ACCIDENT REPORT GENERATOR

A PROJECT REPORT

Submitted by

G BALA SAI SREEJA KUMARI (220701035)

in partial fulfilment 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 **"ACCIDENT REPORT GENERATOR"** is the Bonafide work of **"G BALA SAI SREEJA KUMARI (220701035)"** who carried out the project work for the subject OAI1903-Introduction to Robotic Process Automation under my supervision.

Mrs. J. Jinu Sophia

SUPERVISOR

Assistant Professor (SG)

Department of

Computer Science and Engineering

Rajalakshmi Engineering College

Rajalakshmi Nagar

Thandalam

Chennai - 602105

Submitted to Project and Viva Voce Examination for the subject OAI1903-Introduction to Robotic Process Automation held on _____.

ACKNOWLEDGEMENT

Initially we thank the Almighty for being with us through every walk of our life and showering his blessings through the endeavour to put forth this report. Our sincere thanks to our Chairman **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.

G BALA SAI SREEJA KUMARI (220701035).

ABSTRACT:

The "Accident Report Generator" project is a cutting-edge Robotic Process Automation (RPA) solution developed to streamline the process of accident reporting. Utilizing UiPath, the system efficiently gathers and processes essential accident-related information. The system then automates the retrieval of relevant data from static sources, including a summary of the accident, a list of affected areas or entities, potential causes, and safety recommendations. These operations leverage the Parallel activity to execute simultaneously, ensuring speed and accuracy. The compiled report is formatted into a Word or text file, converted into PDF format, and sent to the user's email via the SMTP Mail Message activity. This automation minimizes manual intervention, delivering a detailed and professional accident report in record time. The project highlights the transformative capabilities of RPA in enhancing operational efficiency and improving reporting processes.

TABLE OF CONTENTS

LIST OF FIGURES:

LIST OF ABBREVIATIONS:

Abbreviation	Full Form
SMTP	Simple Mail Transfer Protocol
ERD	Entity Relationship Diagram
DFD	Data Flow Diagram
HR	Human Resources
API	Application Programming Interface
RE	Robotic Enterprise
RPA	Robotics Process Automation

CHAPTER-1

INTRODUCTION

The "Accident Report Generator" project is an innovative Robotic Process Automation (RPA) solution designed to streamline the accident reporting process. Leveraging UiPath, the system simplifies data collection and report generation by automating key steps. Users input essential details such as the accident location, date, and specific incident information. The system retrieves additional relevant data, including incident summaries, affected areas, potential causes, and safety recommendations, from static sources. These operations are executed in parallel for optimal efficiency. The compiled information is formatted into a detailed report, converted into PDF format, and sent to the user via email. This project showcases the potential of RPA to improve productivity and accuracy by automating labour intensive reporting tasks.

1. 1 GENERAL

The purpose of the "Accident Report Generator" project is to streamline the accident reporting process by automating the collection and presentation of critical incident details. It aims to save time and effort for users by generating a comprehensive accident report, including summaries, causes, and recommendations, in a structured format. This solution enhances accuracy and efficiency, making the reporting process quick and hassle-free.

1.2 OBJECTIVE

The objective of the "Accident Report Generator" project is to develop an automated system that efficiently collects and processes critical accident-related information. It aims to provide users with a detailed report summarizing the incident, including the location,

affected areas, potential causes, and safety recommendations. The project ensures that this information is compiled into a professional report, formatted as a PDF, and delivered to the user via email. This solution simplifies and enhances the accident reporting process, saving time and effort while ensuring accuracy and consistency.

1.3 EXISTING SYSTEM

In the current scenario, accident reporting involves manually gathering information from multiple sources, such as field reports, static records, or online data. Users often need to consolidate details about the location, nature of the incident, causes, and safety recommendations individually, which is time-consuming and error-prone. Additionally, formatting this data into a structured report requires extra effort. Most existing systems lack automation and integration, making the process inefficient, inconsistent, and less user-friendly. The absence of a centralized tool to collect, process, and deliver this information seamlessly highlights the need for an automated solution like the "Accident Report Generator."

1.4 PROPOSED SYSTEM

The "Accident Report Generator" project introduces an automated solution to streamline accident reporting using Robotic Process Automation (RPA) with UiPath. The system allows users to input details such as the accident location, date, and incident specifics, then automates the retrieval of related information, including summaries, affected areas, potential causes, and safety recommendations from static sources. These data retrieval tasks run

concurrently, ensuring speed and efficiency. The information is compiled into a structured Word or text file, converted into a PDF, and sent to the user via email. This centralized, automated approach eliminates the need for manual effort, providing a fast, accurate, and user-friendly reporting experience.

CHAPTER-2

LITERATURE REVIEW

Accident reporting traditionally involves manual data collection and documentation, which is often time-consuming and prone to human error. Research demonstrates that Robotic Process Automation (RPA) tools, such as UiPath, can automate repetitive tasks like data extraction and report generation, enhancing both efficiency and accuracy. This project leverages advancements in RPA to provide an automated solution for collecting, processing, and delivering comprehensive accident reports.

2.1 GENERAL

The advancement of automation technologies, particularly RPA, has revolutionized the way repetitive tasks are managed across various domains. In the field of accident reporting, manual processes for gathering and compiling details such as incident location, causes, and safety recommendations are labour intensive and inefficient. This limitation has paved the way for the development of automated systems that streamline and accelerate the reporting process.

RPA platforms, especially UiPath, have gained prominence due to their ability to perform tasks like data extraction, document creation, and email communication automatically.

While much of the focus has been on industries like healthcare and finance, the application of RPA in incident reporting is relatively new but offers tremendous potential to enhance operational efficiency and accuracy.

Web scraping and data extraction techniques have also seen significant advancements, enabling the automated retrieval of detailed information from static or predefined sources. Using RPA activities such as "Anchor," "Find Element," and "Get Text," relevant data related to accidents—such as affected areas, potential causes, and recommendations—can be extracted efficiently. Parallel processing further optimizes performance by allowing concurrent execution of multiple tasks, such as data retrieval and report formatting.

Additionally, automation in document creation and delivery, such as converting reports into PDFs and emailing them via SMTP protocols, has become an integral feature of modern reporting systems. By incorporating these functionalities, the "Accident Report Generator" project demonstrates the practical application of RPA in creating efficient and user-friendly solutions for accident reporting.

In summary, the literature indicates that RPA, web scraping, and email automation technologies are powerful tools for enhancing the efficiency of accident reporting systems. The "Accident Report Generator" project capitalizes on these advancements to offer an automated, accurate, and streamlined solution for generating comprehensive accident reports.

CHAPTER-3

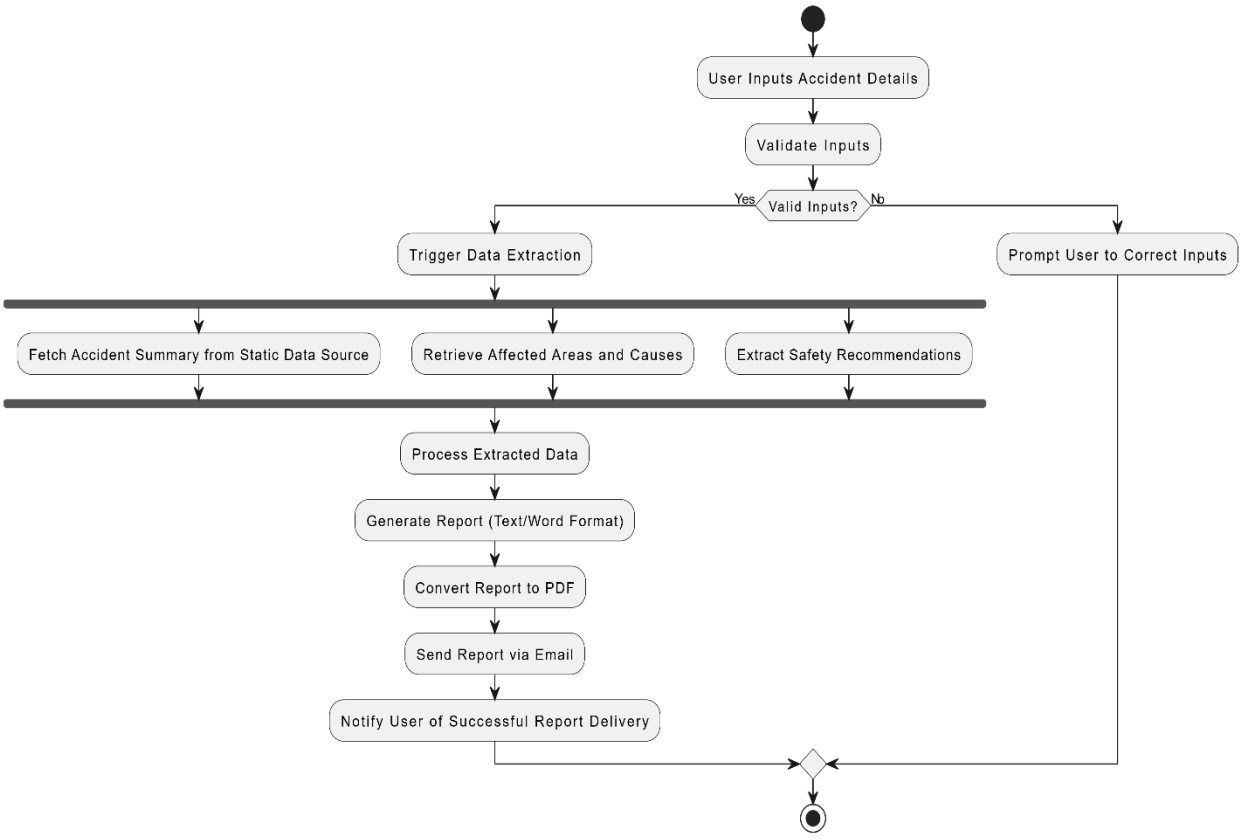
SYSTEM DESIGN

3.1.1 SYSTEM FLOW DIAGRAM

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

Description:

1. **Input**: Candidate data from an Excel sheet, including their hiring status.
2. **Process**:
    - Read the Excel sheet.
    - Identify candidates marked as "hired.".
    - Send letters via email.
3. **Output** Confirmation of email sent and logs for error handling.

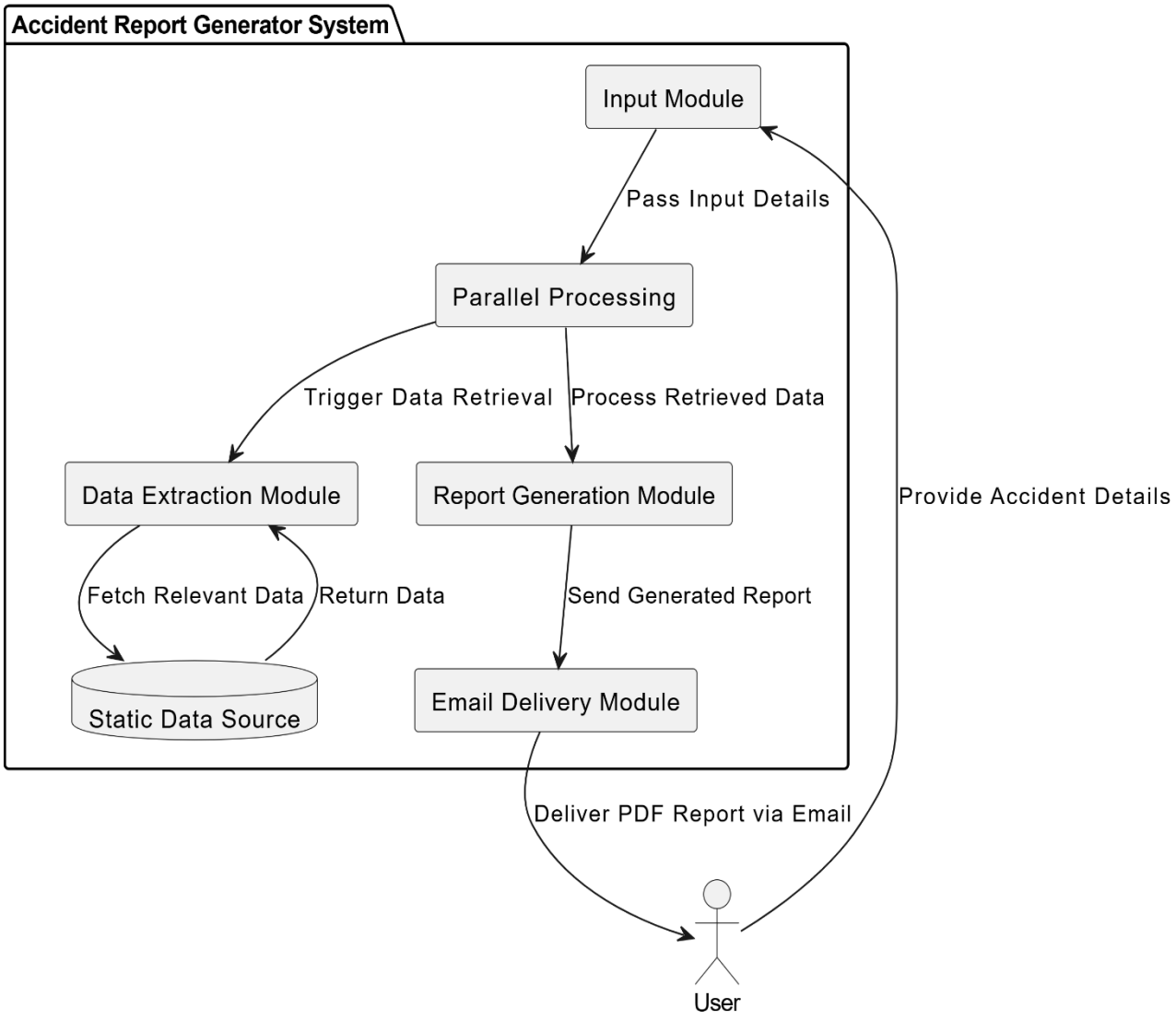


3.1.2 ARCHITECTURE DIAGRAM

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

Components:

1. **Frontend**: User interface for HR personnel (e.g., UiPath Forms or a dashboard).
2. **Backend**: Core logic, including:
   - Excel processing to read candidate data.
   - Recruitment letter generation.
   - Email module for sending letters.
3. **Database/Storage**: To log sent emails and errors.
4. **External Services**: Email server (SMTP) for dispatching letters.

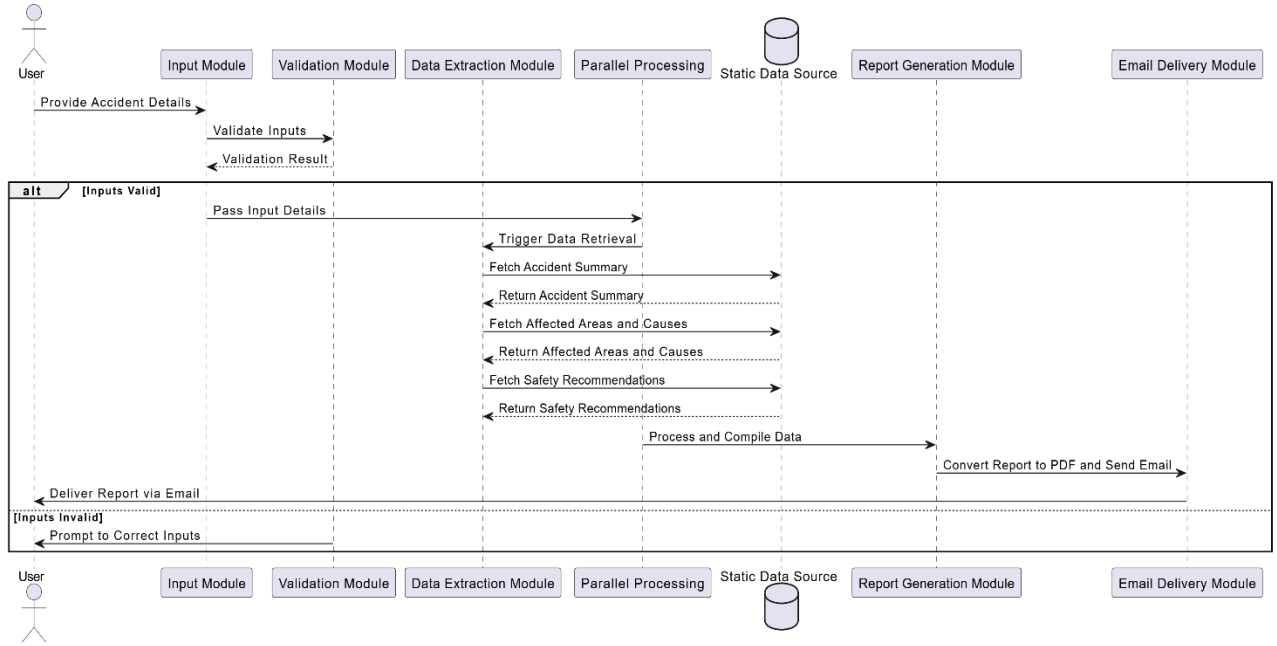


3.1.3 SEQUENCE DIAGRAM

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

Steps:

1. HR personnel trigger the process.
2. The system reads the Excel sheet for candidate data.
3. For each candidate marked as "hired":
    - Generate a recruitment letter.
    - Send the letter via email.
    - Log success or failure.
4. Notify HR personnel of the completion or any errors.



CHAPTER-4

PROJECT DESCRIPTION

The "Accident Report Generator" project automates the accident reporting process by extracting critical details about an incident, such as location, causes, and safety recommendations. Using UiPath, the system retrieves data concurrently, compiles it into a structured report, and delivers the finalized PDF to the user via email. This RPA solution streamlines reporting, saving time and effort while ensuring accuracy.

4.1 METHODOLOGY

This project utilizes UiPath RPA to automate accident reporting tasks. It begins by gathering user input about the accident, such as location, date, and description, via a dialog box. The system extracts relevant data concurrently using parallel activities and stores the information in a structured report. Finally, the data is converted to a PDF and sent to the user via email using SMTP, ensuring an efficient and automated reporting process.

1. Requirements Gathering:

The initial phase focuses on understanding the user's needs, including input format for accident details, required information (e.g., summaries, affected areas, recommendations), and the desired output format (PDF). Interaction with the user occurs via dialog boxes, while the output is delivered via email. Technical requirements like web scraping, parallel processing, document generation, and email automation are defined.

2. System Design:

The system design involves creating a workflow that automates the accident reporting process. It incorporates data extraction from static sources using UiPath activities such as "Anchor," "Find Element," and "Get Text." Parallel activities are utilized to retrieve multiple data points (e.g., causes and recommendations) simultaneously. The system design also includes generating a detailed report, converting it into a PDF, and automating the email delivery process.

3. Implementation:

During implementation, UiPath is used to develop the RPA workflow. The user provides accident details, and the automation script extracts relevant data from predefined sources. Parallel processing is implemented to enable simultaneous data retrieval, enhancing efficiency. The system generates a comprehensive report, converts it to PDF format, and sends it via email using the SMTP Mail Message activity.

4. Testing:

Testing ensures the system's accuracy and efficiency. Unit testing is performed for each activity (e.g., data extraction, PDF conversion, email sending). Integration testing ensures seamless interaction between components. The system is tested for edge cases, such as incomplete inputs or connectivity issues, and for the quality of the output report. User acceptance testing confirms that the solution meets user requirements and expectations.

5. Deployment:

After testing, the system is deployed in a live environment. Users interact with the tool to generate reports based on real-world scenarios. Documentation is provided for maintenance and troubleshooting. The system is monitored post-deployment to ensure smooth operation, with any issues addressed in subsequent updates.

4.1.1 MODULES:

1. User Input Module:

Handles user interaction, prompting for accident details like location, date, and description.

Requirements: Input validation to ensure completeness and accuracy.

2. Web Data Extraction Module:

Automates data retrieval from static sources using UiPath activities like "Anchor," "Find Element," and "Get Text." Collects information such as summaries, causes, and recommendations.

Requirements: Integration with static data sources, handling data variations, and accurate extraction.

3. Parallel Data Extraction Module:

Manages concurrent retrieval of multiple data points (e.g., summaries, causes, recommendations) using UiPath's "Parallel" activity.

Requirements: Effective synchronization to avoid conflicts and ensure efficient execution.

4. Report Generation and Storage Module:

Compiles retrieved data into a structured report using UiPath's "Append Text" activity. The report is organized for clarity and readability.

Requirements: Document formatting and proper organization of extracted data.

5. PDF Conversion Module:

Converts the report into a PDF format to ensure professionalism and easy distribution.

Requirements: Reliable conversion tools and preservation of layout and formatting.

6. Email Notification Module:

Sends the finalized PDF report to the user via email using the "SMTP Mail Message" activity.

Requirements: SMTP configuration, customized email content, and error handling for failed deliveries.

7. Error Handling and Logging Module:

Tracks errors during the process, logs them for troubleshooting, and notifies the user of issues.

Requirements: Comprehensive logging and user-friendly notifications.

8. System Monitoring and Maintenance Module:

Monitors the system after deployment to ensure smooth operation. Includes health checks, updates, and troubleshooting scripts.

Requirements: Tools for monitoring RPA workflows and regular maintenance routines.

CHAPTER-5

OUTPUT SCREENSHOT

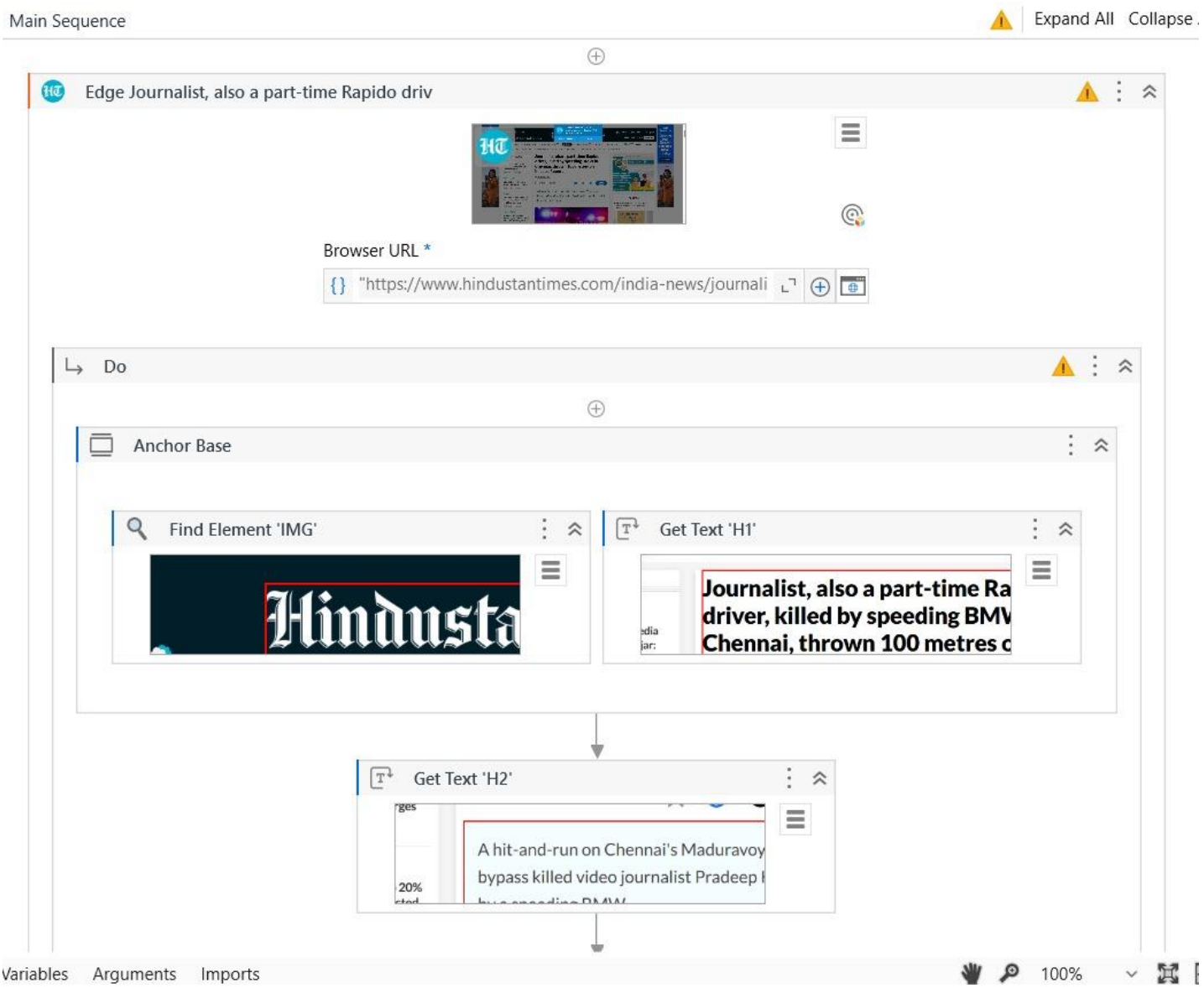


Fig. 5.2. Overview Of RE Framework

From this above figure Contains the Overview of RE Framework

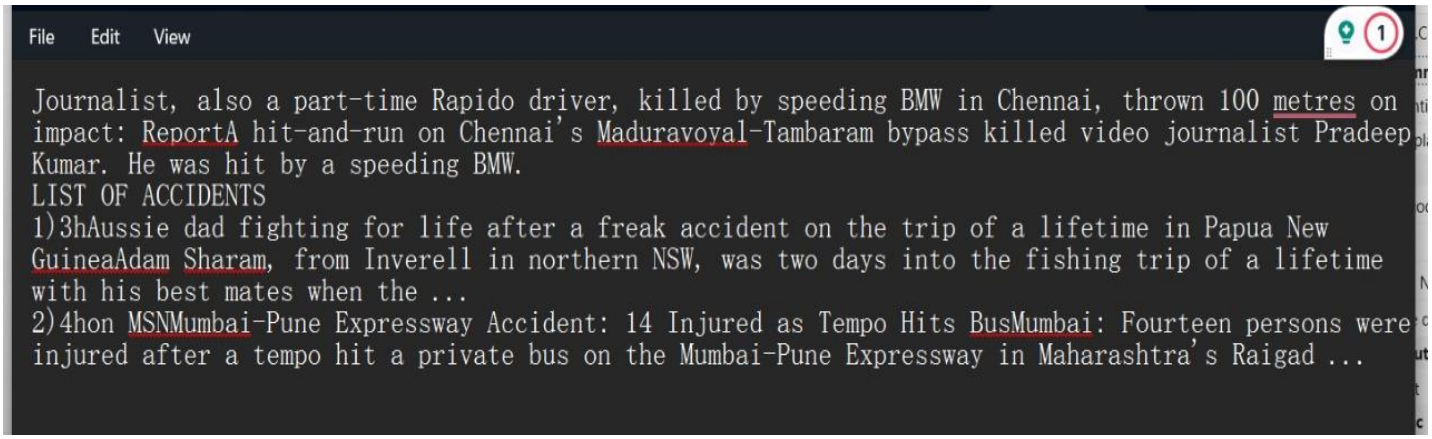
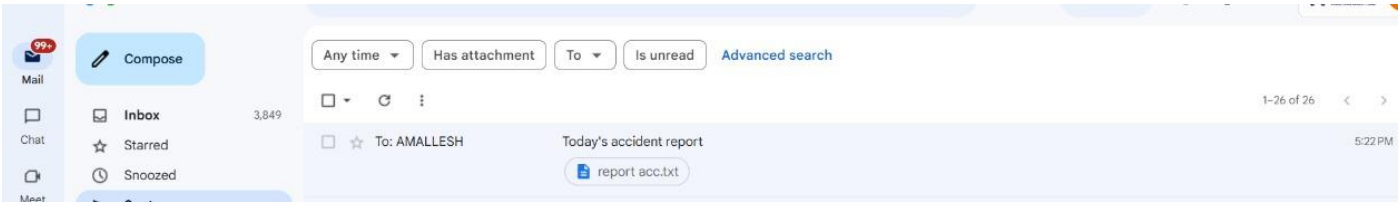


Fig. 5.3. Frame work Of RE Excel

From this above figure Contains the Framework of RE Excel Sheet.



CHAPTER-6

CONCLUSIONS

The "Accident Report Generator" project successfully automates the process of generating detailed accident reports, streamlining the extraction of critical data such as incident location, causes, and safety recommendations. By utilizing UiPath's RPA capabilities, the system efficiently retrieves information, processes it in parallel, and compiles it into a structured report. This project showcases the power of RPA to improve efficiency and enhance the reporting process, eliminating the need for manual data collection and report creation. With automated email delivery, the solution ensures a seamless and user-friendly experience, making accident reporting faster and more

reliable. The "Accident Report Generator" lays the groundwork for future advancements in automation within the safety and accident reporting domain, offering potential for scalability, improved accuracy, and ease of use in future iterations.

6.1 GENERAL:

The "Accident Report Generator" project automates the process of creating detailed accident reports, improving both efficiency and accuracy. Users input essential details about the accident, such as location, date, and incident description, through a simple dialog box. The system then extracts critical information, including causes, affected areas, and safety recommendations, using UiPath's web scraping and data retrieval capabilities. These tasks run concurrently to speed up the process. The collected data is organized into a structured report, which is then converted into a PDF. The system automatically emails the final report to the user via SMTP, eliminating the need for manual research and report creation. This automation saves time, reduces human error, and ensures timely delivery of comprehensive accident reports. The "Accident Report Generator" project demonstrates the power of RPA to streamline the accident reporting process, offering an integrated solution that enhances both productivity and user experience in safety management.

APPENDICES

Appendix 1: Key Code Snippets

This appendix provides code snippets for essential functionalities such as:

1. Extracting the Required Data:

- Code for scraping accident details from a predefined website or database.
- Example: Using UiPath's "Find Element" and "Get Text" activities to extract information like accident location, date, and description.

2. Adding Details to a Text File:
   - Code for appending extracted data into a text file to create a structured accident report.
   - Example: Using UiPath's "Write Text File" or "Append Text" activities to log the extracted data into a text or Word document.
3. Sending Emails via SMTP:
   - Code for configuring and sending the generated PDF accident report to a user's email.
   - Example: Using UiPath's "SMTP Mail Message" activity to send the generated PDF report automatically.

Appendix 2: Process Overview

This appendix includes a Process Overview generated by UiPath, illustrating the dynamic data insertion and format customization. The process overview would highlight the steps involved in:

1. Gathering user input for accident details.
2. Extracting information from external sources.
3. Compiling the report into a structured document.
4. Converting the document into PDF format.
5. Sending the report via email.

The diagram would represent the flow of data through each step of the automated process.

Appendix 3: Testing Logs

This appendix contains a record of the testing process, including:

1. **Test Case IDs**:
    - Unique identifiers for each test case (e.g., TC_001, TC_002).
2. **Test Steps**:
    - Detailed steps taken during the testing phase to validate each component of the system (e.g., inputting accident data, verifying data extraction, confirming email sending).
3. **Expected vs. Actual Results**:
    - A comparison of expected results (e.g., correct extraction of data, accurate report generation) versus the actual system performance during testing.
4. **Notes on Identified Issues and Resolutions**:
    - Any issues encountered during the testing process, along with their resolutions (e.g., failure in email sending due to incorrect SMTP configuration, resolved by correcting email settings).

REFERENCES:

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