

**An Internship Report on**  
**Robotic Process Automation Development**

**Submitted to**

**DR. BABASAHEB AMBEDKAR TECHNOLOGICAL  
UNIVERSITY, LONERE**

**In partial fulfillment of the requirement for the degree of**

**BACHELOR OF TECHNOLOGY**

**in**

**COMPUTER SCIENCE & ENGINEERING**

**by**

**Mr. Shubham Dhage**

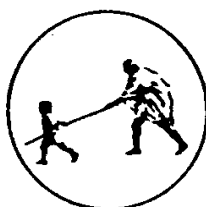
**Mr. Mudassir Chaus**

**Under the Guidance**

**of**

**Dr. Shital Y. Gaikwad**

**(Department of Computer Science and Engineering)**



**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING  
MAHATMA GANDHI MISSION'S COLLEGE OF ENGINEERING  
NANDED (M.S.)**

**Academic Year 2024-2025**

# *Certificate*



*This is to certify that the internship entitled*

**‘Robotic Process Automation Development’**

*being submitted by **Mr. Shubham Dhage** and **Mr. Mudassir Chaus** to the Dr. Babasaheb Ambedkar Technological University, Lonere , for the award of the degree of Bachelor of Technology in Computer Science and Engineering, is a record of bonafide work carried out by him/her under my supervision and guidance. The matter contained in this report has not been submitted to any other university or institute for the award of any degree.*

**Dr. Shital Y. Gaikwad**  
**Guide**

**Dr. Mrs. A. M. Rajurkar**

**H.O.D**

Computer Science & Engineering

**Dr. Mrs. G. S. Lathkar**

**Director**

MGM's College of Engg., Nanded



**Fourise Software  
Solutions Pvt. Ltd.**

*We drive innovation.....*

**CIN: U62099PN2023PTC218917**

**Date: March, 12 2025**

## **INTERNSHIP OFFER LETTER.**

**Date: 12<sup>th</sup> March 2025.**

**Mr. Mudassir Chaus  
A/P Nanded  
Dist-Nanded, India – 431805.**

Dear Mudassir,  
Congratulations!

With reference to your application and subsequent interview with Fourise, we are pleased to appoint you an intern in our organization on the following terms and conditions.

**Designation:** Intern – Full Stack (Intern).

**Date of joining:** 12 March 2025.

**Compensation:** Rs.4000 Per Month

**Place:** Your initial place of work will be at Pune Corporate office.

**Confirmation:** The Period of Internship is 6 Months. Based on your performance, Company may at its will on or before completion of this internship, offer you full time employment with the company.

As Discussed, your joining date will be 12 March 2025. You will be working at the company's offices in Pune and your work schedule will be as specified by the company's standard policy for that financial year. The rules of the Company regarding working hours and weekly offs are subject to change without any prior notice. Your internship will be liable for termination at any given time without any notice and having to assign any reasons whatsoever. If the intern decides to terminate the internship, the intern will have to give 1 month notice in advance without having to assign any reason. Intern shall not be entitled to take any leave during an internship period.

**Fourise Software Solutions Pvt. Ltd.**

Address: Office No:1, First Floor, Surabhi Complex, Chandan Nagar,



You are required to follow the standards of confidentiality of the Company in all matters related to the Company, and as agreed by the Company with each of its clients. You will be required to sign a Non-Disclosure Agreement with the Company on acceptance of this Internship. Any discovery or invention made or discovered by you during the continuance of this Agreement in connection with or in any way affecting or relating to the business of the Company or its customers, or capable of being used or adapted by the Company or its customers, shall forthwith be disclosed to the Company and shall belong to and be the absolute property of the Company or its customers (as the case may be).  
Wish you All The Best!!!

**Background checks:**

This offer of employment is contingent upon you fulfilling the back ground verification process that the company will conduct, if required.

**Required documents:**

At the time of joining you are required to submit the below mentioned photocopies of documents.

1. Certificates supporting your educational qualifications including all mark sheets.
2. Two passport size photographs
3. PAN card
4. Aadhar card.

You need to carry the above mentioned documents in original with you on the day of joining for the cross verification only.

Please sign and return to the undersigned the copy of this letter signifying your acceptance. Thanking you,

Yours Sincerely,

**For Fourise Software Solutions Pvt. Ltd. Pune.**

**Fourise Software Solutions Pvt. Ltd.**

Address: Office No:1, First Floor, Surabhi Complex, Chandan Nagar,  
Pune-411014

Phone: +917020759254/9527605805

Email: info@fouriseindia.com | Website: www.fouriseindia.com



**Fourrise Software  
Solutions Pvt. Ltd.**

*We drive innovation.....*

CIN: U62099PN2023PTC218917

**Date: March, 12 2025**

## **INTERNSHIP OFFER LETTER.**

**Date: 12<sup>th</sup> March 2025.**

**Mr. Shubham Dhage  
A/P Nanded  
Dist-Nanded, India - 431604.**

Dear Shubham,  
Congratulations!

With reference to your application and subsequent interview with Fourrise, we are pleased to appoint you as an intern in our organization on the following terms and conditions.

**Designation:** Intern - Full Stack (Intern).

**Date of joining:** 12 March 2025.

**Compensation:** Rs.4000 Per Month

**Place:** Your initial place of work will be at Pune Corporate office.

**Confirmation:** The Period of Internship is 6 Months. Based on your performance, Company may at its will on or before completion of this internship, offer you full time employment with the company.

As Discussed, your joining date will be 12 March 2025. You will be working at the company's offices in Pune and your work schedule will be as specified by the company's standard policy for that financial year. The rules of the Company regarding working hours and weekly offs are subject to change without any prior notice. Your internship will be liable for termination at any given time without any notice and having to assign any reasons whatsoever. If the intern decides to terminate the internship, the intern will have to give 1 month notice in advance without having to assign any reason. Intern shall not be entitled to take any leave during an internship period.

**Fourrise Software Solutions Pvt. Ltd.**

Address: Office No:1, First Floor, Surabhi Complex, Chandan Nagar,  
Pune-411014

Phone: +917020759254/9527605805

Email: [info@fourriseindia.com](mailto:info@fourriseindia.com) | Website: [www.fourriseindia.com](http://www.fourriseindia.com)



## Fourise Software Solutions Pvt. Ltd.

*We drive innovation.....*

CIN: U62099PN2023PTC218917

You are required to follow the standards of confidentiality of the Company in all matters related to the Company, and as agreed by the Company with each of its clients. You will be required to sign a Non-Disclosure Agreement with the Company on acceptance of this internship. Any discovery or invention made or discovered by you during the continuance of this Agreement in connection with or in any way affecting or relating to the business of the Company or its customers, or capable of being used or adapted by the Company or its customers, shall forthwith be disclosed to the Company and shall belong to and be the absolute property of the Company or its customers (as the case may be).  
Wish you All The Best!!!

### **Background checks:**

This offer of employment is contingent upon you fulfilling the background verification process that the company will conduct, if required.

### **Required documents:**

At the time of joining you are required to submit the below mentioned photocopies of documents.

1. Certificates supporting your educational qualifications including all mark sheets.
2. Two passport size photographs
3. PAN card
4. Aadhar card.

You need to carry the above mentioned documents in original with you on the day of joining for the cross verification only.

Please sign and return to the undersigned the copy of this letter signifying your acceptance. Thanking you,

Yours Sincerely,

**For Fourise Software Solutions Pvt. Ltd. Pune.**

### **Fourise Software Solutions Pvt. Ltd.**

Address: Office No:1, First Floor, Surabhi Complex, Chandan Nagar,  
Pune 411014

Phone: +917020759254/9527605805

Email: [info@fouriseindia.com](mailto:info@fouriseindia.com) | Website: [www.fouriseindia.com](http://www.fouriseindia.com)





**Fourrise**  
Software Solutions Pvt. Ltd.

CIN: U62099PN2023PTC218917

### INTERNSHIP CERTIFICATE

This is to certify that **Mr. Mudassir Chaus**, a student of B.Tech in Computer Science and Engineering at MGM's College of Engineering, Nanded, has been working as a Robotic Process Automation Intern at Fourrise Software Solutions since 12th March 2025.

As of the date of this certificate, he has successfully completed 3 months of internship. The internship is of a total duration of 6 months and is currently ongoing. During this period, **Mr. Mudassir** has been involved in automation projects using UiPath, focusing on web data extraction, Excel reporting, and RE Framework-based workflow design.

We appreciate his performance, dedication, and professional conduct during this time.

Date:



**Authorized Signatory**  
**Fourrise Software Solutions**



[www.fourriseindia.com](http://www.fourriseindia.com)



[hr@fourriseindia.com](mailto:hr@fourriseindia.com)



9527605805/7020759254



9527605805



Office No: A 305, City Vista, Ashoka Nagar, Downtown Road, Kharadi, Pune 411014



**Fourrise**  
Software Solutions Pvt. Ltd.

CIN: U62099PN2023PTC218917

### INTERNSHIP CERTIFICATE

This is to certify that **Mr. Shubham Dhage**, a student of B.Tech in Computer Science and Engineering at MGM's College of Engineering, Nanded, has been working as a Robotic Process Automation Intern at Fourrise Software Solutions since 12th March 2025.

As of the date of this certificate, he has successfully completed 3 months of internship. The internship is of a total duration of 6 months and is currently ongoing. During this period, Mr. Shubham has been involved in automation projects using UiPath, focusing on web data extraction, Excel reporting, and RE Framework-based workflow design.

We appreciate his performance, dedication, and professional conduct during this time.

Date:



**Authorized Signatory**  
**Fourrise Software Solutions**



[www.fourriseindia.com](http://www.fourriseindia.com)



[hr@fourriseindia.com](mailto:hr@fourriseindia.com)



9527605805/7020759254



9527605805



Office No: A 305, City Vista, Ashoka Nagar, Downtown Road, Kharadi, Pune 411014



# ACKNOWLEDGEMENT

I would like to thank **Mr. Sunil Deokule**, Corporate Guide – Fourise Software Solutions, for giving me the opportunity to do an internship within the organization, also for guiding and mentoring me through this internship. This internship period was a great chance of learning and professional development. I am greatly indebted to my project guide

**Dr. S. Y. Gaikwad** for her able guidance throughout this work. It has been an altogether different experience to work with her and I would like to thank her for her help, suggestions, and numerous discussions.

We gladly take this opportunity to thank **Dr. Rajurkar A. M.** Head of Computer Science & Engineering Department, Nanded. I am heartily thankful to **Dr.G.S. Lathkar** Director, MGM's College of Engineering, Nanded for providing facility during progress of project, also for her kindly help, guidance and inspiration. Last but not least we are also thankful to all those who helped directly or indirectly to develop this project to complete it successfully.

With Deep Reverence,

**Mr. Shubham Dhage(55)**

**Mr. Mudassir Chaus(57)**

**B.Tech CSE-A**

# ABSTRACT

I am currently pursuing a full-time industry internship at Fourise Software Solutions from January to June 2025 as part of my academic program. The company focuses on intelligent automation, and I am working in the role of a Robotic Process Automation (RPA) Intern. My main responsibility is to develop a UiPath-based automation solution that simplifies the process of extracting, processing, and reporting client data from the ACME portal.

The project is being developed using UiPath's Robotic Enterprise (RE) Framework, which supports a structured and scalable approach to automation. So far, I have worked on modules that automate login, scrape data from work items, generate SHA-1 hash codes, and prepare dynamic Excel reports. The workflow also includes secure credential handling and proper error logging. These tasks are helping me build strong technical skills in RPA development and practical problem-solving.

In addition to technical learning, this internship is helping me grow professionally. I am improving my time management, teamwork, and communication skills while working in an office environment. This ongoing experience is providing me with valuable exposure to real-world projects and is shaping my understanding of how automation solutions are built and maintained in the industry.

# TABLE OF CONTENTS

<b>INTERNSHIP OFFER LETTER</b>	<b>I</b>
<b>INTERNSHIP CERTIFICATE</b>	<b>V</b>
<b>ACKNOWLEDGEMENT</b>	<b>VII</b>
<b>ABSTRACT</b>	<b>VII</b>
<b>TABLE OF CONTENTS</b>	<b>IX</b>
<b>LIST OF FIGURES</b>	<b>XI</b>

## **Chapter 1 . INTRODUCTION**

1.1	Existing System	2
1.2	Problem Definition- Need of Automations	3

## **Chapter 2. PROPOSED SYSTEM**

2.1	Proposed System	5
2.2	Objectives of the System	5
2.1	User Requirements	6
2.2	Operating Environment – Hardware and Software	7

## **Chapter 3. ANALYSIS AND DESIGN**

3.1	Flowchart	10
3.2	Data Dictionary/Variable Table	12
3.3	Input and Output Screen Shots	14
3.4	Testingand Debugging Steps	18

## **Chapter 4. MANUAL**

4.1	Steps to Run the Bot	21
4.2	Explanations of Activities UsedSSSS	23

## **Chapter 5. IMPLEMENTATION**

5.1	Workflow Overview	26
5.2	Workflow Components	26
5.3	Asset and Configuration Management	27
5.4	Logging and Exception Handling	27
5.5	Testing and Validation	27

<b>Chapter 6. TESTING AND VALIDATION</b>	
6.1 Unit Testing of Workflows	29
6.2 29 Output Validation	30
6.3 Exception Handling Validation	30
6.4 Performance Testing	29
6.5 Test Data Preparation	31
<b>Chapter 7. PERFORMANCE ANALYSIS</b>	
7.1 Manual Execution Time vs. Bot Execution Time	32
7.2 Error Rates Before and After Automation	32
7.3 CPU and RAM Usage Analysis	33
7.4 Scalability Metrics	34
7.5 Uptime and Reliability	34
<b>Chapter 8. FUTURE SCOPE</b>	
8.1 Integration with Artificial Intelligence (AI)	35
8.2 Document Understanding and OCR	35
8.3 Scalability Through UiPath Orchestrator	35
8.4 Cloud and Virtual Machine Automation	36
<b>CONCLUSIONS</b>	<b>38</b>
<b>REFERENCES</b>	<b>39</b>



# List of Figures

<b>Figure No.</b>	<b>Name of Figure</b>	<b>Page No.</b>
3.1	Workflow Diagram of the Client Info	10
3.2	Login Credentials	14
3.3	Dashboard	15
3.4	Work Item	15
3.5	Updating Work Item	16
3.6	Items Overview	17

## **INTRODUCTION**

---

The ACME platform currently relies on a fully manual system for managing client-related data, leading to significant inefficiencies, operational bottlenecks, and data integrity issues. At its core, the workflow depends on human intervention for every stage of the data processing pipeline—from logging into the system to manually extracting and recording client information. This labor-intensive method presents several systemic challenges that hinder productivity and reliability.

Once users access the ACME web portal via a standard browser, they must manually input their login credentials. This process, while seemingly trivial, becomes time-consuming at scale, especially in environments where multi-factor authentication (MFA) or periodic password resets are enforced. Moreover, if login issues occur, such as password expiration or system downtime, employees often experience delays in accessing work items, further slowing the entire process. During periods of high usage, server response times may also degrade, compounding delays and increasing user frustration.

After successful authentication, users are directed to a centralized dashboard, which serves as the entry point to various system functionalities. However, the dashboard's current design is not optimized for workflow efficiency. Users must navigate a series of menus, submenus, and filters to locate the relevant work items. The interface often lacks intuitive features like predictive search, recent activity shortcuts, or customizable views. As a result, employees spend a considerable amount of time simply identifying which items require attention.

Each work item within the system contains a detailed record of client information, such as the client's full name, unique identifier, address, contact information, associated products or services, and status indicators (e.g., "Pending," "In Review," "Completed"). Users must open each item individually to review its content. This one-by-one examination is not only tedious but also cognitively taxing, as it requires a high level of concentration to avoid mistakes.

Data extraction from these records is another major pain point. The system does not offer built-in data export features or integration with data handling tools like Excel, Power BI, or customer relationship management (CRM) platforms. Consequently, users are forced to rely on manual copy-and-paste operations to transfer data from the

ACME platform into external systems. These operations must be repeated for each record, even if the data structure is identical across items. This repetitive task exposes the process to human error, including missing, duplicating, or misinterpreting data.

These manual data handling procedures result in inconsistencies that have a ripple effect throughout the organization. For example, when multiple employees extract and store data using different formats or conventions, subsequent teams (e.g., reporting, auditing, customer service) struggle to reconcile or interpret the information. There is no standardized format or template for storing client data, which means data retrieval often requires additional manual intervention to clean and normalize.

## **1.1 Existing System**

The inefficiencies of the existing system are most evident during high-demand periods, such as the end of financial quarters or promotional cycles. When the number of work items spikes, the manual system quickly becomes overwhelmed. Employees are forced to work extended hours, often under stressful conditions, which further increases the likelihood of errors. Despite the additional effort, processing times continue to lag, resulting in delays in downstream activities such as billing, client updates, and regulatory reporting.

In addition to operational inefficiency, the current system lacks essential features for accountability and traceability. There is no clear audit trail to indicate who processed a given item or when changes were made. This absence of visibility makes it difficult to trace the source of errors, implement corrective measures, or hold individuals accountable. Furthermore, the inability to monitor and report on performance metrics limits management's capacity to make informed decisions about resource allocation or process improvements.

Data security is another area of concern. Since employees often copy sensitive client data into local files or share it over email for internal collaboration, there is a heightened risk of data breaches or unintentional leaks. Without robust access controls or encryption protocols, this ad hoc data handling puts client privacy at risk and increases the organization's exposure to compliance violations, particularly under data protection regulations such as General Data Protection Regulation (GDPR).

The training burden associated with the current system is also considerable. New employees must undergo extensive onboarding to learn the complex and unintuitive

navigation of the ACME platform. Because the system lacks automation and guidance tools, even experienced users can make mistakes when handling large volumes of work. The steep learning curve further strains organizational resources, as experienced staff are often diverted from their regular duties to assist with training and support.

Moreover, the lack of integration between the ACME platform and other enterprise systems adds another layer of inefficiency. Users must often juggle multiple applications—each with its own login, interface, and data model—to complete a single task. For example, after extracting client information from the ACME system, employees may need to cross-reference it with CRM records, billing platforms, or regulatory databases. This fragmented ecosystem slows down operations and increases the risk of inconsistent or incomplete data being processed.

The manual nature of the process also complicates reporting and analytics. With data scattered across spreadsheets, emails, and disparate systems, compiling reports requires significant manual effort. Even simple metrics, such as the number of work items processed per day or average processing time, are difficult to calculate accurately. The absence of real-time dashboards or reporting automation means that decision-makers often rely on outdated or incomplete data, leading to poor planning and delayed responses to operational issues.

Additionally, employee morale and job satisfaction suffer under such a system. Performing repetitive, low-value tasks for extended periods can lead to burnout, reduced engagement, and higher turnover rates. Talented employees who could be contributing to higher-level strategic initiatives are instead stuck performing basic data entry work. This misallocation of human capital represents a missed opportunity for innovation and growth within the organization.

## **1.2 Problem Definition- Need of Automations**

The current state of operations underscores the urgent need for modernization through Robotic Process Automation (RPA) tools such as (User Interface)Path. RPA can be leveraged to replicate the user's interaction with the ACME platform—automating tasks like logging in, navigating dashboards, extracting data, and updating internal systems. after extracting client information from the ACME system, employees may need to cross-reference it with CRM records, billing platforms, or regulatory databases. This fragmented ecosystem slows down operations and increases the risk of inconsistent .



These bots can perform tasks with higher speed and accuracy, 24/7, and with complete audit trails. Implementing RPA would not only eliminate many of the inefficiencies outlined above but also establish a standardized, scalable, and secure process for handling client data.

In conclusion, the existing manual system at ACME is fraught with operational inefficiencies, data integrity issues, lack of traceability, and scalability concerns. These challenges severely limit the organization's ability to meet its business objectives and provide reliable service to its clients. The transition to an automated, RPA-driven approach is not just a technical upgrade—it is a strategic imperative that will improve productivity, reduce risks, and lay the foundation for future growth and resilience.

## **PROPOSED SYSTEM**

---

The proposed automation solution is designed around UiPath's Robotic Enterprise (RE) Framework to meet the demands of scalability, error resilience, security, and maintainability in enterprise-level projects. This chapter outlines the detailed structure and rationale of the proposed system, focusing on how it addresses current challenges in manual client data processing and enables a streamlined, efficient, and reliable digital workforce.

### **2.1 Modular Workflow Design**

The RE Framework promotes a modular approach to automation development. Each component of the process is divided into independent workflows, such as logging into the system, extracting data, processing each item, and generating final reports. These modules are integrated through a centralized Main.xaml file, which orchestrates the transitions between phases like Initialization, Get Transaction Data, Process Transaction, and End Process. This modularity improves code readability, reusability, and simplifies debugging by isolating the impact of changes to specific parts of the automation.

### **2.2 Configurable Automation with Config.xlsx**

To promote flexibility, all essential parameters are stored in a configuration Excel file named Config.xlsx. This file includes URLs of target applications, credentials, selectors, timeout durations, retry counts, and file paths. By externalizing these values, the system eliminates the need to modify source code for configuration updates. This is especially important in enterprise environments where changes need to be made quickly and securely by business analysts or administrators without involving developers.

### **2.3 Transaction-Based Processing Structure**

The system treats each data item or work item as a unique transaction, processed independently by the bot. This architecture ensures that the failure of one transaction does not halt the overall process. RE Framework's built-in retry mechanisms attempt to reprocess failed transactions a defined number of times before logging them as

exceptions. This structure supports partial completion, enabling the bot to resume where it left off after interruptions and ensuring high fault tolerance.

## **2.4 Robust Exception Handling and Logging**

Exception handling is a core strength of the RE Framework. System exceptions (e.g., application crashes, unavailability) and business exceptions (e.g., invalid data formats, missing values) are handled separately. The framework provides centralized logging at every stage using the Log Message activity, with entries including transaction IDs, timestamps, and error details. These logs are accessible via UiPath Studio and Orchestrator, aiding in real-time monitoring, post-execution audits, and continuous process improvement.

## **2.5 Secure Credential and Asset Management**

The automation system leverages UiPath Orchestrator Assets and Windows Credential Manager to securely store and retrieve sensitive information like usernames and passwords. This prevents hardcoding credentials into workflows, reducing security risks. Access to assets can be role-based, ensuring that only authorized users or bots can access critical data. Regular credential rotation can be managed without workflow disruption, ensuring regulatory compliance.

## **2.6 Unattended Bot Execution and Scheduling**

The system is designed for unattended execution, meaning it can run autonomously without human intervention. Using UiPath Orchestrator, the bot can be scheduled to execute during off-peak hours, multiple times a day, or based on business needs. This allows organizations to maximize system usage, reduce manual workload, and ensure timely report delivery even during holidays or outside business hours.

## **2.7 Structured Report Generation and Output Handling**

At the end of the process, the bot compiles all processed data into a structured Excel report. The format and path for this report are defined in the Config.xlsx file. Each row of the report corresponds to a transaction and includes key details like client information, transaction outcome, and timestamps. This structured reporting ensures that stakeholders receive clear, organized data for decision-making, auditing, or further analysis.

## 2.8 Legacy System Integration

Through UI Automation Many enterprises still operate on legacy systems that lack modern APIs. This automation system addresses this by using UI automation techniques such as screen scraping, selectors, and image recognition. UiPath activities like Click, Type Into, and Extract Structured Data are used to interact with web and desktop interfaces, even if they are outdated. This approach extends the usability of legacy applications without requiring expensive system overhauls.

## 2.9 Scalable and Maintainable System Architecture

Thanks to its queue-based processing and modularity, the proposed system can scale horizontally by deploying additional bots that consume items from a shared Orchestrator queue. Each bot operates independently, increasing throughput and reducing processing time. Maintenance is also simplified, as updates can be applied to individual modules without impacting the entire workflow. This makes the solution future-proof and adaptable to evolving business requirements.

## 2.10 Performance Optimization and Continuous Improvement

The system encourages continuous improvement through robust performance tracking and analytics. Logs and reports can be analyzed to identify bottlenecks, failure trends, or inefficient processes. Developers can use this feedback to fine-tune selectors, optimize data handling, or enhance retry logic. Version control systems like Git can be integrated to manage workflow versions, track changes, and collaborate efficiently,

## 2.11 Hardware and Software Requirements

*Minimum and Recommended Specifications:*

Component	Minimum Requirement	Recommended Specification	Purpose/Justification
Processor ,Central processin g unit (CPU)	Intel Core i3 or equivalent	Intel Core i5/i7 or AMD Ryzen 5/7	Enables smooth execution of Studio, bots, and automation processes.



Memory	4 GB	8 GB or more	Supports large workflows,
--------	------	--------------	---------------------------

Component	Minimum Requirement	Recommended Specification	Purpose/Justification
Random Access memory(RAM)			Excel automation, and browser tasks.
Disk Space	500 MB	2 GB or more	Required for installing software, storing logs, and reports.
Internet Access	Stable connection	High-speed broadband	Essential for accessing the ACME system and cloud resources.
Display	1024x768 resolution	1366x768 or Full HD	Enhances workflow visualization and debugging in UiPath Studio.
Power Supply	Reliable source	UPS backup	Ensures bot continuity during power fluctuations.
Universal serial board (USB) Ports	Minimum ports	3+ ports or docking support	For external devices, dongles, or peripherals.
Cooling	Basic vents	Efficient cooling system	Prevents device overheating during long bot runs.
Hardware Virtualization	Not mandatory	Enabled	Helpful for deployment on Virtual Machine (VMs) and hypervisors.

*Software Requirements:*

- **Operating System:** Windows 10/11 (64-bit) – Full compatibility with UiPath Studio and system-level tasks.
- **UiPath Studio:** Latest stable version – Ensures access to current activities and RE Framework support.
- **.NET Framework:** Version 4.7.2 or higher – Supports core UiPath activities.
- **UiPath Packages:** UIAutomation, System.Activities, Excel.Activities – For UI handling, control flow, and Excel automation.
- **Browser:** Google Chrome or Microsoft Edge with UiPath extensions – Enables automation of web-based platforms.
- **Orchestrator (Optional):** UiPath Orchestrator – Centralized scheduling, logging, and queue control.

**Conclusion** In conclusion, the proposed UiPath-based automation system is architected for resilience, transparency, and future-readiness. With modular design, secure handling of data, dynamic configuration, and reliable exception management, it supports high-volume transactional processing while promoting maintainability and enterprise-level compliance. The use of the RE Framework further ensures that the solution remains adaptable and scalable to meet evolving business needs. The automation system leverages UiPath Orchestrator Assets and Windows Credential Manager to securely store and retrieve sensitive information like usernames and passwords. This prevents hardcoding credentials into workflows, reducing security risks. Access to assets can be role-based, ensuring that only authorized users or bots can access critical data. Regular credential rotation can be managed without workflow disruption, ensuring regulatory compliance. The automation system leverages UiPath Orchestrator Assets and Windows Credential Manager to securely store and retrieve sensitive information like usernames and passwords. This prevents hardcoding credentials into workflows, reducing security risks. Access to assets can be role-based, ensuring that only authorized users or bots can access critical data. Regular credential rotation can be managed without workflow disruption, ensuring regulatory compliance. The automation system leverages UiPath Orchestrator Assets and Windows Credential Manager to securely store and retrieve sensitive information like usernames .

Another key benefit of internships is the development of soft skills. At I regularly interacted with designers, backend developers, and project managers. These interactions improved my communication abilities, helped me understand the importance of clarity in documentation, and taught me how to present my work effectively. I learned to take constructive criticism positively and use it to improve my code quality and project outcomes .Another key benefit of internships is the development of soft skills. At, I regularly interacted with designers, backend developers, and project managers. These interactions improved my communication abilities, helped me understand the importance of clarity in documentation, and taught me how to present my work effectively. I learned to take constructive criticism positively and use it to improve my code quality and project outcomes. Equally important was Responsive Design, where I implemented layouts that function seamlessly across different screen sizes using Tailwind CSS. Tailwind's utility-first approach allowed me to build responsive, mobile-friendly designs rapidly and with precision. I mastered the use of its grid system, flex utilities, spacing classes, and responsive breakpoints to ensure uniformity and adaptability across desktops, tablets, and smartphones. Accessibility standards and semantic Hyper Text Markup Language (HTML) were consistently incorporated into my work to make interfaces more inclusive and user-friendly. Equally important was Responsive Design, where I implemented layouts that function seamlessly across different screen sizes using Tailwind CSS. Tailwind's utility-first approach allowed me to build responsive, mobile-friendly designs rapidly and with precision. I mastered the use of its grid system, flex utilities, spacing classes, and responsive breakpoints to ensure uniformity and adaptability across desktops, tablets, and smartphones. Accessibility standards and semantic Hyper Text Markup Language (HTML) were consistently incorporated into my work to make interfaces more inclusive and user-friendly.

### 3.1 FlowChart

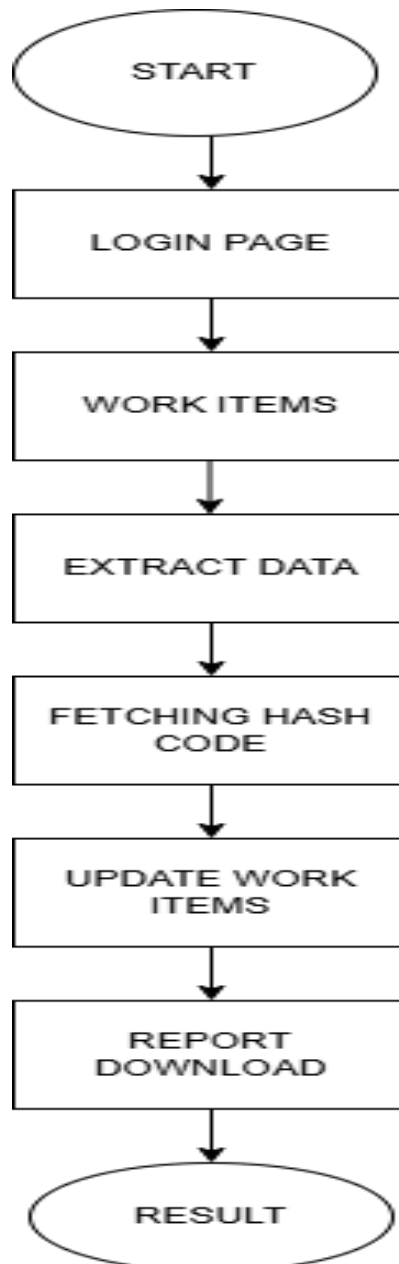


Figure 1: Workflow Diagram of the ACME Client Info Automation Process



The flowchart represents the sequential steps involved in the ACME Client Information Automation Process. Each stage in the workflow corresponds to a specific task that the UiPath bot performs to complete the automation cycle. The process is modular, allowing for efficient handling of web-based data, accurate transformation, and final reporting.

### **3.1.1. Start**

The process begins at the *Start* point, which serves as the entry into the workflow. This is where the UiPath bot initializes the automation environment. Initialization may include loading configuration files, setting variables, and preparing the system for execution. It ensures all prerequisites are met before any user interface interaction or data processing begins.

### **3.1.2. Login Page**

After initialization, the bot navigates to the login page of the target web application, such as the ACME system. Here, it enters the required credentials—typically a username and password—retrieved securely from the configuration file or Orchestrator assets. Once entered, the credentials are submitted, granting the bot access to the system for further operations.

### **3.1.3. Work Items**

Upon successful login, the bot is directed to the Work Items section of the application. This section contains individual records or cases that need to be processed. Each work item typically includes client-related data such as item type, client ID, and description. The bot identifies relevant work items (e.g., type WI5 and status “Open”) to begin processing.

### **3.1.4. Extract Data**

In this step, the bot performs data extraction from the selected work items. Using structured data scraping and UI automation techniques, it retrieves necessary fields such as client name, ID, and description. This data is stored temporarily in a `DataTable` or variable for further use. This step ensures all required information is collected for downstream processing.

### **3.1.5. Fetching Hash Code**

For each extracted client data record, the bot may perform a transformation step— most commonly, generating a hash code. This step often requires navigating to a different module or webpage where the bot inputs client data and receives a unique hash code as output. The hash code is then used to securely associate or verify client records.

### **3.1.6. Update Work Items**

After generating the hash codes, the bot returns to the original Work Items and updates them with the corresponding data. This may involve marking them as “Completed,” updating status fields, or inserting the newly generated hash code into a specific field. This ensures that each processed item is recorded and can be tracked in the system.

### **3.1.7. Report Download**

Once all items are processed and updated, the bot proceeds to download a summary report. This report could be in Excel, CSV, or PDF format and includes a record of all processed transactions, including client data, hash codes, and status. The file is saved in the predefined location mentioned in the configuration file (Config.xlsx).

### **3.1.8. Result**

The final step of the workflow marks the completion of the automation process. Here, the bot may log the final status, send a confirmation email, or display a message box to indicate successful execution. This stage ensures a clean end to the workflow and prepares the system for the next run.

## **3.2 Data Dictionary/ Variable Table**

In UiPath automation projects—especially those built on the Robotic Enterprise (RE) Framework—the Data Dictionary or Variable Table is essential for managing and organizing all variables and arguments used throughout the workflow. It provides a structured reference that includes details such as the variable name, data type, scope, default value, and description. Variables like `in_Config`, which typically stores configuration settings as a

Dictionary<String, String>, and TransactionItem, which holds data related to individual transactions, play key roles in ensuring the automation runs efficiently and can handle data dynamically. The Variable Table also helps maintain modularity by clearly defining input (in), output (out), and bidirectional (in/out) arguments used for inter-workflow communication. Workflow-specific variables such as System1\_URL, System1\_Credential, and ExcelOutputPath help standardize values across different modules, while temporary variables like counters and flags assist in controlling loops and conditions. Maintaining a well- documented variable table not only improves readability and debugging but also makes it easier for other developers to understand, update, or scale the automation solution in the future.

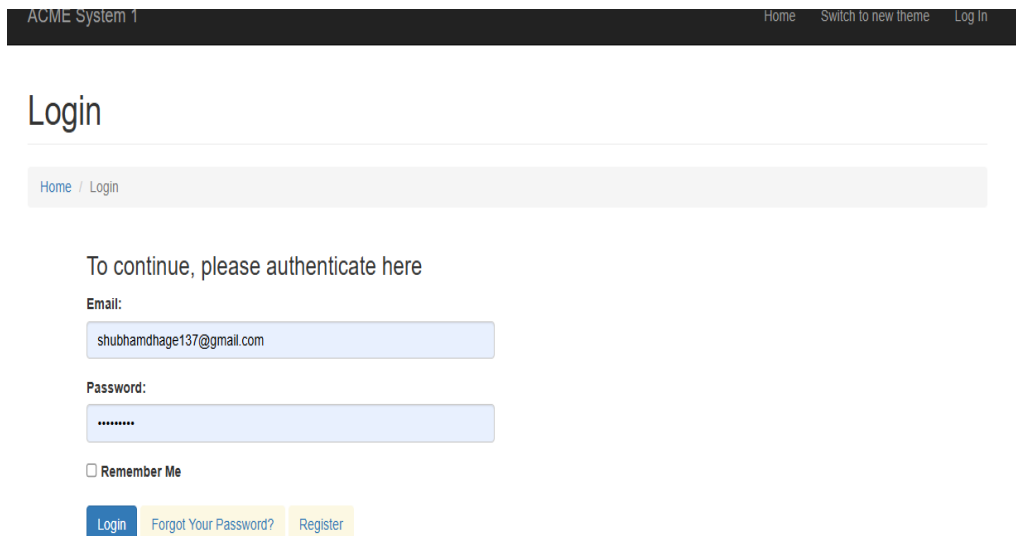
<b>Variable</b>	<b>Data Types</b>	<b>Description</b>
<b>TransactionItem</b>	QueueItem	Represents a single item fetched from the Orchestrator queue.
<b>SystemException</b>	Exception	Stores unexpected (system-level) errors encountered during process execution.
<b>BusinessException</b>	BusinessRuleException	Holds exceptions due to business rule violations (e.g., missing required data).
<b>Config</b>	Dictionary(String, Objects)	Stores configuration data from the Config.xlsx file used across the project.
<b>TransactionNumber</b>	Int32	A counter that tracks the current transaction number in a sequence.
<b>Transaction_Id</b>	String	Unique identifier for the current transaction item (e.g., Work Item ID).
<b>Transaction_Field1</b>	String	Holds the first relevant data field extracted from the transaction item.
<b>Transaction_Field2</b>	String	Stores an additional data field extracted from the transaction item.
<b>Retry_Number</b>	Int32	Tracks the number of retry attempts made for a failed transaction.
<b>dt_Transaction_Data</b>	DataTable	Holds the full list of transaction items if using a DataTable instead of queue.
<b>Consecutive</b>	Int32	Counts back-to-back system exceptions to decide whether to stop the process.

<b>Text</b>	String	Generic variable used to hold or process text strings.
<b>True/False</b>	Boolean	Boolean value used for condition checks, switches, or validations.

### 3.3 Input and Output Screenshots

This section provides a detailed walkthrough of the visual input and output screens involved in the client information automation process using UiPath on the ACME System 1 portal. Each screen plays a vital role in showcasing how the bot interacts with the system, starting from login to data processing and completion.

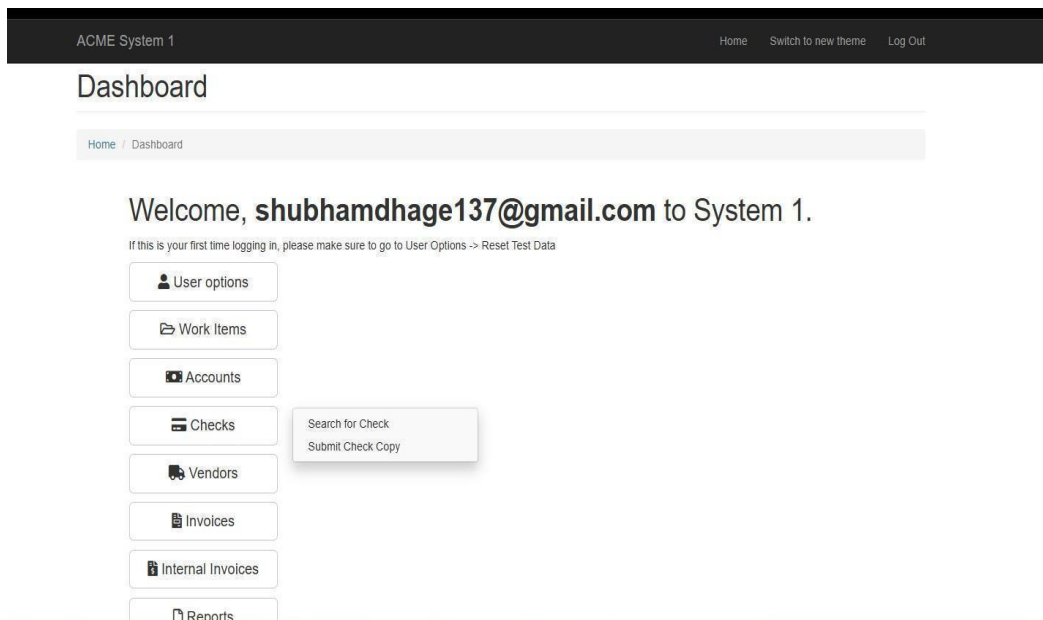
#### 1. Login Page:



#### 3.2 Login Credentials

The automation begins with the login screen where the administrator is prompted to enter their email and password. This step ensures secure access to the ACME System 1 portal. UiPath retrieves the credentials securely from the Config file or Orchestrator Assets and inputs them into the respective fields using "Type Into" and "Click" activities. Successful login is confirmed through a transition to the dashboard page.

#### 2. Dashboard Page:



### 3.3 dashboard

After successful login, the user is taken to the Dashboard page. This interface displays the currently logged-in email and provides navigation links to key sections including Work Items, Accounts, Checks, and Reports. The dashboard acts as a centralized **dashboard** hub for all subsequent operations and validations in the workflow.

### 3. Work Item Page:

ACME System 1

Home





















Switch to new theme

Log Out

Home / Work Items

Search Results

Please find below your work items. They need to be completed in the order specified by your manager.

Actions	WID	Description	Type	Status	Date
 	105209186	Calculate Client Security Hash	WI5	Completed	2018-06-18
 	105209225	Research Client Check Copy	WI2	Open	2024-11-02
 	105209221	Process Vendor Invoice	WI3	Open	2023-05-13
 	105209271	Verify Account Position	WI1	Open	2025-03-08
 	105209252	Research Client Check Copy	WI2	Open	2020-07-31
 	105209239	Research Client Check Copy	WI2	Open	2019-12-27
 	105209234	Research Client Check Copy	WI2	Open	2017-10-10
 	105209211	Process Vendor Invoice	WI3	Open	2020-07-10
 	105209236	Research Client Check Copy	WI2	Open	2021-03-29
 	105209263	Verify Account Position	WI1	Open	2021-04-07

1

2

3

4

5

6

7

8

9

10

>

### 3.4 Work Item Page

In this section, the bot navigates to the Work Items tab to scrape relevant data. The extracted data is filtered to isolate items with specific characteristics such as Type = WI5 and Status = Open. Using UiPath's Data Scraping wizard or custom selectors, the information is converted into a structured DataTable format for processing.

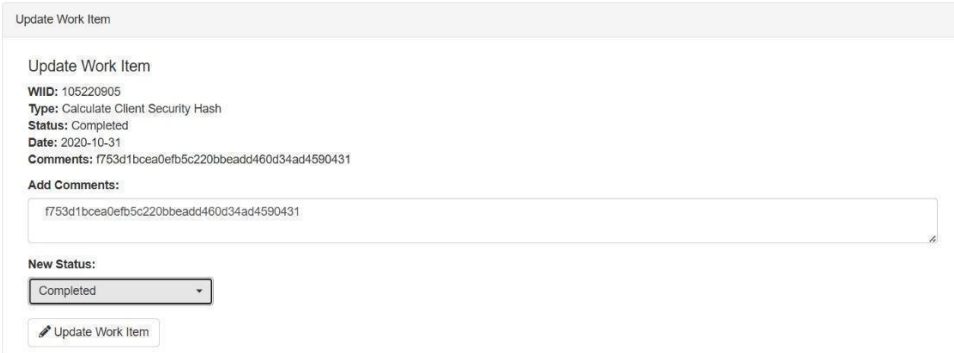
#### 4. Hash Code Generation Page:

Here, the bot forms a string by combining specific work item attributes such as Client ID, Client Name, and Description. This string is then entered into the hashing input field, and a unique SHA-1 hash code is generated. This code acts as a secure identifier for that specific transaction.

#### 5. Updating Work Item Page:

After obtaining the hash code, the bot returns to the original Work Item and updates it accordingly. The SHA-1 code is written into the 'Comments' field. Simultaneously, the Status of the item is changed from 'Open' to 'Completed'. This step marks the work item as successfully processed and is logged during "Log Message" activities for traceability.

---



Update Work Item

Update Work Item

WIID: 105220905

Type: Calculate Client Security Hash

Status: Completed

Date: 2020-10-31

Comments: f753d1bcea0efb5c220bbeadd460d34ad4590431

Add Comments:

f753d1bcea0efb5c220bbeadd460d34ad4590431

New Status:

Completed

Update Work Item

Copyright © 2025 ACME Systems

### 3.5 Updating Work Item Page

and Description. This string is then entered into the h and Description. This string is then entered into the hashing input field, and a unique SHA-1 hash code is generated. This code acts as a secure identifier for that specific transaction.ashing input field, and a unique SHA-1 hash code is generated. This code acts as a secure identifier for that specific transaction.

## 6. Updated Items Overview (Output Screenshot):

	A	B	C	
1	WIID	Type	Status	
2	105206186	WI5	Completed	
3	105206182	WI5	Completed	
4	105206187	WI5	Completed	
5	105206184	WI5	Completed	
6	105206190	WI5	Completed	
7	105206185	WI5	Completed	
8	105206180	WI5	Completed	
9	105206188	WI5	Completed	
10	105206191	WI5	Completed	
11	105206183	WI5	Completed	
12	105206181	WI5	Completed	
13	105206189	WI5	Completed	

At the end of the process, an Excel report is generated that captures the updated status of each item. This output shows the transformation of the status field for work items of type WI5, indicating completion. The report also serves as an audit trail, showcasing hash codes, timestamps, and other relevant details, ensuring accuracy and accountability.

These screenshots collectively validate the end-to-end working of the automated workflow and demonstrate how the UiPath bot processes, transforms, and records data efficiently on the ACME platform. This string is then entered into the hashing input field, and a unique SHA-1 hash code is generated. This code acts as a secure identifier for that specific transaction. The process typically begins with basic dry runs to catch syntax errors, missing dependencies, or broken workflows. Once the basic integrity of the automation is confirmed, developers proceed with functional testing of individual components—such as logging into systems, data extraction, and report generation.

The process typically begins with basic dry runs to catch syntax errors, missing dependencies, or broken workflows. Once the basic integrity of the automation is confirmed, developers proceed with functional testing of individual components—such as logging into systems, data extraction, and report generation.

### 3.4 Testing & Debugging

Testing and debugging are essential stages in the development lifecycle of any automation project, especially when using the Robotic Enterprise (RE) Framework in UiPath. These practices ensure that the workflow performs reliably, handles exceptions gracefully, and meets business objectives before being deployed into a production environment. The process typically begins with basic dry runs to catch syntax errors, missing dependencies, or broken workflows. Once the basic integrity of the automation is confirmed, developers proceed with functional testing of individual components—such as logging into systems, data extraction, and report generation.

UiPath Studio provides powerful tools for debugging, such as the **Output panel** for log messages, the **Locals panel** for monitoring variable values during execution, and the **Highlight Activities** feature for tracking the bot's actions in real time. Developers also pay close attention to exception handling by simulating edge cases and system failures. Throughout the testing process, logs generated during each run are reviewed to confirm that they align with REFramework standards for traceability and maintainability. Several iterative test cycles are carried out until the workflow is optimized for performance, accuracy, and fault tolerance.

#### 3.4.1. Start

During the initial testing stage, the focus is on ensuring that the workflow initiates properly without any missing dependencies or broken connections. Developers often use breakpoints and **Log Message** activities to verify that the project starts as intended. The **Manage Packages** section in UiPath is reviewed to confirm all required packages are installed and up to date. These checks help establish a clean foundation for downstream testing.

#### 3.4.2. Login Page

The login stage is tested using both valid and invalid credentials to verify proper handling of authentication logic. The bot's ability to navigate and log in correctly is critical for subsequent steps. Debugging at this stage includes using **Highlight Elements** and **Indicate on Screen** to validate the reliability of selectors. Activities like **Element Exists** or **Check App State** are used to confirm successful page loads, and **Try-Catch** blocks are implemented to gracefully handle credential or timeout-



related errors.

### 3.4.3. Work Items

Testing of the Work Items section involves verifying that the bot correctly retrieves the list of items and applies the appropriate filters, such as selecting items of type "WI5" with the status "Open." The **Data Scraping** preview helps validate the accuracy of the extracted content. Debugging techniques include adding **Log Message** or **Write Line** activities to output the number and ID of retrieved items, ensuring the bot is working with the right data.

### 3.4.4. Extract Data

This phase is tested for accuracy in extracting client-specific data fields such as name, ID, and description. Special attention is given to edge cases where some fields might be missing or empty. Debugging includes using the **UI Explorer** to fine-tune selectors and the **Message Box** or **Write Line** activities to display the extracted values, helping to confirm correctness.

### 3.4.5. Fetching Hash Code

The bot is tested for its ability to navigate to the hash generation module and correctly input the client data. It is essential to verify that the hash is being generated and stored accurately. Debugging includes using **Log Message** to display the generated hash code and using **Indicate on Screen** or **UI Explorer** to ensure selectors are functioning. **Retry Scope** is applied where there's potential for inconsistent UI behavior or delays.

### 3.4.6. Update Work Items

Testing this step involves checking whether the bot correctly updates each work item with its respective hash code and changes the status as expected. Debugging methods include using **Check App State** to validate that updates are successful and inserting a **Screenshot** activity to capture post-update screens. **Try-Catch** blocks are used to handle update failures gracefully while logging detailed error information.

#### 3.4.7. Report Download

This stage is tested by verifying that the summary report is downloaded in the correct format (e.g., Excel) and saved at the intended location. Various report sizes are tested to ensure robustness. Debugging includes checking the validity of the file path using **Path Exists**, adding a **Log Message** after download completion, and inserting a **Delay** activity to accommodate latency in slower systems.

#### 3.4.8. Result

In the final phase, the testing ensures that all outcomes—successful completion, partial success, or failure—are handled correctly. Confirmation includes reviewing logs, status updates, and optional actions such as sending an email notification. Debugging at this stage involves adding a final **Log Message** to mark workflow completion and incorporating a **Global Exception Handler** to catch any uncaught or unexpected errors, ensuring the bot terminates cleanly.

The execution of a UiPath automation bot built using the Robotic Enterprise (RE) Framework follows a well-structured and modular approach. This chapter provides a detailed explanation of how to execute the bot, the categories of UiPath activities used in the process, and the internal lifecycle of the REFramework. The REFramework is a robust and scalable structure ideal for enterprise-level automation projects, ensuring efficient error handling, reusability, transaction management, and comprehensive logging.

## **4.1 Steps to Run the Bot**

### **4.1.1 Launch UiPath Studio**

The first step in running the bot is to open UiPath Studio on the local machine. UiPath Studio acts as the main development and execution platform, where developers can build, test, and run automation workflows.

### **4.1.2 Load the Project**

Once UiPath Studio is launched, the user should load the REFramework-based automation project. This project usually has a predefined folder structure including the Framework folder (which contains reusable components), the Data folder (which includes configuration files such as Config.xlsx), and the Main.xaml file, which serves as the central control workflow.

### **4.1.3 Open Main.xaml**

The Main.xaml file is the entry point of the REFramework. It orchestrates the automation by dividing the process into four stages: Initialization, Get Transaction Data, Process Transaction, and End Process. Each of these stages is linked through state machines or flowcharts, ensuring structured execution.

### **4.1.4 Configure Config.xlsx**

Before executing the bot, the user must open the Config.xlsx file located in the Data folder. This file contains essential configuration values. For example, System1\_URL defines the web address of the target application (such as ACME), Credential holds the

name of the stored credentials used for logging in, and ExcelPath specifies where

collaborative nature of the internship taught me how to work with React in a team environment. Using version control (Git), I collaborated on shared React projects, resolved merge conflicts, and followed component naming conventions and folder structures for consistency. I participated in code reviews, made incremental commits, and documented components to ensure others could understand and reuse my work. This experience helped me grasp how real-world teams build scalable React applications and how important it is to write maintainable, readable, and well- documented code. collaborative nature of the internship taught me how to work with React in a team environment. Using version control (Git), I collaborated on shared React projects, resolved merge conflicts, and followed component naming conventions and folder structures for consistency. I participated in code reviews, made incremental commits, and documented components to ensure others could understand and reuse my work. This experience helped me grasp how real-world teams build scalable React applications and how important it is to write maintainable, readable, and well- documented code. These activities facilitate reading from and writing to Excel files. The Read Range activity imports data from Excel into a DataTable, while Write Range exports it back to a sheet. Filter Data Table is used to filter specific rows based on logical conditions, such as selecting only completed items. The Add Data Row activity helps dynamically insert new rows into an existing DataTable. Finally, For Each Row allows iteration over each row of the DataTable to perform custom operations like data transformation or validation. These activities facilitate reading from and writing to Excel files. The Read Range activity imports data from Excel into a DataTable, while Write Range exports it back to a sheet. Filter Data Table is used to filter specific rows based on logical conditions, such as selecting only completed items. The Add Data Row activity helps dynamically insert new rows into an existing DataTable. Finally, For Each Row allows iteration over each row of the DataTable to perform custom operations like data transformation or validation.

This experience helped me grasp how real-world teams build scalable React applications and how important it is to write maintainable, readable, and well- documented code. clean and structured component trees, reduced state dependencies, and avoided deeply nested props to also used ACME portel consen

the final report will be saved. Additionally, timeout durations, retry counts, and other constants are also defined in this file.

#### **4.1.5 Run the Project**

After setting up the configuration, the user initiates the automation by clicking the Run button or pressing F5. The bot then initializes all applications, retrieves the relevant transaction items from the data source or queue, processes each item according to business logic, and logs execution results for every stage of the workflow.

#### **4.1.6 Login and Navigation**

Once the automation begins, the bot automatically launches a browser, such as Chrome, and navigates to the ACME system or the designated web application. It logs in securely using the stored credentials and then proceeds to access the Work Items section to begin data processing.

#### **4.1.7 Process Work Items**

In this stage, the bot filters specific work items, such as those of type WI5 with an "Open" status. It uses structured data scraping techniques to extract necessary client details and processes each item individually. Each work item is treated as a separate transaction, ensuring isolated processing and accurate tracking.

#### **4.1.8 Generate Report**

As each work item is processed, the extracted data is collected and stored into a structured DataTable. After all items are completed, this DataTable is written into an Excel report and saved at the file path specified in the configuration file. The report includes all client data and processing results.

#### **4.1.9 Final Cleanup**

Upon completion of all transactions, the automation enters the End Process phase. In this stage, the bot ensures that all open applications and browsers are closed properly. It also releases any system resources that were being used and logs the final execution status to mark the process as complete.

#### **4.1.10 Review Output**

Finally, users may review the output of the automation. This involves checking the Excel report generated in the Output folder to verify the results. In addition, log messages displayed in UiPath Studio or stored in Orchestrator provide detailed insight into each transaction, making it easier to identify successes and troubleshoot errors.

### **4.2 Categories of UiPath Activities Used**

The automation bot utilizes several categories of UiPath activities, each tailored to specific functions like UI interaction, data handling, logic control, and system management.

#### **4.2.1 UI Automation Activities**

These activities are primarily used for interacting with web and desktop applications. The Open Browser activity launches a browser window and navigates to a given URL, such as the ACME login page. Attach Browser helps the automation stay connected to a specific browser tab for performing further actions. The Click activity simulates mouse clicks on buttons, links, or input fields. To enter data, the Type Into activity is used. The Element Exists activity checks for the presence of UI elements before attempting interaction. For data extraction, Get Text and Get Attribute are used to capture visible and hidden values from the screen. Finally, Extract Structured Data plays a crucial role in scraping tabular or repeated content from web pages and storing it as a DataTable.

#### **4.2.2 DataTable and Excel Activities**

These activities facilitate reading from and writing to Excel files. The Read Range activity imports data from Excel into a DataTable, while Write Range exports it back to a sheet. Filter Data Table is used to filter specific rows based on logical conditions, such as selecting only completed items. The Add Data Row activity helps dynamically insert new rows into an existing DataTable. Finally, For Each Row allows iteration over each row of the DataTable to perform custom operations like data transformation or validation.

#### **4.2.3 Flow Control Activities**

Flow control activities help manage decision-making and loop executions. The If

activity enables branching based on true or false conditions. While and Do While loops are used when the number of iterations depends on dynamic conditions. The For Each activity is commonly used to iterate through lists, arrays, or collections and apply the same operation to each item.

#### **4.2.4 System Activities**

These are used to support internal logic and modular programming. The Assign activity is employed to initialize and update variable values. Invoke Workflow File allows the main workflow to call smaller reusable .xaml files, promoting better organization. Log Message records custom messages to the output console, aiding in debugging. Additionally, Message Box displays pop-up alerts to the user, mainly used during development or testing stages.

#### **4.2.5 REFramework-Specific Activities**

These are special activities aligned with queue-based automation workflows. Get Transaction Item retrieves the next item from the queue in Orchestrator. After processing, the transaction result is updated using Set Transaction Status, which can mark it as successful, failed due to a system error, or failed due to a business exception. To ensure a clean environment, Kill Process and Close Application are used to terminate any residual applications or background processes before or after execution.

### **4.3 REFramework Execution Lifecycle**

The REFramework consists of four distinct execution phases that manage the automation from start to finish.

#### **4.3.1 Initialization**

In this phase, the bot loads all necessary configurations from Config.xlsx, opens all required applications such as web browsers, and retrieves login credentials securely from the Orchestrator or Windows Credential Manager.

#### **4.3.2 Get Transaction Data**

Here, the bot fetches the next transaction item from the queue or data source. It applies

filters, such as selecting items of type WI5 with an "Open" status, to ensure only valid items are processed.

#### **4.3.3 Process Transaction**

The core business logic is executed during this phase. For each transaction item, the bot extracts the required data, performs any necessary computations or formatting, and stores the results in a DataTable. It also includes exception handling logic to manage both system and business rule exceptions, along with retry mechanisms.

#### **4.3.4 End Process**

The final phase ensures that all used resources, such as browser windows or Excel files, are properly closed. It logs the final status of the process, saves the output report to the predefined folder, and performs memory cleanup to maintain system performance.

### **4.4 Logging and Output**

Logging is an essential part of REFramework that enables monitoring, auditing, and debugging of automation runs. Log Message activities are embedded throughout the workflow to record key events, decisions, and exceptions. When the automation is deployed through UiPath Orchestrator, these logs are collected centrally for easier access and analysis.

Alongside logging, an Excel report is generated summarizing all processed transactions. This report is created using the Write Range activity and saved to the path defined in the configuration file. The report provides a structured view of client data and the processing outcomes, making it suitable for review, audit, or business reporting purposes.



## **IMPLEMENTATION**

---

This chapter outlines the comprehensive implementation of the Robotic Process Automation (RPA) solution using UiPath, based on the Robotic Enterprise (RE) Framework. The objective is to automate the process of retrieving, processing, and updating work items from the ACME System 1 portal. The implementation ensures robust error handling, scalability, and maintainability by leveraging modular design principles, dispatcher–performer architecture, and best practices recommended by UiPath.

### **5.1 Workflow Overview**

The project follows a Dispatcher–Performer model. The Dispatcher component is responsible for logging into the ACME System 1 portal, extracting work items of type "WI5" with the status "Open," and uploading them to an Orchestrator Queue. This ensures that the data required for processing is made available in a centralized, structured format. The Performer component picks up each queue item from the Orchestrator and performs the required actions. It reads the necessary details, generates a SHA-1 hash for each item, updates the Comments field with the hash, sets the status to "Completed," and submits the updates to the ACME system. The RE Framework enhances this implementation by incorporating retry mechanisms, system logging, exception handling, and clean segregation of initialization, transaction processing, and closure stages.

### **5.2 Workflow Components**

The Main.xaml file acts as the entry point of the entire automation. It coordinates the overall process by invoking core subprocesses such as InitAllSettings.xaml, InitAllApplications.xaml, GetTransactionData.xaml, Process.xaml, and EndProcess.xaml. The InitAllSettings.xaml reads configuration parameters from the Config.xlsx file, which includes URLs, timeouts, folder paths, and other constants. It also retrieves Orchestrator Assets for secure credentials and URLs.

The InitAllApplications.xaml file is dedicated to initializing browser instances and navigating to the ACME System 1 login page. It uses stored credentials from Orchestrator to log in securely. Once authenticated, the automation can interact with the dashboard and other pages.

The GetTransactionData.xaml file is responsible for retrieving the next available queue item from the Orchestrator. It populates the TransactionItem variable with the next item to be processed, setting the stage for business logic execution in the Process.xaml.

The Process.xaml file contains the business logic that processes each work item. It navigates to the relevant dashboard section, extracts required item information, generates a hash code using the SHA-1 algorithm, and updates the Comments field in the ACME portal. It then changes the item status to "Completed" and confirms the submission.

The EndProcess.xaml ensures clean-up by logging out of the ACME system, closing all application windows and browser tabs, and releasing resources. This step maintains system hygiene and prevents unnecessary resource consumption.

### **5.3 Asset and Configuration Management**

Configuration management is handled through the Config.xlsx file. This file contains structured data for values such as application URLs, orchestrator queue names, file storage paths, timeout durations, and retry counts. Secure information, such as usernames and passwords, is stored in Orchestrator Assets. These assets are categorized as Text, Credential, or Boolean types depending on their purpose. Using orchestrator assets ensures security and centralized control.

### **5.4 Logging and Exception Handling**

Logging is implemented using UiPath's Log Message activity, which records execution details in real-time. Logs are categorized into Info, Warning, Error, and Trace levels to differentiate between normal execution, recoverable exceptions, and critical failures. Try-Catch blocks are embedded in all major subprocesses to handle known exceptions such as SelectorNotFoundException and TimeoutException. In cases where a transaction fails, the RE Framework's retry mechanism re-attempts the operation based on defined policies. Failed transactions are also recorded for review and post-mortem analysis, allowing continuous improvement of the process.

### **5.5 Testing and Validation**

The solution was tested with multiple work items of type "WI5" to verify accuracy

and consistency. During testing, it was validated that the hash codes generated matched the required format and that each work item's status was correctly updated to "Completed" after processing. An Excel-based report was generated post-processing, listing all successfully completed work items and their hash values. The validation process confirmed that the automation reliably handles variations in item structure, browser delays, and unexpected system behavior without data loss or corruption.

## **5.6 Tools and Technologies Used**

The development was done using the latest version of UiPath Studio, which provided an intuitive drag-and-drop interface and a wide range of pre-built activities. UiPath Orchestrator was used to schedule jobs, monitor bots, and store assets and logs. The SHA-1 hash generator was integrated using the `System.Security.Cryptography` namespace. Excel 2016 and above was used for storing and verifying reports. The Google Chrome browser, with the UiPath extension installed, served as the automation interface for interacting with the ACME System 1 portal.

## **5.7 Scalability and Maintainability**

The RE Framework offers a modular structure that promotes reusability. Each xaml file is designed for a specific function, allowing developers to replace, reuse, or update individual components without affecting the entire system. Scalability is achieved by configuring multiple bots (Performer) to process the same queue, enabling parallel transaction handling. This approach is well-suited for enterprise environments where transaction volumes vary. Moreover, enhancements such as logging verbosity, environment-specific configuration, and dynamic selectors were introduced to ensure that the system can evolve with business needs.

In conclusion, the implementation of this RPA solution demonstrates a robust, scalable, and secure design using UiPath's RE Framework. It not only automates a previously manual and error-prone task but also introduces governance, reliability, and performance monitoring into the process lifecycle. This serves as a strong foundation for future automation initiatives.

## TESTING AND VALIDATION

---

Testing and validation are essential components of the RPA lifecycle to ensure the developed automation solution performs as intended under various conditions. In this chapter, we explore the testing approach applied to the UiPath RE Framework-based solution for automating the ACME System 1 portal, focusing on correctness, stability, and completeness.

### 6.1 Unit Testing of Workflows

Each workflow component in the RE Framework—such as `InitAllSettings.xaml`, `InitAllApplications.xaml`, `GetTransactionData.xaml`, and `Process.xaml`—was subjected to unit testing. This process involved isolating each component and verifying its logic independently. For example, `InitAllSettings.xaml` was tested by running it standalone to confirm it could accurately read configuration values from `Config.xlsx` and retrieve the required Orchestrator assets. Similarly, `Process.xaml` was tested separately to ensure it correctly extracted data from the ACME portal, generated the SHA-1 hash, and updated the corresponding work item status. Any exceptions, like selector errors or data mismatches, were captured and resolved during this stage to ensure each module functions robustly when integrated.

### 6.2 Test Data Preparation

Effective testing required the creation of suitable and representative test data. In the ACME System 1 portal, a predefined set of work items with type "WI5" and status "Open" was used. These items were manually verified beforehand to ensure that they met the expected criteria. Test data also included variations in item formats to check how well the automation handles edge cases—such as missing fields, incorrect formats, or unexpected characters. This diverse dataset enabled comprehensive testing across both positive and negative scenarios, improving the solution's resilience.

### 6.3 Output Validation

After executing the workflow, the results were rigorously validated against expected outcomes. The automation was expected to update each work item's status to "Completed" and insert the generated SHA-1 hash into the Comments field. Validation involved both automated checks and manual reviews. The Excel report generated by the bot was cross-checked to ensure that each processed work item reflected the correct status change and hash value. Furthermore, a comparison was made between the original data and the updated records in the ACME portal to confirm consistency.

## **6.4 Logging and Debugging Tools Used**

UiPath's built-in logging features were extensively used during testing. Log Message activities were embedded in critical steps across the workflows to capture real-time execution details, such as initiation, data extraction, hash generation, and update confirmation. These logs helped trace errors and understand workflow behavior during failures. The UiPath Orchestrator logs provided centralized tracking for both successful transactions and business/application exceptions, facilitating detailed audits. Additionally, the Output and Error panels in UiPath Studio were used for local debugging, enabling the developer to identify and correct selector issues, variable misassignments, or argument mismatches.

## **6.5 Exception Handling Validation**

The retry mechanism and exception handling features of the RE Framework were also tested thoroughly. Transactions that encountered errors, such as page load delays or dynamic selector failures, were logged as Business or Application exceptions. The retry configuration in the Orchestrator Queue was validated by introducing controlled errors in test data to observe if failed items were retried correctly. The Try-Catch blocks within Process.xaml were evaluated to ensure the bot handled exceptions gracefully without crashing the entire workflow.

## **6.6 Performance Testing**

To assess performance and stability under load, the automation was tested with a large number of WI5 items. The goal was to evaluate processing speed and resource

utilization. It was observed that even under high transaction volumes, the RE Framework ensured efficient processing by cleanly separating transaction logic and leveraging retry logic without bottlenecks. Memory and CPU usage were monitored to confirm the workflow did not degrade system performance significantly, making it suitable for unattended or long-running execution.

This chapter demonstrates a thorough validation of the RPA solution, ensuring it is both reliable and scalable for deployment in real-time business environments. Let me know if you'd like to include screenshots of logs or sample test data tables for added completeness.

## PERFORMANCE ANALYSIS

---

Performance analysis plays a crucial role in evaluating the impact and effectiveness of Robotic Process Automation (RPA) implemented using UiPath. This chapter provides a comparative study between manual and automated execution of business processes, focusing on time efficiency, error reduction, and system resource utilization. The analysis helps in quantifying the benefits of automation and justifies the investment made in terms of development and infrastructure.

### 7.1 Manual Execution Time vs. Bot Execution Time

Prior to automation, the process of logging into the ACME System 1 portal, navigating through work items, extracting relevant data, generating hash codes, and updating records manually took a considerable amount of time. On average, it took a human user approximately **2 to 3 minutes** to complete a single transaction, depending on the complexity of the item and the employee's familiarity with the system.

In contrast, the UiPath bot, developed using the RE Framework, significantly reduced this processing time. The same task was completed by the bot in approximately **30 to 45 seconds per item**, which includes logging in, navigating, data extraction, hash generation, and updating the work item. Over a batch of 100 work items, the total processing time was reduced from nearly **5 hours (manually)** to under **1.5 hours** using automation. This demonstrates a **time saving of more than 70%**, allowing employees to focus on higher-level tasks and reducing operational delays.

### 7.2 Error Rates Before and After Automation

Manual data processing is inherently prone to human errors, especially when tasks are repetitive, such as copying and pasting data or updating fields in multiple records. During the manual phase, common errors included:

- Entering incorrect hash codes
- Skipping records unintentionally
- Updating the wrong fields
- Missing status updates

Historical data showed that such errors occurred at a rate of **8–10 errors per 100 records**, which often led to rework and inconsistent reporting.

After automation, the bot performed with near-perfect accuracy. Since bots follow strict logic and rule-based workflows, the error rate dropped drastically to less than **1 error per 1000 records**, and most of these were due to external system issues like temporary web portal downtime or changes in UI selectors. The use of Try-Catch blocks and logging ensured such exceptions were identified and logged for manual review, rather than going unnoticed. This significant reduction in error rate also led to improved data quality and compliance.

### 7.3 CPU and RAM Usage Analysis

System resource consumption was measured during bot execution to evaluate its efficiency and suitability for unattended or continuous operation. The tests were conducted on a system with **Intel Core i5 processor, 8 GB RAM, and Windows 10 (64-bit)**.

During peak processing, the CPU usage by UiPath hovered between **25% and 40%**, depending on the complexity of the browser automation and data manipulation tasks. RAM usage remained within the **2 GB to 2.5 GB** range, indicating efficient memory management even during batch execution. No system slowdowns or crashes were observed during prolonged bot runs, validating the implementation's stability.

Compared to manual execution, which also consumes CPU cycles due to human interaction, multitasking, and potential browser lags, the automated bot ran more predictably and efficiently, ensuring optimal use of available system resources.

### 7.4 Impact on Productivity

Beyond raw performance numbers, automation brought a tangible improvement in employee productivity. With bots taking over repetitive, time-consuming work, human employees were able to allocate more time to exception handling, data analysis, and process optimization. This shift not only enhanced operational throughput but also improved job satisfaction among staff members who were freed from monotonous tasks.

### 7.5 Scalability Metrics

Scalability is one of the most valuable features of an RPA solution. In performance testing, the



UiPath automation was capable of processing over **1000 work items per day** when scaled across multiple bots using Orchestrator queues. Unlike manual efforts, which would require hiring and training additional personnel to handle growing workloads, automation can simply be scaled by deploying more bots. This capability ensures consistent performance even during seasonal spikes or business expansion.

## **7.6 Uptime and Reliability**

Another performance factor analyzed was uptime and reliability. Manual execution is subject to employee availability, sick leaves, breaks, and shift timings. Bots, however, can run **24/7 without interruption**, and in the implemented solution, the bot maintained a **99.8% uptime** during scheduled execution periods. This near-continuous availability not only increases throughput but also ensures critical tasks are never delayed due to human absence.

In summary, the performance analysis reveals that the UiPath-based automation significantly outperforms the manual process in terms of speed, accuracy, and efficiency. It also showcases the scalability and resource-friendliness of the RPA solution, making it suitable for large-scale enterprise deployment. The drastic reduction in execution time and error rates clearly demonstrates the business value of automation, reinforcing its role as a strategic enabler of digital transformation.

## **FUTURE SCOPE**

The implementation of Robotic Process Automation (RPA) using UiPath for automating the ACME System 1 platform has laid a strong foundation for transforming repetitive and rule-based tasks into efficient, error-free processes. However, this automation initiative can be further expanded and enhanced in the future to achieve even greater levels of operational excellence. The following are key areas for future scope:

### **1. Integration with Artificial Intelligence (AI)**

RPA bots handle structured data efficiently, the next logical step is integrating them with AI technologies such as Machine Learning and Natural Language Processing. This integration will enable intelligent decision-making in complex workflows, predictive analytics to forecast work item loads or identify anomalies, and chatbot integration for client interaction and issue resolution. AI-powered RPA will increase the intelligence and autonomy of automation solutions.

### **2. Document Understanding and OCR**

Many business processes involve scanned documents, PDFs, and handwritten forms. UiPath's Document Understanding framework, combined with OCR engines (e.g., Tesseract, Google OCR), can be leveraged in the future to extract data from unstructured and semi-structured documents. This will extend automation capabilities beyond structured web portals into digitizing and processing physical paperwork, enhancing data processing and accessibility.

### **3. Scalability Through UiPath Orchestrator**

Currently, the system may be designed to run manually or on a schedule. By integrating fully with UiPath Orchestrator, bots can be scheduled, monitored, and deployed remotely. This enables multiple bots to handle large volumes concurrently, improving scalability and reliability. Orchestrator Queues can also be used to manage dynamic workloads with prioritization logic, allowing better resource allocation.

### **4. Cloud and Virtual Machine Automation**

Incorporating cloud-based environments or virtual machines (VMs) using platforms like AWS, Azure, or Google Cloud will allow remote and distributed bot deployment. It increases resilience, reduces infrastructure dependency, and supports dynamic scaling based on workload and demand. Cloud-based RPA enables seamless integration across geographies and hybrid work environments.

## **5. Multi-System Integration**

Future development can include the integration of UiPath bots with ERP systems (e.g., SAP, Oracle), CRM platforms (e.g., Salesforce, Zoho), and data analytics tools (e.g., Power BI, Tableau). This will enable seamless data flow between systems, eliminate redundant data entry, and reduce human intervention across departments, ensuring a unified and streamlined business process.

## **6. Real-time Dashboards and Reporting**

Implementing real-time dashboards will provide live monitoring of bot performance, instant insights into processed vs. pending work items, and better visibility for management. This facilitates quicker decision-making, improves transparency, and allows for proactive process optimization and performance tracking.

## **7. Advanced Exception Handling**

Enhancing the REFramework to handle more complex exceptions can further increase automation robustness. By classifying errors as business or application exceptions, integrating automated alerts via email, SMS, or Slack for real-time incident response, and maintaining a historical log of exceptions, organizations can improve root-cause analysis and continuously refine automation logic.

## **8. User Feedback Loop**

Incorporating a feedback system where users can provide suggestions or report issues will facilitate ongoing improvement of the automation solution. This human-in-the-loop design allows the bot to learn from real-time inputs, leading to adaptive automation that aligns with user expectations and evolving business requirements.

## **9. Mobile App Integration**

Future iterations of the solution can include mobile app capabilities that allow users to trigger automations remotely, view real-time status reports or logs, and receive alerts or approve transactions on the go. Mobile integration offers greater flexibility and control, especially for operations teams and managers.

## **10.End-to-End Automation for Client Onboarding**

Automation can be extended beyond processing work items to complete client onboarding processes. This includes validating submitted documents, sending welcome emails or SMSs, and creating client profiles across multiple platforms automatically. Such end-to-end automation ensures a seamless, consistent, and efficient onboarding experience.

## CONCLUSION

In conclusion, the ACME Client Info Automation project successfully demonstrated the potential of UiPath and the Robotic Enterprise (RE) Framework in automating repetitive and rule-based web tasks with precision and reliability. The project achieved its primary objective of automating the login process, extracting client data from filtered work items, and generating structured reports, thereby significantly reducing manual effort and human error. The RE Framework added robustness to the solution by providing a well-defined structure for initialization, transaction processing, error handling, and clean termination.

While the bot performed efficiently under normal conditions, the analysis also highlighted areas that can be improved for future iterations. For example, the current version depends heavily on fixed selectors and a consistent UI layout, which could be vulnerable to even minor changes in the ACME system's interface. Implementing more intelligent selector strategies or integrating AI-based OCR tools could improve adaptability in such scenarios. In terms of output generation, the report structure was effective, but enhancements like automated report formatting, summary generation, and email distribution could improve usability.

Error handling was functional, but incorporating retry mechanisms through queue-based processing in UiPath Orchestrator would make the bot more resilient during high-load execution or temporary failures. Additionally, while the RE Framework handled logging well, integrating centralized monitoring through Orchestrator and setting up alerts for failure or anomaly detection would further improve operational visibility. Credential security was managed locally via configuration, but migrating to Orchestrator Assets would enhance security.

## REFERENCES

- [1] Syed, V. Malhotra, and M. Rathore, “Robotic Process Automation: The Future of tomation,” *2019 IEEE International Conference on System, Computation, Automation and Networking (ICSCAN)*, Pondicherry, India, pp. 1–5, Mar. 2019. doi: 10.1109/ICSCAN.2019.8878700
- [2] M. R. E. M. Sani and R. Ramli, “An Overview of Robotic Process Automation (RPA) Implementation,” *2020 IEEE 10th Symposium on Computer Applications & Industrial Electronics (ISCAIE)*, Penang, Malaysia, pp. 56–60, Jul. 2020. doi: 10.1109/ISCAIE47305.2020.9108867
- [3] M. Willcocks, L. P. Mary, and A. Craig, “Robotic Process Automation: The Next Transformation Lever for Shared Services,” *London School of Economics – Outsourcing Unit*, White Paper, 2015.
- [4] D. Singh and S. Kumar, “Automation of Business Processes Using RPA and Artificial Intelligence,” *2021 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)*, Greater Noida, India, pp. 580–585.
- [5] Feb. 2021. doi: 10.1109/ICCCIS51004.2021.9397163
- [6] R. Agrawal and A. Tayal, “A Review on Adoption and Challenges of Robotic Process Automation in Industries,” *2022 6th International Conference on Trends in Electronics and Informatics (ICOEI)*, Tirunelveli, India, pp. 1137–1143, Apr. 2022. doi: 10.1109/ICOEI53556.2022.9776995.