templates: the HTML templates within the templates directory, which define the user interface, were integrated with the backend logic implemented in main.py file. This ensured that user interactions on the web pages triggered the appropriate functionalities of all the modules.

text-models and object-models: these directory contain the Python files that define the core functionalities of the application related to text recognition and object detection. They further contain Python files for each of its sub-modules. And then, were imported onto the main.py file.

Finally, for the deployment of 'WeSee' the localhost deployment created the process of setting up and running the web application on the local machine.

Accessing the Application: http://127.0.0.1:5000/.

Open a web browser and navigate to the above IP. This launches the 'WeSee' web application instantly.

4.4] USER INTERFACE DESIGN

The User Interface (UI) design of ‘WeSee’ plays a critical role in its accessibility as it is a critical touch point for user interaction and experience.

The ‘WeSee’ UI prioritizes both simplicity, clarity and minimalist visuals. A clean and uncluttered layout, intuitive icons, and straightforward language ensure users with varying levels of visual impairment can easily understand.

It also employs color strategically to enhance readability and provide visual cues, such as high color contrast between text and background ensures optimal visibility for users with low vision.

- **Core Design Principles of ‘WeSee’:**

  ~ Simplicity and Clarity
  ~ Accessibility Best Practices
  ~ User-Centered Design

- **Effective Accessibility UI Components:**

  ~ Visual Design
  ~ Text and Font
  ~ Icons and Buttons
  ~ Layout and Navigation
  ~ Output Presentation

By prioritizing these design principles and incorporating accessibility best practices, the ‘WeSee’ UI empowers visually impaired users to interact with the application effectively. The result is a user friendly experience that helps user with greater confidence and independence.

- **‘WeSee’ Web Application UI**

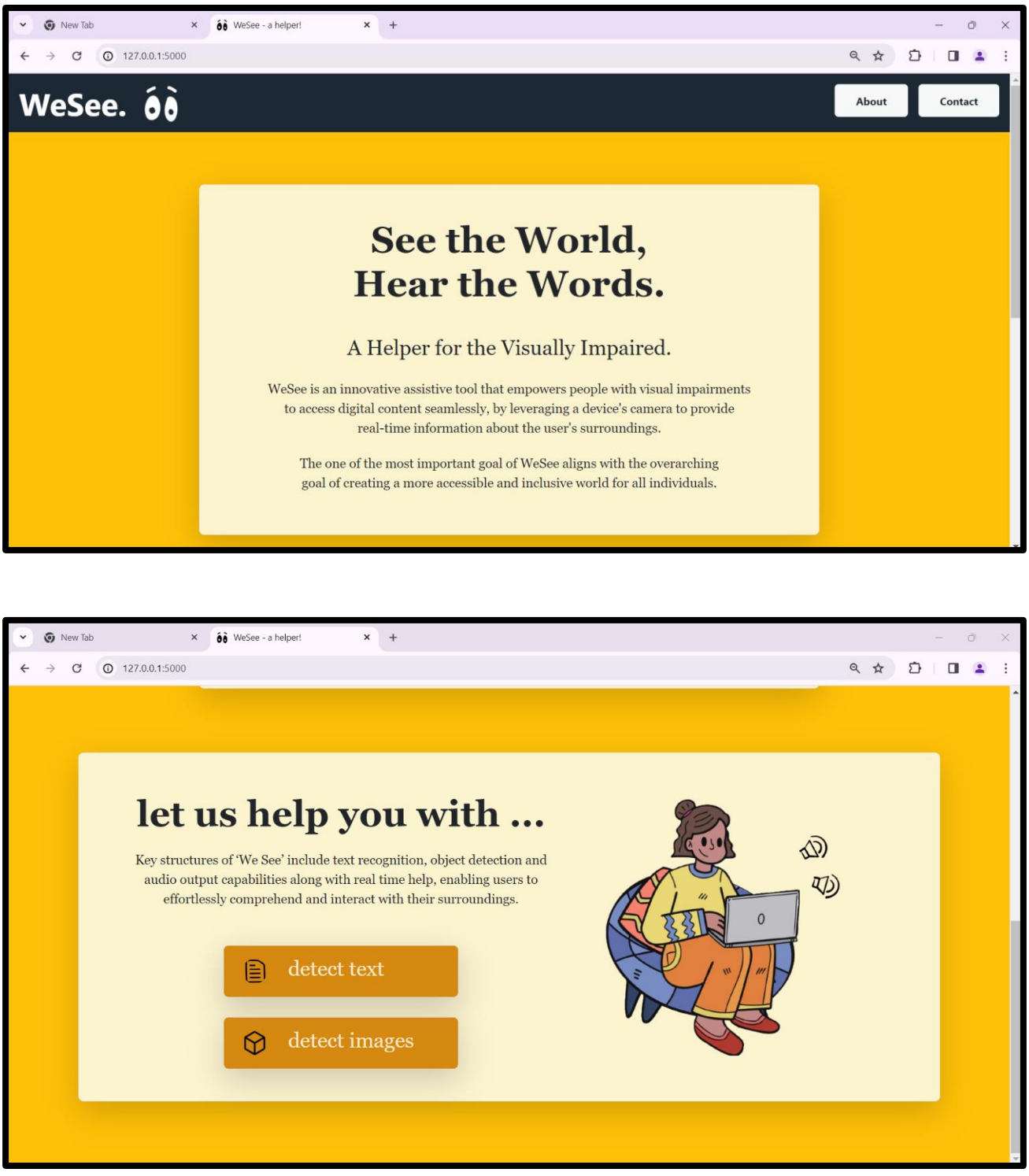

Figure 6: Screenshots of Home Page

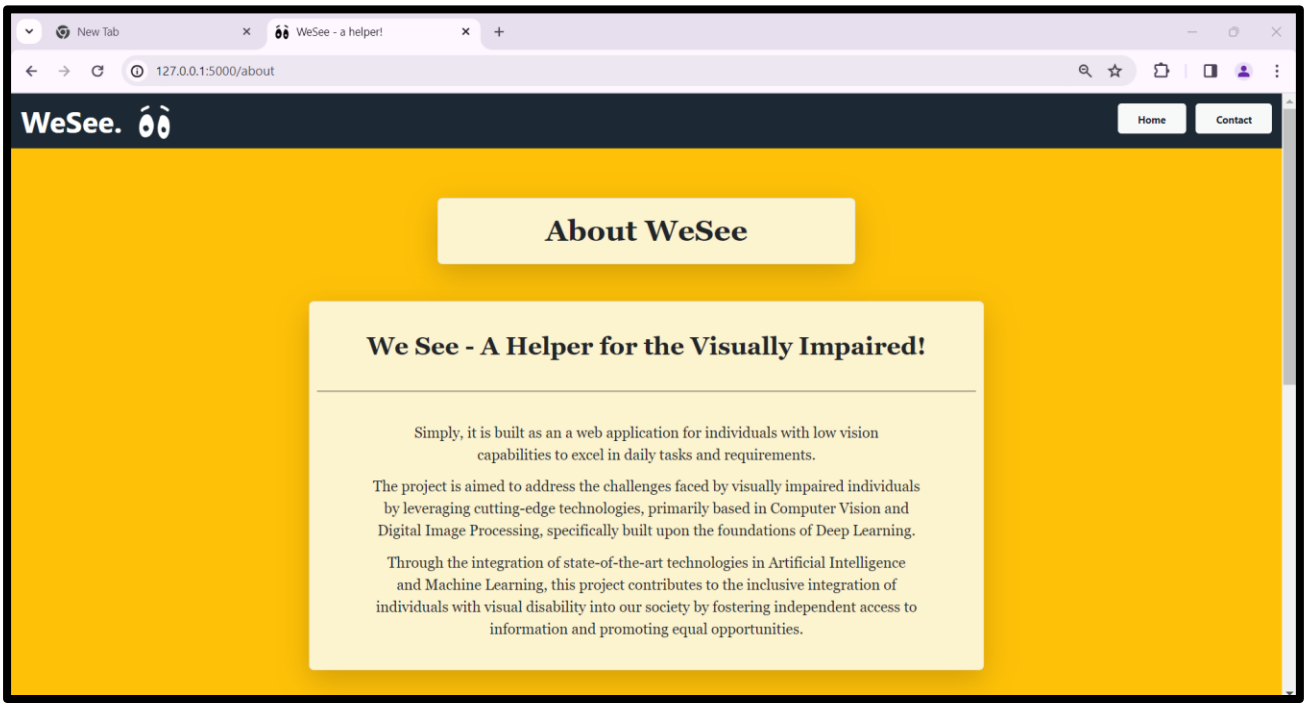

Figure 7: Screenshot of About Page

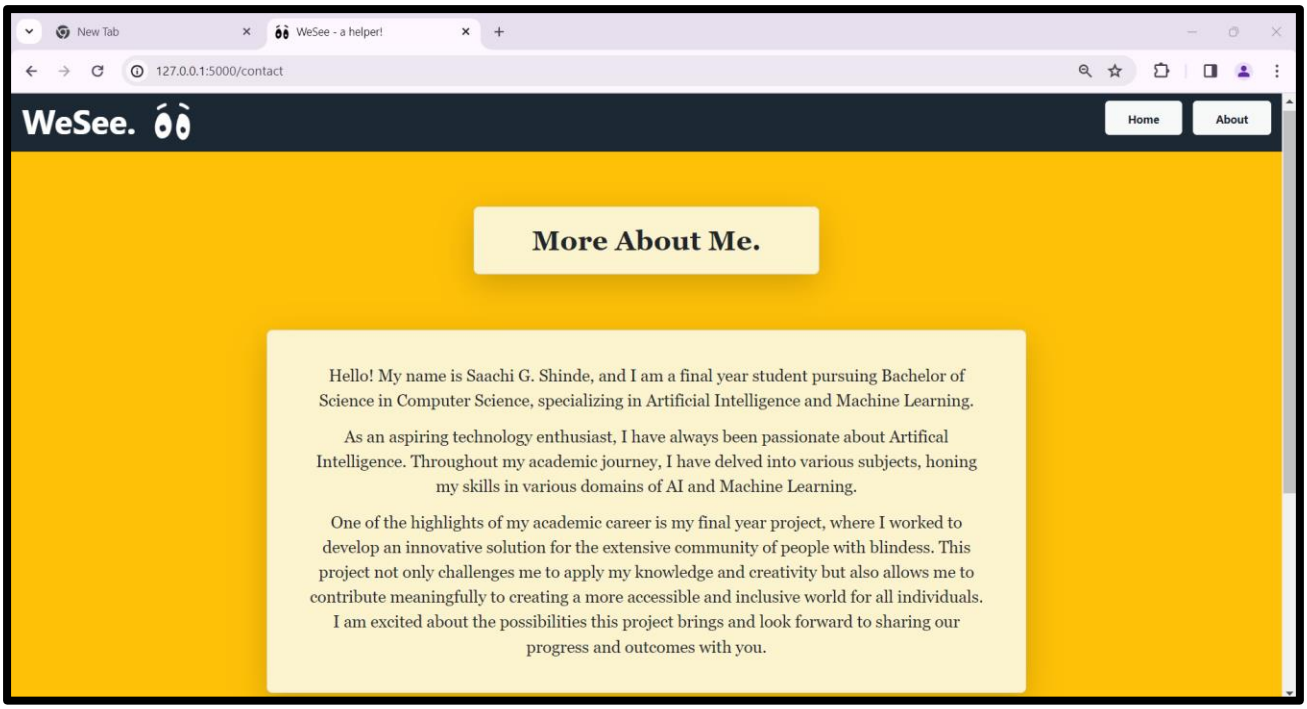

Figure 8: Screenshot of Contact Page

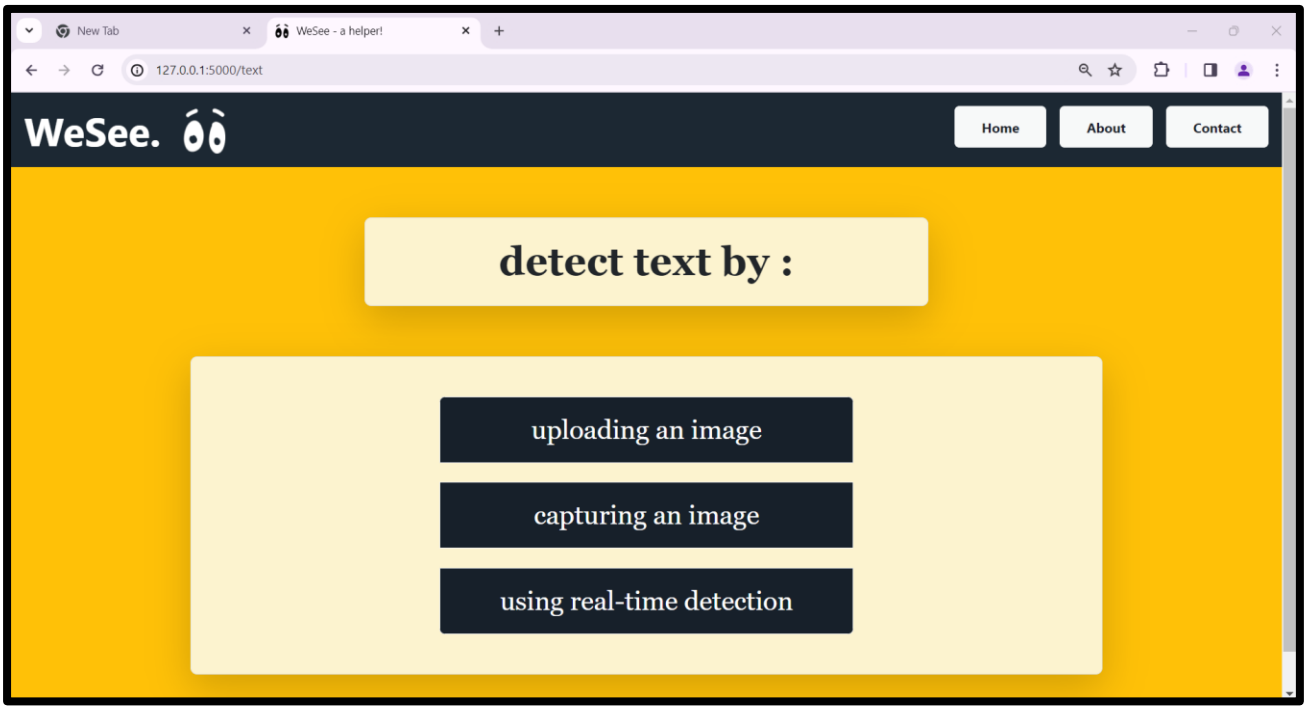

Figure 9: Screenshot of Text Recognition Page

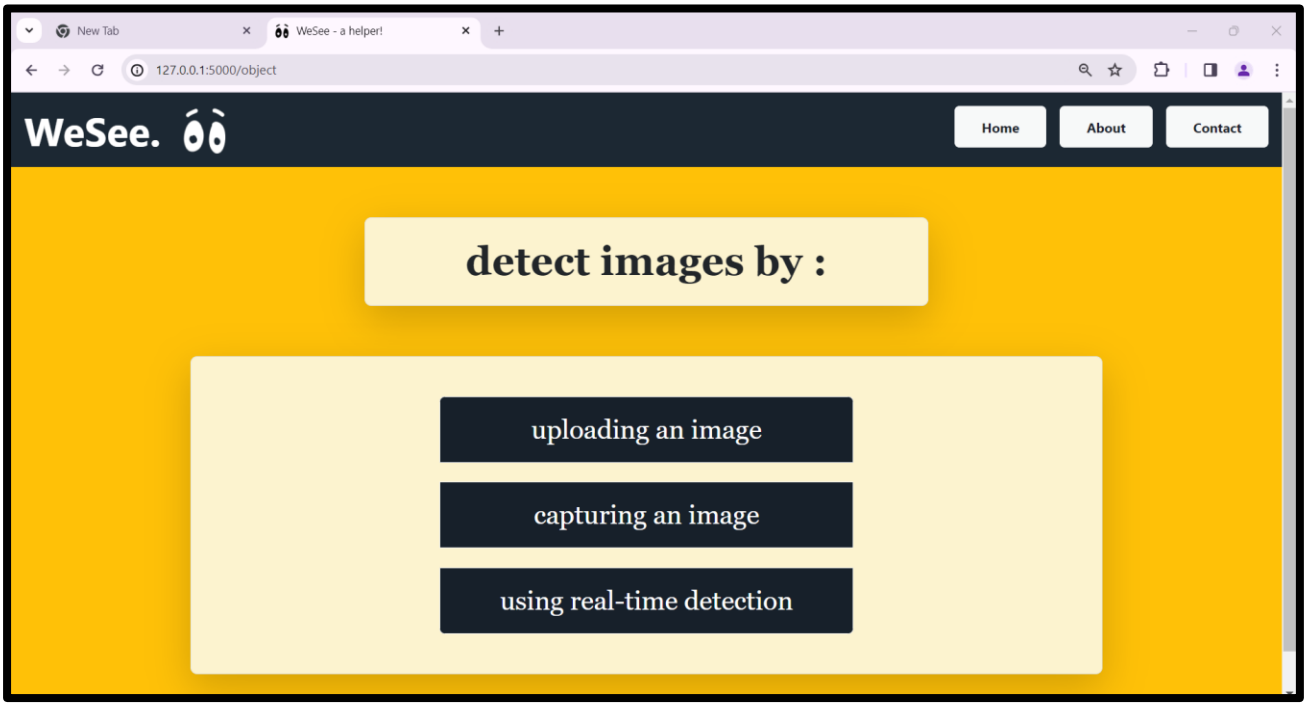

Figure 10: Screenshot of Object Detection Page

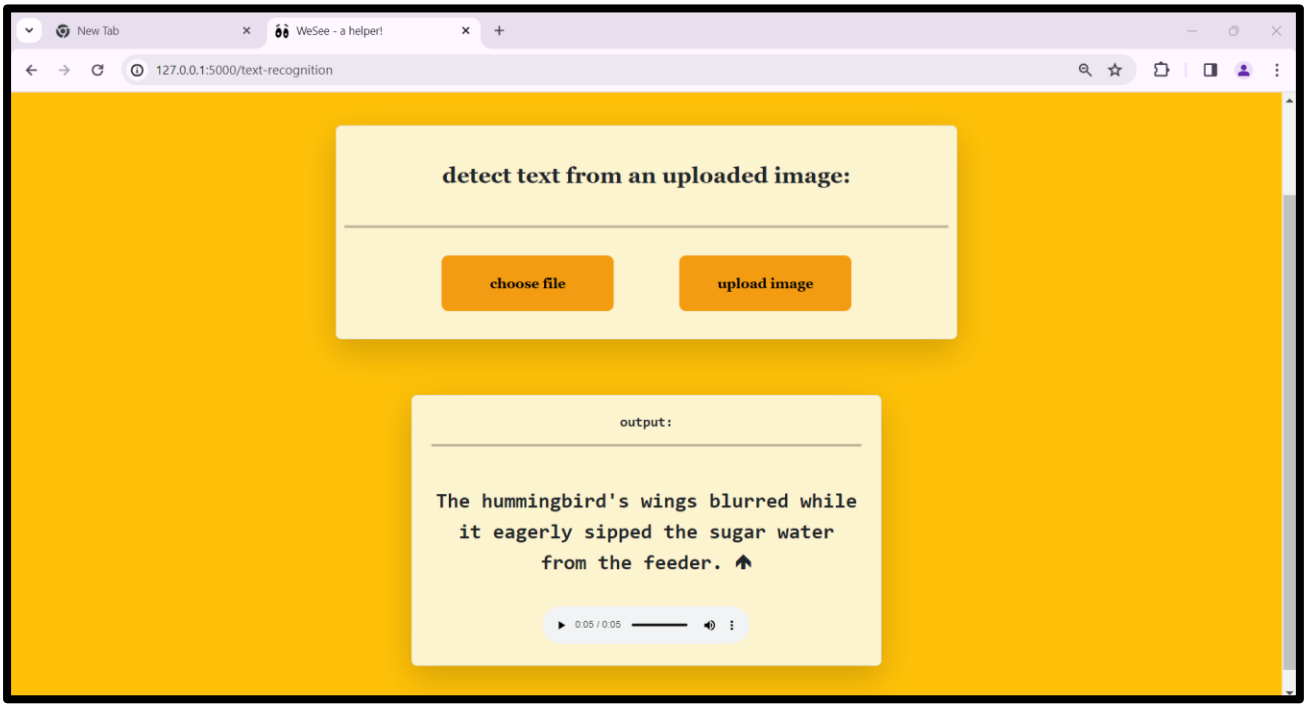

Figure 11: Screenshot of Text Recognition Output

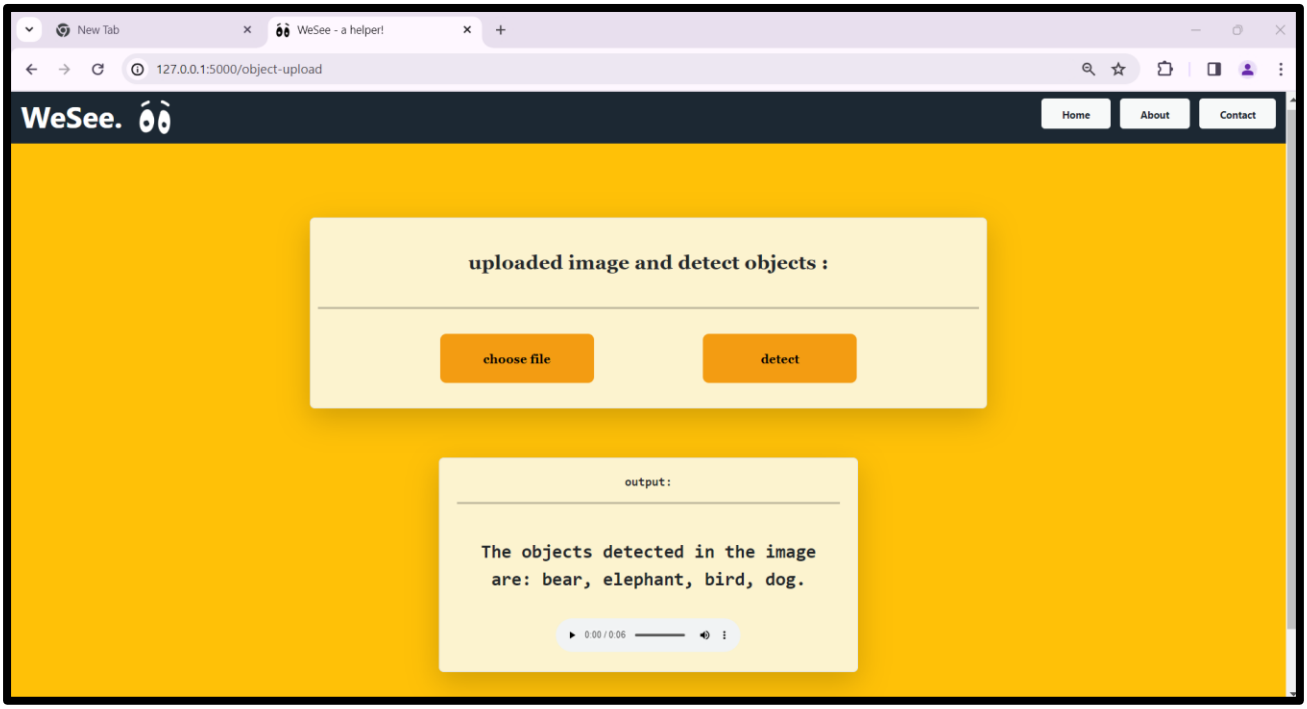

Figure 12: Screenshot of Object Detection Output

4.5] TEST CASES

Here, the ‘Test Cases’ delve deeper into the rigorous assessment of the ‘WeSee’ system. These test cases play a crucial role in ensuring the project is precise and delivers a robust, reliable, and effective experience.

Test Case 1: Detect Text in an Uploaded Image.	
Functionality:	Test OCR accuracy with user-uploaded images.
Test Data:	Image uploaded by the user.
Expected Result:	Extracted text displayed, audio feedback plays.
Actual Result:	Image processed successfully. Text is displayed accurately and audio feedback functions correctly.

Test Case 2: Detect Text in a Captured Image.	
Functionality:	Test OCR with image captured by the platform.
Test Data:	Image captured by user.
Expected Result:	Image capture successful, text displayed with audio feedback.
Actual Result:	Capture function works, text is extracted and displayed accurately, and audio feedback plays as expected.

Test Case 3: Detect Text in the Live Camera Feed.	
Functionality:	Test OCR with real-time camera feed.
Test Data:	Live camera feed with text elements.
Expected Result:	Continuous real-time text recognition, displayed text with audio feedback.
Actual Result:	Camera feed starts, text is extracted and displayed accurately, and audio feedback plays as expected.

Test Case 4: Detect Object in an Uploaded Image.	
Functionality:	Test YOLO5 accuracy with user-uploaded images.
Test Data:	Image uploaded by the user.
Expected Result:	Extracted objects displayed, audio feedback plays.
Actual Result:	Image uploaded successfully, object are identified and displayed. Audio feedback functions correctly.

Test Case 5: Detect Object in a Captured Image.	
Functionality:	Test YOLO5 with image captured by the platform.
Test Data:	Image captured by user.
Expected Result:	Image capture successful, objects identified displayed with audio feedback.
Actual Result:	Capture function works, objects are extracted accurately and audio feedback plays as expected.

Test Case 6: Detect Objects in the Live Camera Feed.	
Functionality:	Test YOLO5 with real-time camera feed.
Test Data:	Live camera feed with object elements.
Expected Result:	Continuous real-time object detection recognition, displayed text with audio feedback.
Actual Result:	Camera feed starts, objects extracted and displayed accurately, and audio feedback plays as expected.

Chapter 5: Evaluation

5.1] LIMITATIONS

While ‘WeSee’ has the potential to be a valuable tool for visually impaired users, it's important to acknowledge the limitations inherent to the current development stage or technology in general.

Acknowledging these limitations allows for a more realistic assessment of the project's scope. So, continuous evaluation is essential for addressing these limitations and ensuring the project's long-term success and sustainability.

Here's a breakdown of some potential limitations to consider:

- **Accuracy and Reliability**

The accuracy and reliability of text recognition and object detection algorithms varies depending on factors such as image quality, lighting conditions, and complexity of scenes. The project encounters challenges in accurately identifying text or objects in certain situations, leading to potential errors or misinterpretations.

- **User Interface Complexity**

The complexity of the user interface (UI) and navigation flow within the application may present challenges for some visually impaired users. That is, ‘WeSee’ is not the most suitable platform for the individuals at the very end of the spectrum who suffer complete vision loss.

- **Limited Device Compatibility**

‘WeSee’ is a web application deployed locally and so, it is functional only on a physical computer, yet. This makes it less commercial and convenient due to lack of its portability unlike a mobile application on smartphones.

- **Hardware Dependence**

The effectiveness of the application is influenced by the hardware capabilities of the devices used by the users. Devices with limited camera quality or processing power impact the performance of real time text recognition and object detection functionalities.

- **Privacy and Data Security**

The project doesn’t address concerns related to privacy and data security yet, especially when handling sensitive information such as images, text, or user interactions. Implementing robust data protection measures and compliance with privacy regulations will be essential to build trust among users.

- **Bias in Algorithms**

Object detection and text recognition algorithms can inherit biases from the data they are trained on. It's important to be aware of potential biases and strive for fair and inclusive functionality.

- **Visual Reliance**

At its core, ‘WeSee’ relies on processing visual information (images or video) to perform text recognition and object detection. Blind users, by definition, lack visual perception, so they cannot directly benefit from the core functionalities of the application in their current form.

However, addressing these limitations through continuous testing, user feedback, technological advancements, and proactive measures can enhance the overall performance, usability, and accessibility of your ‘WeSee’ project for visually impaired users.

5.2] ADDITIONAL CONSIDERATIONS

Innovative features can set ‘WeSee’ apart and provide added value to users. Additional considerations are the features that can go beyond standard functionalities and aim to address specific needs or offer novel experiences.

By incorporating these unique features into the project, a compelling user experience can be created while addressing their specific preferences.

The possible additional consideration and features for ‘WeSee’ are:

- **Scene Description with Context**

‘WeSee’ can provide richer descriptions that go beyond just object names and include details like scene type (kitchen, living room), activities taking place, or potential hazards (e.g., "sidewalk with a parked car blocking half the path").

- **Customizable Output Formats**

Providing options for users to personalize the output based on their needs. This could include text size adjustments, voice selection for text-to-speech, or haptic feedback patterns for specific situations.

- **Learning User Preferences**

Allow users to train ‘WeSee’ to recognize specific objects or locations that are relevant to their daily lives (e.g., medications, favorite landmarks) and also, create communities with specific areas of interest accordingly.

- **Gesture Control**

Integrate gesture recognition technology to detect and interpret user gestures, enabling intuitive and immersive interactions for blind people.

- **Voice Control**

Explore the possibility incorporating voice recognition and accepting voice commands as input. This will led to ‘WeSee’ providing a combination of audio descriptions and haptic feedback for a more accessible user experience.

- **Chat Bot Integration**

Incorporating a ChatBot into ‘WeSee’ would be a valuable way to enhance user experience and accessibility. A chatbot can provide a more natural and intuitive way for users to interact with ‘WeSee’ compared to traditional buttons. Users can ask questions in plain language about their surroundings or how to use the application's functionalities easily.

Hence, these ideas provide a springboard for brainstorming unique functionalities that can elevate ‘WeSee's’ capabilities and make it a truly innovative assistive technology tool.

5.3] FUTURE SCOPE

Regarding the future, ‘WeSee’ holds immense promise for the future of assistive technology. By building upon its current foundation, it can explore exciting avenues to broaden its capabilities and impact.

The Potential Impact of ‘WeSee’:

For empowering the future for visually impaired users, ‘WeSee’ has the potential to create a significant positive impact on the lives of visually impaired individuals in numerous ways.

- **Increased Independence and Autonomy**
  - ~ Enhanced Safety Awareness
  - ~ Navigating Daily Tasks
  - ~ Improved Access to Information
- **Educational and Professional Opportunities**
  - ~ Enhanced Learning Experiences
  - ~ Expanding Career Options
- **Social Interaction and Community Engagement**
  - ~ Breaking Down Communication Barriers
  - ~ Greater Participation in Activities
  - ~ Raised Awareness
- **Overall Improved Quality of Life**
  - ~ Increased Confidence and Self-Esteem
  - ~ Promoting Social Inclusion

Chapter 6: Conclusion

6.1] CLOSING STATEMENT

To conclude, the ‘WeSee’ project successfully developed a foundational application for text recognition and object detection. This report detailed the functionalities, design choices, and potential future directions for the project.

‘WeSee’, a Stepping Stone for Assistive Technology:

The core functionalities of ‘WeSee’ empower users to interact with visual information through text recognition and object detection. Accessibility considerations were a key focus during development, ensuring the application caters to users with varying visual impairments.

Looking ahead, ‘WeSee’ presents a promising platform for further development. The potential for advanced scene understanding, object interaction recognition, and integration with other assistive technologies offers exciting possibilities. Additionally, exploring solutions for users with diverse accessibility needs can broaden the application's reach and impact.

By continuing development with a focus on accessibility and user needs, ‘WeSee’ has the potential to become a valuable tool in the assistive technology landscape. It can empower users to navigate their surroundings with greater independence and contribute to a more inclusive world.

A Rewarding Journey and a Promising Future:

This report serves as a springboard for future iterations of ‘WeSee’. The project's findings and conclusions provide a valuable foundation for ongoing development efforts aimed at creating an impactful assistive technology.

6.2] 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_accessibility_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

***