

## **ANJALAI AMMAL MAHALINGAM ENGINEERING COLLEGE**

### **DEPARTMENT OF INFORMATION TECNOLOGY**

Completed the project named as

### **PASSWORD RESET BOT**

SUBMITTED BY

**B.ABISHEK (820422205005)**

# **PASSWORD RESET BOT**

## **Abstract:**

This project involves developing an RPA (Robotic Process Automation) solution to automate the data entry process from an Excel file into an HTML webpage, capturing a generated passcode, and updating the Excel file with the passcode. The solution is designed to handle errors, such as incorrect input data, by generating Business Exceptions. The process includes downloading and extracting challenge files, reading data from the Excel file, entering it into the HTML form, capturing the passcode, updating the Excel file, and logging both the start and end times of the automation. The total execution time is measured and displayed at the end of the process. This RPA solution ensures efficient, error-resilient automation of the password reset process while providing valuable time metrics.

## **Introduction:**

This document outlines the approach and solution for automating the process of data entry from an Excel file to an HTML webpage, capturing a passcode, and updating the Excel file. The process is designed to handle potential errors gracefully, log execution times, and ensure the overall success of the automation task. This automation will simplify and speed up the manual password reset procedure by leveraging Robotic Process Automation (RPA) tools.

## **Purpose of the Document**

The purpose of this document is to provide a clear and structured overview of the RPA solution for automating data entry from an Excel file to an HTML webpage, capturing the generated passcode, and updating the Excel file. It aims to detail the steps involved in the automation process, the tools used, and the expected outcomes, including how errors are managed and how the execution time is tracked.

## Objectives

The key objectives of this RPA solution are as follows:

- **Automate Data Entry:** Efficiently extract data from an Excel file and input it into the appropriate fields on an HTML webpage.
- **Capture Passcode:** Retrieve the passcode generated by the HTML page after data submission.
- **Update Excel File:** Write the captured passcode back into the original Excel file.
- **Error Handling:** Implement Business Exception handling to address incorrect or missing input data, ensuring the automation process runs smoothly.
- **Logging and Execution Time Measurement:** Log the start and end times of the automation process, calculate the total execution time, and display the result to evaluate performance.

## Problem Statement:

1. Download and Extract Files: Download and unzip the challenge files.
2. Contents: HTML file for data entry & Excel file with data to insert into the HTML page.
3. Open WebPage.html and Read data from SampleData.xlsx.
4. Enter the data into the HTML page fields and Capture the generated passcode.
5. Update the Excel file with the passcode.
6. Log the start and end times of the solution & Measure and display the total execution time.

### **Flow of The Process:**

1. Start
2. Download & Extract Files: Ensure the required files (WebPage.html and SampleData.xlsx) are available.
3. Open WebPage.html: Launch the HTML form using a browser automation tool.
4. Read Data from Excel: Open SampleData.xlsx and extract data to be input into the HTML form.
5. Insert Data into Webpage: Enter data from each row into the appropriate HTML form fields.
6. Submit the Form: Submit the form on the HTML page after entering the data.
7. Capture Passcode: Extract the passcode generated by the HTML page after submission.
8. Update Excel File: Write the captured passcode into the corresponding cell in the Excel file.
9. Handle Errors (if any): If incorrect data or a system error occurs
10. Generate a Business Exception.

11. Log the error and continue the process for the next row of data.
12. End Loop (All Rows Processed): Complete the process for all rows in the Excel file.
13. Log Start and End Times: Log the start time before the process begins and the end time after all rows are processed.
14. Calculate Execution Time: Calculate the total execution time (end time - start time).
15. Display Execution Time: Show the execution time in a message box or log.
16. End.

### **Tools Used:**

- **UiPath Studio:** For designing the automation workflow. For designing the automation workflow.
- **Excel:** For input data and storing results.
- **HTML & JavaScript:** Used for the web form where data is entered and passcode is generated.

## **Implementation:**

### **1. Open the HTML Page**

- Used the "Start Process" activity in UiPath to launch the default browser and open WebPage.html.
- Ensured compatibility across browsers by setting up selectors dynamically.

### **2. Read Data from Excel**

- Utilized "Excel Application Scope" with "Read Range" to extract data from SampleData.xlsx.
- Verified column names and data types to ensure consistency.

### **3. Data Entry into HTML Page**

- Implemented a "For Each Row" loop to iterate through the data.
- Mapped each column in the Excel file to the corresponding field in the HTML form using "Type Into".
- Clicked the "Generate Passcode" button using the "Click" activity.

#### **4. Capture Passcode**

- Used the "Get Text" activity to extract the passcode generated on the HTML page after form submission.
- Stored the passcode in a variable for later use.

#### **5. Update Excel File**

- Used "Write Cell" to update the respective row in SampleData.xlsx with the captured passcode.
- Applied error handling to ensure that file access conflicts (e.g., file being open) were avoided.

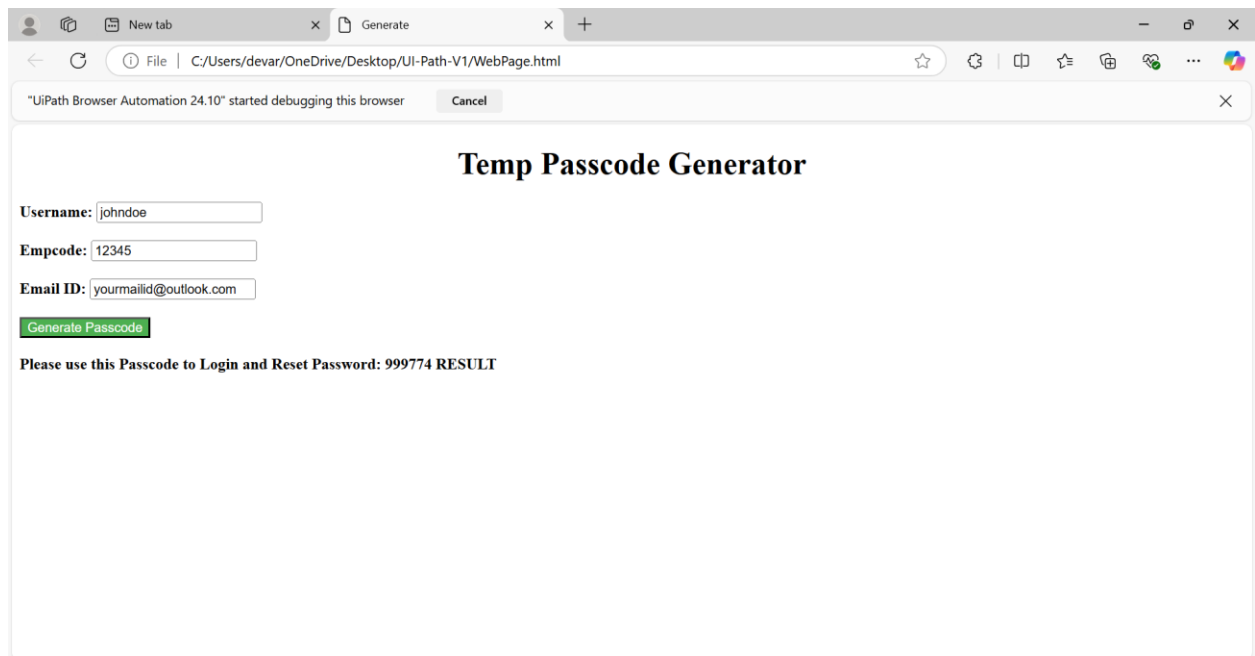
#### **6. Error Handling**

- Implemented a "Try-Catch" block around critical activities like data entry and passcode capture.
- If invalid input data was detected (e.g., empty fields), a "Business Rule Exception" was triggered.
- Logged errors using the "Log Message" activity.

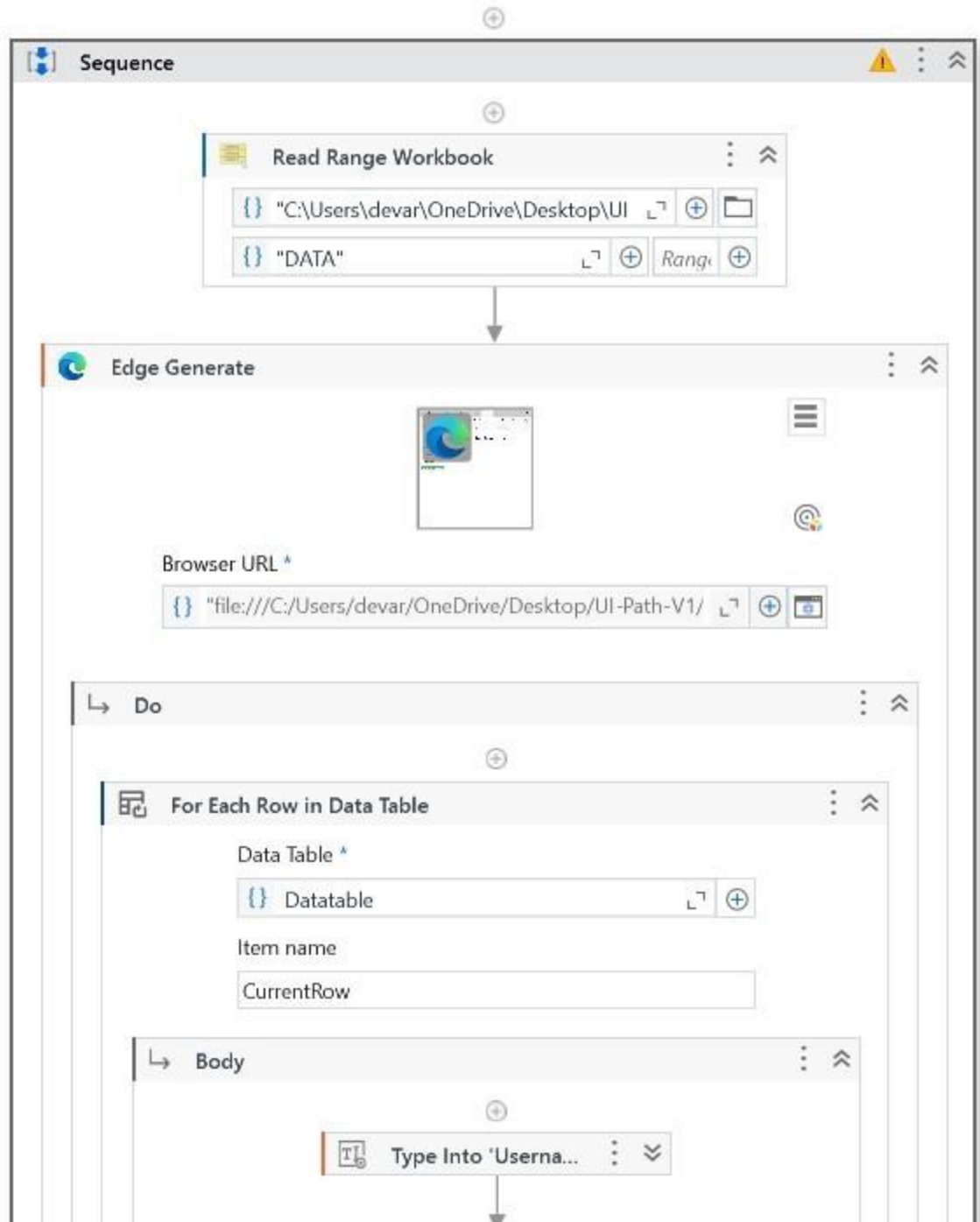
## 7. Logging and Execution Time

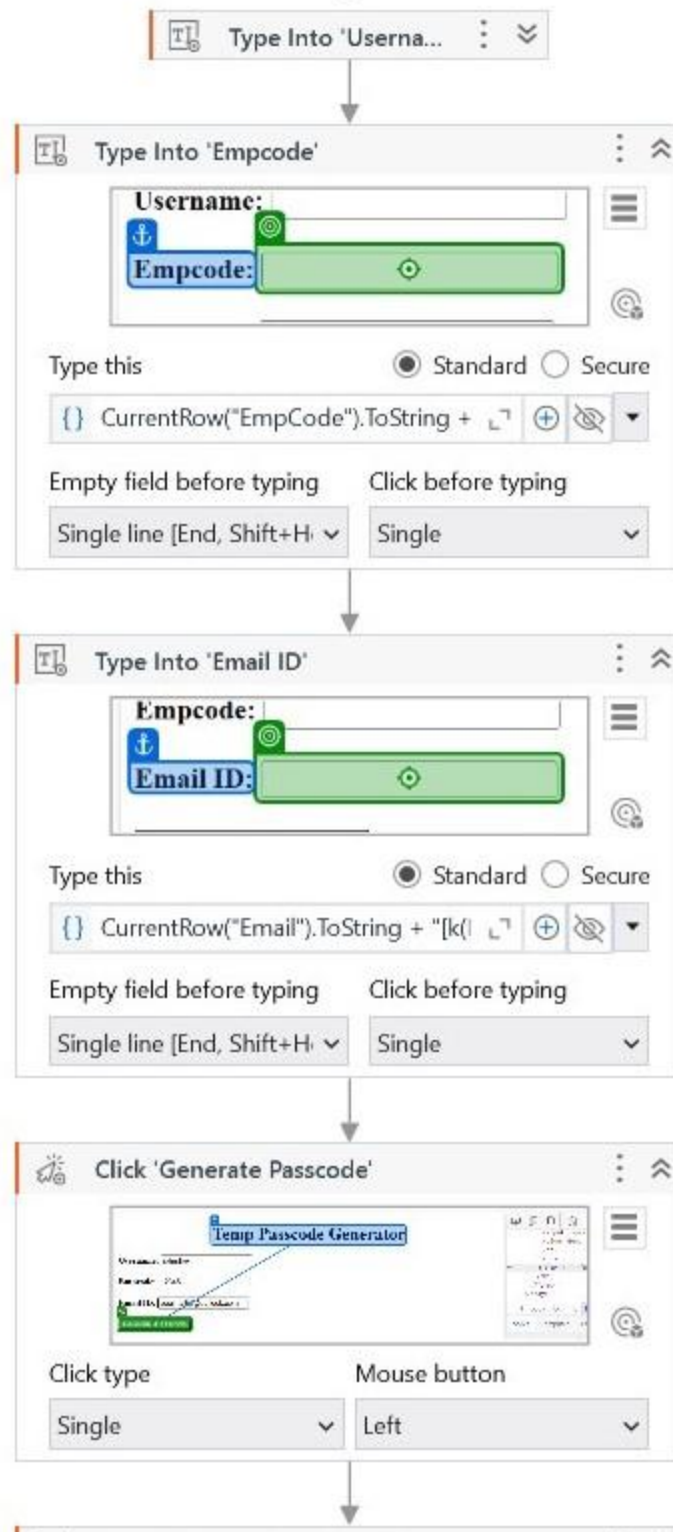
- Captured the current time at the beginning and end of the workflow using the Now variable.
- Calculated execution time and displayed it in a "Message Box".
- Logged both start and end times for audit purposes.

## Implementation Screenshots:

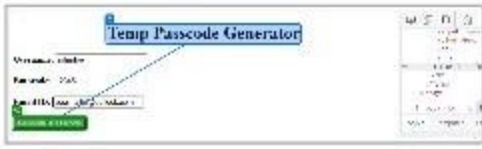









Click 'Generate Passcode'



Click type: Single

Mouse button: Left

Get Text 'Any Word Group'



Save to: {} out

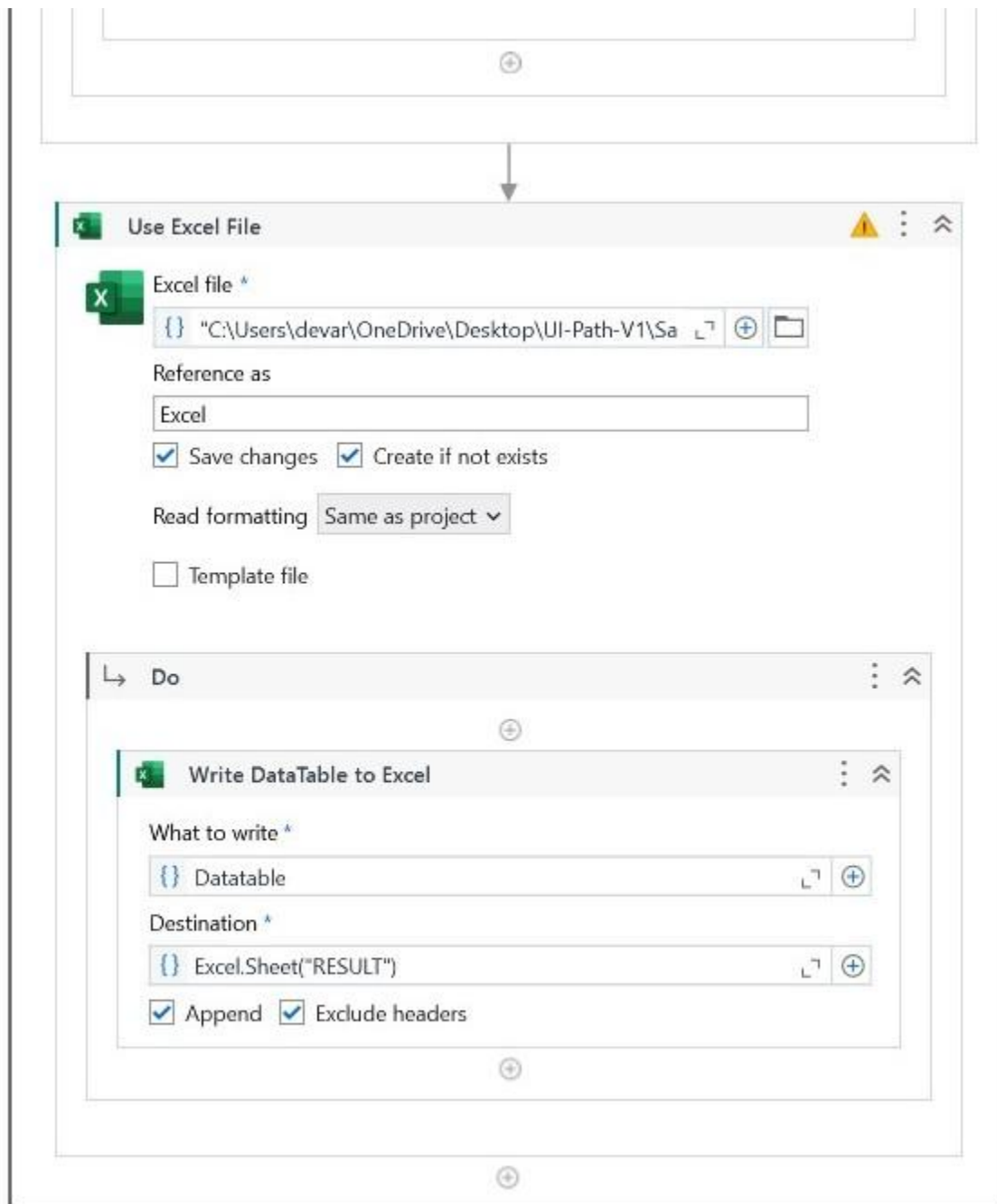
Update Row Item

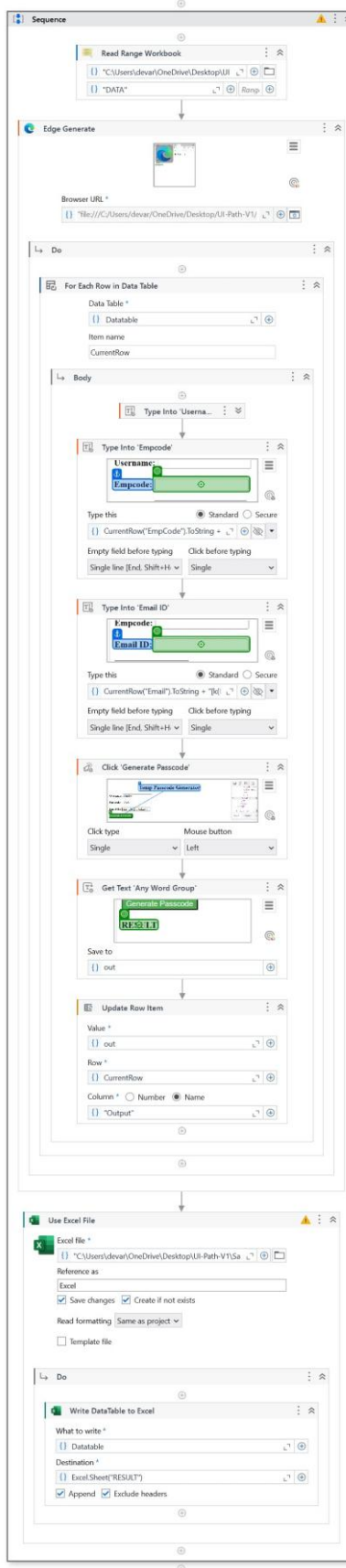
Value \*

Row \*

Column \* ☐ Number ☒ Name

{} "Output"







## **Challenges and Solution :**

### **1. Handling Missing or Invalid Data**

**Challenge:** Excel might contain missing or invalid data, leading to form submission errors.

**Solution:**

Validated data before entry. For empty or incorrect data, a Business Rule Exception was triggered.

**Example:**

```
If String.IsNullOrEmpty(row("FieldName").ToString) Then
    Throw New BusinessException("Invalid input data detected.")
End If
```

### **2. Selector Stability for HTML Elements**

**Challenge:** Changes in the browser or HTML structure could break the selectors.

**Solution:**

Used dynamic selectors in UiPath with anchors where necessary.

Verified and tested selectors on multiple browsers for reliability.

### **3. Concurrent File Access**

**Challenge:** Simultaneous access to the Excel file (e.g., file left open) caused write conflicts.

**Solution:**

Used the "Excel Application Scope" to lock the file during processing.

Added a retry mechanism to handle temporary access issues.

### **4. Execution Time Measurement**

**Challenge:** Accurately capturing and calculating execution time.

**Solution:**

Logged start and end times using Now.

Used `EndTime.Subtract(StartTime)` to compute the duration.

### **5. Browser Compatibility**

**Challenge:** HTML pages behaved differently across browsers.

**Solution:**

Standardized testing in a single browser (e.g., Chrome).

Ensured flexibility by using robust selectors.



## Certificate :



# Diploma of Completion

Proudly presented to

**Abishek B**

For successfully completing the learning plan

**Naan Mudhalvan Robotic Process Automation Foundation Course for  
Engineering Students**

14/11/2024

Date of Issue

A blue, round, cartoonish character with a yellow antenna and red arms, appearing to be in motion.  
*Daniel Dines*

Daniel Dines  
UiPath CEO & Founder

## **Conclusion :**

Using UiPath, the RPA solution successfully automated the data entry process, passcode retrieval, and Excel updates, showcasing the platform's capability to handle repetitive tasks efficiently. The integration of robust error handling through Business Rule Exceptions ensured that incorrect data was managed gracefully without interrupting the workflow.

Additionally, logging the execution time and tracking the process's start and end times provided clear visibility into performance metrics. The use of UiPath's activities, such as "Read Range," "Type Into," "Get Text," and "Write Cell," demonstrated how the platform simplifies complex automation challenges while maintaining accuracy and reliability.

Overall, UiPath enabled the seamless execution of this automation, significantly reducing manual effort, minimizing errors, and enhancing overall productivity. The workflow can be further enhanced by incorporating advanced features such as automated reporting or real-time data validation to make the solution even more robust and scalable.