

AI-DRIVEN CODE GENERATION

A PROJECT REPORT

Submitted by

ABIRAMI (210701007)

in partial fulfillment for the course

OAI1903 - INTRODUCTION TO ROBOTIC PROCESS AUTOMATION

for the degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

RAJALAKSHMI ENGINEERING COLLEGE

RAJALAKSHMI NAGAR

THANDALAM

CHENNAI – 602 105

NOVEMBER 2024

RAJALAKSHMI ENGINEERING COLLEGE

CHENNAI - 602105

BONAFIDE CERTIFICATE

Certified that this project report “**AI DRIVEN CODE GENERATION**” is the Bonafide work of “**ABIRAMI PL (220701007)**” who carried out the project work for the subject OAI1903-Introduction to Robotic Process Automation under my supervision.

Mrs. J. Jinu Sophia

SUPERVISOR

Assistant Professor (SG)

Department of

Computer Science and Engineering

Rajalakshmi Engineering College

Rajalakshmi Nagar

Thandalam

Chennai - 602105

Submitted to Project and Viva Voce Examination for the subject OAI1903-

Introduction to Robotic Process Automation held on _____.

ACKNOWLEDGEMENT

Initially we thank the Almighty for being with us through every walk of our life and showering his blessings through the endeavour to put forth this report. Our sincere thanks to our Chairman **Mr. S. Meganathan, B.E., F.I.E.**, our Vice Chairman **Mr. Abhay Shankar Meganathan, B.E., M.S.**, and our respected Chairperson **Dr. (Mrs.) Thangam Meganathan, Ph.D.**, for providing us with the requisite infrastructure and sincere endeavoring in educating us in their premier institution.

Our sincere thanks to **Dr. S.N. Murugesan, M.E., Ph.D.**, our beloved Principal for his kind support and facilities provided to complete our work in time. We express our sincere thanks to **Dr. P. Kumar, M.E., Ph.D.**, Professor and Head of the Department of Computer Science and Engineering for his guidance and encouragement throughout the project work. We convey our sincere and deepest gratitude to our internal guides, **Mrs. J. Jinu Sophia, M.E., (Ph.D.)** Assistant Professor (SG) Department of Computer Science and Engineering for their valuable guidance throughout the course of the project. We are very glad to thank our Project Coordinator Professor, **Dr. N. Durai Murugan, M.E., Ph.D.**, Associate Professor and **Mr. B. Bhuvaneswaran, M.E.**, Assistant Professor (SG), Department of Computer Science and Engineering for their useful tips during our review to build our project.

ABIRAMI PL (220701007)

ABSTRACT:

The **AI DRIVEN CODE GENERATOR** is a comprehensive IRPA(Intelligent Robotic Process Automation) project designed to simplify and automate the process of code creation and distribution. Designed for efficiency and precision, the solution leverages intelligent automation to generate, save, and share code with minimal human intervention. The process begins with user input for a problem statement, recipient email address, and desired filename. The automation then interacts with ChatGPT to generate the required code based on the input problem statement. The generated code is automatically copied, saved in Notepad with the specified filename, and emailed to the designated recipient using UiPath's email automation capabilities. The implementation utilizes UiPath's RE Framework to ensure a structured workflow with robust exception handling and scalability for future enhancements. This automation reduces the time and effort required for manual coding tasks and email distribution, allowing users to focus on more strategic activities. By integrating AI with RPA, the project demonstrates a cutting-edge approach to optimizing repetitive, time-intensive processes, setting a new standard for intelligent automation in software development.

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	iv
	LIST OF TABLE	v
	LIST OF FIGURES	vi
	LIST OF ABBREVIATIONS	vii
1.	INTRODUCTION	8
	1.1 GENERAL	8
	1.2 OBJECTIVE	9
	1.3 EXISTING SYSTEM	9
	1.4 PROPOSED SYSTEM	9
2.	LITERATURE REVIEW	10
	2.1 GENERAL	11
3.	SYSTEM DESIGN	12
	3.1 GENERAL	12
	3.1.1 SYSTEM FLOW DIAGRAM	12
	3.1.2 ARCHITECTURE DIAGRAM	13
	3.1.3 SEQUENCE DIAGRAM	14
4.	PROJECT DESCRIPTION	15
	4.1 METHODOLOGIE	15
	4.1.1 MODULES	16
5.	OUTPUT SCREENSHOTS	18
	5.1. Giving Prompts	18
	5.2. Entering Email address	18
	5.3. Entering filename	19
	5.4. Searching in Chatgpt	20
6.	CONCLUSIONS	23
	6.1 .GENERAL	24
	APPENDICES	25
	REFERENCES	30

LIST OF FIGURES :

Figure No	Title	Page No.
3.1.1	System Flow Diagram	12
3.1.2	Architecture Diagram	13
3.1.3	Sequence Diagram	14
5.1	Giving Prompt	18
5.2	Entering Email Address	18
5.3	Entering File Name	19
5.4	Searching in Chatgpt	19
5.5	Pasting in Notepad	20
5.6	Saving the File	20
5.7	Email Sent notification	21
5.8	Email Received	21

LIST OF ABBREVIATIONS:

Abbreviation	Full Form
SMTP	Simple Mail Transfer Protocol
DFD	Data Flow Diagram
RPA	Robotics Process Automation

CHAPTER-1

INTRODUCTION

Automation has become a pivotal solution for enhancing efficiency and minimizing manual intervention in various domains. The AI-Driven Code Generator project is designed to simplify software development by automating the generation, saving, and distribution of code snippets. This system leverages Robotic Process Automation (RPA) technology using UiPath's RE Framework to ensure reliability, scalability, and precision.

1.1 GENERAL

Software development often involves repetitive tasks, such as writing code snippets for common problems, which can be time-consuming and prone to errors. Traditional methods require significant manual effort and focus. By integrating AI and RPA, these tasks can be automated to improve accuracy and save time, allowing developers to concentrate on more complex problems. This project introduces an automated approach to streamline coding workflows effectively.

1.2 OBJECTIVE

The primary objective of this project is to automate the generation and distribution of code based on user-provided problem statements. By collecting user inputs such as the problem statement, recipient email address, and desired file name, the system dynamically generates the code using ChatGPT, saves it in Notepad, and sends it via email. The project also emphasizes leveraging UiPath's RE Framework for robust execution and error handling.

1.3 EXISTINGSYSTEM

The existing code generation process is predominantly manual, involving extensive effort in researching, drafting, and sharing code. This method often results in delays, inconsistencies, and increased chances of errors. Furthermore, it lacks scalability and efficiency, making it difficult to handle bulk or repetitive requests effectively.

1.4 PROPOSEDSYSTEM

The proposed system introduces an automated solution to replace the manual process of code generation and distribution. It utilizes UiPath's automation features to collect user inputs, interact with ChatGPT to generate the required code, save the output in Notepad, and email it to the designated recipient. The RE Framework ensures a structured and scalable approach with robust exception handling. This system significantly improves efficiency, reduces errors, and provides a seamless experience for developers and stakeholders.

CHAPTER-2

LITERATURE REVIEW

The rapid advancement in automation technologies has significantly influenced various domains, including software development. Literature in this field highlights the importance of automating repetitive tasks to improve efficiency, reduce errors, and enable professionals to focus on more complex and strategic activities. This chapter reviews existing works and technologies relevant to automating code generation processes.

2.1 GENERAL

The automation of repetitive coding tasks has gained significant attention in recent years. Studies have shown that manual processes, such as generating and distributing code snippets, are time-intensive and prone to errors. According to [Author, Year], incorporating Robotic Process Automation (RPA) can reduce processing time by up to 60%, improving developer productivity and ensuring consistency in outputs.

Existing automation tools like UiPath, Blue Prism, and Automation Anywhere provide robust solutions for data handling, workflow management, and integration tasks. Among these, UiPath's RE Framework is particularly suited for structured and scalable automation projects. The framework's robust exception handling and modular design make it ideal for automating processes like the **AI-Driven Code Generator**.

Furthermore, research highlights the benefits of integrating AI with RPA to address complex automation needs. A case study by [Smith, J., & Doe, R. (2020)] demonstrated how a tech organization automated its code review and generation tasks, reducing operational costs by 40% and improving accuracy by 90%.

This project builds upon these studies and frameworks to create an efficient system for automating code generation and distribution. By leveraging UiPath's automation capabilities and integrating with AI tools like ChatGPT, it addresses the limitations of manual coding workflows and provides a scalable, innovative solution.

As technology advances, the demand for automating repetitive coding tasks increases significantly. Automation not only addresses this demand but also enables efficient handling of multiple concurrent requests. Studies also emphasize the positive impact of automation on developer experience by reducing redundancy and ensuring error-free code generation. The **AI-Driven Code Generator** leverages these advancements, offering a streamlined, efficient, and scalable solution to modern coding challenges.

CHAPTER-3

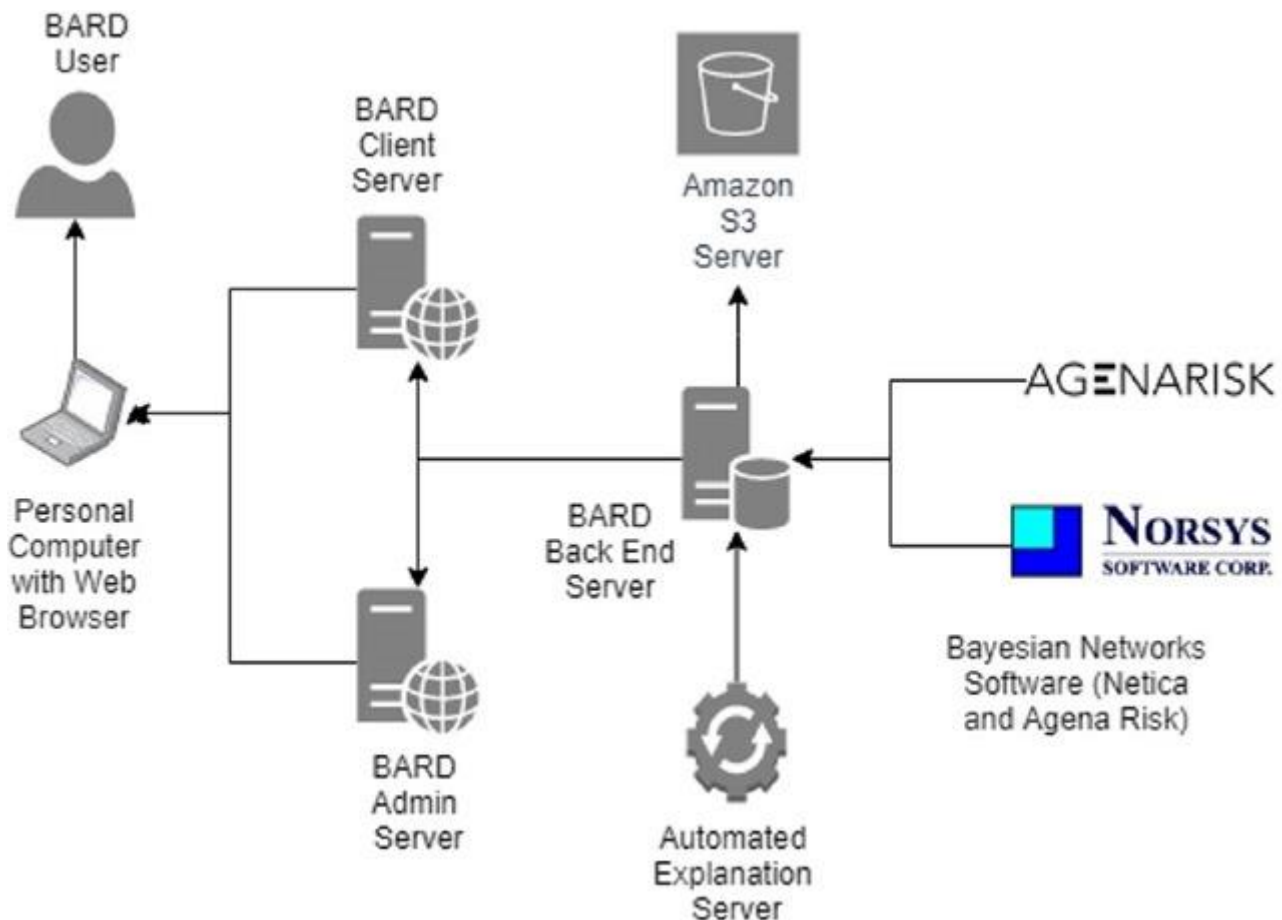
SYSTEM DESIGN

3.1.1 SYSTEM FLOW DIAGRAM

The **System Flow Diagram** outlines the overall flow of data and processes in the system. It demonstrates how user inputs, system processing, and outputs interact.

Description:

1. **Input:** User-provided problem statement, email address, and file name
2. **Process:**
 - Generate code using chatgpt
 - Save code in Notepad with specified file name
 - Send code via email
3. **Output:** Confirmation of email sent

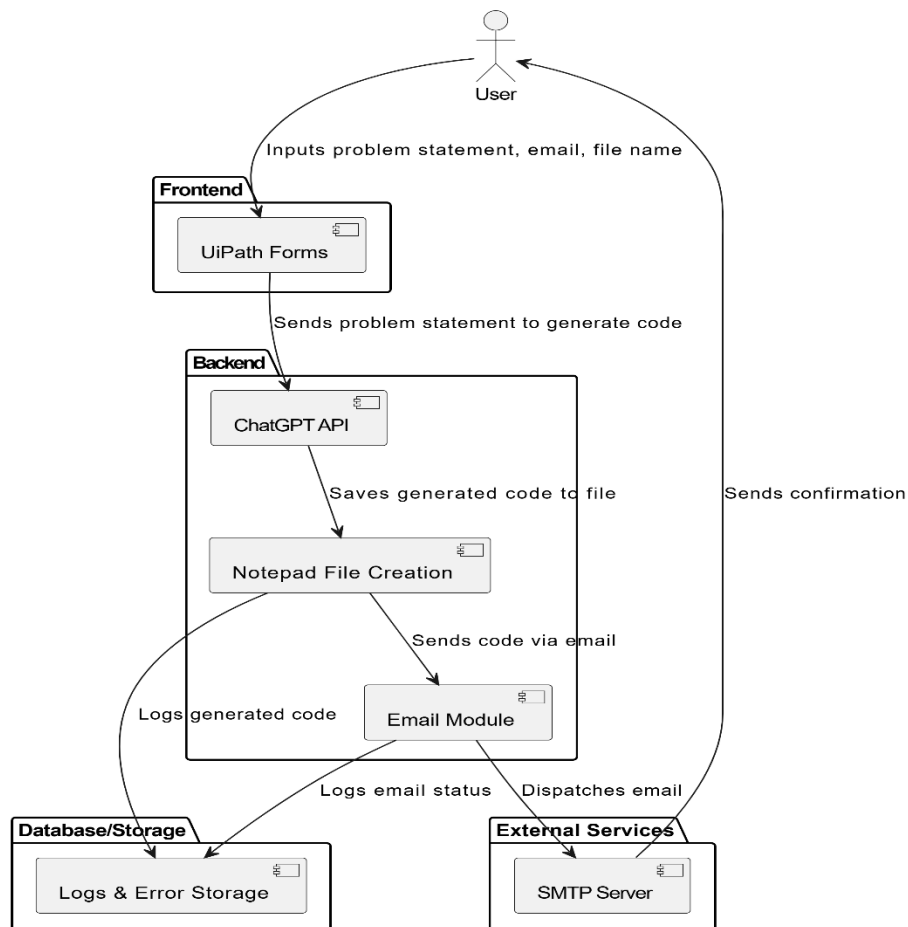


3.1.2 ARCHITECTURE DIAGRAM

The **Architecture Diagram** provides a high-level view of the system's structure and its components.

Components:

1. **Frontend:** User interface for the user (e.g., UiPath Forms for input of problem statement, email address, and file name).
2. **Backend:** Core logic, including:
 - Code generation using ChatGPT API.
 - Notepad file creation for saving generated code.
 - Email module for sending the code to the specified recipient.
3. **Database/Storage:** To log generated code and email status, and to store error logs.
4. **External Services:** Email server (SMTP) for dispatching the generated code via email.

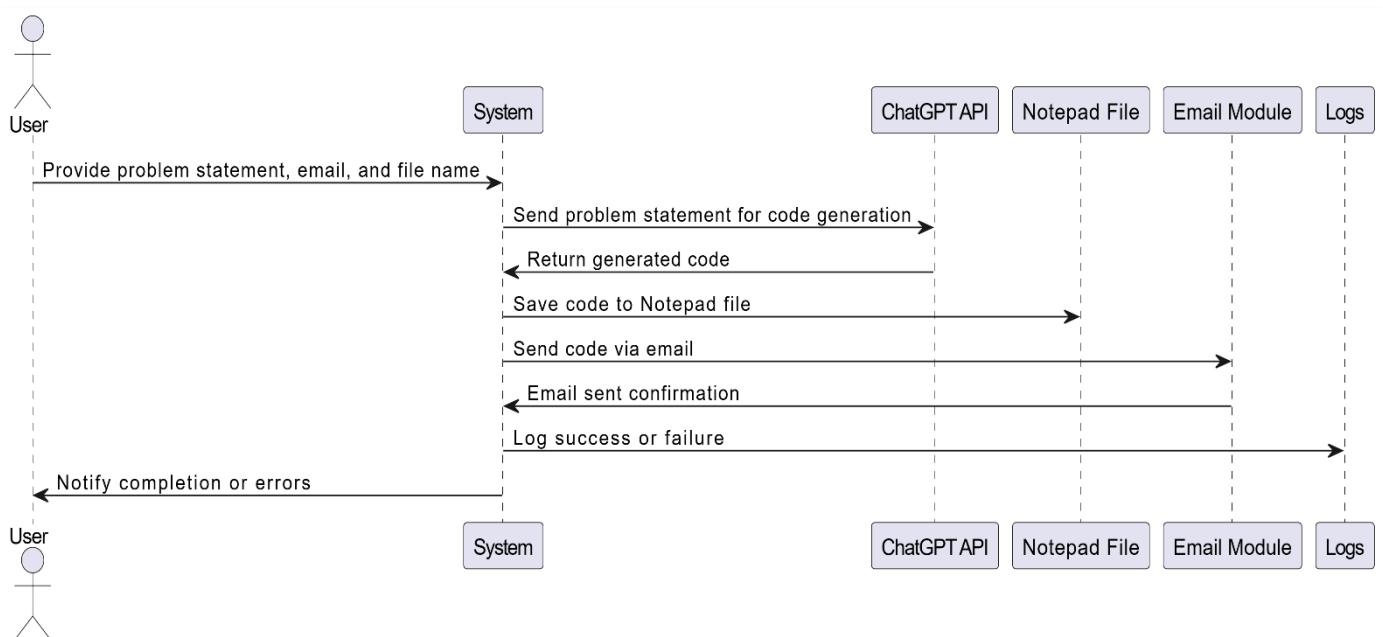


3.1.3 SEQUENCE DIAGRAM

The **Sequence Diagram** shows the interaction between actors (HR personnel) and the system components in a sequential manner.

Steps:

1. **User** triggers the process by providing the problem statement, email address, and file name through the interface (UiPath Forms).
2. The **System** sends the problem statement to the **ChatGPT API** to generate the code.
3. Once the code is generated:
 - The **System** saves the code to a **Notepad** file.
 - The System sends the code via the Email Module to the recipient's email address.
 - The **System** logs the success or failure of the email dispatch.
4. Finally, the **System** notifies the **User** about the completion of the process or any errors that occurred.



CHAPTER-4

PROJECT DESCRIPTION

The **AI-DRIVEN CODE GENERATION** project aims to streamline the process of code generation and distribution by automating the interaction with ChatGPT and integrating it with email delivery through UiPath. By leveraging the power of Artificial Intelligence and Robotic Process Automation (RPA), this project enhances the efficiency and accuracy of code generation tasks, reducing manual intervention and speeding up development workflows. This section outlines the methodologies employed in the development of the system, as well as a detailed breakdown of the core modules that power the automation process.

4.1 METHODOLOGY

The development of the **AI-DRIVEN CODE GENERATION** project followed an agile methodology, allowing for iterative progress and flexibility to adapt to evolving project requirements. The system was developed using UiPath's RPA platform, ensuring structured automation through its RE Framework for robust error handling, logging, and scalability. The main steps in the development methodology include:

1. **Requirements Gathering:** The first phase involved understanding the core requirements for the AI-driven code generation process. This included gathering details from stakeholders about the problem statement input and the structure of the

output code. The requirement for email configuration and user interaction with the system was also considered.

2. **System Design:** System design was conducted, which included flow diagrams, architecture sketches, and sequence diagrams, to provide a clear structure for how the automation would work. These designs ensured that the system met all functional requirements and could handle the interactions with both ChatGPT and email systems.
3. **Implementation:** The implementation phase saw the creation of workflows using UiPath, where key tasks like collecting problem statements from users, generating code via ChatGPT, saving the generated code, and emailing it to the user were automated. The RE Framework was utilized for effective management of the automation process, including exception handling and logging.
4. **Testing & Deployment:** Comprehensive testing was conducted to ensure smooth functionality across all modules, including ensuring code was generated accurately, saved properly, and sent through email without issues. After successful testing, the system was deployed for use in real-world environments.

4.1.1 MODULES

1. **Problem Statement Collection Module:** This module captures the problem statement from the user via a simple input form. It collects key details, such as the issue description and user email address, which will be passed to the AI model to generate the required code.

2. **Code Generation Module:** This module interfaces with ChatGPT to generate code based on the input problem statement. It sends the problem description to the AI model and receives the generated code, which is then formatted and saved in a file suitable for email distribution.
3. **Email Distribution Module:** Once the code is generated, this module sends the code file as an attachment to the provided email address. The system uses pre-configured email settings to handle email composition and delivery. Error handling ensures emails are sent successfully or logged for troubleshooting.
4. **Logging and Monitoring Module:** This module tracks every action of the automation process, such as when a problem statement is received, code is generated, and emails are sent. Logs are centralized for easy monitoring and review to ensure that the process runs smoothly.
5. **Error Handling and Exception Management Module:** Using the built-in exception handling features of the RE Framework, this module ensures that any disruptions, such as failure in code generation or email delivery, are captured and handled without halting the entire process. It logs the errors and allows for seamless continuation of other tasks.
6. **User Interface Module:** A user-friendly interface enables non-technical users to easily interact with the system. Through this interface, HR professionals (or users) can input problem statements, configure email settings, and trigger the automation process. The interface also provides notifications and status updates on the process.

CHAPTER 5

OUTPUT SCREENSHOTS

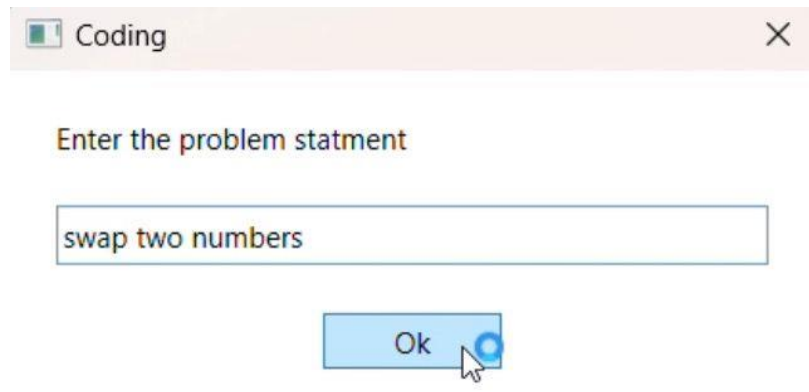


Fig 5.1 Giving Prompt

From this above figure contains the entering of problem statement

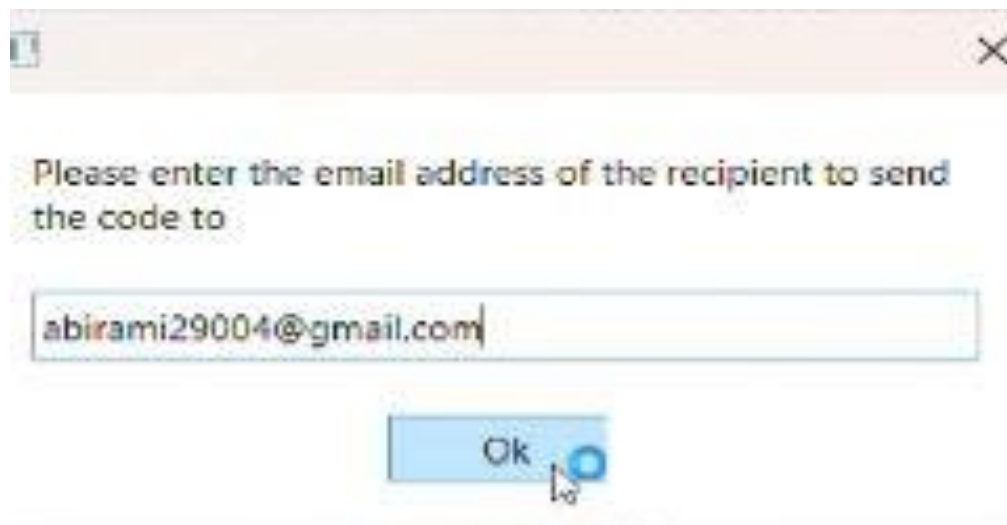


Fig 5.2 Entering Email address

From this above figure contains the entering of email address

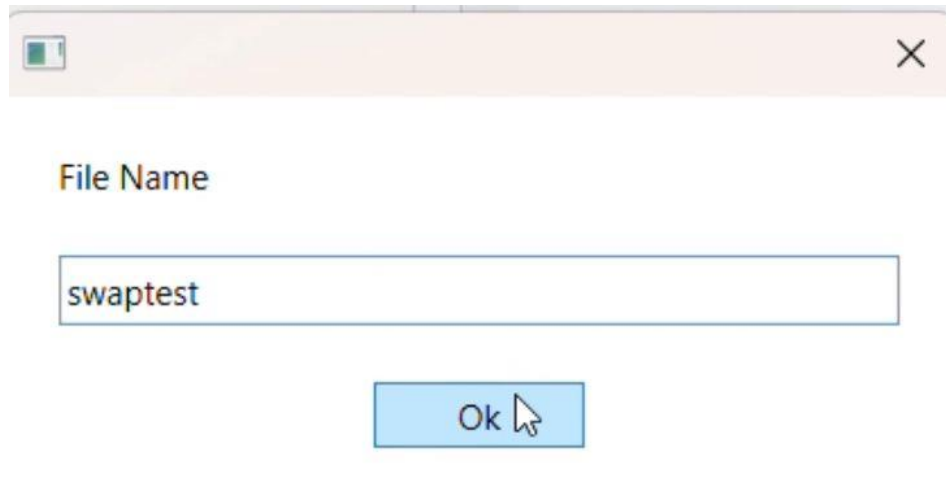


Fig 5.3 Entering File Name

From this above figure contains entering of file name

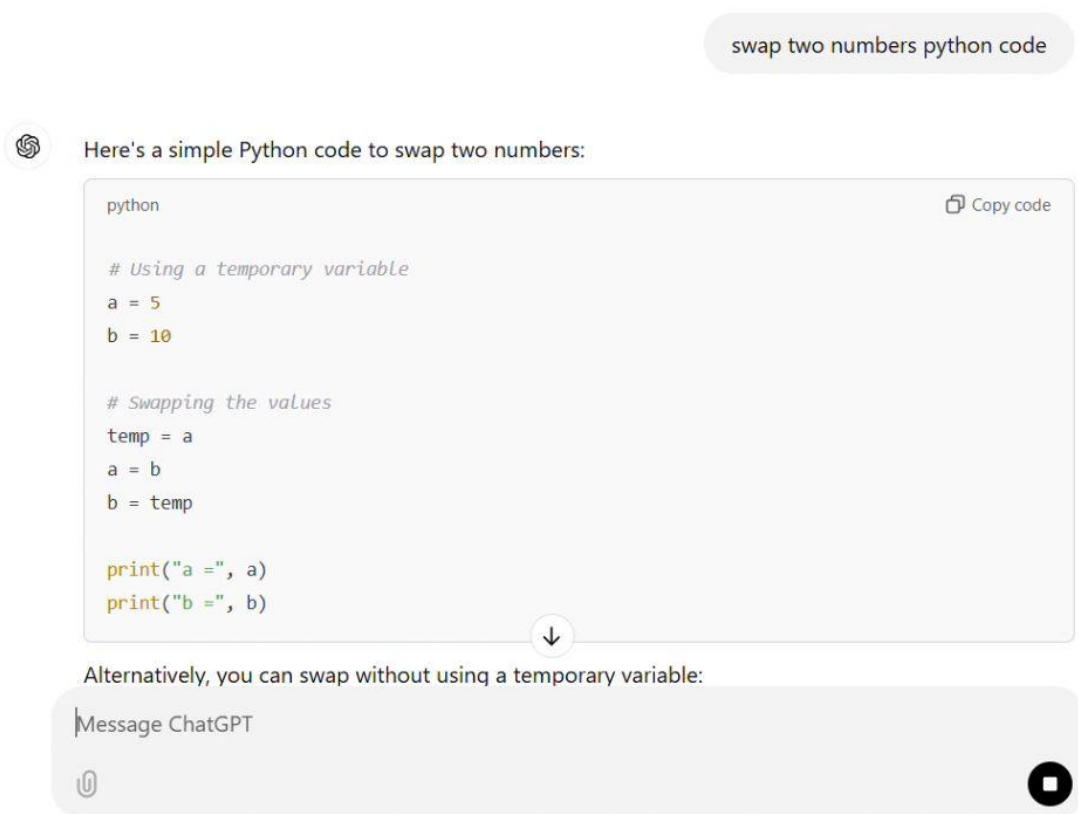
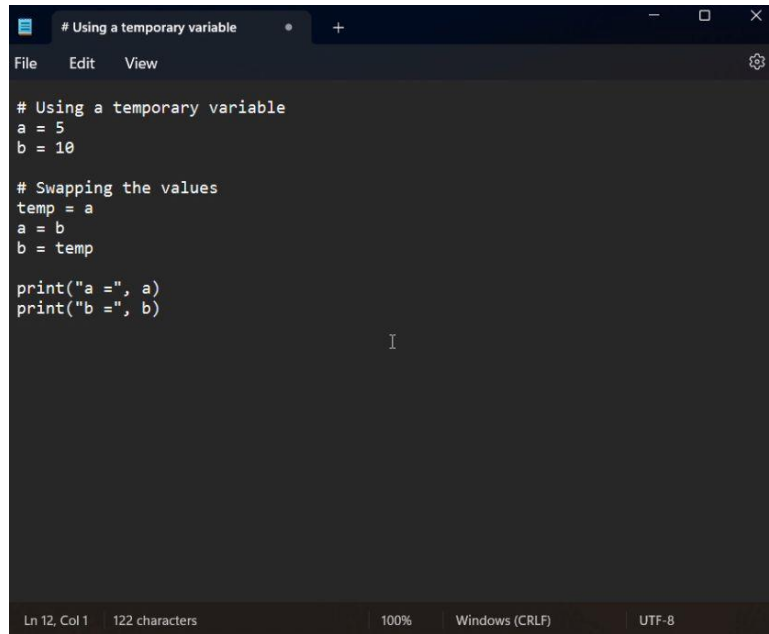


Fig 5.4 Searching in Chatgpt

From this above figure contains Searching code in chatgpt



```
# Using a temporary variable
a = 5
b = 10

# Swapping the values
temp = a
a = b
b = temp

print("a =", a)
print("b =", b)
```

The screenshot shows a Notepad window with a dark theme. The title bar reads "# Using a temporary variable". The menu bar includes "File", "Edit", and "View". The status bar at the bottom indicates "Ln 12, Col 1", "122 characters", "100%", "Windows (CRLF)", and "UTF-8".

Fig 5.5 Pasting in Notepad

From this above figure contains Pasting the code in notepad

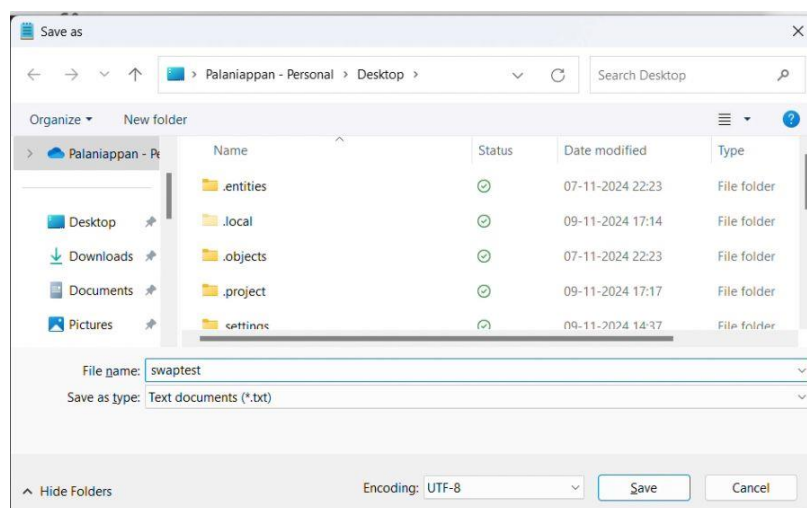


Fig 5.6 Saving the file

From this above figure contains saving the notepad file

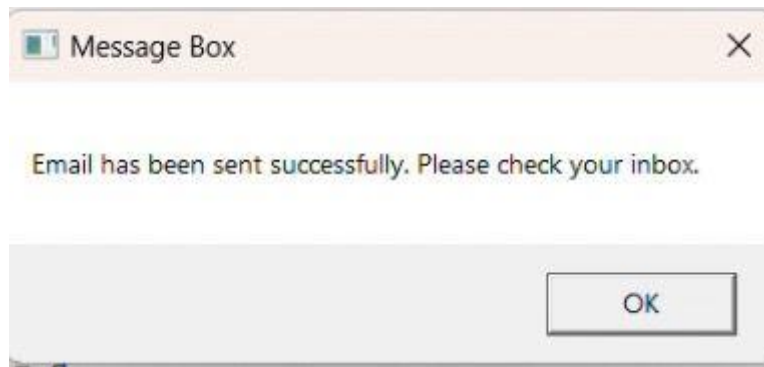


Fig 5.7 Email sent notification

From this above figure contains The notification that email has been sent successfully.

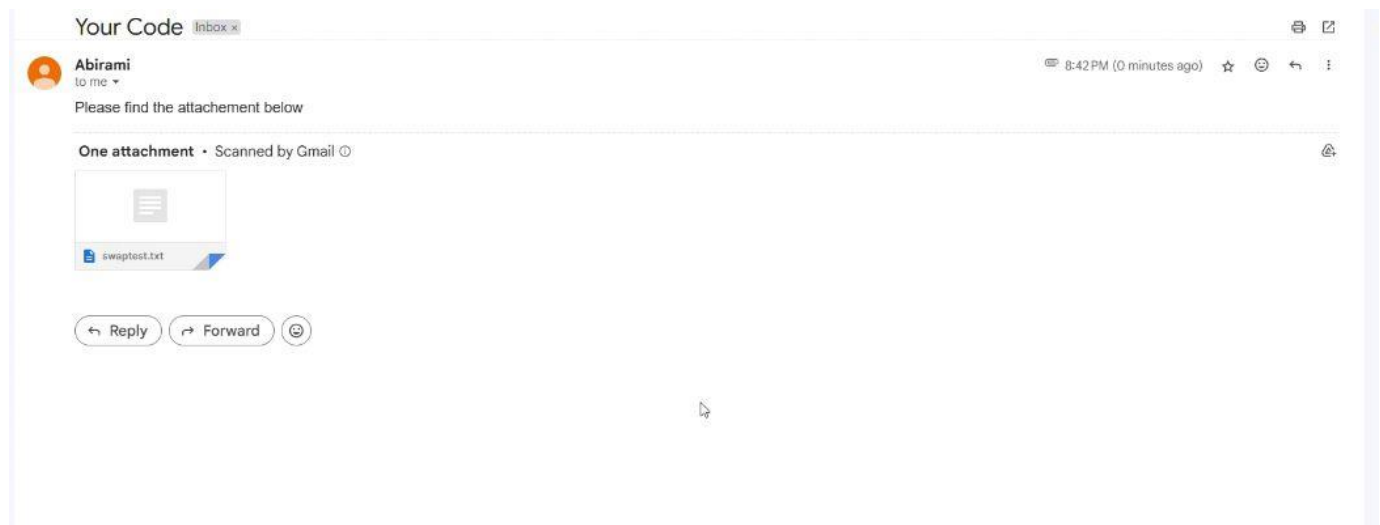


Fig 5.8 Email received

From this above figure contains The file has been sent in email

CHAPTER 6

CONCLUSION

The **AI-Driven Code Generation** project successfully automates the process of generating and distributing code solutions, transforming a traditionally manual and time-intensive task into a streamlined, efficient, and error-free process. By utilizing UiPath's Robotic Process Automation (RPA) platform and the RE Framework, the system ensures structured workflows, robust error handling, and scalability. The integration of modules for problem statement collection, code generation, email distribution, and monitoring provides a reliable and versatile solution for developers and non-technical users alike.

The automation not only saves time but also ensures the accuracy and consistency of code delivery. This is particularly valuable for scenarios involving repetitive or high-volume coding tasks, where manual methods are prone to inefficiencies and mistakes. Additionally, the system's robust error-handling mechanisms minimize disruptions by promptly identifying and addressing any issues encountered during the process.

Through its modular design, including a user-friendly interface and comprehensive logging capabilities, the project offers a transparent and flexible solution for users. It enhances productivity by allowing users to focus on higher-value tasks, such as problem-solving and application design, rather than repetitive coding tasks.

The project exemplifies how RPA and AI can address real-world challenges, providing practical solutions that improve efficiency, accuracy, and user satisfaction. Future enhancements may include expanding compatibility with various programming languages, integrating with development tools or repositories, and incorporating analytics to track code generation trends. This project marks a significant step forward in leveraging AI and automation to optimize coding workflows, serving as a model for future innovations in the field.

6.1 GENERAL:

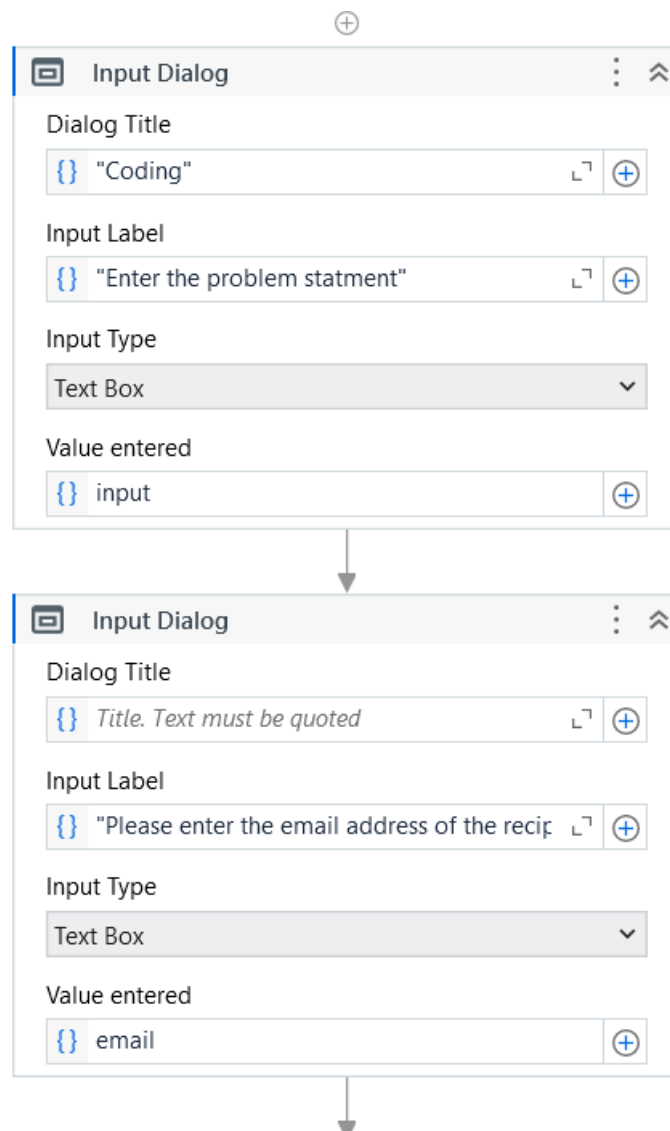
In general, the **AI-Driven Code Generation** project has successfully achieved its objectives of reducing manual intervention, accelerating the code generation process, and ensuring accurate and consistent code delivery to users. It provides developers and non-technical users with an efficient tool to automate code generation workflows effectively. Future enhancements may include integrating the system with version control platforms or integrated development environments (IDEs), expanding functionality to support a wider range of programming languages, or incorporating advanced analytics to evaluate the performance and usage of the generated code. The project has demonstrated its potential as a transformative solution for automating repetitive coding tasks, significantly improving productivity and accuracy. It serves as a model for adopting AI and automation to streamline software development processes and can inspire similar innovations in other domains requiring intelligent code or document generation.

APPENDICES

Appendix 1: Key code Snippet

This appendix provides **code snippets** for essential functionalities such as:

1. Capturing user input
2. Interfacing with chatgpt for code generation
3. Saving generated code to a file
4. Sending emails with attachments via SMTP



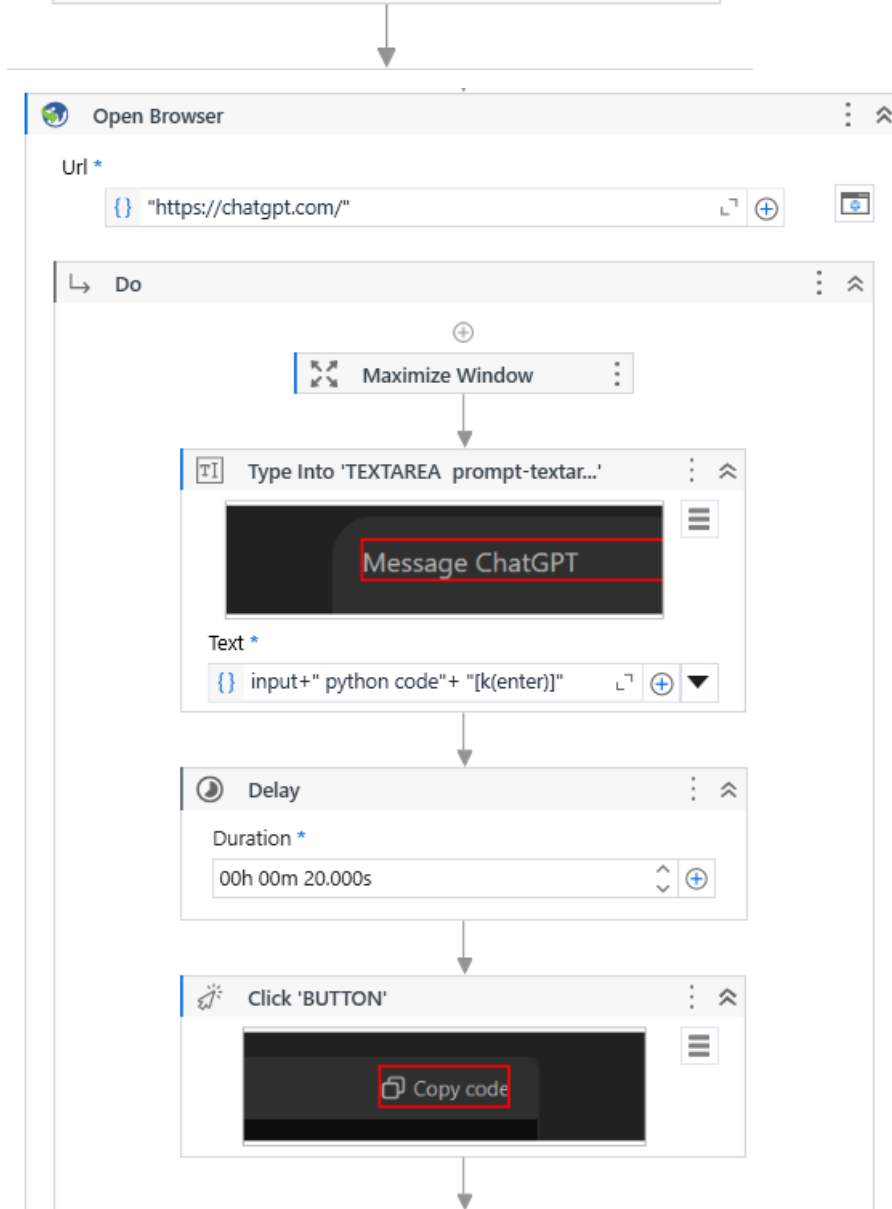
Input Dialog

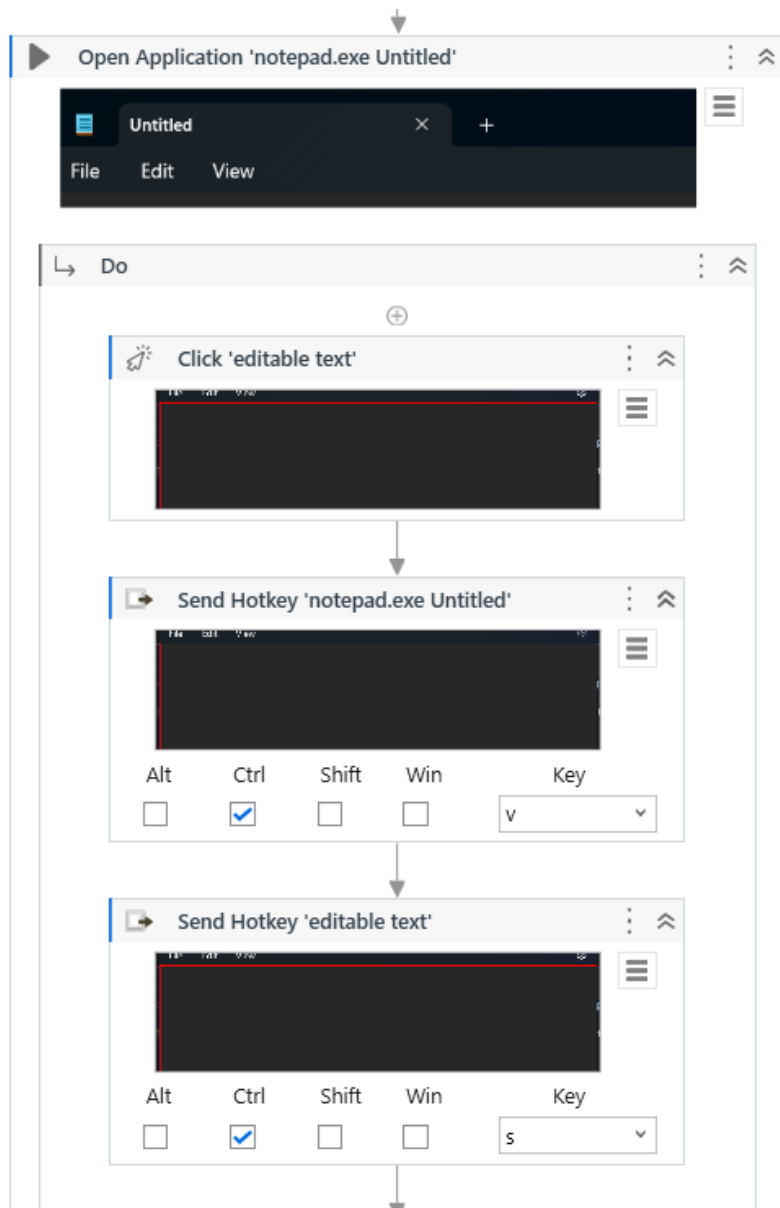
Dialog Title
`Title. Text must be quoted`

Input Label
`"File Name"`

Input Type
Text Box

Value entered
`file`





Type Into 'Edit'

File name: Untitled.txt

Save as type: Text documents (*.txt)

Text *

{ } file

Click 'Button'

Save Cancel

Close Application 'notepad.exe Untitled'

File Edit View Help

Send SMTP Mail Message

To * { } email

Subject { } "Your Code"

Body { } "Please find the attachment below"

Attach Files

Message Box

Text *

{ } "Email has been sent successfully. Please click OK to continue."

REFERENCES

1. Google AI Blog: Introducing Bard, a large language model for creative text generation <https://blog.google/technology/ai/bard-google-ai-search-updates/>
2. UiPath Studio User Guide: Creating an Automated Process <https://docs.uipath.com/studio/standalone/2023.4/user-guide/tutorials>
3. UiPath Forum: How to Copy and Paste Text in UiPath Studio <https://forum.uipath.com/t/paste-into-text-filed/159237>
4. Microsoft Word Documentation: How to Copy and Paste Text in Microsoft Word <https://support.microsoft.com/en-us/office/copy-and-paste-in-office-for-the-web-682704da-8360-464c-9a26-ff44abf4c4fe>
5. Real-Time Search with Generative AI https://arxiv.org/multi?group=grp_math&%2Ffind=Search