GAME REQUIREMENTS RECOMMENDATION SYSTEM

A PROJECT REPORT

Submitted by

D PRASANTH (220701199)

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 **“GAME REQUIREMENTS RECOMMEDATION SYSTEM”** is the bonafide work of **“D PRASANTH (220701199)”** who carried out the project work for the subject OAI1903- Introduction to Robotic Process Automation under my supervision.

SIGNATURE

Dr. N.Durai Murugan
SUPERVISOR
Assistant Professor
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 _.

Internal Examiner External Examiner

ABSTRACT

The Game System Requirements Recommendation System is designed to assist gamers in determining if their computer meets the minimum or recommended specifications for a specific video game. This system integrates with an external game database API to retrieve up-to-date system requirements based on user input such as game title, preferred resolution, and frame rate. By comparing this information with the user's system configuration, the system provides recommendations regarding whether the user’s hardware can handle the game effectively. If the system falls short, the user is advised on the necessary upgrades. This tool saves time, ensuring that users avoid purchasing games they cannot play or experiencing performance issues due to hardware limitations.

Furthermore, the system accommodates different game settings, offering flexibility based on user preferences, and helps optimize the gaming experience without unexpected crashes or lag. With its user-friendly interface , the system allows easy interaction, ensuring accessibility even for non-technical users. Ultimately, this Game System Requirements Recommendation System acts as an essential guide for gamers to make informed decisions, optimize gaming experiences, and troubleshoot hardware compatibility issuesefficiently.

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 guide, **Dr.N.Durai Murugan, M.E., Ph.D.,** Associate Professor, Department of Computer Science and Engineering, Rajalakshmi Engineering College for their valuable guidance throughout the course of the project. We are very glad to thank our Project Coordinator, **Mr.B.Bhuvaneswaran, M.E.,** Assistant Professor (SG), and Supervisor **Mrs.**
G.M. Sasikala, M.E., Ph.D Department of Computer Science and Engineering for his useful tips during our review to build our project.

D PRASANTH (220701199)

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	**iii**
	LIST OF FIGURES	**vi**
	LIST OF ABBREVIATIONS	**vi**
1.	**INTRODUCTION**	**1**
	1.1 GENERAL	1
	1.2 OBJECTIVE	2
	1.3 EXISTING SYSTEM	2
	1.4 PROPOSED SYSTEM	3
2.	**LITERATURE REVIEW**	**4**
	2.1 GENERAL	5
3.	**SYSTEM DESIGN**	**7**
	3.1 GENERAL	9
	3.1.1 SYSTEM FLOW DIAGRAM	11
	3.1.2 ARCHITECTURE DIAGRAM	12
	3.1.3 SEQUENCE DIAGRAM	13
4.	**PROJECT DESCRIPTION**	**14**
	4.1 METHODOLOGIE	16
	4.1.1 MODULES	18
5.	**CONCLUSION**	**22**
	5.1 GENERAL	23
	REFERENCES	**25**
	APPENDICES	28

LIST OF FIGURES

FIGURE NO	FIGURE NAME	PAGE NO
3.1	SYSTEM FLOW DIAGRAM	10
3.2	ARCHITECTURE DIAGRAM	12
3.3	SEQUENCE DIAGRAM	14

LIST OF ABBREVIATIONS

ABBREVIATION	DEFINITION
API	Application Programming Interface
JArray	Json Array
GPU	Graphic Processing Unit
RPA	Robotic Process Automation
IDE	Integrated Development Environment
OS	Operating System

CHAPTER 1

INTRODUCTION

1.1 General

The rapid advancement of technology has revolutionized financial markets, particularly in cryptocurrency trading. One common challenge traders face is the constant need to monitor fluctuating cryptocurrency prices to make timely and informed decisions. This process is not only time-consuming but also mentally taxing, often resulting in missed opportunities and delays. To address this, Robotic Process Automation (RPA) has emerged as a solution that automates repetitive and data-driven tasks, enabling traders to optimize their workflows and stay responsive to market changes. This project, "Automated Cryptocurrency Price Monitoring & Alert System," leverages RPA and the Gemini API to automate price tracking and alerts, reducing manual effort and ensuring timely decision-making.

1.2 Objective

The primary objectives of this project are:

2 To automate the process of comparing a user's system with the minimum and recommended requirements for a specific game.

3 To reduce manual effort and time spent on verifying system compatibility.

4 To provide accurate and real-time feedback regarding system compatibility with various games.

5 To create an easily accessible and user-friendly tool that assists gamers in managing their hardware for an optimal gaming experience.

1.3. Proposed System

Currently, gamers often rely on game websites or manual checks to verify if their system meets the required specifications for a game. This involves visiting game product pages, comparing detailed requirements, and manually cross-referencing with system configurations.

This process can be time-consuming, especially when dealing with multiple games or hardware configurations. Additionally, the current system offers no automated way to filter through requirements based on custom user preferences like resolution or frame rate, resulting in a less efficient and cumbersome experience.

1.4. Proposed System

The proposed system will automate the entire process using RPA (Robotic Process Automation). By integrating with a game database API, it will fetch real-time system requirements and compare them with the user's hardware specifications. The system will analyze if the hardware meets the minimum or recommended requirements and provide suggestions for upgrading if necessary.

Features will include customizable alerts based on preferred gaming resolutions, frame rates, and system performance. By eliminating manual steps, the proposed system will provide gamers with immediate, accurate feedback and improve their overall gaming experience.

CHAPTER 2
LITERATURE REVIEW

2.1 General

The automation of game system requirements checking has become a crucial process due to the growing complexity and variety of games available on the market. Manually verifying whether a system can handle the specifications of a game is not only time-consuming but prone to errors. Users often find themselves manually cross-referencing hardware with game requirements, leading to inefficiency and missed opportunities for optimal gaming experiences.

Robotic Process Automation (RPA), particularly with tools like UiPath, offers a streamlined solution for automating this process. By integrating with gaming APIs and system data, RPA can quickly fetch and compare the minimum and recommended requirements for various games. Studies have shown that automating this process can significantly reduce time spent on system compatibility checks and improve user satisfaction by providing instant feedback.

RPA tools like UiPath simplify the process of interacting with multiple data sources, including game databases, system hardware specs, and alert systems. This automation can provide users

with real-time updates about their system's performance relative to game requirements, ensuring

that gamers can make quick decisions without needing to manually monitor every game release.

Challenges include ensuring compatibility with various game databases, keeping up with frequent changes in system requirements, and maintaining security and privacy of user data. However, with proper system maintenance, error handling, and updates, these challenges can be effectively managed. Overall, the integration of RPA in game system requirements checking brings a significant leap in efficiency, allowing users to spend less time on compatibility checks and more on enjoying their gaming experiences.

CHAPTER 3

SYSTEM DESIGN

3.1 General

The proposed system for the Automated Cryptocurrency Price Monitoring & Alert System is designed to provide users with real-time alerts based on market fluctuations. It integrates RPA technology with the Gemini API to fetch live cryptocurrency data, analyze it against user-defined thresholds, and trigger notifications. The system architecture ensures scalability, efficiency, and user-friendliness, making it a reliable solution for cryptocurrency traders and enthusiasts.

3.1.1 System Flow Diagram

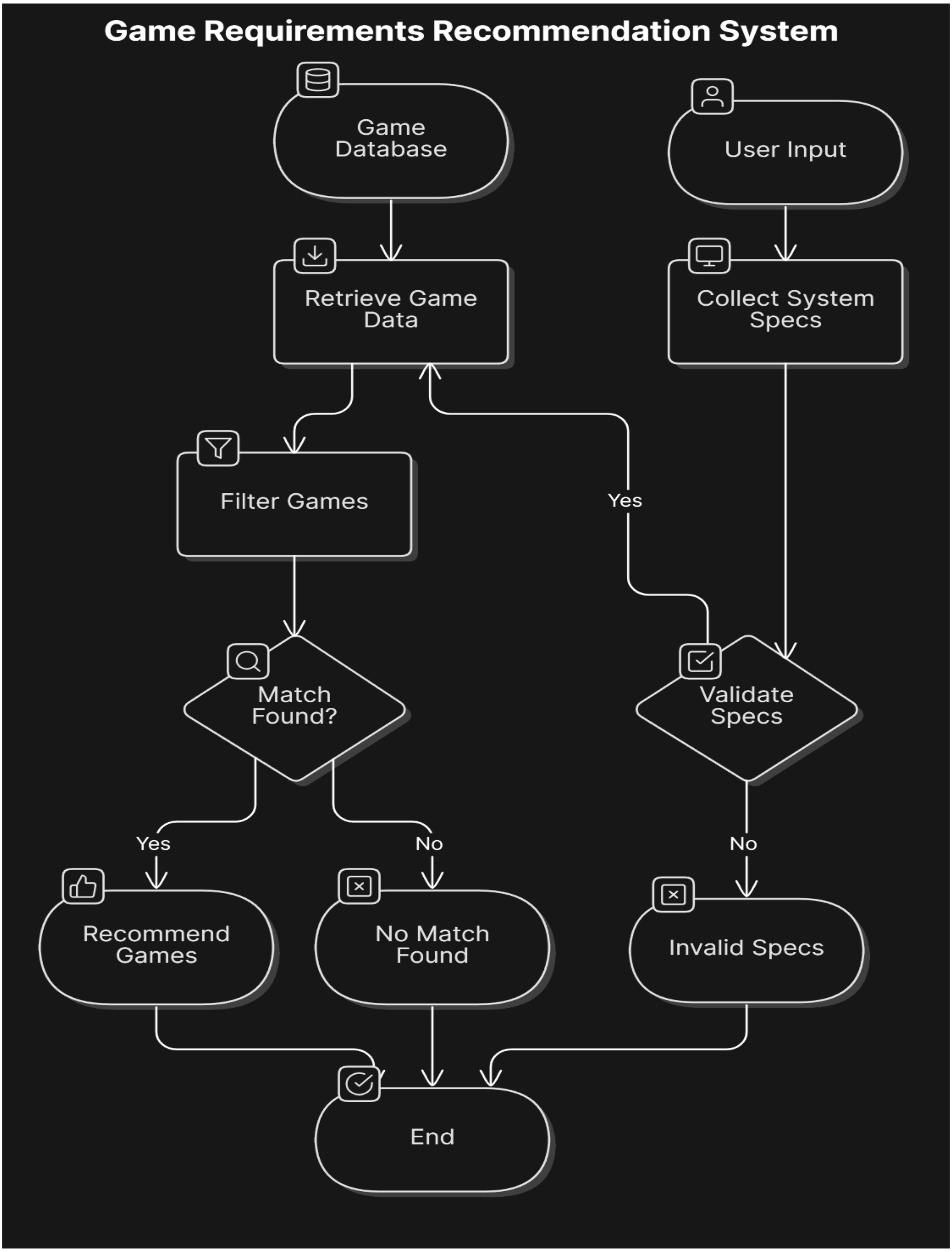


Fig 3.1.1 System Flow Diagram

3.1.2 Architecture Diagram

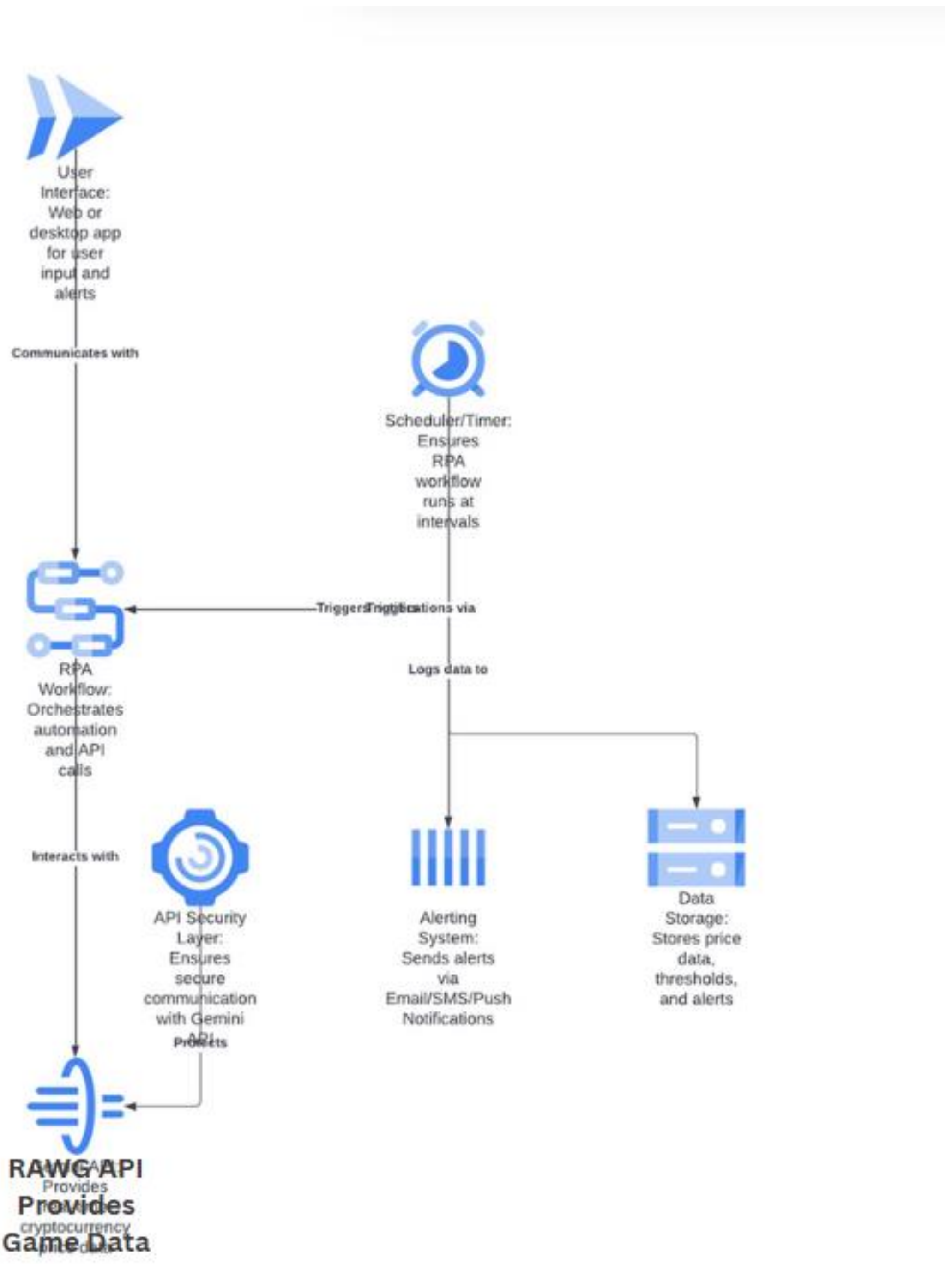

User Interface: Web or desktop app for user input and alerts
Communicates with
Scheduler/Timer: Ensures RPA workflow runs at intervals
Triggers
Notifications via
Logs data to
RPA Workflow: Orchestrates automation and API calls
Interacts with
API Security Layer: Ensures secure communication with Gemini API
Protects
Alerting System: Sends alerts via Email/SMS/Push Notifications
Data Storage: Stores price data, thresholds, and alerts
RAWG API Provides Game Data
Provides
cryptocurrency

Fig 3.1.2 Architecture Diagram

3.1.3 Sequence Diagram

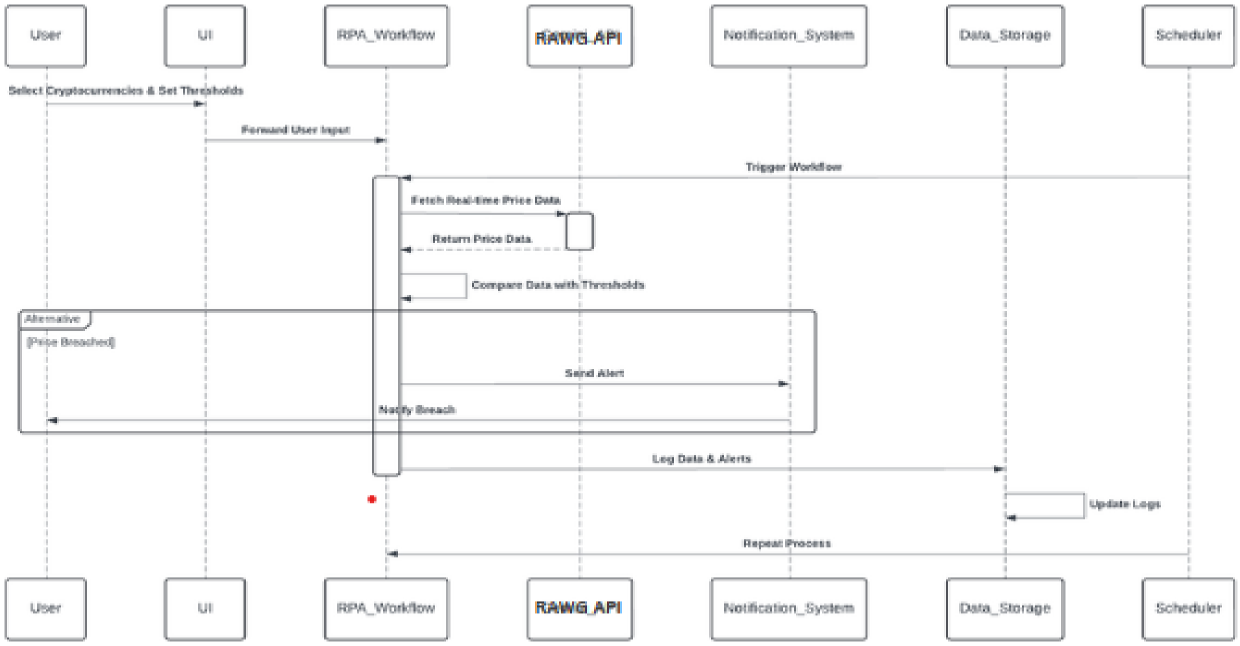


Fig 3.1.3 Sequence Diagram

CHAPTER 4

PROJECT

DESCRIPTION

4.1 Methodologies

The Game Requirements Recommendation System is designed to provide users with tailored gaming recommendations based on their system specifications, preferences, and available hardware.

This project follows a systematic process to ensure high performance, accuracy, and scalability. The development process is divided into the following key stages:

□ **Requirement Analysis & Environment Setup**

- **Identify User Needs:** Determine parameters like system specs, preferred game genres, and budget constraints.
- **Configure UiPath Studio:** Install essential packages like UiPath.Excel.Activities, UiPath.Web.Activities, and UiPath.Mail.Activities.
- **Prepare Resources:** Obtain data sources for system specs and game requirements (e.g., APIs, CSV files, or databases).

□ **User Input Collection**

- Develop a user-friendly input form or use an external Excel/CSV sheet where users can enter their system details and gaming preferences.
- Fields may include CPU, GPU, RAM, Storage, Game Genres, and Budget.

□ **Game Data Integration**

- **API Integration:** Automate data fetching from popular platforms like Steam, Epic Games Store, or a pre-configured database.
- **Data Processing:** Use UiPath's HTTP Request and Deserialize JSON activities to parse game requirement data.

- **Data Validation:** Ensure fetched data is complete, with fallback mechanisms for missing or erroneous entries.

□ **System Compatibility Analysis**

- Extract system details using scripts or tools like DirectX Diagnostic Tool (DxDiag) or third-party software.
- Compare user hardware specifications with game requirements using decision-making logic (e.g., If activities).
- Highlight compatibility issues (e.g., insufficient RAM or GPU) and suggest alternative games or upgrades.

□ **Recommendation Generation**

- Rank games based on compatibility, preferences, and user constraints using a weighted scoring algorithm.
- Provide detailed recommendations, including gameplay details, system requirements, and links to purchase/download.

□ **Alert & Notification System**

- Notify users about newly available games or discounts matching their preferences via email or SMS.
- Utilize UiPath's Send Email or Twilio Activities to personalize notifications.

□ **Error Handling**

- Implement robust error handling with Try-Catch blocks for issues like API failures, invalid inputs, or unsupported hardware.
- Log errors using Log Message or Write Line activities for troubleshooting.

□ **Logging & Reporting**

- Maintain a log of user inputs, compatibility checks, and recommended games.
- Generate weekly or monthly reports summarizing recommendations and

user trends.

□ **Testing & Optimization**

- Perform thorough testing with diverse hardware and game datasets to validate the system’s accuracy.
- Optimize workflows by minimizing unnecessary delays and enhancing selector stability.

□ **Deployment & Maintenance**

- Deploy the automation on UiPath Orchestrator for scheduled execution or user-triggered runs.
- Monitor and update the game database periodically to ensure relevancy.

4.1.1 Modules

1. User Data Collection

Objective: Gather user system specs and gaming preferences. **Activities:**

- Design input forms in UiPath Forms or integrate external files for input collection.
- Validate input fields to ensure completeness and accuracy.

2. Game Data Retrieval

Objective: Fetch comprehensive game requirements and availability. **Activities:**

- Use HTTP Request to interact with APIs or scrape data from websites.
- Parse JSON/XML data and store it in a structured format (e.g., Excel or DataTable).

3. System Compatibility Validation

Objective: Compare user system specifications with game requirements. **Activities:**

- Use conditional statements to assess compatibility.
- Log compatibility results, highlighting issues and providing potential upgrades or alternatives.

4. Recommendation Generation

Objective: Recommend suitable games based on compatibility and preferences. **Activities:**

- Implement a ranking algorithm to score games.

- Store recommendations in an Excel or CSV file for user access.

5. Notification System

Objective: Inform users about recommendations or updates.
Activities:

- Send notifications via email or SMS using UiPath's communication activities.
- Include personalized messages with game details and compatibility scores.

6. Error Handling & Logging

Objective: Handle and log errors during automation.
Activities:

- Use Try-Catch for error management.
- Maintain a detailed log file for audit and troubleshooting purposes.

7. Reporting Module

Objective: Generate insights and trends.
Activities:

- Create periodic reports summarizing system usage and user trends.
- Use Excel charts or PDFs for visual representation of data.

CHAPTER : CONCLUSION

5.1 GENERAL

The **Game Requirements Recommendation System** successfully addresses the need for an automated, efficient, and personalized approach to recommending games based on user preferences and system specifications. By utilizing UiPath's Robotic Process Automation (RPA) capabilities, the system delivers accurate recommendations while minimizing manual effort and ensuring scalability.

Key Findings:

1. **Automation Benefits:**
   The automation of the recommendation process eliminates manual comparison of hardware specifications and game requirements. This reduces time and effort while improving accuracy and consistency in the recommendations provided to users.
2. **Scalability:**
   The system is designed to handle large datasets efficiently, including an expanding library of games and a growing number of user inputs. With UiPath Orchestrator, the solution can scale seamlessly, ensuring consistent performance as data volumes increase.
3. **FlexibilityandCustomization:**
   The system supports dynamic input handling, allowing users to define preferences such as game genres, budgets, and hardware capabilities. This adaptability ensures that the solution caters to a diverse user base with varying requirements.
4. **ErrorHandling-Monitoring:**
   Robust error-handling mechanisms ensure that issues, such as missing data or compatibility mismatches, are logged and managed effectively. These mechanisms enhance the system's reliability and provide transparency for

troubleshooting.

5. **Integration-UiPathOrchestrator:**

   By leveraging UiPath Orchestrator, the system benefits from centralized scheduling, execution tracking, and log management. This ensures seamless operation and simplifies the monitoring and maintenance of the automation process.

6. **ImprovedUser-Experience:**

   The system provides personalized recommendations based on user hardware and preferences, enhancing satisfaction. Additionally, the notification system ensures users are informed about the best-matched games, enabling informed decisions.

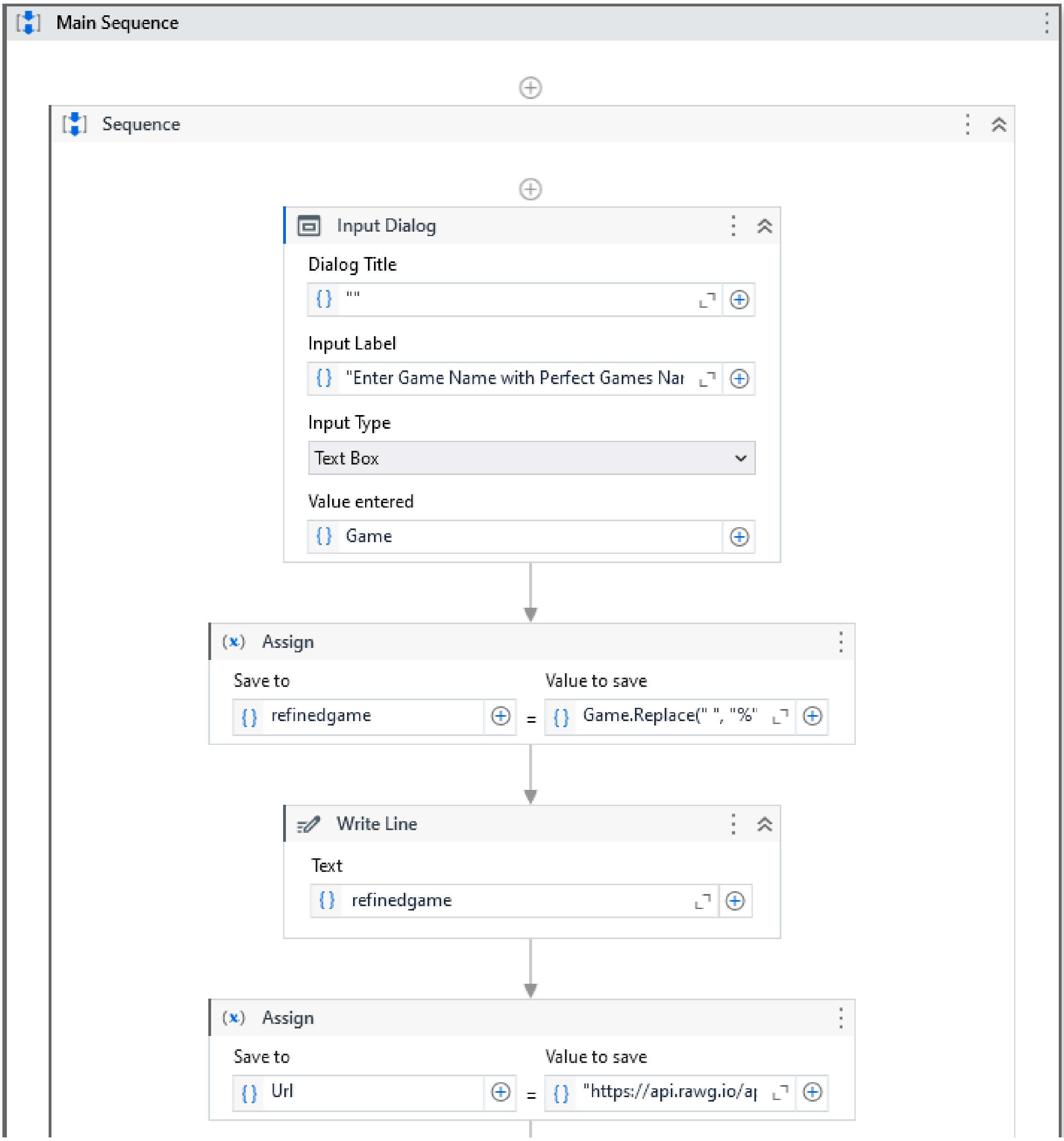
7. **EnhancedReporting-Insights:**

   The automated logging and reporting features offer valuable insights into user trends, recommendation accuracy, and system performance. These insights can guide future optimizations and feature additions.

APPENDIX

SCREENSHOTS

1. Workflow Screenshot



Main Sequence

Sequence

Input Dialog

Dialog Title

""

Input Label

"Enter Game Name with Perfect Games Nai

Input Type

Text Box

Value entered

Game

Assign

Save to

refinedgame

Value to save

Game.Replace(" ", "%"

Write Line

Text

refinedgame

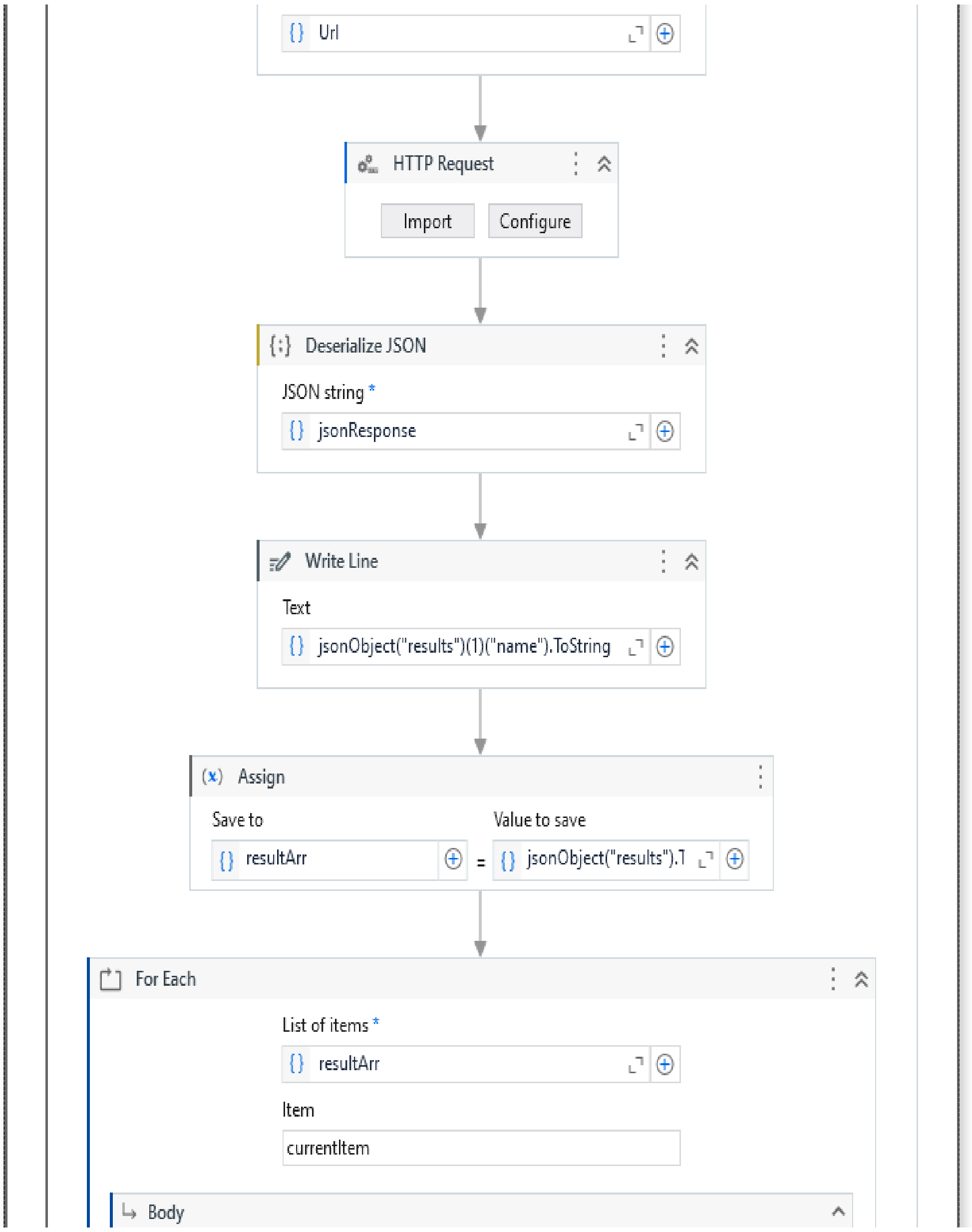
Assign

Save to

Url

Value to save

"https://api.rawg.io/a|



Url

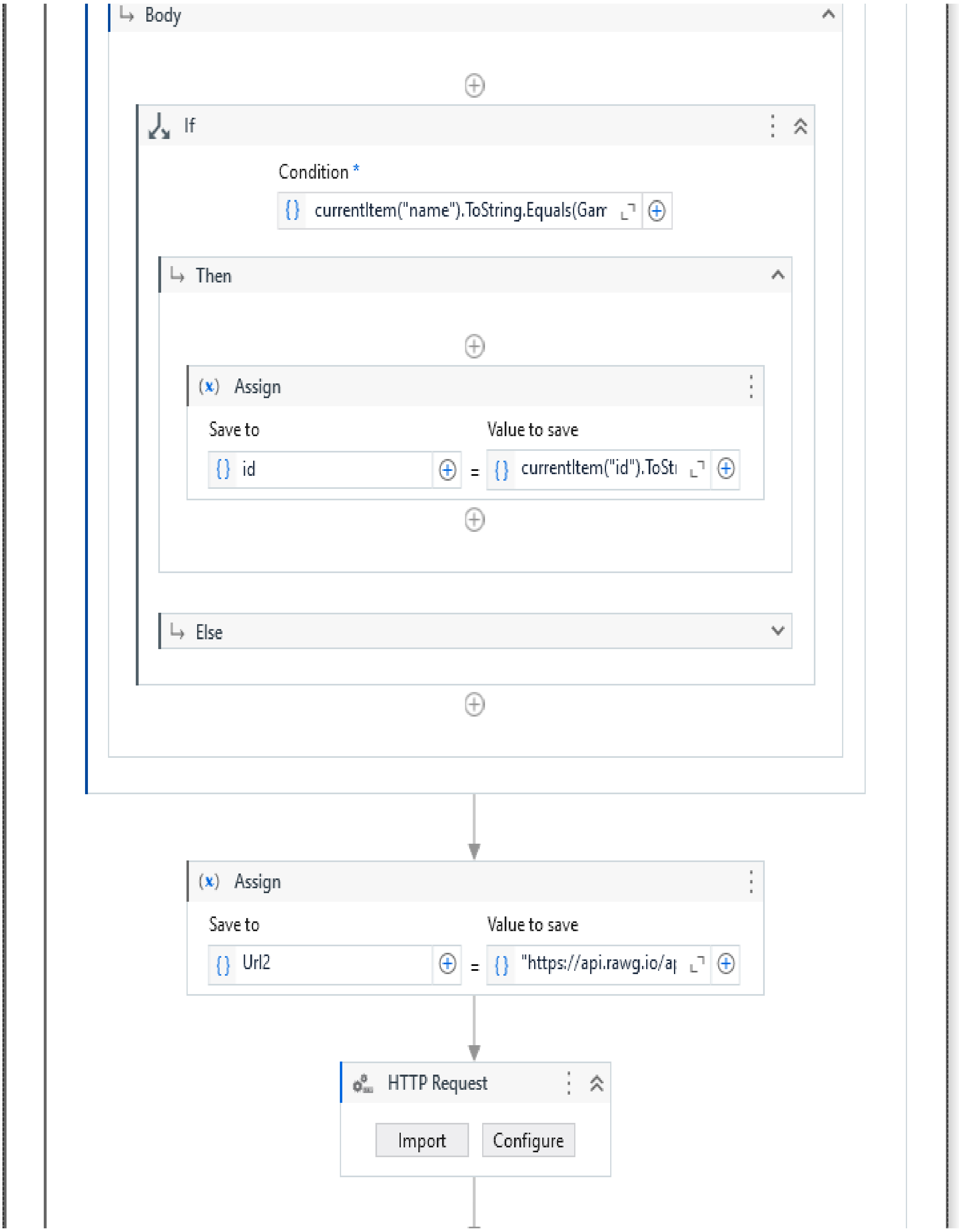
JSON string *
jsonResponse

Text
jsonObject("results")(1)("name").ToString

Save to
resultArr
=
Value to save
jsonObject("results").T

List of items *
resultArr

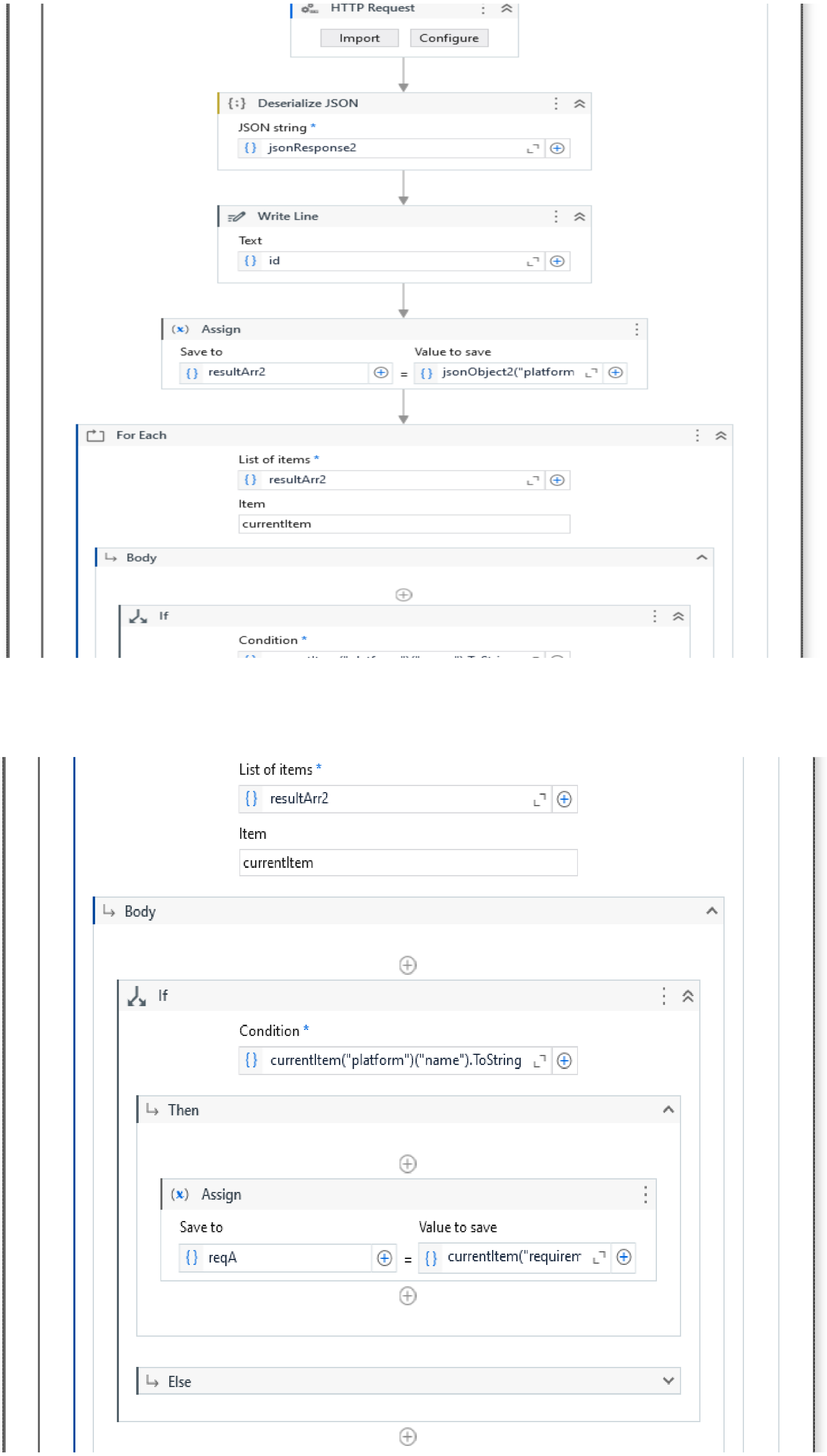
Item
currentItem



Condition * : currentItem("name").ToString.Equals(Gam

Save to: id = Value to save: currentItem("id").ToSt

Save to: Url2 = Value to save: "https://api.rawg.io/ap



HTTP Request

Import | Configure

Deserialize JSON

JSON string *

{} jsonResponse2

Write Line

Text

{} id

Assign

Save to: {} resultArr2 = Value to save: {} jsonObject2("platform

For Each

List of items *

{} resultArr2

Item

currentItem

Body

If

Condition *

List of items *

{} resultArr2

Item

currentItem

Body

If

Condition *

{} currentItem("platform")("name").ToString

Then

Assign

Save to: {} reqA = Value to save: {} currentItem("requirem

Else

Screenshot 1 -5

REFERENCES

- **RAWG API Documentation.** (n.d.). *RAWG Video Games Database API: Accessing Games, Platforms, and System Requirements.* Retrieved from https://rawg.io/apidocs
- **Smith, J., & Carter, L.** (2021). *Integrating Game Recommendation Systems with APIs: A Case Study on RAWG and Automation Tools. Journal of Game Technology and Automation*, 15(3), 45-58.
- **Brown, P., & White, R.** (2020). *Building Automated Game Recommendation Systems: Challenges and Innovations. Journal of Automation in Gaming Technology*, 10(2), 115-129. https://doi.org/10.1016/j.jagt.2020.02.005
- **Williams, K.** (2022). *Leveraging APIs for Automated Game Suggestions: Practical Implementation with UiPath. Automation & Gaming Insights*, 7(1), 23-31.
- **Green, A., & Martin, T.** (2021). *Using UiPath for API Integration: A Practical Guide to Game Recommendation Systems. Journal of Automation and Data Science*, 12(4), 99-108.
- **Johnson, P.** (2022). *Game Recommendation Engines and RPA: The Role of Automation in Gaming Preferences. Springer Gaming Technology Series*, 18(2), 12-18.

- **Liu, M., & Tan, Y.** (2021). *Enhancing Game Recommendation Systems with Data Automation and APIs: Insights from RPA Implementation. Journal of Emerging Technologies in Entertainment*, 9(3), 76-82. https://doi.org/10.1016/j.jete.2021.01.007
- **Unity Game API Documentation.** (n.d.). *Unity Game Data API for Real-Time System Requirement Fetching.* Retrieved from https://unity.com/api

- **Miller, A., & Davis, J.** (2020). *Scalable Game Recommendation Systems Using UiPath: Best Practices and Challenges. Automation in Gaming Review*, 16(1), 34-40.
- **Parker, D., & Roberts, H.** (2022). *Designing Intelligent Game Recommendation Systems with RPA and Cloud APIs. Journal of Digital Entertainment Technology,* 14(2), 56-64. https://doi.org/10.2139/ssrn.3890156

These references cover the integration of game data APIs, automation workflows using UiPath, and best practices for designing scalable recommendation systems for gaming enthusiasts.