VIRTUAL ASSISTANT

Sheraz Ibrahim

Saeed Muhammad

Uzair Iqbal

Supervised By

Dr. Sajjad Haider

Submitted for the partial fulfillment of BS Computer Science degree to the Faculty of Engineering & Computer Science

DEPARTMENT OF COMPUTER SCIENCE

NATIONAL UNIVERSITY OF MODERN LANGUAGES ISLAMABAD

January, 2022

ABSTRACT

The proposed system is a Virtual Assistant. The virtual assistants now a days provide user the ability to give textual/voice commands and perform actions according to those commands. These existing systems do not allow the user to automate the tasks, nor do they allow the user to add custom commands/actions. On the other hand, the proposed system does provide user the ability to automate the tasks as well as the ability to add custom commands /actions for them.

The assistant takes voice as input, and it then performs the desired task as output/response (response can be user defined) to that command (voice input). At present people do not want to perform a task with same steps repeatedly. The proposed system helps them automate those tasks so they would not have to repeat them. The unique functionality the proposed system has is that it gives user the ability to add custom commands/actions. The proposed system has different modules. The main module of this proposed system is the voice assistant itself with some already defined (default actions) commands. The one of other modules of the proposed system provides the functionality of automating his/her tasks by recording his/her performed actions in a sequence. Those recorded tasks can then be used as the outputs for the customized (user defined) commands. The proposed system achieves this task automation by recording the actions such as Mouse Movements, Keyboard Inputs, events like mouse right/left click etc. It records all the actions performed in a sequence and then it can repeat those actions the same way user performed them. These actions can be performed for the required/desired amount of time. The user can add as many new customized commands as he/she wants. The proposed system is developed using Python. Visual Studio Community is the IDE which is used as the tool for the development of the proposed system. This system is meant for windows operating system.

The developed system was successfully evaluated on different operating systems like windows 8.1 and windows 10 and different systems such as HP and Dell.

CERTIFICATE

Dated: _____

Final Approval

It is certified that project report titled 'Virtual Assistant' **submitted by** Sheraz Ibrahim, Saeed Muhammad**, and** Uzair Iqbal **for the partial fulfillment of the requirement of** "Bachelors Degree in Computer Science" **is approved.**

COMMITEE

Dr. Basit Shehzad

Dean Engineering & CS: Signature: _____

Dr. Sajjad Haider

HoD Computer Science: Signature: _____

Ms. Mehvish Sabih

FYP Committee Head: Signature: _____

Mr. Zain ul Abideen

FYP Coordinator: Signature: _____

Dr. Sajjad Haider

Supervisor: Signature: _____

DECLARATION

We hereby declare that our dissertation is entirely our work and genuine / original. We understand that in case of discovery of any PLAGIARISM at any stage, our group will be assigned an F (FAIL) grade and it may result in withdrawal of our Bachelor's degree.

Group members:

Name	**Signature**
Sheraz Ibrahim	_____
Saeed Muhammad	_____
Uzair Iqbal	_____

PLAGIARISM CERTIFICATE

This is to certify that the project entitled "**Virtual Assistant**", which is being submitted here with for the award of the "**Degree of Bachelors" in "Computer Science**". This is the result of the original work by **Sheraz Ibrahim, Saeed Muhammad** and **Uzair Iqbal** under my supervision and guidance. The work embodied in this project has not been done earlier for the basis of award of any degree or compatible certificate or similar title of this for any other diploma/examining body or university to the best of my knowledge and belief.

Turnitin Originality Report

Processed on 16-Dec-2021 01:23 PKT

ID: 1731876157

Word Count: 8981

Similarity Index

05%

Similarity by Source

Internet Sources:	04%
Publications:	02%
Student Papers:	02%

_____

Date: 16/12/2021 Dr. Sajjad Haider (Supervisor)

TURNITIN ORIGINALITY REPORT

Virtual Assistant [BSCS] by **Sheraz Ibrahim, Saeed Muhammad,** and **Uzair Iqbal**
From **Dr. Sajjad Haider**

Processed on 16-Dec-2021 01:23 PKT
ID: 1731876157
Word Count: 8981

Similarity Index:
05%
Similarity by Source

Internet Sources:	04%
Publications:	02%
Student Papers:	02%

SOURCES:

1. 1% match (student papers) Submitted to Higher Education Commission Pakistan
2. 1% match (student papers) Submitted to University of Greenwich
3. 1% match (publication) Costin Bădică, Maria Ganzha, Marcin Paprzycki. "Chapter 110 Rule-Based Automated Price Negotiation: Overview and Experiment", Springer Science and Business Media LLC, 2006
4. 1% match (publication) "Proceeding of First Doctoral Symposium on Natural Computing Research", Springer Science and Business Media LLC, 2021
5. 1% match (publication) Masaki Hasegawa, S. Bhalla, T. Izumita. "A User Oriented Query Interface for Web-Based Geographic Information Systems", 2007 Japan-China Joint Workshop on Frontier of Computer Science and Technology (FCST 2007), 2007
6. 1% match (publication) Reduced Order Methods for Modeling and Computational Reduction, 2014.
7. 1% match (internet source)www.coursehero.com
8. 1% match (internet source) www.invensis.net

TABLE OF CONTENTS

Chapter **Page**

CHAPTER 1: INTRODUCTION....1

1.1. Project Domain....2

1.2. Problem Identification....3

1.2.1. Proposed Solution....3

1.2.2. Objectives....3

1.2.3. Scope of the Project....4

1.3. Effectiveness / Usefulness of the System....4

1.4. Resource Requirements....4

1.4.1. Hardware Requirements....5

1.4.2. Software Requirements....5

1.5. Report Organization....6

CHAPTER 2: BACKGROUND AND EXISTING SYSTEM....7

2.1. Related Literature Review....8

2.1.1. Personal Assistant to Facilitate User Task Automation....8

2.1.2. SIRI Apple's Assistant....8

2.1.3. An Introduction to Voice Assistants....9

2.2. Related Systems/Applications....9

2.2.1. SIRI....10

2.2.2. Cortana....10

2.3. Identified Problem from Existing work....11

2.4. Selected Boundary for Proposed Solution....12

CHAPTER 3: SYSTEM REQUIREMENTS AND SPECIFICATIONS....13

3.1. System Specification....14

3.2. System Modules....14

3.2.1. Voice Recognition Module....14

3.2.2. Command Verification Module....15

3.2.3. Command Execution Module....15

3.2.4. Task Automation Module..... 15
3.3. Functional Requirements..... 15
3.3.1. Take Voice Input 16
3.3.2. Speech To Text Conversion..... 16
3.3.3. Verify Valid Input/Command..... 16
3.3.4. Command Execution..... 16
3.3.5. Automate User Tasks..... 16
3.3.6. Add Custom Commands..... 16
3.3.7. Replay Automated Tasks..... 16
3.4. Non-Functional Requirements..... 17
3.4.1. Performance..... 17
3.4.2. Efficient In Use..... 17
3.4.3. Modularity..... 17
3.4.4. Quality..... 18
3.4.5. Easy To Use..... 18
3.4.6. Functional..... 18
CHAPTER 4: SYSTEM MODELLING AND DESIGN..... 19
4.1. System Design and Analysis..... 20
4.2. Use Case Diagram..... 20
4.3. Activity Diagram..... 21
4.4. System Sequence Diagram..... 22
CHAPTER 5: SYSTEM TESTING AND VALIDATION..... 24
5.1. System Testing..... 25
5.2. Testing Techniques..... 26
5.2.1. Unit Testing..... 27
5.2.2. Integration Testing..... 28
5.2.3. System Testing..... 29
5.2.4. Functional Testing..... 29
5.3. Test Cases..... 29
5.3.1. Test Case1: Execute Command..... 30

5.3.2. Test Case2: Add Custom Command.....31
5.4. Non-Functional Requirements.....33
5.4.1. Performance.....33
5.4.2. Efficient In Use.....33
5.4.3. Modularity.....33
5.4.4. Quality.....34
5.4.5. Easy To Use.....34
5.4.6. Functional.....34
CHAPTER 6: CONCLUSION.....35
6.1. Conclusion.....36
6.2. Limitation and Future Work.....37
APPENDIX - 1.....38
APPENDIX - II.....40
REFERENCES.....42

LIST OF FIGURES

Figure **Caption** **Page No.**

4.1: Use-Case Diagram.....21
4.2: Activity Diagram.....22
4.3: System Sequence Diagram.....23

LIST OF TABLES

Table **Caption** **Page No.**

1.1: Hardware Requirements (Minimum)..... 5
1.2: Hardware Requirements (Recommended)..... 5
1.3: Software Requirements..... 6
2.1: Summary of Reviewed Literature..... 9
2.2: Summary of Existing Systems..... 11
5.1: Test Case for Command Execution.....30
5.2: Test Case for Add Custom Command.....31

Chapter 1

INTRODUCTION

At present, more than half of our routine work are done by machines. In this age man relies so much on machines that in the near future almost everything will become automatic. The same is true of computer-related tasks. Tasks performed routinely on a PC can be automated, and software that is used to assist in performing these automated tasks following a user's command is called Intelligent Virtual / Personal Assistant. So many voice or virtual assistants are out there like Cortana, Siri, Alexa, and Google Assistant. These are the existing systems that help the user to manage basic tasks like email, to-do lists, and calendars and other tasks with verbal commands. Another word that comes to mind when discussing the virtual assistant is custom command or user-defined command. The custom / user-defined command is a command that is defined by the user and the action to be taken while executing this command is also described by the user. Existing systems mentioned are limited in scope when it comes to customized commands. No existing voice assistant provides that feature which enables the user to add custom commands. The question is why would anybody need such a feature. The answer to this question is that not all tasks can be automated nor can they be provided in one package / software. That's why one has to develop a system that has the basic capabilities of a voice assistant as well as a feature that will enable the users to automate their tasks.

The proposed system is a virtual assistant which has the basic capabilities of the voice assistant and it provides user the functionality to automate his/her tasks. It achieves this by recording the user’s actions and replaying them for the required/desired amount of time. The proposed system records user’s actions such as Mouse Movements, Mouse Click Events (Like Mouse Left Click, Mouse Right Click, Mouse Middle Click and etc.) and the keyboard keystrokes etc. The user is provided the functionality to replay those actions as many time as user wants. The user has the option to add a new custom command for the automated task so that user can perform that task whenever he/she wants with the user-defined verbal commands

1.1. Project Domain

Intelligent Virtual/Personal Assistant is a software that uses artificial intelligence and provides the basic functionality and capabilities of an intelligent voice assistant but the unique feature that it offers is that it allows user to automate his/her tasks and for those automated tasks user can add custom commands. User can call those verbal custom

commands to execute those user-defined automated tasks. The proposed system provides a generic way to automate user's tasks instead of trying to develop a system which has all the tasks automated which is impossible to develop.

1.2. Problem Identification

So many voice or virtual assistants are out there like Cortana, Siri, Alexa, and Google Assistant. These are the existing systems that help the user to manage basic tasks like email, to-do lists, and calendars and other tasks with verbal commands. Another word that comes to mind when discussing the virtual assistant is custom command or user-defined command. The custom / user-defined command is a command that is defined by the user and the action to be taken while executing this command is also described by the user. Existing systems mentioned are limited in scope when it comes to customized commands. No existing voice assistant provides that feature which enables the user to add custom commands. The question is why would anybody need such a feature. The answer to this question is that not all tasks can be automated nor can they be provided in one package / software. That's why one has to develop a system that has the basic capabilities of a voice assistant as well as a feature that will enable the user to automate their tasks.

1.2.1. Proposed Solution

The proposed system is a virtual assistant which has the basic capability of the voice assistant and it provides users the functionality to automate their tasks. It achieves this by recording the user's actions and replaying them for the required/desired amount of time. The proposed system records user's actions such as Mouse Movements, Mouse Click Events (Like Mouse Left Click, Mouse Right Click, Mouse Middle Click and etc.) and the keyboard keystrokes etc. The user is provided the functionality to replay those actions as many time as user wants. The user has the option to add a new custom command for the automated task so that users can perform that task whenever they want with the user-defined verbal commands.

1.2.2. Objectives

The objectives of developing the proposed system are listed as below:

1. To design a system which provides users a generic way to automate their tasks.

2. To develop a system that allows users to add custom verbal commands for their automated tasks.
3. To improve already existing systems (Existing Voice Assistants) as they have the limitation that neither they allow their users to automate their tasks nor do they allow to define/specify custom commands for those automated tasks.
4. To increase usefulness of existing systems by expanding their scope.

1.2.3. Scope of the Project

The scope of this project can be very clearly stated as the problem has been identified and the solution to the problem has already been stated. The proposed system aims to create a single virtual assistant with the basic capabilities of a voice assistant, as well as giving users the ability to automate their tasks. The proposed system is a desktop application and it is meant for windows operating system. Main and unique functionality of the proposed system is that it provides a generic way to automate users' tasks and then define custom commands for those automated tasks.

1.3. Effectiveness / Usefulness of the System

Now a days, more than half of our routine work is done by machines. In this age man relies so much on machines that in the near future almost everything will become automatic. The same is true of computer-related tasks. Existing voice assistants help the user manage basic tasks via verbal commands, but Existing systems are limited in scope and use when it comes to customized commands. No existing voice assistant provides that feature which enables the user to add custom commands. Whereas, the proposed system aims to create a single virtual assistant with the basic capabilities of a voice assistant, as well as giving users the ability to automate their tasks. This unique feature will allow users to automate a lot of their tasks and then specify custom commands for them. Furthermore, there is no limit to how many tasks can be automated. There is no limit to the number of tasks as well as the type of tasks that can be automated. This makes the proposed system very efficient and useful.

1.4. Resource Requirements

One software may work on one operating system and may not work on another. Similarly, compatible hardware may be required to run the software efficiently. Such needs will be discussed in this section.

1.4.1. Hardware Requirements

Minimum hardware resources that are required to run the proposed system are a laptop or desktop with minimum Core i3 2nd Generation, 4 GB RAM, 60 GB Hard Disk and an internet connection of 3Mbps. Whereas the recommended requirements are a laptop or desktop with Core i5 3rd Generation, 6 GB RAM, 60 GB Hard Disk. A microphone is also required for voice commands. Following tables Table1.1 and Table1.2 describe these hardware requirements as follows:

Table 1.1: Hardware Requirements (Minimum)

Hardware	**Specification**
Processor	Core i3 2nd Generation
RAM	4 GB
Hard Disk	60 GB

Table 1.2: Hardware Requirements (Recommended)

Hardware	**Specification**
Processor	Core i5 3rd Generation
RAM	6 GB
Hard Disk	60 GB

1.4.2. Software Requirements

The development and implementation of the proposed system requires a system with at least Windows 7 operating system. While Visual Studio Code is used as an IDE tool and Python is used as a programming language to develop this system. Table 1.3 below lists software requirements for the development and execution of the proposed system:

Table 1.3: Software Requirements

Software	Specification
Operating System	Windows 7, 8, 8.1, 10
IDE	Visual Studio Community
Language	Python

1.5. Report Organization

The first chapter of this report contains an introductory section. This chapter discusses in detail the problem identification, solution and various requirements of the proposed system.

The second chapter of the report contains information about the related systems and their applications along with their limitations. Literature review and boundary of the proposed system are also discussed in this chapter.

Chapter 3 covers the various modules of the proposed system. It also talks about functional and non-functional requirements of the proposed system.

The fourth chapter of this report is about system modelling and various diagrams. These diagrams include use-case diagram, system sequence diagram, and activity diagram of the proposed system.

Chapter 5 is about testing and validation of the proposed system. It discusses various testing techniques and test cases for the testing of the proposed system.

The sixth chapter of this report is the conclusion. It talks about the limitations of the proposed system and the future work that should be done to enhance the proposed system.

Chapter 2

BACKGROUND AND EXISTING SYSTEM

The proposed system is a virtual assistant which has the basic capability of the voice assistant and it provides users the functionality to automate their tasks. It achieves this by recording the user's actions and replaying them for the required/desired amount of time. The user has the option to add a new custom command for the automated task so that users can perform that task whenever they want with the user-defined verbal commands.

This chapter of the report contains information about the related systems and their applications along with their limitations. Literature review and selected boundary of the proposed system are also discussed in this chapter.

2.1. Related Literature Review

2.1.1. Personal Assistant to Facilitate User Task Automation

In this paper the authors discuss all the existing systems. Salient features of all the existing systems are discussed briefly in this paper. After discussing all the existing systems authors proposed a new system (Personal Assistant) which will provide features of all the existing systems in one place. The new system will also allow the user to add new commands and this assistant will have an additional feature of remote access. However, this article does not discuss how the user will be able to automate new tasks using the proposed system. This is the limitation of this paper.[1]

2.1.2. SIRI Apple's Assistant

This paper discusses the Apple's personal Assistant SIRI in detail. SIRI is a built-in intelligent personal assistant for Apple products. It takes voice commands from the user and it performs corresponding task against the user command. Working of SIRI can be simply stated as it takes voice commands from the user then it performs voice to text conversion (Voice Recognition) after that it sends everything to apple servers on the cloud to use maximum resources then it tries to understand what was implied and performs desired action based on what was commanded. SIRI can perform several tasks against voice commands. SIRI is incapable of understanding different English accents.[4]

2.1.3. An Introduction to Voice Assistants

Different voice assistants are discussed in this article. Voice assistants are the software agents that are becoming so popular now a days due to their efficient human computer interaction. These agents can understand human voice and respond accordingly. Alexa, Siri, Cortana and Google's Assistant are the famous voice assistants available in different devices and operating systems. User can ask questions and perform different tasks by giving voice input. They can also perform basic activities like email sending, to-do list, internet searching, map navigation and much more through verbal commands. Voice assistant listens to the voice input, it extracts text from voice command and sends the command to the dedicated server. The server interprets the command and responds back to the user accordingly. Every assistant performs some basic tasks such as they can make phone calls, read and send messages, and internet searching etc. The limitation of this paper is that it does not talk about addition of custom commands and task automation in generic way.[2]

Table 2.1: Summary of Reviewed Literature

Year	Authors	Limitation
2014	Rasika Anerao, Utkarsh Mehta2, Sharangdhar Vaze, G. Hrishikesh	This article does not discuss task automation.
2017	Bisma Shakeel, Tabasum, Mir Shahnawaz Ahmad	SIRI does not understand different English accents
2018	Matthew B. Hoy	This article does not discuss custom command addition.

2.2. Related Systems/Applications

So many voice or virtual assistants are out there like Cortana, Siri, Alexa, and Google Assistant. These are the existing systems that help the user to manage basic tasks. Existing systems mentioned are limited in scope when it comes to customized commands.

2.2.1. SIRI

SIRI is a built-in intelligent personal assistant for Apple products. It takes voice commands from the user and it performs corresponding task against the user command. In other words SIRI uses voice input and a natural language user interface for user interaction, it delegates requests to a set of internet services to perform actions. Working of SIRI can be simply stated as it takes voice commands from the user then it performs voice to text conversion (Voice Recognition) after that it sends everything to apple servers on the cloud to use maximum resources then it tries to understand what was implied and performs desired action based on what was commanded. The user has a lot of voice commands to interact with SIRI. SIRI can perform several tasks against voice commands for example Phone Actions (Call someone, Read Messages and etc.), Schedule Events, Set Reminders, Search The Internet and Navigation etc. SIRI lacks information on nearby places. It also is incapable of understanding different English accents. In comparison to proposed system it is unable to provide user a generic way to automate user tasks. [4]

2.2.2. Cortana

Cortana is Microsoft's HCI based digital assistant built into Windows 10 and Windows Phones. Cortana is designed to help the user perform different tasks. User uses voice commands to communicate with the Cortana. It can perform a variety of tasks such as searching the Internet, locating local files, setting reminders and alarms, sending emails, predicting the weather and many more. It uses Microsoft's API, a cloud-based automatic translation service for decoding and interpreting voice commands. Speech recognition involves signal processing to convert an analog signal to a digital signal, then it understands the command and performs the desired task against the user command. Some notable features of Cortana are Cortana’s Notebook (It learns your preferences and habits based on how you interact with her), Composes Email, Sets Reminders, and Improved Internet Searches. In comparison to proposed system the Cortana is limited in scope as it does not provide user the ability to automate tasks in a generic manner.[3]

Table 2.2: Summary of Existing Systems

Year	Developer	System	Technology	Limitations
February 2010	Apple	SIRI	ASR, NLP	Inability to allow the user automate tasks in a generic manner
April 2014	Microsoft	Cortana	Microsoft's API (Automated Translation Service),NLP	Inability to allow the user automate tasks in a generic manner

2.3. Identified Problem from Existing work

So many voice or virtual assistants are out there like Cortana, Siri, Alexa, and Google Assistant. These are the existing systems that help the user to manage basic tasks like email, to-do lists, and calendars and other tasks with verbal commands. Another word that comes to mind when discussing the virtual assistant is custom command or user-defined command. The custom or user-defined command is a command that is defined by the user and the action to be taken while executing this command is also described by the user. Existing systems mentioned are limited in scope when it comes to customized commands. No existing voice assistant provides that feature which enables the user to add custom commands. The question is why would anybody need such a feature. The answer to this question is that not all tasks can be automated nor can they be provided in one package / software. That's why one has to develop a system that has the basic capabilities of a voice assistant as well as a feature that will enable the user to automate their tasks. The proposed system solves the identified problem as it is a virtual assistant which has the basic capability of the voice assistant and it provides users the functionality to automate their tasks. It achieves this by recording the user's actions and replaying them for the required/desired amount of time. The user has the option to add a new custom command for the automated task.

2.4. Selected Boundary for Proposed Solution

The scope of this project can be very clearly stated as the problem has been identified and the solution to the problem has already been stated. The proposed system aims to create a single virtual assistant with the basic capabilities of a voice assistant, as well as giving users the ability to automate their tasks. The proposed system is a desktop application and it is meant for windows operating system. Main and unique functionality of the proposed system is that it provides a generic way to automate users' tasks and then define custom commands for those automated tasks.

Chapter 3

SYSTEM REQUIREMENTS AND SPECIFICATIONS

This chapter covers system specifications. System specifications include functional requirements and non-functional requirements. Functional requirements are the requirements that are required at the time of implementation. These requirements define the characteristics of the product and these requirements also state what the software should do. Whereas, non-functional requirements are quality attributes. Any requirement can be called a quality attribute or a non-functional requirement that describes how the system should perform a particular task or function. So basically system requirements and specifications provide details of the work to done for the development of the proposed system. In this chapter various modules of the proposed system are also discussed.

3.1. System Specification

This section will discuss system specifications. System specifications are the functional and non-functional requirements. System specifications includes requirements that are needed at time of development that define what the system should do whereas other requirements called as quality attributes are also a part of system specifications. These functional and non-functional requirement or in other words system specifications will be discussed later in this chapter.

3.2. System Modules

Proposed system is divided into different modules. The question arises “What is a module?”. This question must be answered before diving into the details of the various modules of the proposed system. A computer program can be broken down to sub-programs called modules and this approach is called modular programming. This approach has many advantages such as Ease of Use, Reusability, and Ease of Maintenance. The proposed system is mainly divided into four modules. Voice Recognition Module, Command Verification Module, Command Execution Module, and Task Automation Module are the four main modules of the proposed system. The functionality of each module can be easily assessed by looking at the name of each module. These modules are discussed briefly.

3.2.1. Voice Recognition Module

This module has two different functionalities “Input Voice” and “Recognize The Command”.

Input Voice: The user will give the input in the form of audio (Voice Commands)

Recognize The Command: The voice input will then be converted to text.

3.2.2. Command Verification Module

This module will verify whether the converted text is a valid command or not. There are two types of commands that are used "User-Defined Commands" and "Default/Built-In Commands".

User-Defined Commands: Commands that are specified by the user while automating the tasks.

Default/Built-In Commands: Commands that are defined by the developers while developing the basic voice assistant for the proposed system.

3.2.3. Command Execution Module

This module will perform action by executing the verified command. There are two types of actions or task that the proposed system can perform by executing the verified command.

User-Automated Task/Action: Task which is automated by the user using the Task Automation Module of the proposed system.

Developer Automated Task/Action: Task which is automated by the developer while developing the basic voice assistant for the proposed system.

3.2.4. Task Automation Module

This module provides a generic way to automate the tasks (User-defined Task).

Record Actions: The user will be able to record his/her actions while performing a task. Actions can be anything (Like Mouse Movement, Mouse Click Events, Keyboard Keystrokes and etc.)

Replay: This module will be able to replay the recorded actions in the same sequence as performed by the user.

Automate: The user will be able to add custom commands for the actions he/she performed.

3.3. Functional Requirements

Any requirement or feature which determines what the system should do is called a functional requirement. Before developing a software product developers need to know

the functional requirements of that system. A functional requirement will define a particular way of operating for system when certain conditions are met. Take Voice Input, Speech To Text Conversion, Verify Valid Input/Command, Command Execution, Automate User Tasks, Add Custom Commands, Replay Automated Tasks are functional requirements that the proposed system should meet.

3.3.1. Take Voice Input

The proposed system should be able to take voice input from user using microphone.

3.3.2. Speech To Text Conversion

The proposed system should be able to convert speech into text (also text into speech).

3.3.3. Verify Valid Input/Command

The system should verify the command and check whether the command is a valid command or not.

3.3.4. Command Execution

The proposed system should perform corresponding task while executing the verified command.

3.3.5. Automate User Tasks

The proposed system should allow user to automate different tasks in a generic manner.

3.3.6. Add Custom Commands

The proposed system should enable the user to add custom verbal commands for the automated tasks.

3.3.7. Replay Automated Tasks

The proposed system should also allow user to replay automated tasks anytime user wants to using corresponding verbal custom commands.

3.4. Non-Functional Requirements

Any requirement that determines how the system performs a particular function. Any requirement which is not a functional requirement is a non-functional requirement. Non-functional requirements are required to develop an efficient and a quality software product that is why they are also known as the Quality Attributes. Performance, Efficient In Use, Modularity, Quality, Intuitiveness, Functional are the non-functional requirements that the proposed system should meet.

3.4.1. Performance

Throughput, utilization, response time are some parameters to evaluate the performance of a system. The performance of a system depends on the time and space consumed by that system. Simply put, if a system uses less time and uses less space and runs efficiently, then the system has the best performance. The proposed system should has the best performance that a virtual assistant can have. It takes less time and requires less space.

3.4.2. Efficient In Use

A software product can be said to be efficient if it responds quickly and with the desired output. The proposed system is efficient in use as it takes less time and produces the desired output. The proposed system required a mechanism for finding and verifying commands because the system allows the user to automate any number of tasks and specify verbal custom commands for them. Therefore, efficient and effective techniques are used to develop the proposed system to achieve efficiency.

3.4.3. Modularity

A computer program can be broken down to sub-programs called modules and this approach is called modular programming. This approach has many advantages such as Ease of Use, Reusability, and Ease of Maintenance. The proposed system is mainly divided into four modules. Voice Recognition Module, Command Verification Module, Command Execution Module, and Task Automation Module are the four main modules of the proposed system.

3.4.4. Quality

When implementing software, developers consider the quality of the software. Software Quality deals with code's consistency and readability. Appropriate structure and comments are very helpful in this regard. This can help other developers understand the code and upgrade the system to new requirements. Efforts were made to meet this need during the development of the proposed system.

3.4.5. Easy To Use

For the convenience of the user, every software product should be easy to use. This means that the user can easily interact with the system and achieve the desired results without much effort. Efforts were also made to meet this need during the development of the proposed system.

3.4.6. Functional

A software is functional if it works fine and has no errors. The proposed system is functional as it works fine and does not show any error.

Chapter 4

SYSTEM MODELING AND DESIGN

Chapter 4 is about system modeling and various modeling techniques. These modeling techniques can be use-case diagrams, system sequence diagrams, data flow diagrams, sequence diagrams, class diagrams, and activity diagrams of the proposed system. The proposed system aims to create a single virtual assistant with the basic capabilities of a voice assistant, as well as giving users the ability to automate their tasks. The modules and the architecture of the proposed system can be better described using modeling techniques called system design. The various diagrams or modeling techniques used to describe the architecture of the proposed system include use-case diagrams, activity diagrams, and system sequence diagrams.

4.1. System Design and Analysis

As already discussed, the modeling techniques used for the proposed system include the use-case diagram, the activity diagram, and the system sequence diagram. Use-case diagram is used to depict different roles in a system and how do these roles interact with the system. The second modelling technique used is the activity diagram. This is a type of flow chart used to model the flow from one activity to another. The last modelling technique used is the system sequence diagram or SSD. SSDs are used to indicate when and how different tasks are completed in a system. Each of the diagrams for the proposed system will now be discussed in detail.

4.2. Use Case Diagram

The use-case diagram used is a very important modeling technique because it shows the interaction of the user with the system. It is used to demonstrate the functionalities of the proposed system performed with user’s interaction. It is used to view the system from a user's perspective. The main purpose of the use case diagram is to identify the functions and the way the user interacts with them. The use-case diagram of the proposed system depicts the user’s interaction with system. Use-case diagram of the proposed system depicts different functionalities of the system such as Execute Command, Automate Task, Add Custom Command, and Replay Automated Task in the Figure 4.1.

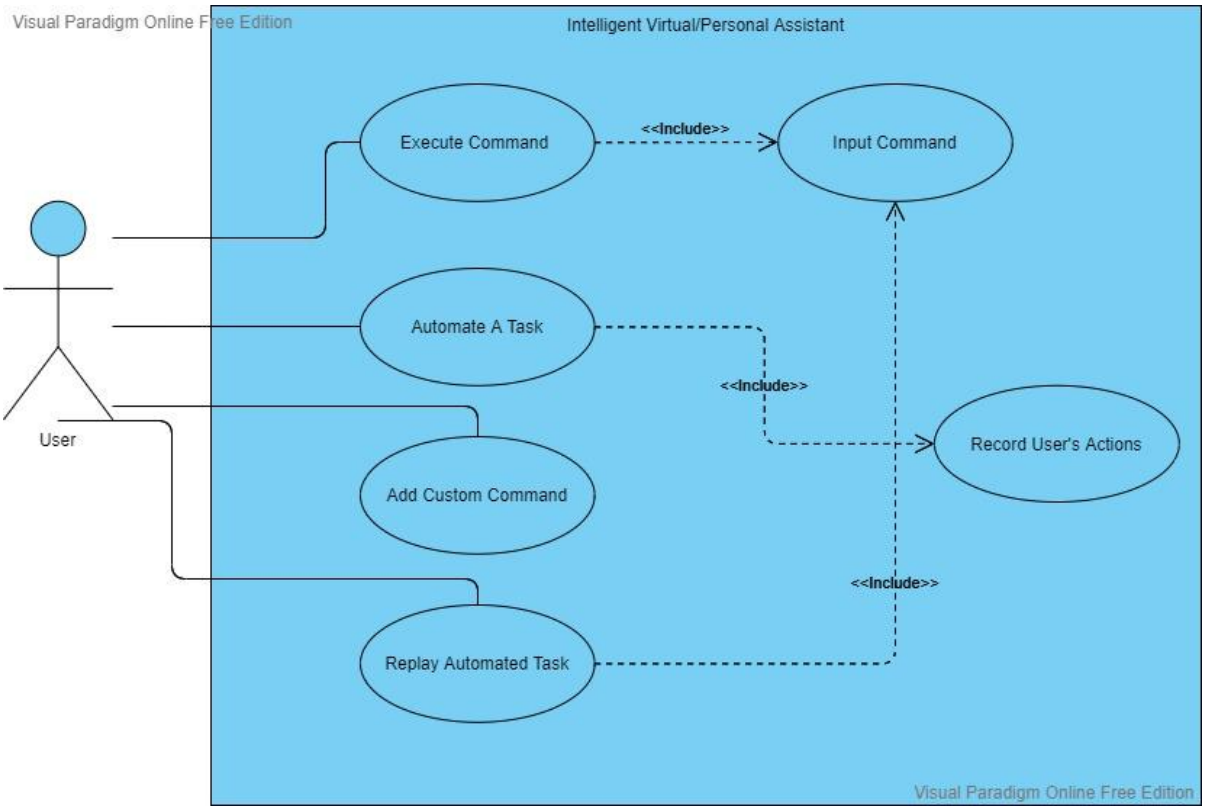


Figure 4.1: Use-Case Diagram

4.3. Activity Diagram

The second modeling technique used to describe the architecture of the proposed system is the activity diagram. This is a type of flow chart used to model the flow from one activity to another. How activities are coordinated to provide services is described by the activity diagram. It is used to model the workflow between the use-cases. The activity diagram of the proposed system is used to model the control or work flow of various activities (Such as Input User Command, Record User's Actions, Add Custom Command, Link Command With Automated Task, and Execute Command) of the proposed system as shown in the Figure 4.2.

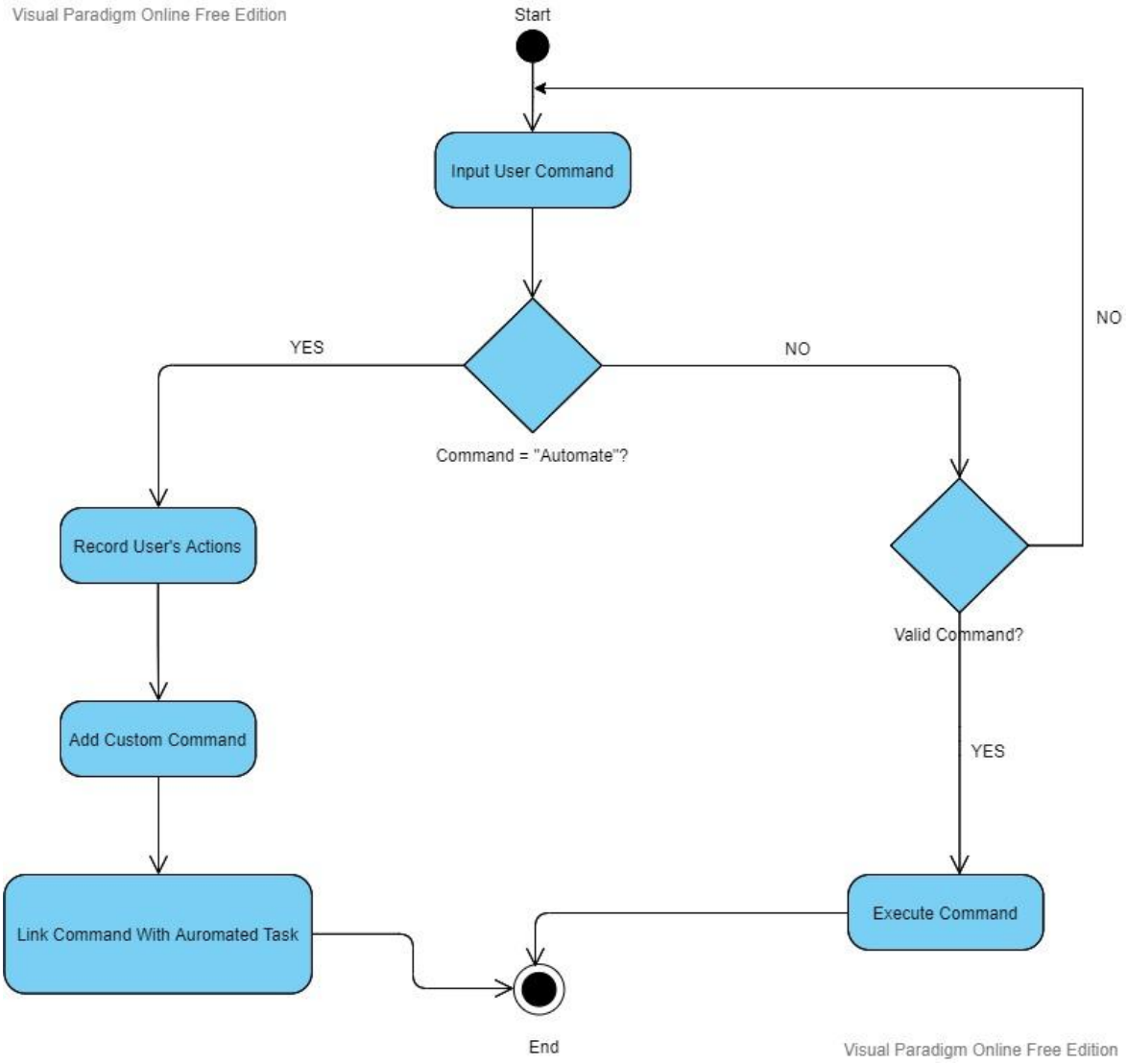


Figure 4.2: Activity Diagram

4.4. System Sequence Diagram

The last modelling technique used is the system sequence diagram or SSD. SSDs are used to indicate when and how different tasks are completed in a system. It shows how the user uses the software. The system sequence diagram is used to describe how different functions are performed. It is also used to describe how a message is sent by the actor and the system responds to a request. System sequence diagram for proposed system is shown in the given Figure 4.3.

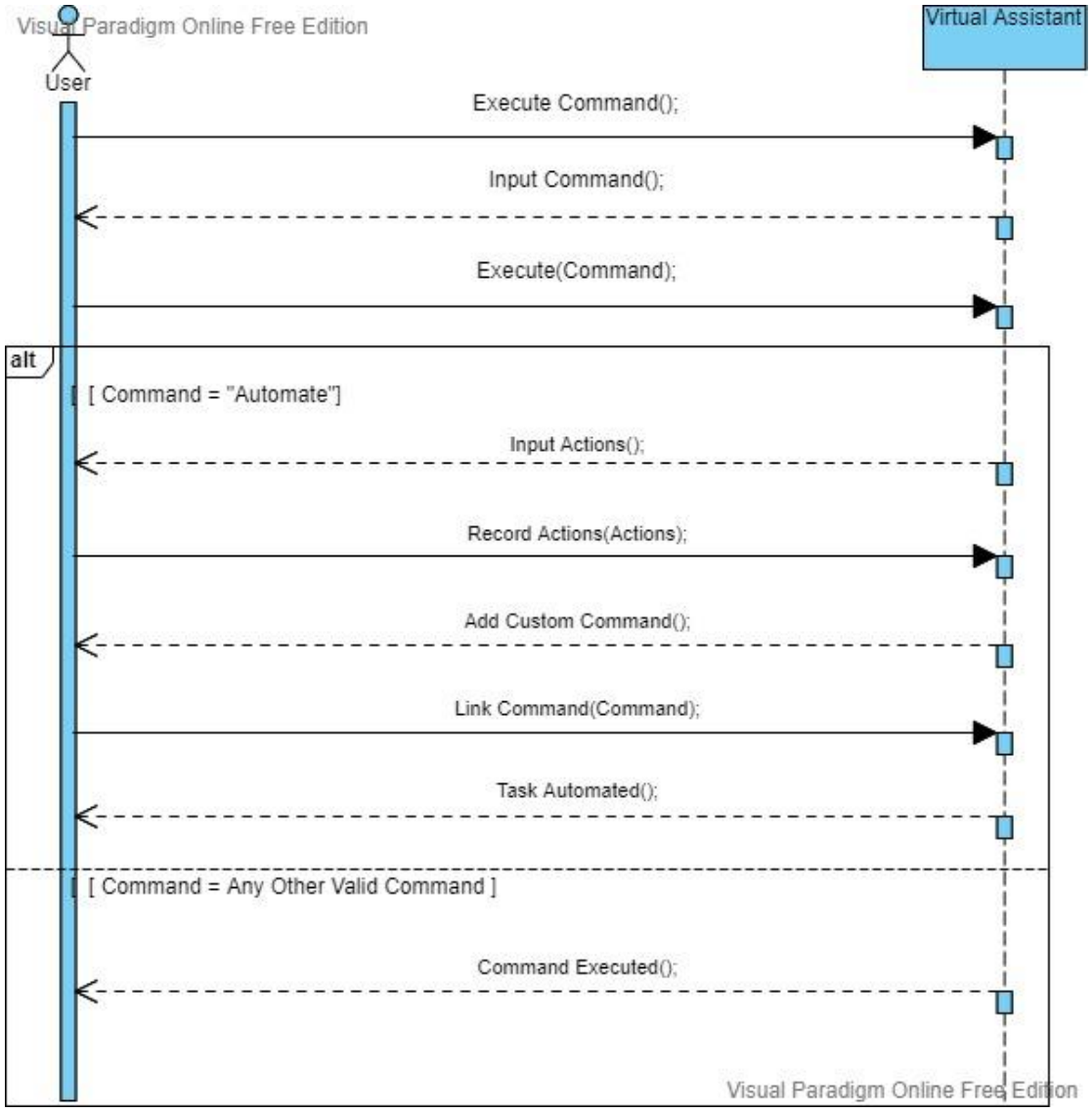


Figure 4.3: System Sequence Diagram

SSDs are used to indicate when and how different tasks are completed in a system.The system sequence diagram is used to describe how different functions are performed.

Chapter 5

SYSTEM TESTING AND VALIDATION

Software testing and validation is the most crucial phase in the development of any of any software product. Without testing the software product one can not handover the software product to the customer or consumer or user and it can not be deployed. There are many testing techniques and test stages. Unit Testing, Functionality Testing, Integration Testing, System Testing, Interface Testing, Performance Testing, Regression Testing, Acceptance Testing, and Pilot Testing are different stages for testing software product. Almost every software has more than one module so first stage of testing a software is Unit Testing which means every module will be tested individually. After unit testing comes the second stage of testing which is Integration Testing. In this stage different modules are integrated and then Integration Testing is performed. Functionality Testing is testing technique in which software product is tested to know that whether the proposed system fulfills the functional requirements and specifications. When software is fully and completely integrated then System Testing is performed. System Testing is done to check how all the components of the proposed system interacts with each other and proposed system is tested against all the possible inputs. There two types of testing: White Box Testing and Black Box Testing. White Box Testing is to test the internal workings and code of the proposed system. Whereas the Black Box Testing is to test the external workings of the proposed system from user's perspective. System Testing is a type of Black Box Testing. There are many other testing techniques which will be discussed later on. This chapter talks about all the testing techniques that are used to test the proposed system.

5.1. System Testing

It has already been discussed that the testing and validating the software product is the most crucial phase in the development of any software product. Different testing techniques are designed to test and validate the proposed system. Some of the testing techniques are: Unit Testing, Integration Testing, Functionality Testing, and System Testing. System Testing is performed when the proposed system is fully integrated. This test is performed to test whether all the components are working properly and has the quality coordination with each other. Software product is tested against every possible input. System Testing is a type of Black Box Testing. System Testing is very important because without testing the fully integrated software, the product can not be handed over

to user or customer. System Testing is done to remove all the errors and fix all the bugs found during testing the proposed system. Different testing techniques are applied to the proposed system. Some of them are: Unit Testing, Integration Testing, Functional Testing, System Testing or Black Box Testing.

Proposed system has been tested using different testing techniques. First testing technique used is Unit Testing. Proposed system has four modules. Voice Recognition Module, Command Verification Module, Command Execution Module, Task Automation Module are the four modules of proposed system. Unit Testing is done against every module individually and every module produced desired outputs. Next technique used to test the proposed system is Integration Testing. All the modules were integrated one by one and then were Integration Testing was performed at all levels of integration. The proposed system generated required outputs against Integration Testing at each level of integration. After fully Integrating the proposed system the System Testing or Black Box Testing was performed on proposed system. Every possible input (user or consumer perspective was considered) was given to proposed system. Against each and every input the proposed product produced desired results. After Successful completion of System Testing successor testing technique called Functional Testing was performed against the proposed system. This Functional Testing was performed check whether software products fulfills the functional requirements and specifications. Success was the result against the Functional Testing. Details of applied testing techniques are discussed below in this chapter.

5.2. Testing Techniques

Testing techniques are designed to test any software product. Software product must be tested using different testing techniques before handing it over to the consumer. Unit Testing, Integration Testing, System Testing, Functional Testing, White Box Testing, Black Box Testing, Performance Testing and Acceptance Testing are various testing techniques that are used to test a software product. Every testing technique has its own features and benefits and purpose. Almost every software has more than one module so first stage of testing a software is Unit Testing which means every module will be tested individually. After unit testing comes the second stage of testing which is Integration

Testing. In this stage different modules are integrated and then Integration Testing is performed. Functionality Testing is testing technique in which software product is tested to know that whether the proposed system fulfills the functional requirements and specifications. When software is fully and completely integrated then System Testing is performed. System Testing is done to check how all the components of the proposed system interacts with each other and proposed system is tested against all the possible inputs. Suitable testing techniques are chosen according to different requirements to test the proposed system keeping the functional requirements of the proposed system in view. Different testing techniques are applied to the proposed system. Some of them are: Unit Testings, Integration Testing, Functional Testing, System Testing or Black Box Testing.

5.2.1. Unit Testing

As almost every software product has more than one module. And each module is called a unit. Testing performed on individual unit is called Unit Testing. Each module is tested against every possible input. The module should produce desired results at unit level. The reason why one should apply Unit Testing is that the possible bugs and error will be removed at unit level. These bugs and errors if not dealt at unit level will cause troubles and problems later on. So it is better to remove these bugs and error at beginning.

The proposed system has four modules. Voice Recognition Module, Command Verification Module, Command Execution Module, Task Automation Module are the four modules of proposed system. Unit Testing is done against every module individually and every module produced desired outputs. Voice Recognition Module takes input in the form of audio and then converts that audio to text. This module produced desired results when tested. Command verification Module is the second module. The output of Voice Recognition Module is input to Command Verification Module. This text is checked whether it is a valid command or invalid command. This module worked properly when tested. Successor to Command Verification Module is the Command Execution Module it takes the output of its predecessor as input. The verified command is executed. When Unit Testing applied to this module it worked properly without any error.

Task Automation module is the last module. This modules enables user to automate any task he/she wants using mouse and keyboard events or actions and by adding image conditioning. This module automated tasks when tested using Unit Testing technique.

5.2.2. Integration Testing

After testing every module at unit level comes the Integration Testing. In Integration Testing all of the modules are integrated one by one and Integration Testing is performed at each integration level. The integration is done to check whether the modules at any integration level work properly or not. Sometimes modules that were working fine tested at unit level but when integrated with other modules they generate errors. This is because the coordination between multiple modules has some flaws or logical errors. So to remove these errors and bugs Integration Testing is used to test modules at each integration level. Integration can be done in two ways. First approach is to integrate the modules from top to bottom and the second approach is to integrate the modules from bottom to top. The proposed system is integrated using first approach (Top to Bottom Approach).

Voice Recognition module and the Command Verification module were integrated at first level of integration. When Integration Testing was performed against the product of first level of integration the proposed system produced desired results and system was successful in taking audio input and converting that audio to text and then passing that text to next module and this modules was successful in verifying that text as command. At second level of integration Command Execution Module was integrated to the product of first level integration. When tested the product of second level of integration was successful in taking command as input form the product of first level of integration and then executing that command. At third level of integration the Task Automation Module was integrated to the product of second level of integration. The product of third level of integration was successful in automating the task and then adding custom command for that automated task.

5.2.3. System Testing

System Testing is performed when the software product is completely integrated. System Testing is used to check whether all the components of the proposed system is working fine as a whole. System Testing is also called Black Box Testing. In this test the proposed system is tested against every possible input as a whole. System Testing is concerned with external working of the software or user's perspective of the proposed system.

Proposed system was tested using System Testing technique. Proposed system produced desired results when tested as a whole against every possible input (User's Perspective of Proposed System). Proposed system at the end of this test was able to automate any task and executing any command when provided with audio as input.

5.2.4. Functional Testing

Functional Testing is done to check whether the propose system fulfills the functional requirements or not. It is used to validate the proposed system against functional requirements and specifications.

The functional requirements of the proposed system are Take Voice Input, Voice To Text Conversion, Command Verification, Command Execution, Task Automation, Adding Custom Command, Replay Automated Tasks. Proposed system was able to fulfill all the functional requirements when tested using Functional Testing Technique.

5.3. Test Cases

Test case is some situation or scenario to test some functionality of a system. The test case is designed to test specific functionality of proposed system. To test the proposed system using any testing technique test cases are required. First testing team must design the test cases then the proposed system will be tested against each test case. If the proposed system passes the test case and produces desired result it means the proposed system is able to perform the required functionality. To test any software

product test cases are must. In this section different test cases for the proposed will be discussed in detail. Two test cases was designed to test the proposed system.

5.3.1. Test Case1: Execute Command

Execute Command is the first test case. This test case is designed to check whether the proposed system is able to perform the required functionality or not against the voice input. Test case is described in detail below in table 5.1.

Table 5.1: Test Case For Command Execution

General Information			
Test stage:	System Testing		
Test date:	November/15/2021		
Tester:	Uzair Iqbal And Saeed Muhammad	**Test case number:**	Test Case 01
Test case description:	Testing whether the proposed system executes the required functionality provided with audio as input.		
Results:	Pass		
Introduction			
Requirement(s) to be tested:	Requirement is to execute a command when provided with voice input.		
Roles and Responsibilities:	Anyone can perform the role. (User or Tester)		
Set up procedures:	Tester requires microphone to provide the voice input. Fast internet connection is necessary for voice recognition.		
Stop procedures:	With voice input when required command is executed or required functionality is performed the test will be terminated.		
Environmental Needs			
Hardware:	Desktop PC or Laptop, 4GB RAM, Fast Internet Connection		
Software:	Windows 8/8.1/10/11		

Test	
Test Items and Features:	Features to be tested are the voice input, audio to test conversion, command verification, command execution and check if the required functionality is performed against the command.
Input Specifications:	To execute a command the tester will require a microphone to give voice input.
Procedural Steps:	When proposed system is executed the tester or user will be asked for voice input. The user will provide the voice input. This input will be converted to text and checked if the text is a valid command or not. If text is valid command then the required functionality will be executed.
Expected Results of Case:	The expected result is that the desired functionality will be executed and the proposed system will ask for next command to execute.

5.3.2. Test Case2: Add Custom Command

Add custom command is the second test case. This test case is designed to check whether the proposed system is able to automate any task and then add custom command for that custom command or not. Task automation will be done through mouse and keyboard events and the conditioning can be added using images or screenshots. This test case is discussed in details below in table 5.2.

Table 5.2: Test Case For Add Custom Command

General Information			
Test stage:	System Testing		
Test date:	November/20/2021		
Tester:	Sheraz Ibrahim	**Test case number:**	Test Case 02
Test case description:	Testing whether the proposed system can automate any task and then add custom command for it or not provided with voice input.		
Results:	Pass		

Introduction	
Requirement(s) to be tested:	Requirement is to automate any task and then add custom command for that automated task when provided with voice input.
Roles and Responsibilities:	Anyone can perform the role. (User or Tester)
Set up procedures:	Tester requires microphone to provide the voice input. Fast internet connection is necessary for voice recognition.
Stop procedures:	With voice input when any task is automated and a custom command for that automated task is added the test will be terminated.
Environmental Needs	
Hardware:	Desktop PC or Laptop, 4GB RAM, Fast Internet Connection,
Software:	Windows 8/8.1/10/11
Test	
Test Items and Features:	Features to be tested are the voice input, audio to test conversion, command verification, command execution, task automation, and adding custom command.
Input Specifications:	To execute a command the tester will require a microphone to give voice input.
Procedural Steps:	When proposed system is executed the tester or user will be asked for voice input. The user will provide the voice input. This input will be converted to text and checked if that text is a valid command or not. If text is a valid command and if command is to automate any task and add custom command for it. It will record user's actions (through keyboard and mouse inputs and adding conditions using images). When task is automated the user will be asked to add custom command for it. After adding the custom command the test will be terminated.

Expected Results of Case:	The expected result is that any task is automated and custom command is added for it and the proposed system will ask for next command to execute.

5.4. Non-Functional Requirements

Any requirement that determines how the system performs a particular function. Any requirement which is not a functional requirement is a non-functional requirement. Non-functional requirements are required to develop an efficient and a quality software product that is why they are also known as the Quality Attributes. Performance, Efficient In Use, Modularity, Quality, Intuitiveness, Functional are the non-functional requirements that the proposed system should meet.

5.4.1. Performance

Throughput, utilization, response time are some parameters to evaluate the performance of a system. The performance of a system depends on the time and space consumed by that system. Simply put, if a system uses less time and uses less space and runs efficiently, then the system has the best performance. The proposed system should have the best performance that a virtual assistant can have. It takes less time and requires less space.

5.4.2. Efficient In Use

A software product can be said to be efficient if it responds quickly and with the desired output. The proposed system is efficient in use as it takes less time and produces the desired output. The proposed system required a mechanism for finding and verifying commands because the system allows the user to automate any number of tasks and specify verbal custom commands for them. Therefore, efficient and effective techniques are used to develop the proposed system to achieve efficiency.

5.4.3. Modularity

A computer program can be broken down to sub-programs called modules and this approach is called modular programming. This approach has many

advantages such as Ease of Use, Reusability, and Ease of Maintenance. The proposed system is mainly divided into four modules. Voice Recognition Module, Command Verification Module, Command Execution Module, and Task Automation Module are the four main modules of the proposed system.

5.4.4. Quality

When implementing software, developers consider the quality of the software. Software Quality deals with code's consistency and readability. Appropriate structure and comments are very helpful in this regard. This can help other developers understand the code and upgrade the system to new requirements. Efforts were made to meet this need during the development of the proposed system.

5.4.5. Easy To Use

For the convenience of the user, every software product should be easy to use. This means that the user can easily interact with the system and achieve the desired results without much effort. Efforts were also made to meet this need during the development of the proposed system.

5.4.6. Functional

A software is functional if it works fine and has no errors. The proposed system is functional as it works fine and does not show any error.

Chapter 6
CONCLUSION

The sixth chapter of this report is the conclusion. It talks about the limitations of the proposed system and the future work that should be done to enhance the proposed system. This chapter has a brief summary of the proposed system. The summary contains brief details of salient features, limitations of the proposed system and future work that should be done to improve the proposed system. This chapter has two sections. In the first section of this chapter the key features, attributes and major functionalities of the proposed system are discussed. Whereas the second part of this chapter gives brief explanation of limitations and future work of the proposed system.

6.1. Conclusion

The proposed system is a personal assistant which enables the user automate his/her tasks in generic manner. The proposed system has two main modules voice assistant module and task automation module. The first module of the proposed system takes voice input and then performs voice to text conversion. The text obtained from voice to text conversion will be checked whether its is a valid command or not. If that command is a valid command, the proposed system will perform desired functionality corresponding to that command.

Whereas the second modules allows user to automate any task in a generic way. This task automation is achieved by recording user's action through mouse events (mouse move, mouse left click, mouse right click, mouse middle click, and mouse scroll) and keyboard events (key down, key up, key press and, hot key combination). These recorded actions will be saved in a temporary file called script. Another very interesting feature of the proposed system is that the user can edit that script in any text editor. User has the option to add conditions using images or screenshots. When user is done editing the script the user will save the file in a specific folder where the proposed system can access it and execute it when required. The user can add custom command for that automated task. The user can execute that automated task recorded and saved in the form of a script using that custom command.

The proposed system as a whole can perform desired functionality when provided with voice input. It can execute custom commands by executing the corresponding script.

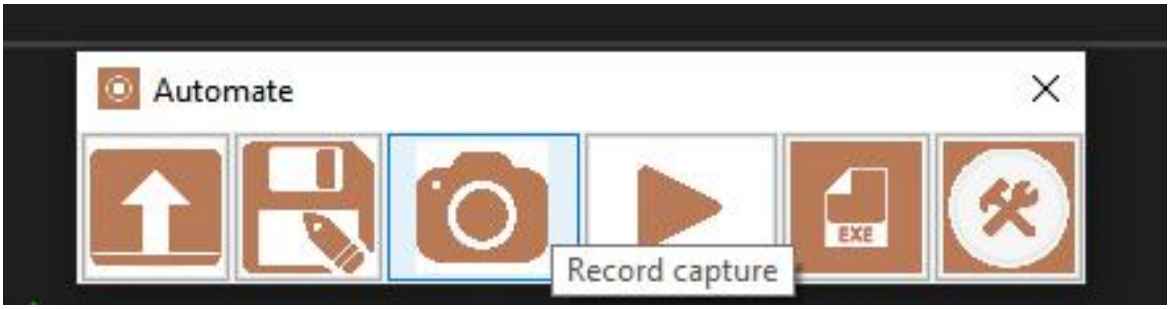
6.2. Limitation and Future Work

In this section limitations of the propose system and the future work to done to enhance the proposed system will be discussed in detail. The proposed system has some limitations which can be overcome in future. Major limitation is that it stores scripts of the automated tasks in a specific folder on local storage of the computer. The future work that should be done to overcome this limitation is that a database should be used to save the scripts and a proper mechanism should be devised to access and load the script from database when required.

Some assistants has a special feature of remote access to access the assistant through any other device. The proposed system has this limitation that it has no such feature. To overcome this feature in future a website should be designed through that website the assistant can be accessed and controlled. But one thing that should be considered is that website should have a proper authentication system. Whosoever wants to access his/her assistant will login to that website and then he/she can access his/her assistant and control his/her computer through assistant using remote access feature.

APPENDIX - I

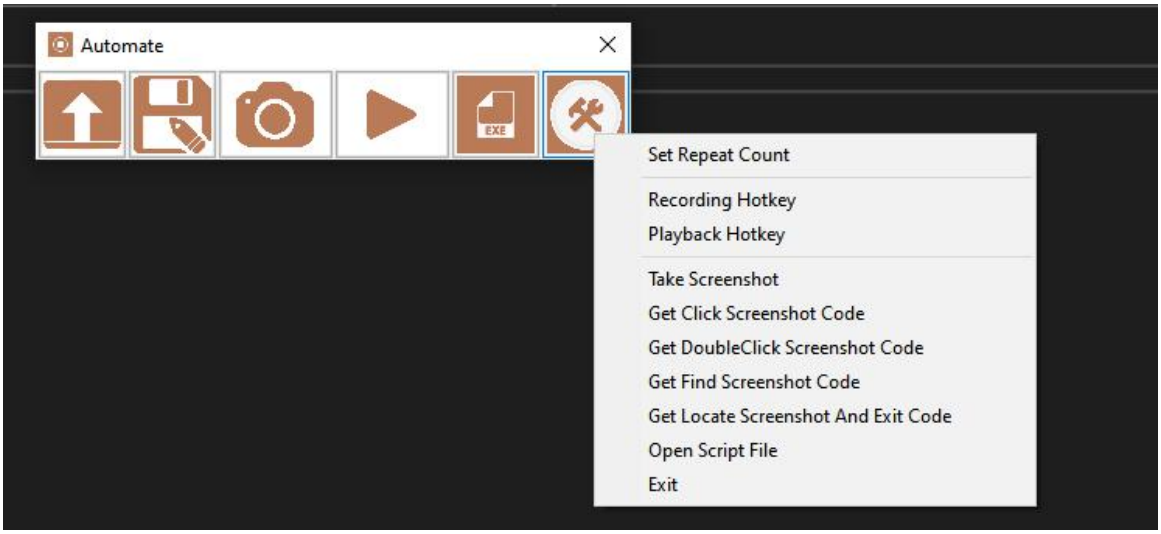
This following image shows the GUI of the Task Automation Module.



The first button is a Load File Button which is used to load a script of any automated task. The second button in this UI is Save File Button which is used to save a script in a specific folder where proposed system can access it to perform execution. **"C:\Users\Username\AppData\Roaming\automate"** is the folder where all the scripts are saved. The third button in the same row is the Record Button with camera icon on it. This button is a toggle button when it is toggled to true proposed system starts recording user's actions. When toggled again proposed system will stop recording and will save the recorded actions in a temporary file in the form of a script. The fourth button the same row is Play Button which is used to replay the recorded actions to check whether the automated task has been recorded in the right sequence or not. The Fifth in the same row is Execution Button. This button is used to convert the script file into object file of python and it saves the object file where ever user wants with ".pyc" extension. The last button is Settings Button which is used to change settings of the proposed system. It also has the ability to make changes to the script files. This functionality of this button is discussed in detail in Appendix - II.

APPENDIX - II

The image below shows the Settings Button Menu.



The menu of the Settings Button has ten options. First option is *Set Repeat Count* which is used to set count of repetition that how many time the automated task should be replayed. The second option is *Recording Hotkey* which is used set hot key for recording. *Playback Hotkey* is used to set playback hot key. *Take Screen Shot* is used to take a screen shot of the screen of a specific region. *Get Click Screenshot Code* is used to generate code to add condition to click a specific area on screen based on the information provided in image. *Get Double Click Screenshot Code* is used to generate code to add condition to double click a specific area on screen based on the information provided in image. *Get Find Screenshot Code* is used to generate code to add condition to find a specific image on screen based on the information provided in image. *Get Locate Screenshot And Exit Code* is used to generate code to add condition to locate an image on screen and stop execution of script based on the information provided in image. *Open Script File* is used to edit the script and add conditions discussed above. *Exit* is used to exit this Task Automation Module.

REFERENCES

1. Rasika Anerao , Utkarsh Mehta , Sharangdhar Vaze , G. Hrishikesh. “Personal Assistant to Facilitate User Task Automation.” International Journal of Computer Trends and Technology (IJCTT) – volume 15 number 4 – Sep 2014 [online], Available: https://www.ijcttjournal.org/archives/ijctt-v15p133 [Accessed : May 2021]

2. Matthew B. Hoy (2018) Alexa, Siri, Cortana, and More: An Introduction to Voice Assistants, Medical Reference Services Quarterly, 37:1, 81-88, DOI: 10.1080/02763869.2018.1404391 [online], Available: https://doi.org/10.1080/02763869.2018.1404391 [Accessed : May 2021]

3. ZubairM Paul, HeenaReyaz Bhat, Tanveer lone. “CORTANA-INTELLIGENT PERSONAL DIGITAL ASSISTANT: A REVIEW.” International Journal of Advanced Research in Computer Science - Volume 8, No. 7, July – August 2017 [online], Available:DOI: http://dx.doi.org/10.26483/ijarcs.v8i7.4225 [Accessed : May 2021]

4. Bisma Shakeel, Tabasum, Mir Shahnawaz Ahmad. “SIRI – APPLE’s PERSONALASSISTANT: A REVIEW”. IJCSMC, Vol. 6, Issue. 7, July 2017, pg.44 – 48 [online], Available: https://ijcsmc.com/docs/papers/July2017/V6I7201706.pdf [Accessed : May 2021]