NUST

NATIONAL UNIVERSITY OF SCIENCES & TECHNOLOGY

Project Progress Report -1

VocalXpert

AN INTELLIGENT VOICE ASSISTANT

Submitted By:
Student Name: Ghulam Murtaza Khan
Course: BEE – 61 C

Submitted To:
Lt Col Muhammad Imran Javaid

ABSTRACT

The World of Artificial intelligence (AI) has greatly evolved, advancements, and enhancements in the past decade, changing how the people interact with computers for their regular task and operations. One of the great examples is voice assistants, which helps to make work easier, automated and save time. This project will introduce VocalXpert, an intelligent voice assistant designed and will develop to handle and operate everyday tasks with accuracy, ease and simplicity. VocalXpert will be Created using Python, VocalXpert uses the Speech Recognition API to understand and act on spoken commands. It will also include facial recognition powered by OpenCV to ensure and validate that only authorized users can access and control it. In VocalXpert, this feature adds an extra layer of security while making the experience more personal. The assistant (VocalXpert) will have a user- friendly interface built with Tkinter. This interface let the users to choose between voice commands or text based commands, making it simple ease of use and convenient for everyone. VocalXpert will offers many valuable and important features. Users could easily send their emails, WhatsApp messages; adjust volume or brightness, and even use keyboard shortcuts like closing and switching the Windows shortcut keys and more like that. It will also allow you to open applications like the command prompt, IDEs, Paint, Calculator, Control Panel, and Notepad, control media playback, and setting reminders to stay organized. To find information, VocalXpert will search through Google or Wikipedia or open websites like and YouTube using just your voice. With these features, VocalXpert becomes an intelligent and helpful companion, making everyday tasks faster and easier to manage. Me and My Group will try our best to submit it in the given timeline. We aill make every effort to complete it within given time and upload it, Despite being more straightforward than some state-of-the-art voice assistants, this project demonstrates the vast potential of AI technologies in simplifying daily tasks, enhancing the user experiences and productivity and save time, and fostering more efficient human-computer interactio

CHAPTER - 1
PROGRESS

Requirement and Information Gathering

We have completed the Requirement Gathering. We divide the Requirement Gathering into different Parts and then each group member has completed its Task. Broadly we divide into 2 parts, the backend part which includes the functionality and the gathering the information about the which are the Best APIs and free ones. Other part is to research about the Frontend Design of our App, how it looks and how it will interact with the user. Me and Capt. Asim are working on the Part 1 and the Capt. Bilal and the Huzaifa Kahut are working on the 2nd Part.

Proceedure

As usage of AI tools are everywhere and they are very help full in our daily lives. Now they became as a part of our life. We read different articles that are talking about the voice recognition and we have analyzed different Previously created Voice Recognition Apps and we have tested them with different tests that we have defined and devised, from these tests we concluded their Pros and Cons. From this we Get an idea about how we can make our App VoiceXpert different and unique from others. Our Next Step is to find the best Possible Technologies that can full fill our Needs. So, we came up with a clever idea to choose the best options, so we open the GitHub and searched the apps like us which has these features. then we run these projects locally and analyze their code quality we use different AI tools also to analyze the code quality and the modules and technologies used.
For the APIs our goal is to find such APIs that are free or have free tiers so we selected the OpenWeather API and for the News we have selected the News API.

Findings

We have selected the Python as the Primary Language as it supports vast number of libraries and best known for its flexibility and useability.
We are using the OpenCV (for faicial recognization) , Pyaudio (for the audio processing), pyautogui (for automating the system). And for the frontend we are using the Tkinter as it is easy to use and python native.
Below I will discuss every aspect of the whole project in detail.

Design Formulation and Planning

We have successfully completed the planning and the plane and all things are discussed below and explained in a well manner and only here I am giving an overview of what we have done yet. The planning is very important for any project whatever it's a software project or any other Project. For Its successful completion and execution, the Planning Plays a central role. To became a good developer, you have to became a good planner.
We use all possible techniques and tools we have available to make best possible plane for our App Project.

In below Chapters, you will Find our plane and how we can execute our plane to achieve the best performance and what is our project. Diagrams our time routines and our Goals.

CHAPTER - 2

SOFTWARE PROJECT MANAGEMENT PLAN (SPMP)

SOFTWARE PROJECT MANAGEMENT PLAN (SPMP)

Introduction

The Software Project Management Plan for VocalXpert lays out what the project aims to achieve, its boundaries, the resources needed, the timeline, and how to handle risks. It keeps everyone in the loop and highlights the need to check out customer calls to action (CTAs) for smooth communication. The main idea is to create VocalXpert, a voice assistant that makes life easier by boosting productivity and simplifying everyday tasks on personal computers, like opening and closing windows, taking screenshots, launching apps, and browsing websites.

Project Overview

VocalXpert is an innovative AI-powered voice assistant uniquely built with Python to enhance human-computer interaction through rule-based intelligence. Utilizing the SpeechRecognition API, it converts speech to text, allowing users to perform tasks like sending emails, normal chatting to users with greetings, general knowledge, and more like that, controlling system settings, launching applications, and opening websites and searching through Google and youtube. This project meets the demand for intelligent systems that automate routine tasks and improve productivity. Also, VocalXert features a face recognition system through OpenCV for secure user authentication. VocbalXpert has a smooth, user-friendly GUI built with Python's Tkinter , supporting both Speech(voice) and text input for flexible interaction and usability.

Project Deliverables

The VocalXpert project will develop a set of features and functionalities that will help achieve its objectives and guarantee its success. The primary deliverables include.

1. **Voice Recognition Module**: A reliable SpeechRecognition API voice recognition system that converts audio to text, allowing a user to use voice commands to communicate with VocalXpert.
2. **User Authentication System**: A computer can use the face recognition system, which uses OpenCV, to access the voice assistant. Users can enter their profile details on the system and teach the assistant about their faces, so when they restart their application, they are authenticated automatically.
3. **Graphical User Interface (GUI)**: With support for input via voice and text, Tkinter makes for a user-friendly interface. The user interface (GUI) is made intuitive so the users can easily interact with the assistant without any problems, and the user experience is smooth and accessible.
4. **Functionality Implementation**. A lot of things that make the assistant more useful include.

- Users can send emails and WhatsApp messages as voice using this application.
- Users can search on Google and Wikipedia using voice commands.
- User's voice controls system settings such as volume, brightness, and keyboard shortcuts.
- Users can play your music and videos and access other media content.
- Up-to-date weather reports and the ability to set reminders are convenient and useful app features.
- Having basic conversations with the user to make the assistant more human-like.
- Managing windows enables you to open, close, minimize, maximize windows, and take screenshots.
- File Management: Enabling users to voice-dictate file saving, selecting, and deleting actions.
- Access website tools of famous brands like Google and YouTube with simple voice commands.
- Users can auto-type in the application or Google search if they speak what they want typing.

5. **Documentation**: User manuals and documentation to help users use VocalXpert and future developers with what they require to extend and maintain this system.
6. **Testing and Evaluation**: Reports should be revised to analyze results. I will conduct extensive testing and provide evaluation reports to determine whether testing is up to performance. Similarly, We will conducting the usability tests and user satisfaction surveys to refine the system further.

Project Organization

The VocalXpert project is organized to streamline development and efficient management. Basically, Me Ghulam Murtaza and my Group Members Capt. Asim Iqbal, Bilal Zaib, Huzaifa Kahut, we are developing this Application. We are handling all aspects including development, planning, testing and deployment. We are using agile methodology in which We planned to break project into smaller parts and then incremental model in which we are continuous testing to align progress with objectives.

Software Process Model

For our VocalXpert project, we selected the Incremental Model as its software development framework. Incremental software process model approach involves creating the system in small, manageable parts. Each part follows a sequence of designs implementation, and testing steps, contributing to the overall functionality of VocalXpert. Benefits of the Incremental Model in VocalXpert:

- Dividing the project into smaller increments facilitates the early delivery of key features to users, allowing for timely feedback that enhances development. And Dividing our Work with respect to every group member Skills.
- The Incremental Model integrates stakeholder feedback, ensuring the system meets user preferences.
- The incremental development approach enables early issue detection, allowing for proactive management and minimizing project delays.

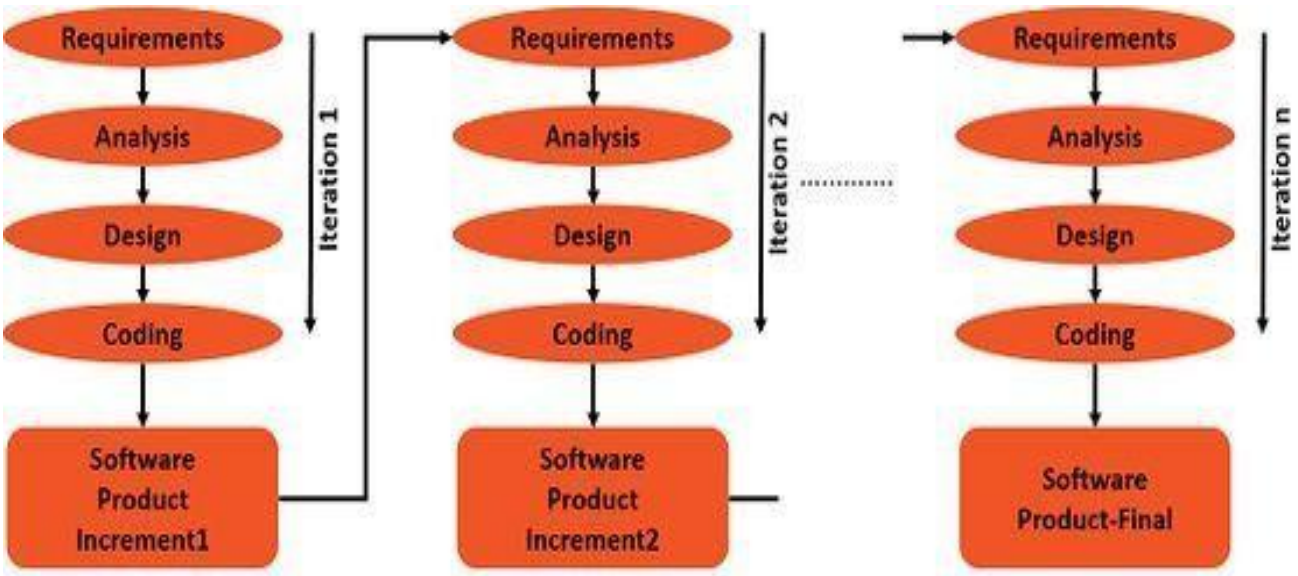

Figure 1 Incremental Model

1.1.1 Roles and Responsibilities

As Me and my Group, we are developing VocalXpert project, So, we will divide key responsibilities, including project management, development, testing, and deployment but not like in a schematic manner, so before doing some thing we all discuss with each, if anyone face any problem, we together tackle that problem to find the solution. That's why we didn't call anyone as Sole Project Manager or lead developer as we all works and works like a team.

- **Project Manager:** We all together with mutual discussion, define the project goals, set timelines, and oversee the execution of all tasks to ensure the project is completed on schedule.
- **Lead Developer:** As I previously mentioned we will divide the project into different small parts and each person do his own job and if he gets any trouble, we all work together to solve that problem. This is how we are developing and planning to develop all features of VocalXpert, including voice recognition, face authentication, GUI design, and system automation, and more.
- **Tester:** I will perform testing of the VocalXpert along with my team or group to ensure the assistant functions correctly and address any issues that arise during the process.
- **Documentation:** I will create the project's(VocalXpert) technical(project report) and user documentation(SRS) to guide development and assist users later on.

Tools and Techniques

- **Programming Language:** For VocalXpert, We are using Python as primary language for developing the voice assistant,, leveraging its vast libraries, flexibility, and strong support for the artificial intelligence applications.

- **Speech Recognition API:** For VocalXpert, this API converts spoken commands into text, forming the core of the voice assistant's functionality.
- **Tkinter:** For GUI of our project VocalXpert, we are using TKinter that is a toolkit and famous library for creating basic and easily developing the GUI. Tkinter will provides a responsive, user-friendly environment that supports both voice and text input to improve user accessibility.
- **OpenCV:** OpenCV, an open-source computer vision library. It is used to implement the face-recognition feature for secure user authentication to ensure a personalized experience for authorizing users.
- **PyAudio:** This library will process audio input, enabling the system to capture and recognize speech commands.
- **pyautogui:** It will automate system tasks such as taking screenshots, controlling system settings (volume, brightness.), and managing files (saving, selecting, deleting).
- **Integrated Development Environment (IDE):** We are using Visual Studio Code as it provides vast extensions, different AI tools (especially the Copilot for searching any module or error very easily) , easy integration with different environments that will be very helpful in efficient coding, debugging, and testing.
- **Testing Tools:** Frameworks like unit tests or integrate testing will be utilized for automated testing to ensure the system's functionality and reliability as needed through developing.

Project Management Plan

The Project Management Plan for the VocalXpert project outlines and specifies the tasks, descriptions, deliverables, milestones, and resources needed to ensure the systematic approach to project development. This plan is essential for tracking the progress, managing time effectively, and ensuring the project remains on schedule of the project.

Tasks

The given below tasks for the VocalXpert :

1. **Requirements Gathering:** We identified user needs and outlining key features for the voice assistant based on social media and the Internet. There is a vast amount of the data that you can collect about this on the internet.
2. **System Design:** We also planned the systems architecture design and the user interface design.
3. **Implementation:** We are developing the code for VocalXpert main functionalities, including speech and user authentication through face recognition.
4. **Testing:** We will test the system for bugs, optimal functionality and ensuring all features work correctly and bug free as possible. For that we will use different techniques and frame Works that are discussed in earlier in this document.
5. **Documentation:** Me with the help of my group and especially Capt. Asim, we will try to make the Documentation Guide or User Manual of it so anyone can use it very easily.
6. **Deployment:** Basically, it's a Desktop App, so we will make it's exe file so it can easily deploy on windows and in later we will introduce cross-platform application of it.
7. **Feedback and Iteration:** We will Collect user feedback via the forms and will make necessary improvements in later versions.

Deliverables and Milestones

The Following below Tables is clear understanding of project VocalXpert Deliverables and Milestones.

Milestone	Description	Deliverable
Milestone 1	Completion of Requirements Gathering	Documented requirements specification
Milestone 2	Finalization of System Design	System architecture and UI design mockups
Milestone 3	Completion of Backend Phase	Functionality voice assistant
Milestone 4	Creating the Front-End	Creating the GUI
Milestone 5	Integrating APIs	Fully Functional Voice Assistant with GUI
Milestone 4	Successful Completion of Testing	Testing reports and resolved issues
Milestone 5	Documentation Completion	User manuals and technical documentation
Milestone 6	Deployment of VocalXpert	Live voice assistant available for users

Table 1 Project Deliverables and Milestones

Resources Needed

To successfully execute the VocalXpert project, the following resources will be required:

Human Resources

- Me Ghulam Murtaza Khan and my Group which includes Capt. Asim Iqbal, Capt. Bilal Zaib and Huzaifa Kahut for design, implementation, testing, and documentation.

Technical Resources

- A computer with sufficient processing power for development and testing.
- Software tools, including:
    - Python and relevant libraries (e.g., SpeechRecognition, Tkinter).
    - IDE (e.g., Visual Studio Code).

Financial Resources

- We are using Open-Source Software's for our project. And we aimed its budget as zero as we totally relay on the open-source software, APIs and modules.

Dependencies and Constraints

Dependencies

In project management, dependencies define the sequence of tasks, ensuring one task is completed before another begins. For the VocalXpert project, the dependencies are:

- **Requirements Gathering → System Design:** A clear understanding of requirements is essential before proceeding with system design.
- **System Design → Implementation:** Implementation relies on the finalized system design to align the developed features with the architecture.
- **Implementation → Testing:** Testing requires a fully implemented system to evaluate functionality and performance.
- **Testing → Documentation:** Documentation is based on the final, tested version to describe the features and usage accurately.

Constraints:

Constraints are the limitations that may impact project execution. For VocalXpert, these include:

- Time: The project must be completed within the 10 to 12-week timeline.
- Resources: As a we are not much skilled programmers, we are going through our early learning process so with our limited skills may affect the scope of implemented features.
- Technical Limitations: The technologies used, like the SpeechRecognition API and Tkinter, may impose functional or performance constraints.

Risks and Contingencies

Risks

Identifying the potential risks is very important for successful project management. Some main risks are given below:

- **Technical Challenges:** Difficulty in accurately implementing speech recognition could hinder the core functionalities and operations.
- **Scope Creep:** The temptation to add more features beyond the initial plan could delay project completion.
- **Time Management:** Balancing project development with other responsibilities may impact deadlines.

Contingencies

To address and aviod these given risks, the given below contingency plans are:

- **Regular Checkpoints**: Schedule weekly reviews to assess the progress and make adjustments as necessary to stay on track.
- **Feature Prioritization:** Clearly define and prioritize core features, deferring any non-essential functionalities to future iterations if needed.
- **Time Buffer:** Allocate extra time within the project timeline to accommodate unforeseen challenges or delays.

Timetable

Task	Start Date	End Date	Duration
Requirement Gathering	**25-July-2025**	**05-Aug-2025**	**11 days**
Design Formulation and Planning	**06-Aug-2025**	**6-Sep-2025**	**34 days**
Partial Development	**7-Sep-2025**	**1-Nov-2025**	**41 days**
Testing	**2-Nov-2025**	**5-Nov-2025**	**3 days**
GUI Development	**6-Nov-2025**	**24-Nov-2025**	**18 days**
Integration of GUI and Backend	**25 – Nov - 2025**	**29-Nov - 2025**	**4 days**
Testing	**30-Nov-2025**	**2-Dec-2025**	**3 days**
Deployment	**03-Dec-2025**	**05-Dec-2025**	**5 days**

Table 2 Time Table

Project Gantt Chart:

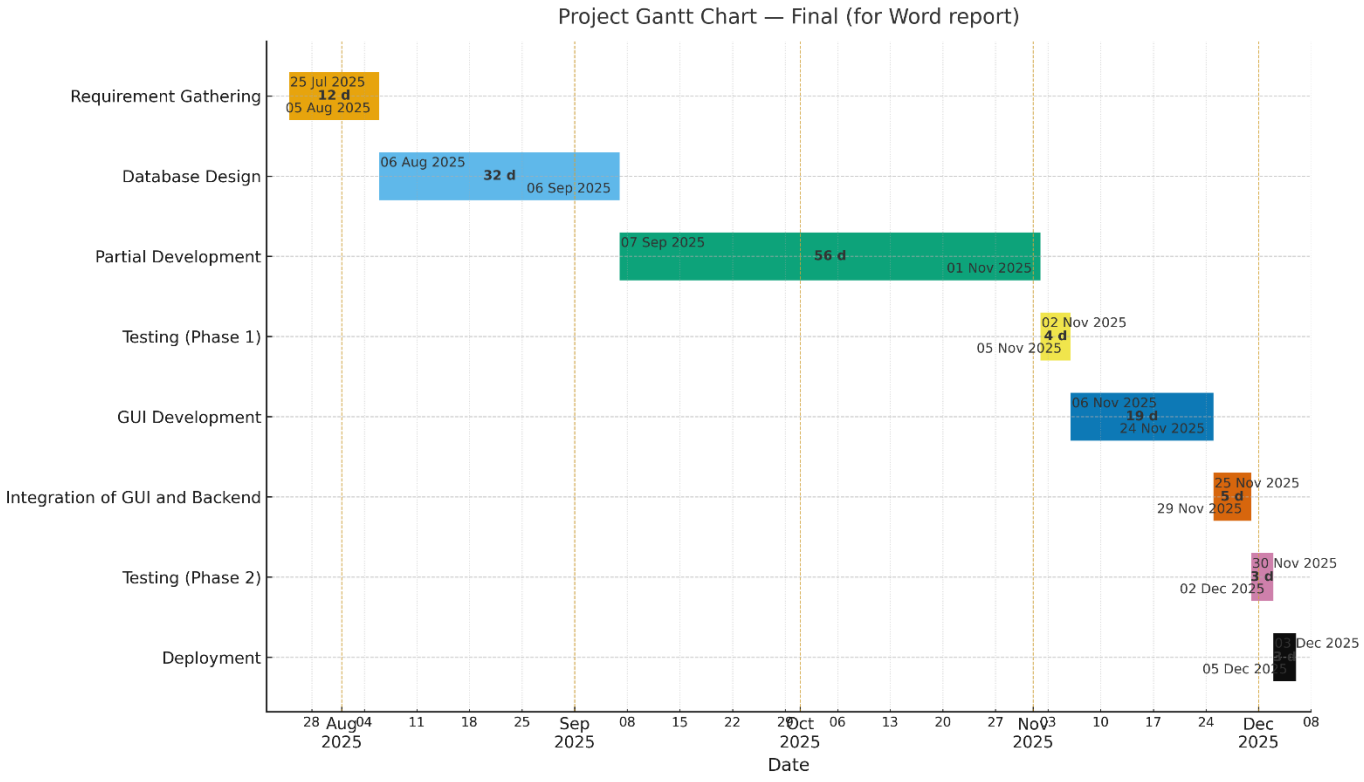

Figure 2 Gantt Chart

CHAPTER - 3

PROJECT BACKGROUND AND UNDERSTANDING

PROJECT BACKGROUND AND UNDERSTANDING

Project Direction

The VocalXpert project is designed to develop to an advanced and an intelligent voice assistant to interpret efficiently and execute user-specific commands through voice recognition at fast pace and with great accuracy. The main focus is on building an intuitive, hands-free interface that enhances the users ease of use and productivity and convenience. By utilizing artificial intelligence techniques and modern technologies like SpeechRecognition and face recognition, the project aims to meet the rising demand for reliable, efficient, accessible, future proof and digital assistants. The ultimate goal is to provide a seamless, user-friendly environment of an application that simplifies everyday tasks, offering a more accessible way for users to interact with technology and improving their daily tasks.

Background

In recent years, voice-activated assistants such as Amazon's Alexa and Apple's Siri have significantly transformed how users interact with technology. These systems utilize advanced algorithms to process and understand spoken language, allowing users to perform tasks hands-free. However, many of these solutions are proprietary and lack the flexibility for personalization. This project VocalXpert addresses this gap by developing a customizable voice assistant tailored to individual needs, particularly in educational and professional contextsThe shift to remote work and digital communication has increased the need for productivity tools, boosting the value of personalized voice assistants

Technology used

Technology	Description
Python	The main programming language; easy to use with many libraries suited for building voice assistants.
SpeechRecognition API	Converts speech into text, enabling interaction with VocalXpert via voice commands and natural language.
Tkinter	A Python library for creating a user-friendly graphical interface for voice or text interaction.
OpenCV	An open-source library for facial recognition, ensuring security by verifying users' faces.
PyAutoGUI	Automates GUI tasks, such as opening applications and adjusting volume.
Wikipedia and Google APIs	Allows users to search for real-time information through voice commands.
pyttsx3	A text-to-speech library that audibly reads responses to users.
Pandas and Plotly	Used for handling and displaying data, particularly for analytics and task timelines within the assistant.

Table 3 Technologies Used in Project Development

CHAPTER - 4

SOFTWARE REQUIREMENTS SPECIFICATIONS (SRS)

SOFTWARE REQUIREMENTS SPECIFICATIONS (SRS)

Introduction

In this project, the SRS for VocalXpert project outlines what's needed for the voice assistant app, which is designed and developed to make the user interactions and experience easier through speech recognition. Also this SRS acts as a guide and blueprint for the overall design and implementation of the system, ensuring that all the essential features and security measures are in place for smooth and reliable operation of product.

Project Overview

VocalXpert is a Python-based an intelligent voice assistant that allows users to complete tasks by speaking or text to your device. It uses the SpeechRecognition API to convert your voice into text for hands-free interaction via a simple Tkinter interface and also with keyboard inputs for seemless interaction and for the ease of users.With features like face recognition for security, VocalXpert enables you to send emails, WhatsApp messages, and automate tasks like adjusting volume and brightness. You can also launch applications like command prompts and Notepad quickly. Furthermore; it helps with the media management, web browsing through voice commands and text commands, weather updates, and setting up desktop reminders. VocalXpert streamlines tasks, minimizes typing, and enhances your tech experience.

Specific Requirements

External Interface Requirements

VocalXpert connects and integerates and uses with hardware like microphones and cameras, software such as the SpeechRecognition API, weather, Wikipedia APIs and GUI libraries, and user input/output devices. These connections and integrates to ensure efficient and optimal communication between the user and the system.

User Interfaces

VocalXpert provides a user-friendly interface that supports both voice and text input, allowing users to choose their preferred interaction method. The main user interfaces include:

Voice Interface

- **Speech Recognition:** Converts spoken commands into text for processing.
- **Audio Feedback:** Provides verbal responses to user commands

Text Interface

- **Tkinter GUI:** A graphical interface with buttons, labels, and input fields for user interaction and feedback.
- **Command Input Field:** Users can type commands directly into the interface for immediate execution.
- **Response Display:** Shows results from commands, including searches, weather updates, and reminders.

Face Recognition Interface

- **Face Authentication:** Verifies user identity using facial recognition to provide access to personalized features.

User Interface Diagram

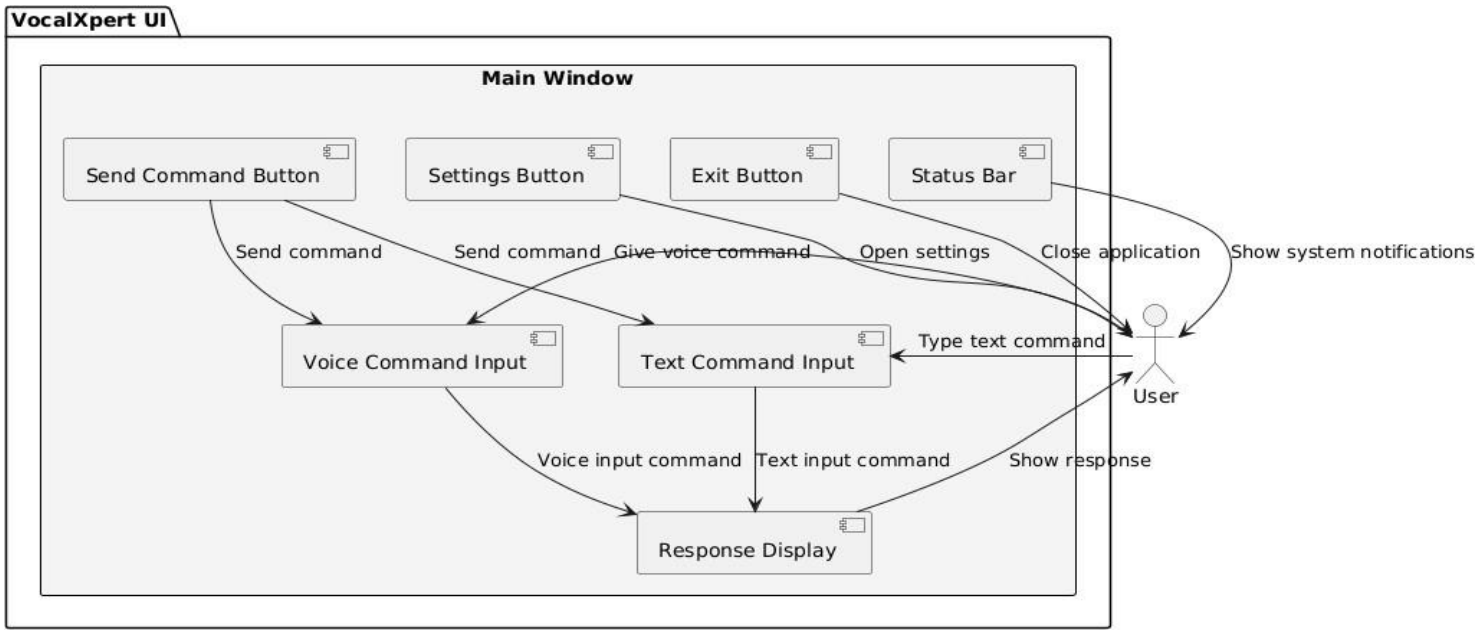

Figure 3 User Interface Diagram

Use Case Diagram

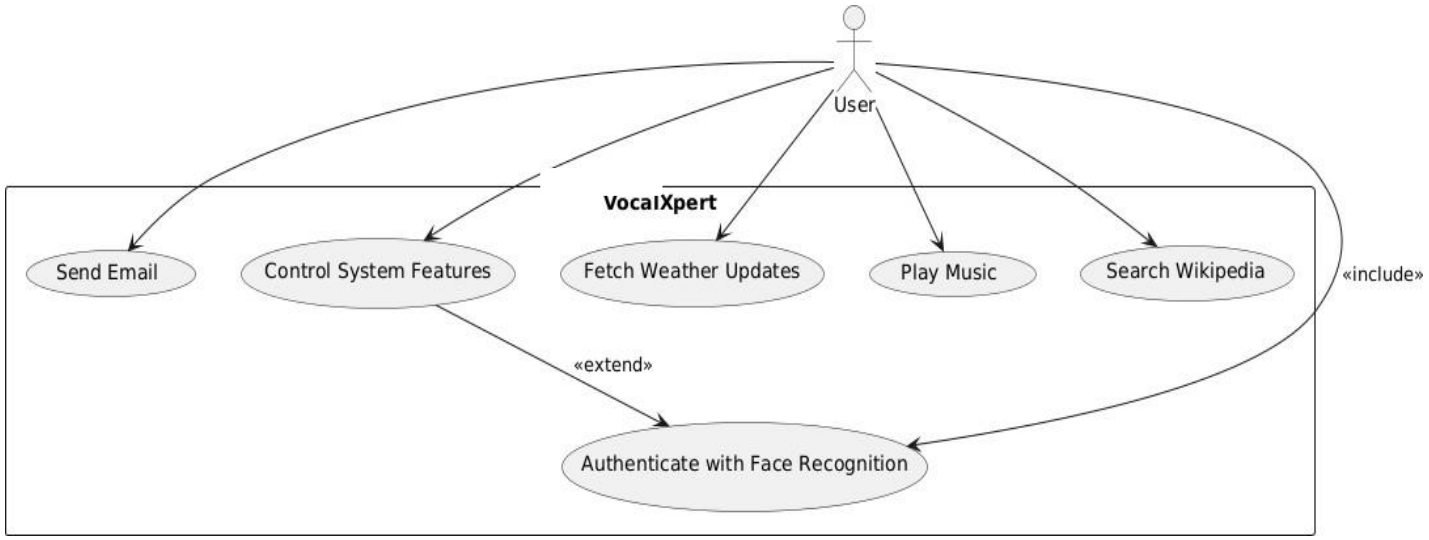

Figure 4 Use Case Diagram

Hardware Interfaces

VocalXpert requires the given below hardware interfaces:

Microphone

- Captures user's voice commands for processing by the SpeechRecognition API.
- Works via the system's audio input interface, typically using a USB or built-in microphone.

Camera (for Face Recognition)

- Captures the user's face for authentication through facial recognition.
- Integrated with the system's webcam, providing real-time video feed for face detection and recognition using OpenCV.

Speaker/Headphones

- Provides audio feedback to the user, confirming actions and delivering responses (e.g., weather updates, task completion).
- Outputs sound through the system's audio output interface.
-

Display (for GUI)

- Presents the Tkinter-based graphical user interface to the user, displaying buttons, input fields, and feedback.
- Utilizes the system's display adapter to show the interface on the monitor.

HARDWARE INTERFACE DIAGRAM

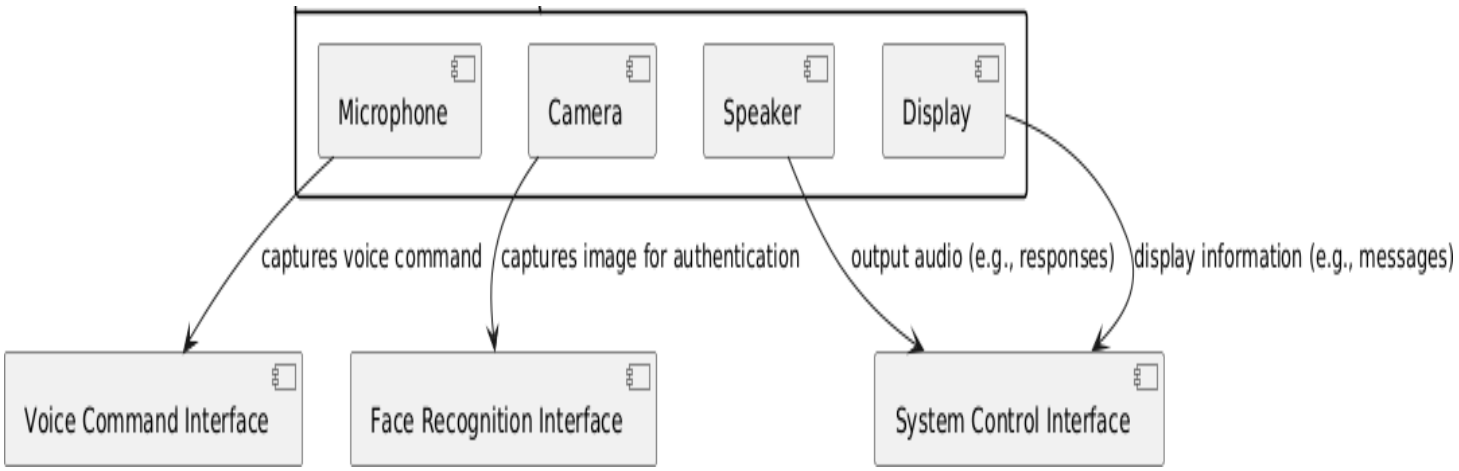

Figure 5 Hardware Interface Diagram

Software Interfaces

VocalXpert integrates with multiple software components to deliver its functionality, ensuring smooth operation and efficient task execution:

SpeechRecognition API

- **Purpose:** Converts spoken language into text, enabling VocalXpert to understand and process voice commands.
- **Integration:** The API listens to user input via the microphone, processes the audio, and converts it into text for further action, such as composing emails, opening applications, or conducting searches.

Tkinter

- **Purpose:** Provides the assistant's graphical user interface (GUI), allowing users to interact with the system through voice and text.
- **Integration:** Tkinter is used to create buttons, text fields, and dialogue boxes that enhance the user experience, enabling task selection, settings adjustments, and more.

OpenCV (Face Recognition):

- **Purpose:** Secures and personalizes the system by authenticating users based on face recognition.
- **Integration:** OpenCV analyzes the video feed from the camera to match the user's face with stored profiles, granting access only to authorized individuals.

PyAutoGUI:

- Automates system tasks, such as taking screenshots, controlling volume, or managing window states.

It allows the assistant to perform system operations like minimizing/maximizing windows, typing, or taking screenshots via voice commands.

External Web Services:

- Provides additional functionalities like weather updates and Google search.
- VocalXpert uses external APIs to fetch real-time weather data, conduct web searches, and provide other dynamic information, enriching the assistant's capabilities.

SOFTWARE INTERFACE DIAGRAM

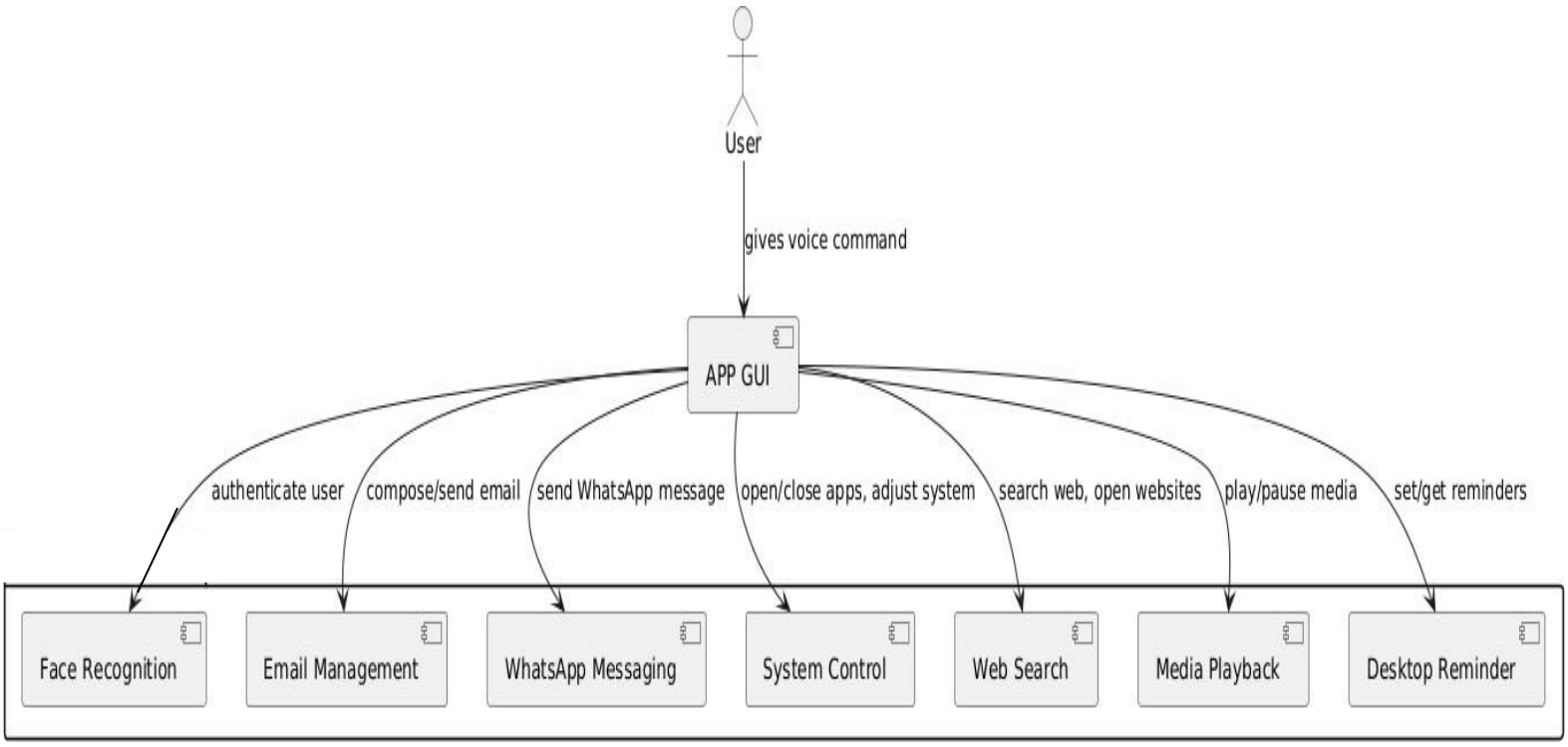

Figure 6 Software Interface Diagram

Communication Protocols

- **HTTP/HTTPS:** Used to access online resources like weather data and websites.
- **SMTP:**For sending emails securely.
- **WebSocket Protocol:** Enables real-time communication, such as WhatsApp messages and live feedback.
- **Inter-Process Communication (IPC):** Connects the GUI with backend services.

Software System Attributes

The below attributes of the VocalXpert software system are:

- **Scalability:** Designed to support adding new features without significant architectural changes.
- **Maintainability:** Built with clean, modular code to ensure easy debugging and updates.
- Usability: Incorporates a user-friendly GUI with intuitive navigation and accessibility

features.

- **Security:** Includes face recognition for secure access and data protection.

Reliability

VocalXpert's reliability is increased through below procedures:

- **Error Handling:** Implementing a robust error detection and exception handling mechanisms, such as handling unrecognized commands gracefully.
- **Testing:** Conducting an extensive and deep unit and integration testing to minimize the bugs and ensure the overall consistent performance.
- **Redundancy:** Critical processes like speech recognition and task execution include contingency methods to handle failures and also handle those for optimum performance of assistant.
-

Availability

VocalXpert's availability is increased by given below:

- **Offline Functionality:** Allows you through core features, such as playing music and launching applications, to work without an internet connection.
- **Cloud Integration:** For features requiring internet, such as weather updates, ensuring reliable online availability.
- **Resource Optimization:** Designed and developed to consume minimal system resources, allowing the software to run smoothly and efficiently on standard or low-end hardwares.
- **Crash Recovery:** Incorporates the mechanisms to recover from unexpected crashes and resume operations without data loss or crash state of system.

Security

VocalXpert's security increased through given:

- **Face Recognition Authentication:** Only authorized users can access the system through face-recognition technology.
- **Data Encryption:** Data encryption of sensitive data such as user credentials and email content for enhancing the privacy of users.
- **Secure Communication Protocols:** Using common HTTPS for web requests and SMTP with encryption for email sending.
- **Command Validation:** Verifying voice commands if needed for before execution to prevent accidental or malicious actions.
- **Access Logs:** Maintains logs of all system activities for audit and troubleshooting purposes

Maintainability

The maintainability of VocalXpert is enhanced and achieved through the given below approaches and techniques:

- **Modular Design:** Each functionality is encapsulated in a separate module, making updates and debugging easier.
- **Documentation:** Comprehensive documentation of code and architecture ensures smooth maintenance and onboarding.
- **Error Logging:** Detailed logs and feedback mechanisms simplifies the issue for diagnosing and resolving.
- **Testing:** An extensive testing of frameworks to ensure code reliability during updates.

Portability

VocalXpert is developed and designed to run seamlessly across given below environments:

- **Cross-Platform Compatibility:** VocalXpert is run on Windows, provided a Python- based and required libraries are installed in future it will also available for linux OS.
- **Hardware Independence:** It depends and works on any system with a microphone, camera, and basic processing capabilities.
- **Portable Packaging:** If required or needed an executable file created using tools like PyInstaller for easy distribution and deployment.

Performance

The VocalXpert performance is achieved through below:

- **Low Latency:** Commands are made to be processed in real-time for immediate execution.
- **Resource Efficiency:** Designed to operate with optimal CPU and memory usage.
- **Multi-threading:** Ensuring the simultaneous execution of tasks without lag and smooth.

Database Requirements

VocalXpert primarily operates without a database, certain functionalities may require in future (not available on current version of VocalXpert) structured data storage:

- **Database Type:** SQLite for local storage, as it is lightweight and portable.
- **Usage:**
  - **User Data:** Stores encrypted login credentials and face recognition details.
  - **Reminder Management:** Saves and retrieves user-defined reminders.
  - **Logs:** Maintains system logs for performance analysis and debugging.
- **Scalability:** The database schema is designed to accommodate future expansions, such as saving user preferences or historical commands.

CHAPTER - 5

SOFTWARE DESIGN DESCRIPTION (SDD)

SOFTWARE DESIGN DESCRIPTION (SDD)

Introduction

The **Software Design Description (SDD)** provides detailed insights into how the system operates internally and externally. It identifies the relationship between components, user interaction mechanisms, and the logical flow of operations.

Design Overview

The VocalXpert system combines advanced speech recognition and task execution in a Tkinter-based GUI. Its modular design allows easy addition of new functionalities. Core features include email handling, system control, and facial recognition for optimal performance.

Key Features of the Design

- Modular structure for extensibility.
- Separation of logic and UI for maintainability.
- Real-time processing with asynchronous support.

Requirements Traceability Matrix

Requirement ID	Requirement Description	Design Component	Verification Method
R1	Voice Command Recognition	SpeechRecognition API	Unit Testing
R2	Task Execution (Email, Control)	Command Processor, Modules	Functional Testing
R3	User Authentication	Face Recognition (OpenCV)	Integration Testing
R4	GUI Interaction	Tkinter GUI Framework	User Acceptance Testing (UAT)

Table 4 Requirement Traceability Table

System Architecture Design

For VocalXpert the modular architecture is selected with three layers, each serving a specific function, making development, testing, and maintenance more efficient.These three layers explains briefy below:

Input Layer

- Responsibilities: In this layer is responsible for capturing user input, both through voice and text.
- Components:

- SpeechRecognition API: this API converts spoken commands into text.
- Text Input Interface (Tkinter): this python library allows users to interact with the assistant via text input in the GUI.

Processing Layer

- Responsibilities: In this layer it infers the input and determines the suitable actions and responces.
- Components:
  - Command Processor: In this process evaluates the user's request and processes commands and actions, such as opening applications, sending emails, or fetching information.
  - Authentication Module: Uses facial recognition (OpenCV) for user authentication to ensure secure access.

Output Layer

- Responsibilities: This layer executes actions based on the processed commands and provides feedback to the user.
- Components:
  - Task Executor: Handles the execution of tasks like sending emails, launching apps, or controlling system functions (volume, brightness, etc.).
  - Tkinter GUI: Provides the graphical interface that shows system status, displays commands, and allows interaction with the assistant.
  -

System Architecture Diagram

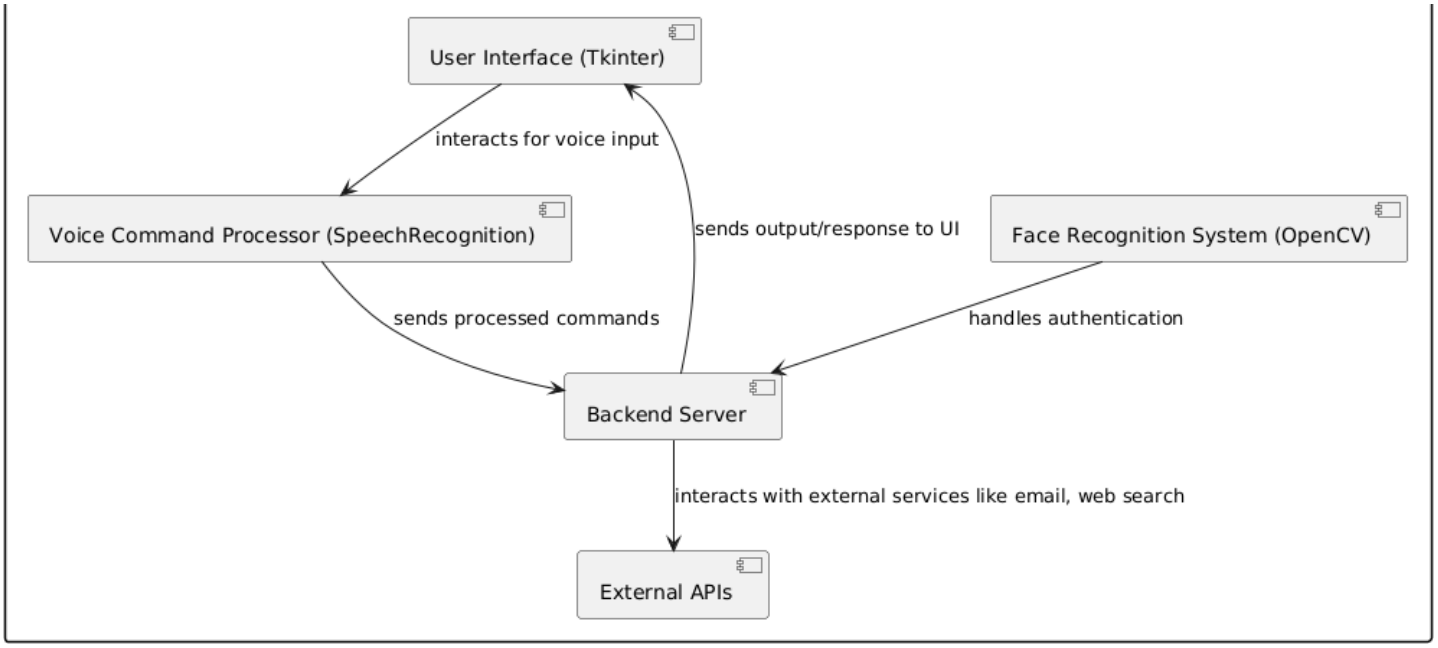

Figure 7 System Architecture Diagram

Chosen System Architecture

VocalXpert's architecture uses a modular layered approach for scalability and easy maintenance. Each layer handles a specific function, ensuring a clear separation of concerns and efficient interaction between components. This design effectively manages different

aspects of the application. It makes the system easier to manage, maintain, and expand in the future. Given below the architecture consists key layers;

Input Layer:

- **Description:** This layer handles the collection of user commands via voice and text input.
- Components:
  - **SpeechRecognition API:** It converts the spoken commands into text for further processing.
  - **Tkinter GUI:** Provides a user-friendly interface for text-based input.
- **Purpose:** To ensure seamless and flexible interaction between the user and the assistant.

Processing Layer

- **Description:** This is the core layer of processes and maps the user inputs to their corresponding actions.
- **Components:**
  - **Command Processor;** Interprets user commands using predefined rules and Natural Language Processing (NLP) for better command execution.
  - **Face Recognition (OpenCV):** Authenticates and validates the user to ensure secure operations.
- **Purpose:** To enable the accurate command interpretation and secure user identification.

Execution Layer

- **Description:** Responsible for executing commands and providing feedback to the user.
- **Components**
  - **Task Executor;** Manages system controls (e.g., opening applications, managing files).
  - **Output Manager;** Communicates the results of actions via the Tkinter GUI or audio feedback.
- **Purpose;** To carry out actions and offer intuitive feedback, ensuring a seamless user experience.

Communication Layer

- **Description**

  Facilitates interaction with external services like email, weather APIs, or web platforms.
- **Components**
  - **API Integrations:** Connects to external platforms (e.g., Google, YouTube, Wikipedia).
  - **Network Manager:** Handles internet-based queries and communication.
- **Purpose**

  To expand functionality by integrating with external resources.

Pros

- **Separation of Concerns**: Each layer has a distinct responsibility;, making the system easier to manage and understand.
- **Scalability**: New features can be added without disrupting existing functionality, ideal for future expansion;.
- **Maintainability**: Independent layers simplify updates or fixes without affecting the entire system;.
- **Reusability**: Components can be reused across different systems or future projects.
- **Flexibility**: Allows easy integration of new technologies and features.
- **Testing**: Simplifies unit testing by isolating components.

Cons

- **Initial Complexity**: Designing clear boundaries between modules adds complexity.
- **Overhead**: Communication between layers can introduce slight performance overhead.
- **Inter-layer Dependencies**: Issues in one layer may affect others, requiring careful management.
- **Debugging Challenges**: Multi-layered systems may complicate debugging processes.
- **Increased Design Effort**: Requires upfront effort in defining the architect

Architecture Diagram

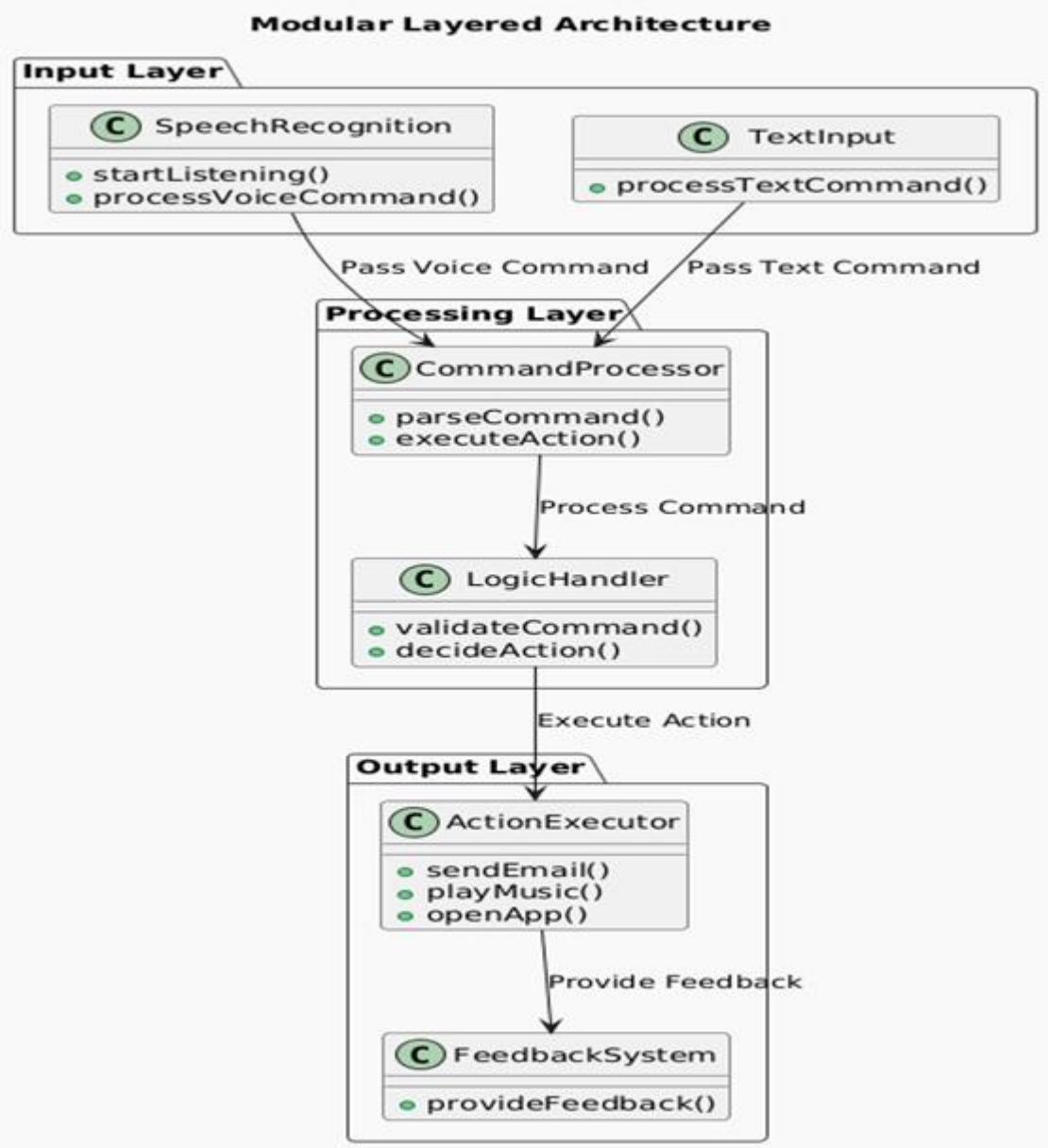

Figure 8 Modular layered Architecture

Discussion of Alternative Designs

During the development of VocalXpert, several architectural designs were considered before finalizing the modular layered approach. Below are the alternative designs and their evaluations:

Monolithic Architecture

Monolithic architecture is a traditional software architecture style in which all system components like (such as the user interface, business logic, and data access layer) are integrated into a single codebase. All application functions are tightly coupled and run as a single process or unit.

Pros

- **Simplicity:** Monolithic systems are relatively easy to develop and deploy due to their unified structure.
- **Performance:** Since all components are within the same process, communication between components is fast.
- **Development Speed:** Because everything is tightly integrated, developers can move faster in the early stages.

Cons

- **Scalability Issues:** As the application grows, the single codebase becomes more challenging to maintain and scale. Scaling the system requires scaling the entire application.
- **Difficult Maintenance:** Changing one part of the system often requires changes to other parts, and testing can become cumbersome.
- **Limited Flexibility:** It is challenging to use different technologies or programming languages for other parts of the system
- **Reason for Rejection:** The lack of flexibility and scalability makes it unsuitable for a project like VocalXpert, which requires future extensibility.

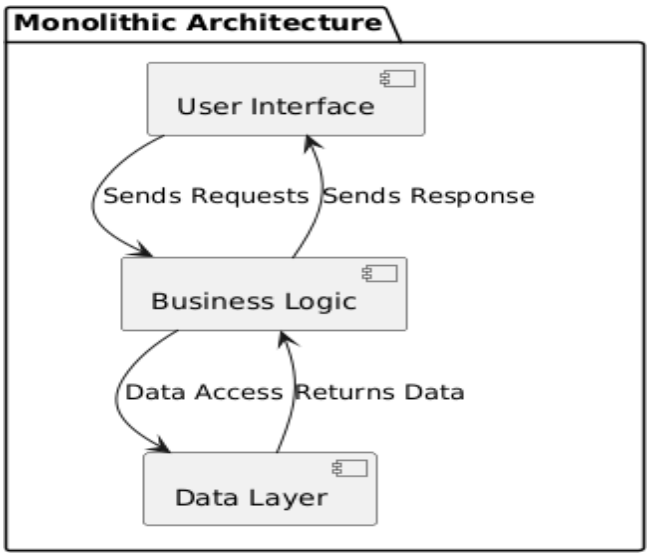

Figure 9 Monolithic Architecture

Client-Server Architecture

The client-server architecture divides the system into two main parts: **Client** and **Server**. The client is responsible for the interaction with the user (providing the interface), while the server handles processing, data storage, and business logic. The client communicates with the server over a network, sending requests and receiving responses.

Pros

- **Separation of Concerns:** The client and server both have clearly defined roles, making the system easier to manage and adopt.
- **Scalability:** Servers can be scaled independently from the clients, allowing for better resource management..
- **Flexibility:** New clients can be added without changing the server-side logic. Different types of clients (web, desktop computers, mobiles.) can use same server.

Cons

- **Network Dependency:** The system relies heavily on the network, and if the connection between the client and server is slow or disrupted, the performance can degrade..
- **Complexity in Communication:** Communication between the client and server may require additional logic, such as message handling, authentication, and encryption.
- **Server Load:** If many clients connect to the server simultaneously, the server can become overloaded and require more resources..
- **Reason for Rejection:** The project is designed as a desktop application with offline functionality, making this architecture less suitable..

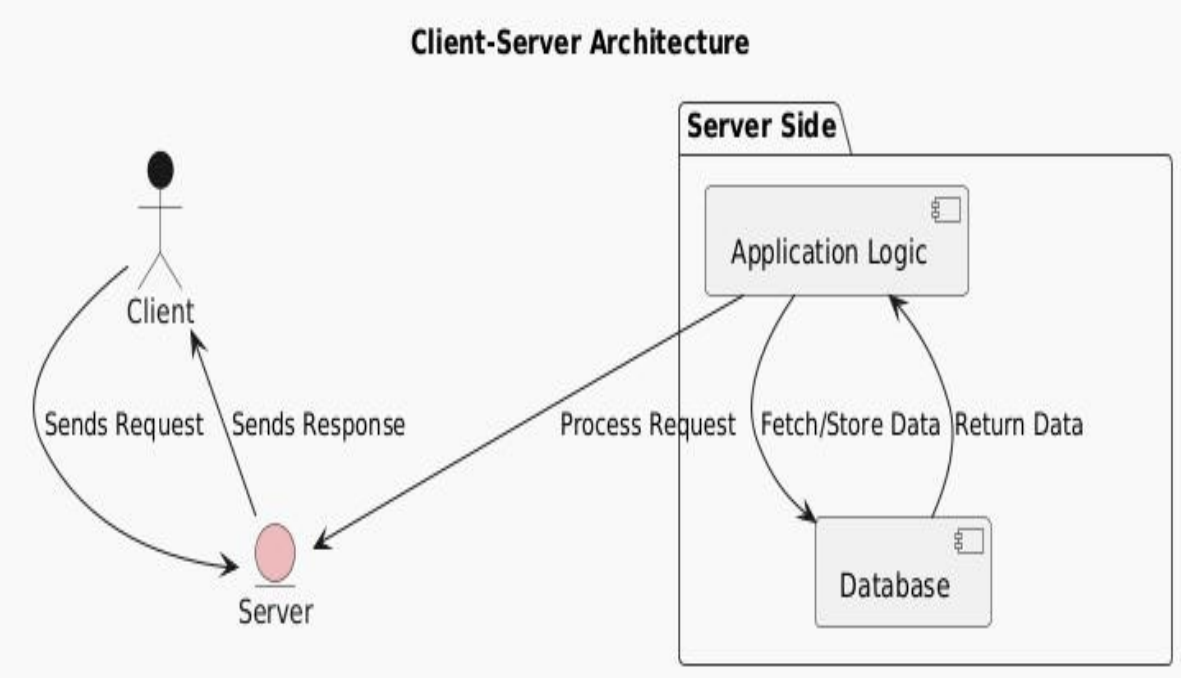

Figure 10 Client-Server Architecture

Event-Driven Architecture

Event-driven architecture (EDA) decouples system components by using events to trigger actions. When a component performs an action or detects a change, it emits an event that other components can handle..

Pros

- **Decoupling of Components:** Components doesn't need to know about each other, allowing them to evolve independently.;
- **Scalability and Flexibility:** The system can scale more easily because events can be handled asynchronously and dynamically adjust to workload changes.;
- **Responsiveness:** An event-driven approach benefits systems that need to react in real time to changes or inputs (such as real-time notifications or updates).

Cons

- **Complexity:** The architecture can become complex, especially when handling large numbers of events or ensuring that events are processed in the correct order.
- **Difficult Debugging:** Tracing and debugging issues can be more challenging since asynchronous events and components are decoupled.
- **Latency:** Event processing might increase the system's response time, especially if multiple events need to be handled in parallel.
- **Reason for Rejection:** The complexity outweighed the benefits for a solo-developed project with a moderate feature set.

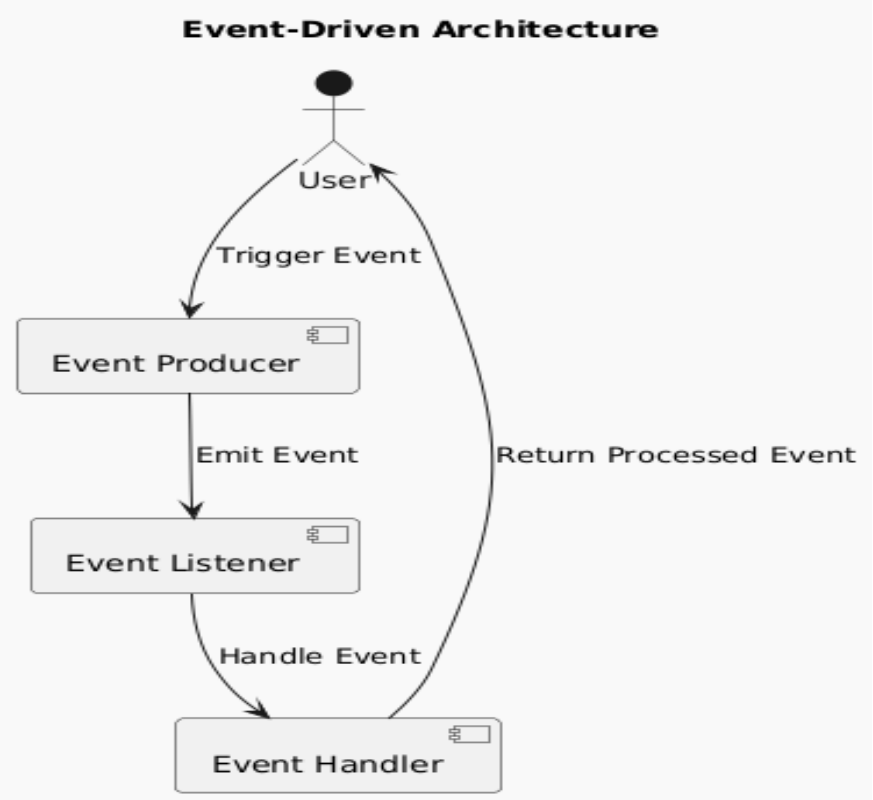

Figure 11 Event Driven Architecture

Chosen Approach: Modular Layered Architecture

The modular layered architecture was chosen for our Project(VocalXpert) due to its clear separation of concerns, scalability, and maintainability, adoptability,' which perfectly aligns with the project's requirements..

- **Separation of Concerns:** Divides the system into independent layers (Input, Processing, Output), each handling specific tasks, simplifying development and maintenance.
- **Scalability & Flexibility:** In this architecture it allows to the easy enhancements or additions of new features by modifying or adding modules without disruption or interference of the entire system.
- **Maintainability;** In modular layered architecture each layer is independent, facilitating localized bug fixes,' optimizations, cost effective and time saving, and updates, reducing system-wide disruptions.
- **Balanced Complexity:** While introducing some initial design complexity, the modular approach ensures long-term simplicity for making it easier to maintain,operate and extend.
- **Reusability:** Modules can be reused across different system areas or future versions, improving efficiency and reducing development time.
- **Improved Clarity;** Modular Layered Architecture also distincts layers and defined interactions, the system structure is easy to understand and navigate,, leading to reduced errors.

Class Diagram of VocalXpert

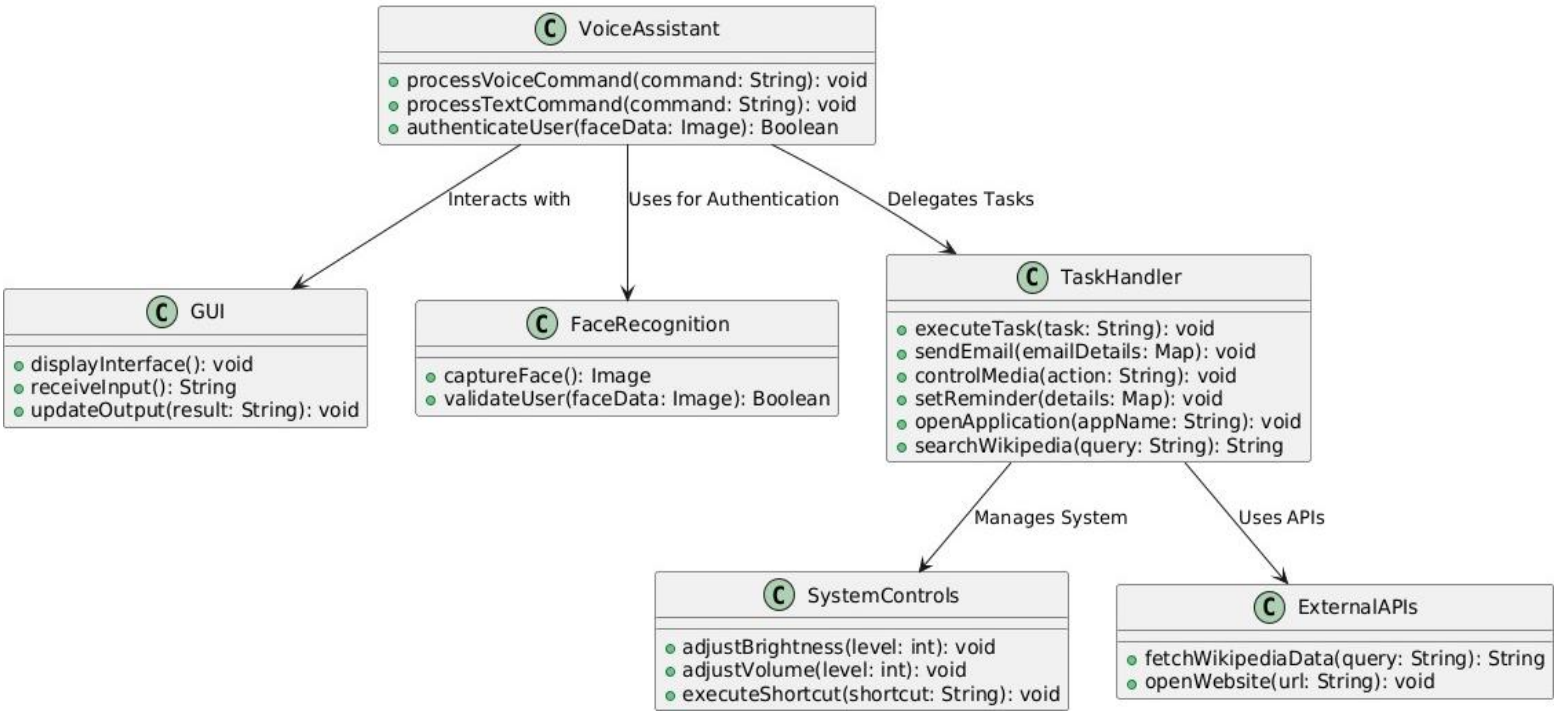

Figure 12 Class Diagram

System State Diagram of VocalXpert

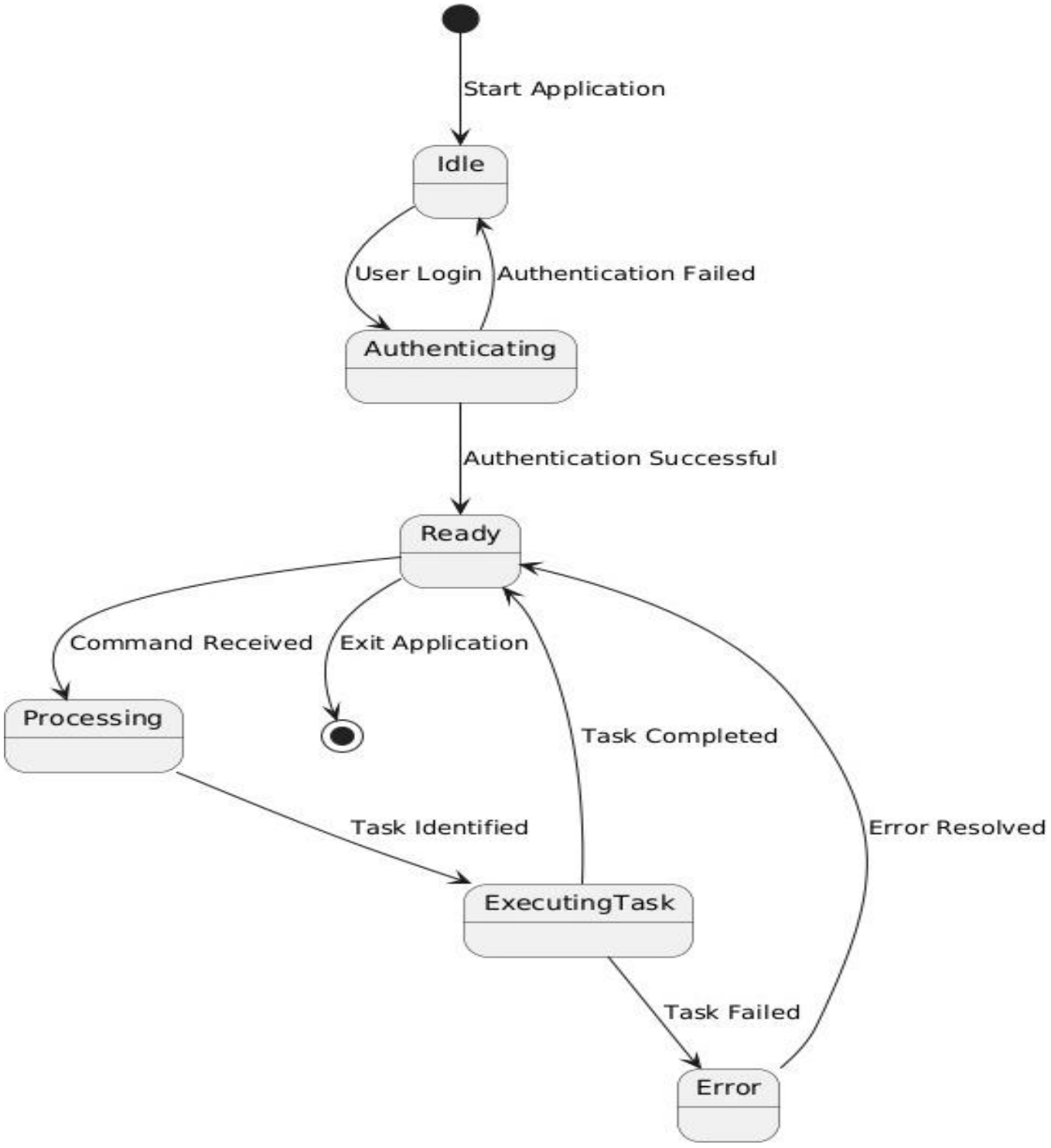

Figure 13 System State Diagram

System Sequence Diagram of VocalXpert

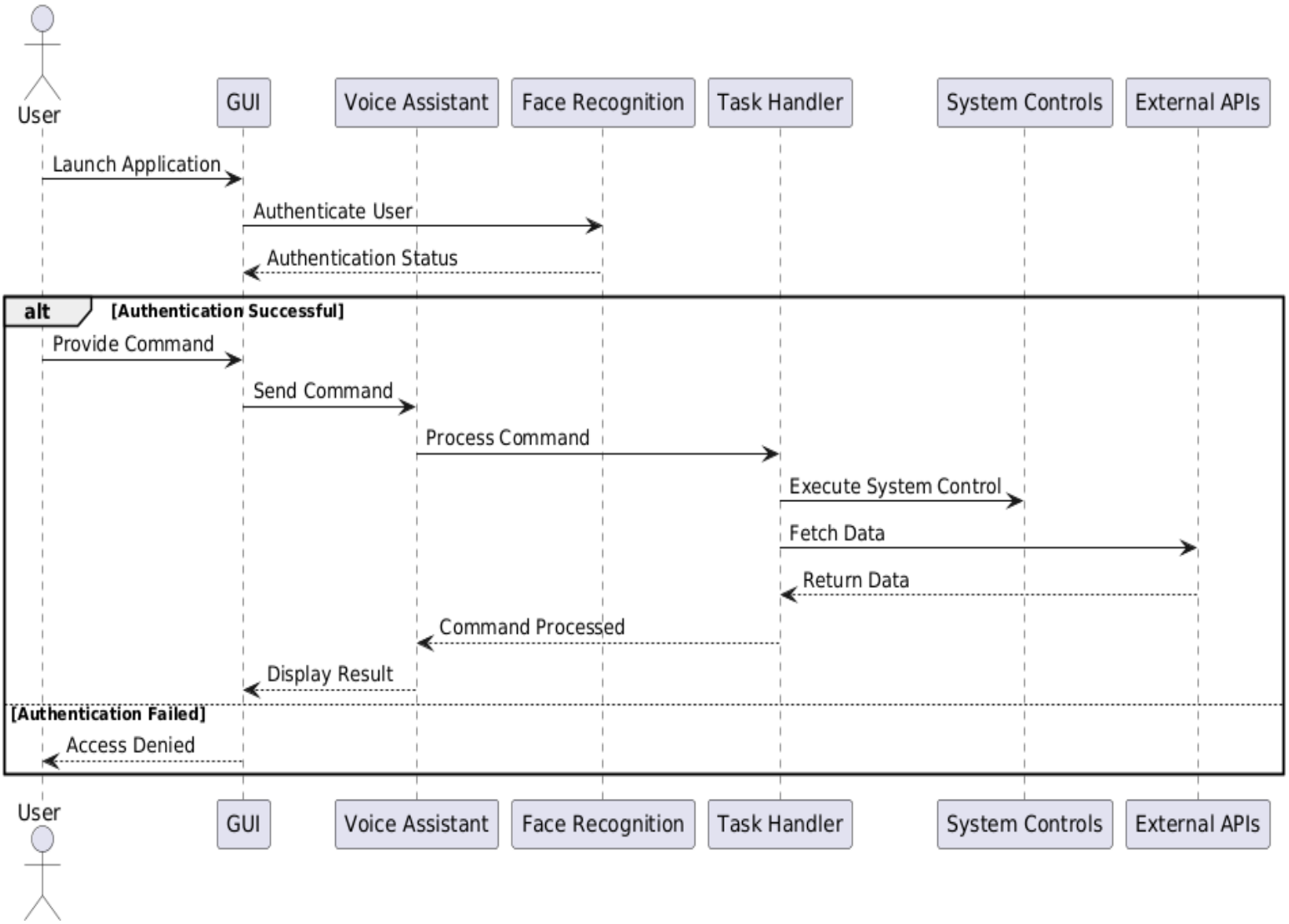

Figure 14 System Sequence Diagram

Component Diagram

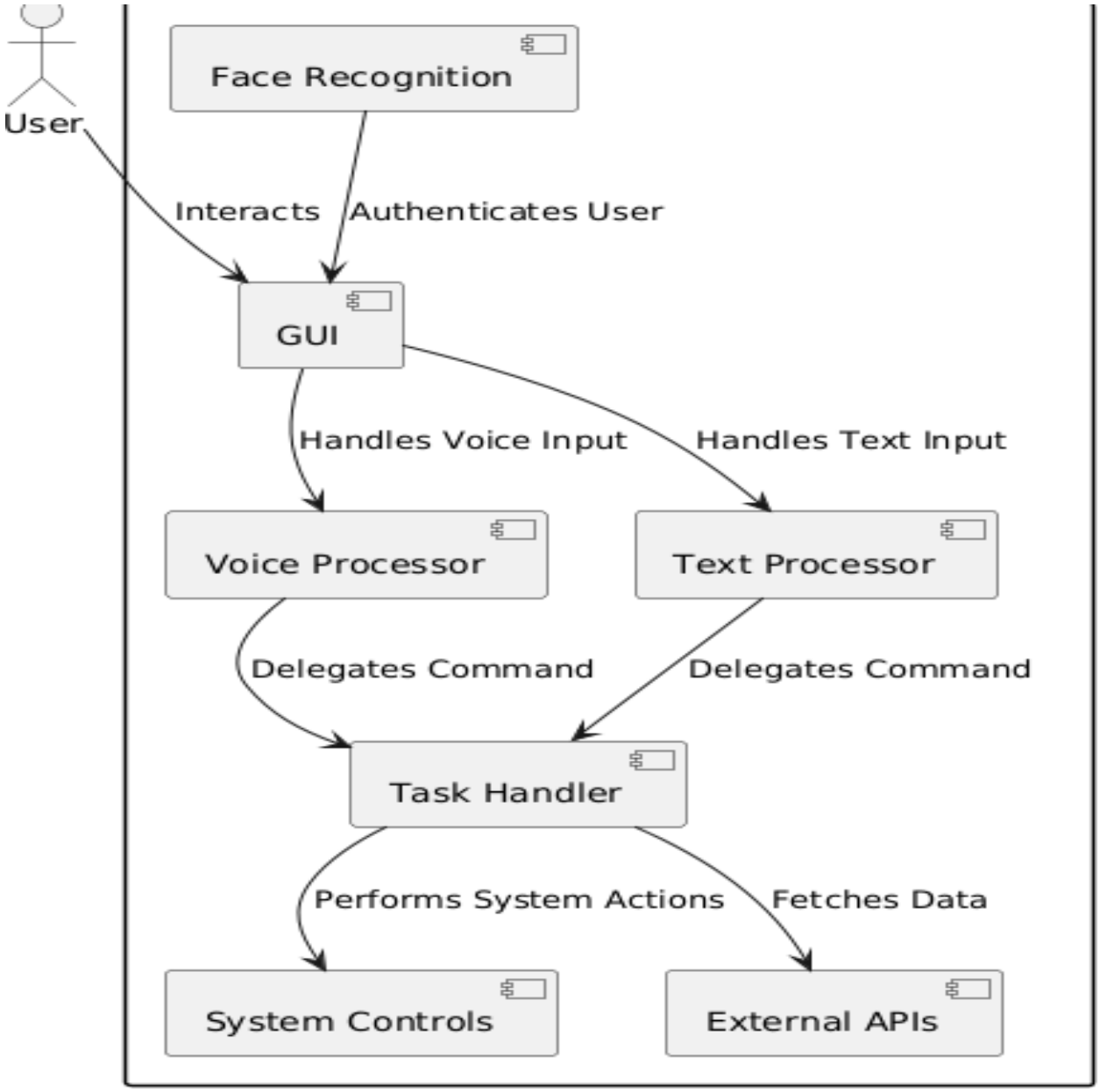

Figure 15 Component Diagram

User Interface Design

Description of the User Interface

The **VocalXpert** interface is designed to be user-friendly and versatile, offering both **voice** and **text** command input options to ensure ease of use. There is a brief overview of its key components below for main GUI components:

1. **Main Window:**
   - The central window or main screen of the UI, showing the assistant's name and status updates and main processes of the VocalXpert. It includes a toolbar for quick access to common actions and other elements.

2. **Voice Command Area:**
   - Allowing users to give commands through speech for execution of commands. The system listens for voice input and processes it in real-time, providing feedback.

3. **Text Command Area:**
   - Users can also type text commands in this area if they prefer not to use voice input. It’s straightforward and easy to use.

4. **Command Output Area:**
   - This command output area displays the results and main feedbacks of the commands, either through text or voice feedback. It shows the success messages or errors based on the system's response.

5. **Face Recognition:**
   - Adding a separate and independent layer of security by authenticating the user via facial recognition before they can use sensitive features in separate windows and threads and GUI elements.

6. **Settings:**
   - Customizes the assistant's settings, such as voice options and language preferences, ensuring a personalized experience.

7. **Recent Commands:**
   - Provides a history of previous commands for easy re-access, allowing users to quickly repeat actions.

8. **Notifications:**
   - Provides real-time feedback, alerting the user about the success or failure of commands and actions.

Prototype GUI of VocalXpert

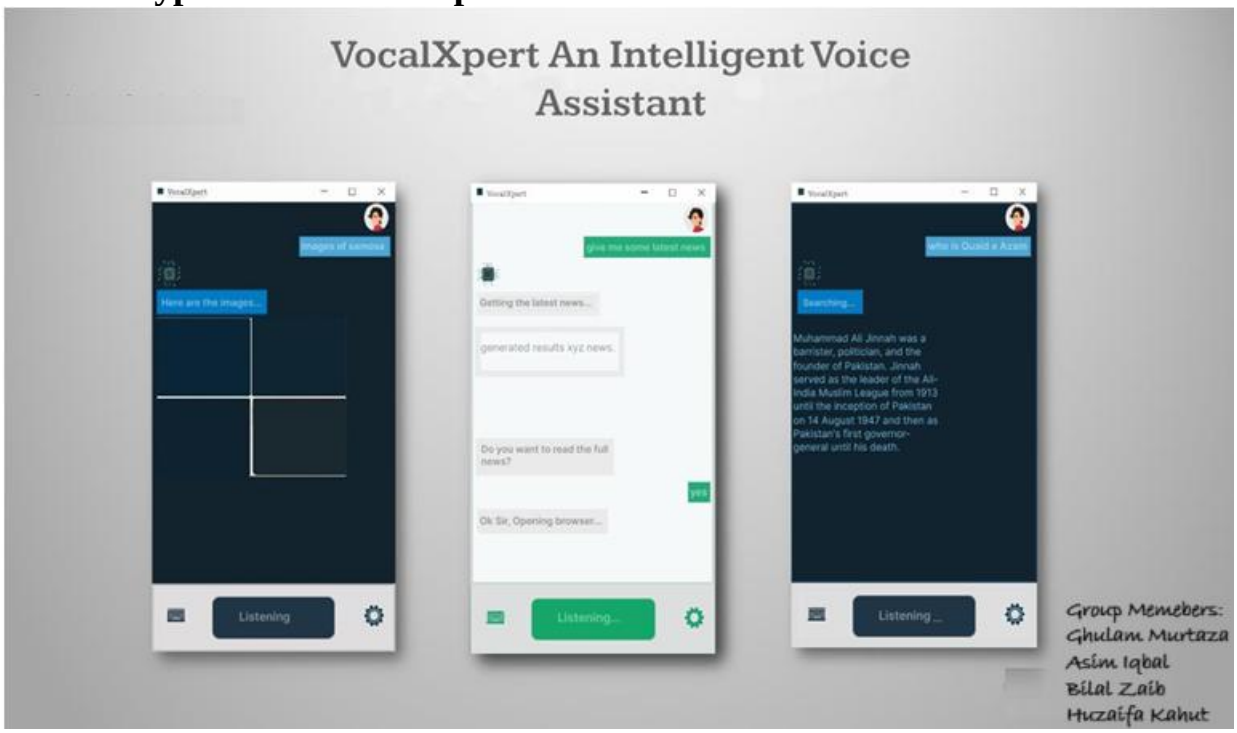

Figure 16 VocalXpert GUI Design Prototype

Objects and Actions

This section describes the VocalXpert's object interactions and actions in the VocalXpert system, accompanied by a UML Activity Diagram.

Activity Diagram (Flow Chart):

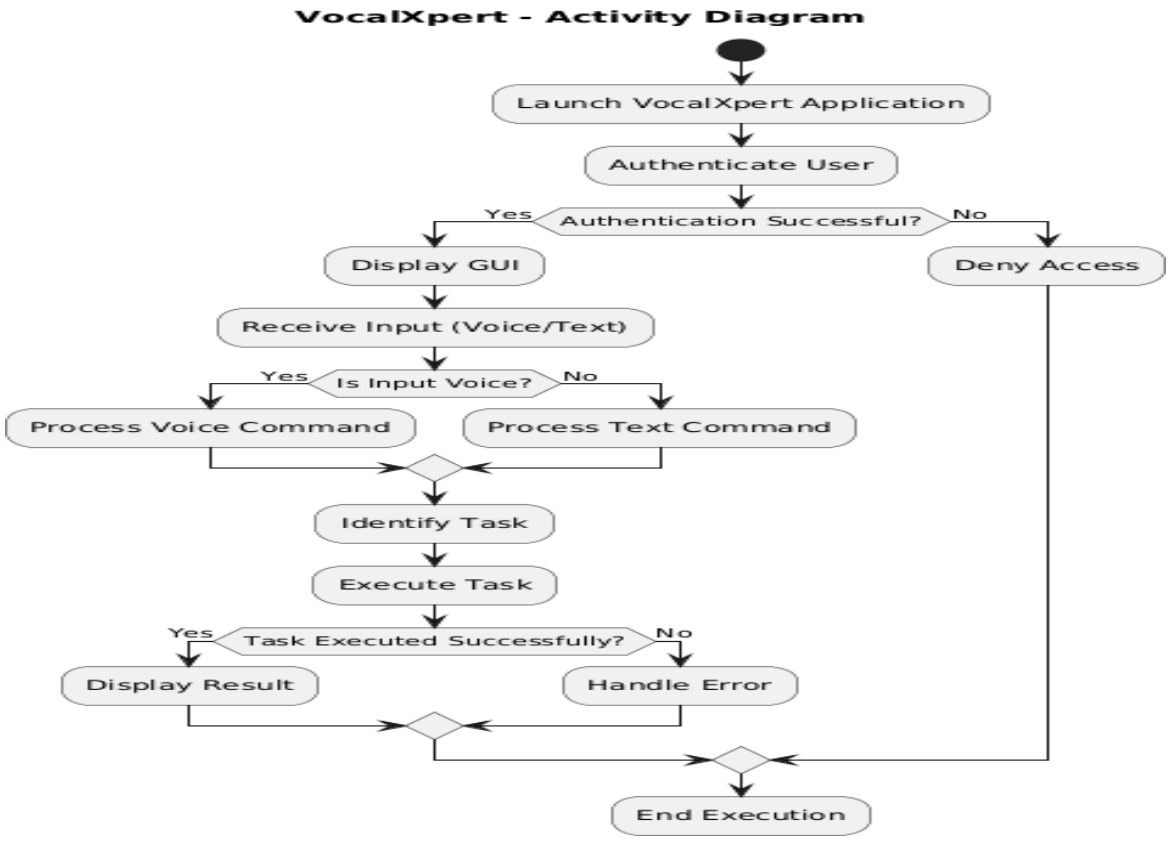

Figure 20 Activity Diagram

Additional Material

DFD Level 0:

Figure 21 DFD Level 0 (Context Diagram)

DFD Level 1:

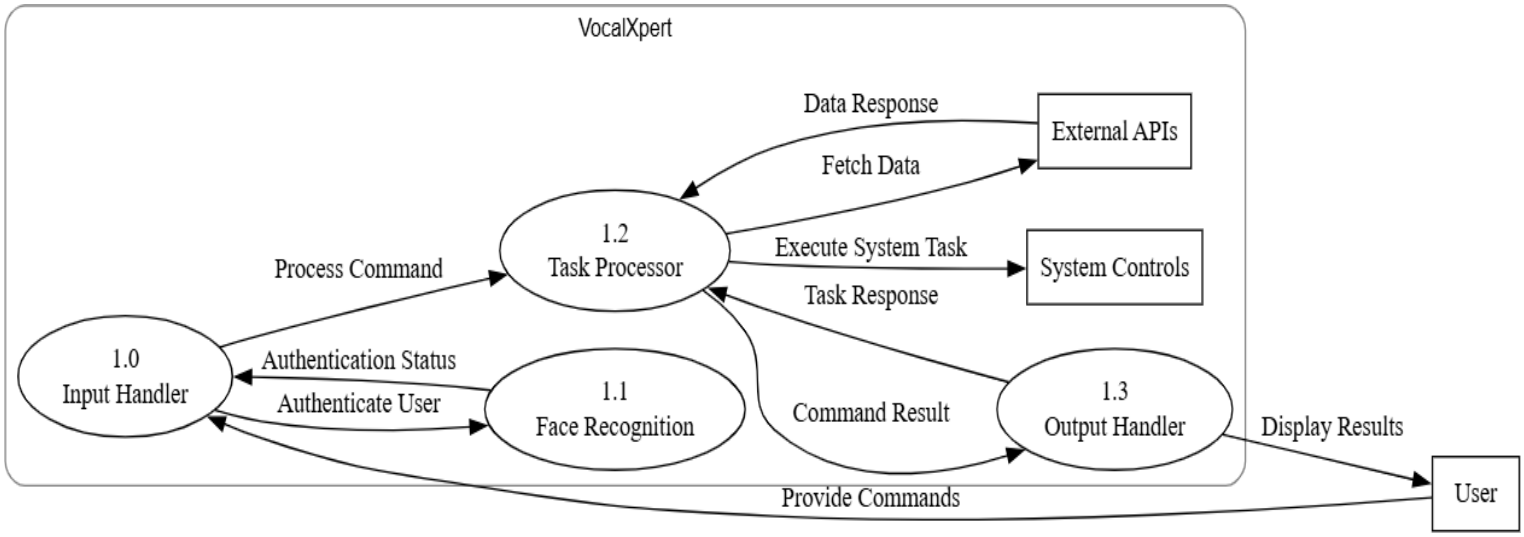

Figure 22 DFD Level 1 Updated

DFD Level 2:

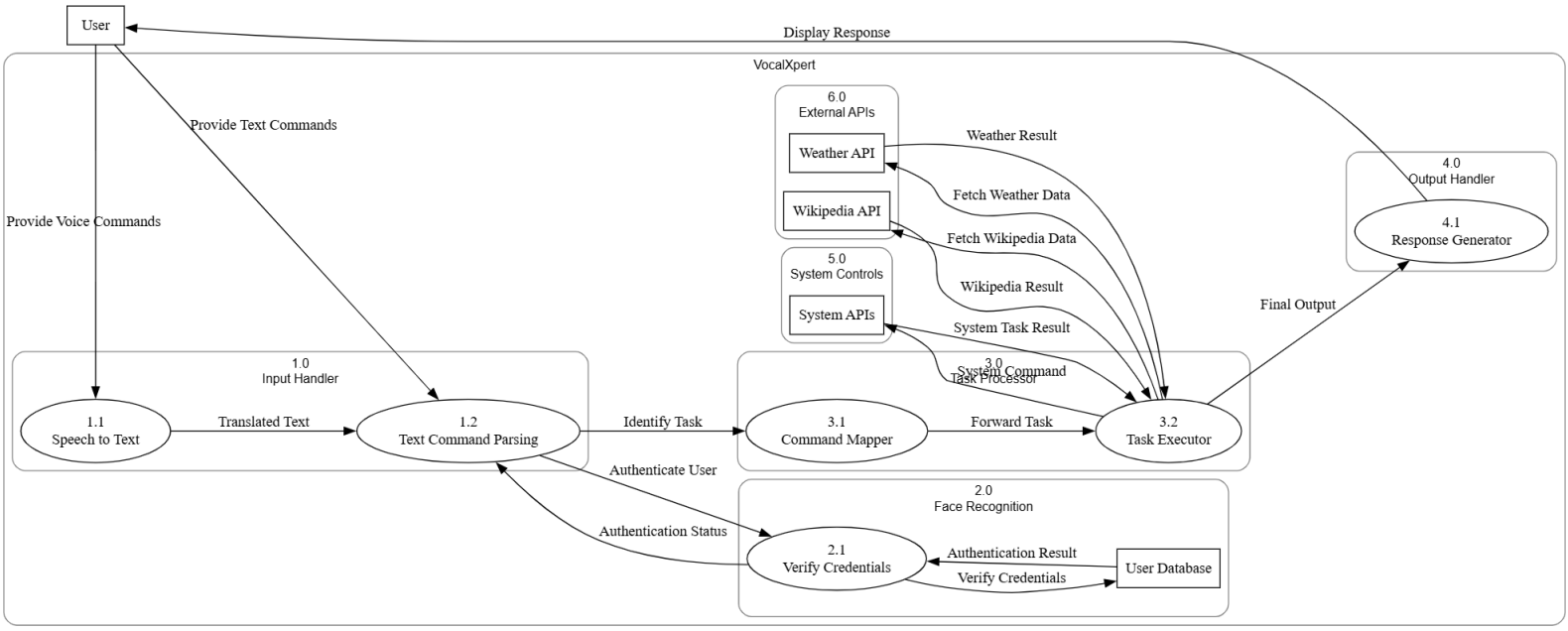

Figure 23 DFD Level 2 Updated

CHAPTER - 6

SOFTWARE TEST DOCUMENTATION (STD)

SOFTWARE TESTING DOCUMENTATION

Introduction

This document describes the testing strategy, and execution plan for VocalXpert project a desktop-based intelligent voice assistant developed in Python. This documentation is basically designed to test the system for its functional and non-functional requirements using structured testing process. These tests are basically termination criteria to test the software when the project states that it is working as expected and that there is no bug.

Test Plan

The testing plan defines the scope, approach, and resources required for testing the VocalXpert system. The main focus is on functional testing GUI validation, integration testing, and usability evaluation.

Features to be tested

- **Voice Command Recognition:** Ensuring accurate recognition of user speech.
- **GUI Functionality:** Testing Tkinter-based GUI responsiveness and event handling.
- **Face Recognition Login:** Validating the success/failure of facial authentication.
- **Task Execution:** Checking proper execution of tasks like playing music, sending emails, opening applications, etc.
- **News & Weather Retrieval:** Verifying the correct fetching and display of categorized news and weather data.
- **File Operations:** Confirming correct file selection, saving, and auto-typing.
- **Web & App Control:** Testing features like opening/closing websites and system applications.
- **Fallback/Chat Responses:** Ensuring assistant gives meaningful responses to casual conversation using normal chats json.

Testing tools and environments

Operating System

- Windows 10 (64-bit architecture) or Windows 11 (64-bit) is the main primary operating system for testing and developing the VocalXpert.

Development Stack

- VS Code for Developing and Testing the VocalXpert.
- Python 3.12.x is the primary Developing Language of VocalXpert.
- Tkinter for developing the GUI of VocalXpert.

Testing Tools

- Manual GUI Testing to test interface and interactions.
- Mock Testing for simulating SpeechRecognition inputs.
- OpenCV for testing facial recognition module.

Hardware Requirements

- Microphone for voice command input and Webcam for facial login authentication.
- Minimum 4GB RAM and Recommended 8GB RAM.
- Internet Access for the online features or modules.
- Dual-core processor or higher.

REFERENCES

[1.]Artificial Intelligence Basics: Russell & Norvig - *AI:* A Modern Approach

[2.]Incremental Software Development Model: Pressman - Software Engineering: A Practitioner's Approach

[3.]OpenCV for Face Recognition: https://docs.opencv.org/

[4.]PlantUML Documentation: https://plantuml.com/

[5.]PyInstaller for Packaging: https://pyinstaller.org/

[6.]Tkinter for GUI Development: https://wiki.python.org/moin/TkInter

[7.]Python Programming Language: https://www.python.org/

[8.]Real-Time Communication Protocols: https://www.w3.org/TR/websockets/

[9.]SMTP for Email: https://datatracker.ietf.org/doc/html/rfc5321

[10.] SpeechRecognition API: https://pypi.org/project/SpeechRecognition/

[11.] System Control via Python: Python OS Module Documentation

[12.] Voice Assistants and AI: *Srivastava* - Voice Assistant for Home Automation: A Review

[13.] Weather Data API: https://openweathermap.org/api

[14.] Wikipedia API: https://www.mediawiki.org/wiki/API

[15.] PyAutoGUI for System Automation: https://pyautogui.readthedocs.io/en/latest/

[16.] Windows Control Automation: https://docs.microsoft.com/en-us/windows/win32/api/

[17.] Amazon Alexa Voice Service: https://developer.amazon.com/en-US/alexa

[18.] Speech Synthesis and Voice Generation: http://www.espeak.sourceforge.net/

[19.] Real-Time Voice Interaction Systems: Lowe, D. G., & Wu, Z. - *Conversational Agents* and Virtual Assistants

[20.] Face Recognition Libraries in Python: https://github.com/ageitgey/face-api.js

[21.] Text-to-Speech Synthesis: https://pypi.org/project/gTTS/

[22.] Component-Based Software Engineering: Szyperski, C. - *Component Software: Beyond Object-Oriented Programming*

[23.] Event-Driven Architecture Overview: *Architecting Event-Driven Systems* by Ben Stopford

[24.] Client-Server Architecture: Designing Client-Server Systems by Michael J. Kachakbar