Minor Project Report on

Workspace Pro - My Control Center

Bachelor of Vocation

In

Software Development

Guide:

Dr. Kamlesh kumar Pandey
Mr. Anurag Singh

Submitted By:

Name – Kuldeep Chouhan

Enrollment No - 2201123008

Submitted To:

Department of Vocational Education

Faculty of Technical, Vocational Education and Skill Training

Date of Submission: 05/11/2024

इन्दिरा गांधी राष्ट्रीय जनजातीय विश्वविद्यालय
Indira Gandhi National Tribal University
अमरकंटक (म.प्र.) | Amarkantak (M.P)
(भारतीय संसद में पारित अधिनियम द्वारा स्थापित केन्द्रीय विश्वविद्यालय)
(A Central University Established by an Act of Parliament of India)

Minor Project Report on

Workspace Pro - My Control Center

Subject Code: SDL-507
Session: 2024-2025

Submitted By:

Name: Kuldeep Chouhan — Signature

Enrolment: 2201123013

B. VOC in Software Development

Under the guidance of:

Dr. Kamlesh Kumar Pandey — Signature

Mr. Anurag Singh — Signature

Department of Vocational Education

Prof. Raghavendra Mishra Sir

HoD (Department of Vocation Education) — Signature

Department of Vocational Education
Faculty of Technical, Vocational Education and Skill Training

इन्दिरा गांधी राष्ट्रीय जनजातीय विश्वविद्यालय
Indira Gandhi National Tribal University
अमरकंटक (म.प्र.) | Amarkantak (M.P.)
(भारतीय संसद में पारित अधिनियम द्वारा स्थापित केन्द्रीय विश्वविद्यालय)
(A Central University Established by an Act of Parliament of India)

CERTIFICATE OF APPROVAL

This is to Certify that the project the entitled "WORKSPACE PRO - MY CONTROL CENTER", carried out by "Kuldeep Chouhan", Enrollment No.2201123008, a student of B.Voc in Software Development 5th semester at Indira Gandhi National Tribal University Amarkantak (M.P.), is hereby approved after proper examination and evaluation as a creditable work for the partial fulfillment of the requirement for minor project of Bachelor of Vocation (B.VOC) in Software Development from Indira Gandhi National Tribal University Amarkantak (M.P.). The work carried out during a period from July, 2024 to Nov, 2024 under the guidance of Dr. Kamlesh Kumar Pandey and Mr. Anurag Singh, Department of Vocational Education.

(Internal Examiner)	(External Examiner)
Name:	Name:
Designation:	Designation:
Date: 05/12/2024	Date: 05/12/2024

इन्दिरा गांधी राष्ट्रीय जनजातीय विश्वविद्यालय
Indira Gandhi National Tribal University
अमरकंटक (म.प्र.) | Amarkantak (M.P.)
(भारतीय संसद में पारित अधिनियम द्वारा स्थापित केन्द्रीय विश्वविद्यालय)
(A Central University Established by an Act of Parliament of India)

DECLARATION

I hereby declare and certify that, the Minor Project entitled “WORKSPACE PRO - MY CONTROL CENTER” is a bonafide record of project development carried out by me during the year 2024-25. Further certify that the work presented in the report is original and carried out according to the plan as possible in the proposal and guidelines of the, Department of Vocational Education, IGNTU.

Name – Kuldeep Chouhan

Enrollment No:- 2201123008

इन्दिरा गांधी राष्ट्रीय जनजातीय विश्वविद्यालय
Indira Gandhi National Tribal University
अमरकंटक (म.प्र.) | Amarkantak (M.P.)
(भारतीय संसद में पारित अधिनियम द्वारा स्थापित केन्द्रीय विश्वविद्यालय)
(A Central University Established by an Act of Parliament of India)

ACKNOWLEDGMENT

We have great pleasure in submission of this project report entitled software in python ‘WORKSPACE PRO - MY CONTROL CENTER’ for partial fulfillment of the Degree Of Vocational Education in Software Development, while submitting this project report.

I would like to express my sincere appreciation and gratitude to all those who have contributed to the “WORKSPACE PRO - MY CONTROL CENTER” software in python.

We would like to extreme delight and thank fullness our sir Dr. Kamlesh Kumar Pandey, Mr. Harish Vishawakara, Mr. Anurag Singh and Head of the Department Prof. Raghavendra Mishra sir, who have provided us the opportunity and organized project for us.

Without their active co-operation and guidance. It would have become very difficult to complete task in time. I would also like to thank my all faculty members, my family, and my friends to being a foundation of love and support during the term of my minor project.

Name- Kuldeep Chouhan

Enrollment No.2201123008

इन्दिरा गांधी राष्ट्रीय जनजातीय विश्वविद्यालय
Indira Gandhi National Tribal University
अमरकंटक (म.प्र.) | Amarkantak (M.P.)
(भारतीय संसद में पारित अधिनियम द्वारा स्थापित केन्द्रीय विश्वविद्यालय)
(A Central University Established by an Act of Parliament of India)

ABSTRACT

This project is a personal GUI desktop application created to help increase productivity and reduce distractions. Often, starting a laptop can lead to unplanned activities like watching movies, browsing the internet, or playing games for hours. These distractions can waste valuable time and affect focus. To solve this issue, this application provides a structured way to access only the essential tools and tasks.

The application features a clean and simple interface with a sidebar that contains buttons for opening important applications, sending WhatsApp messages, and sending emails. These functionalities are designed to streamline tasks and provide quick access to the tools needed for work or personal projects.

The main goal of this project is to keep the user focused by reducing the temptation to engage in distracting activities. This application is not meant for public use; it is tailored specifically to the creator’s needs. By using this tool, the user can manage their time more effectively, ensuring that they stay on track with their goals.

Built entirely in Python, the project utilizes various libraries to create a user-friendly graphical interface and implement functionalities like email sending and messaging. This application is an example of how technology can be used to create personalized solutions for improving productivity and achieving a more disciplined workflow.

Table of Contents

1. INTRODUCTION

1.1 Project Description

This project is a personal GUI desktop application developed to help increase productivity and reduce distractions. It is designed for my personal use, addressing the problem of wasting time on unproductive activities when starting a laptop. The idea behind this project is to create a tool that focuses on helping me to stay disciplined and get work done efficiently.

The application is built using Python and has a user-friendly graphical interface created with the Tkinter library. When launched, the application opens a window with a sidebar. This sidebar contains buttons that allow me to quickly access important applications, send WhatsApp messages, and send emails.

Purpose of the Project

The primary goal of this project is to provide a structured and focused environment. Often, **when starting a laptop, there is a temptation to watch movies, play games, or browse social media, which leads to hours of wasted time.** This application prevents such distractions by providing quick access to essential tools, helping the user stay on track with their work.

Key Features of the Application

- **Sidebar with Buttons**:
  The sidebar contains buttons for opening frequently used applications. For example, it can open a browser, an IDE like Visual Studio Code, or other productivity tools the user needs for work.

- **Messaging Functionality**:
  The application includes a feature for sending WhatsApp messages directly from the interface. This makes communication easier without the need to open a browser or a separate app.

- **Email Sending**:
  The application has a built-in email-sending feature. This helps the user quickly compose and send emails without switching to another email client, saving time and effort.

- **No Distraction Design**:
  The interface is simple and does not include any unnecessary or distracting features. It is designed to provide only what is needed for productive work.

- **Custom Features for Personal Use**:
  Since the application is created for personal use, it is tailored to the specific needs of the user. For example, it includes shortcuts to tools and files that are essential for daily tasks.

How It Works

When the application is launched, the main window appears. The sidebar displays buttons for all the important features. Clicking on a button will execute the corresponding functionality. For instance, if the "Send WhatsApp Message" button is clicked, the application will allow the user to compose and send a message. Similarly, other buttons open specific applications or perform specific actions.

The application is designed to be lightweight and efficient. It is written in Python, and the Tkinter library is used to create the GUI. Additional libraries like pywhatkit, smtplib, and os are used for specific functionalities such as sending messages, emails, and opening applications.

Why This Project is Important

This application helps solve a very common problem faced by many people: getting distracted. By providing quick access to essential tools, it saves time and ensures that I will stay productive. It encourages a disciplined approach to work by making it easier to avoid distractions.

Challenges and Learning

During the development of this project, the main challenges included designing an intuitive interface, ensuring compatibility with different applications, and integrating messaging and email functionality. Working on this project allowed me to improve my Python programming skills and gain experience in building GUI applications.

This project is a simple but effective solution for improving productivity. It demonstrates how technology can be customized to suit individual needs and help achieve personal goals. By using this application, I can avoid distractions and focus on what truly matters, making it a valuable tool for everyday use.

2. System Study

The system study provides an overview of how this project was designed, developed, and implemented. It discusses the existing system, the proposed solution, and the tools, technologies, and requirements used for this project.

2.1 Existing and Proposed System

Existing System:

Before this project, there was no specific system in place to help manage distractions. When I started my laptop, I would often get distracted by activities like watching movies, playing games, or browsing social media. This resulted in a lot of wasted time and reduced productivity. The lack of a structured system meant that I could not focus on my work efficiently.

Proposed System:
The proposed system is a GUI desktop application designed to solve this problem. It provides a simple interface with buttons to access essential applications and features like sending emails and WhatsApp messages. This system ensures that the user can focus on important tasks without being tempted by distractions. It creates a structured environment where work can be prioritized over leisure activities.

2.2 Feasibility Study

Before starting this project, a feasibility study was conducted to check if this idea was practical and possible to implement.

1. Technical Feasibility:

The project is technically feasible because Python provides powerful libraries like Tkinter for creating graphical user interfaces. Additional libraries like pywhatkit and smtplib make it easy to integrate features like sending WhatsApp messages and emails. The required tools are readily available, and the application can run smoothly on most modern laptops.

2. Economic Feasibility:

This project is economically feasible as it does not require any special hardware or software purchases. All the tools and technologies used are free or open-source. The project was developed using the existing system, so no extra cost was involved.

3. Operational Feasibility:

The system is easy to use and directly addresses the problem of distractions. Its simple interface ensures that even users with basic computer knowledge can operate it. The application can significantly improve productivity, making it a valuable tool for the user.

2.3 Tools and Technologies Used

Several tools and technologies were used to develop this project. Below are the main ones:

1. Python Programming Language:
Python was used to write the application's code. It is a powerful, easy-to-learn language with many libraries that helped in creating this project.

2. Tkinter Library:
Tkinter was used to design the graphical user interface. It provided tools to create buttons, frames, menus, and other elements of the application window.

3. pywhatkit Library:
This library was used to add WhatsApp messaging functionality to the application. It made it easy to send messages directly from the app.

4. smtplib Library:
The smtplib library was used for the email-sending feature. It enabled the application to connect to an email server and send emails easily.

5. OS Module:
The OS module was used to open important applications and manage file paths.

6. Other Libraries:
Additional libraries like random and datetime were used for supporting functionalities like playing random music files and showing the current time.

2.4 Hardware and Software Requirements

For the smooth functioning of the application, certain hardware and software requirements need to be met. Below are the details:

Hardware Requirements:

1. **Processor:** A system with at least a dual-core processor for smooth execution.
2. **RAM:** A minimum of 2 GB of RAM is recommended for better performance.
3. **Storage:** At least 500 MB of free storage space is required to store files and dependencies.
4. **Display:** A standard display with a resolution of 1280x720 or higher.
5. **Other:** A microphone and internet connection are optional but useful for certain features like WhatsApp messaging and online searches.

Software Requirements:

1. **Operating System:** Windows 10 or higher.
2. **Python Version:** Python 3.10 or higher.
3. **Libraries:** Pre-installed Python libraries such as Tkinter, pywhatkit, smtplib, and os.
4. **Text Editor or IDE:** Any text editor like Visual Studio Code, PyCharm, or IDLE for writing and modifying the code.
5. **Other Software:** Email and WhatsApp accounts are required for the messaging and email functionalities.

This system study provides a complete understanding of the project's background, the problem it solves, and how it was implemented. By focusing on simplicity and usability, the application ensures the user can effectively manage their time and stay productive.

3.

Software Requirements Specification

This section describes the requirements for the software. It explains who the users are, what the system should do (functional requirements), and the qualities the system should have (non-functional requirements).

3.1 Users

The primary user of this application is **one person: KC7 - the developer of the project**. The application is designed specifically for personal use.

My Role:

1. I am both the developer and the only operator of the application.
2. I use the application to improve productivity and reduce distractions.

My Needs:

1. A simple interface to open important applications quickly.
2. Tools to send WhatsApp messages and emails directly from the desktop.
3. A distraction-free environment to avoid time-wasting activities like movies or games.

My Skills:

1. I has basic knowledge of computers and familiar with using GUI-based software.
2. I have sufficient technical skills to operate the app effectively.

3.2 Functional Requirements

Functional requirements describe the actions the system must perform. These are the features and functionalities of the application:

- **Opening Applications:** The system must allow me to open essential applications like Visual Studio Code, Command Prompt, Python IDLE, and web browsers with a single click.
- **Sending WhatsApp Messages:** The application must let me send WhatsApp messages directly from the desktop without opening the WhatsApp app manually.
- **Sending Emails:** The system must provide a feature to send emails using my email account.
- **Music Playback:** The application should allow me to play music from a specified folder on the system.
- **Accessing Python Documentation:** The app must provide a button to quickly open Python documentation for reference.
- **Task Status Display:** The application should display a simple status message in the interface when a task (like opening an app or sending a message) is performed.
- **Customizable Options:** I can modify file paths for applications, music folders, or other configurable options if needed.
- **Safe Exit:** The system must provide an option to exit the application safely without causing errors.

3.3 Non-Functional Requirements

Non-functional requirements define the qualities and standards the application should meet. These include performance, usability, and reliability aspects.

Performance Requirements:

1. The system must respond quickly to my inputs, with no noticeable delay in opening applications or performing tasks.
2. Sending emails or WhatsApp messages should not take more than a few seconds.

Usability Requirements:

1. The interface must be simple and easy to navigate, even for users with limited technical skills.
2. Buttons and labels must have clear names to indicate their functionality.
3. The design must focus on reducing distractions by avoiding unnecessary elements or animations.

Reliability Requirements:

1. The application must perform its functions without crashing or freezing.
2. All features like sending messages and emails must work as intended.
3. The application must handle errors gracefully, such as showing an error message if an email fails to send.

Scalability Requirements:

1. While designed for a single user, the system should allow the addition of new features or buttons if needed in the future.

Security Requirements:

1. The application should not store sensitive information like email passwords in plain text.
2. It must ensure that no unauthorized person can use the application.

Portability Requirements:

1. The system should work on any modern laptop or desktop running Windows 10 or higher. Minor adjustments may allow compatibility with other operating systems like Linux or macOS.

Maintainability Requirements:

1. The application code must be easy to understand and modify for future updates. Proper comments and documentation should be included in the codebase.

Resource Requirements:

1. The system must use minimal system resources like CPU and memory to ensure that it does not affect the performance of other applications.

This **Software Requirements Specification** provides a detailed understanding of the needs and expectations from the application. By focusing on both functional and non-functional requirements, the system is designed to be effective, reliable, and user-friendly.

4.
System Design

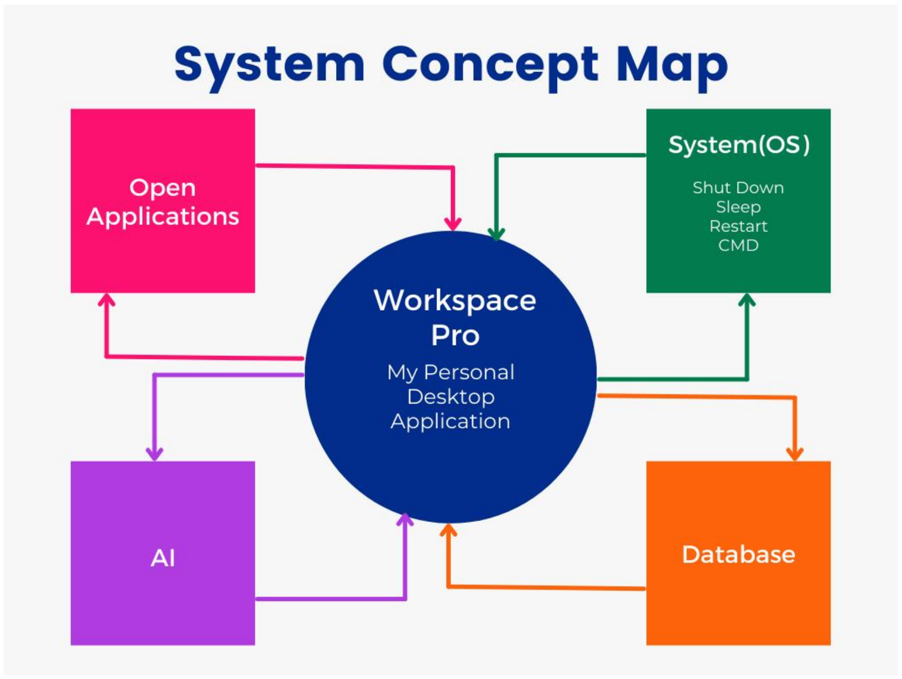
This section explains the design of the system, focusing on its structure and how its components interact.

4.1 System Perspective

The application is a personal productivity tool designed for a single user. It integrates features like app launching, messaging, and emailing into one interface. The system is standalone and does not depend on external servers, ensuring a fast and secure experience.

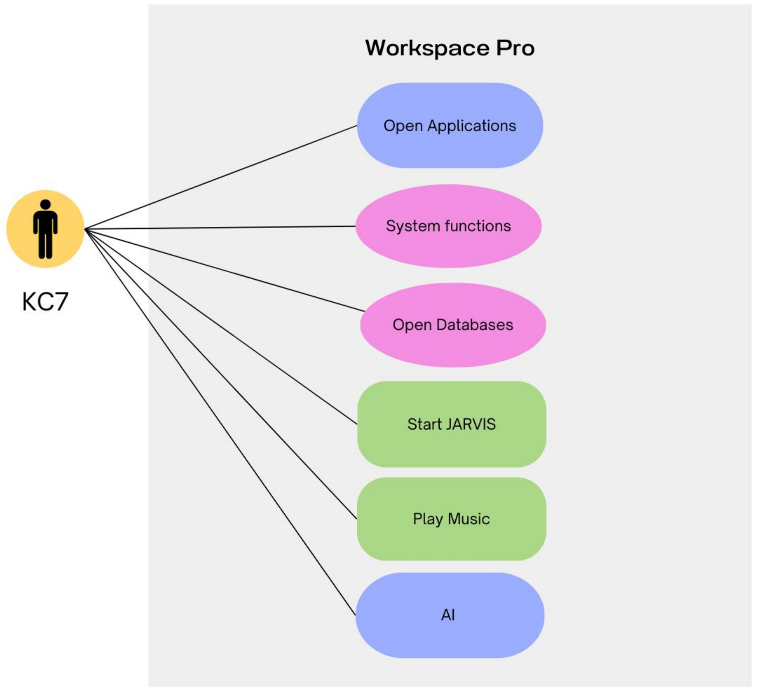
4.2 Context Diagram

The context diagram shows how the system interacts with the user. The user inputs commands (like opening apps or sending messages), and the system processes these tasks and provides the output through a simple GUI.



4.3 Use Case Diagram

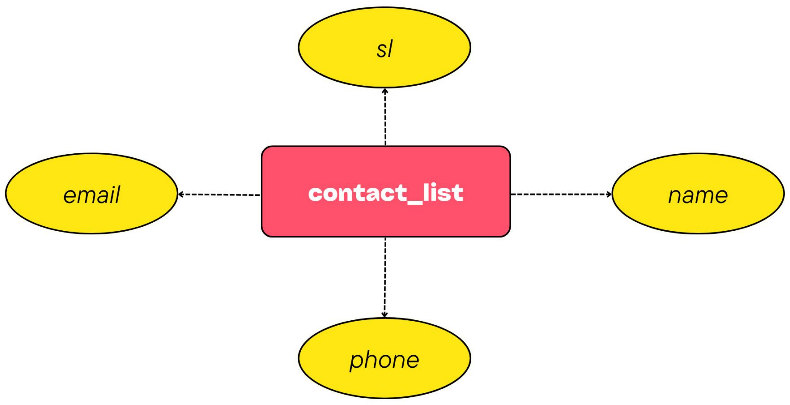
The use case diagram outlines the primary actions the user can perform, such as:



4.4 Database Design (ER)

The database stores the contact information of different persons, like their name, phone number and email.

This design ensures that the application is efficient, easy to use, and adaptable to future needs.



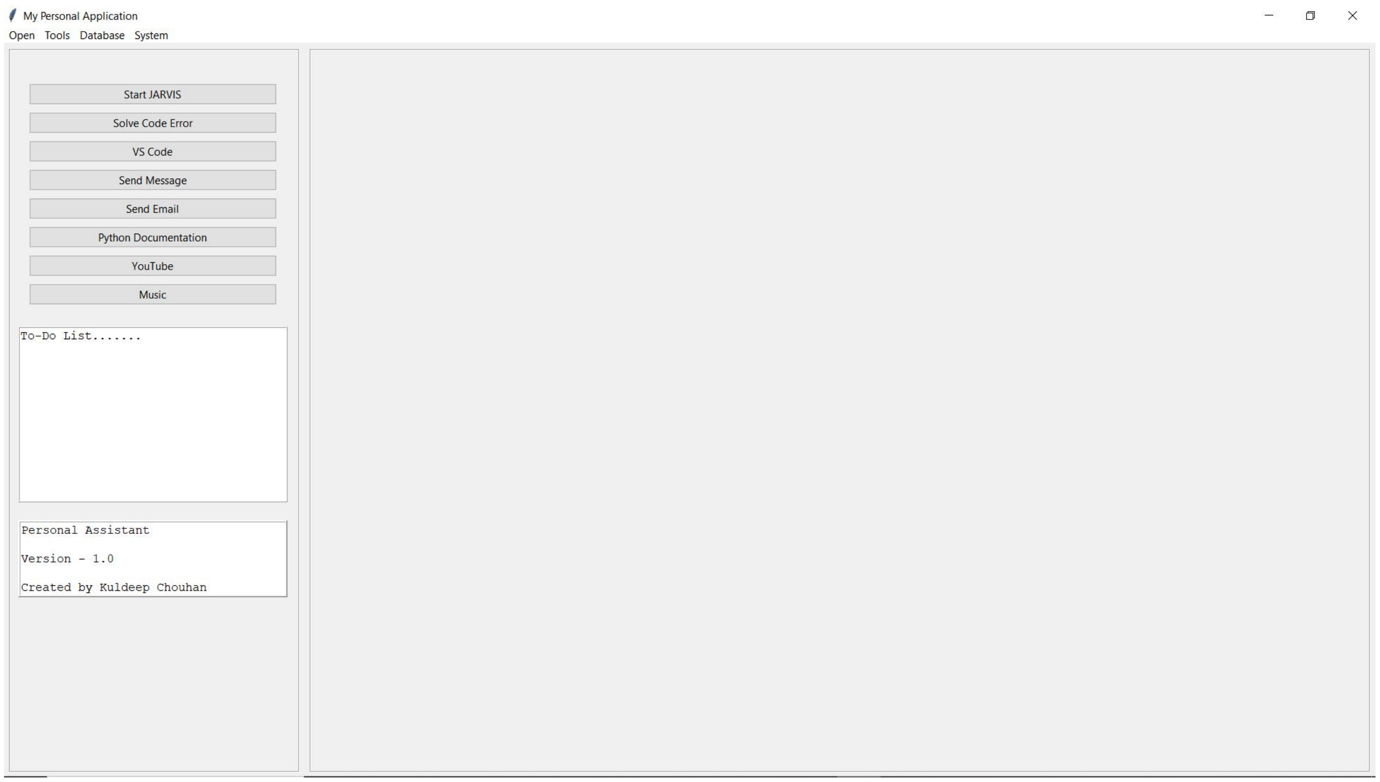
5.

Implementation

5.1- Screen Shots :-

Workspace Pro - My Command Center

Main GUI Window



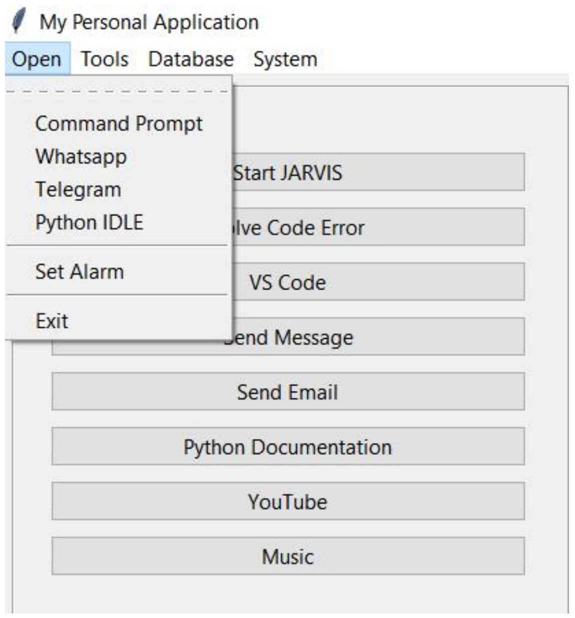
To-Do List.......

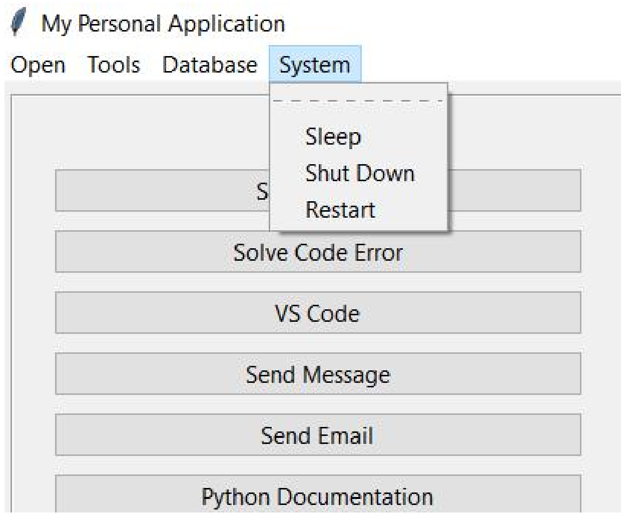
Personal Assistant

Version - 1.0

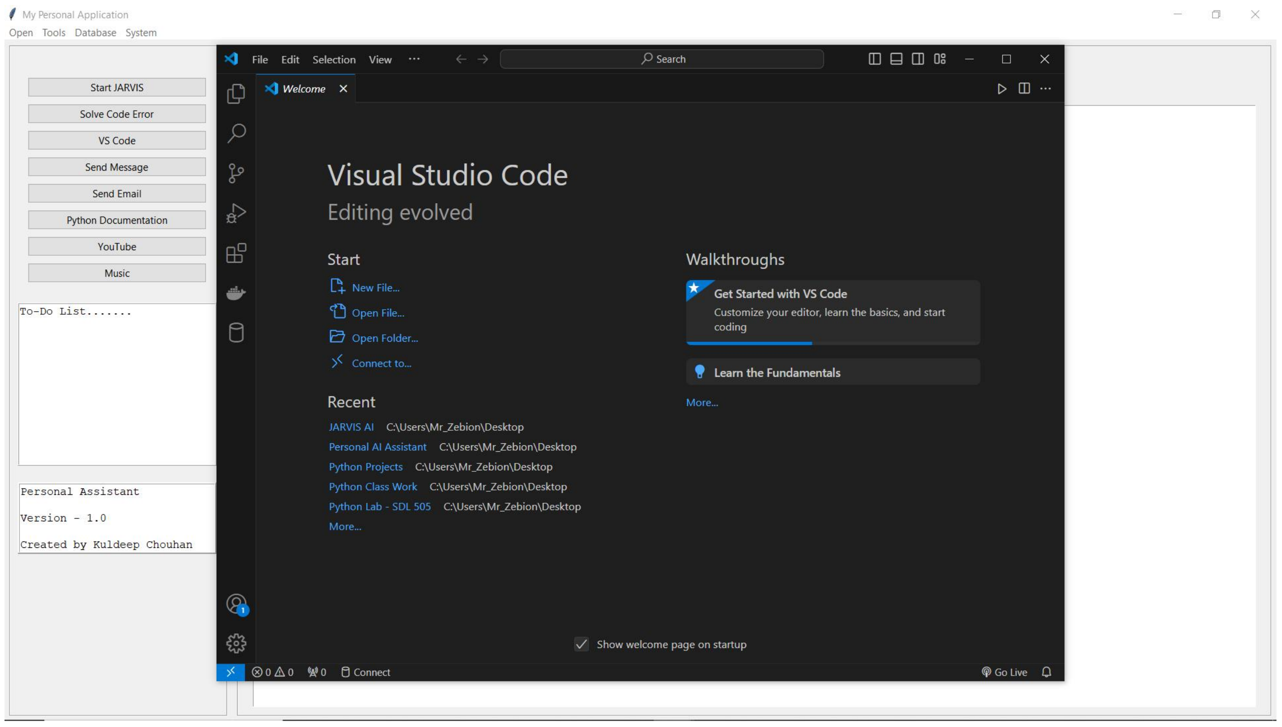
Created by Kuldeep Chouhan

Menu Bar





Opening Applications

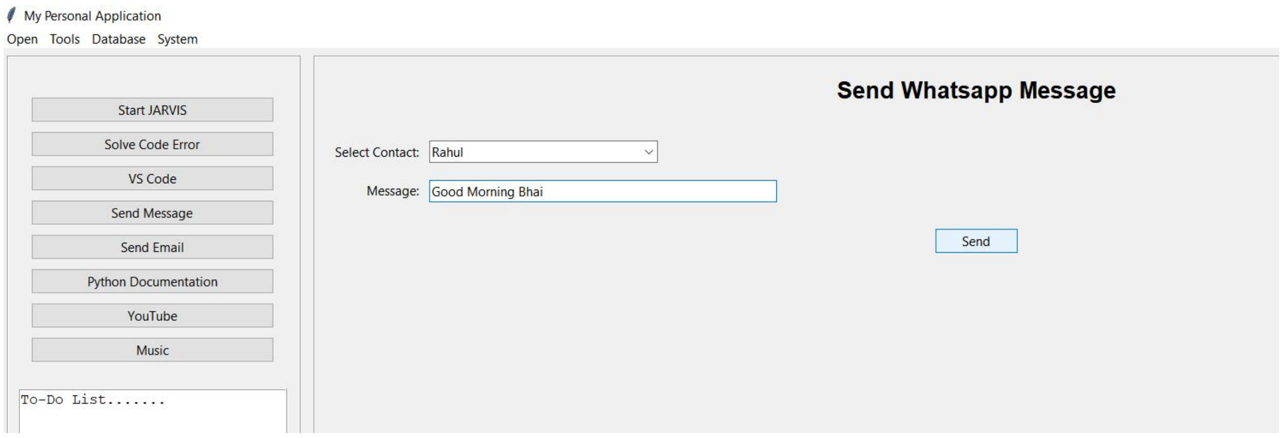


```
To-Do List.......
```

```
Personal Assistant
Version - 1.0
Created by Kuldeep Chouhan
```

JARVIS AI C:\Users\Mr_Zebion\Desktop
Personal AI Assistant C:\Users\Mr_Zebion\Desktop
Python Projects C:\Users\Mr_Zebion\Desktop
Python Class Work C:\Users\Mr_Zebion\Desktop
Python Lab - SDL 505 C:\Users\Mr_Zebion\Desktop

Button Functions: Send Message

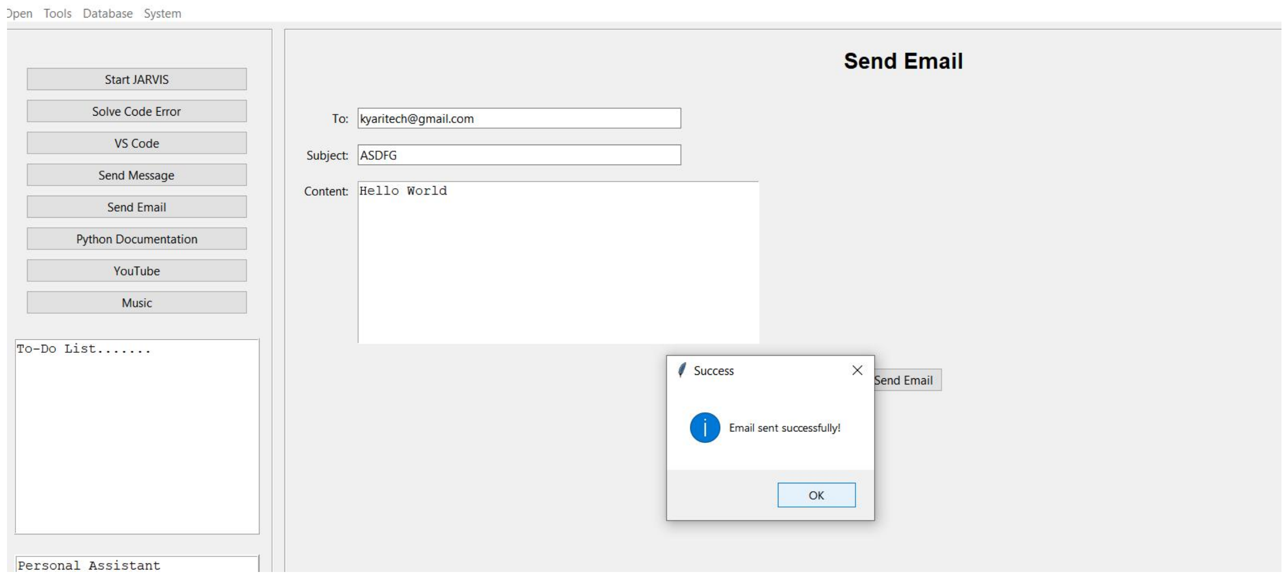


Select Contact: Rahul

Message: Good Morning Bhai

To-Do List.......

Send Email Function



To: kyaritech@gmail.com

Subject: ASDFG

Content: Hello World

Email sent successfully!

To-Do List.......

Personal Assistant

6.

Software Testing

Software testing ensures that the application works as expected and meets the requirements. For this project, testing focused on the main features, such as opening applications, sending messages, sending emails, and playing music. The testing process was done manually, and the results confirmed the system's functionality and reliability.

Testing Process

The testing process included the following steps:

Functional Testing
Each button and feature was tested individually to check if it performs its assigned task. For example:

1. Clicking the "VS Code" button correctly opened Visual Studio Code.
2. Clicking the "Python Documentation" button displayed the relevant documentation.

Integration Testing
The interaction between different components of the application, like the GUI and background functions (e.g., sending emails), was tested to ensure smooth operation.

Boundary Testing
The application was tested for invalid or extreme inputs, such as:

1. Incorrect email addresses in the send email feature.
2. Empty or invalid commands in the voice recognition system.

Test Cases

Some key test cases:

Opening Applications:

- Action: Click the button for a specific app.
- Expected Result: The selected app opens.
- Actual Result: Passed (e.g., VS Code, YouTube, Python IDLE opened correctly).

Sending Emails:

- Action: Input email content and recipient address, then send.
- Expected Result: Email sent successfully.
- Actual Result: Passed (emails were sent without errors).

Playing Music:

- Action: Click the "Music" button.
- Expected Result: A random song plays from the designated folder.
- Actual Result: Passed (songs played as expected).

Testing Outcome

The application passed all the functional and integration tests. It performs its tasks accurately and responds correctly to my commands. Testing confirmed that the system is reliable and ready for use. Future improvements will focus on refining the features based on personal needs.

7. Conclusion

This project successfully achieved its goal of creating a personalized GUI desktop application to improve productivity and minimize distractions. The application provides an easy-to-use interface with features like opening important applications, sending WhatsApp messages, and emailing. These functionalities help in managing tasks quickly and efficiently.

By integrating these tools into one system, the project addresses the issue of time wastage caused by distractions like movies and games. The application ensures that the user can focus on productive activities immediately after starting the laptop.

The testing phase confirmed that the system is functional, reliable, and meets all the requirements. While the project is tailored for personal use, it has the potential for further customization and enhancements based on future needs.

In conclusion, this project serves as a practical solution for boosting productivity and maintaining focus during work sessions. It is a step towards better time management and efficient use of technology.

8.

Future Enhancements

While the current application is functional and serves its purpose effectively, there are areas for improvement and new features that can be added in the future to make it more robust and user-friendly.

1. Fixing Crashes

Sometimes, the program crashes unexpectedly. Debugging and optimizing the code will help resolve this issue, ensuring the application runs smoothly without interruptions.

2. Improving Voice Recognition

Currently, JARVIS sometimes fails to understand user commands. Enhancing the speech recognition feature by integrating advanced tools or refining the logic will make the assistant more reliable and accurate.

3. Adding Login/Logout Functionality

To enhance security, a login/logout system will be integrated into the application. This will ensure that only the authorized user (me) can access the app and its features.

4. Enhancing JARVIS Voice Assistant Performance

At times, the JARVIS voice assistant responds slowly. Improving the performance by optimizing the backend processes and integrating faster text-to-speech and response systems will make it more efficient and responsive.

5. Improving GUI Responsiveness

Some buttons in the GUI do not respond properly when clicked. The interface will be made more responsive and user-friendly by fixing these bugs and testing the layout on different devices and screen resolutions.

These enhancements aim to make the application more secure, reliable, and efficient. By addressing these areas, the project will evolve into an even better productivity tool tailored to personal needs.

“Appendix”

A. Bibliography

Websites and Online Articles:

1. "Tkinter GUI Programming" - GeeksforGeeks (https://www.geeksforgeeks.org/)
2. "Python Database Access with MySQL" - W3Schools (https://www.w3schools.com/)
3. "Introduction to Speech Recognition" - Real Python (https://realpython.com/)

Video Tutorials:

1. "JARVIS AI by Code With Harry " - YouTube (https://youtube.com/)
2. "Tkinter GUI Part 1 & 2" - Corey Schafer (https://youtube.com/)

Online Documentation:

1. Python Official Documentation (https://docs.python.org/)
2. Pyttsx3 Documentation (https://pyttsx3.readthedocs.io/)

Software Tools and Libraries:

1. Python 3.10 (https://www.python.org/)
2. MySQL Connector for Python (https://dev.mysql.com/doc/connector-python/)
3. SpeechRecognition Library (https://pypi.org/project/SpeechRecognition/)

B. Project Daily Report

Day 1 - October 1, 2024
Started the project by planning the overall structure of the application. Set up the development environment for Python and installed necessary libraries. Began working on the basic GUI layout using Tkinter.

Day 2 - October 2, 2024
Designed the sidebar with buttons for opening applications. Integrated the "VS Code" and "YouTube" buttons. Tested the basic layout and fixed any issues related to button alignment.

Day 3 - October 3, 2024
Worked on the "Music" feature, allowing the application to play songs from a specific folder. Added functionality to open and play random songs. Tested the music feature to ensure it works correctly.

Day 4 - October 4, 2024
Created the "Send Email" feature by integrating Python’s smtplib. Wrote the function to send emails through a Gmail account. Tested the email feature with a test message and verified it sent successfully.

Day 5 - October 5, 2024
Developed the voice recognition feature using the speech_recognition library. Integrated it with the JARVIS assistant to listen to commands. Began testing voice recognition for basic commands.

Day 6 - October 6, 2024
Refined the voice recognition functionality to better understand basic commands. Added more commands such as "open Google" and "tell time". Continued testing to ensure smoother interaction.

Day 7 - October 7, 2024
Added the feature to open the Python IDLE using a button. Worked on integrating the "Play Music" button with a more dynamic music selection. Fixed some minor bugs with button actions.

Day 8 - October 8, 2024
Started working on integrating JARVIS to respond with voice feedback. Added text-to-speech functionality with pyttsx3. Tested basic voice responses to commands.

Day 9 - October 9, 2024
Worked on integrating the "Send WhatsApp Message" feature using the pywhatkit library. Tested sending a message to a test contact. Fixed issues with message delays.

Day 10 - October 10, 2024
Developed the functionality to open the camera. Integrated OpenCV to start the webcam. Fixed bugs related to camera feed display.

Day 11 - October 11, 2024
Refined the GUI layout by adjusting the frames for better user experience. Improved button responsiveness. Started working on a more responsive interface for different screen sizes.

Day 12 - October 12, 2024
Worked on integrating a system to check and display the user's IP address. Added the "IP Address" feature to JARVIS. Tested and confirmed it correctly displays the IP address.

Day 13 - October 13, 2024
Created the functionality to open the Command Prompt. Added it to the GUI and tested the button to ensure it opens the command prompt without any issues.

Day 14 - October 14, 2024
Integrated more advanced commands into JARVIS. Added commands for opening various applications like Telegram and File Explorer. Started testing for voice command accuracy.

Day 15 - October 15, 2024
Worked on the "To-Do List" feature, allowing users to add and remove tasks. Integrated a simple task management system. Tested the list functionality.

Day 16 - October 16, 2024
Enhanced the "To-Do List" functionality by saving the tasks to a database. Started working on the database integration for saving user tasks. Fixed issues related to saving and retrieving tasks.

Day 17 - October 17, 2024
Continued working on the database for storing tasks. Added the ability to edit and delete tasks. Fixed some database connection issues.

Day 18 - October 18, 2024
Tested the entire "To-Do List" feature. Checked the interaction between the GUI, buttons, and database. Fixed minor bugs in task management features.

Day 19 - October 19, 2024
Worked on refining the email functionality. Integrated automatic email replies and fixed bugs related to sending attachments. Tested sending different types of emails.

Day 20 - October 20, 2024
Improved the overall voice recognition accuracy. Worked on handling background noise and improving the system's ability to recognize commands. Continued testing voice commands.

Day 21 - October 21, 2024
Worked on implementing login/logout functionality. Started writing the code for secure user login and logout. Tested basic login functionality.

Day 22 - October 22, 2024
Refined the login/logout system to ensure that only authorized users can access the application. Added a logout button to the GUI and tested it.

Day 23 - October 23, 2024
Fixed bugs in the login/logout system. Added security features to prevent unauthorized access. Conducted multiple tests to ensure the system works securely.

Day 24 - October 24, 2024
Continued working on the JARVIS assistant. Optimized the text-to-speech functionality for faster responses. Reduced delay in voice feedback.

Day 25 - October 25, 2024
Improved voice command accuracy and response speed. Worked on integrating more complex commands for JARVIS. Fixed bugs related to voice recognition failures.

Day 26 - October 26, 2024
Added more features to JARVIS, such as searching Wikipedia and opening Google. Continued testing and optimizing for better voice recognition.

Day 27 - October 27, 2024
Refined the functionality to send messages via WhatsApp. Fixed issues with pywhatkit and ensured the messages send without delay.

Day 28 - October 28, 2024
Started improving the GUI responsiveness. Fixed issues with button click events and ensured that the frames respond better to clicks.

Day 29 - October 29, 2024
Tested the complete application with all features enabled. Fixed minor bugs and ensured that every button works correctly.

Day 30 - October 30, 2024
Worked on improving the overall system performance. Optimized the backend code to make the application run smoother. Tested all features together for reliability.

Day 31 - October 31, 2024
Refined the JARVIS voice assistant. Fixed issues where it didn’t respond to some commands. Enhanced overall interaction.

Day 32 - November 1, 2024
Focused on testing email features, ensuring it can handle multiple recipients and attachments. Fixed minor issues related to email format.

Day 33 - November 2, 2024
Enhanced the music player functionality. Integrated a better random song selection system. Tested and confirmed it worked smoothly.

Day 34 - November 3, 2024
Fixed GUI responsiveness issues. Improved button layout to ensure that all buttons are clickable and functional.

Day 35 - November 4, 2024
Tested the system for crashes and fixed some bugs that caused the program to close unexpectedly. Increased the stability of the application.

Day 36 - November 5, 2024
Worked on improving voice recognition. Integrated better noise filtering to make JARVIS more accurate in noisy environments.

Day 37 - November 6, 2024
Integrated a database system for storing user settings and preferences. Tested saving and loading user preferences successfully.

Day 38 - November 7, 2024
Tested the application on different devices and screen sizes. Improved layout to make the application responsive to all screen resolutions.

Day 39 - November 8, 2024
Started adding advanced features like automated reminders and notifications. Improved the efficiency of these tasks.

Day 40 - November 9, 2024
Finalized the application for personal use. Conducted extensive tests to ensure that all features work together as expected.