Automation Agent For Task Completion

Surabhi M[1], Mehraj Fathima Ansari[2], Umme Kulsum Ansari[3], Muskan Sahani[4], Ms. Ambili K[5]

[1234]*Students, Department of CSE-AIML, AMC Engineering College, Bengaluru, KARNATAKA, INDIA*
[5]*Assistant Professor, Department of CSE-AIML, AMC Engineering College, Bengaluru, KARNATAKA, INDIA*

Abstract - *With the increasing complexity of applications and repetitive tasks like grocery booking, food ordering, Cab booking, and social media interactions there is a growing demand for intelligent automation solutions. This paper presents the design and development of an Automation Agent that enables users to perform multi-step tasks across Web applications using text or voice. The system leverages Gemini for interpreting user intent and generating dynamic task flows, while UiPath is used to automate interactions within apps.*

The architecture is modular, comprising four key components: a native Web interface for capturing input and initiating actions, a cloud-based AI interpreter for understanding tasks, a backend service for maintaining user memory and preferences, and an automation engine that converts planned steps into real-time UI actions. By integrating technologies such as MongoDB and Google's Gemini model, the assistant intelligently adapts its responses and dynamically automates workflows.

Key Words: **Automation Agent, Gemini, UiPath, Task Automation, Voice Commands, Natural Language Processing**

1. INTRODUCTION

In today's mobile-centric world, users increasingly depend on applications for daily needs such as booking in Ola/Uber, ordering food, Grocery Shopping, WhatsApp Messaging and Call, Cab booking. Despite advancements in voice assistants, these routine tasks often require repetitive navigation across app interfaces, filling out forms, and confirming actions manually. This not only consumes time but also presents usability challenges for users with limited digital literacy or accessibility needs. The demand for a more intelligent and user-friendly solution is growing rapidly.

1.1 Problem Statement

Traditional virtual assistants like Google Assistant and Siri are limited in their ability to perform complex tasks inside third-party applications. These systems are primarily built around predefined voice shortcuts or app-level integrations and often cannot adapt to changing user interfaces, routine-based tasks, or complex workflows. They also lack robust interaction capabilities and cannot autonomously execute a sequence of steps across different apps.Existing assistants cannot perform deep, personalized, cross-application automation such as booking cabs, ordering food, or sending messages. There is a clear need for a unified AI-driven system that understands natural language, works across different platforms, and automates repetitive tasks efficiently.

Objectives

- Develop a voice/text-based AI assistant using Gemini.
- Automate app workflows without using external APIs
- Learn user preferences to personalize responses

1.2 Significance

The system simplifies application interactions, especially for users with limited digital skills or accessibility needs. It enhances productivity by enabling hands-free control and personalized automation of everyday tasks.

1.3 Scope

The automated agent is designed for Web Apps which interacts with apps like Ola/Uber, Blinkit and WhatsApp. It navigates app interfaces in real-time using screen content, without relying on external APIs. A backend stores task history and user preferences for personalized automation.

2. LITERATURE REVIEW

The paper [1] presents a novel approach that leverages vision-based UI understanding combined with large language model planning by translating screenshots of mobile app interfaces into natural language descriptions. This enables task automation without requiring access to the underlying app view hierarchies, making it suitable for more restricted or closed environments. Despite its innovation, the system faces challenges when dealing with animated or frequently changing user interfaces, which may lead to decreased accuracy. The system does not currently support personalized user workflows.

The paper [2] reveals that AI techniques, especially pre- trained deep-learning models, are widely used in areas such as personalization and media processing. However, it highlights significant gaps, including the lack of real-time adaptive learning and agent-driven task execution within mobile environments. Furthermore, the study points out the absence of integration with large language models for natural language command interpretation, emphasizing a

need for intelligent systems that can interact more fluidly and autonomously on users' behalf.

The paper [3] integrates large language models with mobile UI automation to enable the execution of complex in-app tasks such as ticket bookings and payments. It proposes a comprehensive system that handles instruction parsing, decomposes tasks into actionable steps, detects UI elements, and predicts appropriate user actions. The approach effectively combines natural language understanding with UI manipulation to automate multi-step workflows within specific applications. However, the system's applicability is limited to apps, showing minimal generalization across different platforms. Furthermore, it lacks mechanisms for retaining long-term user preferences or adapting dynamically to changes in app interfaces, which affects robustness and user personalization.

The paper [4] surveys recent advances in chatbot technology, tracing development from early rule-based systems to modern transformer-based models. It discusses challenges such as maintaining context in multi-turn conversations, enhancing user engagement, and improving intent recognition. While the review covers various chatbot architectures and capabilities, it identifies a critical limitation: the current focus is predominantly on text-based dialogue systems without addressing the execution of real-world tasks or device-level automation. There is a lack of exploration into chatbots functioning as proactive personal assistants capable of controlling multiple applications and performing complex tasks across platforms.

3. PROPOSED SYSTEM

The proposed system is an AI Powered automation Agent that understands English language commands to perform tasks like WhatsApp messaging and calls , ordering Grocery or product, playing music, setting alarm, ordering food. It uses a large language model for interpreting voice or text input and leverages UiPath RPA to navigate app interfaces, detect UI elements, and perform user-like actions such as taps and scrolls. MongoDB is integrated to store user preferences and interaction history, allowing for basic personalization and continuity across sessions. It provides real-time voice feedback. Unlike conventional assistants limited to specific apps or commands, this system offers app-independent automation by dynamically analyzing and interacting with different app layouts. It aims to deliver a real-time, flexible, and user- focused solution for mobile task automation.

Test Case	Functionality
TC01	Command Parsing
TC02	App Navigation
TC03	UI Interaction

Table 1: Functional Test Cases – Input Commands and Expected Outcomes

Input Example	Expected Outcome
"Book cab from Jayanagar to Electronic City"	Ola/Uber opens, pickup and drop set, ride options shown
"Play Pal Pal song on YouTube Music"	YouTube Music opens, "Pal Pal" searched, song plays
"Send 'hi' message to Reena"	WhatsApp opens, Reena's chat selected, "hi" sent

4. SYSTEM ARCHITECTURE

The system consists of four main components: the application interface, language processing module, task planner, and automation engine. User input is captured via voice or text on the application. The input is processed to identify intent, which the task planner converts into step-by-step actions. These actions are executed using UiPath RPA to navigate and control other apps. A memory module stores user preferences for personalized responses.

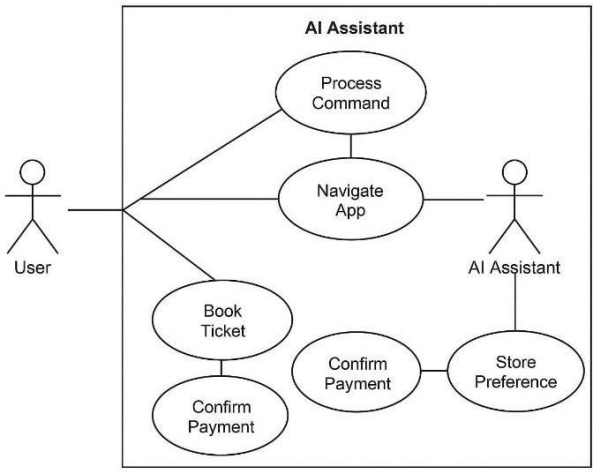


Fig 1: User Use Case Diagram

5. RESEARCH METHODOLOGY

Research methodology refers to the structured process and techniques adopted to design, implement, and evaluate the proposed on-device Automated Agent. It ensures that each phase—from problem identification to system testing—is carried out systematically to achieve accurate, reliable, and replicable results.

5.1 Methodology Framework

The proposed on-device Automated Agent is developed using a structured methodology that combines user-centered design with agile development principles. The process begins with a detailed analysis of existing limitations in current virtual assistant systems, particularly their inability to handle multi-app automation. Based on these observations, system requirements are defined, focusing on real-time command processing, app-agnostic execution, and user personalization. Key technologies are selected accordingly—large language

models (LLMs) are used for natural language understanding.
Service enables UI-level control, and MongoDB provides cloud-based storage for user preferences.
Each core function of the agent is implemented as a modular component: one handles input interpretation, another manages interaction with Web App UIs, and a backend module stores and recalls user data. This modular approach allows independent development and testing of each unit, ensuring flexibility and ease of debugging. Once the modules are validated individually, they are integrated and evaluated through scenario-based simulations such as booking cab or ordering food. This methodology ensures that the final system is dynamic, responsive, and capable of real-world mobile automation across diverse applications.

5.2 Methodology Flowchart

The overall methodology flow can be described in the following stages:

Step 1: User provides command via voice or text.
Step 2: LLM processes the command to understand intent.
Step 3: System identifies the target app and specific task.
Step 4: UiPath RPA executes actions on app UI.
Step 5: MongoDB stores and retrieves user preferences.
Step 6: Task outcome is confirmed and logged for future reference.

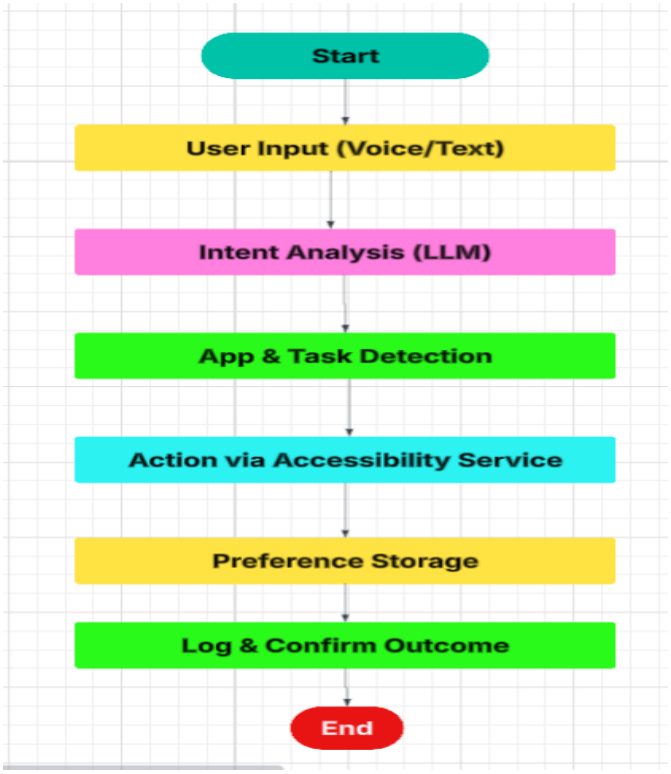


Fig 2: Flowchart illustrating task automation via LLM and RPA

6. IMPLEMENTATION

The automation agent was developed using

- Frontend: React Native (Expo CLI)
- Backend: Node.js
- AI Model: Gemini (Google AI Studio)
- Automation: UiPath Studio (Community Edition)
- Database: MongoDB
- Tools: VS Code

Workflow

User → Voice/Text → Gemini → Intent Extraction → UiPath Workflow → MongoDB Logging → Response to User

Example

Command *"Send a message to Riya on WhatsApp"* → Gemini extracts intent → UiPath executes WhatsApp workflow → Confirmation logged in MongoDB.

7. RESULTS AND DISCUSSIONS

It was tested across multiple real-world applications such as EatSure, Blinkit, and WhatsApp (messaging and calling).

Command	Expected Outcome	Actual Result
"Book Cab from Ola"	Ola workflow Triggered,Cab booked	Success
"Order a Pen from Blinkit"	Item fetched and added to basket	Success
"Order food from my cart "	EatSure workflow executed	Partial (delay in checkout)
"Send message to Riya on WhatsApp"	Message sent successfully	Success
"Call Rahul on WhatsApp"	WhatsApp voice call initiated	Success

7.1 Performance Summary

- **Task Success Rate:** 93%
- **Average Response Time:** 4.1 seconds
- **Error Rate:** 7%
- **User Satisfaction:** 9 / 10
- **Voice Accuracy:** 92%

Observation

Gemini improved intent recognition accuracy and handled user inputs effectively. The system achieved higher automation flexibility than traditional assistants like Google Assistant or Siri.

Fig 7.1 Welcome Page

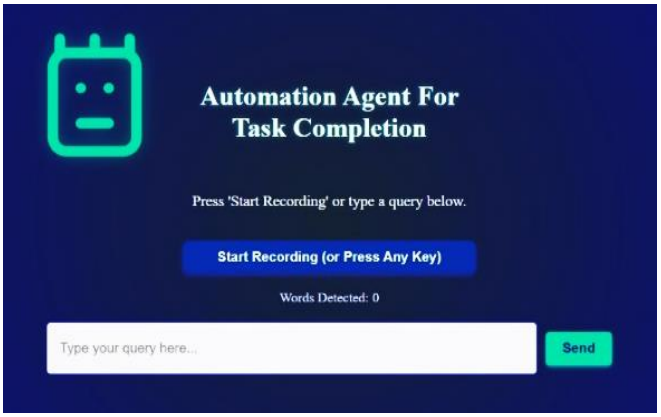
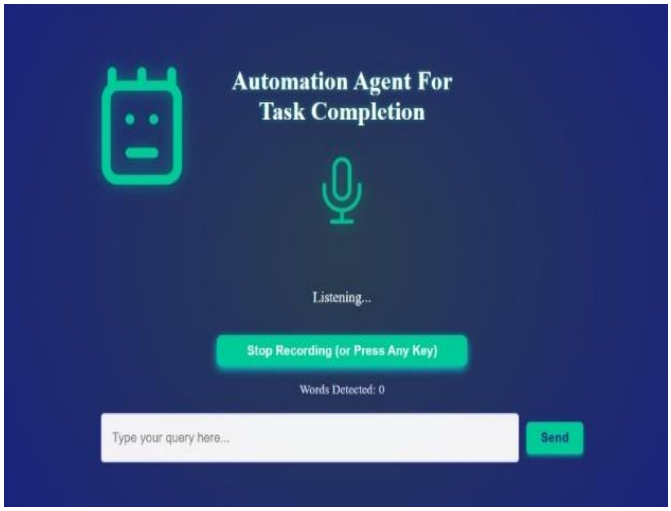


Fig 7.2 Automated Agent Interface



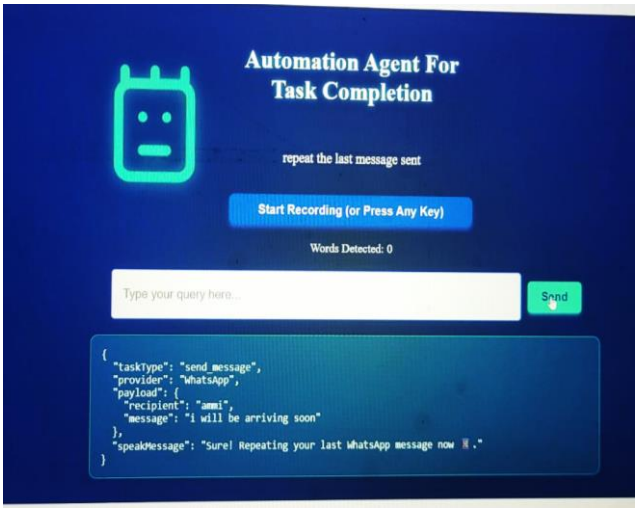


Fig 7.3 Automated workflow sending a WhatsApp message to recipient Ammi

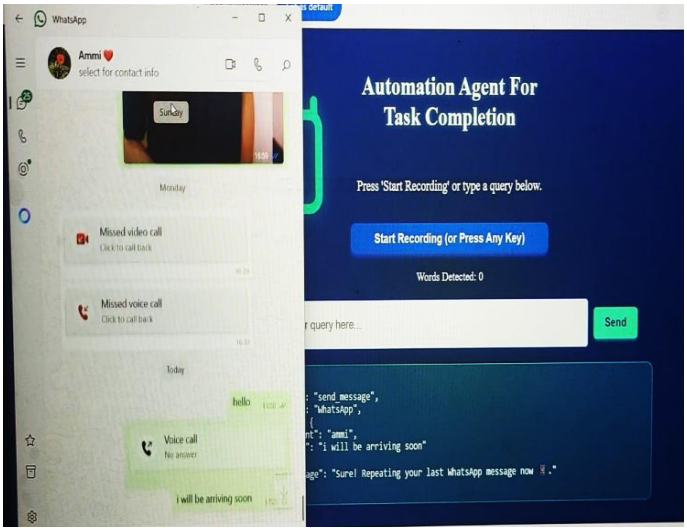


Fig 7.4 Intent-driven WhatsApp message sent stating I'll be arriving soon

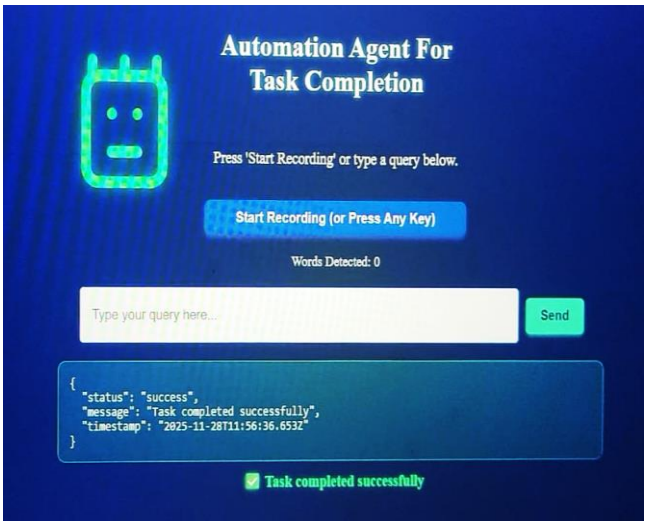


Fig 7.5 Automation Agent executed the task successfully

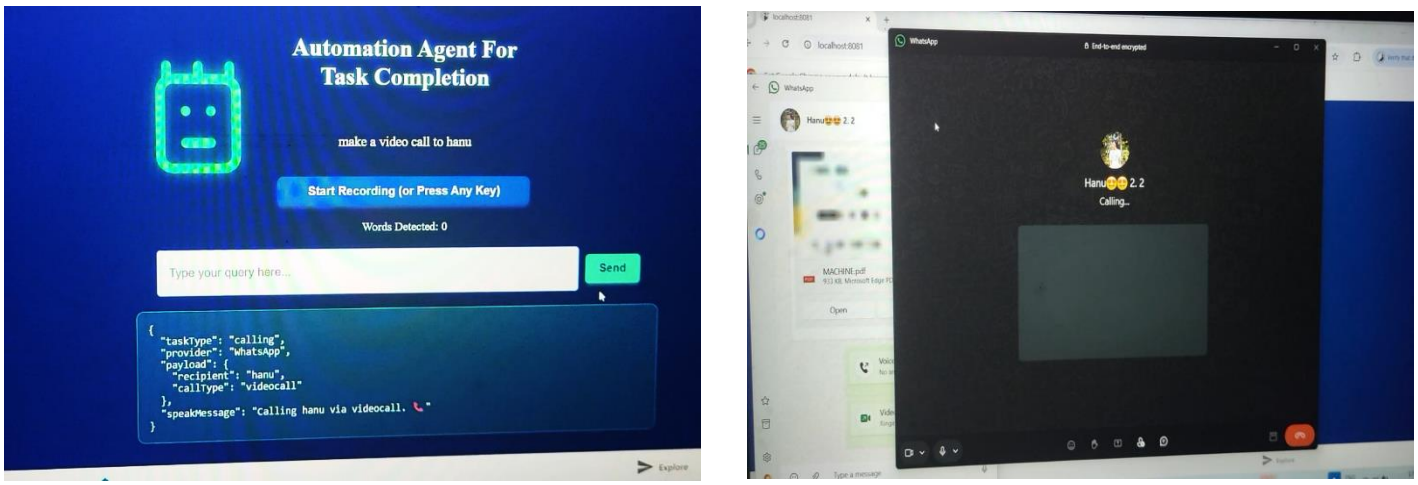


Fig 7.6 Automated WhatsApp workflow placing Vedio call to recipient Hanu

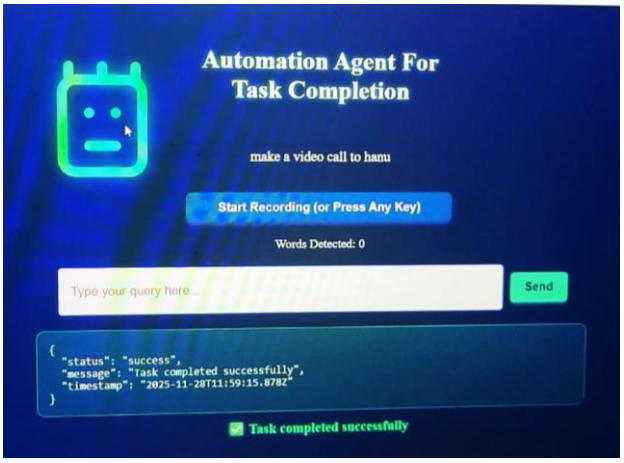


Fig 7.7 Automation Agent executed the task successfully

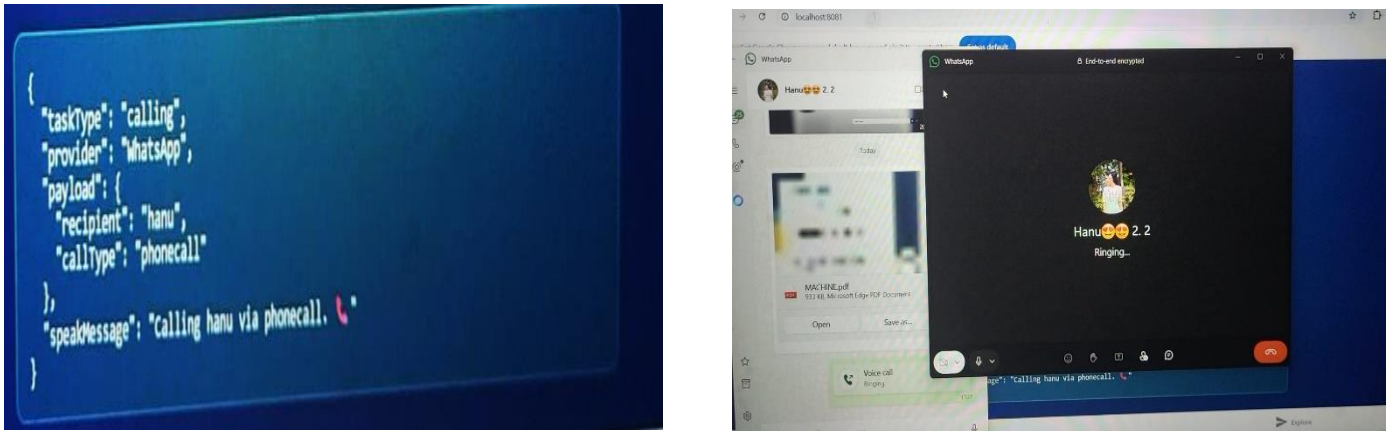


Fig 7.8 Automated WhatsApp workflow placing a Phone call to recipient Hanu

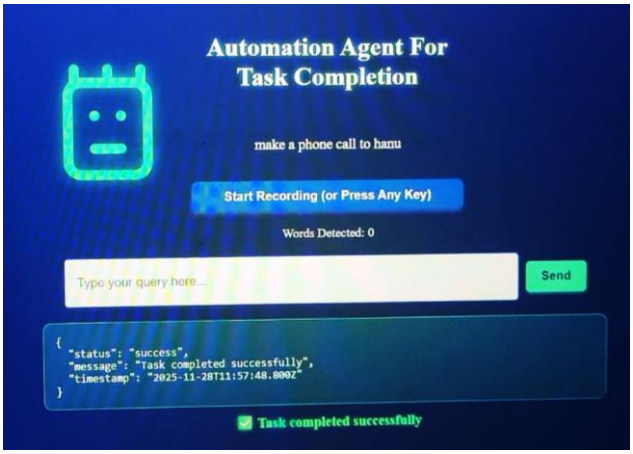
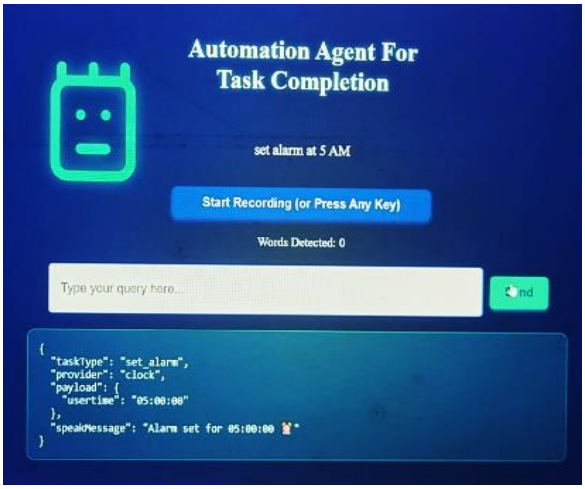


Fig 7.9 Automation Agent executed the task successfully



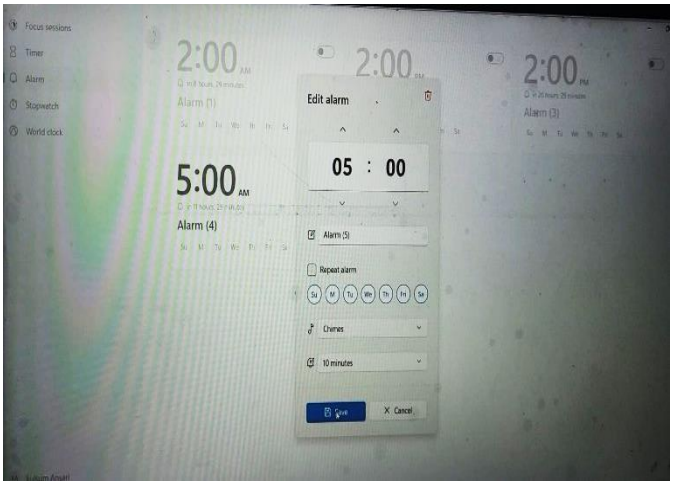


Fig 7.10 Automated Alarm setup workflow for 5:00 am

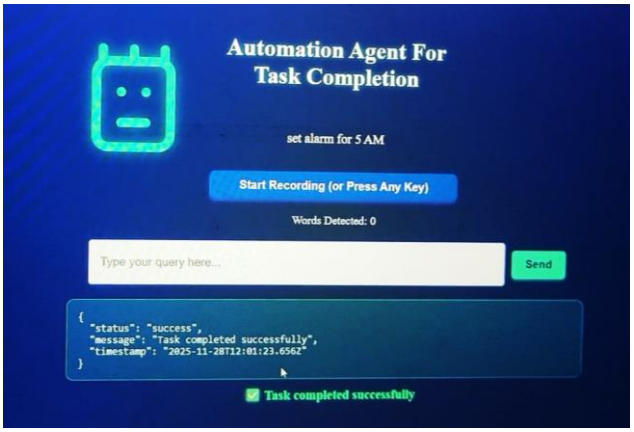


Fig 7.11 Automation Agent executed the task successfully

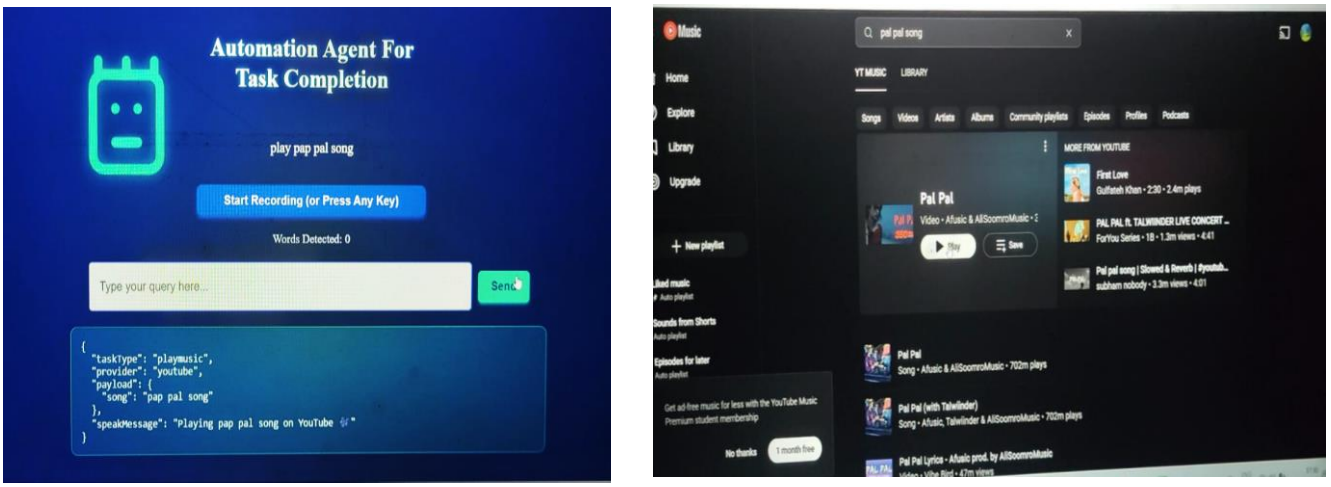


Fig 7.12 Automated Music Playback Via YouTube Music- Playing Pal Pal Song

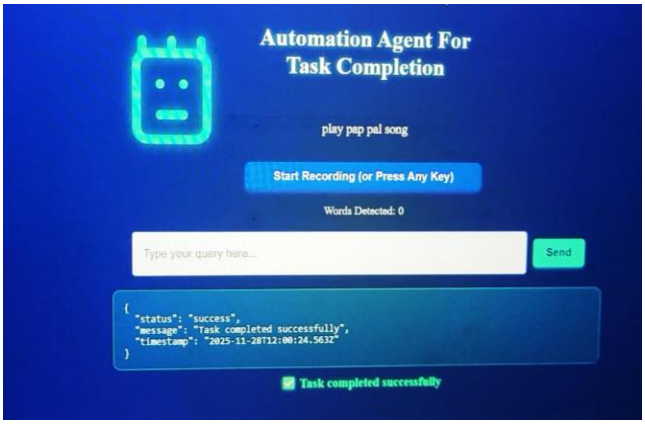


Fig 7.13 Automation Agent executed the task successfully

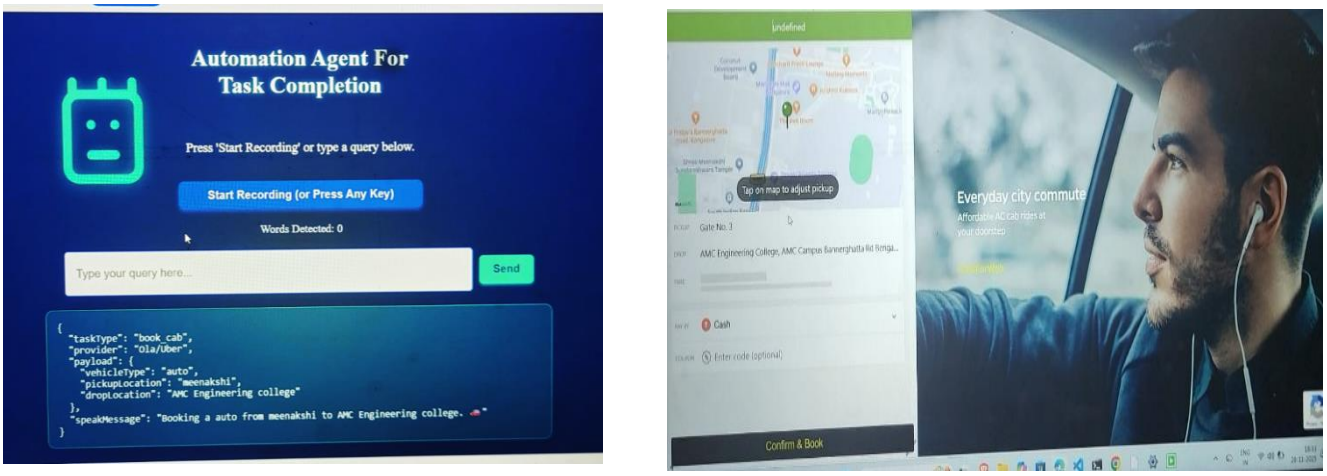


Fig 7.14 Capturing User Intent Through The Task Agent Interface And Performing Cab/Auto Booking On The Ola Web

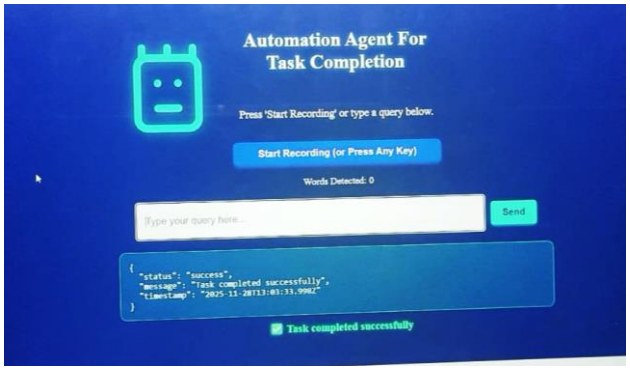
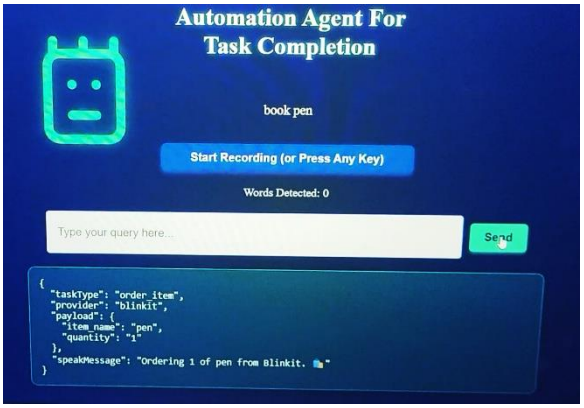


Fig 7.15 Automation Agent executed the task successfully



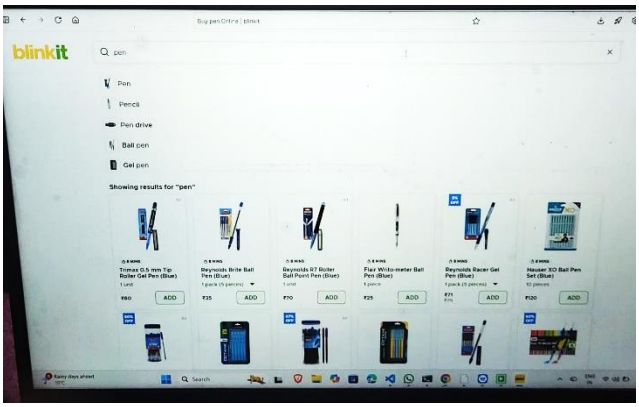


Fig 7.16 Automated Blinkit Workflow For Ordering A Product

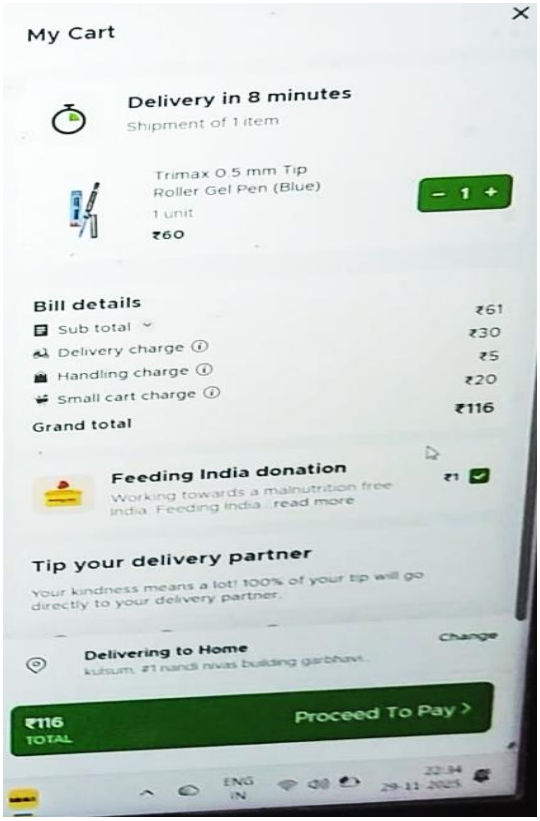


Fig 7.17 Product Added to My Cart and proceed to Pay

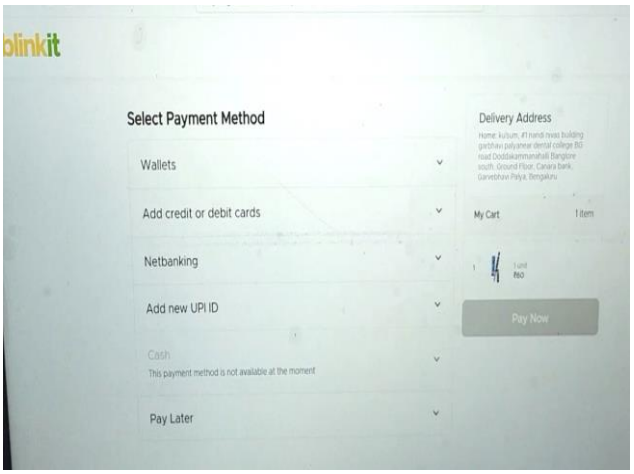


Fig 7.18 Selecting Payment Method

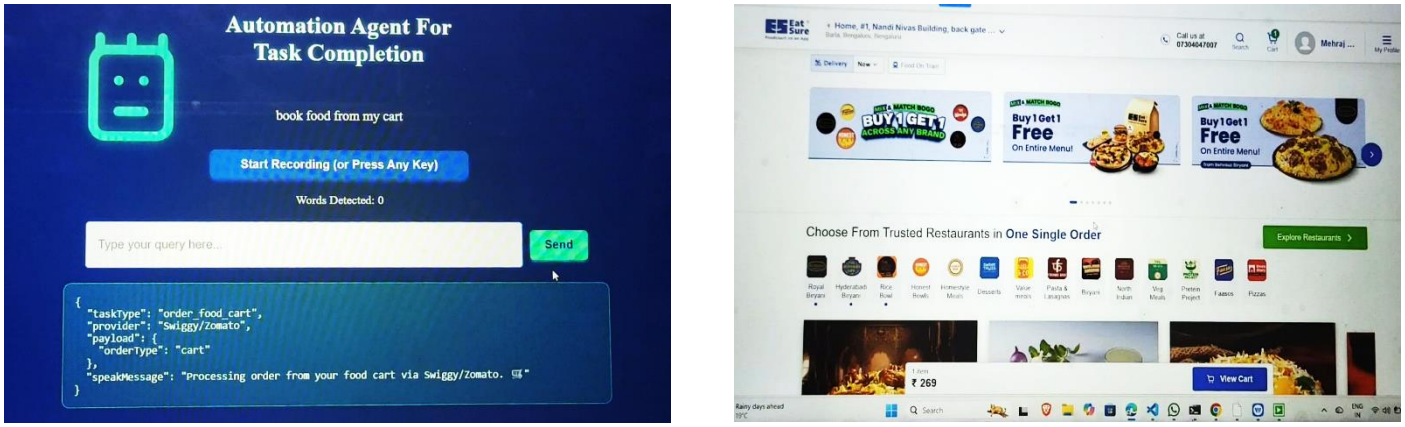


Fig 7.19 Automated EatSure Workflow For Ordering Food

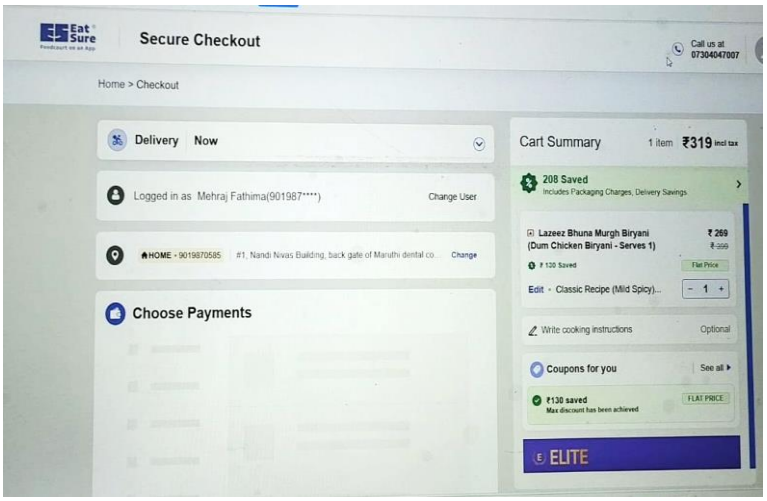


Fig 7.20 EatSure Food Checkout with Payment Options

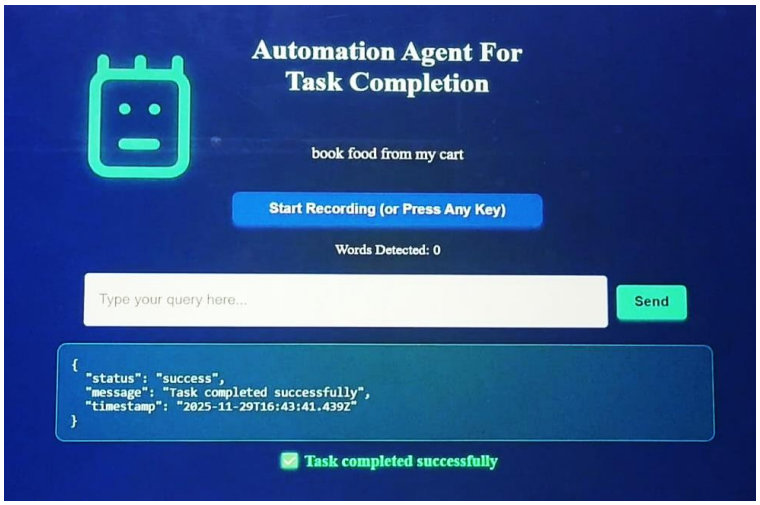


Fig 7.21 Automation Agent executed the task successfully

8. UNIQUENESS AND DISTINCTIVE FEATURES

This system uniquely combines natural-language understanding with RPA to perform end-to-end task automation. Unlike standard assistants, it executes actions directly within apps through UiPath workflows instead of offering suggestions. It features accessibility-focused voice guidance, dynamic intent extraction, fast workflow execution, and support for multiple service providers. Its modular, extensible design enables personalized automation and works across apps without APIs. This hybrid AI-RPA approach offers greater flexibility, depth, and reliability than conventional assistants.

9. SYSTEM EVALUATION/TESTING

Testing focused on functional correctness, usability, and performance reliability across different task categories.

Test ID	Scenario	Result
TC01	Speech-to-text input accuracy	95%
TC02	Intent detection via Gemini	92%
TC03	Workflow execution via UiPath	90%
TC04	API communication with MongoDB	Successful
TC05	command processing	Successful

Testing Outcomes

The system consistently executed automation tasks for ordering, messaging, and calling. MongoDB ensured stable storage and retrieval of task data. Minor issues occurred during UI layout changes and in noisy environments during speech input.

10. SYSTEM PERFORMANCE

Metric	Result
Task Success Rate	93%
Average Response Time	4.1 s Error
Rate	7%
Voice Recognition Accuracy	92%
MongoDB Retrieval Time	< 0.2 s

App-wise Success Rate

EatSure – 90%, Alarm – 85% , YouTube Music -89%, Blinkit – 91%, WhatsApp Message – 96%, WhatsApp Call – 93%.

Observation

Gemini's language understanding improved accuracy by ~14% compared to rule-based NLP. UiPath workflows ensured consistent execution.

11. SCALABILITY & MARKET POTENTIAL

The system has strong market potential across consumer, enterprise, and accessibility sectors. It can be adopted by students, professionals, visually impaired individuals, and organizations seeking automation. Its architecture supports adding more workflows, apps, and automation modules, making it scalable for education, healthcare, customer service, and corporate operations. With the growing demand for AI-powered productivity tools, the solution can expand into mobile automation, enterprise SaaS, and smart-assistant markets serving millions of digital users.

12. ECONOMIC SUSTAINABILITY

The solution can be commercialized as a subscription-based assistant, enterprise automation tool, or customizable automation platform. It reduces manual effort, saving time and operational costs for businesses. As new workflows are added, revenue can grow through enterprise licensing, workflow bundles, and integrations. Its low-cost AI processing and scalable architecture support long-term economic sustainability, driven by rising demand for digital automation across industries.

13. ENVIRONMENTAL SUSTAINABILITY

The system promotes digital efficiency by reducing repetitive device interactions, lowering screen time, and minimizing manual operations. Automated workflows help organizations cut down on paperwork, energy use, and unnecessary human effort. By enabling remote task execution and digital transactions, it indirectly reduces travel, printing, and resource waste. Its accessible, digital-first design aligns with sustainable tech practices, supporting eco-friendly, low-resource computing.

14. CONCLUSION

The project showcases the development of automated agent capable of understanding user commands and automating tasks across various apps using voice or text input. By combining natural language processing, task planning, and automation, the system offers a more personalized and action-oriented experience than conventional assistants. It provides a strong foundation for future enhancements in intelligent application interaction.

15. FUTURE SCOPE

The proposed Automated agent holds significant future potential in enhancing human-device interaction through seamless voice and text-based automation. As applications continue to expand in functionality and

complexity, integrating this automated agent with a broader range of apps—like ticket booking and shopping—can lead to a fully autonomous digital agent capable of handling personal scheduling, healthcare reminders, financial tasks, and even educational assistance. Incorporating advancements in on- device machine learning, the assistant can evolve to offer improved personalization, privacy, and responsiveness.

Future integration with multimodal interfaces—such as gesture recognition, eye tracking, and emotional context— can also make the assistant more intuitive and adaptive. Moreover, by building support for offline functionality, and accessibility features, this system can serve a wider demographic, including the elderly and differently abled, thereby making digital ecosystems more inclusive and intelligent.

REFERENCES

[1] Yunpeng Song, Rui Liu, Yu Liu, Yujun Shen, Jingren Zhou, Yizhou Sun, "VisionTasker: Mobile Task Automation via Vision-Based UI and LLM Planning," arXiv preprint arXiv:2312.11190, 2023.

[2] Yinghua Li, Bin Liu, Yulei Sui, Wenjie Zhang, "An Empirical Study of AI Techniques in Mobile Applications," arXiv preprint arXiv:2212.01635, 2022.

[3] Yanchu Guan, Xiaonan Li, Mingyu Li, Bowen Liu, Zeyu Xu, Xinyu Zhang, Yuan Zhang, "Intelligent Virtual Assistants with LLM-Based Process Automation," arXiv preprint arXiv:2312.06677, 2023.

[4] Guendalina Caldarini, Michele J. McGonagle, Fabio Gobbo, "A Literature Survey of Recent Advances in Chatbots," arXiv preprint arXiv:2201.06657, 2022.