International Journal for Multidisciplinary Research (IJFMR)
E-ISSN: 2582-2160 ● Website: www.ijfmr.com ● Email: editor@ijfmr.com

Finance Receipting Using Robotic Process Automation

Dr Prasad P S[1], Manoj J[2], Anjan G M[3], Mohammed Maaz Rehman[4]

[1,2,3,4]School of Computer Science & Engineering, Presidency University, Bengaluru

ABSTRACT

Automation of a back-office process which has a volume of 25000 receipts per month. The information is structured with pre-determined rules and conditions. The process has 10% of exceptions and also involves the use of paper. The source file is downloaded from SAP application, from which data is retrieved.
Based on transaction type, client details are segregated as either ups, cash, cheque, online banking or card. Segregated user information is fed into the web application. For each Receipt number, templates are used to generate receipts and the same is mailed to the vendor specified in the source file. Each transaction has its own template type, which is available in the File Server.

INTRODUCTION

In today's fast-paced business environment, automation has become a crucial tool for streamlining repetitive and labor-intensive tasks. This is especially true for back-office operations, such as financial receipting, which require high accuracy and efficiency. With organizations processing large volumes of transactions daily, manual processing not only slows down operations but also increases the risk of human error.
Our client, a large enterprise, faces the challenge of processing approximately 25,000 receipts per month, a task made even more complex by the need to handle both digital and paper receipts, apply predefined rules, and manage a 10% exception rate. The existing process involves downloading receipt data from an SAP application, segregating transactions based on the payment type (Upi, Cash, Cheque, Online Banking or Card), and generating customized receipts for each transaction type using pre-defined templates stored in a file server. Once generated, these receipts must be emailed to the corresponding vendor, as specified in the source file.
The objective of this project is to design and implement a Robotic Process Automation (RPA) solution that automates this entire receipting process. The solution will retrieve data from SAP, apply business rules, generate receipts using templates, and send the receipts to vendors, all while handling exceptions efficiently. This automation will significantly reduce processing time, increase accuracy and ensure compliance with predefined business rules.

LITERATURE SURVEY

Manual Finance Receipting - High likelihood of human errors due to repetitive and manual data entry.
Manual process is very much time consuming; and slow in processing especially for high volumes (e.g., 25000 receipts per month). Also, it is very expensive which requires a large work force as volume increases, leading to higher operational costs. Manual Processing is hard to scale in order to handle increasing workloads without proportionally increasing staff.
Basic Script-based Automation - Difficult to modify scripts for complex business rules or large volumes of

data. Scripts are not built to handle enterprise-level volumes or complex scenarios. Any process change requires manual updates to the script, leading to higher maintenance time and effort

Robotic Process Automation - Easily handles large volumes of transactions, such as the 25,000 receipts per month. Practically application handle up to 100,000 receipts per run.

Automates repetitive tasks, significantly reducing human errors. RPA (Robotic Process Automation) can be integrated with systems like SAP, Email Servers and other enterprise platforms seamlessly. RPA (Robotic Process Automation) Manages predefined exceptions effectively, reducing manual intervention to just the most complex cases.

ARCHITECTURE

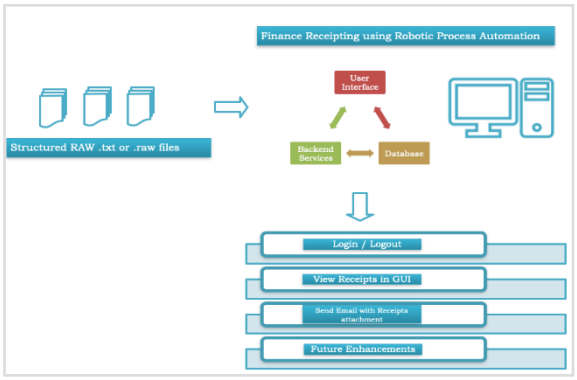

Fig: Finance Receipting using Robotic Process Automation

Assuming that the source raw files are pushed from SAP Application to source directory with .txt extension, Application will read through all the files and parsing and processing is done by parser and processor engines.

Once the processing is done, the data will be stored into the database.

User can login to the application and view each user information with options to view the Receipt in pdf format and also able to send an email with the receipt as an attachment.

Above architecture enables the application to process any number of source or raw files provided that the system must have sufficient hardware capabilities.

METHODOLOGY

Application Layer

Front-End Framework - The application layer is developed using modern front-end frameworks such as React.js. This framework provides robust component-based architecture, enabling reusable code and efficient rendering of the user interface.

Service Layer

Back-End Framework: The service layer is developed to implement the business logic. This framework provides middleware support, routing mechanisms and enable fast development of API services and interaction with database and external APIs

Database Layer

Database Management System [DBMS]: Database Management System (DBMS): The project employs a relational DBMS like Oracle to manage structured financial data. This system support SQL for performing complex queries and transactions.

DESIGN PATTERN

MVC [Model View Controller] - The Model-View-Controller (MVC) design pattern is a widely used software architectural pattern that helps in the separation of concerns within a software application.

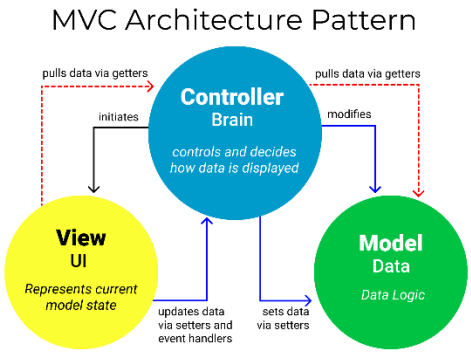

Fig: Model-View-Controller design pattern

For an engineering project, especially if it involves web or software development, the MVC design pattern can be leveraged to structure the application efficiently. Below is a detailed breakdown of how the MVC pattern can be applied to your project, including the roles of each component and its advantages.

The Model-View-Controller (MVC) design pattern divides an application into three interconnected components, each with distinct responsibilities:

Model - Represents the data and the business logic.

View - Responsible for the user interface and presentation.

Controller - Acts as an intermediary between the Model and the View, handling user input and updating the model.

ALGORITHMS

Step1: User login to application on successful authentication

Step2: User clicks on 'PROCESS'

- Application Retrieves raw files from source directory

Parser Engine: Reads the data from raw file and parses it to understand the customer information

Processor Engine: Creates customer objects with specific information w.r.t customer type - ups, cash, cheque, online-banking, card customers

Step3: User clicks on UPI / CASH / CHEQUE / ONLINE-BANKING / CARD button

Step4: User sees relevant customer information in dashboard screen

Step5: User can click on 'Send Email' button in order to trigger email to respective customers

Step6: User can Logout

RESULTS

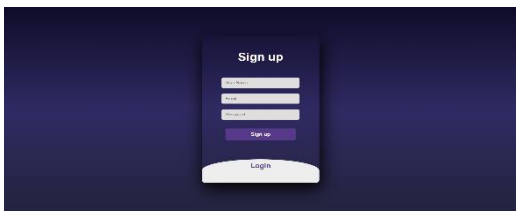

Fig: Application Login Page

Graphical Analysis - Below graph illustrates the execution time of manual process vs execution time by RPA Process.

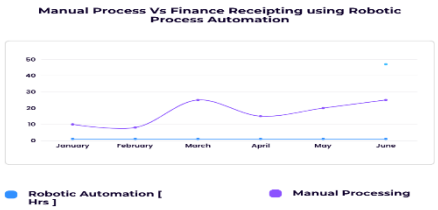

Fig: Graph showing average time taken for Manual process vs Finance Receipting using Robotic Process Automation

CONCLUSION

The conclusion for implementing Finance Receipting using Robotic Process Automation is that it offers significant improvements in operational efficiency, accuracy and cost effectiveness.

The result is a faster, more accurate receipting process that minimizes human errors and ensures better data integrity. It reduces manual workloads, cuts operational costs, enhances compliance, and offers scalable solutions for managing large transaction volumes, ultimately leading to more efficient and effective financial management.

By integrating Robotic Process Automation and Email automation, the system ensured seamless data extraction, template-based receipt generation and timely dispatch to vendors. The automation solution also demonstrated scalability, handling large transaction volumes with ease and enabling exception management for complex cases. This resulted in reduced processing time, lower operational costs and improved accuracy in finance receipting, contributing to smoother business operations and vendor satisfaction.

The project's outcomes highlight its success in achieving the predefined objectives of efficiency, accuracy and scalability. Future enhancements may focus on further refining exception handling and expanding the system's capabilities to integrate additional business processes.

REFERENCES

1. UiPath, Automation of SAP Processes with RPA: Case Study, 2022.
2. Blue Prism, Case Study: Automating Financial Services Back Office, 2021
3. Deloitte, Finance Automation: Transforming the Back Office with RPA, 2021
4. M. Lacity and L. Willcocks, Robotic Process Automation: The Next Transformation Lever for Shared Services, The Outsourcing Unit, London School of Economics and Political Science, 2018