Lab Manual

of

Term Work

In

Robotic Process Automation

By

Drumil Kotecha
C052
BTech Integrated
Computers

Under the supervision of

Prof. Deepak Kumar Sinha
(Department of Mechatronics Engineering, MPSTME)

SVKM's NMIMS University

(Deemed-to-be University)

MUKESH PATEL SCHOOL OF TECHNOLOGY MANAGEMENT & ENGINEERING
Vile Parle (W), Mumbai-56

2024-2025

INDEX

Date of Experiment: 10/01/25
Date of Submission: 24/01/25

SVKM'S NMIMS
Mukesh Patel School of Technology Management & Engineering
Department of Mechatronics Engineering
AR-VR Lab
Subject- Robotic Process Automation
EXPERIMENT NO. 1

Aim:
The objective of this experiment is to assess the installation process and download efficiency of Power Automate Desktop, a robust automation tool developed by Microsoft. This experiment aims to provide insights into the installation process.

Materials:
1. Computer system with minimum requirements for Power Automate Desktop.
2. Internet connection.
3. Access to Microsoft's official website for Power Automate Desktop.

Procedure:

1. Preparation:
- Ensure that the computer system meets the minimum requirements specified by Microsoft for Power Automate Desktop.
- Connect the computer to a stable and reliable internet connection.

2. Access Microsoft's Official Website:
- Open a web browser and navigate to Microsoft's official website or the Power Automate Desktop product page.

3. Navigation to Download Section:
- Locate the download section on the website.
- Identify the correct version and edition of Power Automate Desktop that is suitable for your system.

4. Initiate Download:
- Click on the download link or button to start the download process.
- Monitor the download progress and note the time taken for the download to complete.

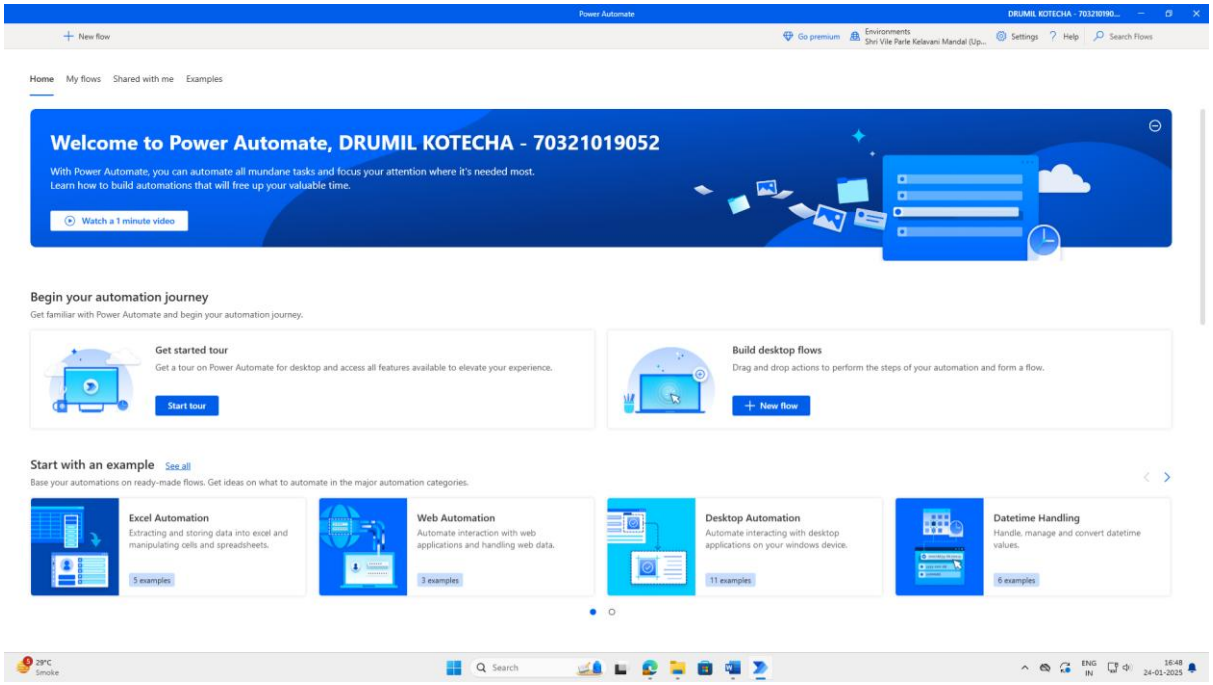
5. Installation:
- Once the download is complete, initiate the installation process.
- Follow the installation wizard, providing any necessary information and configuring settings as required.
- Document the time taken for the installation process to finish.

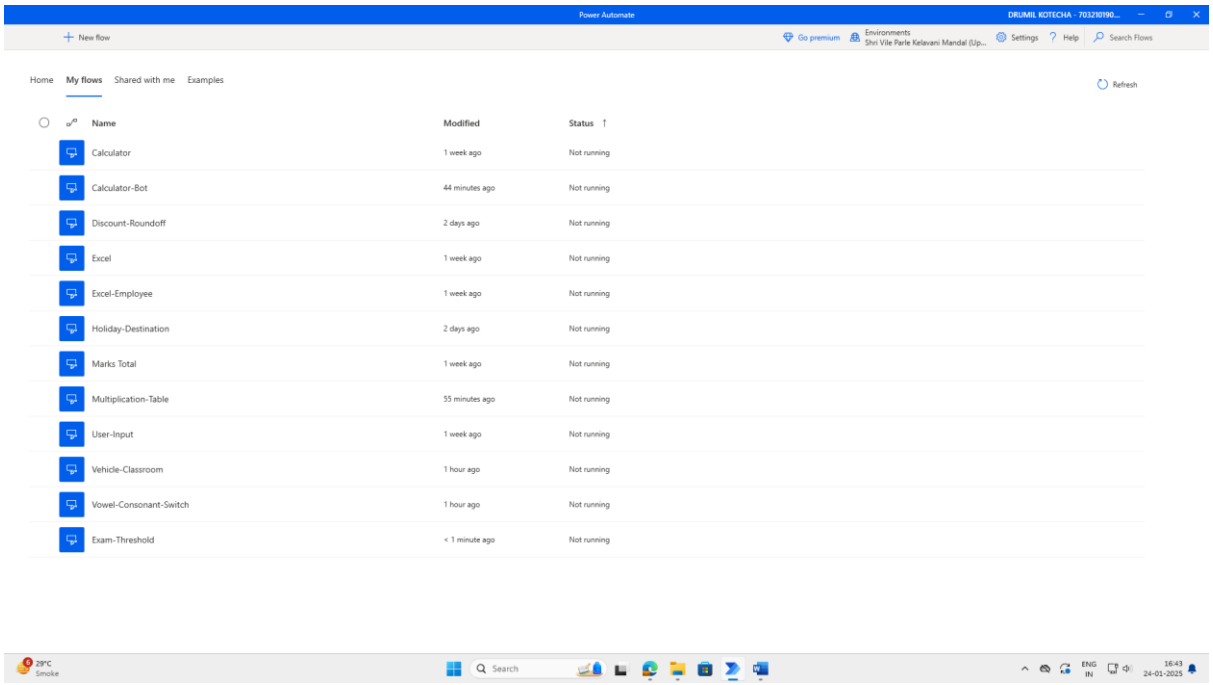
6. User Experience Evaluation:
- Assess the user interface and clarity of instructions provided during the installation.

- Note any issues or difficulties encountered during the installation process.
- Evaluate the user-friendliness of the software.

7. Verification and Testing:

- After installation, launch Power Automate Desktop to ensure that it functions as expected.
- Create a simple automation task to test the software's capabilities.
- Document any issues faced during the verification and testing phase.





Conclusion:

In conclusion, I successfully completed the installation and setup of Power Automate Desktop, which provided a comprehensive understanding of its environment. Through the process, I learned how to boot up the Power Automate interface, navigate its features, and initiate new automation flows. I was able to create simple bots, such as automating tasks with lists, working with Excel files, calculating averages, and implementing conditional logic (if-else). The user-friendly interface and clear instructions made it easy to create basic automation tasks, which enhanced my understanding of Robotic Process Automation (RPA) tools and their applications. This experiment has laid a strong foundation for further exploration and development of more complex automation solutions using Power Automate.

Date of Experiment: 24/01/25
Date of Submission: 24/01/25

SVKM'S NMIMS
Mukesh Patel School of Technology Management & Engineering
Department of Mechatronics Engineering
AR-VR Lab
Subject- Robotic Process Automation
EXPERIMENT NO. 2A

Objective:
This lab manual aims to reinforce your understanding of Power Automate Desktop, emphasizing variables for data manipulation and conditional actions for dynamic decision-making, ultimately enabling you to construct more flexible and efficient automation flows.

Prerequisites:
1. Power Automate Desktop installed on your computer.
2. Basic understanding of Power Automate Desktop interface.

Challenge Overview:
In this experiment, you will create 3 flows
1.Takes input for three subject marks, calculates their average.
2.Checks if the number is above the passing threshold
3.Assigns grades based on the number input from the user.

Important Actions:
1. Input Dialog:
- Use the "Input Dialog" action to prompt the user to enter marks for three subjects.
- Configure the input dialog to request numerical input for each subject.

2. Calculate Average:
- Utilize the "Set Variable" action to sum the three subject marks obtained from the user.
- Divide the sum by 3 to calculate the average.

Hint:enclose the calculation in "%%" any thing inside this will be evaluated.

3. Decision:
- Insert a "Decision" action to evaluate whether the average is above the passing threshold.
- Configure the decision to have two branches - one for passing and another for failing.
-You may use If-else or switch case here

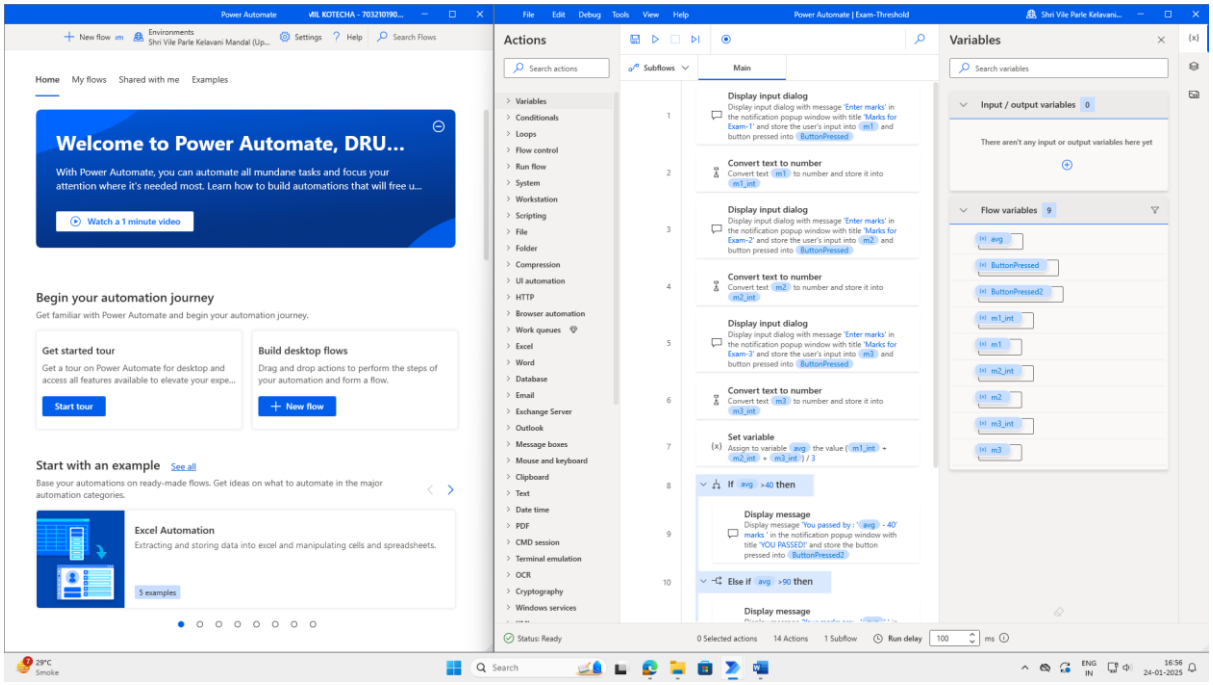
4. Assign Grades:
- In the passing branch, use the "Set Variable" action to assign the appropriate grade based on the average.
- For example, if the average is above a certain value, assign 'A'; if it's between another range, assign 'B', and so on.
- In the failing branch, you may choose to assign an 'F' grade.
-You may use If-else ladder or switch case here

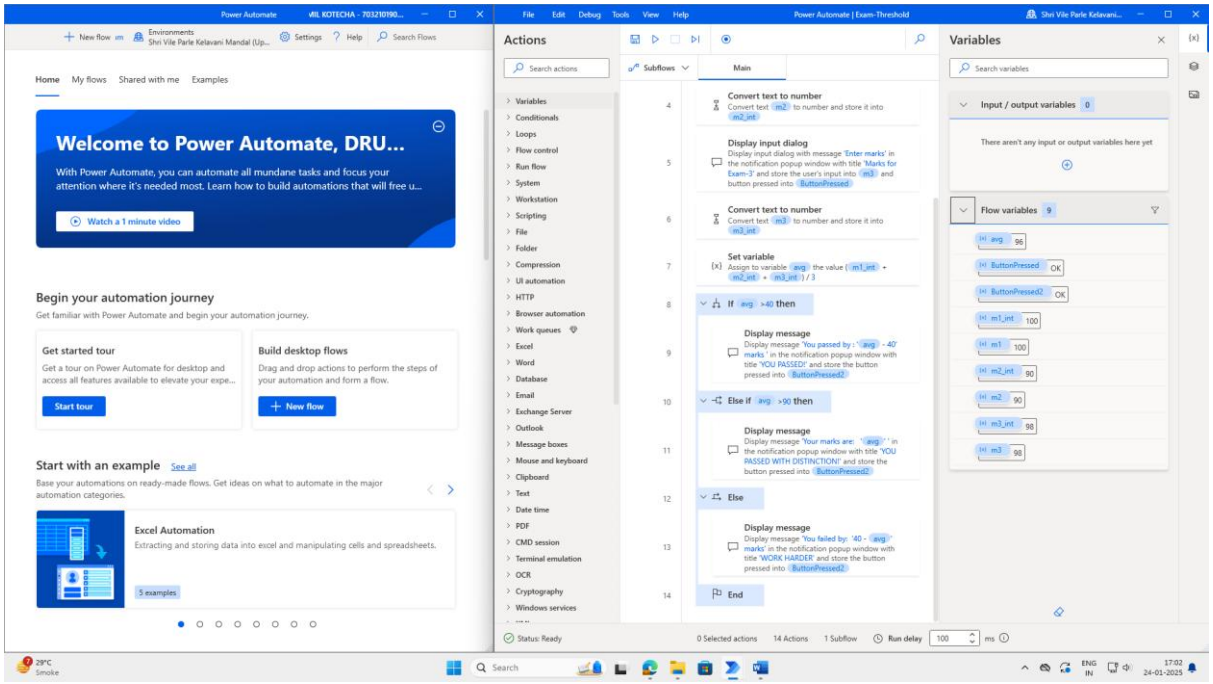
5. Display Results:
- Use the "Message Box" action to display the calculated average and assigned grade to the user.

Tasks:

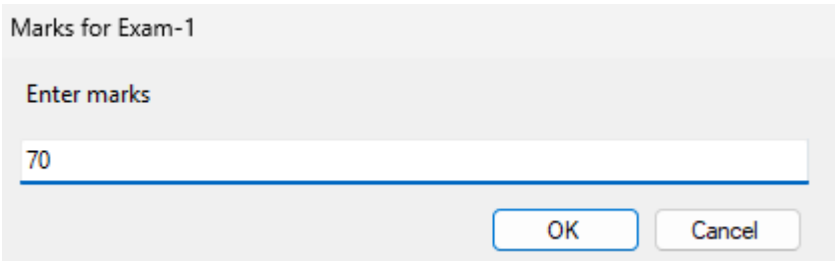
1. Create a Power Automate Desktop flow that incorporates the described actions.
2. Test the flow by providing different sets of marks to ensure accurate calculation and grade assignment.
3. Debug and troubleshoot any errors that may arise during the execution of the flow.
4. Optimize the flow for efficiency, considering factors such as readability and simplicity.

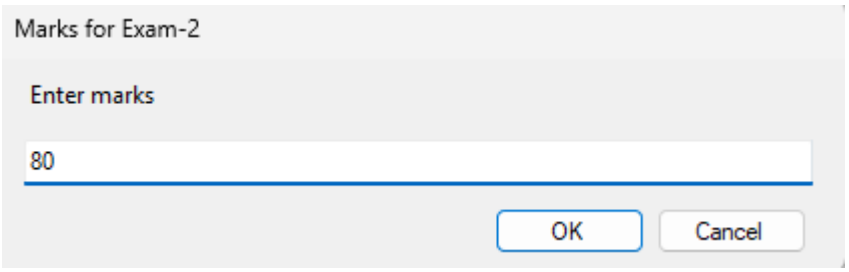
Flow Screenshots:

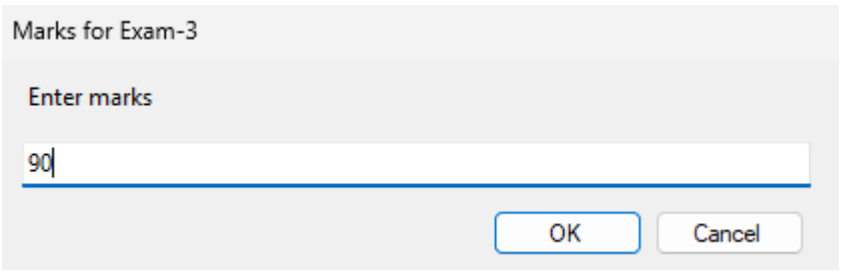


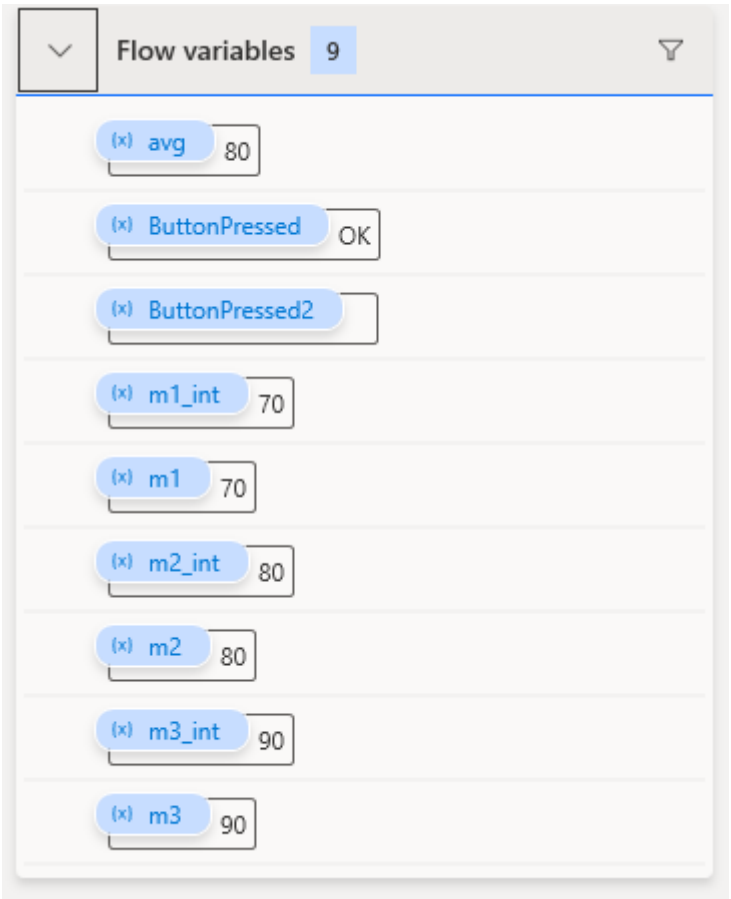


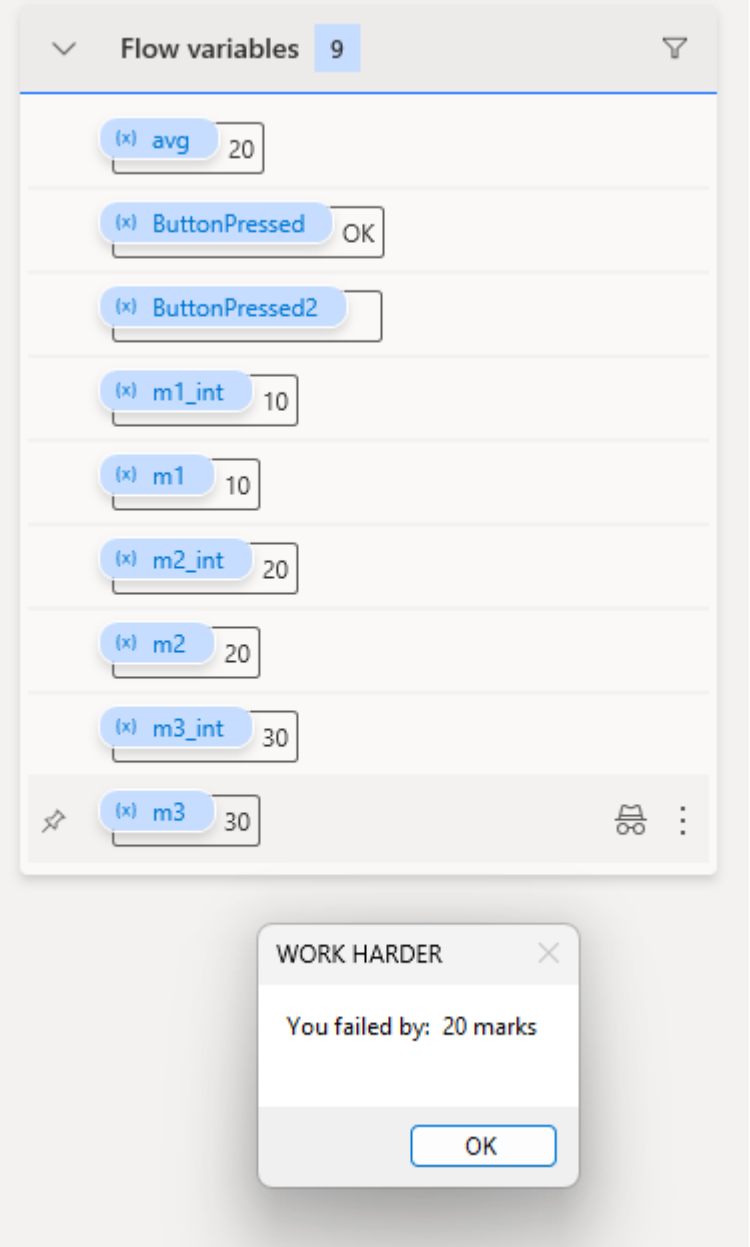
Input Screenshots:


Marks for Exam-1
Enter marks
70
OK
Cancel

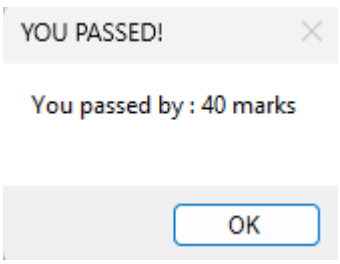

Marks for Exam-2
Enter marks
80
OK
Cancel

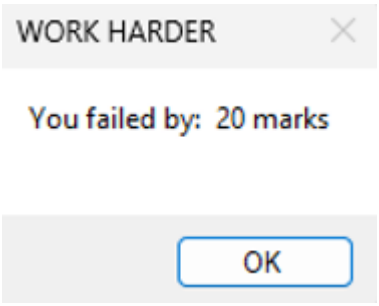

Marks for Exam-3
Enter marks
90
OK
Cancel


Flow variables 9
(x) avg 80
(x) ButtonPressed OK
(x) ButtonPressed2
(x) m1_int 70
(x) m1 70
(x) m2_int 80
(x) m2 80
(x) m3_int 90
(x) m3 90



Output Screenshot:





Conclusion:

I gained a clear understanding of how to apply if-else conditions in Power Automate by developing a flow for grade calculation. Additionally, I utilized various utility actions, such as the 'Convert text to number' action, to facilitate performing necessary calculations and operations within the flow.

Lab Manual

Date of Experiment:12/02/2025
Date of Submission:12/02/2025

SVKM'S NMIMS
Mukesh Patel School of Technology Management & Engineering
BTI
Subject- Robotic Process Automation
EXPERIMENT NO. 03

Objective:
This lab is designed to deepen your understanding of loops in Power Automate Desktop. The objective is to learn how to utilize loops for repetitive tasks, enabling you to create more efficient and dynamic automation flows.

Prerequisites:

1. Power Automate Desktop installed on your computer.
2. Basic understanding of Power Automate Desktop interface.

Challenge Overview:

In this experiment, you will create 3 flows
1.Create a flow that checks whether a given input is a palindrome or not. A palindrome is a sequence that reads the same forwards and backward.
2.Build a flow that generates the multiplication table for a given number.

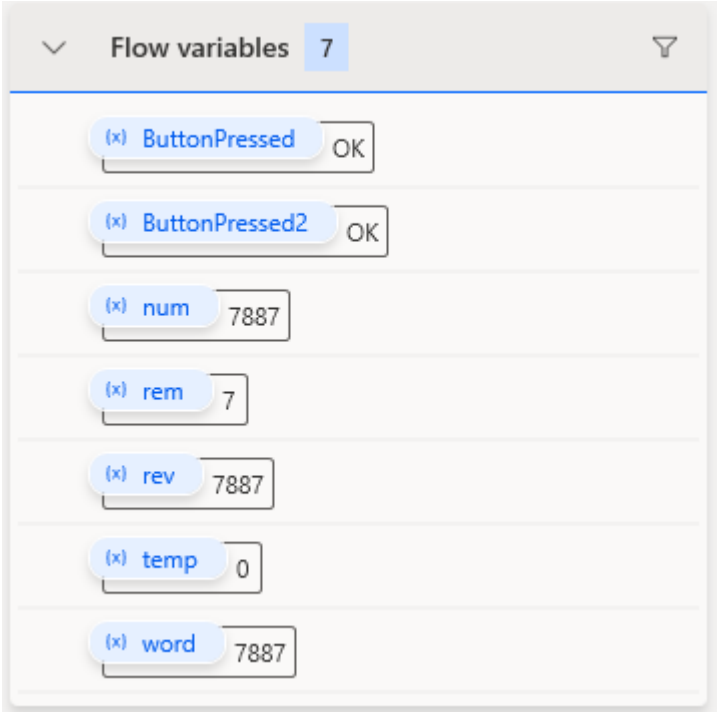
Important Actions:

1. **Input Dialog:**
   - **Purpose:** Captures user input during the flow execution.
   - **Usage:** Request information, such as text or numerical values, from the user.
2. **Message Box:**
   - **Purpose:** Displays messages, alerts, or results during the flow execution.
   - **Usage:** Communicate information to the user or provide feedback on the flow's progress.
3. **Set Variable:**
   - **Purpose:** Assigns a value to a variable.
   - **Usage:** Store and manipulate data within the flow.
4. **Decision:**
   - **Purpose:** Branches the flow based on a specified condition.
   - **Usage:** Enables dynamic decision-making within the flow.
5. **Loop (e.g., For Each, While, Do Until):**
   - **Purpose:** Repeats a set of actions multiple times based on a specified condition.
   - **Usage:** Efficiently handle repetitive tasks and iterate through collections of data.
6. **Log Message:**
   - **Purpose:** Records messages in the log for debugging and troubleshooting.
   - **Usage:** Assists in diagnosing issues and understanding the flow's behavior.

Tasks:

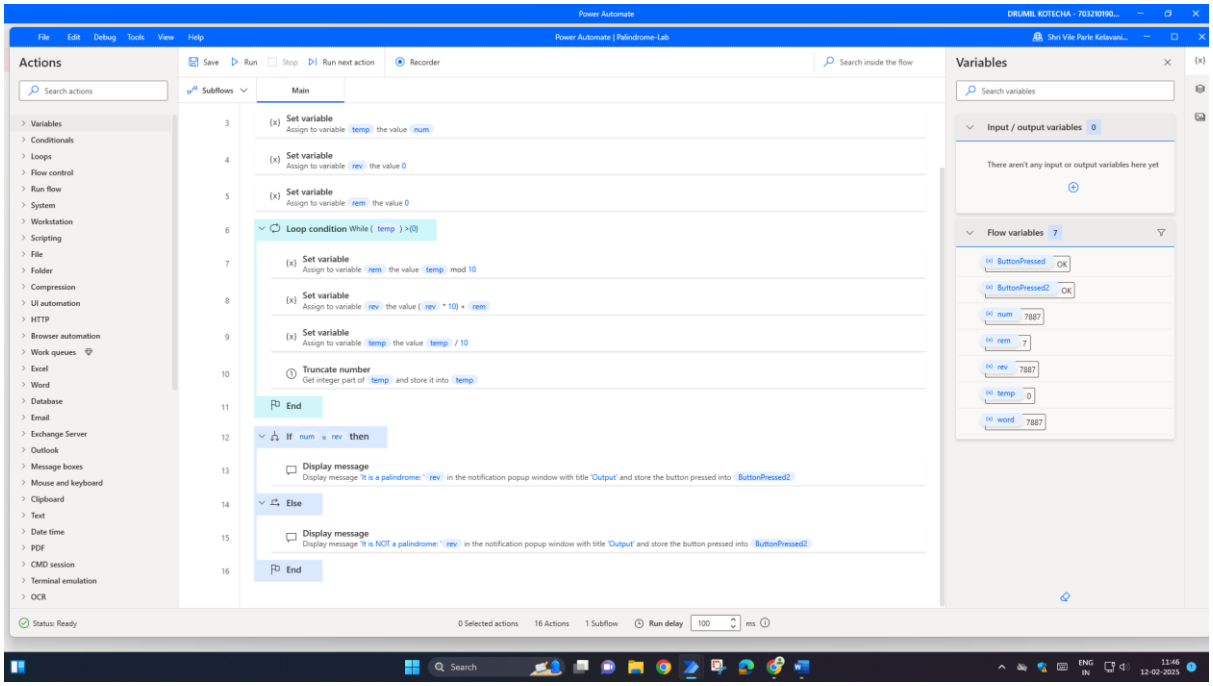
1. Create a Power Automate Desktop flow that incorporates the described actions.
2. Test the flow by providing different sets of Test cases
3. Debug and troubleshoot any errors that may arise during the execution of the flow.
4. Optimize the flow for efficiency, considering factors such as readability and simplicity.

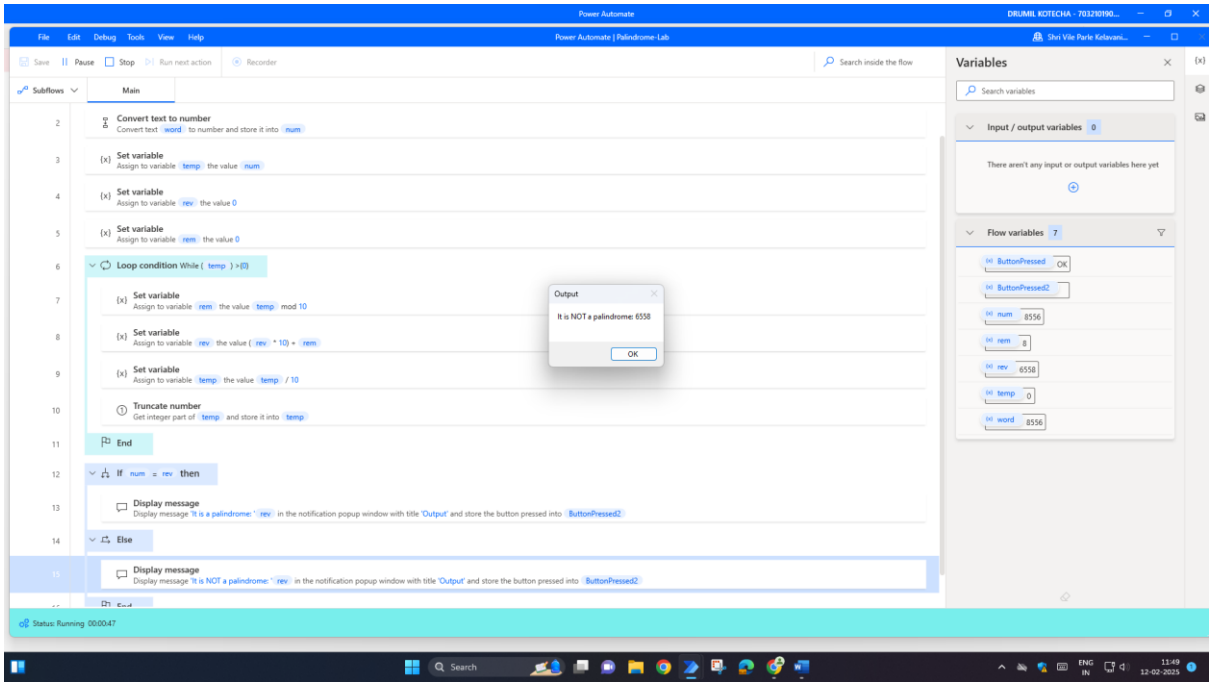
Flow Screenshots:

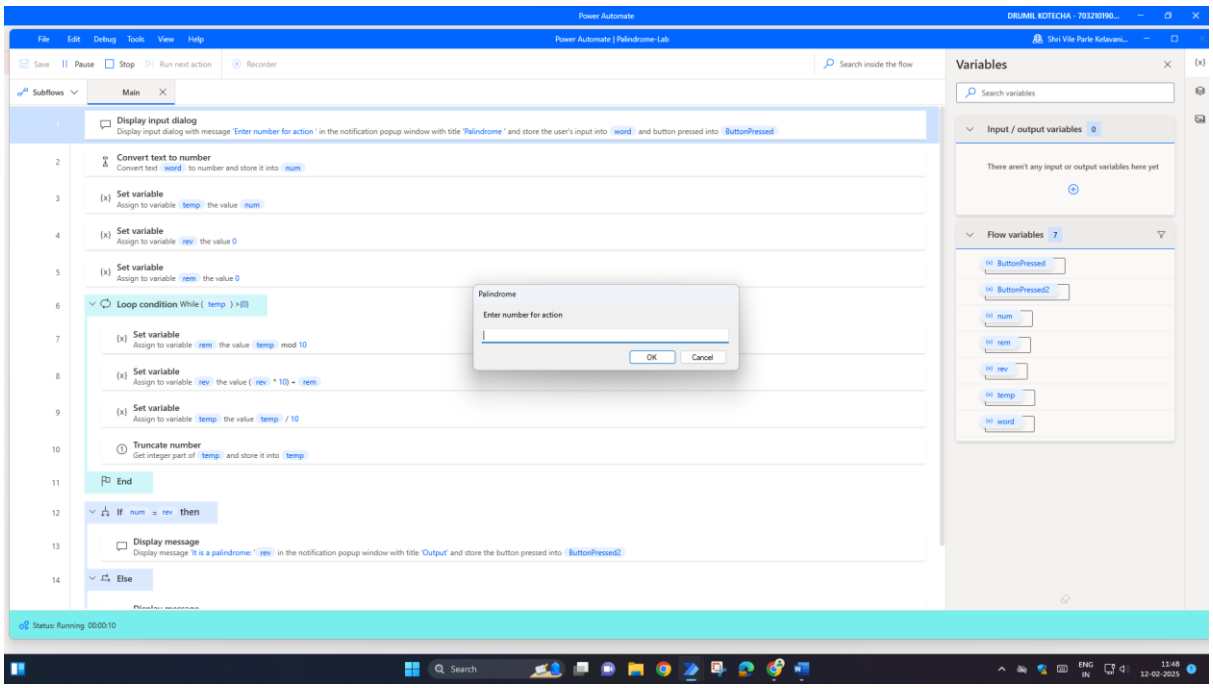


Input Screenshots:

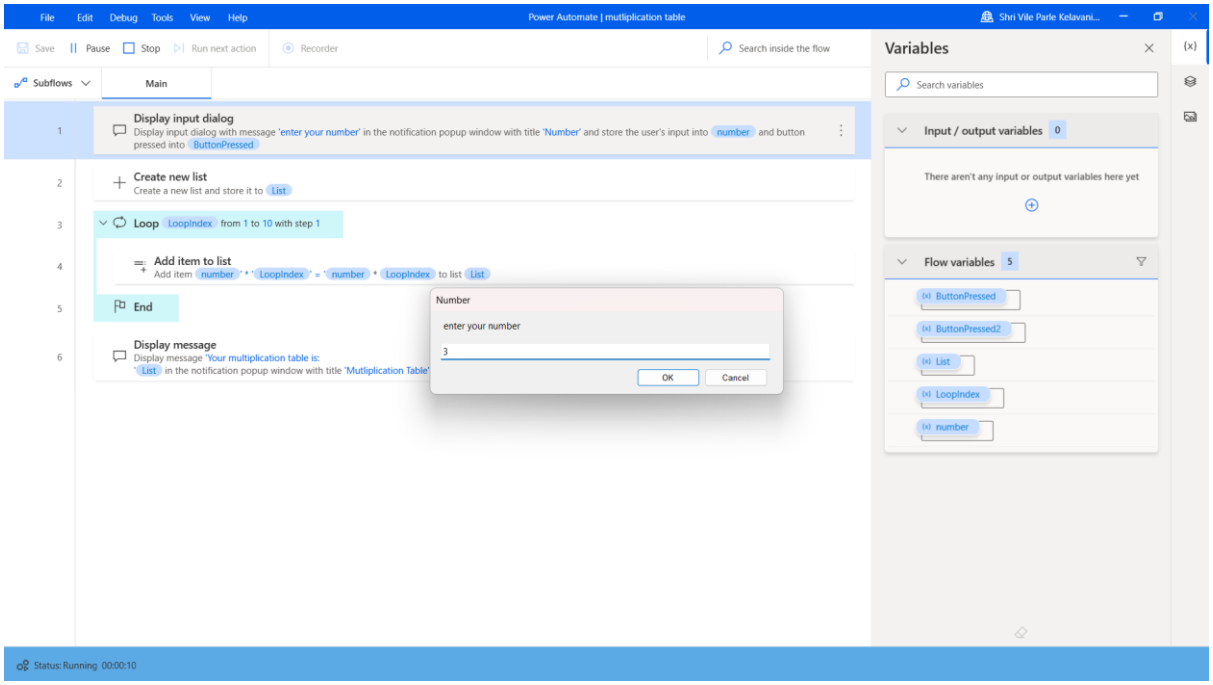
Palindrome:





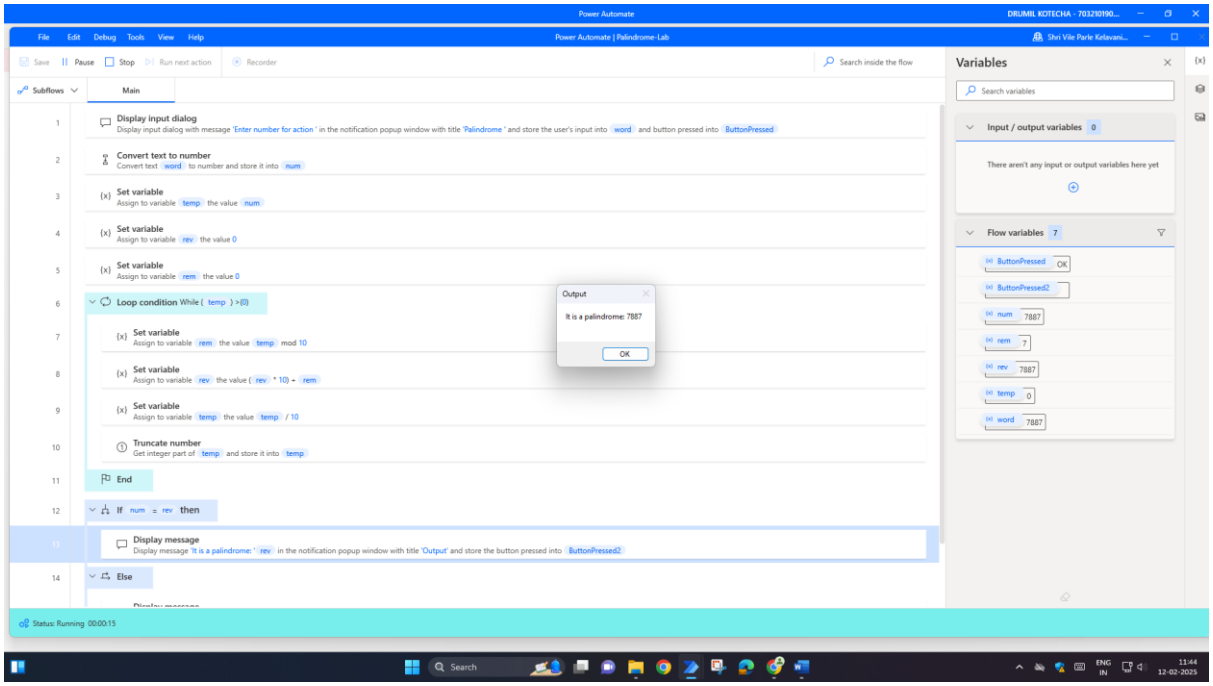


Multiplication Table:

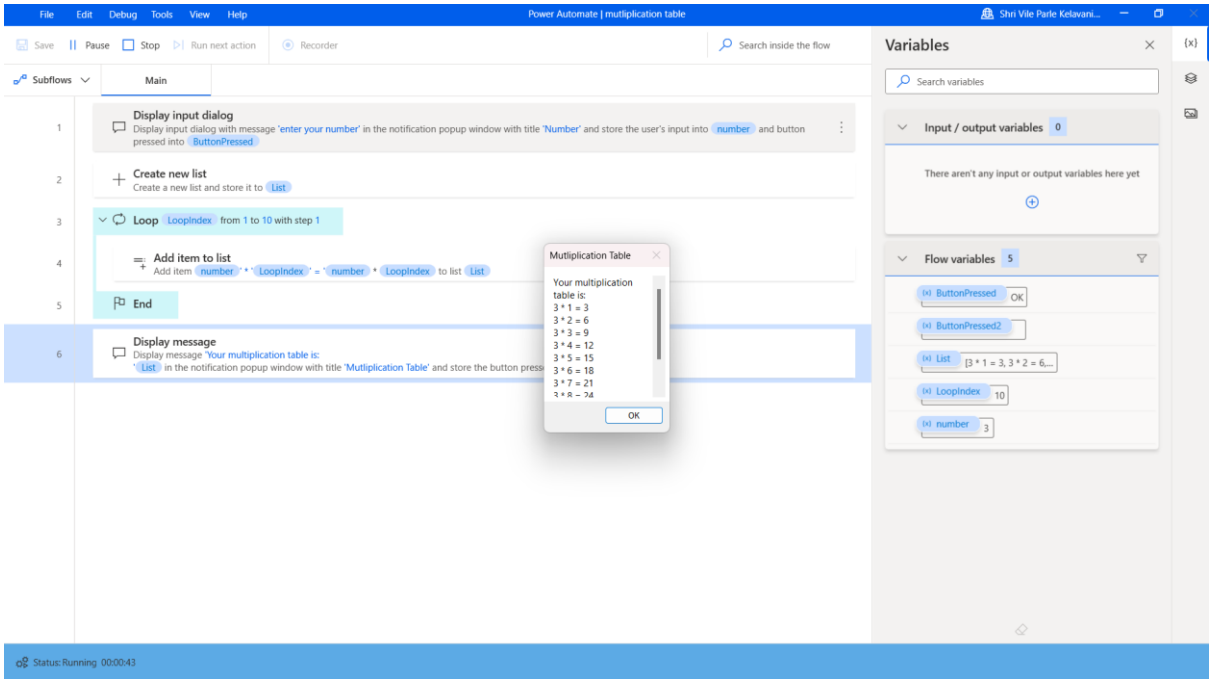


Output Screenshot:

Palindrome:



Multiplication Table:



Conclusion:

In conclusion, the integration of Robotic Process Automation (RPA) with tasks such as generating multiplication tables and checking for palindromes offers a streamlined, efficient approach to automating repetitive processes. By automating the creation of multiplication tables, we save valuable time while ensuring consistency and accuracy in calculations. Additionally, the ability of RPA to check palindromes allows for the quick validation of string data, contributing to error-free operations. This demonstrates how RPA can simplify everyday tasks in education, data analysis, and programming, making processes faster, more reliable, and freeing up human resources for more complex tasks. The use of RPA in these areas showcases its potential to enhance productivity and optimize performance across a variety of domains.

Date of Experiment: 13/02/25
Date of Submission:13/02/25

SVKM'S NMIMS
Mukesh Patel School of Technology Management & Engineering
Department of Mechatronics Engineering
Subject- Robotic Process Automation
EXPERIMENT NO. 4

Objective:
The aim of this experiment is to cultivate expertise in Excel automation through Power Automate Desktop, focusing on efficient data manipulation and task automation.

Prerequisites:
1. Power Automate Desktop installed on your computer.
2. Basic understanding of Power Automate Desktop interface.

Challenge Overview:
In this experiment, you will create 2 flows
1. Develop a Power Automate Desktop flow to identify Armstrong numbers.

Important Actions:
1. **Read Cell:**
   - Purpose: Extract data from specific cells in an Excel sheet.
   - Usage: Retrieve test cases for Armstrong number checking and student information for report card generation.
2. **Calculate:**
   - Purpose: Perform arithmetic operations for evaluating Armstrong numbers and calculating grades.
   - Usage: Utilize mathematical calculations within the flow for precise computations.
3. **Decision:**
   - Purpose: Implement conditional branching based on specified criteria.
   - Usage: Guide the flow to different paths for handling Armstrong number identification and grade assignment.
4. **Set Variable:**
   - Purpose: Assign values to variables for storage and manipulation.
   - Usage: Store interim results or information extracted from the Excel sheet for subsequent actions.
5. **Loop (e.g., For Each, While):**
   - Purpose: Repeat a set of actions for each item in a collection or until a condition is met.
   - Usage: Iterate through test cases and student records for efficient processing.
6. **Write Cell:**
   - Purpose: Input data into specific cells in an Excel sheet.
   - Usage: Update the Excel sheet with results, such as Armstrong number identification and generated report card details.
7. **Message Box:**
   - Purpose: Display messages, alerts, or results during the flow execution.

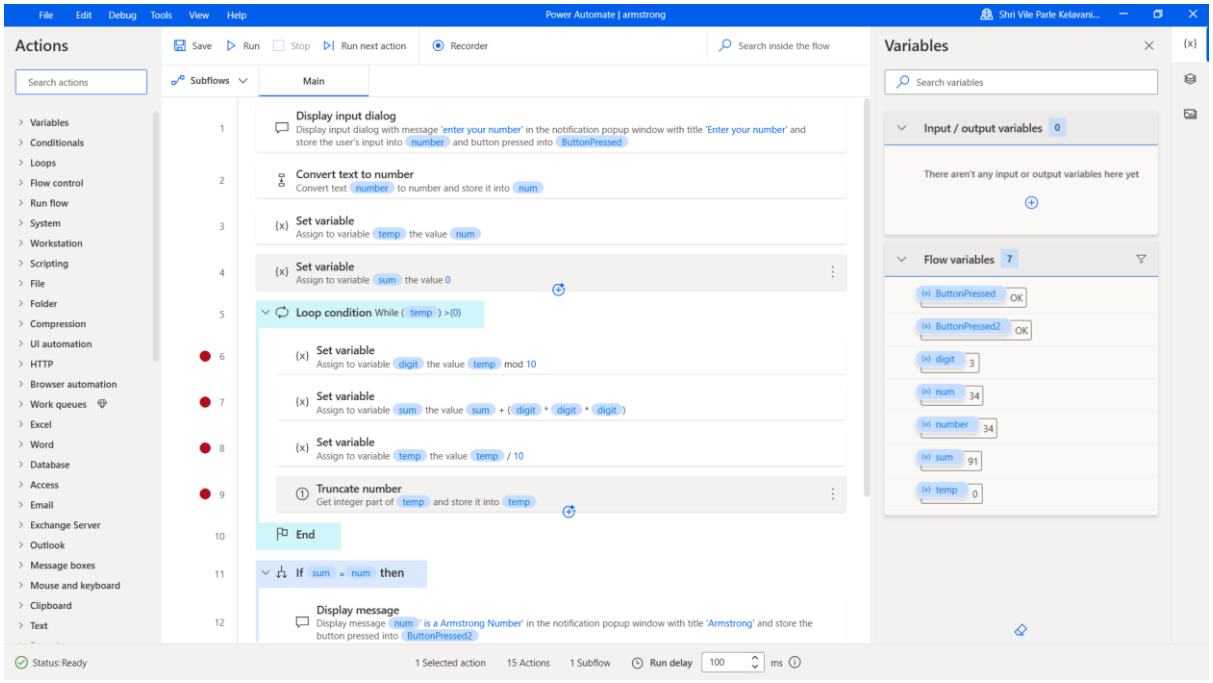
- Usage: Communicate information to the user or provide feedback on the progress of the flow.

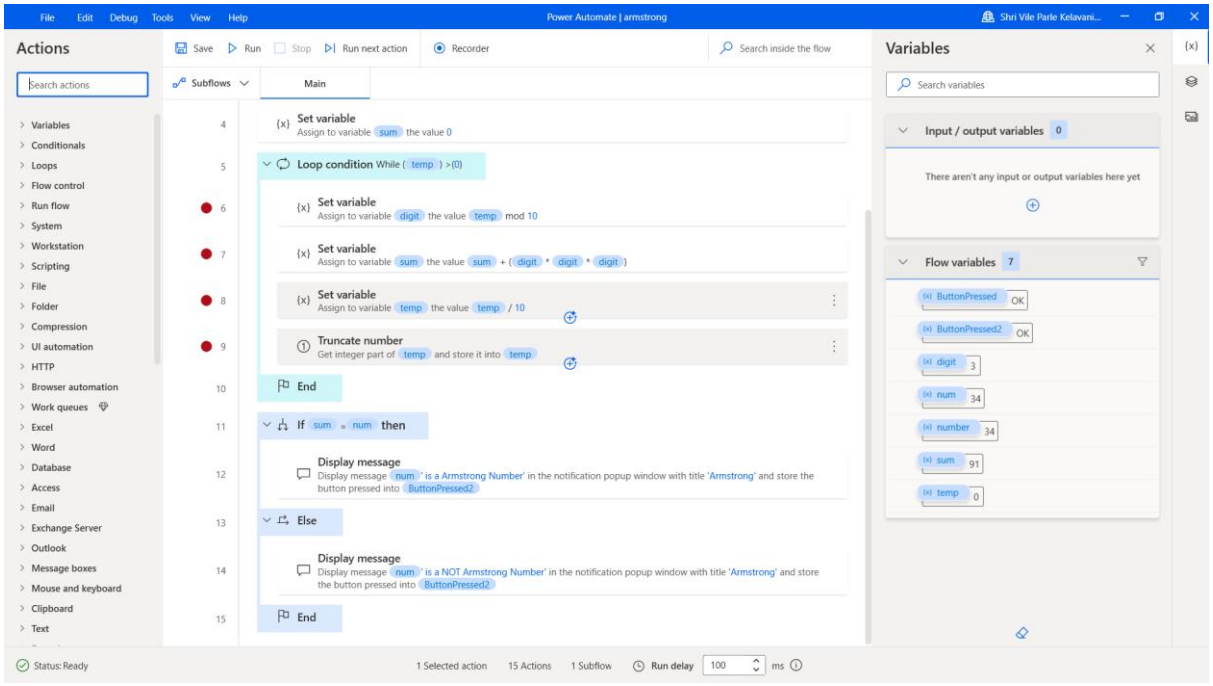
8. **Open Workbook:**
   - Purpose: Open a specified Excel workbook.
   - Usage: Establish a connection with the Excel file to read and write data as needed.
9. **Log Message:**
   - Purpose: Record messages in the log for debugging and troubleshooting.
   - Usage: Assist in diagnosing issues and understanding the flow's behavior, especially during testing and development.

Tasks:

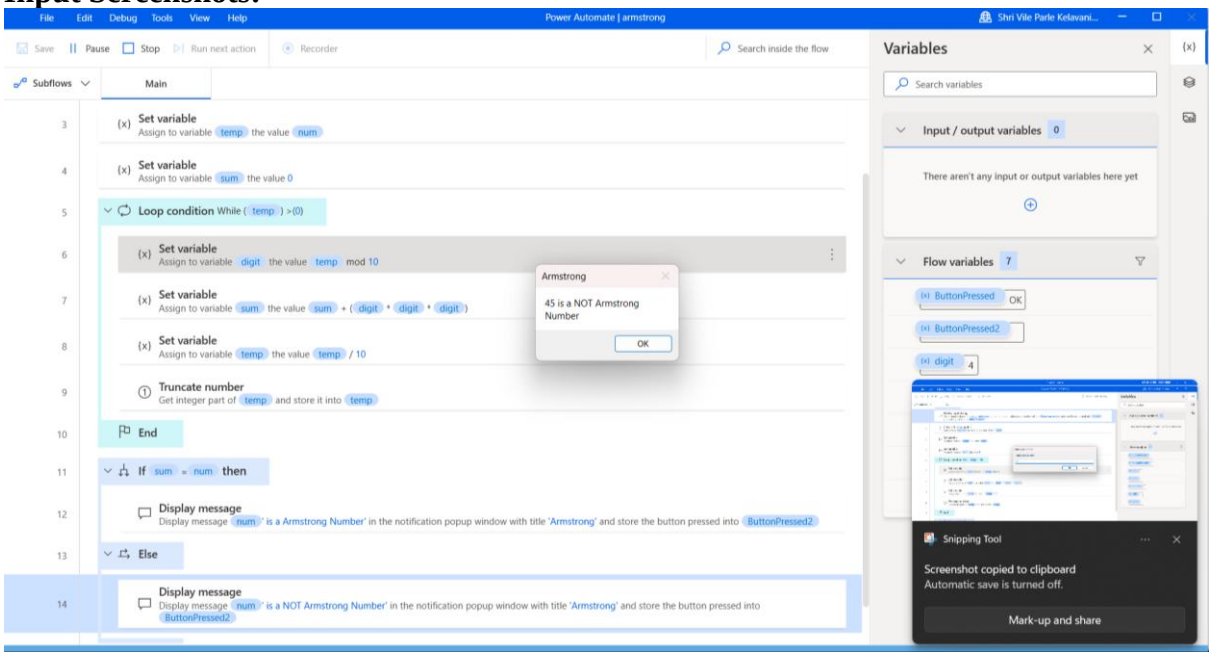
1. Create a Power Automate Desktop flow that incorporates the described actions.
2. Test the flow by providing different sets of marks to ensure accurate calculation and grade assignment.
3. Debug and troubleshoot any errors that may arise during the execution of the flow.
4. Optimize the flow for efficiency, considering factors such as readability and simplicity.

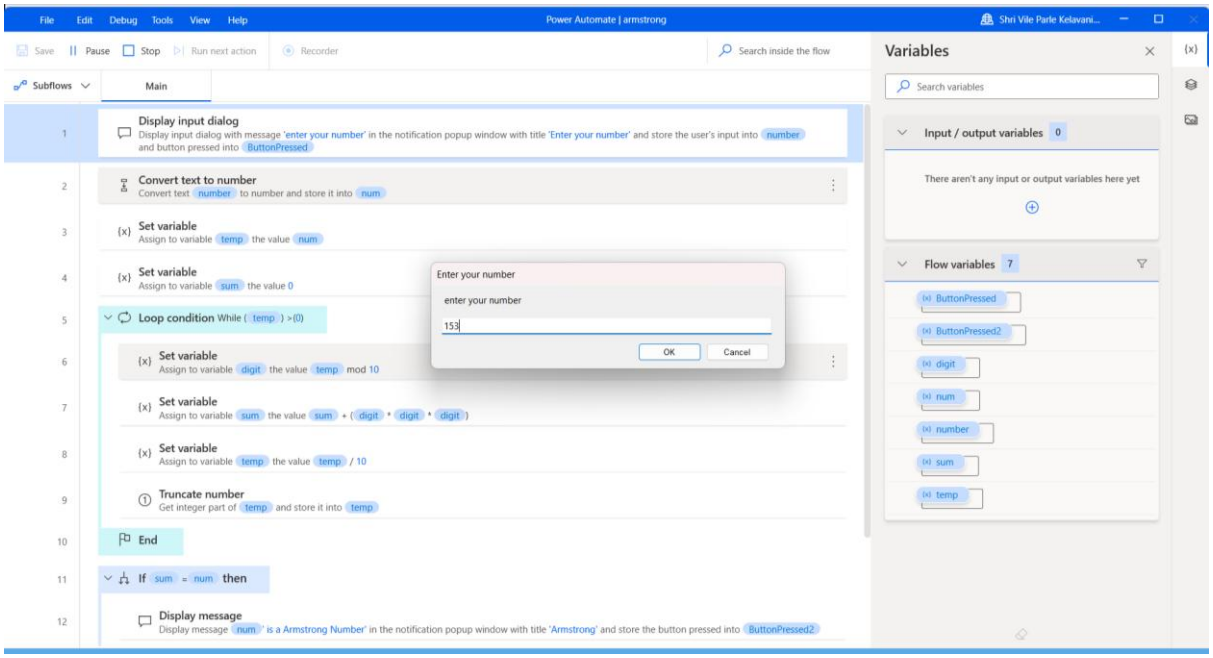
Flow Screenshots:



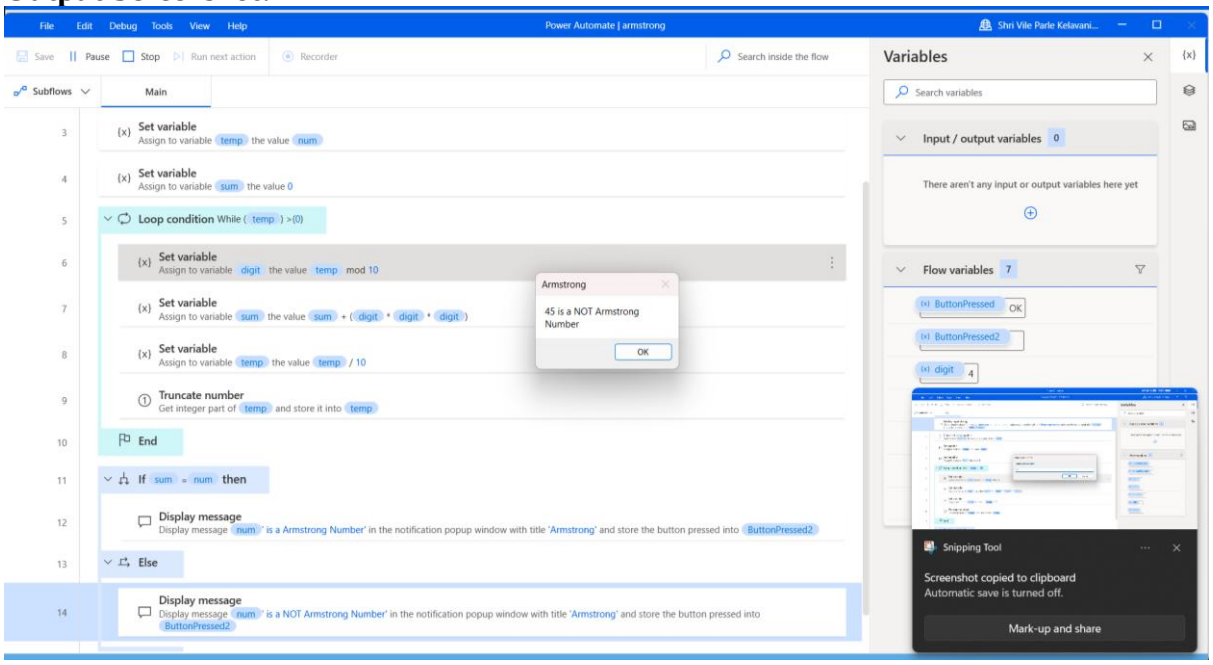


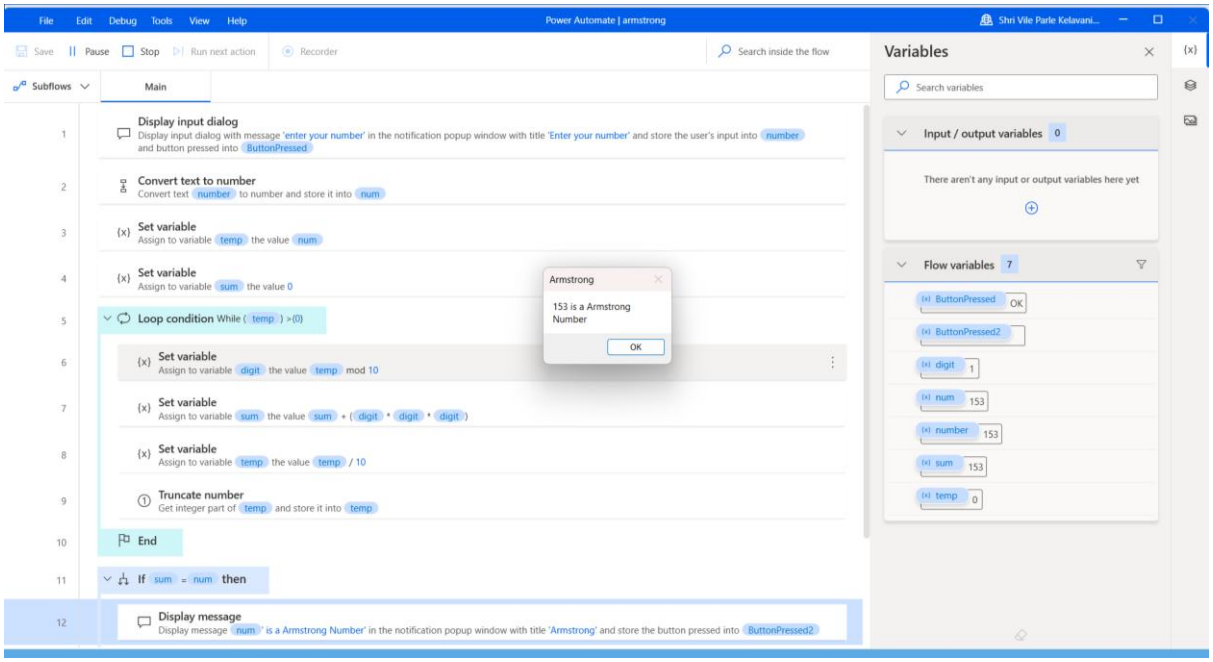
Input Screenshots:





Output Screenshot:





Conclusion:

This approach involves converting the input number into a string and iterating through each digit in the string. For each digit, we raise it to the power of the number of digits in the input number, and sum up the results. If the final sum equals the input number, it is an Armstrong number.

Algorithm→
1. Convert the input number into a string using str(num).
2. Find the length of the string using len(num_str) and store it in n.
3. Initialize a variable sum to zero.
4. Iterate through each digit in the string using a for loop, and convert each digit back to an integer using int(digit).
5. Raise each digit to the power of n using int(digit)**n, and add the result to sum.
6. After the loop is complete, check whether sum is equal to num.
7. If sum is equal to num, return True (the input number is an Armstrong number).
8. If sum is not equal to num, return False (the input number is not an Armstrong number).

And learnt how to implement a bot to check whether a given number is an Armstrongs number or not.

Date of Experiment: 28/02/2025
Date of Submission: 04/03/2025

SVKM'S NMIMS
Mukesh Patel School of Technology Management & Engineering
Department of Mechatronics Engineering
RPA Lab
Subject- Robotic Process Automation
EXPERIMENT NO. 5

Objective:

The objective of this experiment is to design and implement an automated bot using Power Automate Desktop for scraping the data from a flight booking website and organizing the information into an Excel sheet. The experiment aims to showcase proficiency in web scraping, data manipulation, and automation through the creation of a streamlined and efficient data extraction and processing workflow.

Prerequisites:

1. Power Automate Desktop installed on your computer.
2. Basic understanding of Power Automate Desktop interface.

Challenge Overview:

The challenge involves designing an automated bot to extract comprehensive data for flight booking from the website e.g, http://makemytrip.com. The bot fills in the information available on the excel sheet and fills it in the respective fields on the website. Further, it extracts the data from the website and the extracted information will be organized into an Excel sheet with dedicated columns for different attributes. The ultimate challenge lies in submitting the details of the highest and lowest price of the flight for the specified data.

Task List: Flight Fare Scraping Bot Design

1. **Scraping Data:**
   - Develop a web scraping bot to extract data from a flight booking website.
   - Create a structured data collection process for accurate retrieval.
2. **File Creation:**
   - Generate an excel file with a dataset for flight booking.
3. **Excel Data Entry:**
   - The extracted information must be filled in the excel sheet for the respective search happened in the previous step.
4. **Finding the Max/Min Fare:**
   - Calculate the Maximum and Minimum fares and save it in a excel file.

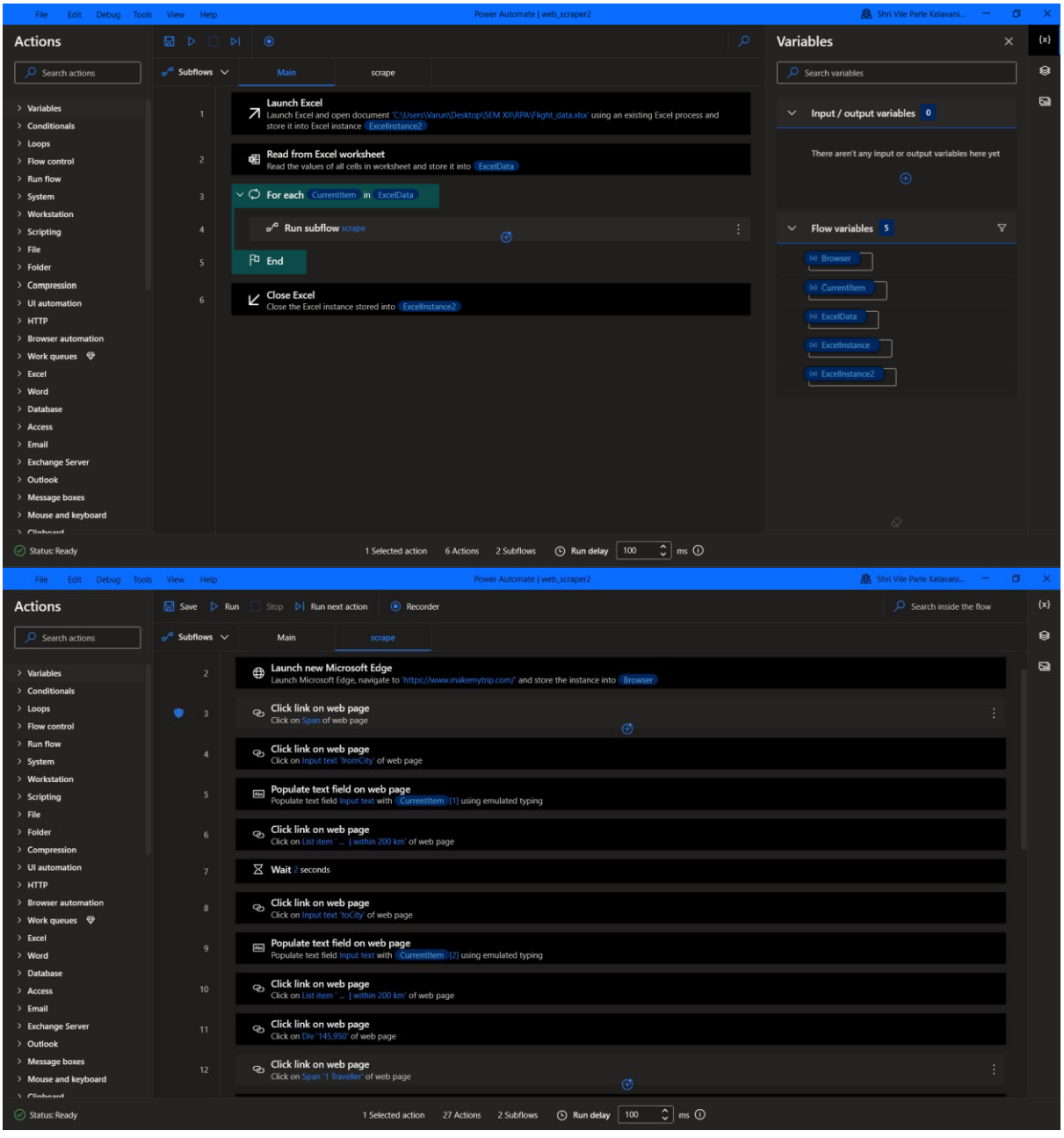
This task list outlines the steps involved in designing a bot for scraping flight details from the information available in the excel file and writes back the scrapped data into it.

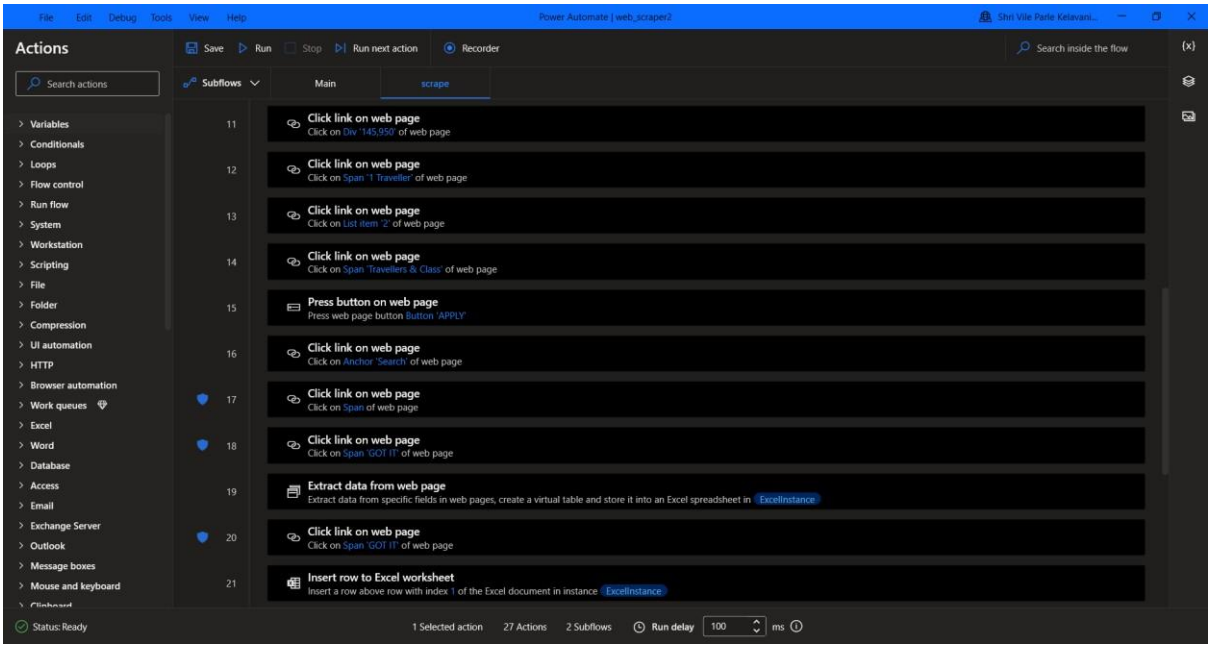
Important Actions:

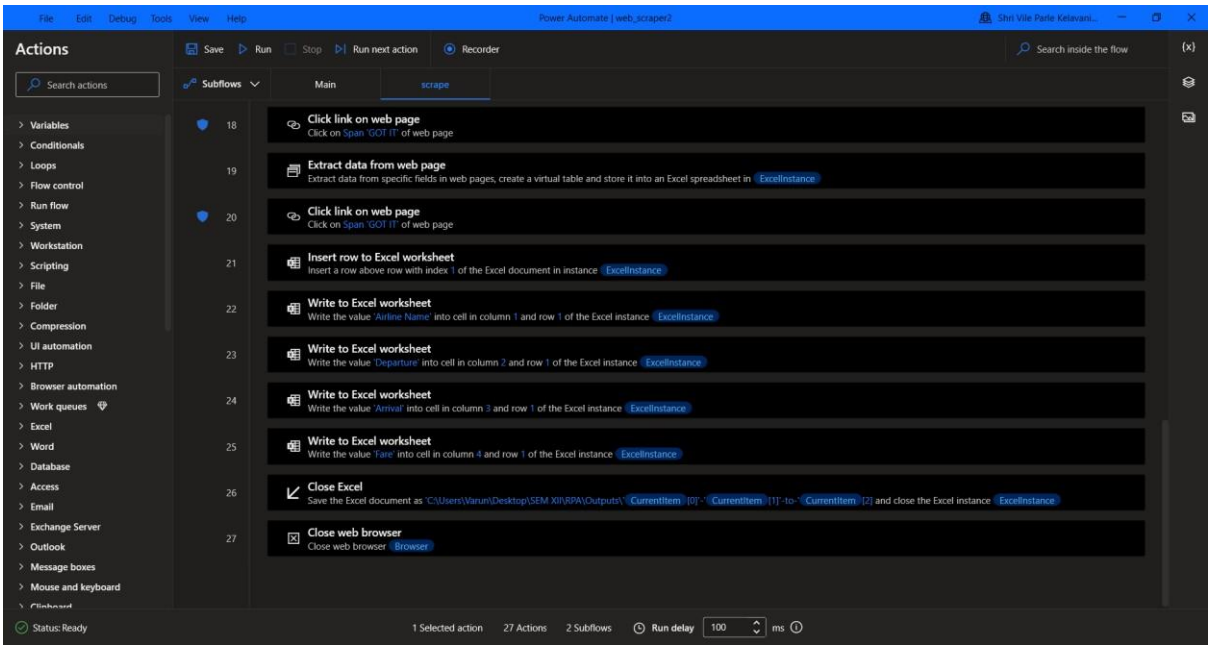
1. **Web Scraping:**
   - Utilize web scraping actions to extract the flight fare details.
2. **File Creation:**
   - Implement actions to create an excel file the has the data to be used as input and written back after scraping.

3. **Excel Data Entry:**
    - Utilize Excel actions to populate a spreadsheet with the scraped data.
4. **Finding the Max/Min Fare:**
    - Calculate the Maximum and Minimum fares and save it in a excel file.

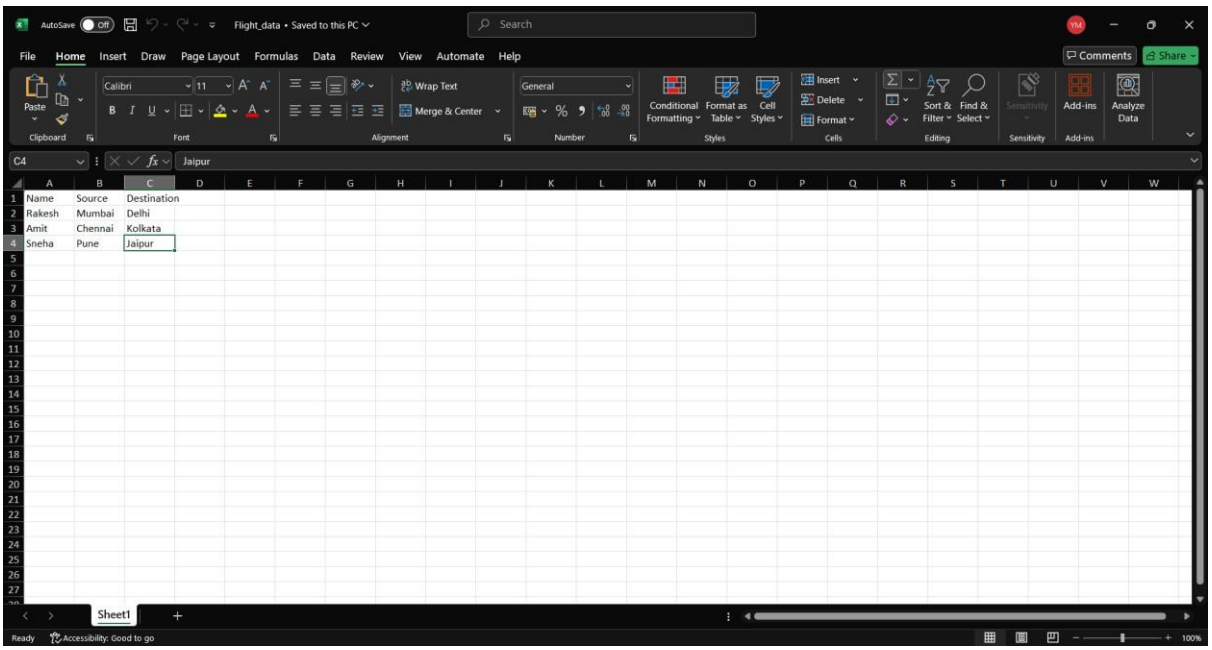
Flow Screenshots:



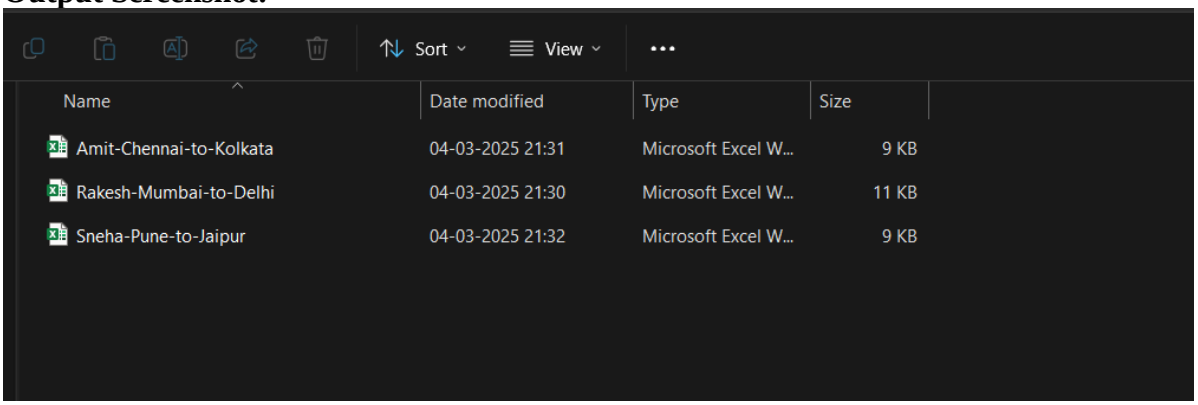


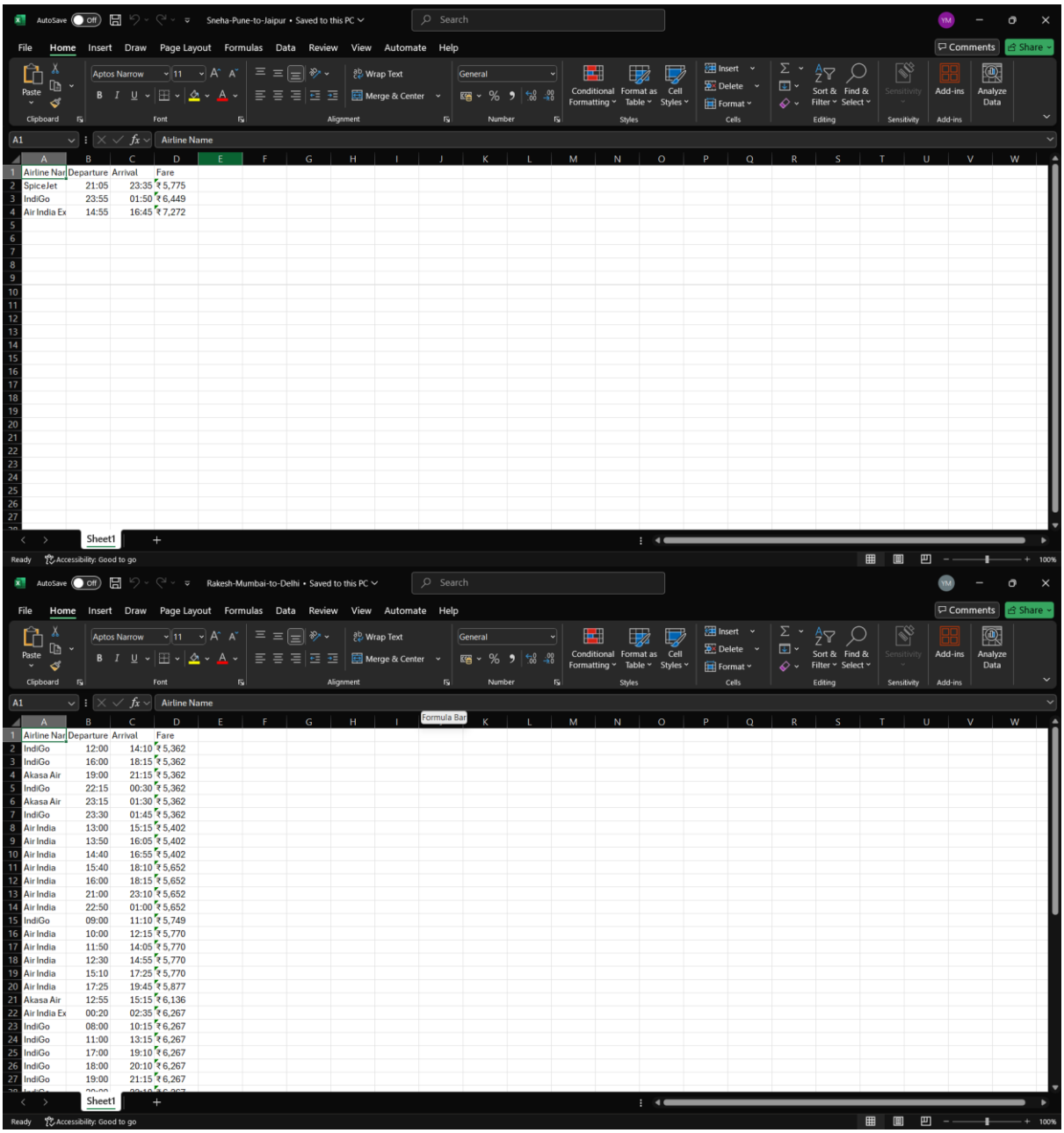


Input Screenshots:



Output Screenshot:





Airline Name	Departure	Arrival	Fare
SpiceJet	21:05	23:35	₹5,775
IndiGo	23:55	01:50	₹6,449
Air India Ex	14:55	16:45	₹7,272

Airline Name	Departure	Arrival	Fare
IndiGo	12:00	14:10	₹5,362
IndiGo	16:00	18:15	₹5,362
Akasa Air	19:00	21:15	₹5,362
IndiGo	22:15	00:30	₹5,362
Akasa Air	23:15	01:30	₹5,362
IndiGo	23:30	01:45	₹5,362
Air India	13:00	15:15	₹5,402
Air India	13:50	16:05	₹5,402
Air India	14:40	16:55	₹5,402
Air India	15:40	18:10	₹5,652
Air India	16:00	18:15	₹5,652
Air India	21:00	23:10	₹5,652
Air India	22:50	01:00	₹5,652
IndiGo	09:00	11:10	₹5,749
Air India	10:00	12:15	₹5,770
Air India	11:50	14:05	₹5,770
Air India	12:30	14:55	₹5,770
Air India	15:10	17:25	₹5,770
Air India	17:25	19:45	₹5,877
Akasa Air	12:55	15:15	₹6,136
Air India Ex	00:20	02:35	₹6,267
IndiGo	08:00	10:15	₹6,267
IndiGo	11:00	13:15	₹6,267
IndiGo	17:00	19:10	₹6,267
IndiGo	18:00	20:10	₹6,267
IndiGo	19:00	21:15	₹6,267

Conclusion:

Learnt how to take data from an excel and how to use the data to perform tasks on a web page and then finally extract data from a web page. This flow was used to get information for flights to and from a destination for multiple users.

Date of Experiment and Submission: 07-03-25

SVKM'S NMIMS
Mukesh Patel School of Technology Management & Engineering
Department of Mechatronics Engineering

RPA Lab
Subject- Robotic Process Automation

EXPERIMENT NO. 6

Objective:

The objective of this experiment is to understand the capabilities of Microsoft Power Apps. This experiment involves creating a Calculator application.

Prerequisites:

1. Computers or laptops with access to Internet.
2. Power Apps account

Theory:

1. Overview of PowerApps

PowerApps is a low-code development platform by Microsoft that allows users to create custom applications without requiring extensive programming knowledge. It provides a drag-and-drop interface for designing apps and supports integration with various Microsoft and third-party services.

2. Introduction to Basic Arithmetic Operations

Arithmetic operations form the foundation of mathematical calculations in computing. The four fundamental operations are:

- **Addition (+)**: Combining two numbers to get their sum.
- **Subtraction (-)**: Finding the difference between two numbers.
- **Multiplication (×)**: Calculating the product of two numbers.
- **Division (÷)**: Splitting a number into equal parts.

In this experiment, we use PowerApps to perform these operations dynamically based on user inputs.

3. Using PowerApps for Calculator Development

PowerApps provides built-in formulas and functions to perform calculations on user inputs. We utilize:

- **Text Input Controls** to enter numbers.
- **Button Controls** to trigger operations.
- **Labels** to display results.
- **Formulas** (e.g., Value(Input1.Text) + Value(Input2.Text)) to process calculations.

Task List:

1. Planning & Setup

- Define requirements (addition, subtraction, multiplication, division)
- Open PowerApps and create a new canvas app

2. Designing the UI

- Add a **Text Input** control for user input (two number fields)
- Add **Buttons** for operations (+, -, ×, ÷, Clear)
- Add a **Label** to display the result

3. Implementing Functionality

Assign formulas to each operation button:

- Addition (Text(Result_Label) = Value(Input1.Text) + Value(Input2.Text))
- Subtraction (Text(Result_Label) = Value(Input1.Text) - Value(Input2.Text))
- Multiplication (Text(Result_Label) = Value(Input1.Text) * Value(Input2.Text))
- Division (Text(Result_Label) = If(Value(Input2.Text) <> 0, Value(Input1.Text) / Value(Input2.Text), "Error"))
- Assign a formula to the **Clear** button (Reset(Input1); Reset(Input2); Reset(Result_Label))

4. Enhancements & Validations

- Validate inputs (ensure only numbers are entered)
- Handle division by zero (show an error message)
- Format the output for better readability

5. Testing & Debugging

- Test each arithmetic operation
- Test edge cases (e.g., empty input, zero division)
- Fix any issues found

6. Publishing & Deployment

- Save and publish the app
- Share with users if needed

Procedure Overview:

1. Creating a New PowerApps Canvas App

1. Open PowerApps and create a new Canvas app.
2. Choose Blank App and select Tablet or Phone Layout based on preference.

2. Designing the User Interface (UI)

1. Add two Text Input controls for user number inputs.
2. Add four Button controls for operations (+, -, ×, ÷).
3. Add a Label to display the result.
4. Add a Clear button to reset inputs and results.
5. Customize the layout, button colors, and text sizes for better usability.

3. Implementing Functionality with Formulas

1. Assign PowerApps formulas to each operation button:
    - Addition: Set(Result, Value(Input1.Text) + Value(Input2.Text))
    - Subtraction: Set(Result, Value(Input1.Text) - Value(Input2.Text))
    - Multiplication: Set(Result, Value(Input1.Text) * Value(Input2.Text))
    - Division: Set(Result, If(Value(Input2.Text) <> 0, Value(Input1.Text) / Value(Input2.Text), "Error"))
2. Set the Result Label to display Result.
3. Configure the Clear Button with Reset(Input1); Reset(Input2); Set(Result, "").

4. Adding Input Validation & Enhancements

1. Ensure only numbers are entered in input fields.
2. Prevent division by zero by displaying an error message.
3. Format the result label for better readability (e.g., decimal precision).

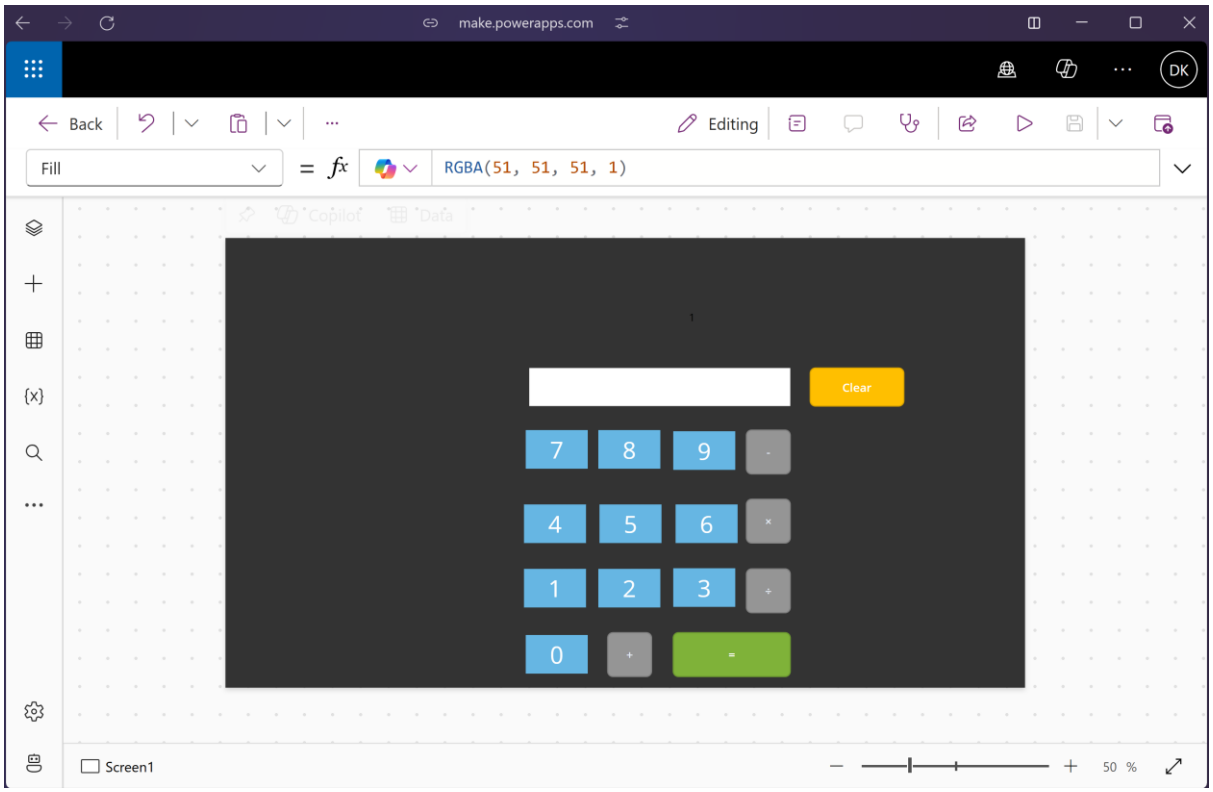
5. Testing & Debugging

1. Test each arithmetic operation.
2. Check how the app handles blank inputs and zero division.
3. Make UI/UX improvements based on test results.

6. Publishing & Deployment

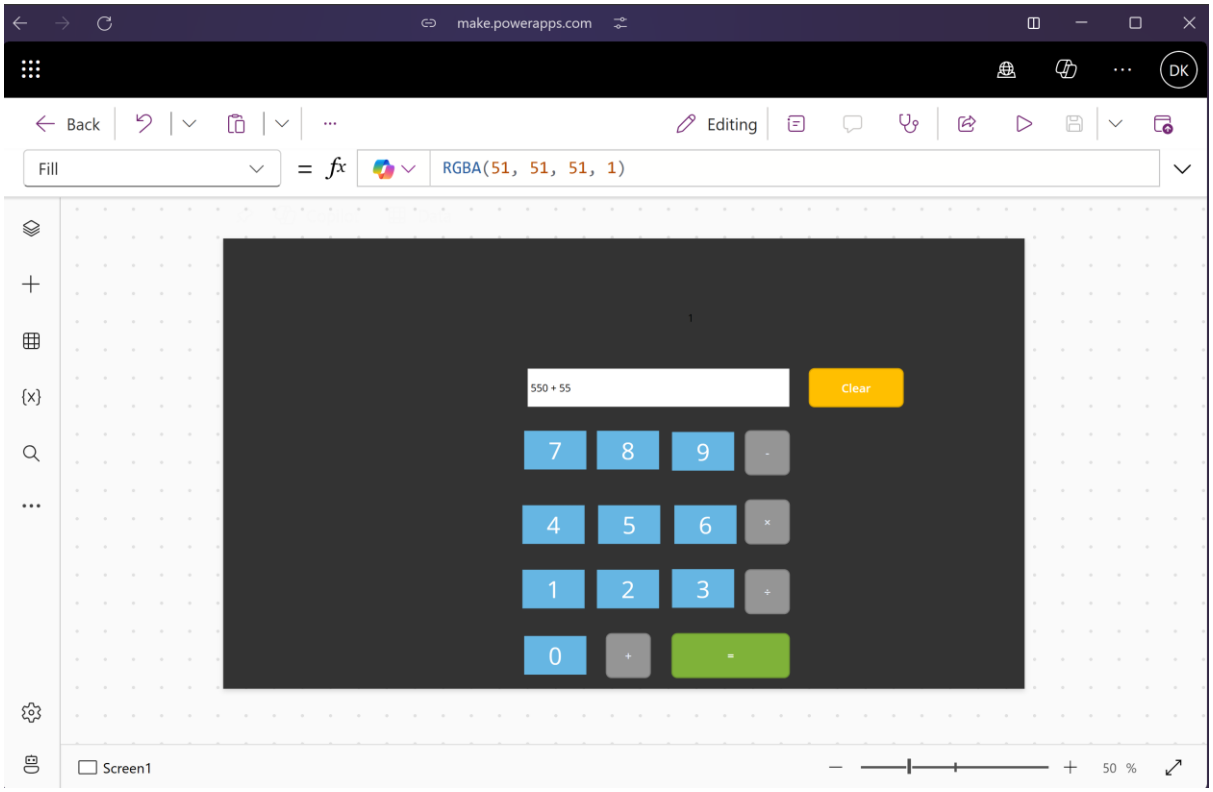
1. Save the app in PowerApps.
2. Publish and share with users if needed.
3. Provide documentation or a short guide for users.

Flow Screenshots:

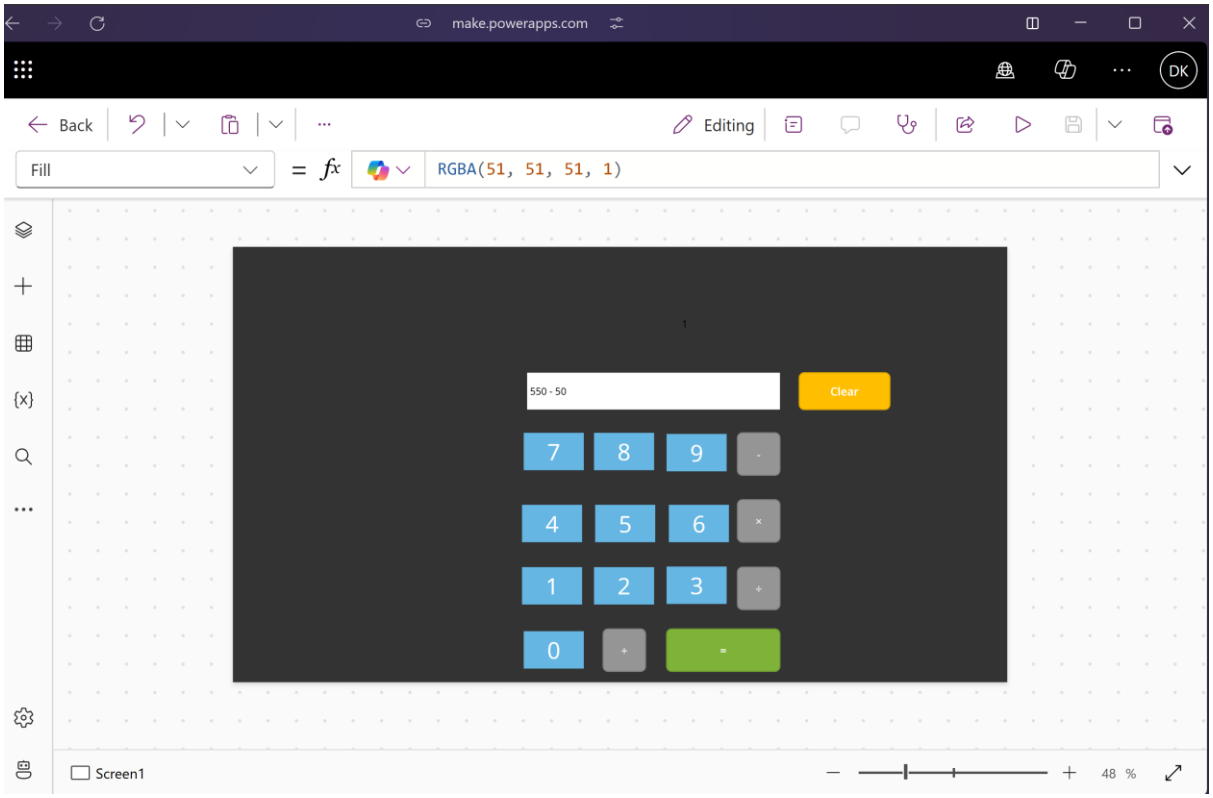


Input Screenshots:

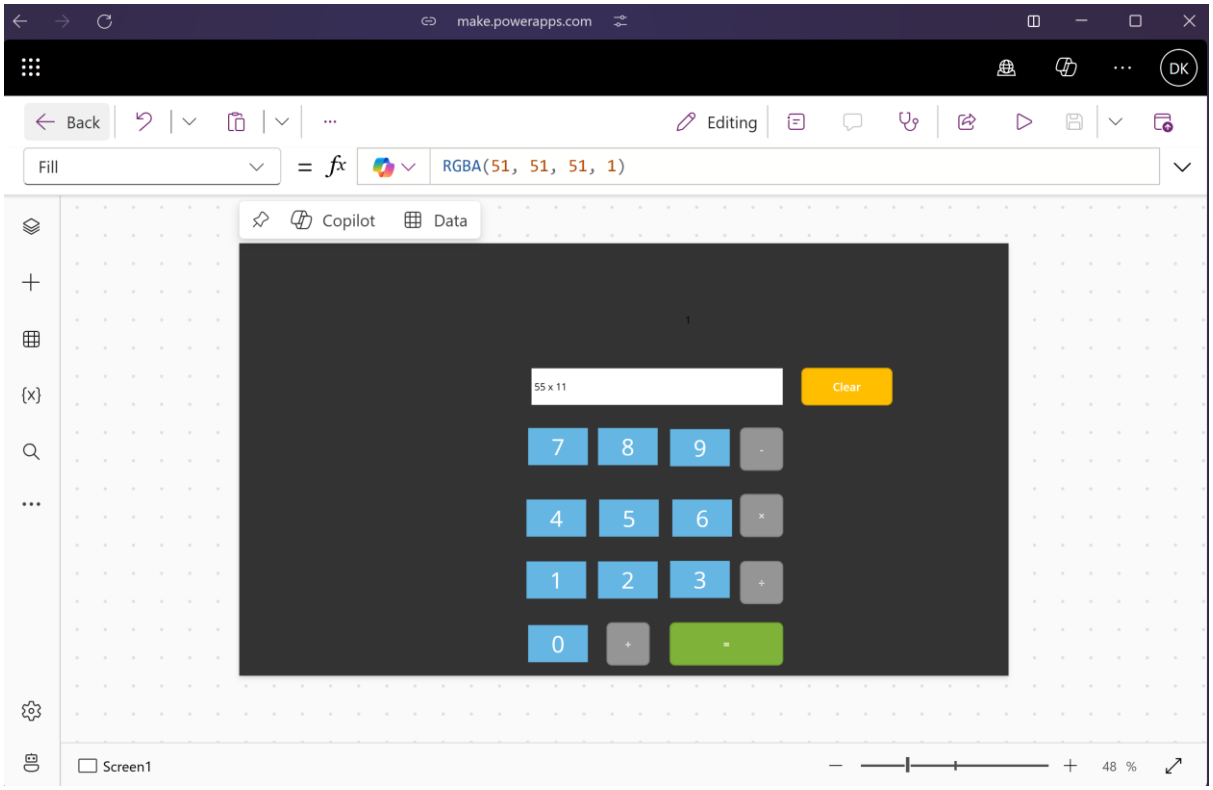
Addition:



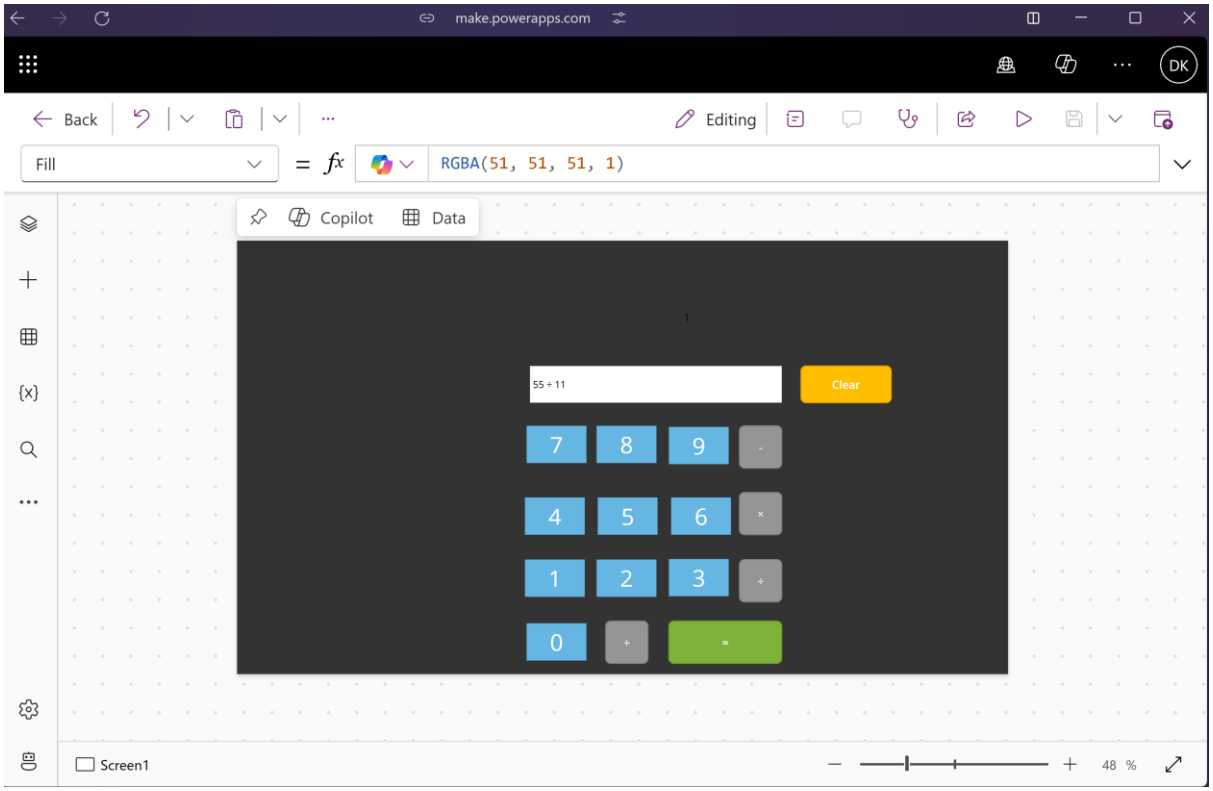
Subtraction:



Multiplication:

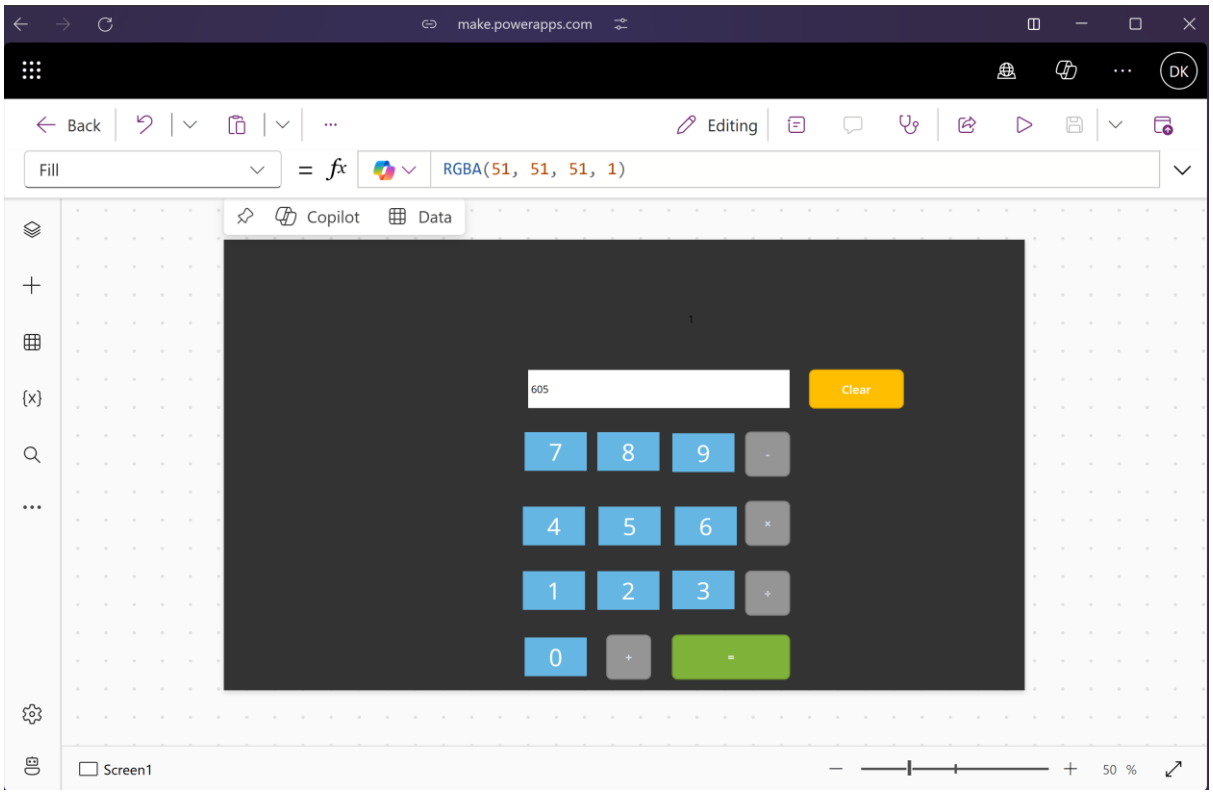

make.powerapps.com
Back
Editing
Fill
RGBA(51, 51, 51, 1)
Copilot
Data
55 x 11
Clear
7
8
9
4
5
6
1
2
3
0
Screen1
48 %

Division:

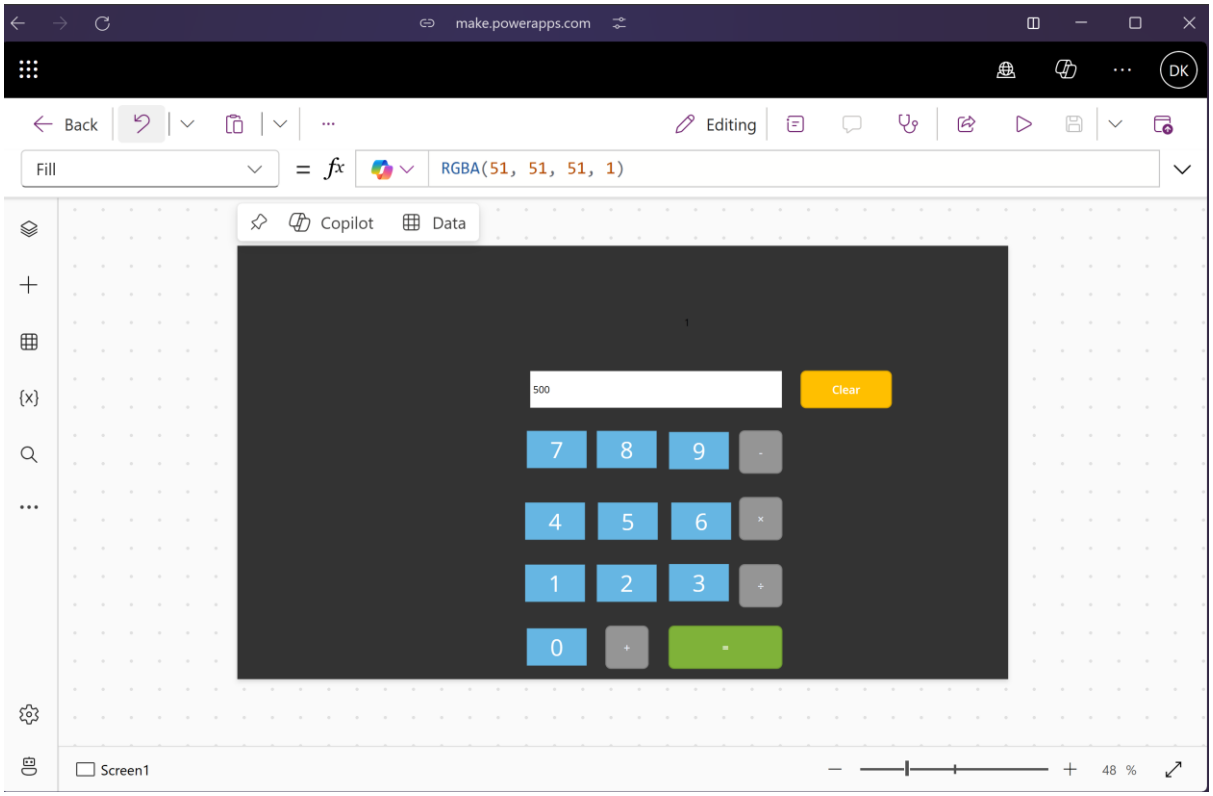

make.powerapps.com
Back
Editing
Fill
RGBA(51, 51, 51, 1)
Copilot
Data
55 ÷ 11
Clear
7
8
9
4
5
6
1
2
3
0
Screen1
48 %

Output Screenshot:

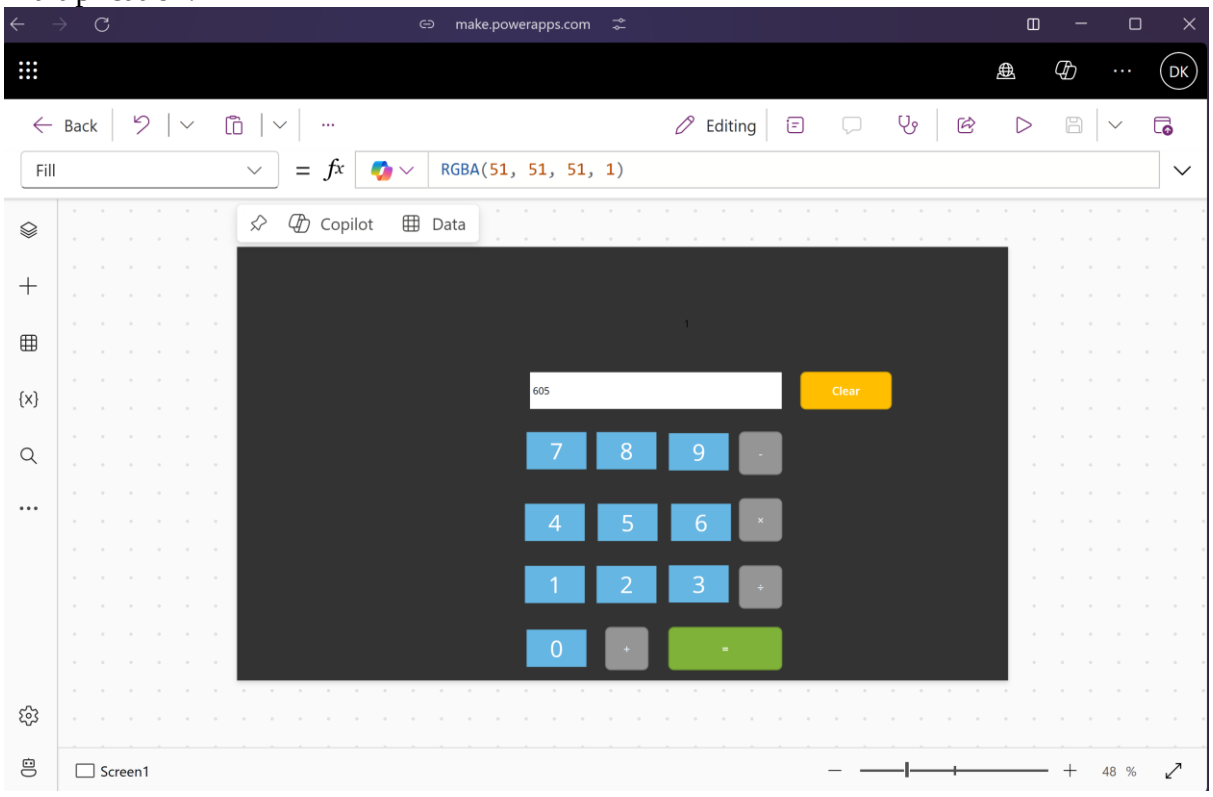
Addition:



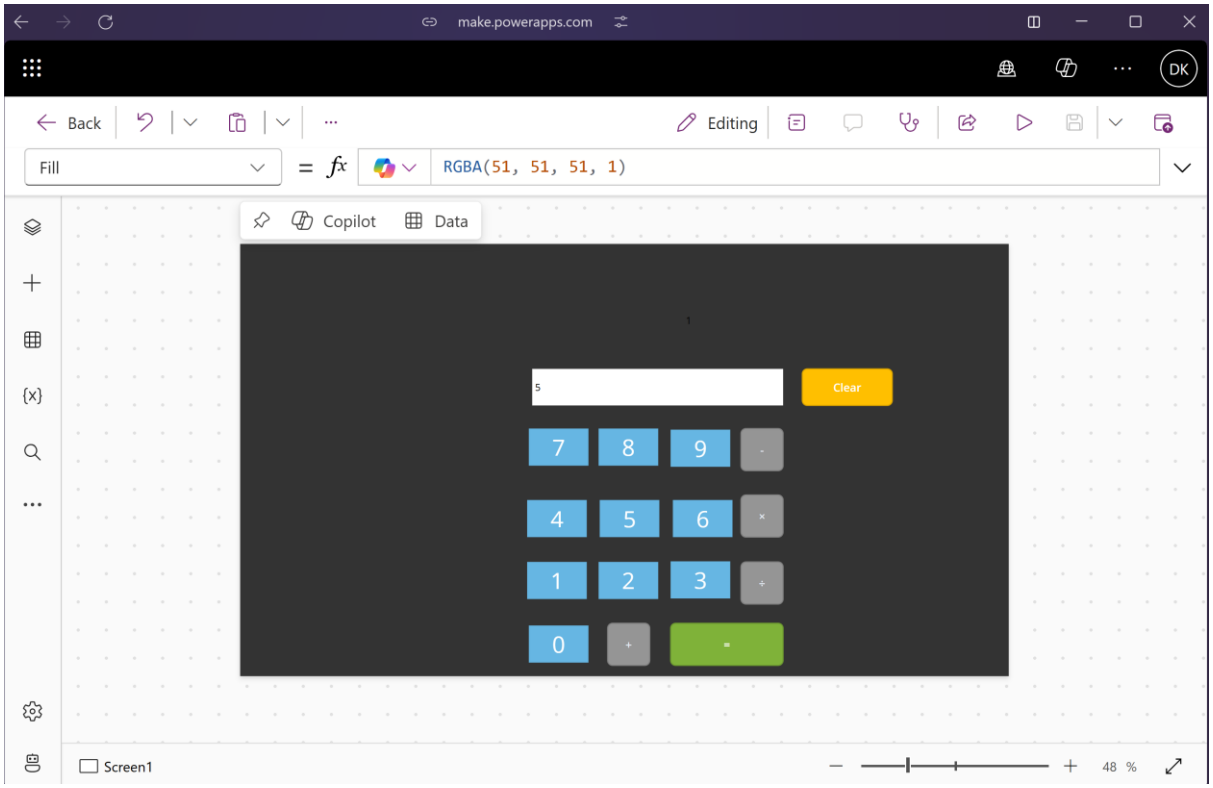
Subtraction:



Multiplication:



Division:



Conclusion:

Created and tested the working of a calculator app using Microsoft PowerApps successfully.

Date of Experiment: 12-03-25
Date of Submission: 12-03-25

SVKM'S NMIMS
Mukesh Patel School of Technology Management & Engineering
Department of Mechatronics Engineering
RPA- Lab
Subject- Robotic Process Automation
EXPERIMENT NO. 7

Objective:
Building a Professional Login Application in Power Apps with SharePoint List Authentication

Prerequisites:
3. Computers or laptops with access to Internet.
4. Power Apps account
5. SharePoint List With Given Fields

Theory:
1. Overview of Power Apps
Power Apps is a Microsoft platform that enables users to create custom applications without extensive coding. It's a powerful tool for app development with a user-friendly interface.
2. Introduction to SharePoint as a Data Source
SharePoint is a collaborative platform that integrates with Microsoft Office. In this experiment, we'll utilize SharePoint as a data source for our Power App.

Procedure Overview
1. Setting Up SharePoint List
1.1 Creating a SharePoint List
1. Open SharePoint and create a new list with necessary fields.
2. Populate the columns of the list with following data:
   a) User_Name b) Password

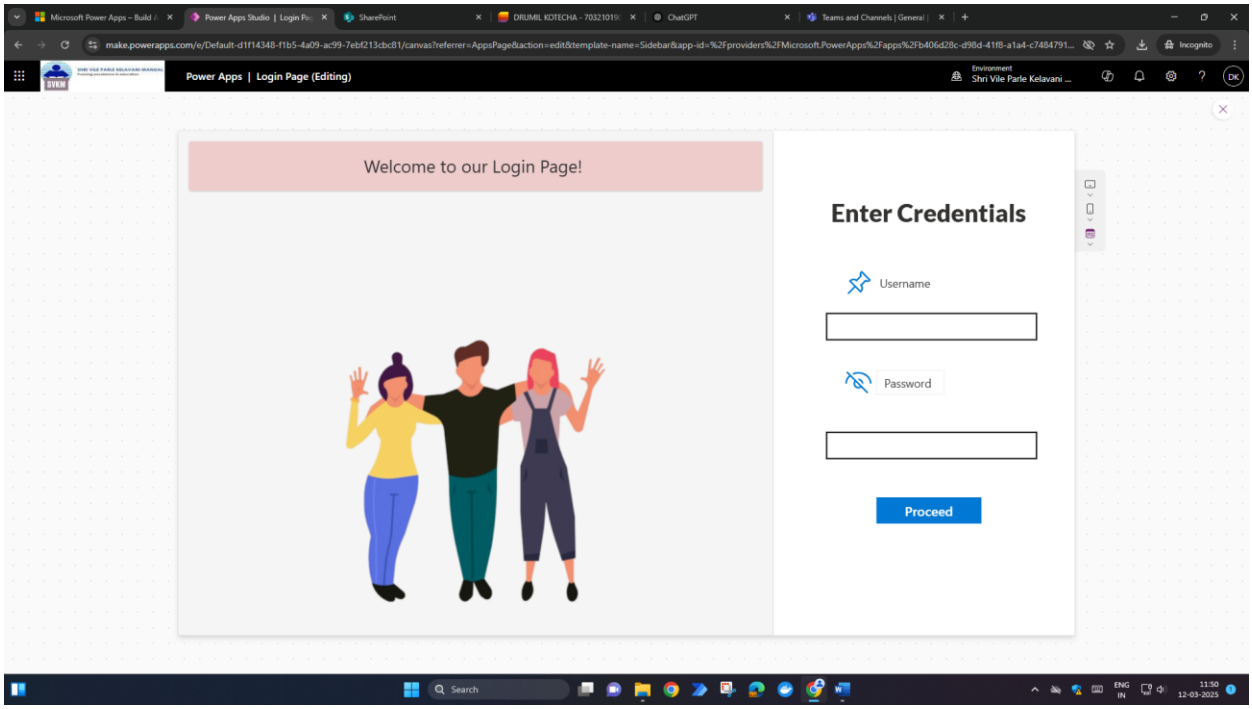
2. Creating a Form in Power Apps
2.1 Connecting Power Apps to SharePoint
1. Open Power Apps and create a new canvas app.
2. Connect the app to the SharePoint list created in Step 2.
3. Explore available data connections.

2.2 Designing a Customized Screen
1. Design a login screen, a welcome screen and an Error Screen layout suitable for data input.
2. Implement the buttons necessary for the application on each screen and apply the necessary logic.

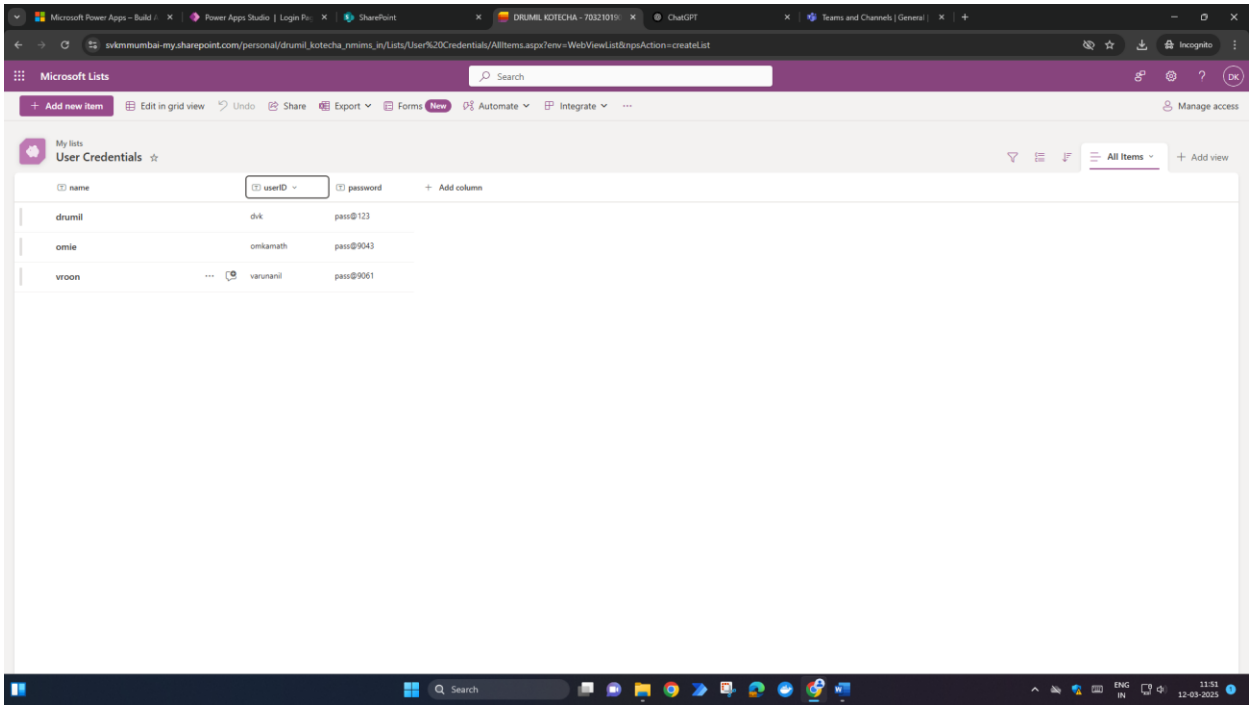
2.2 Implementing the Logic
1. Your login screen should check the login credentials defined at the SharePoint List.

2. If the user is authenticated, it should go to the welcome screen, else it should go to the error screen.

Power Apps (Screenshot):

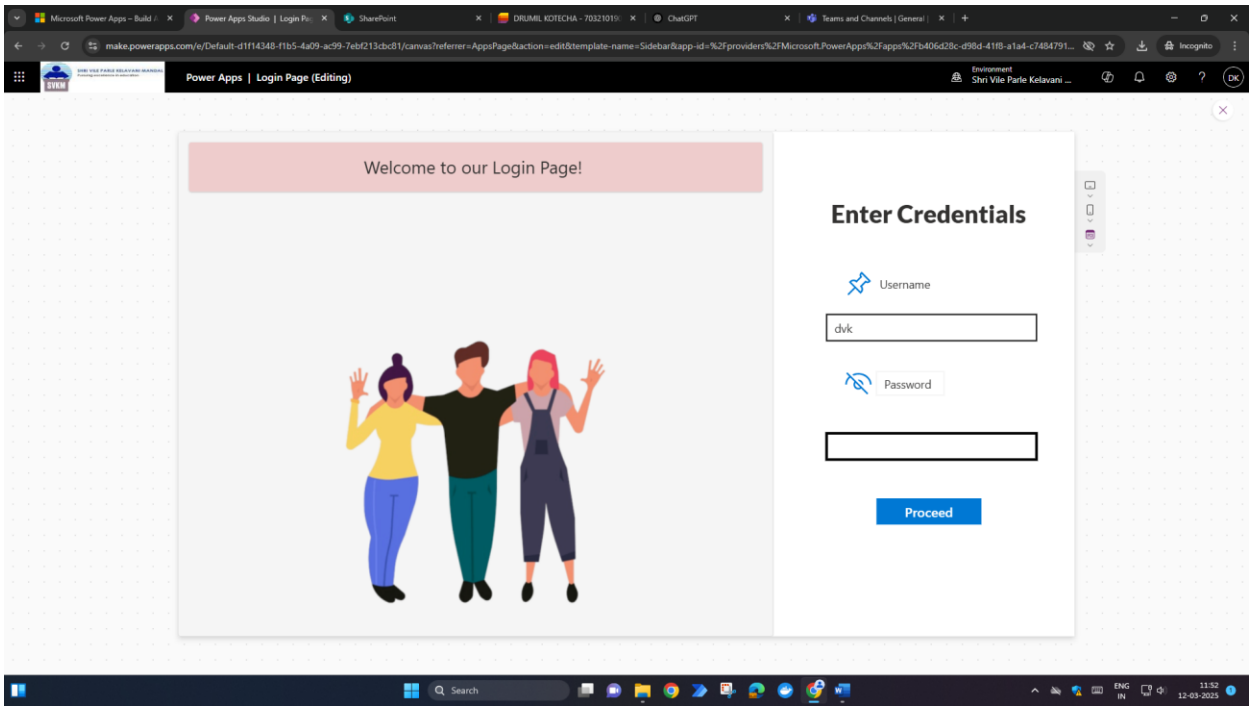


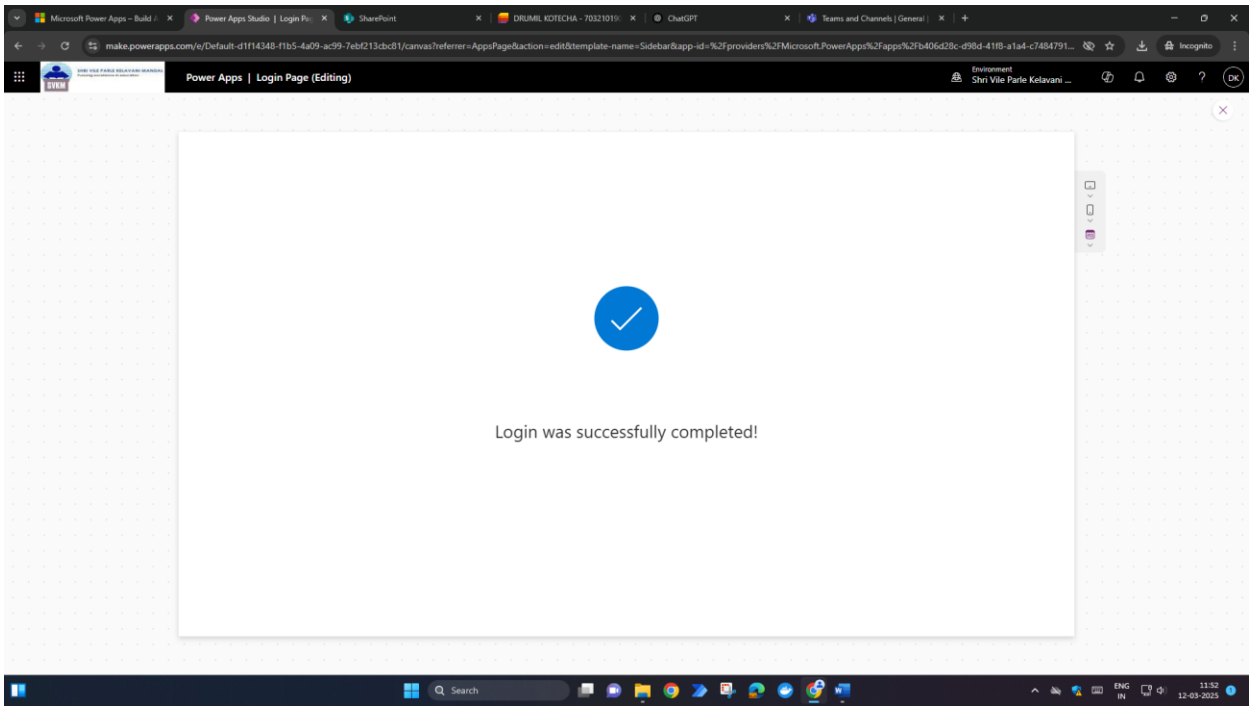
SharePoint Data List (Screenshot):



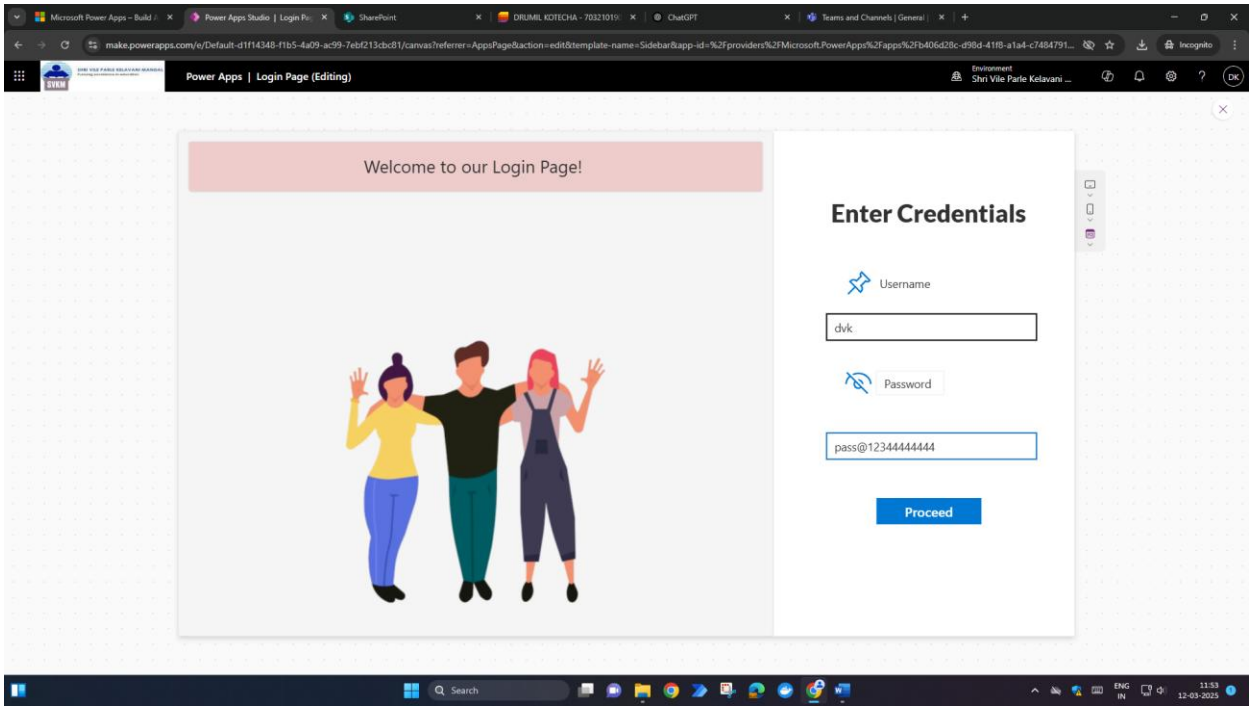
User Authentication (Screenshot):

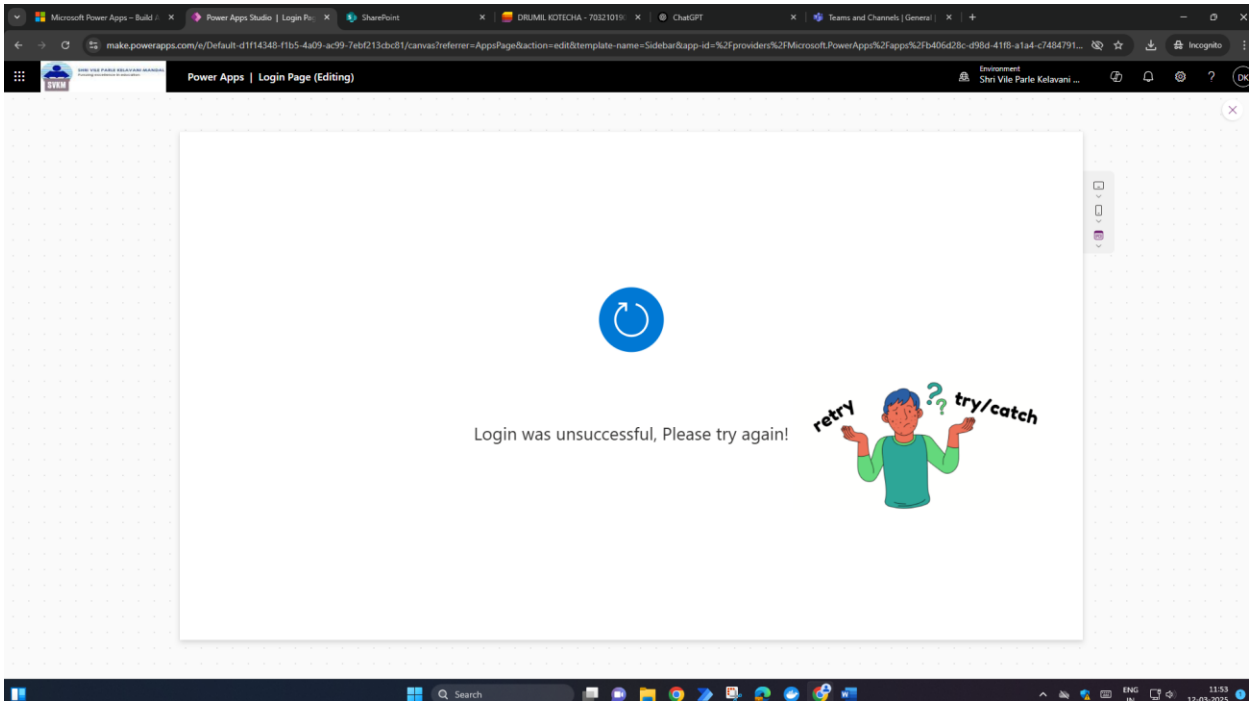
Successful →





Unsuccessful →





User Authentication:

```
If(
    IsBlank(
        LookUp('User Credentials', userID = username.Text && password = 
password_input.Text)
    ),
    Navigate(Screen3), // Navigate to Screen3 if login fails
    Navigate(Screen2)  // Navigate to Screen2 if login is successful
)
```

Conclusion:

Successfully implemented authentication functionality into power apps using functions such as Lookup.

Reference Material:

1. https://www.youtube.com/watch?v=BN74SsN4mm8

2. https://www.youtube.com/watch?v=QaFkl5168S4

Date of Experiment:
Date of Submission:

SVKM'S NMIMS
Mukesh Patel School of Technology Management & Engineering
Department of Mechatronics Engineering
RPA - Lab
Subject- Robotic Process Automation
EXPERIMENT NO. 8

Objective:
The objective of this experiment is to understand the capabilities of Microsoft Power Apps. This experiment involves creating a CRUD based application from an excel table and making a form from a hand drawn form.

Prerequisites:
6. Computers or laptops with access to Internet.
7. Power Apps account
8. An Excel sheet with sample data.
9. Hand-drawn forms.

Theory:
1. Overview of Power Apps
Power Apps is a Microsoft platform that enables users to create custom applications without extensive coding. It's a powerful tool for app development with a user-friendly interface.
2. Introduction to CRUD Operations
CRUD stands for Create, Read, Update, and Delete – fundamental operations in database-driven applications. In this experiment, you'll learn to implement these operations using Power Apps.
Task List: Intro To Power Apps
1. **Creating a CRUD-Based App from Excel**
   - Creating a CRUD based application on Power Apps is very easy and intuitive.
2. **Creating an App from a Hand-Drawn Form**
   - Generating an form from the layout drawn by hand is a very effective way of designing and developing a form this helps in reducing the timeline of development

Procedure Overview:
1. **Creating a CRUD-Based App from Excel**
   1. Open Power Apps and create a new app.
   2. Import sample data from the provided Excel sheet.
   3. Understand the structure of the data and columns.
   4. Establish data connections between Power Apps and the Excel sheet.
   5. Explore data sources and understand how Power Apps interacts with external data.
   6. Create screens for viewing and editing data.
   7. Customize the layout and design of each screen.
   8. Add functionality to create new records.
   9. Enable editing and updating of existing records.
   10. Implement the ability to delete records.

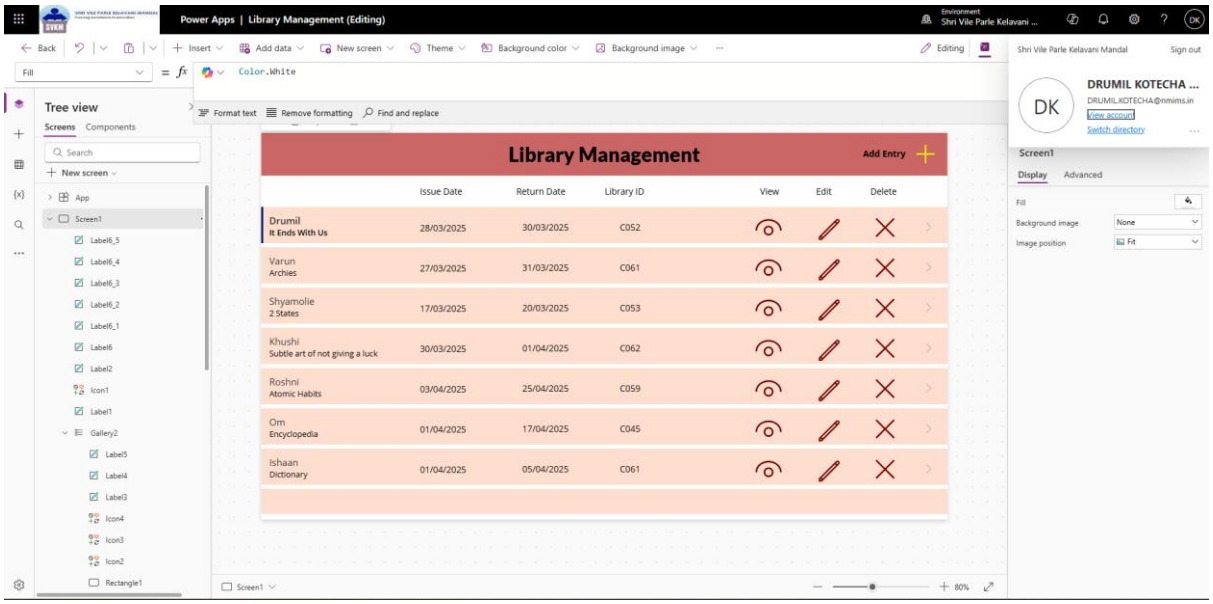
2. Creating an App from a Hand-Drawn Form

1. Receive a hand-drawn form for the exercise.
2. Understand the fields and functionalities represented in the form.
3. Use smartphones to capture images of the hand-drawn form.
4. Save images for reference in Power Apps.
5. Import the captured images into Power Apps.
6. Design an app based on the hand-drawn form, replicating fields and functionalities.
7. Customize the app layout and design.
8. Add creative elements to enhance the user experience.

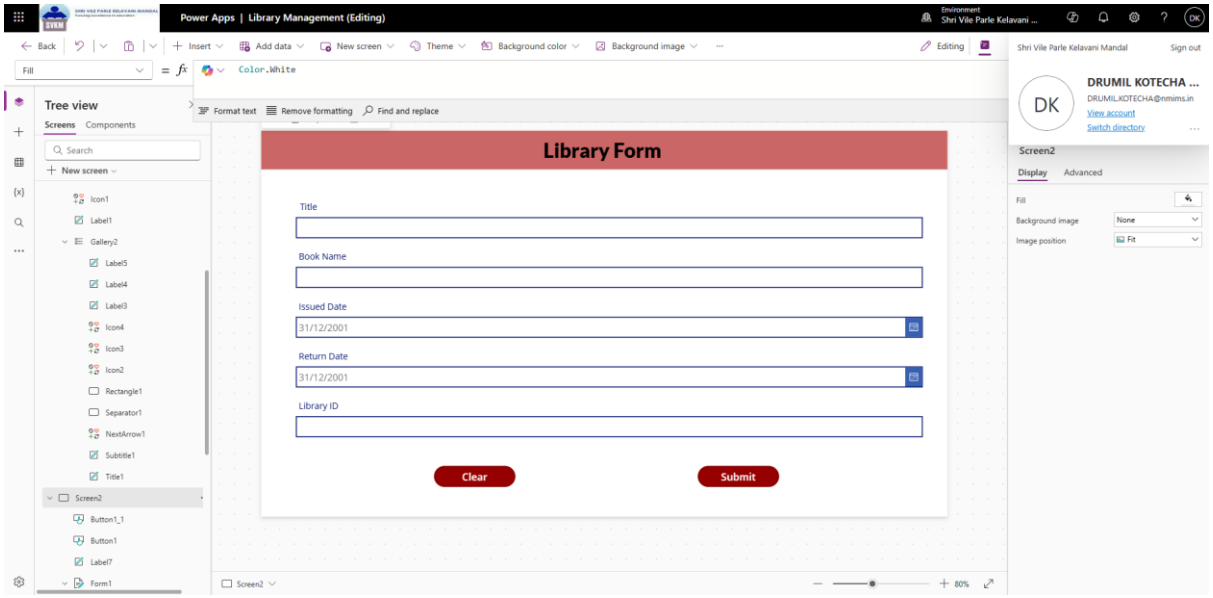
CRUD Application:

1. **Create Screen**
2. **Read (With Delete Icon Marked) Screen**
3. **Update Screen**

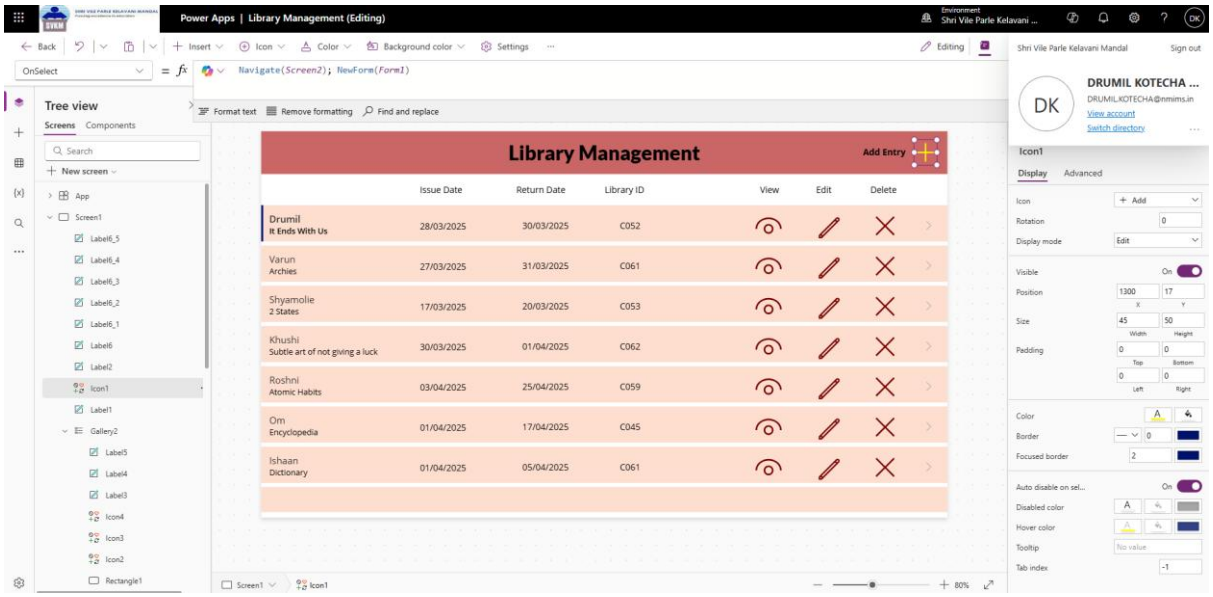
Home Screen:



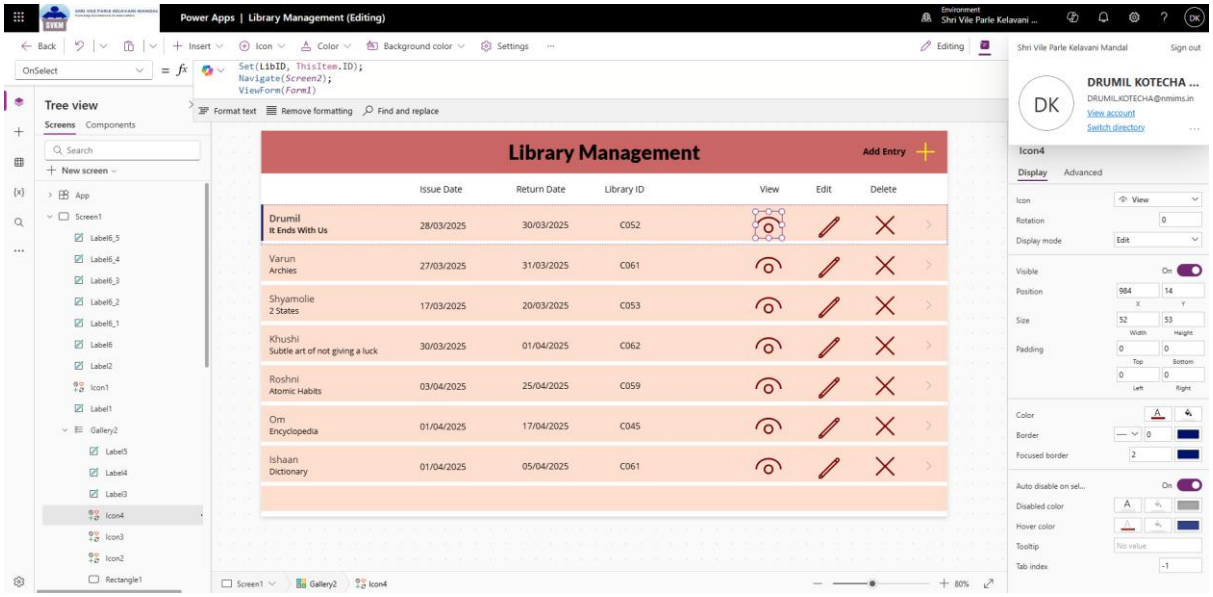
Form Screen:



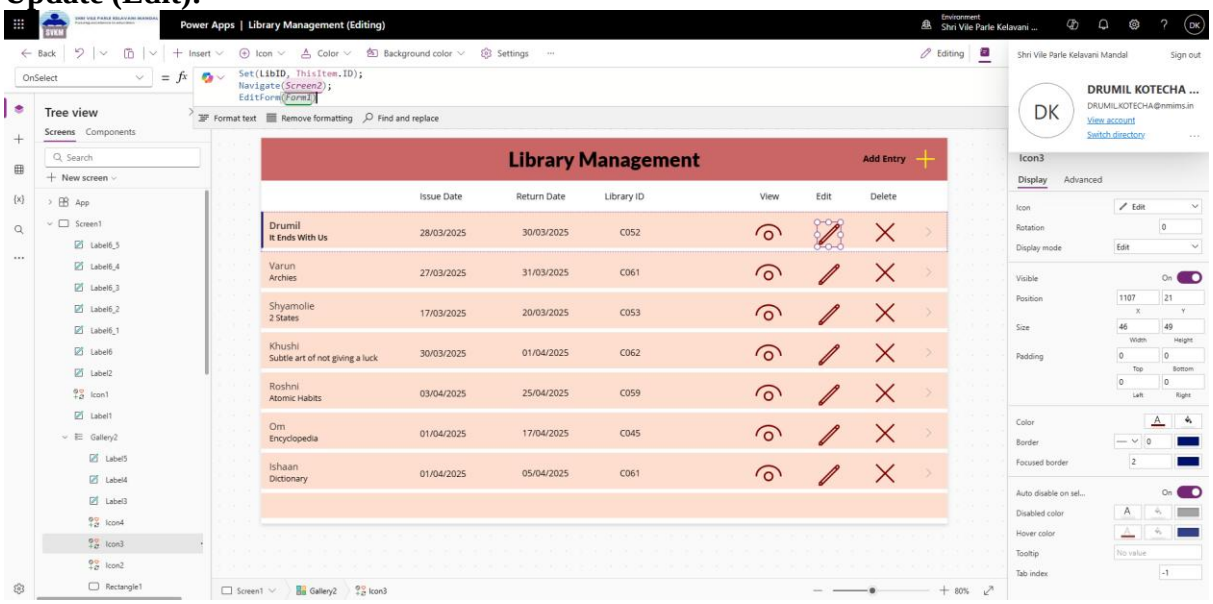
Create:



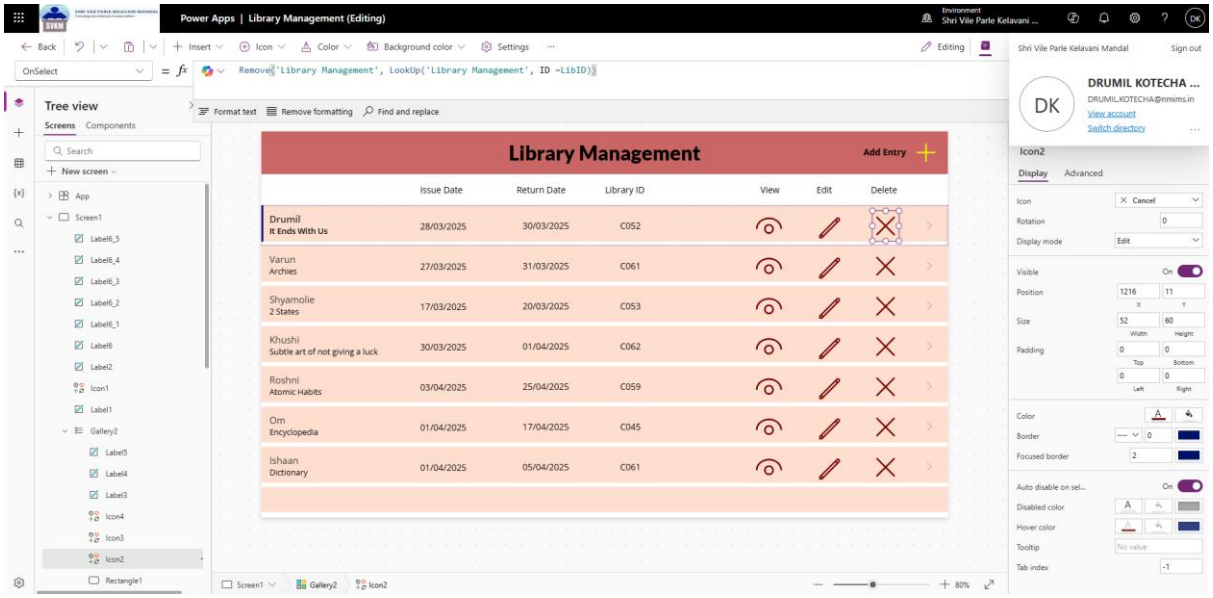
Read (View):



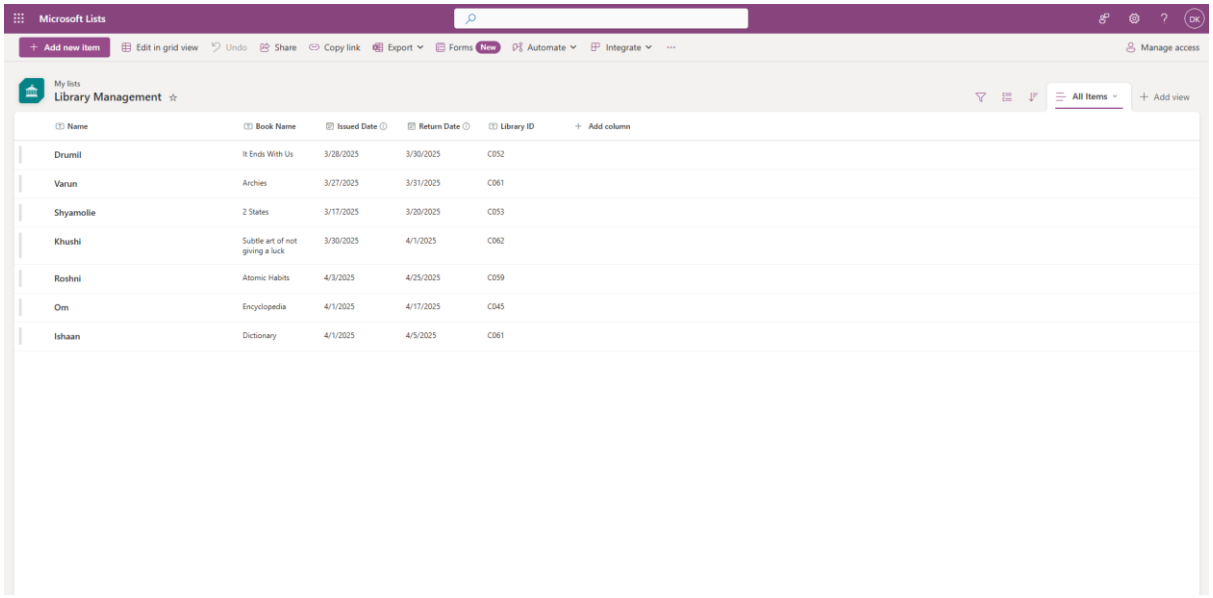
Update (Edit):



Delete:



List: https://svkmmumbai-my.sharepoint.com/personal/drumil_kotecha_nmims_in/Lists/Library%20Management/AllItems.aspx?env=WebViewList



Conclusion: CRUD stands for Create, Read, Update, and Delete – fundamental operations in database-driven applications. In this experiment, I learn to implement these operations using Power Apps.

SVKM'S NMIMS
Mukesh Patel School of Technology Management & Engineering
Robotic Process Automation
EXPERIMENT NO. 9

Date of Experiment: 04-04-2025 **Date of Submission: 04-04-2025**
Name: Drumil Kotecha **Roll no: C052**

Objective:
Create a flow that sends an email using the data submitted through a form using Power Automate.

Prerequisites:
1. Computers or laptops with access to the Internet.
2. Power Apps account

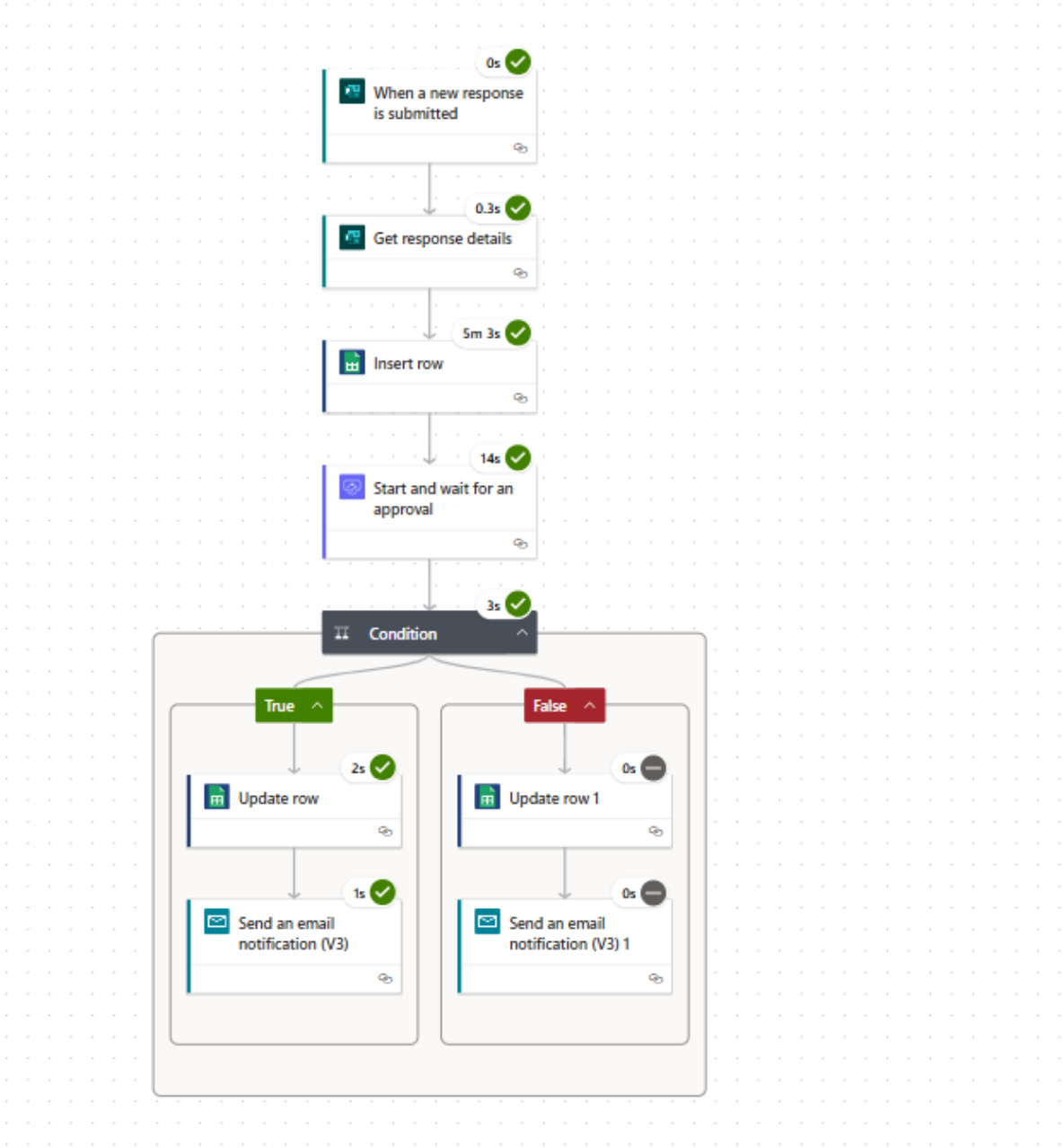
Procedure:
1. **Trigger:**
   - Choose the Microsoft forms as trigger.
   - Select the trigger action that fires when a new form response is submitted.
2. **Get Form Data:**
   - Add an action to retrieve the submitted form data.
   - Configure the action to fetch the relevant fields from the form response.
3. **Compose Email:**
   - Add an action to compose the email.
   - Use dynamic content to populate the email subject, body, and recipient fields with the form data.
4. **Send Email:**
   - Add the "Send an email" action.
   - Configure the action with the desired email subject, body, and recipient.
   - Utilize dynamic content to include form data in the email.
5. **Testing:**
   - Submit a test response through the form.
   - Verify that the flow triggers successfully and sends the email with the submitted data.

Task List:

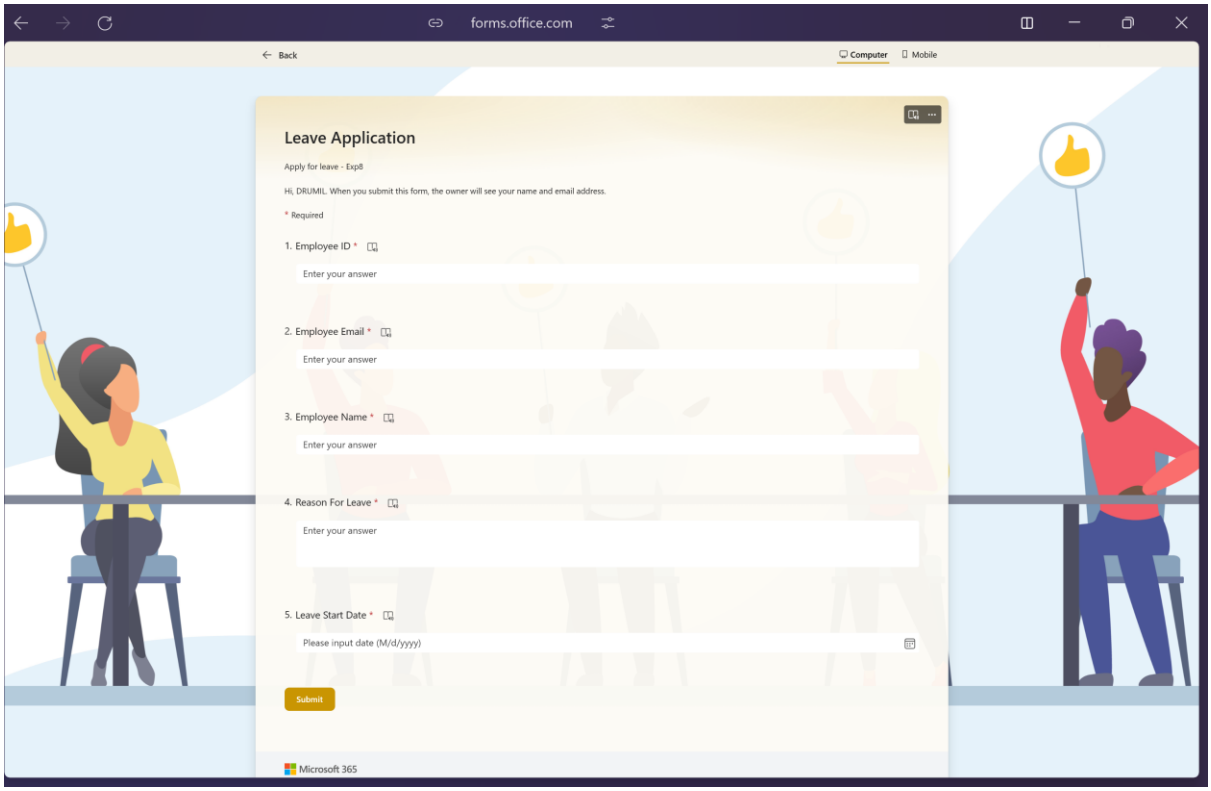
- Create a PA cloud flow which processes the leave requests and sends email about the details to the department managers for approval. Send the email to the leave applicant, and share the final response.
- Create a PA cloud flow which stores the leave responses on google sheets, shares the leave details with the manager, who can approve or reject the leave. Based on his response, update the google sheet and inform the leave applicant about the decision.

Screenshots:

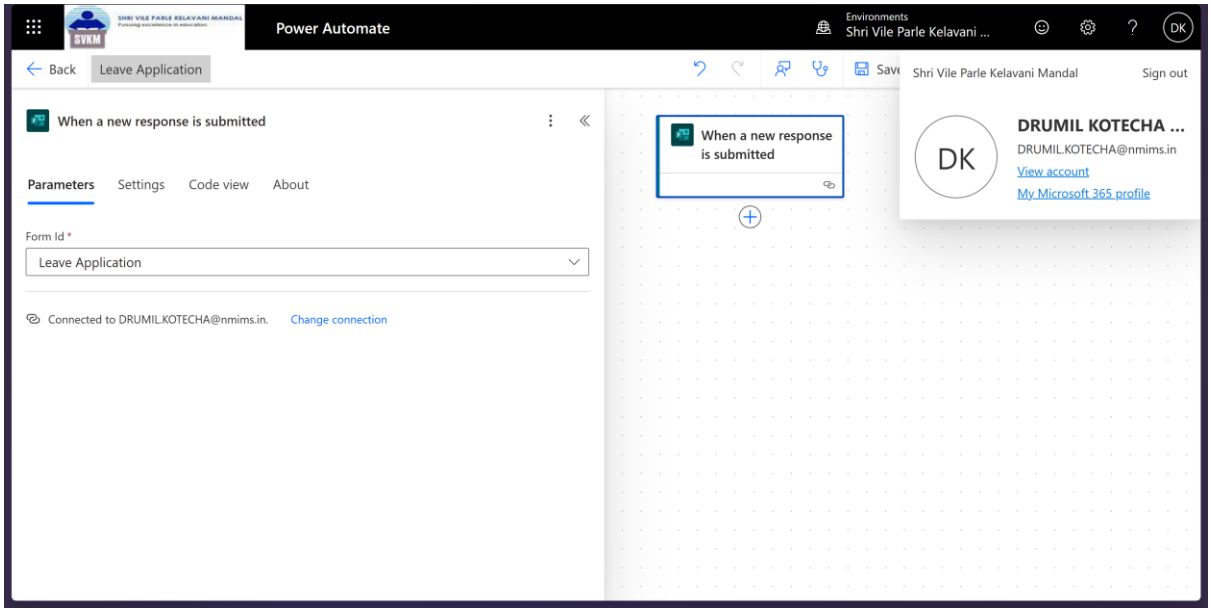
Flow Screenshot:

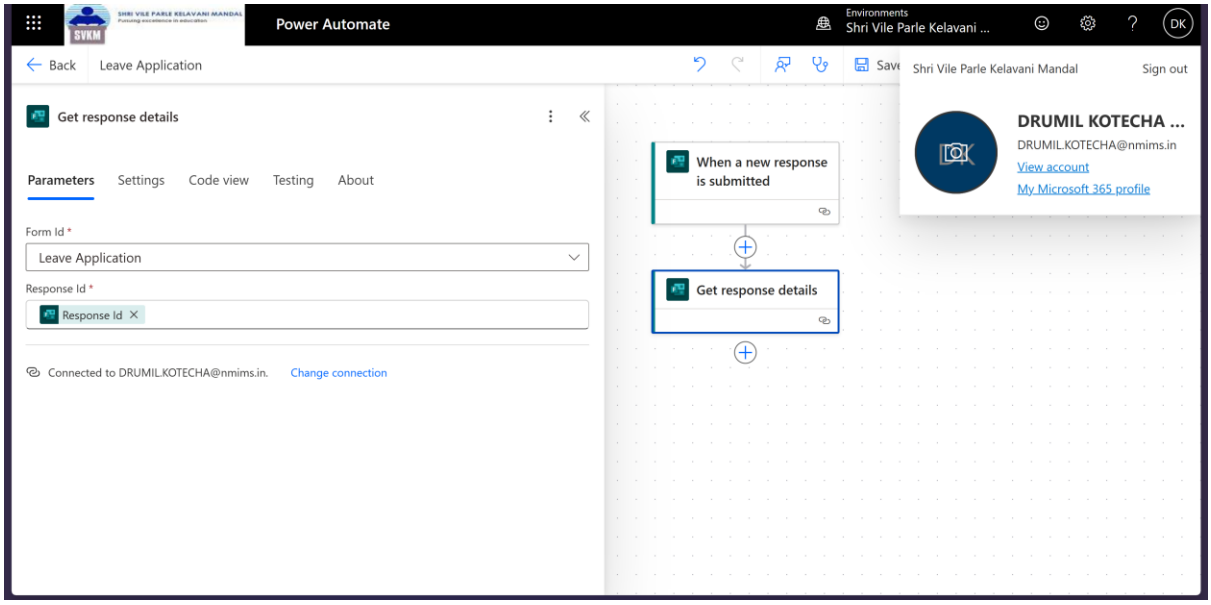


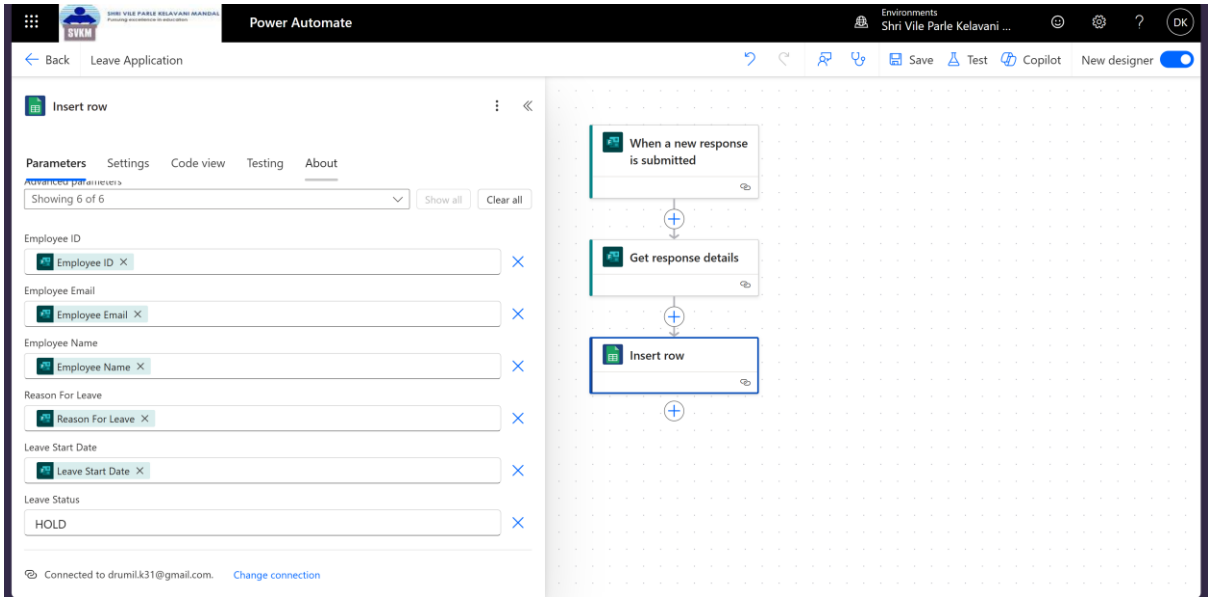
Empty form:

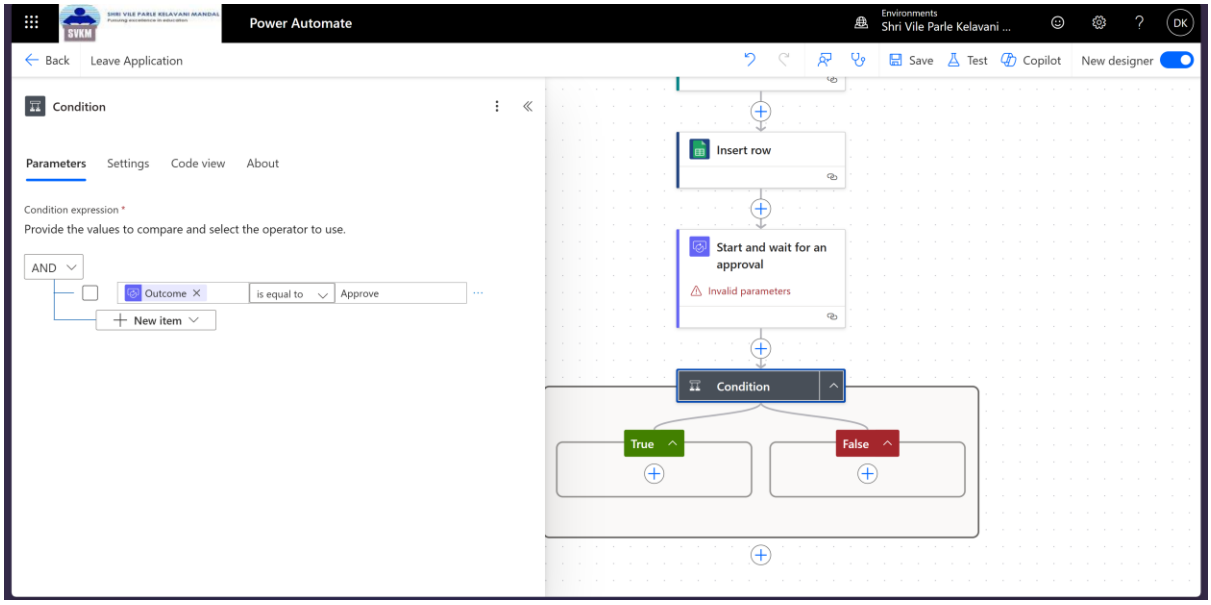

forms.office.com
Back
Computer
Mobile
Leave Application
Apply for leave - Exp8
Hi, DRUMIL. When you submit this form, the owner will see your name and email address.
* Required
1. Employee ID *
Enter your answer
2. Employee Email *
Enter your answer
3. Employee Name *
Enter your answer
4. Reason For Leave *
Enter your answer
5. Leave Start Date *
Please input date (M/d/yyyy)
Submit
Microsoft 365

Flow:

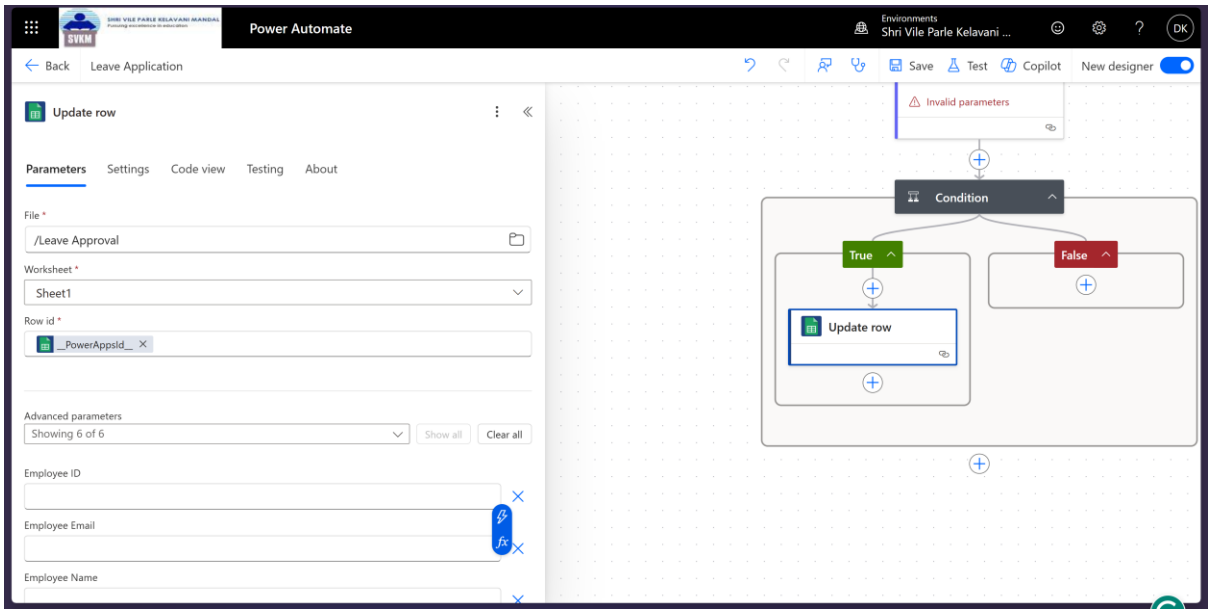

Power Automate
Environments
Shri Vile Parle Kelavani ...
Back
Leave Application
Save
When a new response is submitted
Parameters
Settings
Code view
About
Form Id *
Leave Application
Connected to DRUMIL.KOTECHA@nmims.in.
Change connection
When a new response is submitted
Shri Vile Parle Kelavani Mandal
Sign out
DK
DRUMIL KOTECHA ...
DRUMIL.KOTECHA@nmims.in
View account
My Microsoft 365 profile


Power Automate
Environments
Shri Vile Parle Kelavani ...
Back
Leave Application
Save
Get response details
Parameters
Settings
Code view
Testing
About
Form Id *
Leave Application
Response Id *
Response Id
Connected to DRUMIL.KOTECHA@nmims.in.
Change connection
When a new response is submitted
Get response details
Shri Vile Parle Kelavani Mandal
Sign out
DRUMIL KOTECHA ...
DRUMIL.KOTECHA@nmims.in
View account
My Microsoft 365 profile

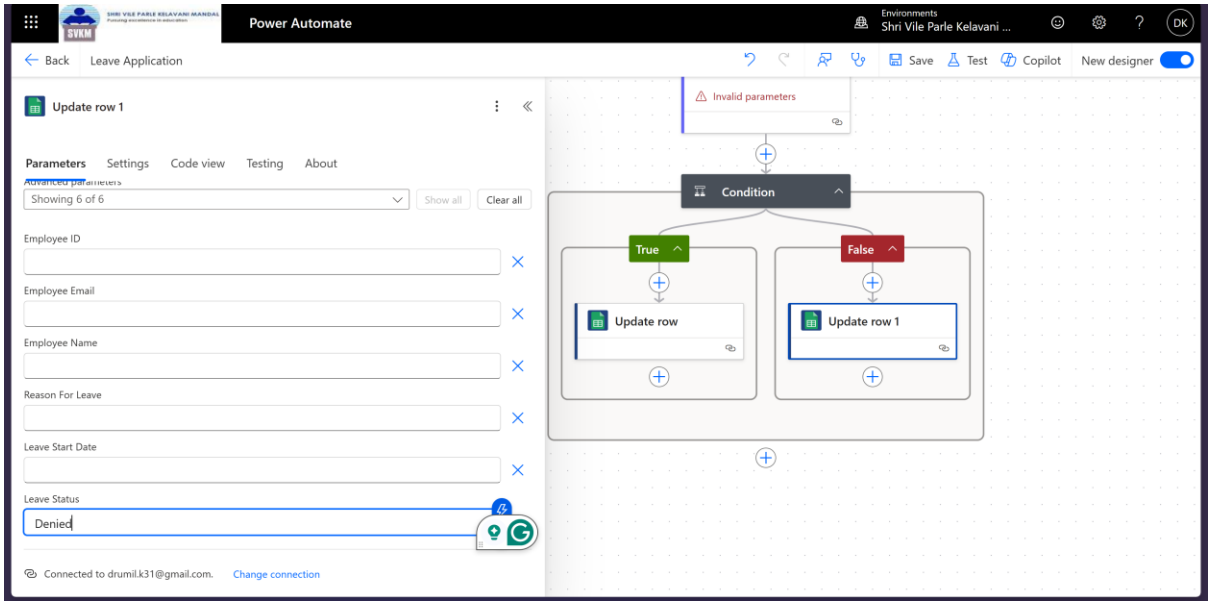

Power Automate
Environments
Shri Vile Parle Kelavani ...
Back
Leave Application
Save
Test
Copilot
New designer
Insert row
Parameters
Settings
Code view
Testing
About
Showing 6 of 6
Show all
Clear all
Employee ID
Employee ID
Employee Email
Employee Email
Employee Name
Employee Name
Reason For Leave
Reason For Leave
Leave Start Date
Leave Start Date
Leave Status
HOLD
Connected to drumil.k31@gmail.com.
Change connection
When a new response is submitted
Get response details
Insert row



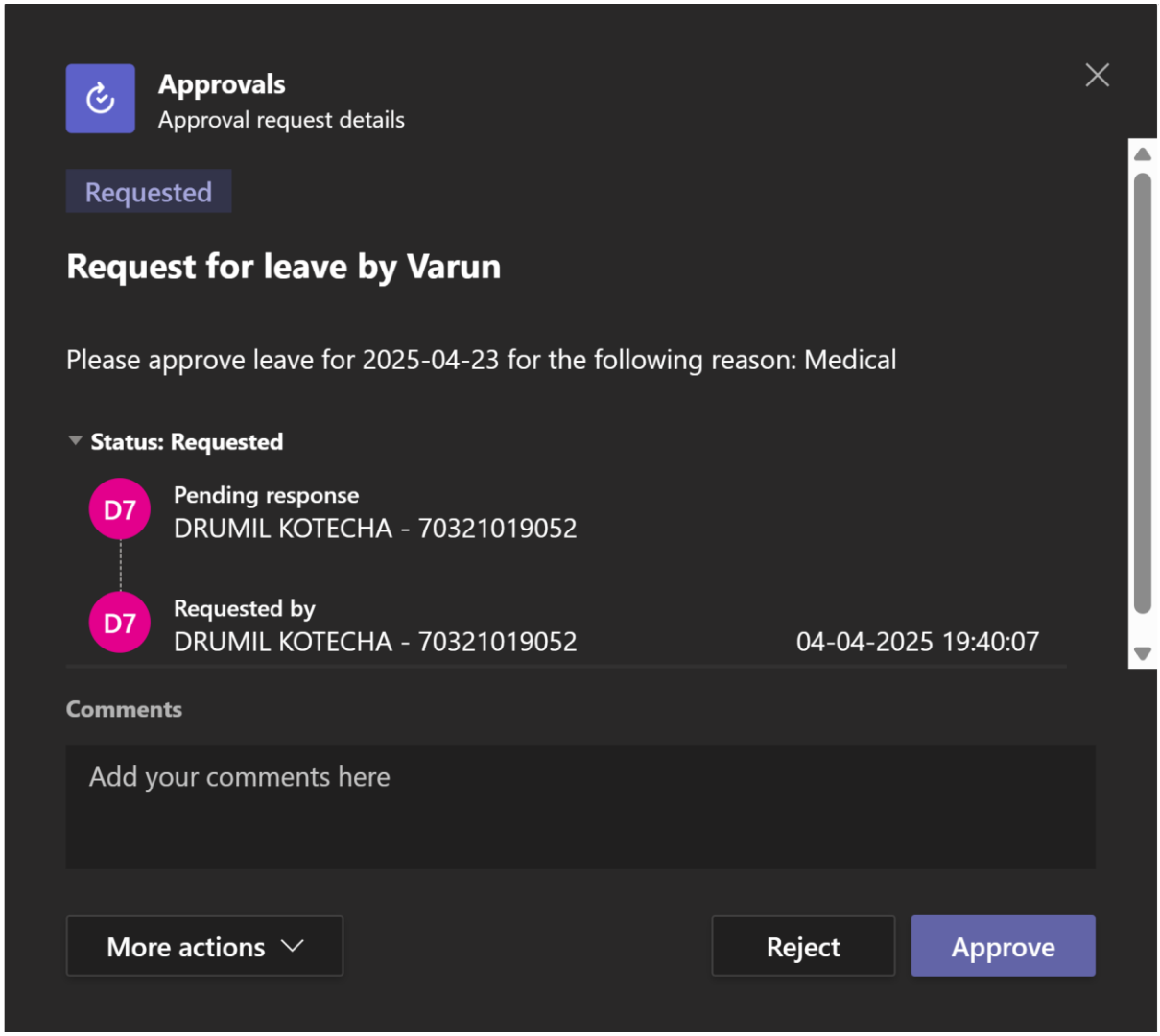
For Approve:



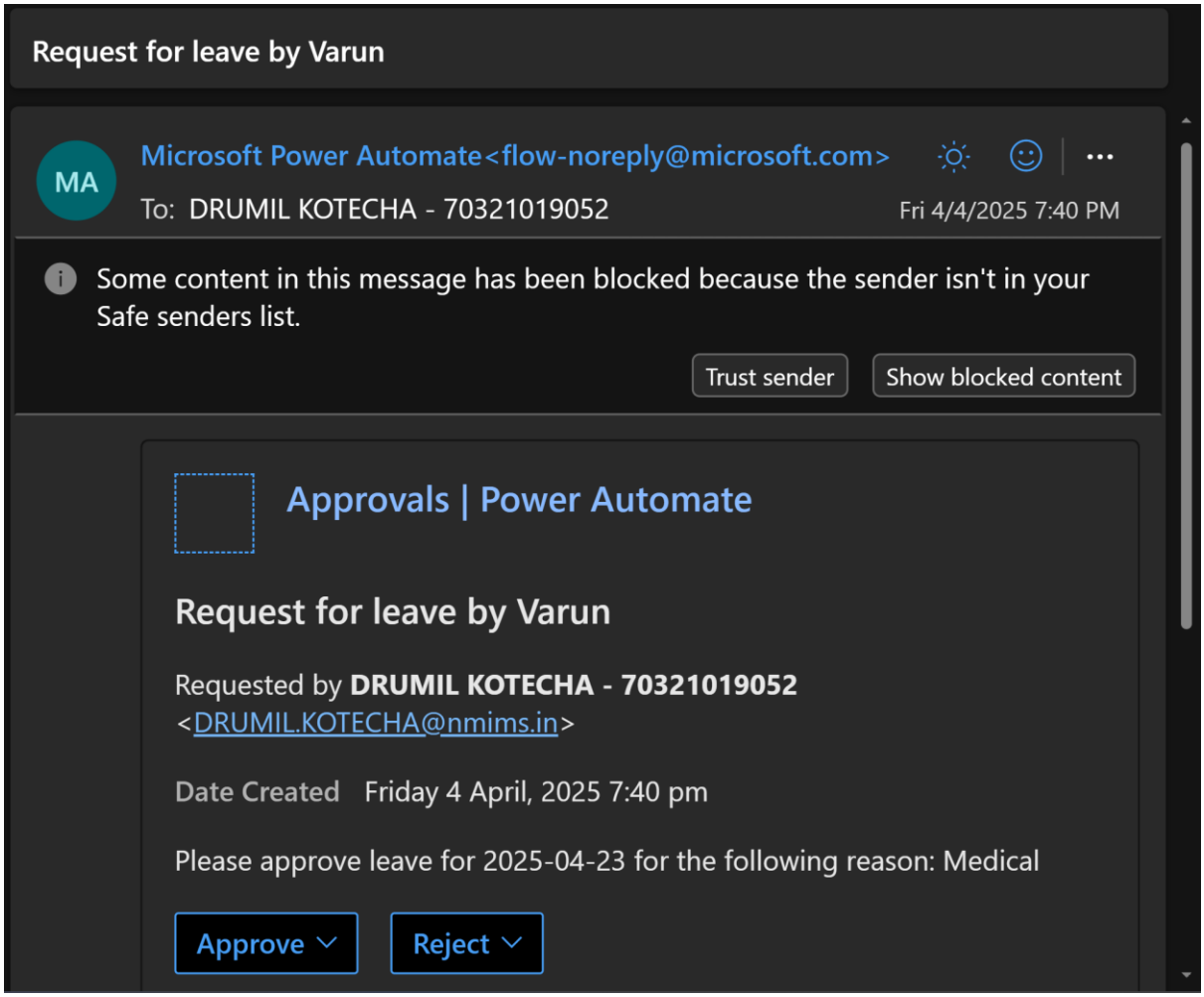
For deny:

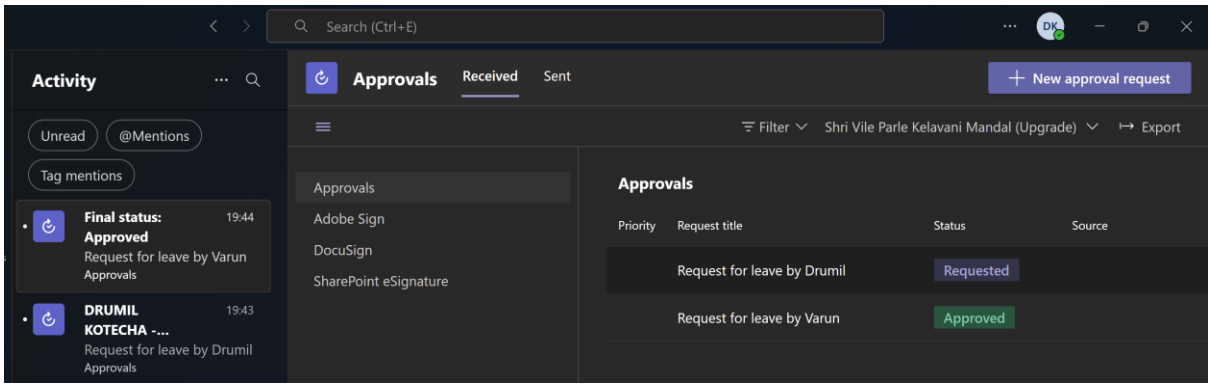


Approval Notification:

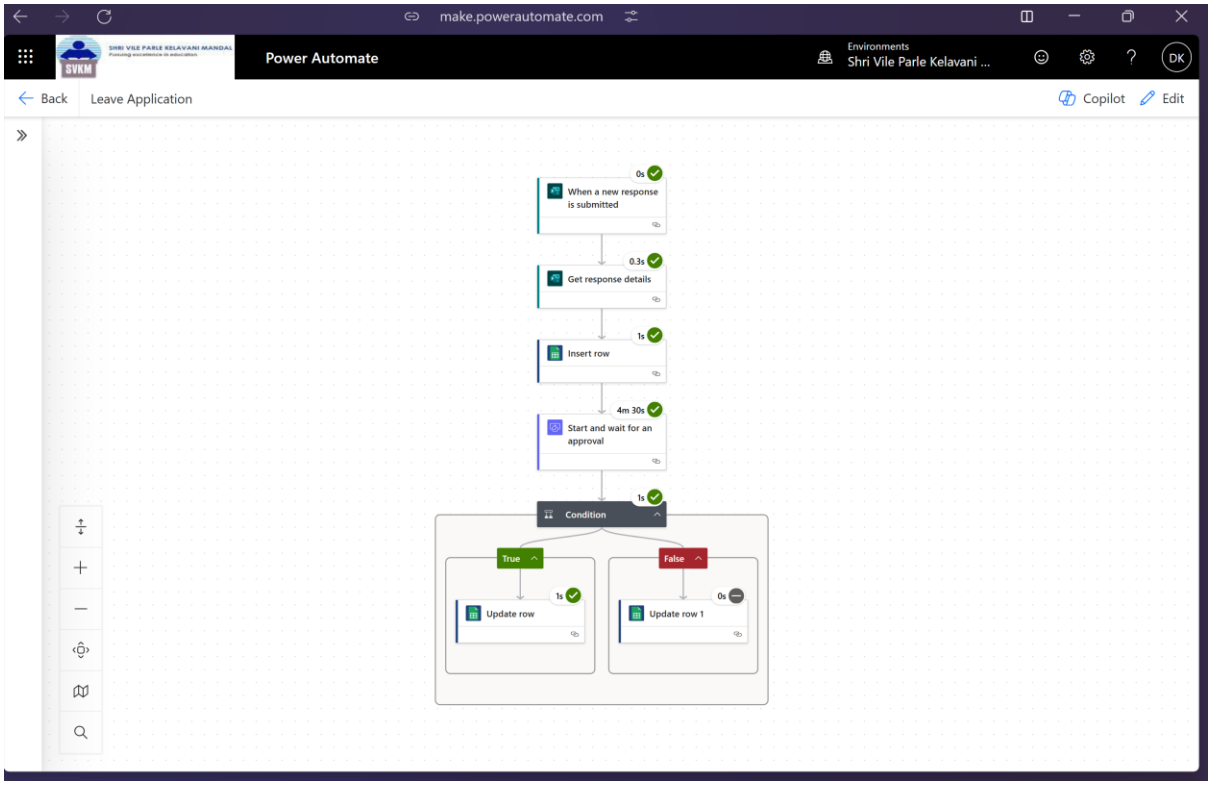

Approvals
Approval request details
Requested
Request for leave by Varun
Please approve leave for 2025-04-23 for the following reason: Medical
Status: Requested
D7
Pending response
DRUMIL KOTECHA - 70321019052
D7
Requested by
DRUMIL KOTECHA - 70321019052
04-04-2025 19:40:07
Comments
Add your comments here
More actions
Reject
Approve

After Approval, Email sent to Applicant:

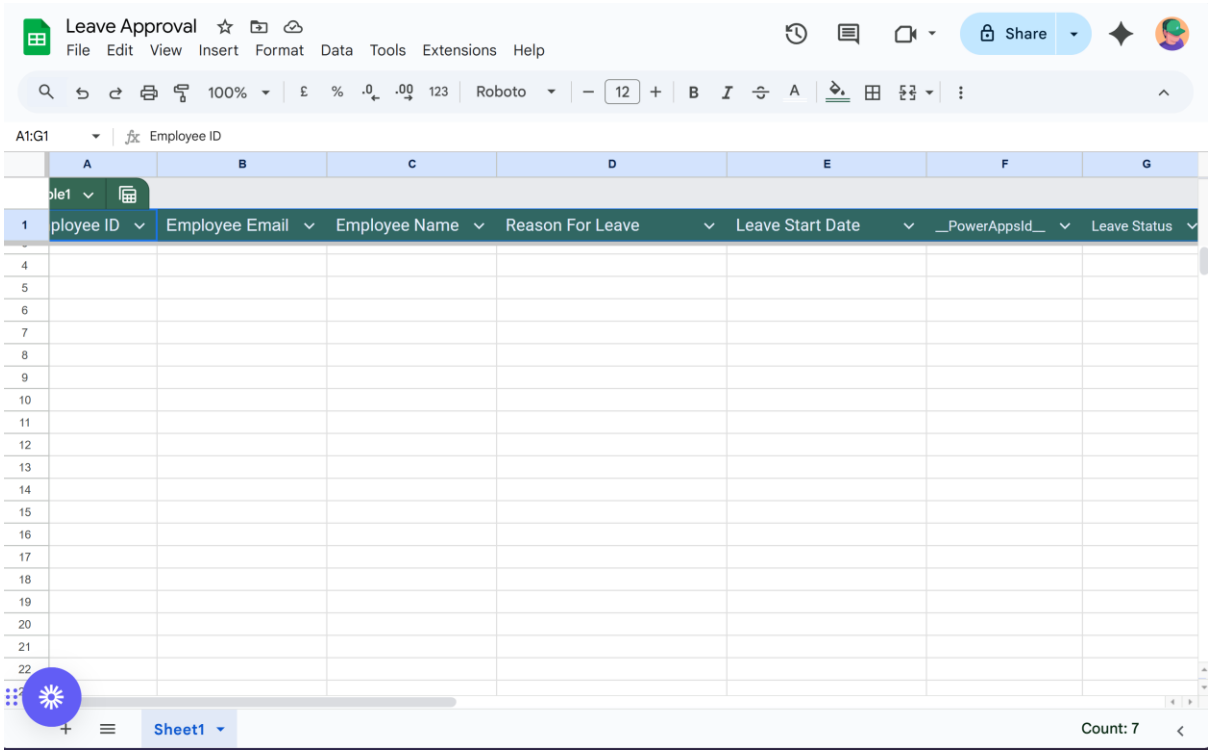




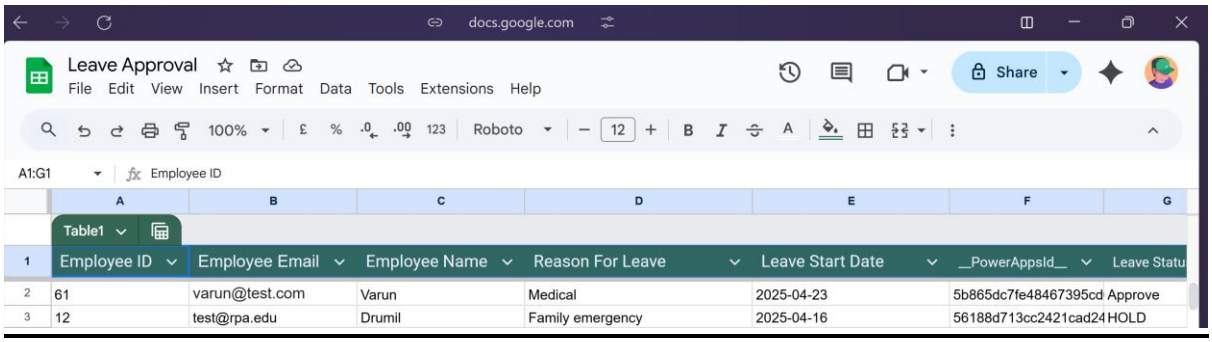
Leave application flow



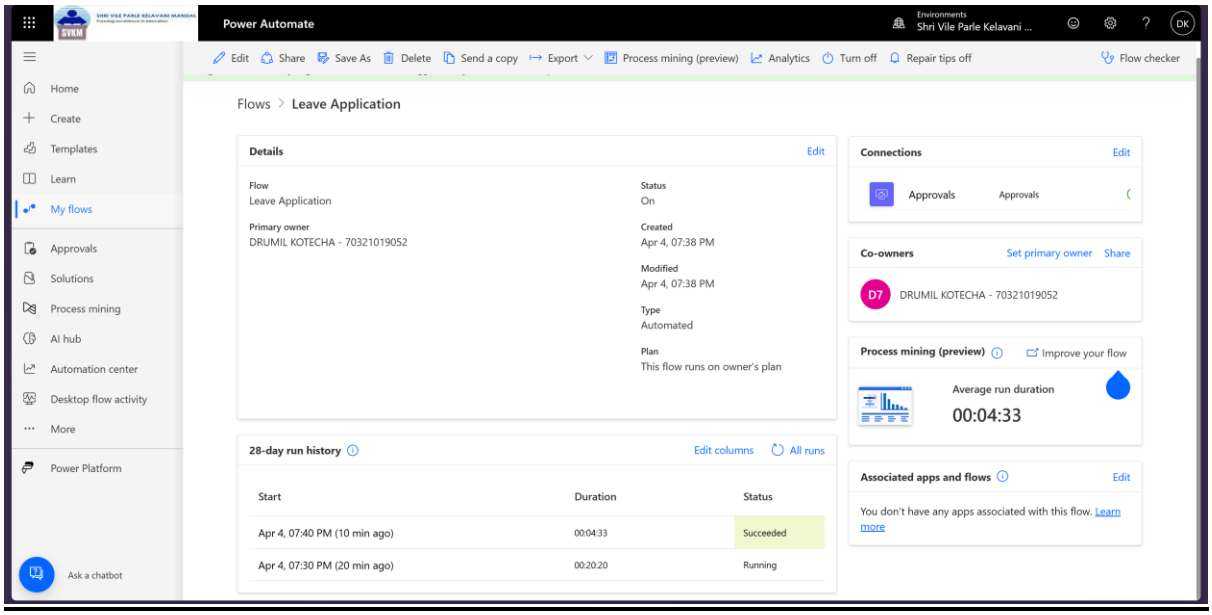
Google sheet before response submitted:



Google sheet after response submitted:



Conclusion:



Through this experiment, we learnt about power automate cloud flow

Date of Experiment: 14/4//25
Date of Submission:14/4/25

SVKM'S NMIMS
Mukesh Patel School of Technology Management & Engineering
Department of Computer Engineering

Subject- Robotic Process Automation
EXPERIMENT NO. 10

Objective: The objective of this procedure is to create a model-driven app for _____ (fill it as per your model app that you have designed)

Material Required:

- Access to Power Platform environment
- PowerApps Studio
- Power Automate (Flow)

Scenario: Describe the Scenario

Theory: Model-driven apps in PowerApps allow users to create sophisticated, data-centric applications without writing code. These apps are built on a relational data model, where entities (tables) represent different business objects, and relationships define how these entities are connected.

Procedure:

1. **Create a Solution:**
   - Open PowerApps Studio and create a new solution named "_____".
2. **Create Publisher:**
   - Click on "New Publisher" and create a publisher using your name with initials as the prefix.
3. **Create Course Table:**
   - Create a new table named "Course".
   - Add the following columns:

     _____
     _____
     _____
4. **Auto-creation of Contact Table:**
   - Upon creating the instructor field, it automatically adds the Contact table.
   - Add the following columns to the Contact table:

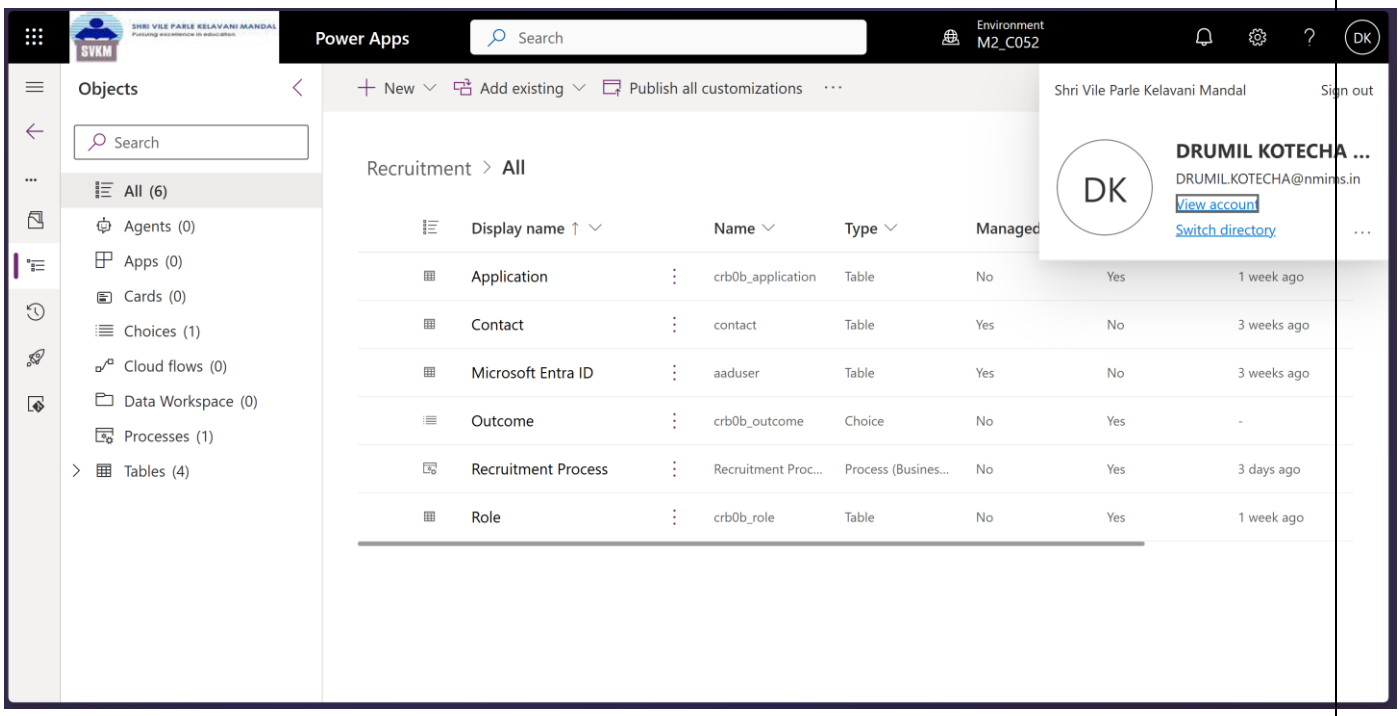
     _____
     _____
5. **Create Views:**

   _____
   _____

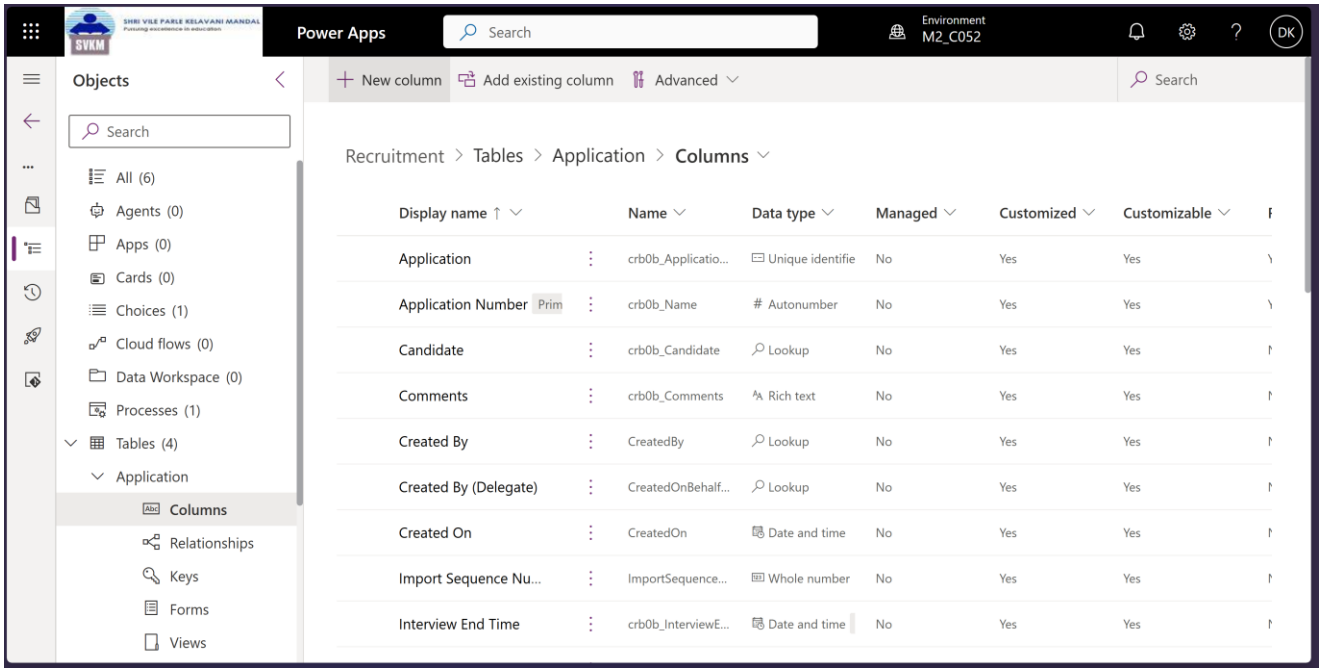
6. **Create new form:**
   - In the course table click on form
   - Click on New Form on the top
   - And select main form

7. Similarly add the steps that are required in your experiment.
8. **Create Model-Driven App:**
    - From the PowerApps Studio, create a new Model-Driven App.
    - Add existing tables: _____
    - Customize app layout and navigation.
9. **Testing:**
    - Test the app to ensure all functionalities work as intended.
    - Verify data input, approval process, and navigation flow.

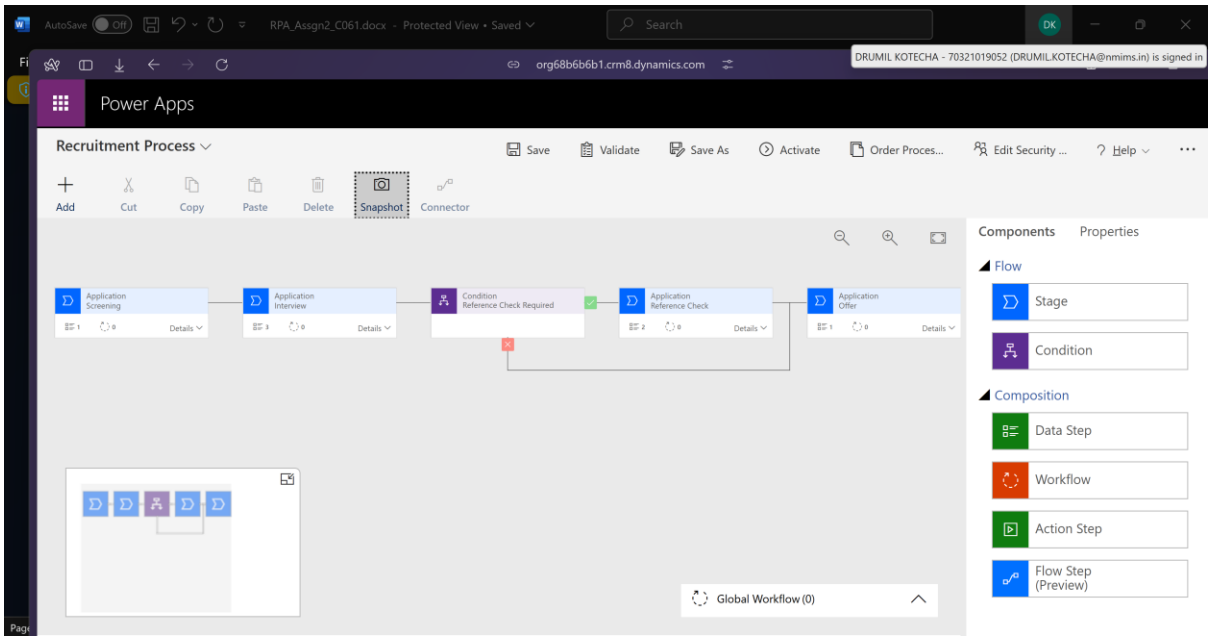
Screenshots of _____ (Create the Different Sections for Screenshots)



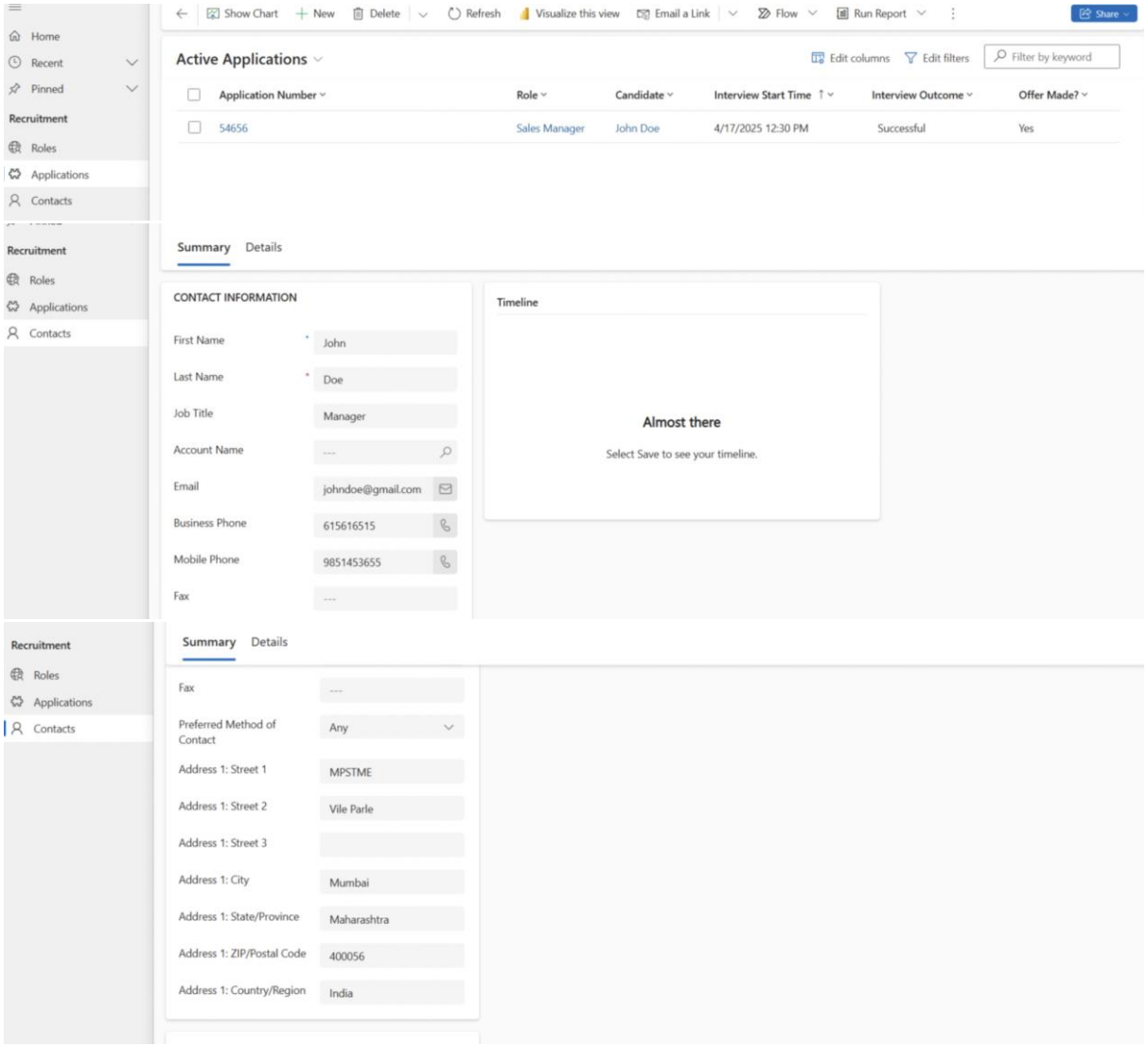
Application columns:



Business Process Flow:



Screenshots of Final app:



Conclusion:

Successfully implemented a module-driven app in Power Apps.

Reference:
https://www.youtube.com/watch?v=HrILchHvMUA&t=2922s