THE INTELLIGENT SCREEN TIME MONITORING BOT FOR LAPTOPS

A PROJECT REPORT

Submitted by

AGILA SHREE A (220701015)

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 **"THE INTELLIGENT SCREEN TIME MONTORING BOT FOR LAPTOPS"** is the Bonafide work of **"AGILA SHREE A (220701015)"** 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.

AGILA SHREE A (220701015)

ABSTRACT:

"The Intelligent Screentime Monitoring Bot for laptops" is an innovative Robotic Process Automation (RPA) solution designed to promote digital well-being and productivity. Developed on the UiPath platform, this bot automates the tracking of application usage on laptops over a designated period. By utilizing Python scripting integrated with UiPath workflows, the bot monitors active windows, records screen time for each application, and compiles the data into a detailed Excel report. This report is then automatically emailed to the user for easy access and review.

This system improves upon existing methods, which typically rely on manual tracking or basic third-party applications that offer limited insights. The Intelligent Screentime Monitoring Bot automates the entire process accurately capturing usage data, generating reports, and sending them to users thus eliminating the need for manual intervention. Its versatility makes it suitable for various scenarios, such as analysing workplace productivity, monitoring children's screen time, and supporting digital detox efforts. By offering an efficient, automated solution, this bot helps users optimize their screen usage and make more informed decisions about their digital habits, ultimately fostering healthier interactions with technology.

TABLE OF CONTENTS

LIST OF FIGURES:

LIST OF ABBREVIATIONS:

ABBREVIATION	ACCRONYM
RPA	Robotic Process Automation
SMTP	Simple Mail Transfer Protocol
RE	Robotic enterprise
OCR	Optical Character Recognition

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

"The Intelligent Screentime Monitoring Bot for laptops" is an innovative solution leveraging Robotic Process Automation (RPA) to promote digital well-being and productivity. Designed on the UiPath platform, this project automates the tracking and reporting of screen usage on laptops, enabling users to gain valuable insights into their app usage behaviour.

As our reliance on technology grows, understanding and managing screentime has become increasingly vital. This bot offers an advanced, user-friendly tool to monitor application activity over a specified duration. By integrating Python scripting with UiPath workflows, it records the active screentime of each application, consolidates the data into an Excel report, and sends it directly to the user's email for convenient review.

Beyond its core functionality, the Intelligent Screentime Monitoring Bot For laptops serves as a versatile tool that can be adapted to various contexts, including workplace productivity analysis, parental control over children's screen usage, or personal digital detox initiatives. By automating the traditionally manual process of time tracking, it minimizes user effort and maximizes data accuracy. The use of Python scripting allows for precise computation of active windows, while UiPath ensures a smooth, user-friendly interface for configuration and report delivery. This integration not only streamlines monitoring but also empowers users to make informed decisions about their digital habits, fostering a healthier balance between work and leisure.

The UiPath Automation Platform, the backbone of this project, employs low-code development, OCR capabilities, and integration with Python to simplify complex tasks. Through its flexible and robust design, this screentime monitoring bot sets the

stage for smarter, more efficient digital resource management, underscoring the transformative potential of RPA in daily life.

The Intelligent Screentime Monitoring Bot for laptops extends its utility beyond personal monitoring by addressing diverse use cases such as workplace productivity analysis, digital detox efforts, and parental control. Its adaptability and automation empower users to track and optimize screen usage effectively. By offering actionable insights, the bot helps promote mindful technology use. This makes it a versatile tool in an increasingly digital world.

1.1 OBJECTIVE

The objective of the Intelligent Screentime Monitoring Bot for laptops is to automate the process of monitoring application usage on laptops, providing users with detailed insights into their screentime. The bot aims to promote digital well-being by enabling users to track, analyze, and optimize their screen usage through automated reporting and email integration.

1.2 EXISTING SYSTEM

In the current scenario, screentime monitoring primarily relies on manual tracking or third-party applications, which often lack efficiency and accuracy. Users need to monitor their activity themselves or depend on basic tools that provide limited insights into app-specific usage. These systems are typically not automated, requiring significant effort to collect and analyze data. Additionally, existing solutions often fail to deliver consolidated reports in convenient formats like Excel or integrate seamlessly with email services, making the process cumbersome and less effective for users seeking comprehensive screentime management.

1.3 PROPOSED SYSTEM

The Intelligent Screentime Monitoring Bot for laptops provides a comprehensive and automated solution to address the inefficiencies of existing systems by tracking application usage on laptops over a user-defined duration. Leveraging the robust capabilities of the UiPath platform combined with Python scripting, the bot ensures precise data collection and real-time monitoring of active windows. It calculates the screentime of each application, consolidates the data into a detailed Excel report, and seamlessly integrates with email services to automatically send the report to the user’s email address for convenient access. By automating the entire process—from tracking screentime to report delivery, the bot eliminates

manual intervention, enhances data accuracy, and streamlines usability. Its adaptability and scalability make it suitable for diverse applications, such as workplace productivity monitoring, digital detox initiatives, and parental control. This system not only simplifies screentime management but also empowers users with actionable insights to foster healthier and more productive digital habits.

CHAPTER 2

SYSTEM DESIGN

2.1 SYSTEM FLOW DIAGRAM

The workflow Fig 2.1 starts with user input validation; invalid input terminates the process, while valid input initializes monitoring and tracks app usage. Upon completion, a report is generated, saved to Excel, and emailed to the user. If time isn't complete, tracking continues until the set duration ends.

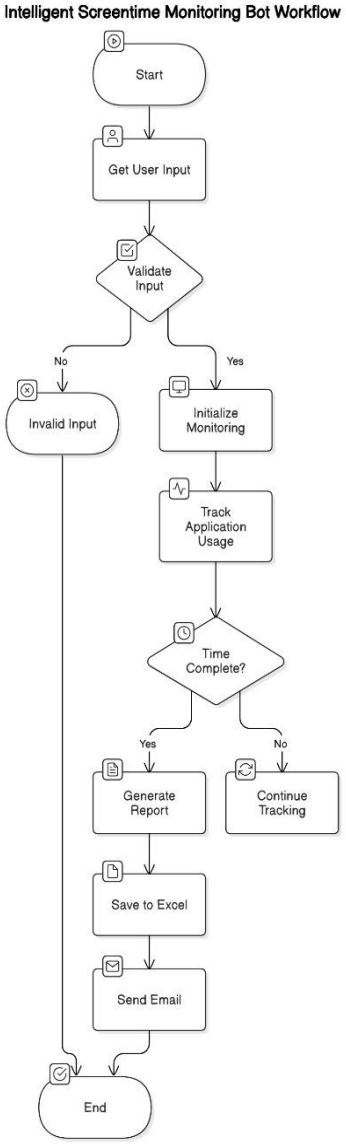


FIG 2.1 SYSTEM FLOW DIAGRAM

2.2 ARCHITECTURE DIAGRAM

This architecture diagram Fig 2.2 illustrates the flow of the Screentime Monitoring system. It begins with User Input, which is processed by a Python Script for monitoring activities. The script integrates with UiPath to automate the tracking process and perform Screentime Calculation. The results are compiled into an Excel Report, which is then sent to the user through an Email Notification.

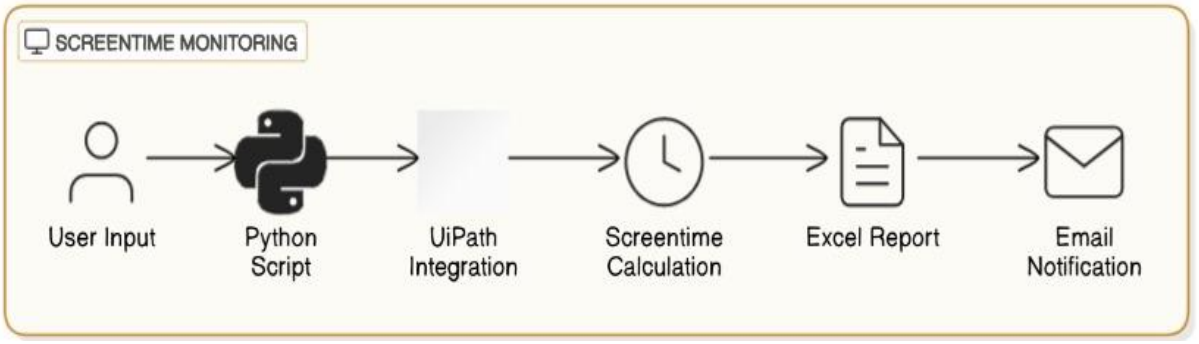


Fig 2.2 Architecture Diagram

2.3 SEQUENCE DIAGRAM

The sequence diagram Fig 2.3 depicts the interaction flow for an Intelligent Screentime Monitoring Bot. It starts with user input, tracks app usage via a Python script, generates an Excel report, and sends it to the user through email service after monitoring is complete.

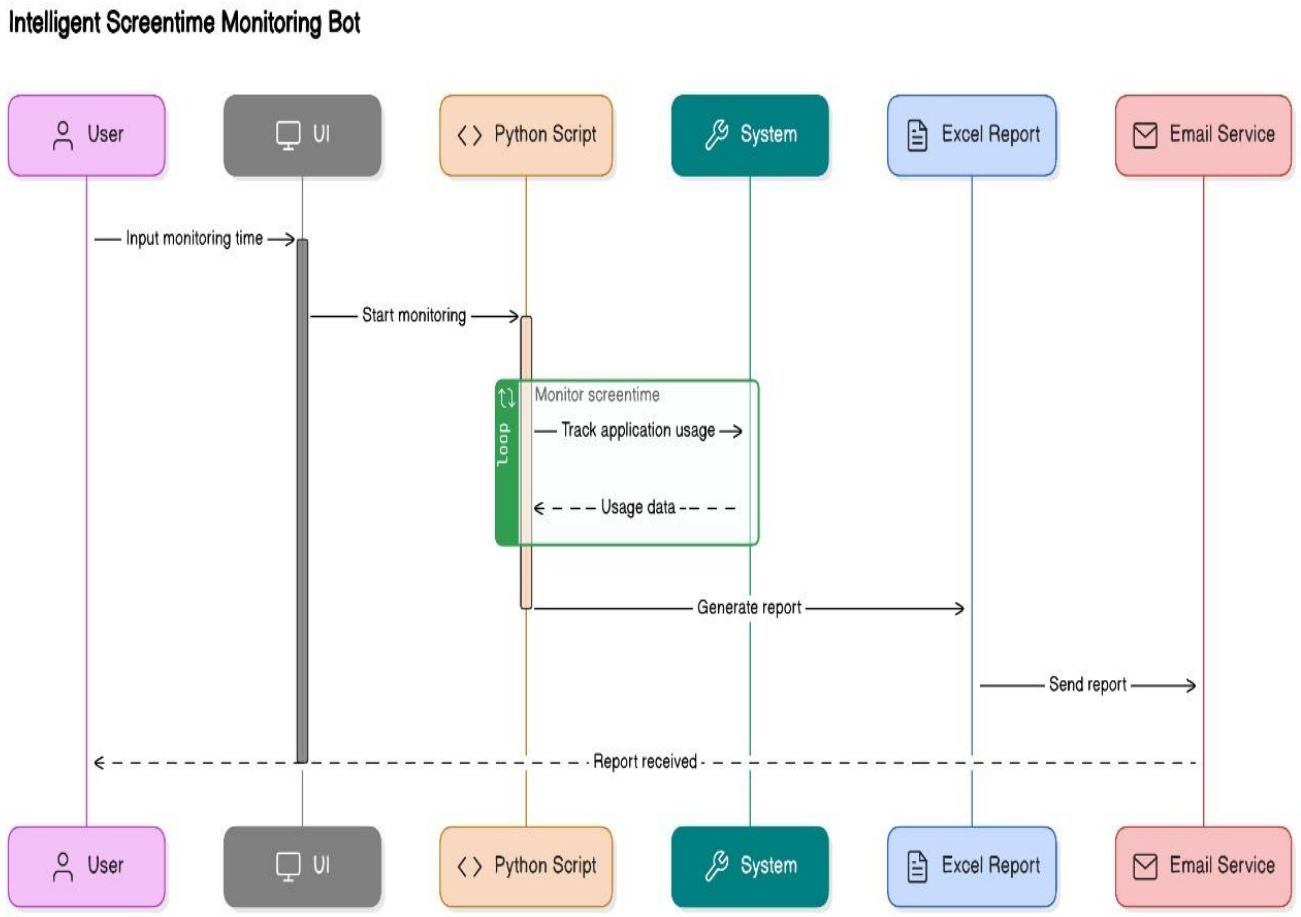


Fig 2.3 Sequence Diagram

CHAPTER 3

PROJECT DESCRIPTION

The Intelligent Screentime Monitoring Bot is a productivity-focused automation tool designed to monitor and regulate digital screen usage. The bot allows users to set monitoring durations and tracks application usage during that time. A Python script integrated with UiPath records screentime data and calculates the usage. At the end of the monitoring period, the bot generates an Excel report detailing the usage statistics and sends it to the user via email. This system helps users understand their screen habits, reduce excessive screen time, and improve productivity through data-driven insights.

3.1 MODULES:

3.1.1. Duration Input

- Receive user input for the monitoring duration via an input dialog box.
- Convert the user-provided input into an integer value for further processing.

3.1.2. Python Script Initialization

- Load and execute the Python script to track screentime and capture application usage.
- Pass the user-defined monitoring duration as input to the Python script for real-time tracking.

3.1.3. Screentime Data Retrieval

- Retrieve application usage data from the Python script.
- Process and prepare the output (e.g., application names and usage durations) for further use.

3.1.4. Data Output Preparation

- Use JSON deserialization to transform the Python script's output into a structured format.
- Convert the deserialized data into a Data Table for easy manipulation and reporting.

3.1.5. Report Creation

- Dynamically generate an Excel file named "op.xlsx" within the project directory.
- Populate the Excel file with screentime statistics, including application names and usage durations, using the Write Range activity.

3.1.6. Email Initialization

- Configure SMTP settings to send the generated Excel report to the specified recipient.
- Attach the dynamically created "op.xlsx" file and include a subject line and body text in the email.

3.1.7. Report Dispatch

- Automatically send the Excel report via email to the user.
- Ensure the email includes all necessary attachments and message details.

3.1.8. Completion Notification

- Display a message box indicating the successful completion of the process.
- Notify the user that the Excel report has been generated and sent.

CHAPTER 4

OUTPUT SCREENSHOTS

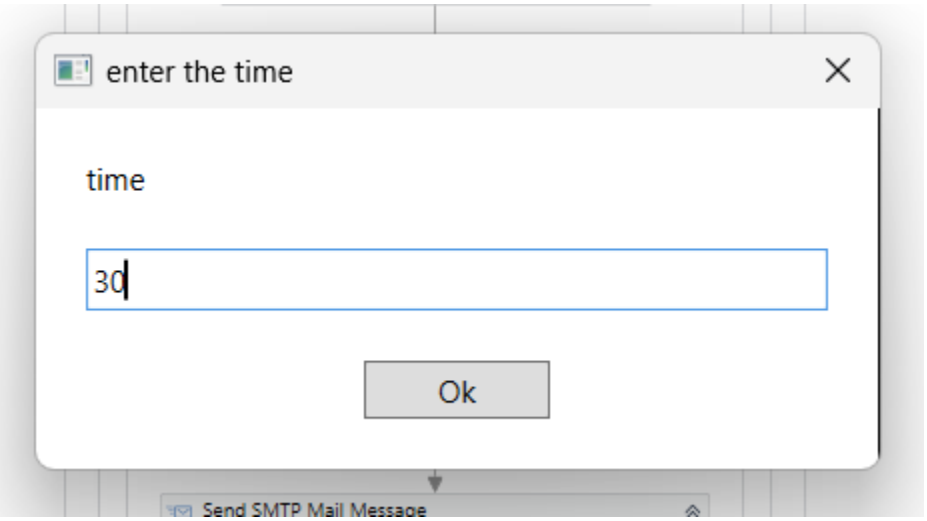


Fig 4.1 Time Input

The intelligent screen time monitoring system allows users to input their desired screen time limits or track their actual screen usage by entering time minutes in Fig 4.1

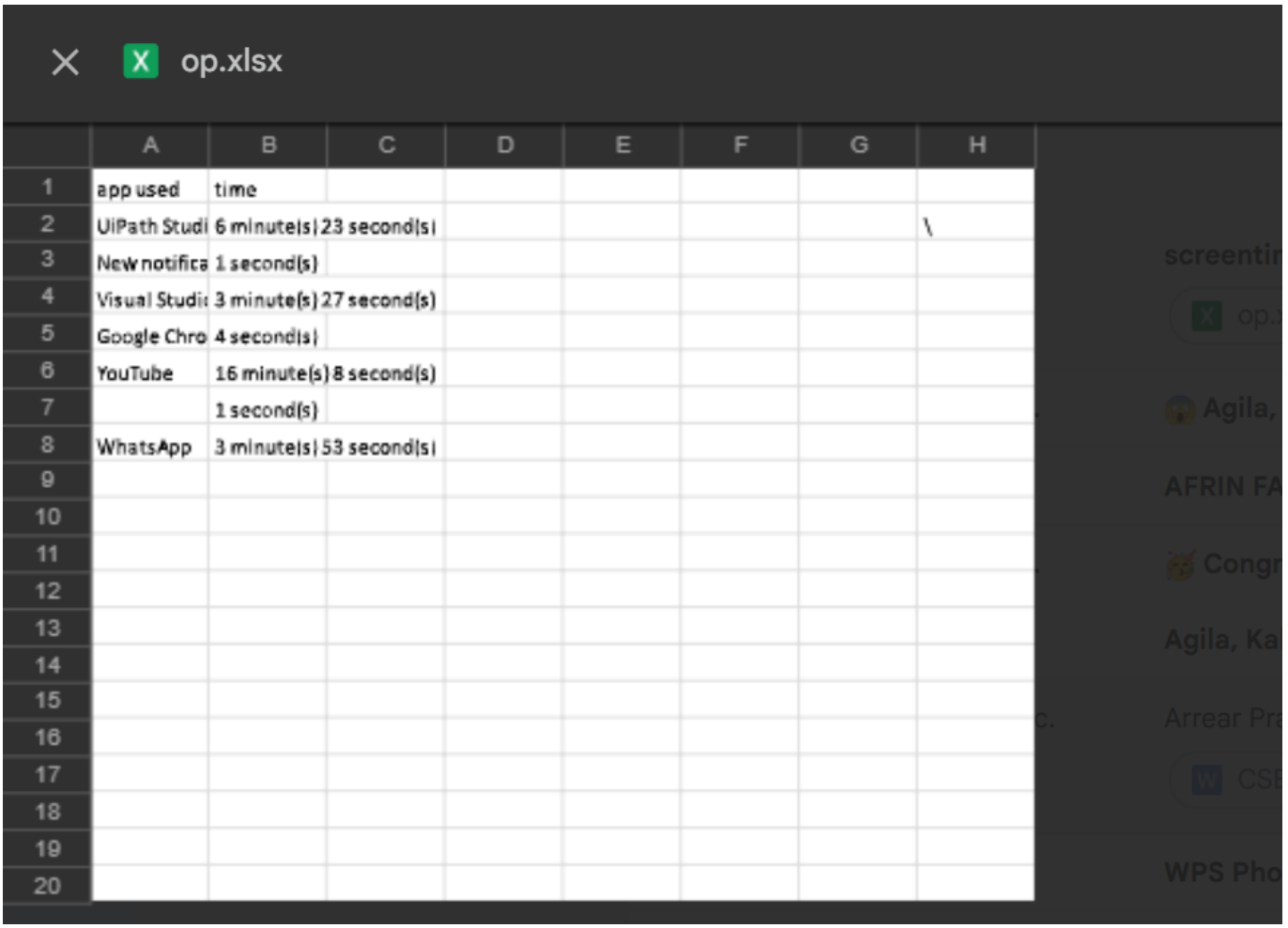


Fig 4.2 Excel sheet

The Excel sheet Fig 4.2 generated by the Intelligent Screentime Monitoring Bot provides a detailed summary of application usage during the specified monitoring duration

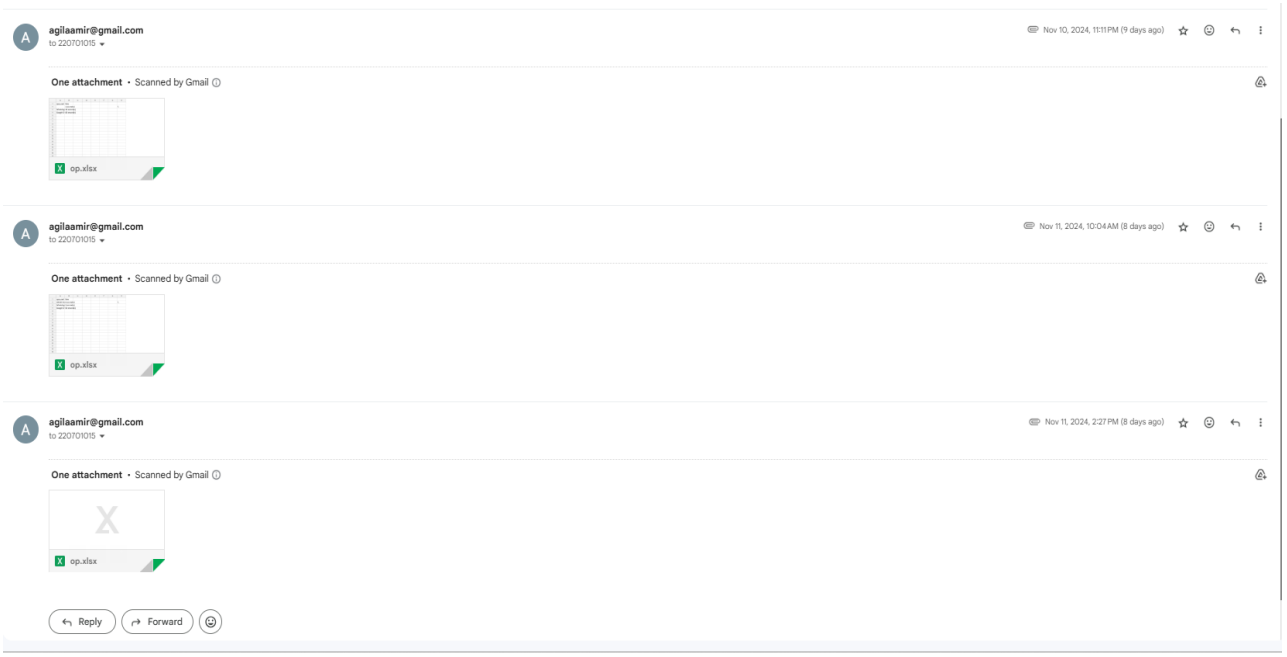


Fig 4.3 Email Integration

The email sent by the Intelligent Screentime Monitoring Bot Fig 4.3 contains the generated Excel report ("op.xlsx") as an attachment. It provides a summary of the user's screentime statistics for the specified monitoring duration, helping them analyze their application usage effectively.

CHAPTER 6

CONCLUSION

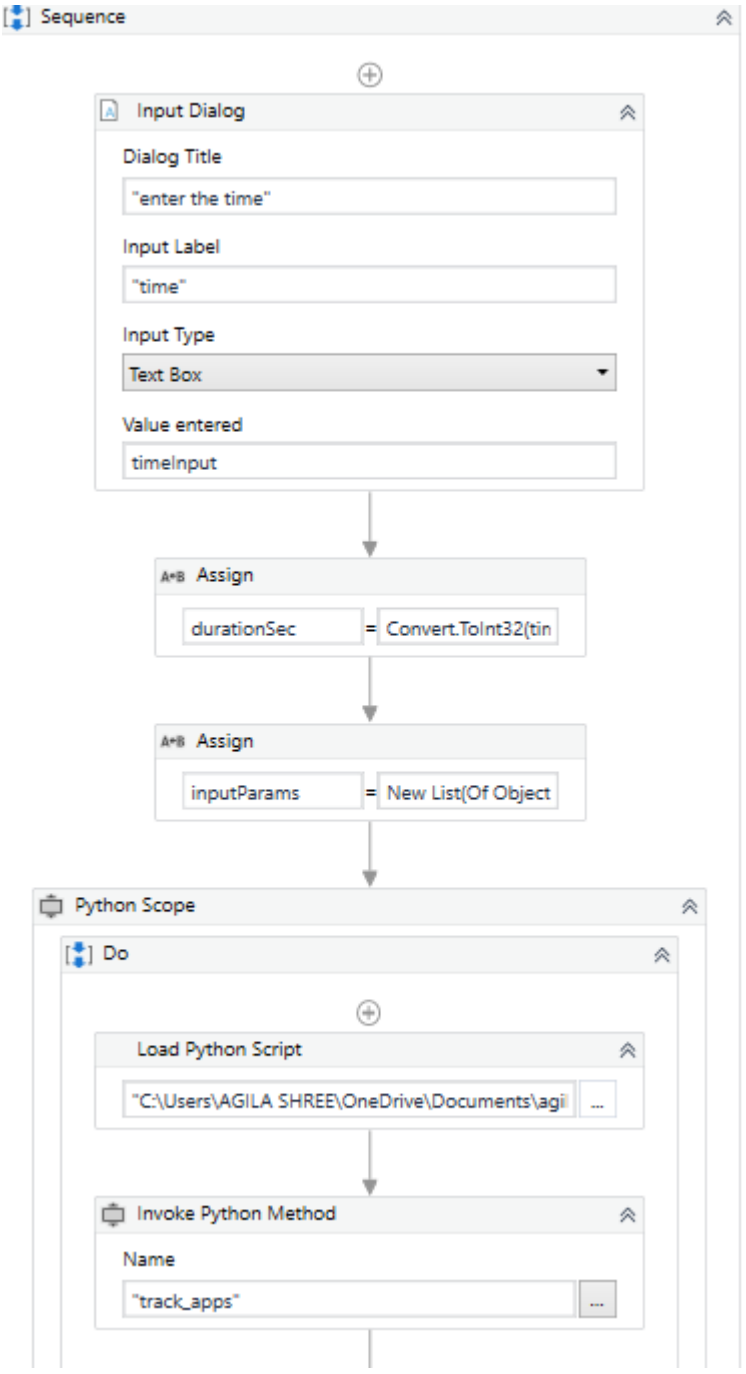
The Intelligent Screentime Monitoring Bot is an innovative solution aimed at helping users track and manage their screen usage effectively. By combining the power of UiPath automation with Python scripting, the bot provides real-time application usage tracking and delivers detailed statistics in a structured and user-friendly manner. This automation system empowers users to monitor their digital habits, identify excessive screen time, and take informed steps to improve their productivity.

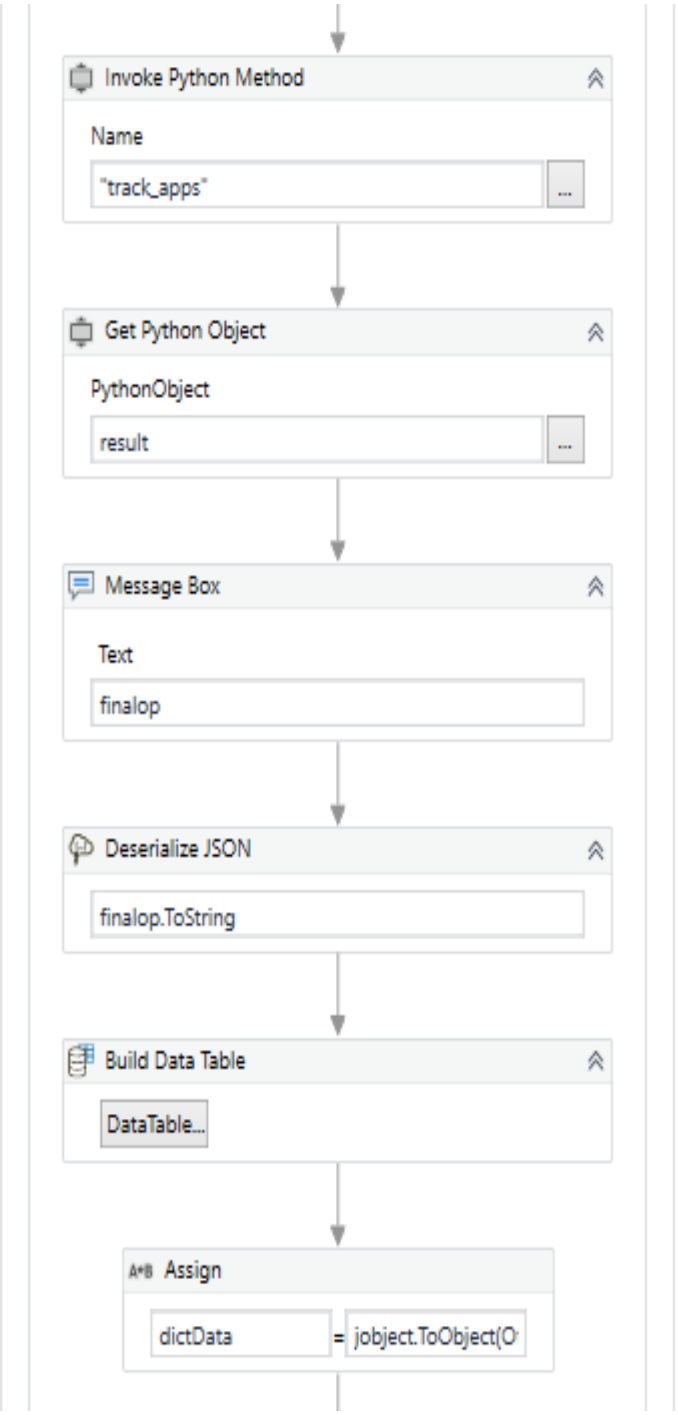
A key strength of this project is its dynamic and user-centric design. Through features like duration input, real-time data retrieval, and automated report generation, the bot ensures an intuitive user experience. The integration of a Python script for application tracking allows for precise data collection, while UiPath's capabilities enable the seamless transformation of this data into a well-organized Excel report.

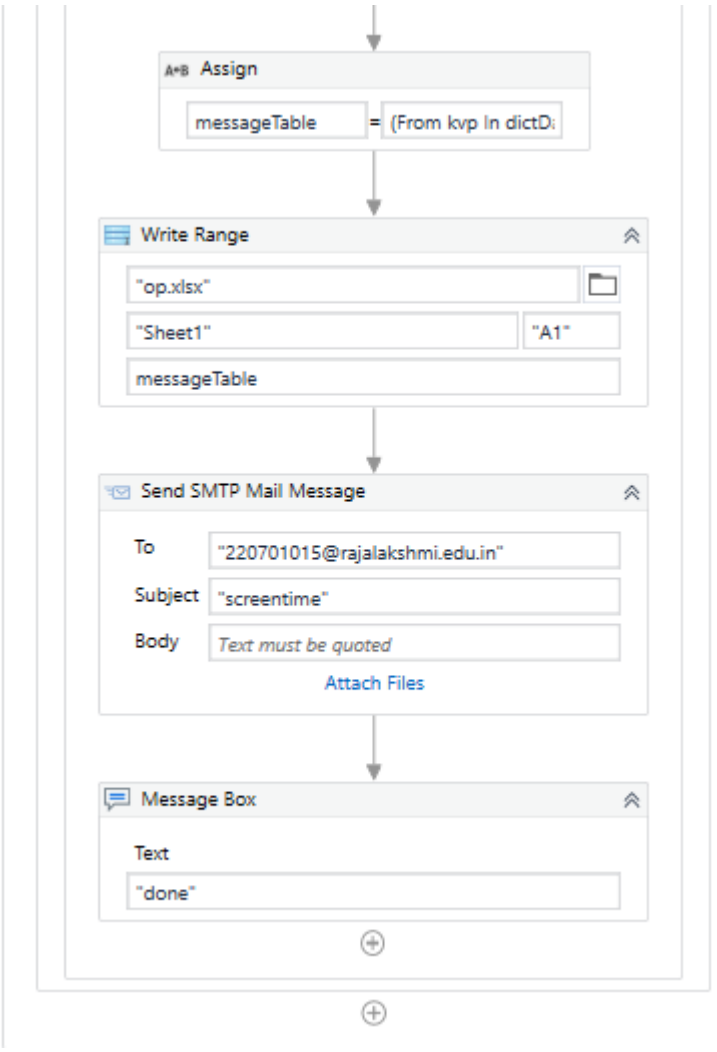
The bot's automated email dispatch functionality adds an additional layer of convenience for users. By delivering the generated Excel report directly to the user's inbox, it eliminates the need for manual intervention, ensuring that important insights are always accessible. This feature showcases the project's emphasis on improving efficiency, reducing time-consuming tasks, and making the system as user-friendly as possible. In conclusion, the Intelligent Screentime Monitoring Bot is a robust productivity tool that leverages the synergy between RPA and Python to deliver practical, real-world solutions. This project not only aids in better screentime management but also sets the foundation for future enhancements, such as advanced analytics or personalized recommendations, to further empower users in managing their digital lifestyles.

APPENDIX

PROCESS WORKFLOW




Invoke Python Method
Name
"track_apps"
...
Get Python Object
PythonObject
result
...
Message Box
Text
finalop
Deserialize JSON
finalop.ToString
Build Data Table
DataTable...
Assign
dictData
= jobject.ToObject(O


Assign
messageTable
=
(From kvp In dictD
Write Range
"op.xlsx"
"Sheet1"
"A1"
messageTable
Send SMTP Mail Message
To
"220701015@rajalakshmi.edu.in"
Subject
"screentime"
Body
Text must be quoted
Attach Files
Message Box
Text
"done"

```python
import time
import pygetwindow as gw
from collections import defaultdict
import json

app_usage_time = defaultdict(float)

def format_time(seconds):
    """Format the time into 'X minutes Y seconds' or 'X seconds'."""
    minutes = seconds // 60
    remaining_seconds = seconds % 60

    if minutes > 0:
        if remaining_seconds > 0:
            return f"{int(minutes)} minute(s) {int(remaining_seconds)} second(s)"
        else:
            return f"{int(minutes)} minute(s)"
    else:
        return f"{int(remaining_seconds)} second(s)"

def simplify_app_name(app_title):
    """Simplify app name to show the general app, not the specific window title."""
    if "YouTube" in app_title:
        return "YouTube"
    elif "Chrome" in app_title:
        return "Google Chrome"
    elif "Visual Studio Code" in app_title:
        return "Visual Studio Code"
    # Add more apps as needed here.
    else:
        return app_title.split(" - ")[-1]  # Simplify title by removing everything before ' - '

def track_apps(duration):
    previous_time = time.time()  # Initialize previous time at the start of tracking

    start_time = time.time()
    end_time = start_time + duration  # Duration is already in seconds

    while time.time() < end_time:
        current_window = gw.getActiveWindow()
        current_time = time.time()

        if current_window is not None:
            app_name = simplify_app_name(current_window.title)  # Get simplified app name
            app_usage_time[app_name] += current_time - previous_time

        previous_time = current_time
        time.sleep(1)  # Check every second

    # Format the usage time for each app
    app_usage_formatted = {app: format_time(seconds) for app, seconds in app_usage_time.items()}

    # Prepare the result as a JSON string (with human-readable time format)
    return json.dumps(app_usage_formatted)

```

The code is part of the "Load Script" concept within ARPA, where it is responsible for loading and executing the necessary scripts to manage and track user input, such as screen time limits or usage tracking. This ensures the system functions efficiently by processing the data in real time.

REFERENCES

[1] **Elnagar, M., & Ibrahim, M. (2020).** "Smart Screen Time Monitoring and Management System for Healthier Digital Habits." *International Journal of Computer Science and Information Security (IJCSIS), 18*(6), 89-94.

[2] **Vasquez, M., & Roberts, J. (2019).** "Screen Time: A Behavioral Analysis and Management System for Children and Teens." *Journal of Digital Health, 5*(3), 1-8.

[3] **Garg, V., & Vashistha, V. (2021).** "Development of a Smart Screen Time Tracker and Reminder System." *Proceedings of the 2021 International Conference on Artificial Intelligence and Machine Learning (ICAML), 105-110.*

[4] **Sarkar, A., & Bandyopadhyay, A. (2022).** "An IoT-Based Screen Time Management System for Health and Productivity Monitoring." *International Journal of Smart Systems and Applications, 12*(4), 1-10.

[5] **Chakraborty, P., & Roy, S. (2023).** "A Real-Time Screen Usage Monitoring Tool for Digital Well-being." *International Journal of Human-Computer Interaction, 39*(2), 217-225. DOI