**Smart Sight App:**

**Libraries Used In Project:**

Core Backend Components (Python):

**app.py:**

Main Flask application that handles all HTTP routes and API endpoints. It coordinates between different services and serves the web interface.

**openai\_service.py:**

Handles image analysis and object recognition. It includes functions for

Image preprocessing

Object detection

Scene analysis

Navigation guidance

Text recognition

**voice\_service.py**:

Manages speech-to-text and text-to-speech conversions using:

Google Speech Recognition for voice input

gTTS (Google Text-to-Speech) for voice output

Command intent detection

**chatbot\_service.py**:

Provides conversational AI capabilities with:

Natural language understanding

Response generation

Context management

Knowledge base integration

**models.py:** Defines database models using

**SQLAlchemy:**

ChatbotResponse: For storing predefined responses

UserQuery: For logging user interactions

KnowledgeBase: For storing learned information

**web\_search.py:** Implements web search functionality to answer user questions using:

Web scraping

Content extraction

Result processing

**Frontend Components (JavaScript):**

**static/js/main.js:** Core frontend functionality:

App initialization

UI event handling

Camera integration

Audio feedback

Accessibility features

**static/js/voice-commands.js:** Voice command processing:

Speech recognition setup

Command interpretation

Visual feedback

Continuous listening mode

**Emergency mode**

static/js/camera-handler.js: Camera functionality

static/js/audio-utils.js: Audio processing

static/js/knowledge-base.js: Knowledge management

**User Interface (HTML/CSS):**

templates/index.html: Main web interface with:

Voice activation area

Action buttons

Feedback display

**Accessibility features :**

static/css/styles.css: Main styling

static/css/eye-theme.css: Visual theme styling

Key Features:

* Voice Control: Users can control the app entirely through voice commands
* Object Recognition: Identifies objects and describes surroundings
* Navigation Assistance: Provides guidance based on camera input
* Text Reading: Extracts and reads text from images
* Emergency Mode: Quick activation of high-visibility emergency display
* Knowledge Base: Learns from user interactions
* Accessibility: Screen reader support and high-contrast interface
* Continuous Listening: Hands-free operation mode
* Technical Architecture:
* Web Framework: Flask (Python)
* Database: SQLAlchemy
* Frontend: Vanilla JavaScript
* Speech: Web Speech API + Google Services
* Image Processing: Custom implementation
* Deployment: Set up to run on port 5000 on Replit

The app is designed to be fully accessible and provides a voice-first interface while maintaining touch controls as a backup. It runs as a web application that can be accessed through any modern browser.