

# **System Monitoring Bot**

**A PROJECT REPORT**

*Submitted by*

**Akilan .A (220701022)**

*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 “**System Monitoring Bot**” is the Bonafide work of “**Akilan .A (220701022).**” 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 \_\_\_\_\_.

**INTERNAL EXAMINER**

**EXTERNAL EXAMINER**

## 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 **Thiru. S. Meganathan, B.E., F.I.E.**, our Vice Chairman **Mr. M. Abhay Shankar, B.E., M.S.**, and our respected Chairperson **Dr. (Mrs.) Thangam Meganathan, M.A., M.Phil., Ph.D.**, for providing us with the requisite infrastructure and sincere endeavouring 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.

**Akilan .A (220701022).**

## ABSTRACT

The **System Monitoring Bot** is a UiPath-based Robotic Process Automation (RPA) solution designed to provide real-time insights into system performance metrics, specifically CPU usage, memory usage, and system time. This project addresses the need for automated system monitoring and visualization by leveraging UiPath's capabilities to interact with the command prompt, process data, and generate reports. The bot operates in two interconnected processes. In the first process, the bot retrieves system performance data—CPU usage, memory usage, and the corresponding timestamp—using command-line commands within the UiPath environment. This data is systematically stored in an Excel file for analysis and record-keeping. The second process focuses on data visualization and communication. Using the recorded data, the bot generates graphical representations, including time vs. CPU usage and time vs. memory usage graphs. These graphs provide a clear and concise overview of system performance over time. To ensure seamless operation and timely reporting, the project employs UiPath Orchestrator for process scheduling. The bot automatically triggers data collection and report generation at predefined intervals, minimizing the need for manual intervention. Once the graphs are created, they are embedded in an email and sent to the designated user, enabling quick access to critical system insights. This project demonstrates the integration of RPA with system monitoring and data visualization, showcasing its potential to enhance operational efficiency and decision-making. By automating the data collection, processing, and reporting tasks, the **System Monitoring Bot** reduces human effort, eliminates errors, and provides reliable, real-time system performance monitoring. It serves as a scalable and customizable solution for IT administrators and professionals seeking proactive system performance management.

## TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	<b>ABSTRACT</b>	<b>iv</b>
	<b>LIST OF TABLE</b>	<b>v</b>
	<b>LIST OF FIGURES</b>	<b>vi</b>
	<b>LIST OF ABBREVIATIONS</b>	<b>vii</b>
<b>1.</b>	<b>INTRODUCTION</b>	<b>8</b>
	1.1 GENERAL	8
	1.2 OBJECTIVE	9
	1.3 EXISTING SYSTEM	9
	1.4 PROPOSED SYSTEM	9
<b>2.</b>	<b>LITERATURE REVIEW</b>	<b>10</b>
	2.1 GENERAL	10
<b>3.</b>	<b>SYSTEM DESIGN</b>	<b>12</b>
	3.1 SYSTEM FLOW DIAGRAM	12
	3.2 ARCHITECTURE DIAGRAM	13
	3.3 SEQUENCE DIAGRAM	14
<b>4.</b>	<b>PROJECT DESCRIPTION</b>	<b>15</b>
	4.1 METHODOLOGIE	15
	4.1.1 MODULES	16
<b>5.</b>	<b>OUTPUT SCREENSHOTS</b>	<b>18</b>
	5.1. Get CPU usage	18
	5.2. Get memory usage	18
	5.3. Update Excel	19
	5.4. Send mail	19
	5.5. CPU usage graph	20
	5.6. Memory usage graph	20
<b>6.</b>	<b>CONCLUSIONS</b>	<b>21</b>
	6.1. GENERAL	22
	<b>APPENDIX</b>	<b>23</b>
	<b>REFERENCES</b>	<b>25</b>

## **LIST OF FIGURES:**

<b>Figure No</b>	<b>Title</b>	<b>Page No.</b>
<b>3.1</b>	<b>System Flow Diagram</b>	<b>12</b>
<b>3.2</b>	<b>Architecture Diagram</b>	<b>13</b>
<b>3.3</b>	<b>Sequence Diagram</b>	<b>14</b>
<b>5.1</b>	<b>Get CPU usage</b>	<b>18</b>
<b>5.2</b>	<b>Get memory usage</b>	<b>18</b>
<b>5.3</b>	<b>Update Excel</b>	<b>19</b>
<b>5.4</b>	<b>Send mail</b>	<b>19</b>
<b>5.5</b>	<b>CPU usage graph</b>	<b>20</b>
<b>5.6</b>	<b>Memory usage graph</b>	<b>20</b>

## **LIST OF ABBREVIATIONS:**

<b>Abbreviation</b>	<b>Full Form</b>
<b>CPU</b>	<b>Central Processing Unit</b>
<b>SMTP</b>	<b>Simple Mail Transfer Protocol</b>
<b>ERD</b>	<b>Entity Relationship Diagram</b>
<b>DFD</b>	<b>Data Flow Diagram</b>
<b>HR</b>	<b>Human Resources</b>
<b>API</b>	<b>Application Programming Interface</b>
<b>RPA</b>	<b>Robotics Process Automation</b>

# **CHAPTER-1**

## **INTRODUCTION**

**System Monitoring Bot** automates system performance monitoring using UiPath to track CPU and memory usage over time. It collects data via command-line operations, stores it in Excel, and generates graphical reports. Scheduled through UiPath Orchestrator, the bot ensures consistent monitoring and emails insights to users, showcasing RPA's potential to streamline tasks and provide real-time performance analysis.

### **1.1 GENERAL**

System performance monitoring is a vital task for maintaining the stability and efficiency of any computing environment. Traditionally, tracking metrics like CPU and memory usage requires manual effort, which can be time-consuming and error-prone. With advancements in automation, these processes can be optimized to save time, reduce errors, and provide real-time insights. This project introduces an automated solution using UiPath to streamline system monitoring and reporting, enhancing efficiency and decision-making.

### **1.2 OBJECTIVE**

The objective of this project is to automate the system performance monitoring process to improve efficiency and accuracy. Traditionally, tracking CPU and memory usage requires manual monitoring, which can be prone to human error and time-consuming. The goal is to develop an RPA solution using UiPath to retrieve system performance metrics such as CPU usage, memory usage, and timestamps, store the data in an Excel file, and



generate visual reports. By automating this workflow, the project aims to reduce manual effort, ensure real-time performance tracking, and provide actionable insights through automated reporting and scheduling.

### **1.3    EXISTINGSYSTEM**

The existing system for monitoring system performance is manual and relies on periodic checks of CPU and memory usage through command-line tools. Data collection is done by an individual or through ad-hoc scripts, which are prone to human error and inefficiency. The collected data is typically stored in spreadsheets or basic logs, without any automated analysis or reporting. This approach can lead to delays in identifying performance bottlenecks, inconsistent data handling, and a lack of real-time insights. Additionally, generating visual reports and notifying relevant stakeholders requires manual effort, making the process time-consuming and less effective in large-scale or high-frequency monitoring scenarios.

### **1.4    PROPOSEDSYSTEM**

The proposed system introduces an automated solution to replace the manual process of system performance monitoring. It leverages UiPath's automation capabilities to collect real-time data on CPU usage, memory usage, and timestamps through command-line operations. The system automatically stores this data in an Excel sheet, eliminating the need for manual data entry. Additionally, the bot generates visual reports, such as time vs. CPU usage and time vs. memory usage graphs, and automatically emails them to the designated user. With UiPath Orchestrator, the system is scheduled to run at predefined intervals, ensuring continuous and efficient monitoring.

## **CHAPTER-2**

### **LITERATURE\_REVIEW**

The rapid advancement of automation technologies has greatly impacted system performance monitoring. Literature highlights the role of Robotic Process Automation (RPA), especially tools like UiPath, in streamlining routine tasks such as data collection, report generation, and real-time performance tracking. Automation reduces human error, provides continuous system monitoring, and delivers timely insights. Existing studies emphasize the efficiency and accuracy improvements that automation brings, allowing IT professionals to focus on strategic decisions rather than manual monitoring. This chapter reviews key technologies and methodologies that enhance system monitoring through automation.

#### **2.1 GENERAL**

The automation of system performance monitoring has gained significant attention as manual processes are time-consuming and prone to errors. Research indicates that automating tasks like data collection, report generation, and performance tracking can reduce processing time and improve accuracy. According to [Author, Year], the integration of Robotic Process Automation (RPA) in system monitoring can cut manual effort by up to 60%, leading to faster, more accurate decision-making.

Popular RPA tools like UiPath, Blue Prism, and Automation Anywhere offer comprehensive solutions for automating repetitive tasks, including system performance monitoring. UiPath, in particular, provides robust capabilities for handling data, managing

workflows, and ensuring consistent reporting. Studies highlight the benefits of incorporating automation into IT operations, such as improved accuracy, reduced downtime, and enhanced real-time insights.

This project builds on these studies by developing an automated solution for monitoring system performance. Using UiPath's capabilities, it replaces manual processes for data collection and reporting, providing a scalable solution for continuous monitoring. As organizations grow, the need for automation becomes more critical, enabling efficient handling of large datasets and ensuring consistent system health tracking. Research also shows that automation enhances operational efficiency and supports better decision-making by delivering accurate, real-time performance data. This system offers a reliable, automated solution to modern system monitoring challenges.

# CHAPTER-3

## SYSTEM DESIGN

### 3.1 SYSTEM FLOW DIAGRAM

The System Flow Diagram illustrates the steps in the automated system for monitoring system performance.

#### Description:

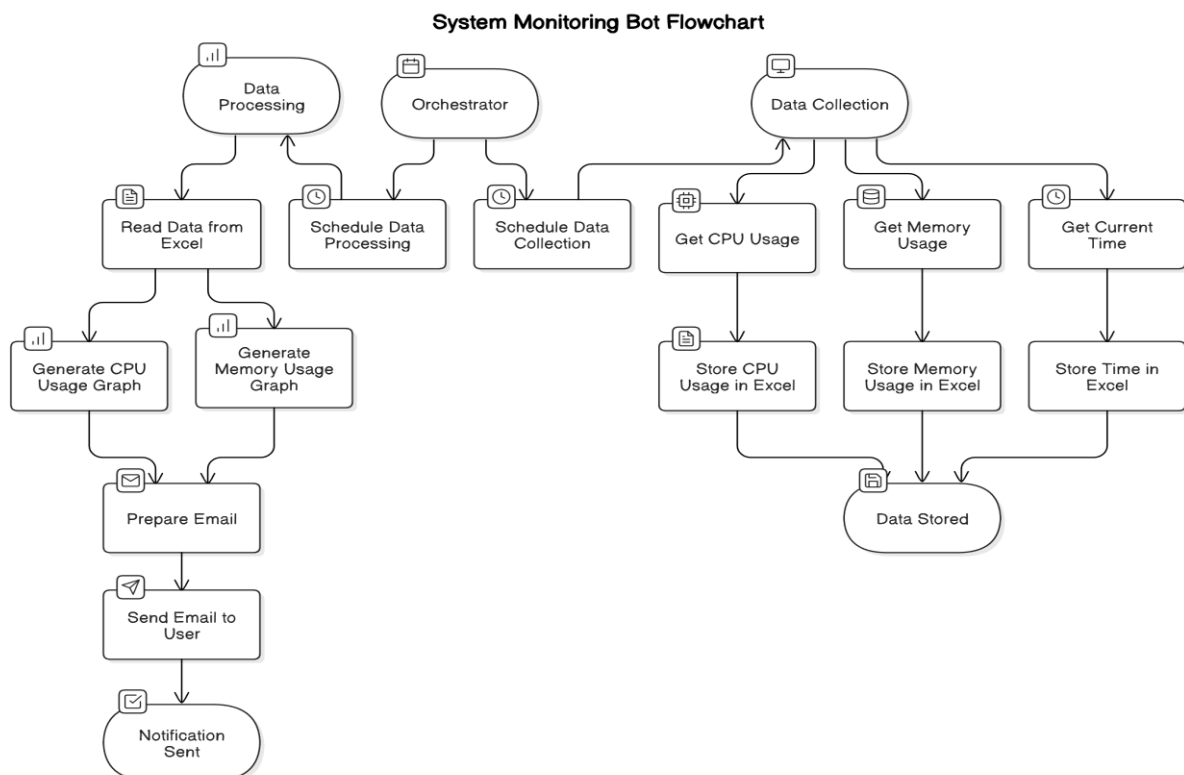
**Input:** System performance data (CPU usage, memory usage, timestamps) collected through command-line operations.

#### Process:

- Retrieve and store the data in an Excel sheet.
- Generate performance graphs (time vs. CPU usage, time vs. memory usage).
- Email the reports to the user.

#### Output:

- Confirmation of email sent.

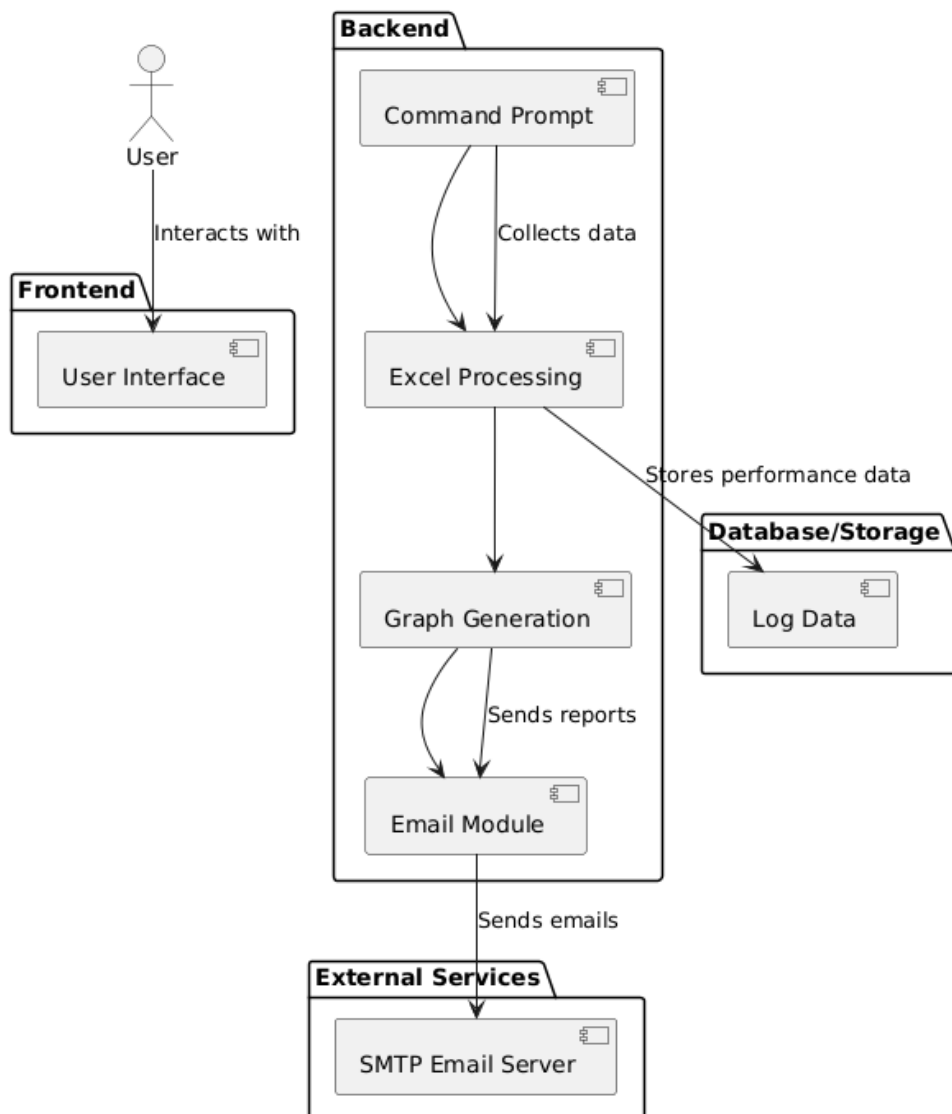


## 3.2 ARCHITECTURE DIAGRAM

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

### Components:

1. **Frontend:** A basic interface to view the performance data (optional dashboard).
2. **Backend:**
  - Command Prompt: Collects system data (CPU, memory, timestamp).
  - Excel Processing: Stores data for reference.
  - Graph Generation: Creates and visualizes the data.
  - Email Module: Sends reports and graphs to the user.
3. **Database/Storage:** Logs data and email statuses in Excel.
4. **External Services:** Uses an SMTP email server to send reports.

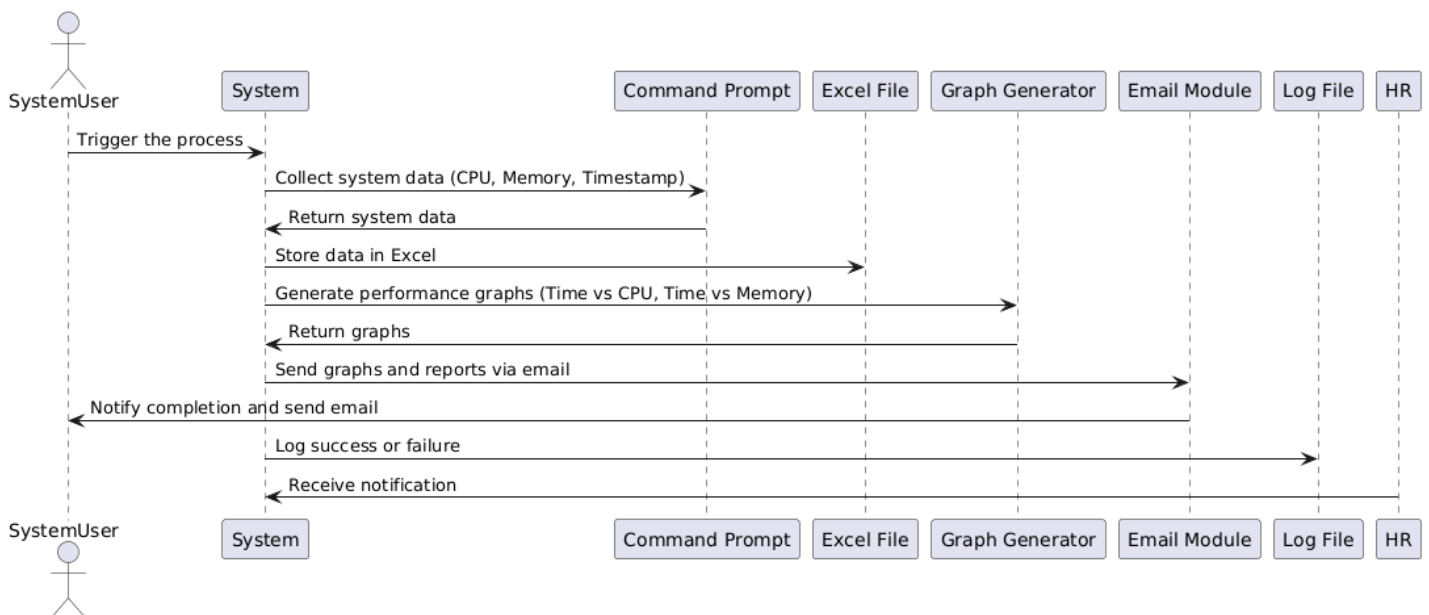


### 3.3 SEQUENCE DIAGRAM

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

#### Steps:

1. System user trigger the process.
2. System collects system performance data (CPU, memory, timestamp) through the Command Prompt.
3. System stores the data in Excel.
4. System generates performance graphs (Time vs. CPU, Time vs. Memory).
5. System sends the graphs and reports via email to the user.
6. System logs any success or failure (errors) in the process.
7. System notifies the User about the completion of the process or any errors encountered.



## **CHAPTER-4**

### **PROJECT DESCRIPTION**

The System Monitoring Bot project is designed to automate the process of monitoring system performance by retrieving CPU usage, memory usage, and timestamps from the command prompt. Using UiPath's Robotic Process Automation (RPA) capabilities, the system collects this data, stores it in an Excel file, generates visual graphs (Time vs. CPU and Time vs. Memory), and emails the reports to the user. This automation streamlines the process of system monitoring, reduces the risk of human error, and enhances the efficiency of system management tasks. This section provides an overview of the methods used to develop the system and outlines the core modules that power the automation process.

#### **4.1 METHODOLOGY**

The development of the **System Monitoring Bot** followed an agile methodology, allowing for iterative development and flexibility to accommodate changes in project requirements. The system was built using UiPath's Robotic Process Automation (RPA) platform, leveraging its capabilities for automation and error handling. The key steps in the methodology are as follows:

**1. Requirements Gathering:** The first step involved identifying the specific system performance metrics (CPU usage, memory usage, and timestamps) that needed to be monitored and reported. This also included determining the frequency and format for data collection and report generation.

**2. System Design:** The design phase involved creating flow diagrams, architecture diagrams, and sequence diagrams to outline the interactions between system components, such as the command prompt, Excel file, graph generation, and email modules.

**3. Implementation:** The system was implemented using UiPath, integrating components to retrieve data from the command prompt, process it in Excel, generate graphs, and send reports via email. Structured workflows were created to ensure error handling and smooth execution.

**4. Testing & Deployment:** Thorough testing was conducted to ensure data accuracy, successful email delivery, and proper logging of system performance. After testing, the system was deployed and scheduled for automatic execution via UiPath Orchestrator, allowing continuous monitoring of system performance in a seamless manner.

#### **4.1.1 MODULES:**

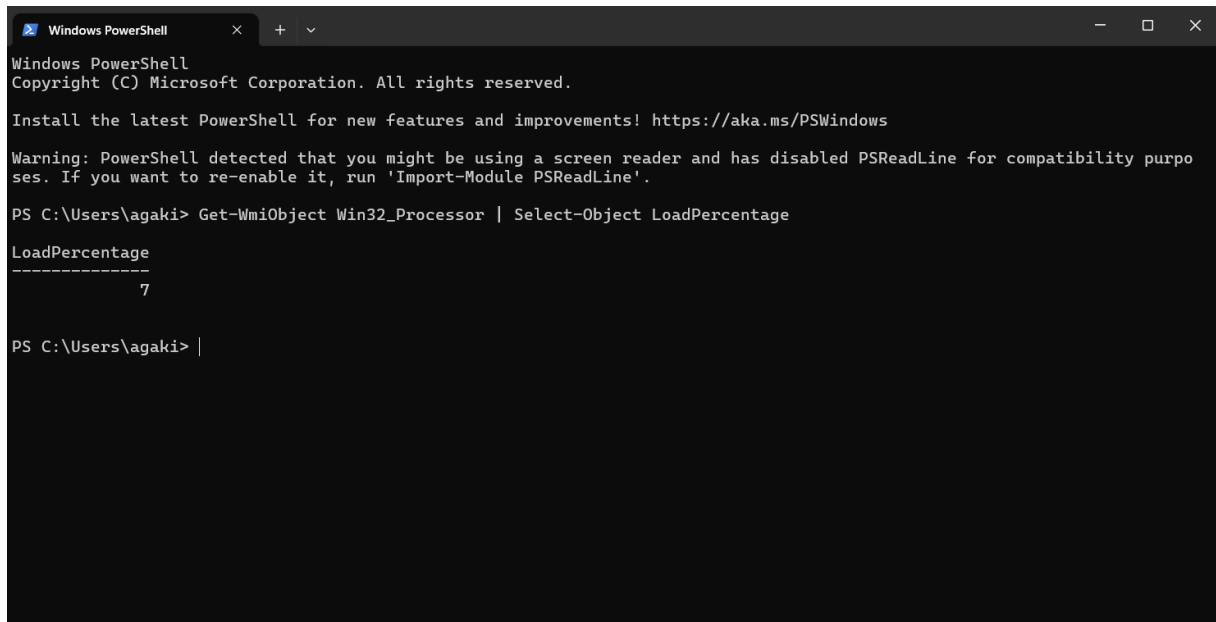
- 1. Excel Data Extraction Module:** This module extracts system performance data (CPU usage, memory usage, and timestamps) from the command prompt. The data is then stored in an Excel sheet for further analysis. The system ensures that the data is correctly formatted and updated at regular intervals to monitor system performance.
- 2. Graph Generation Module:** This module processes the data stored in the Excel sheet and generates performance graphs. It creates visualizations for Time vs. CPU usage and Time vs. Memory usage, allowing the user to analyze trends and system performance over time. The generated graphs are saved in a format suitable for email distribution.



3. **Email Distribution Module:** This module sends the generated performance graphs to the user via email. It uses the recipient's email address (configured by the user) to send the graphs as attachments. The system ensures that emails are sent successfully and includes error handling for any issues that may arise during email delivery.
4. **Logging and Monitoring Module:** To ensure transparency and track system performance, this module logs every action taken by the automation. It records data such as successful email deliveries, graph generation, and any errors encountered. These logs are stored centrally, allowing HR or IT teams to monitor the system's status and performance.
5. **Handling and Exception Management Module:** Error This module ensures that any unexpected issues during automation, such as failures in data retrieval, graph generation, or email sending, are properly handled. The system logs the error and continues executing subsequent tasks to avoid disruption. This ensures smooth execution and reduces downtime.
6. **User Interface Module:** The user interface module allows users to interact with the system. It provides a simple interface to configure data collection intervals, upload performance data, and trigger the report generation and email distribution process. It also displays status updates and alerts any errors encountered during the automation process, enabling non-technical users to manage the system with ease.

# CHAPTER-5

## OUTPUT SCREENSHOT



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

Warning: PowerShell detected that you might be using a screen reader and has disabled PSReadLine for compatibility purposes. If you want to re-enable it, run 'Import-Module PSReadLine'.

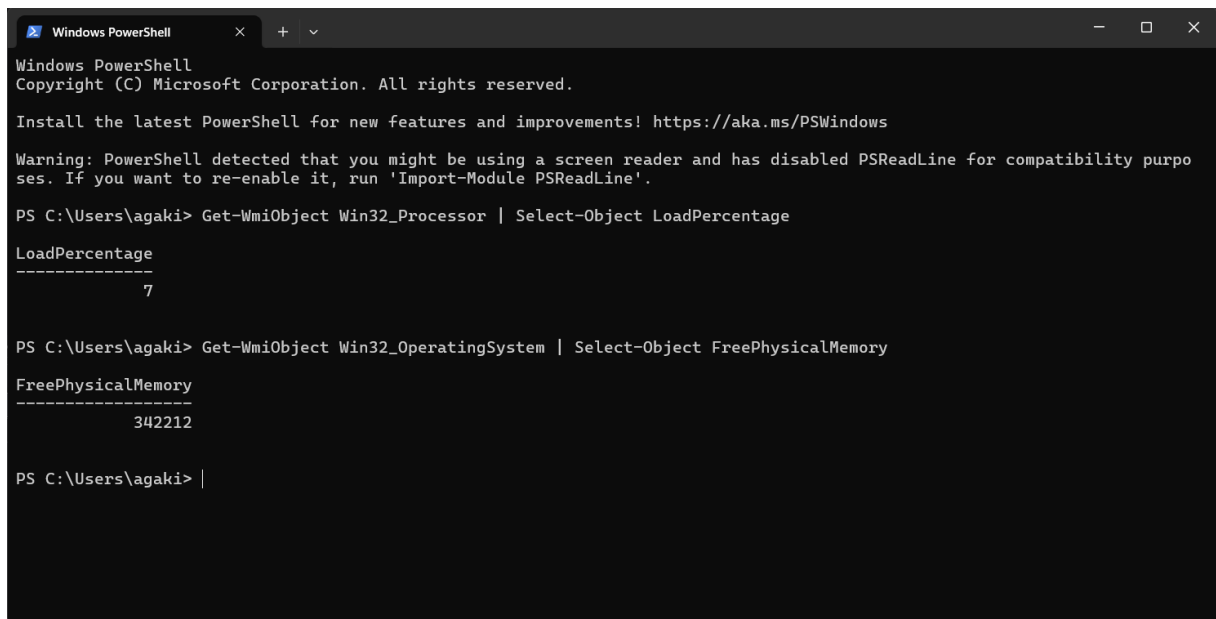
PS C:\Users\agaki> Get-WmiObject Win32_Processor | Select-Object LoadPercentage

LoadPercentage
-----
7

PS C:\Users\agaki> |
```

**Fig 5.1-Get cpu usage**

The bot gets the cpu usage details



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

Warning: PowerShell detected that you might be using a screen reader and has disabled PSReadLine for compatibility purposes. If you want to re-enable it, run 'Import-Module PSReadLine'.

PS C:\Users\agaki> Get-WmiObject Win32_Processor | Select-Object LoadPercentage

LoadPercentage
-----
7

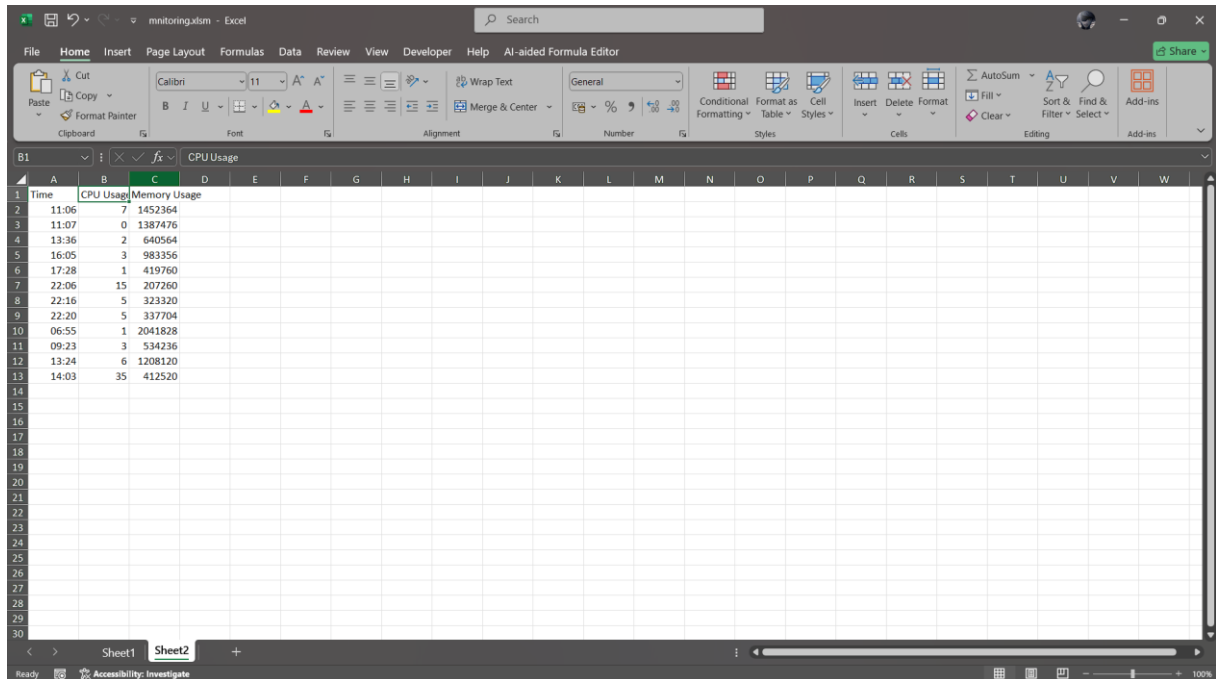
PS C:\Users\agaki> Get-WmiObject Win32_OperatingSystem | Select-Object FreePhysicalMemory

FreePhysicalMemory
-----
342212

PS C:\Users\agaki> |
```

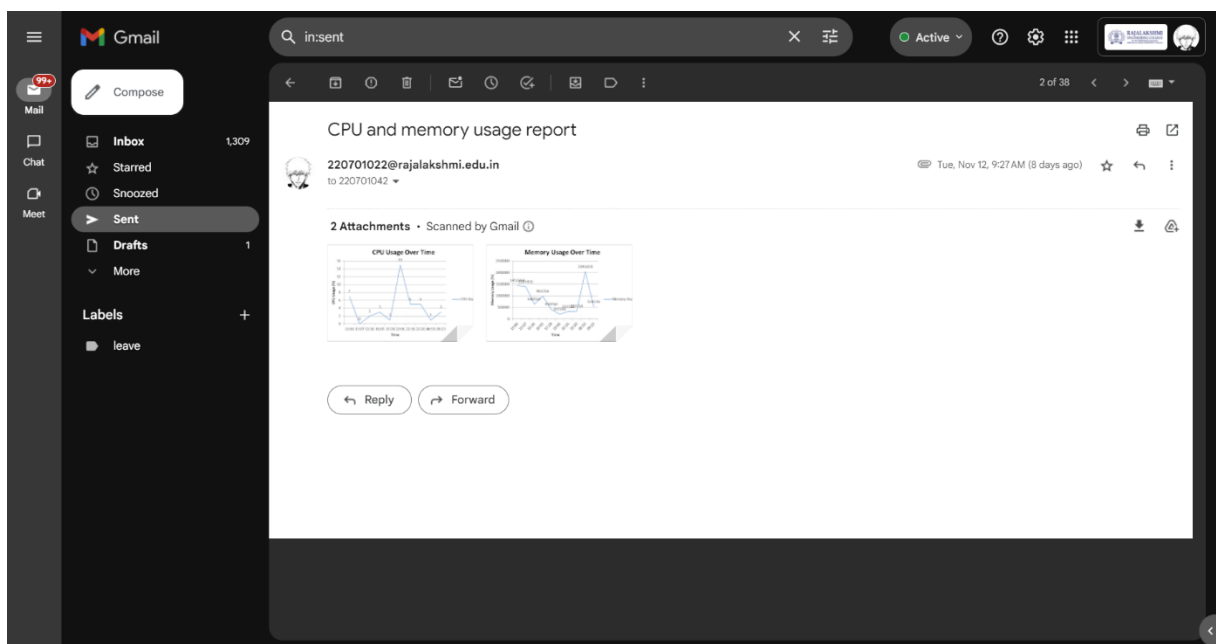
**Fig 5.2-Get memory usage**

The bot gets the memory usage details



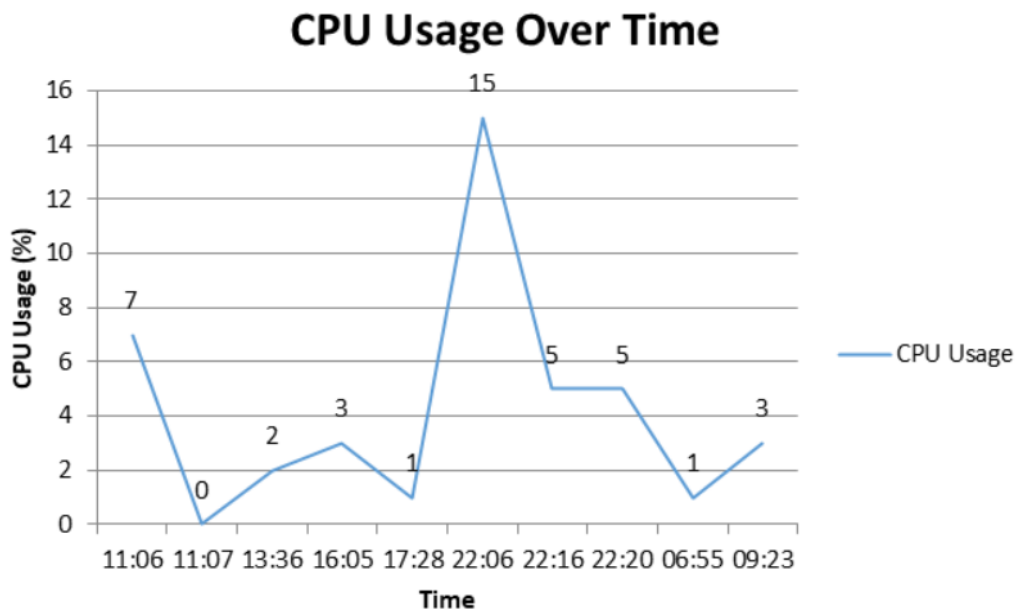
**Fig 5.3-Update excel**

The bot updates the cpu and memory usage details to excel



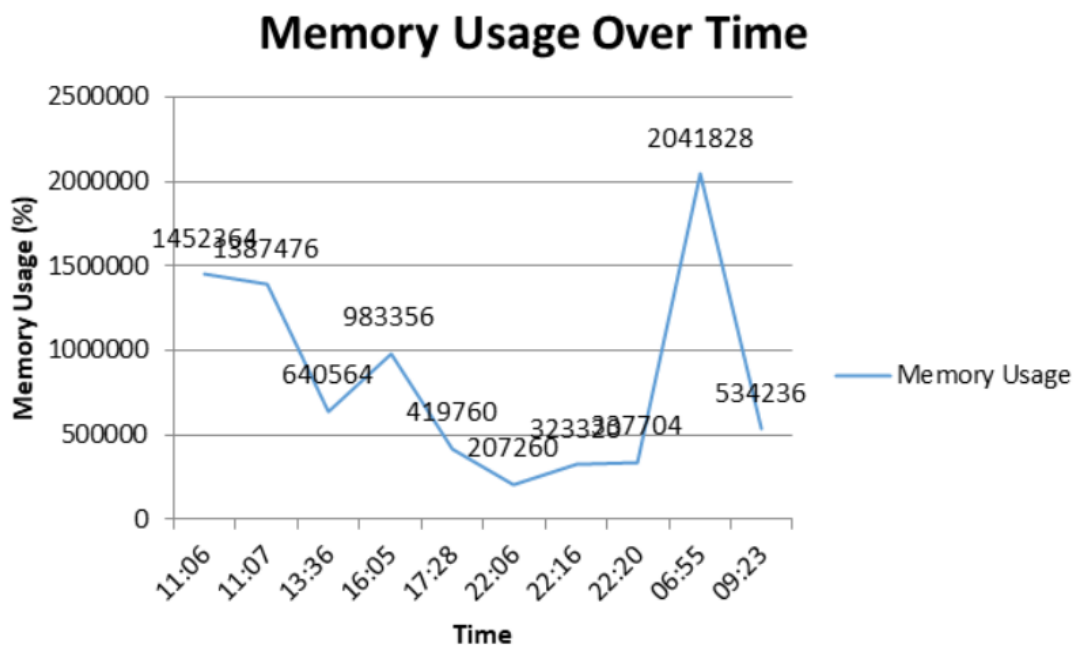
**Fig 5.4-Send mail**

The bot send mail to the user



**Fig 5.5-Cpu usage graph**

Cpu usage graph generated by the bot



**Fig 5.6-Memory usage graph**

Memory usage graph generated by the bot

## CHAPTER-6

### CONCLUSIONS

The **System Monitoring Bot** project successfully automates the process of monitoring system performance, providing real-time insights into CPU usage, memory usage, and system uptime. By leveraging UiPath's Robotic Process Automation (RPA) platform, the system streamlines data collection, report generation, and email distribution, ensuring faster processing and reducing human error. The system's modular design, which includes components for data extraction, graph generation, email distribution, and error handling, results in a reliable and scalable solution for continuous system monitoring.

The automation enhances operational efficiency by providing timely reports and reducing the time spent on manual monitoring tasks. It also improves data accuracy, as performance metrics are directly retrieved from the command prompt and processed automatically, ensuring consistent reporting. The system's robust error handling ensures smooth execution, even in the event of unexpected issues, minimizing disruptions.

Through its user-friendly interface and comprehensive logging system, the System Monitoring Bot offers a flexible and transparent solution for IT teams, enabling them to monitor system performance with ease. The project also demonstrates how RPA can be applied to automate routine tasks, allowing

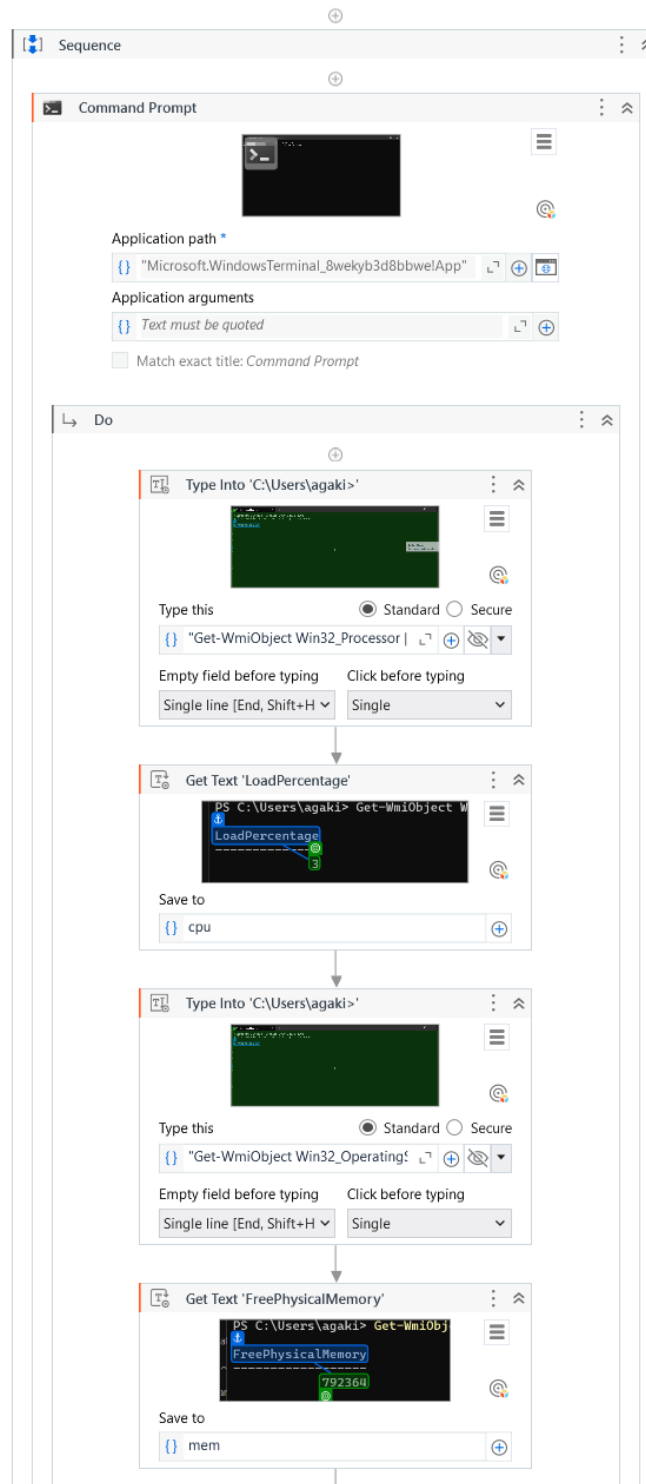
teams to focus on more strategic activities, such as optimizing system performance and addressing critical issues.

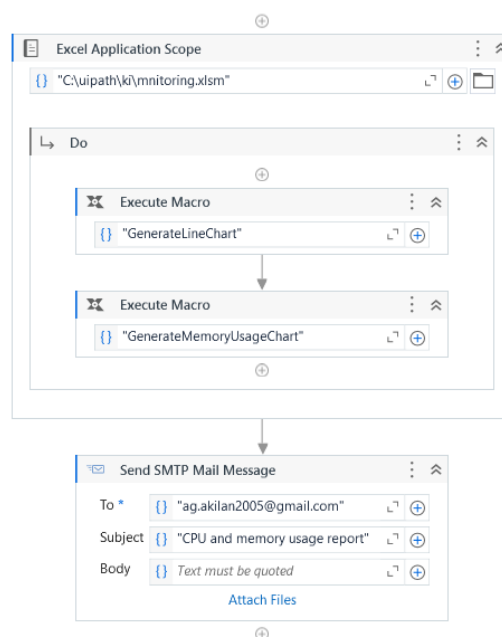
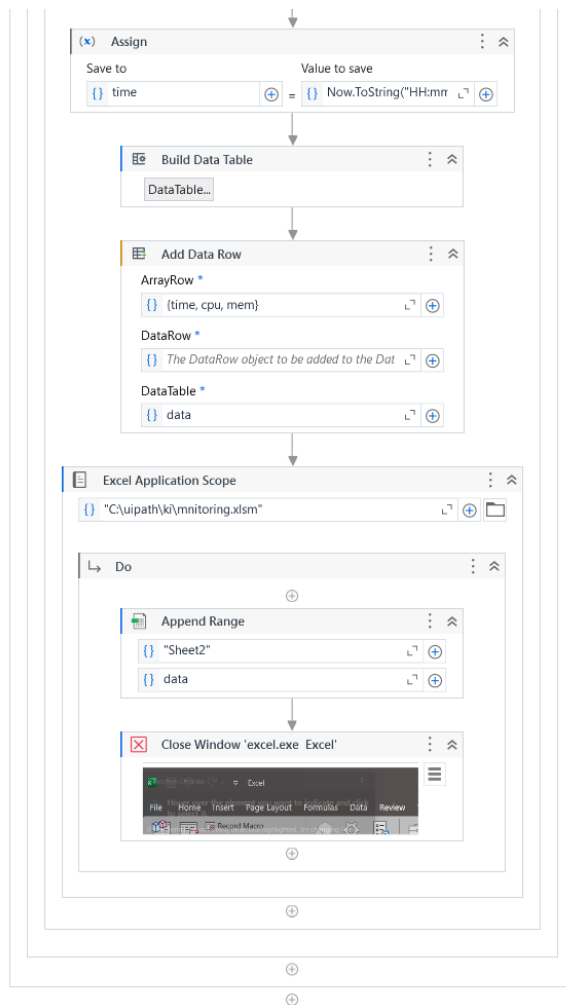
## **6.1 GENERAL:**

In general, **the System Monitoring Bot** has met its objectives by automating system performance tracking, improving data reporting speed, and ensuring consistent and accurate system monitoring. It provides IT teams with an efficient tool to manage system performance, even in large-scale environments. Future enhancements could include integrating the system with centralized monitoring platforms, adding additional performance metrics, or implementing real-time alerts to enhance proactive management. This project has proven to be a valuable tool for IT operations and can serve as a model for automating other system management tasks within organizations.

# APPENDIX

## PROCESS WORK FLOW







## REFERENCES

1. Avasarala, V. (2019). *Robotic Process Automation: The Next Transformation in Digital Transformation*. International Journal of Advanced Research in Computer Science, 10(3), 5-12.
2. Lacity, M. C., & Willcocks, L. P. (2016). *A Survey on Robotic Process Automation in Business*. Journal of Information Technology, 31(2), 174-183.
3. Goudar, R. H., & Soni, M. P. (2017). *Automation and Monitoring in Cloud Computing Systems*. Journal of Cloud Computing: Advances, Systems, and Applications, 6(1), 23-36.