

# **AI Integrated Chatbot-system to enhance government schemes**

## **A PROJECT REPORT**

*Submitted by*

**BENO FELIX V.P(961820104021)**

**AJISH S.K(961820104007)**

**BIJU N.P(961820104023)**

**FEVIN RAJ .S(961820104036)**

*in partial fulfillment for the award of the degree*

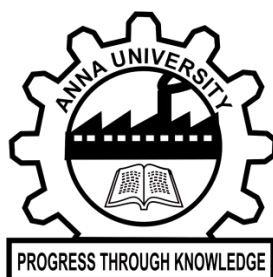
*of*

**BACHELOR OF ENGINEERING**

*in*

**COMPUTER SCIENCE AND ENGINEERING**

**PONJESLY COLLEGE OF ENGINEERING, NAGERCOIL**



**ANNA UNIVERSITY: CHENNAI 600 025**

**MAY 2024**

# **ANNA UNIVERSITY: CHENNAI 600 025**

## **BONAFIDE CERTIFICATE**

Certified that this project report “**AI Integrated Chatbot-system to enhance government schemes**” is the bonafide work of “**BENO FELIX V.P (961820104021), AJISH S.K (961820104007), BIJU N.P (961820104023), FEVIN RAJ .S (961820104036)**” who carried out the project work under my supervision.

### **SIGNATURE**

**Dr. MANJU C. THAYAMMAL**  
**M.E., Ph.D**

### **HEAD OF THE DEPARTMENT**

Professor,  
Department of Computer Science and  
Engineering,  
Ponjesly College of Engineering,  
Nagercoil-629 003.

### **SIGNATURE**

**Mrs. T. ADLIN AKHILA KENZ**  
**M.Tech.,**

### **SUPERVISOR**

Assistant Professor,  
Department of Computer Science  
and Engineering,  
Ponjesly College of Engineering,  
Nagercoil-629 003.

Submitted for the B.E Project viva-voce held on ..... at  
**PONJESLY COLLEGE OF ENGINEERING.**

**Internal Examiner**

**External Examiner**

## **ABSTRACT**

This project introduces an innovative web application with an integrated chatbot system, aiming to revolutionize accessibility to government schemes. This user-friendly platform empowers citizens by providing seamless access to essential scheme information, covering eligibility criteria and application processes. Users can interact with the chatbot for inquiries or initiate direct calls for personalized assistance from knowledgeable administrators. Additionally, users can input their village names to access contact information for local information centers, ensuring tailored support at the grassroots level. Moreover, citizens are encouraged to actively contribute to the platform by submitting information about government schemes, which undergoes thorough review and verification by administrators to ensure accuracy and reliability. This collaborative approach ensures that users have access to trustworthy information, empowering them to make informed decisions about engaging with government programs effectively.

## ACKNOWLEDGEMENT

First of all we thank **GOD** almighty for his grace enabling us to complete this work on time. We would like to extend our deep sense gratitude to our chairman **Thiru. PON ROBERT SINGH, M.A.**, for his encouragement in accomplishing this project work.

We express our deep sense gratitude and thanks to our beloved **Principal Dr. G. NATARAJAN, M.E., Ph.D.**, who is the guiding light of this outstanding institution for having laid tracks that lead to a bright future. We are very much thankful to our director **Prof. S. ARULSON DANIEL, M.Sc., M.Phil.** For his encouragement and construction ideas for my project.

We express our kind gratitude and thanks to our Head of the department **Dr. MANJU C. THAYAMMAL, M.E., Ph.D.**, who has given the source of inspiration throughout our project. We express our kind gratitude and thanks to our internal guide **Mrs. T. ADLIN AKHILA KENZ, M.Tech.**, who has given the sound guidance throughout my project.

We hardly find words to express our gratitude for the help and warm encouragement that we have received from our **parents** because, without their sacrificial help we could not have dream of completing our project successfully.

BENO FELIX V.P

AJISH S.K

BIJU N.P

FEVIN RAJ .S

## **TABLE OF CONTENTS**

<b>CHAPTER NO.</b>	<b>TITLE</b>	<b>PG NO.</b>
1.	<b>INTRODUCTION</b>	
	1.1 INTRODUCTION	1
	1.2 OBJECTIVE	1
	1.3 MOTIVATION	2
2.	<b>LITERATURE SURVEY</b>	4
3.	<b>SYSTEM ANALYSIS</b>	
	3.1 EXISTING SYSTEM	9
	3.2 PROPOSED SYSTEM	10
	3.3 METHODOLOGY	10
	3.4 SYSTEM REQUIREMENT SPECIFICATIONS	11
	3.4.1 SOFTWARE REQUIREMENT	11
	3.4.2 HARDWARE REQUIREMENT	12
4.	<b>SYSTEM DESIGN</b>	
	4.1 SYSTEM ARCHITECTURE	13
	4.2 UML DIAGRAMS	15

## **TABLE OF CONTENTS**

<b>CHAPTER NO.</b>	<b>TITLE</b>	<b>PG NO.</b>
5.	<b>TESTING AND IMPLEMENTATION</b>	
	5.1 LIST OF MODULES	21
	5.1.1 MODULE 1	21
	5.1.2 MODULE 2	23
	5.1.3 MODULE 3	24
	5.1.4 MODULE 4	25
6.	<b>CONCLUSTION AND FUTURE ENHANCEMENT</b>	
	6.1 CONCLUSION	28
	6.2 FUTURE ENHANCEMENT	28
	<b>APPENDIX I-SAMPLE CODE</b>	29
	<b>APPENDIX II-SCREENSHOTS</b>	38
	<b>REFERENCES</b>	41

## **LIST OF TABLE**

<b>TABLE NO.</b>	<b>DESCRIPTION</b>	<b>PAGE NO.</b>
3.4.1	SOFTWARE REQUIREMENT TABLE	11
3.4.2	HARDWARE REQUIREMENT TABLE	12

## **LIST OF FIGURES**

<b>FIGURE NO.</b>	<b>DESCRIPTION</b>	<b>PAGE NO.</b>
4.1	SYSTEM ARCHITECTURE	13
4.2	UML DIAGRAM	15
5.1.1	HOME PAGE	21
5.1.2	CHATBOT	23
5.1.3	ADD SCHEMES	24
5.1.4	ADMIN PAGE PROCESSES	25

## **LIST OF ABBREVIATIONS**

**AI** – Artificial Intelligence

**NLP**- Natural Language Processing

**IVR** - Interactive Voice Response

**NGO** - Non-Governmental Organizations

**UML** - Unified Modeling Language



# **CHAPTER I**

## **INTRODUCTION**

### **1.1 INTRODUCTION**

Accessing information about government schemes in the digital era remains a challenge for citizens. Our project introduces an innovative web application with an integrated AI chatbot system to address this issue. This system enhances accessibility to scheme-related information, empowering citizens with the knowledge they need for effective engagement with government assistance.

At the core of our solution is a sophisticated AI chatbot that provides seamless access to relevant information through natural language interactions. Users can easily inquire about various schemes, including eligibility criteria and application processes, in an intuitive and user-friendly manner. Additionally, our platform includes a direct call feature, connecting users with knowledgeable customer care representatives for personalized assistance.

Emphasizing local accessibility, users can input their village names to retrieve contact information for nearby information centers, ensuring assistance is readily available at the community level. Through these efforts, our project seeks to revolutionize citizen engagement with government schemes, fostering transparency, empowerment, and ultimately, improving the overall quality of life for communities.

### **1.2 OBJECTIVE**

The primary goal of our project is to develop a comprehensive and user-friendly platform that enhances accessibility to information about government schemes.

We aim to empower citizens, bridge the gap between them and the government, and promote transparency and engagement.

Key objectives include creating a responsive web application as a centralized hub for scheme details, integrating a conversational AI chatbot for understanding natural language queries, and implementing a direct call functionality for personalized assistance. Facilitating local accessibility by enabling users to retrieve nearby information center contacts based on village names is also a priority.

Additionally, we seek to empower citizens through a secure platform for contributing scheme information, with a robust review process to ensure accuracy. Developing an efficient database system to store and manage scheme data, user contributions, and other relevant information is crucial.

Ultimately, our project aims to foster transparency, informed decision-making, and active participation in government schemes by providing an easily accessible platform. By achieving these objectives, we strive to revolutionize how citizens access scheme information, contributing to an improved quality of life and a more engaged society.

### **1.3 MOTIVATION**

Our project is driven by the belief in ensuring access to information about government schemes for all citizens, bridging the digital divide. Despite numerous initiatives, many remain unaware or unable to comprehend the complexities of these programs.

We are motivated to leverage modern technologies like AI and user-friendly interfaces to create a seamless platform transcending traditional barriers, making scheme information readily available to diverse users.

Additionally, we aim to foster transparency and accountability by enabling citizen contributions and rigorous review, promoting accuracy, ownership, and government-citizen engagement.

Recognizing the importance of localized support, our motivation includes providing a platform that resonates with local communities by allowing users to retrieve nearby information center contacts based on their village names.

Ultimately, our project is driven by the desire to revolutionize how citizens access and engage with government schemes, fostering empowerment, transparency, informed decision-making, and an improved quality of life for society.

## **CHAPTER II**

### **LITERATURE REVIEW**

**Androutsopoulou et al. (2019)** investigated the potential of AI-guided chatbots in transforming the communication between citizens and government. Their study highlighted the ability of chatbots to provide personalized and interactive access to information, thereby improving citizen engagement and service delivery. They also explored the challenges and limitations associated with chatbot implementation, such as ensuring data security and privacy.

**Jain et al. (2018)** conducted a study to evaluate and inform the design of chatbots for various applications, including government services. Their research emphasized the importance of user-centered design principles and natural language processing techniques to create intuitive and effective chatbot interactions. They also highlighted the need for continuous improvement and adaptation based on user feedback and evolving requirements.

**Tripathi and Navlakha (2021)** presented a case study of India's open government data platform, which leverages crowdsourcing to enhance the availability and quality of government data. Their findings demonstrated the potential of citizen engagement and contribution in improving the accuracy and relevance of information, while also acknowledging the challenges associated with data verification and quality control.

**Thakur and Sharma (2020)** conducted a study to assess the awareness and utilization of government schemes among rural women in the Himachal Pradesh region of India. Their research revealed a significant gap between the availability of schemes and their actual utilization, highlighting the need for improved

information dissemination and awareness campaigns, particularly in rural and underserved areas.

**Bhakta and Ghosh (2020)** evaluated the impact of government schemes on rural development in a village in West Bengal, India. Their study highlighted the importance of localized support and community engagement in ensuring the effective implementation and utilization of government schemes. They also emphasized the need for robust monitoring and evaluation mechanisms to assess the impact and effectiveness of these schemes.

**Bhaduri and Nandi (2021)** conducted a study on the role of mobile applications in facilitating access to government schemes and services in rural India. Their research highlighted the potential of mobile apps to overcome barriers such as limited internet connectivity and digital literacy, enabling users to access information and services through intuitive interfaces and voice-based interactions. However, they also emphasized the need for robust user feedback mechanisms and localized content to ensure effective adoption and utilization.

**Kameswaran and Kishore (2019)** examined the use of interactive voice response (IVR) systems for disseminating information about government schemes in rural areas. Their findings suggested that IVR systems can be an effective means of reaching populations with limited digital literacy, as they leverage familiar technologies like feature phones and voice-based interactions. However, they also identified challenges related to language barriers, user interface design, and the need for regular content updates.

**Patel et al. (2020)** proposed a community-based information dissemination model for government schemes, leveraging local community centers and trained volunteers. Their approach involved empowering community members to act as

information ambassadors, disseminating scheme-related details and assisting citizens with application processes. The study highlighted the importance of community engagement, localized support, and capacity-building initiatives to ensure the sustainability and effectiveness of such models.

**Rajalakshmi and Arora (2022)** explored the potential of leveraging existing community-based organizations, such as self-help groups and women's collectives, for disseminating information about government schemes. Their research demonstrated the effectiveness of utilizing established community networks and trusted intermediaries in reaching marginalized populations and fostering awareness and utilization of government programs.

**Chaudhary et al. (2020)** explored the use of chatbots for delivering e-government services, focusing on user experience and adoption factors. Their study highlighted the importance of chatbot design, including natural language processing capabilities, personalization, and seamless integration with existing government portals.

**Lakshmi and Kumar (2019)** investigated the role of digital platforms in enhancing citizen participation and engagement in government processes. Their research emphasized the need for user-friendly interfaces, transparency, and mechanisms for two-way communication between citizens and government agencies.

**Gupta and Singh (2021)** conducted a study on the challenges faced by rural communities in accessing information about government schemes, identifying factors such as lack of awareness, digital literacy barriers, and language obstacles. Their findings underscored the importance of tailored information dissemination strategies for rural areas.

**Sharma and Mishra (2020)** examined the potential of artificial intelligence and machine learning techniques for personalized delivery of government services and scheme information. Their research explored the use of intelligent chatbots, predictive analytics, and recommendation systems to provide customized and relevant information to citizens.

**Kumar and Rao (2018)** proposed a framework for crowdsourcing and citizen-driven data collection to improve the accuracy and completeness of information related to government schemes. Their study highlighted the potential of citizen engagement in enhancing the quality and relevance of scheme data.

**Das and Bhattacharya (2021)** investigated the role of social media platforms in disseminating information about government schemes, particularly among youth and urban populations. Their research explored the opportunities and challenges associated with leveraging social media for citizen outreach and engagement.

**Singh and Sharma (2019)** conducted a comparative analysis of government portals and mobile applications for accessing scheme information, evaluating factors such as usability, content quality, and user satisfaction. Their findings provided insights into best practices for designing effective digital platforms.

**Gupta and Srivastava (2020)** explored the potential of voice-based interfaces and conversational agents for delivering government services and scheme information to populations with limited digital literacy. Their study highlighted the importance of natural language processing and multi-lingual support.

**Yadav and Jain (2022)** investigated the role of community-based organizations and non-governmental organizations (NGOs) in bridging the information gap and facilitating access to government schemes, particularly in rural and marginalized communities.

**Mishra and Pandey (2021)** conducted a study on the impact of government schemes on socioeconomic development, emphasizing the need for effective monitoring and evaluation mechanisms to assess the reach and outcomes of these programs.



## **CHAPTER III**

### **SYSTEM ANALYSIS**

#### **3.1 EXISTING SYSTEM**

The current system for providing information about government schemes suffers from fragmentation, complexity, and lack of a centralized platform. Citizens struggle with navigating multiple websites, relying on traditional methods like brochures or physical visits. Personalized support is lacking, making it difficult to understand eligibility criteria and application processes. Additionally, there are no mechanisms for citizen engagement or contribution, leading to potential information gaps. The digital divide further exacerbates accessibility issues for those with limited technology access or digital literacy. These drawbacks, including lack of user-friendliness, limited support, absence of citizen engagement, and inadequate digital accessibility, necessitate an innovative solution for seamless access to scheme information.

##### **3.1.1 DISADVANTAGES**

- **Lack of centralized information:** Citizens must navigate multiple government sources, causing confusion.
- **Limited personalized support:** Interactive assistance is lacking, making it difficult to understand eligibility and processes.
- **Insufficient citizen engagement:** Lack of mechanisms for citizen input leads to information gaps.
- **Digital accessibility barriers:** The system overlooks the digital divide, excluding those with limited access or skills.

## 3.2 PROPOSED SYSTEM

The proposed system is a web application with an integrated chatbot to enhance accessibility to government scheme information. It features a user-friendly chatbot for inquiries, direct call option to customer care, local support via village name input, and a user contribution feature with admin verification before publishing. This comprehensive approach improves accessibility, promotes citizen involvement, and ensures accurate localized support.

### 3.2.1 ADVANTAGES

- **Simplified Access:** The web app offers easy access to government schemes, overcoming digital hurdles and confusion from scattered information sources.
- **Intuitive Interface:** The AI chatbot simplifies information retrieval, making it effortless for users regardless of their tech skills.
- **Localized Help:** Village-specific info centers ensure relevant assistance, bridging the gap between citizens and government schemes at the community level.
- **Enhanced Engagement:** By empowering citizens with scheme details, the project fosters active participation and transparency in government assistance programs.

## 3.3 METHODOLOGY

This innovative project aims to revolutionize how citizens access information about government schemes through the development of a user-friendly web application integrated with an AI-powered chatbot. By leveraging natural language processing and machine learning techniques, the chatbot provides an intuitive conversational interface for inquiring about scheme details, eligibility criteria, and application processes. The web platform offers a centralized hub for scheme

information, enabling users to retrieve local information center contacts based on their village names. Moreover, the system empowers citizens to contribute information about government schemes, fostering transparency and ensuring the accuracy of data through a robust review process. This comprehensive solution bridges the digital divide, promotes active citizen engagement, and ultimately enhances accessibility to government assistance programs.

### 3.4 SYSTEM REQUIREMENT SPECIFICATIONS

**AI Integrated chatbot system to Enhance government schemes** is based on Python coding with MongoDB database .As it requires an Intel or Ryzen 5000 series ahead.

#### 3.4.1 SOFTWARE REQUIREMENT

Operating System	Any Operating System
Software	Pycharm,MongoDB compass
Graphics	Any Simple Integrated graphics

**Table 2.4.1 Software Requirement Table**

### 3.4.2 HARDWARE REQUIREMENT

<b>Processor</b>	Any Dual core processor
<b>RAM</b>	4 GB RAM
<b>Display properties</b>	VGA 1920*1080 or higher-resolution screen supported by Microsoft Windows.
<b>Disk space requirement</b>	<p><b>Standard Edition:</b> typical installation 48 MB, full installation 80 MB.</p> <p><b>Professional Edition:</b> typical installation 48 MB, full installation 80 MB.</p> <p><b>Enterprise Edition:</b> typical installation 128 MB, full installation 147 MB.</p> <p><b>Additional components</b> (if required): MSDN (for documentation): 67 MB, Internet Explorer 4.x: approximately 66 MB.</p>

**Table 2.4.2 Hardware Requirement Table**

## CHAPTER IV

### SYSTEM DESIGN

#### 4.1 SYSTEM ARCHITECTURE

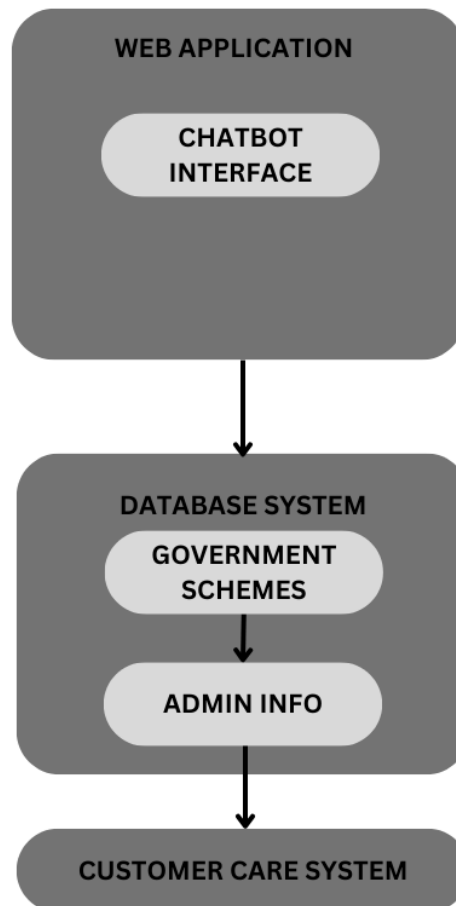


FIGURE 3.1 SYSTEM ARCHITECTURE

The system architecture of our web application with an integrated chatbot system is designed to provide a seamless and efficient experience for users seeking information about government schemes. The architecture follows a modular and scalable approach, ensuring flexibility and ease of maintenance.

At the core of the system lies the Web Application module, which serves as the primary interface for users to interact with the platform. This module is responsible for rendering the user-friendly web pages, facilitating user

registration and authentication, and enabling various functionalities such as chatbot integration, and local information retrieval.

The Chatbot Integration module is a crucial component that incorporates a conversational AI chatbot capable of understanding and responding to natural language queries from users. This module leverages advanced natural language processing techniques to interpret user inputs, retrieve relevant information from the database, and generate appropriate responses. The chatbot is designed to handle a wide range of queries related to government schemes, eligibility criteria, application processes, and other relevant details.

The Local Information Retrieval module allows users to input their village names and retrieve contact information for the nearest information center. This module leverages a comprehensive database of local information centers, providing users with localized support and ensuring community engagement.

The Database System module serves as the central repository for storing and managing all relevant data, including information about government schemes, user contributions, local information centers, and other related data. This module employs robust database management techniques to ensure efficient data storage, retrieval, and synchronization with the other modules of the system.

The system architecture also incorporates various security measures, such as user authentication, data encryption, and access control mechanisms, to protect sensitive information and ensure the privacy and security of user data.

Overall, the modular and scalable design of the system architecture enables seamless integration of various components, facilitating efficient information

retrieval, user engagement, and personalized support, while ensuring flexibility for future enhancements and expansions.

## 4.2 UML DIAGRAMS

### 1. USECASE DIAGRAM



**Fig 4.2.1: Usecase Diagram**

This image represents a use case diagram for a chatbot application system. the primary actor, "user," can engage with the system through three use cases: "chat with bot," allowing users to converse with a chatbot interface; "check with local info centers," enabling users to query local information centers for relevant data; and "contact through e-mail," facilitating communication with the system or support team via email.

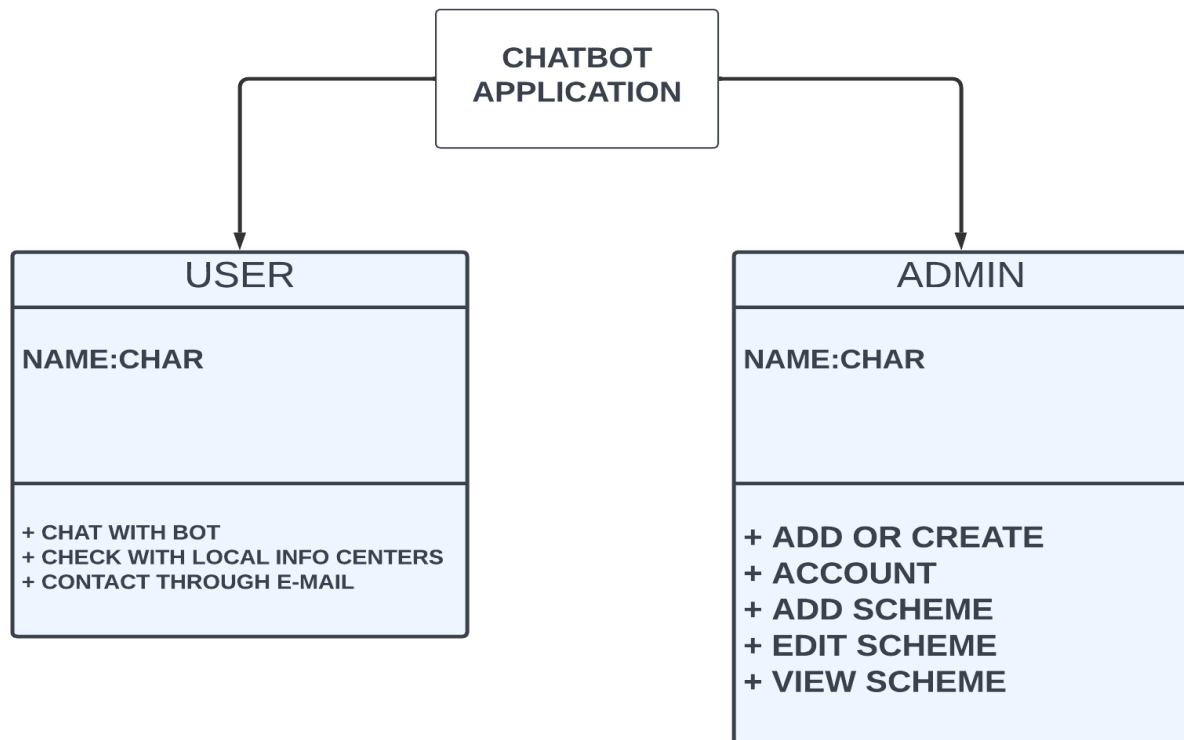
The actor "admin," representing administrators with elevated privileges, has access to several use cases. in addition to "add scheme," "edit scheme," and "view scheme" for managing system functionalities, administrators can also "chat with bot" and "add or create account." this means that administrators can interact with the chatbot interface and have the ability to create new accounts or add existing ones to the system.

The diagram also depicts the "system" entity, representing the underlying infrastructure or platform that supports the interactions between users, administrators, and the chatbot application.

Overall, this use case diagram illustrates the key actors, their respective use cases, and the relationships within the context of a chatbot application system, where regular users have limited functionality, while administrators have expanded capabilities, including chatting with the bot and account management.



## 2. CLASS DIAGRAM

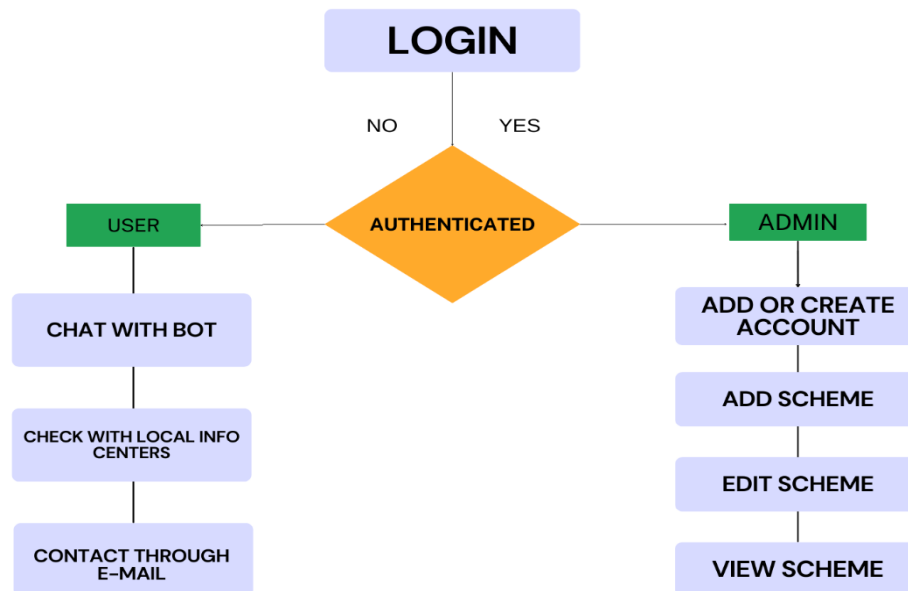


**Fig 4.2.2: Class Diagram**

The class diagram illustrates the structure and relationships of the key components in a chatbot application system. The central entity is the "chatbot application" class, which interacts with two main actor classes: "user" and "admin." The "user" class has a single attribute, "name:char," representing the user's name, and three operations: "chat with bot," "check with local info centers," and "contact through e-mail." These operations allow users to converse with the chatbot, access local information centers, and communicate via email, respectively. The "admin" class also has a "name:char" attribute for the administrator's name and four operations: "add or create account," "add scheme," "edit scheme," and "view scheme." These operations enable administrators to manage accounts, introduce new schemes or functionalities, modify existing schemes, and view available schemes within the system. The class diagram provides a structural overview of the

key entities, their attributes, and the operations they can perform, facilitating an understanding of the chatbot application's design and functionality.

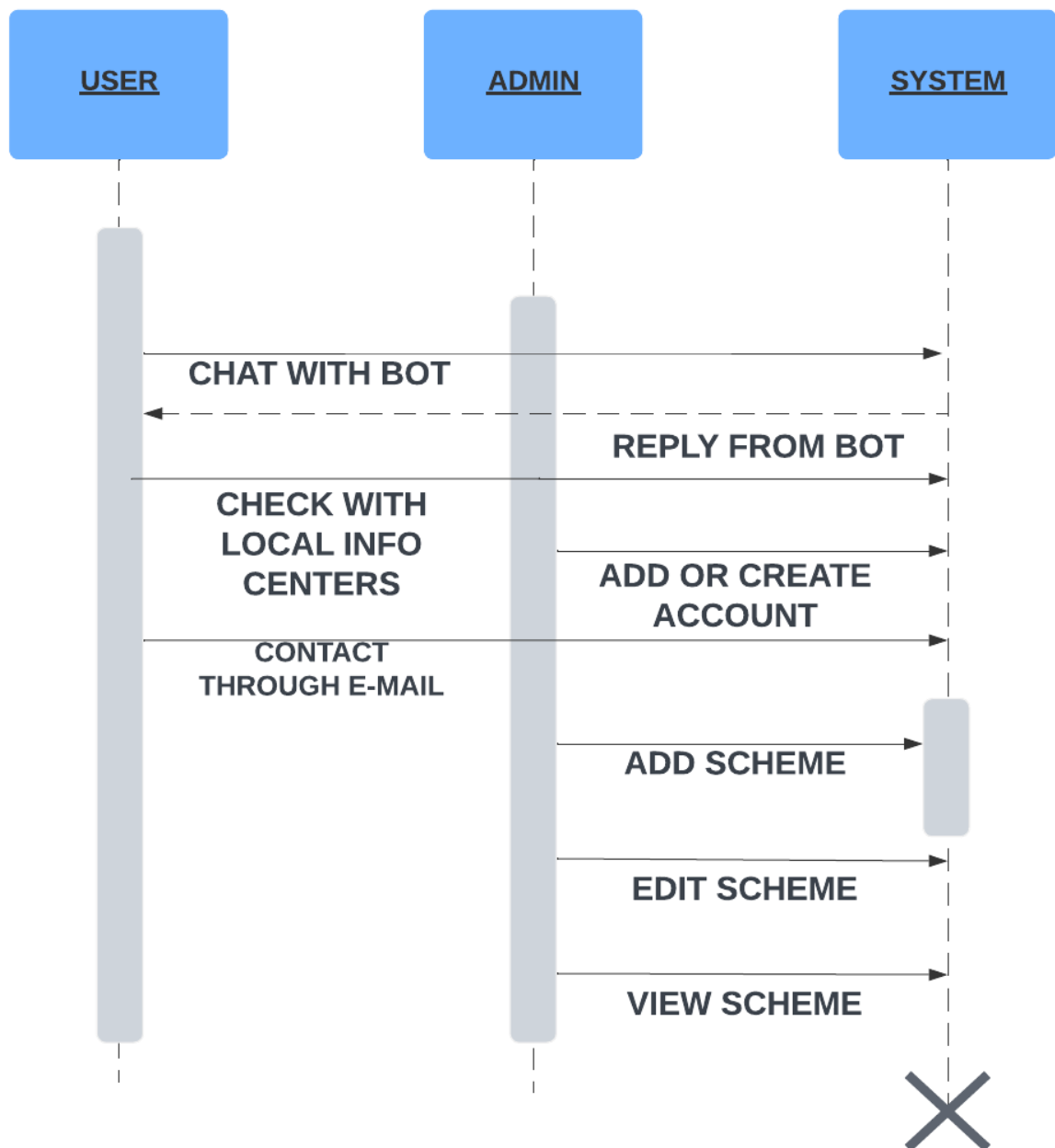
### 3. ACTIVITY DIAGRAM



**Fig 4.2.3: Activity Diagram**

This flowchart represents the user flow for a website or application. After the initial login process, it determines if the user is an authenticated admin or an unauthenticated regular user. If the user enters valid login credentials, they are considered an admin and have access to advanced features such as adding or editing schemes, creating accounts, and viewing schemes. On the other hand, if the user does not have login credentials or fails to authenticate, they are treated as a regular user with limited access. Regular users can only perform basic actions like chatting with a bot, contacting someone through email, or checking local information centers. The flowchart effectively differentiates between the privileges and functionalities available to admins versus regular users based on their authentication status upon logging in.

#### 4. SEQUENCE DIAGRAM



**Fig 4.2.4: Sequence Diagram**

The sequence diagram illustrates the interaction flow between three main components: User, Admin, and System. The User initiates a request, which is processed by the Admin component, potentially involving authentication or authorization tasks. The Admin then communicates with the System component to

execute the requested action or retrieve data. The System processes the request and sends responses back to the Admin. After any additional processing by the Admin, a final response is sent back to the User, completing the sequence of interactions. This diagram provides a high-level overview of the communication flow between these components in the context of the specific application or system being represented.

## **CHAPTER V**

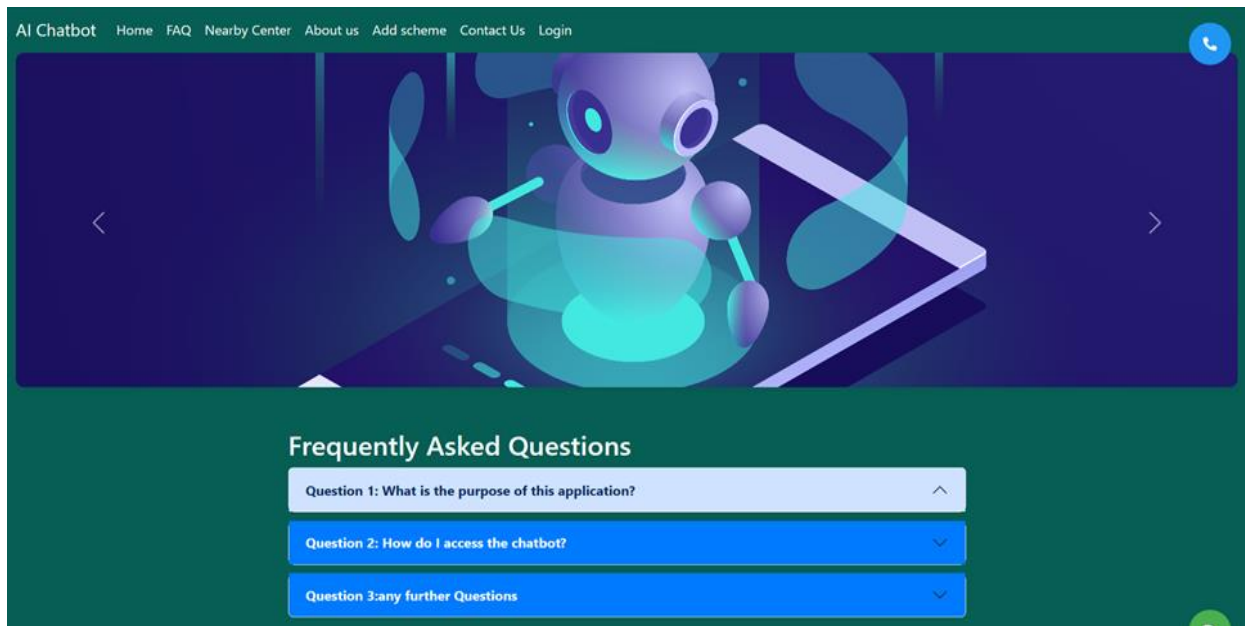
### **SYSTEM IMPLEMENTATION**

#### **5.1 LIST OF MODULE**

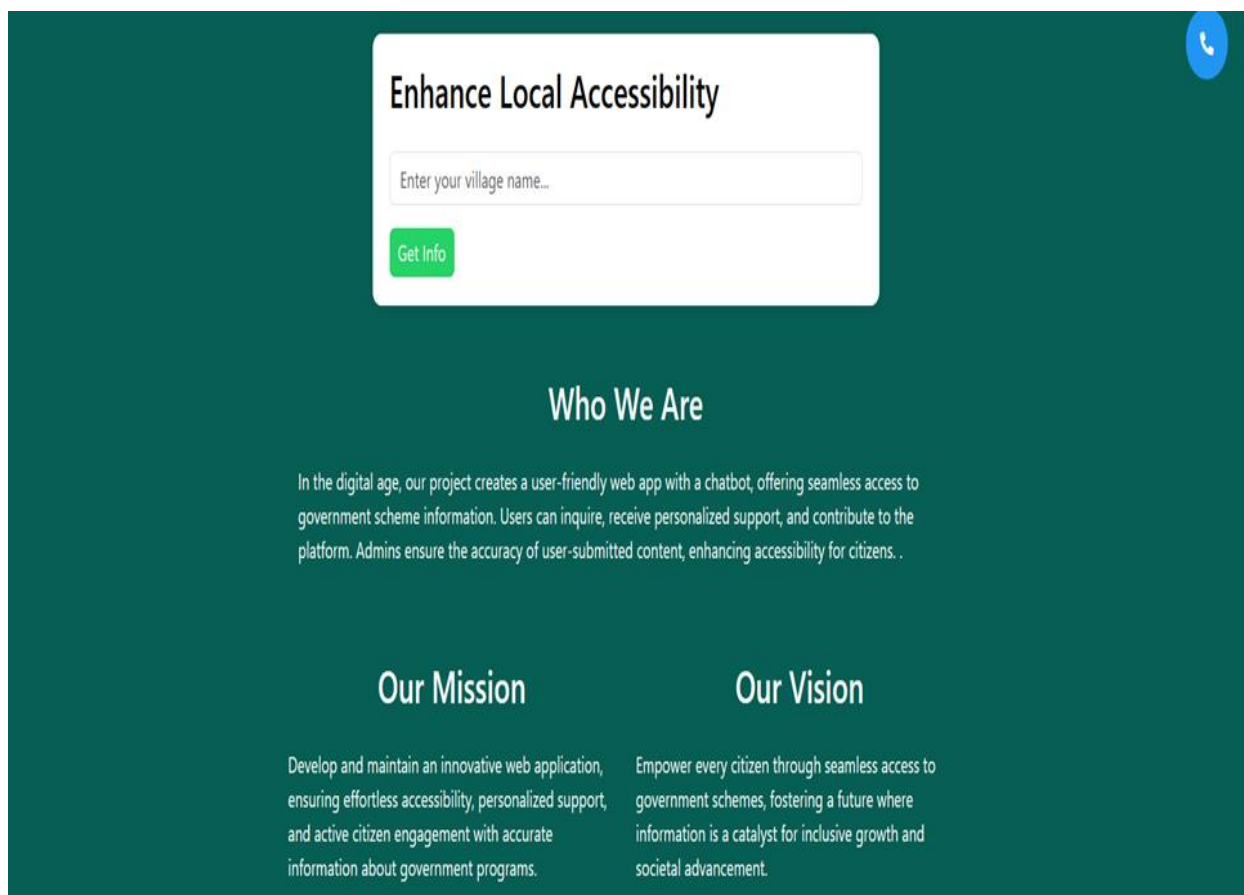
- i. HOME PAGE
- ii. CHATBOT
- iii. ADD SCHEMES
- iv. ADMIN PAGE PROCESSES

##### **5.1.1 MODULE 1: HOME PAGE**

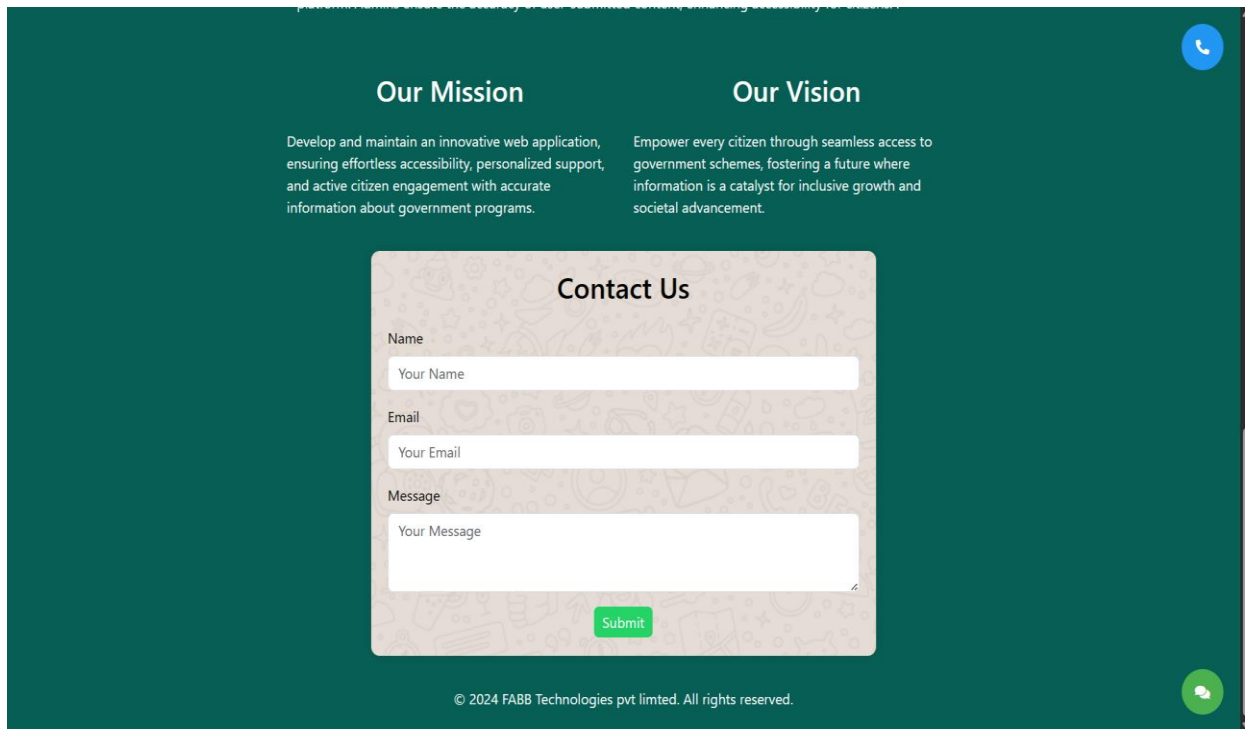
Our webpage features a responsive navbar, carousel, FAQ accordion, village info input, About Us details, contact form, floating chat and call icons, and opening modals for chatbot interaction and contact list, respectively. The design follows a WhatsApp-inspired color scheme using Bootstrap components and custom styles. JavaScript functions handle user interactions like toggling modals, sending or displaying chat messages, and simulating chatbot responses, though some functionalities like retrieving village information and making calls are not fully implemented. Overall, it provides a user-friendly interface for accessing government scheme information and interacting with a chatbot.



**Fig 4.1.1 Home Page**



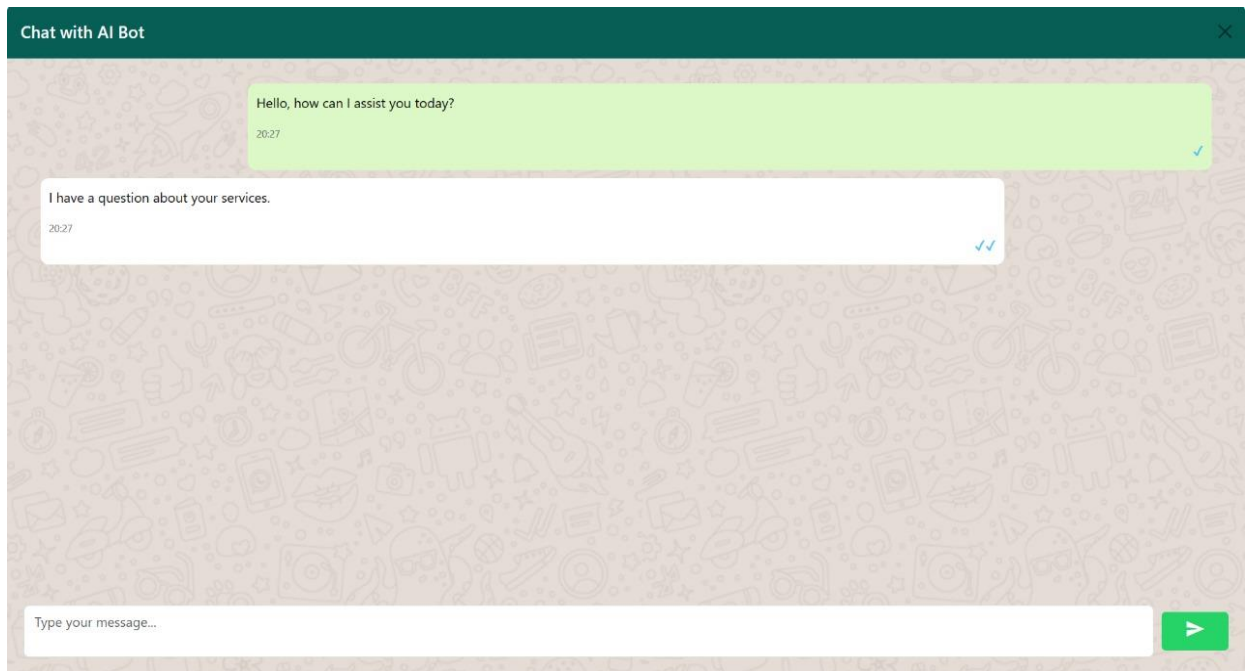
**Fig 4.1.1 Home Page**



**Fig 4.1.1 Home Page**

## **5.1.2 MODULE 2: CHATBOT**

The webpage prominently features a chatbot interface accessed through a floating chat icon that opens a modal dialog. The chat interface displays message history between the user and chatbot. Users can type queries in an input field and submit with a send button. The chatbot responds with relevant information based on user input. JavaScript functions handle input, display messages, and simulate basic responses like greetings or generic replies. The chatbot aims to provide an intuitive conversational experience, though enhancements can be made to improve NLP, integrate knowledge bases, and offer more contextual responses.

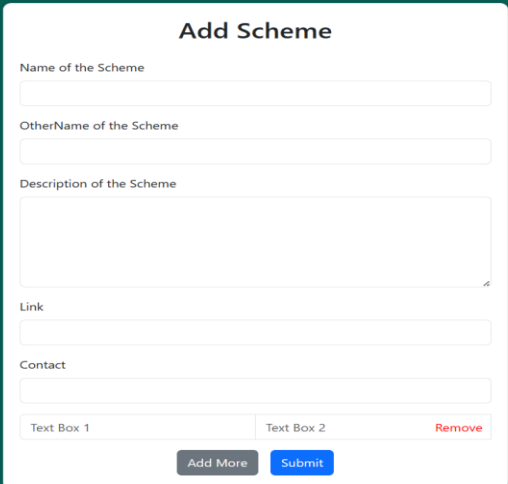


**Fig 4.1.2 Chatbot**

### **5.1.3 MODULE 3: ADD SCHEMES**

The "Add Schemes" page is a form-based interface accessible only to authorized administrators, allowing them to enter details about new government schemes. The form includes fields for the scheme name, description, link, and contact information. It also features a dynamic section for adding multiple pairs of additional text input fields to accommodate scheme-specific data points. The form has a clean design with a maximum width of 600 pixels, is centered on the page, and utilizes Bootstrap 5 for responsive styling. Administrators can add new input field pairs with an "Add More" button and remove them using the "Remove" button. The "Submit" button is provided to submit the form data for processing and storage in the database or content management system. Appropriate authentication and authorization mechanisms are required to restrict access to this page.

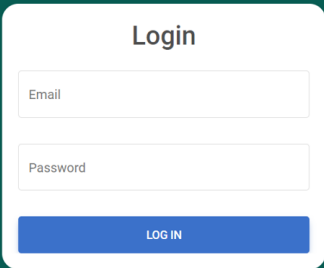


A screenshot of a web form titled "Add Scheme" centered on a dark teal background. The form is white with rounded corners and contains several input fields: "Name of the Scheme" (a single-line text box), "OtherName of the Scheme" (a single-line text box), "Description of the Scheme" (a multi-line text area), "Link" (a single-line text box), and "Contact" (a single-line text box). At the bottom of the form, there are two small text boxes labeled "Text Box 1" and "Text Box 2", with a red "Remove" link next to "Text Box 2". Below these are two buttons: a grey "Add More" button and a blue "Submit" button.

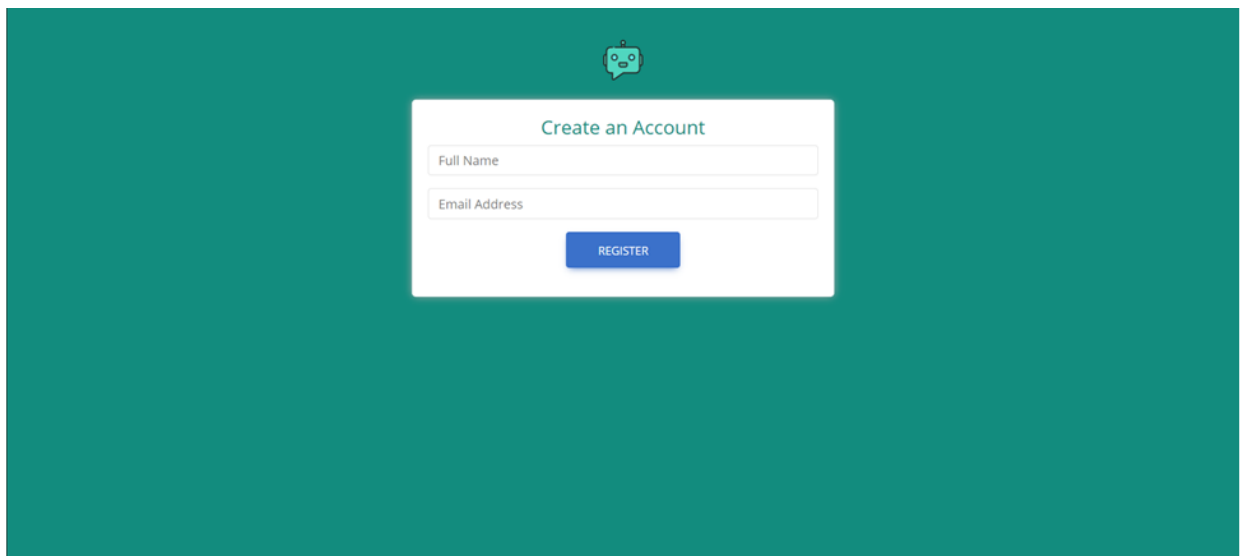
**Fig 4.1.3 Add Schemes**

#### **5.1.4 MODULE 4:ADMIN PAGE PROCESSES**

In Admin Page ,the Admin can control the overall backend of the system which includes login the account,adding new Schemes,List the schemes,register in the account.

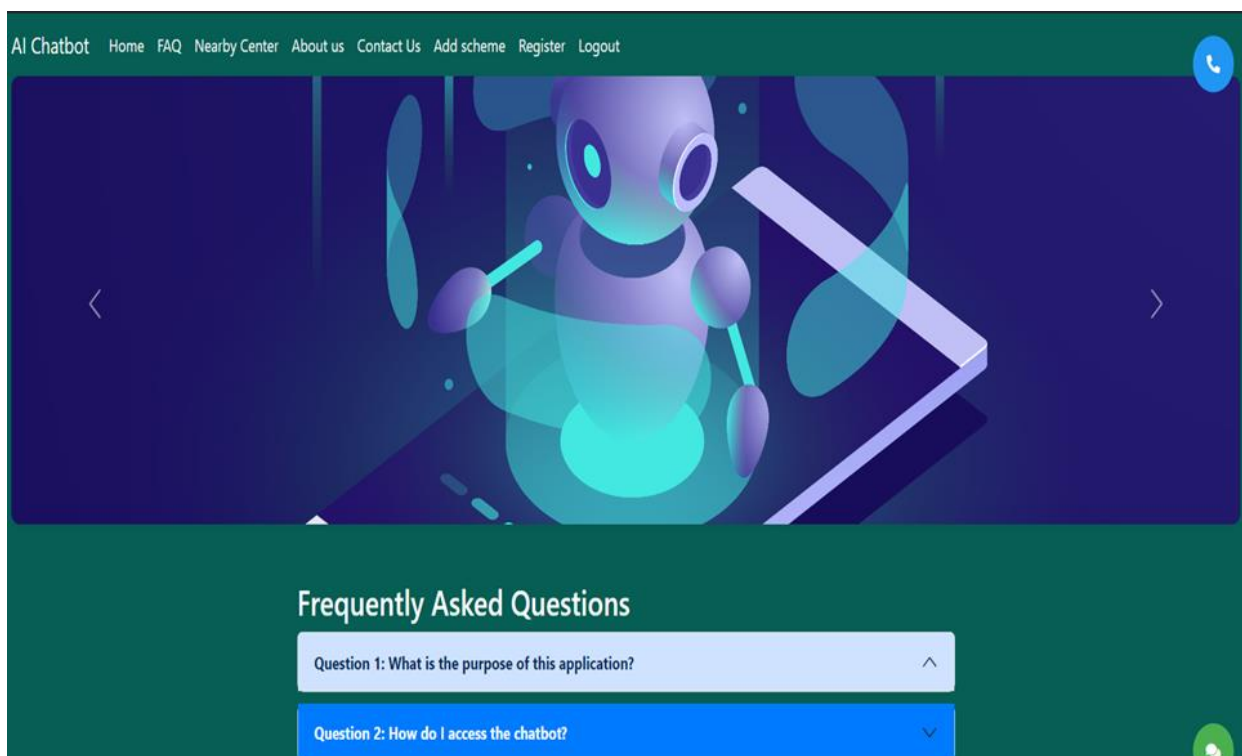
A screenshot of a web form titled "Login" centered on a dark teal background. The form is white with rounded corners and contains two input fields: "Email" (a single-line text box) and "Password" (a single-line text box). Below these fields is a blue button with the text "LOG IN" in white capital letters.

**Fig 4.1.4 Admin Page Processes (Login)**



The image shows a registration form titled "Create an Account" centered on a teal background. At the top of the form is a small robot icon. Below the title are two input fields: "Full Name" and "Email Address". At the bottom of the form is a blue button labeled "REGISTER".

**Fig 4.1.4 Admin Page Processes (Register)**



**Fig 4.1.4 Admin Page Processes**

## List of Schemes

#	Scheme Name	View	Edit	Delete
1	Atal Pension Yojana	<a href="#">view</a>	<a href="#">Edit</a>	<a href="#">Delete</a>
2	Ayushman Bharat Yojana	<a href="#">view</a>	<a href="#">Edit</a>	<a href="#">Delete</a>
3	Antyodaya Anna Yojana	<a href="#">view</a>	<a href="#">Edit</a>	<a href="#">Delete</a>
4	Credit Guarantee Scheme for Startups (CGSS)	<a href="#">view</a>	<a href="#">Edit</a>	<a href="#">Delete</a>
5	Deen Dayal Upadhyaya Grameen Kaushalya Yojana	<a href="#">view</a>	<a href="#">Edit</a>	<a href="#">Delete</a>
6	Deen Dayal Antyodaya Yojana	<a href="#">view</a>	<a href="#">Edit</a>	<a href="#">Delete</a>
7	Pradhan Mantri Awaas Yojana-Gramin	<a href="#">view</a>	<a href="#">Edit</a>	<a href="#">Delete</a>
8	Pradhan Mantri Matritva Vandana Yojana	<a href="#">view</a>	<a href="#">Edit</a>	<a href="#">Delete</a>
9	Pradhan Mantri Adarsh Gram Yojana	<a href="#">view</a>	<a href="#">Edit</a>	<a href="#">Delete</a>
10	Pradhan Mantri Kaushal Vikas Yojana	<a href="#">view</a>	<a href="#">Edit</a>	<a href="#">Delete</a>
11	Pradhan Mantri Suraksha Bima Yojana	<a href="#">view</a>	<a href="#">Edit</a>	<a href="#">Delete</a>
12	Pradhan Mantri Jeevan Jyoti Bima Yojana	<a href="#">view</a>	<a href="#">Edit</a>	<a href="#">Delete</a>
13	Pradhan Mantri Jan Dhan Yojana	<a href="#">view</a>	<a href="#">Edit</a>	<a href="#">Delete</a>
14	Pradhan Mantri Awas Yojana - Urban	<a href="#">view</a>	<a href="#">Edit</a>	<a href="#">Delete</a>
15	Pradhan Mantri Ujjwala Yojana	<a href="#">view</a>	<a href="#">Edit</a>	<a href="#">Delete</a>
16	Rashtriya Krishi Vikas Yojana	<a href="#">view</a>	<a href="#">Edit</a>	<a href="#">Delete</a>
17	Swamitva Yojana	<a href="#">view</a>	<a href="#">Edit</a>	<a href="#">Delete</a>
18	Skill India Mission	<a href="#">view</a>	<a href="#">Edit</a>	<a href="#">Delete</a>
19	Startup India Scheme	<a href="#">view</a>	<a href="#">Edit</a>	<a href="#">Delete</a>
20	UJALA - Unnat Jeevan by Affordable LEDs and Appliances for All	<a href="#">view</a>	<a href="#">Edit</a>	<a href="#">Delete</a>

Fig 4.1.4: Admin Page Processes(List the Schemes)

## **CHAPTER VI**

### **CONCLUSION AND FUTURE ENHANCEMENT**

#### **6.1 CONCLUSION**

The development of this innovative web application with an integrated AI chatbot represents a significant step towards enhancing access to information about government schemes. By leveraging cutting-edge technologies and user-centric design, our project empowers citizens, fosters transparency, and promotes active engagement with government initiatives.

Through chatbot integration, direct call functionality, local accessibility features, and citizen contribution mechanisms, our platform provides a seamless experience for users to navigate scheme details, receive personalized support, and contribute to the accuracy and reliability of information. This collaborative approach strengthens the bond between citizens and the government, fostering a more transparent and accountable system.

#### **6.2 FUTURE ENHANCEMENT**

While our web application represents a significant advancement, we recognize the ever-evolving nature of technology and the need for continuous improvement. To further augment the capabilities and reach of our platform, we envision incorporating multilingual support, voice integration, advanced analytics and personalization, and mobile application development.

Additionally, we aim to establish secure connections with government databases, ensure up-to-date information, and enhance citizen engagement through forums, discussion boards, and social media integration. By continuously evolving and incorporating these enhancements, our project solidifies its position as a pioneering solution, empowering citizens, promoting transparency, and driving positive societal change.

## APPENDIX I

### SAMPLE CODE FOR CHATBOT CONTROL

```
import random

import string

import nltk

from flask import Flask, request, jsonify, render_template, url_for, redirect, flash

from pymongo import MongoClient

from flask_login import UserMixin, login_user, login_required, logout_user,
current_user, LoginManager

from flask_bcrypt import Bcrypt

from extra_functions import encrypt, encrypt_password, hash0, hash1, hash2,
decrypt, mail

from bson import ObjectId

nltk.download('punkt')

app = Flask(__name__, template_folder="template", static_folder="static")

bcrypt = Bcrypt(app)

app.config

client = MongoClient('mongodb://localhost:27017')

database = client["Dummy"]

db = database["users"]

collection = database["sample"]

login_manager = LoginManager(app)
```

```
login_manager.login_view = 'login'
```

```
class User(UserMixin):
```

```
    def __init__(self, user_data):
```

```
        self.id = str(user_data['_id'])
```

```
        self.email = user_data['email']
```

```
        self.password = user_data['password']
```

```
@login_manager.user_loader
```

```
def load_user(user_id):
```

```
    user_data = db.find_one({'_id': ObjectId(user_id)})
```

```
    return User(user_data) if user_data else None
```

```
remove_punc_dict = dict((ord(punc), None) for punc in string.punctuation)
```

```
greet_inputs = ('hello', 'hi', 'whassup', 'how are you?', 'hai', 'Whatsup')
```

```
greet_responses = ('hi', 'Hey', 'Hey There!', 'There there!!')
```

```
def greet(sentence):
```

```
    for word in nltk.word_tokenize(sentence):
```

```
        if word.lower() in greet_inputs:
```

```
            return random.choice(greet_responses)
```

```
def get_keywords(user_response, scheme_data):
```

```
    user_keywords = [word.lower() for word in nltk.word_tokenize(user_response)]
```

```
    scheme_names = [scheme["scheme_name"].lower() for scheme in scheme_data]
```

```
    return user_keywords + scheme_names
```

```

def generate_chatbot_response(user_message):

    robo1_response = ""

    matched_schemes = []

    scheme_data = [scheme for scheme in collection.find()]

    greeting = greet(user_message)

    if greeting:

        return greeting

    if not matched_schemes:

        for scheme in scheme_data:

            scheme_keywords = scheme.get("keywords", [])

            for keyword in scheme_keywords:

                if keyword.lower() in user_message.lower():

                    matched_schemes.append(scheme)

    if len(matched_schemes) == 1:

        scheme = matched_schemes[0]

        robo1_response = f'Here are the details for the matched scheme - {scheme["scheme_name"].capitalize()}: \n'

        for key, value in scheme.items():

            if key != "_id" and key != "keywords":

                if isinstance(value, str):

                    robo1_response += f'{key.capitalize()}: {value} \n'

                elif isinstance(value, dict):

```

```

        robo1_response += f"{key.capitalize()}: \n"

    for k, v in value.items():

        robo1_response += f"    {k.capitalize()}: {v}\n"

elif len(matched_schemes) > 1:

    robo1_response = "Multiple schemes matched. Please type the number of the
scheme you want to know more about:\n"

    for i, scheme in enumerate(matched_schemes, 1):

        robo1_response += f"{i}. {scheme['scheme_name'].capitalize()}\n"

else:

    robo1_response = "Scheme details not found in the database."

return robo1_response

def generate_keywords(main_list):

    sub = [main_list[i:j] for i in range(len(main_list)) for j in range(i + 1,
len(main_list) + 1)]

    m = []

    for i in sub:

        k = "

        for j in i:

            k += j

        m.append(k)

    return m

def generate_password(length=12):

```



```

characters = string.ascii_letters + string.digits + string.punctuation

password = ''.join(random.choice(characters) for _ in range(length))

return password

@app.route("/", methods=["GET", "POST"])

def home():

    return render_template("index.html")

@app.route('/chat', methods=["POST"])

def chat():

    data = request.get_json()

@login_required

def scheme():

    if request.method == "POST":

        scheme = request.form["name"]

        other = request.form["other"]

        existing_record = collection.find_one(

            {"$or": [{"Scheme Name": scheme}, {"Other Name": other}]}

        )

        if existing_record:

            return jsonify({"error": "Scheme name or other name already exists!"}), 400

        # Get values from the form

        description = request.form["description"]

```

```

link = request.form['link']

contact = request.form['contact']

textbox1_values = request.form.getlist('textbox1[]')

textbox2_values = request.form.getlist('textbox2[]')

# Generate keywords

keyword      =      generate_keywords(list(scheme.split()))      +
generate_keywords(list(other.split()))

# Add non-empty values from textbox fields to data1

data = {box1: box2 for box1, box2 in zip(textbox1_values, textbox2_values)
if box2.strip() != ""}

data1.update(data)

data1 = {key: value for key, value in data1.items() if value.strip() != ""}

collection.insert_one(data1)

return 'Data submitted successfully!'

return render_template("scheme.html")

@app.route("/login", methods=['GET', 'POST'])

def login():

    if request.method == 'POST':

        username = encrypt(request.form['email'])

        salt = 'wqPDlMOPw5PChcOkwoHDn8OZw6I='

        user = db.find_one({"email": username})

        if user:

```

```

if encrypt(password) == user["password"]:

    user_object = User(user) # Create an instance of User class

    msg = "You had logged in our site."

    sub = "Login Detected!!"

    mail(request.form['email'], msg, sub)

    login_user(user_object) # Pass the user object to login_user

    flash('Logged in successfully.')

    return redirect(url_for('home'))

else:

    msg = "Someone is trying to log in to our site using your credentials."

    sub = "Someone is trying to log in."

    mail(request.form['email'], msg, sub)

    flash("You have entered the wrong password!")

    return render_template("login.html")

else:

    flash("There is no account found with the given email!")

    return render_template("login.html")

return render_template('login.html')

@login_required

@app.route("/signup", methods=['GET', 'POST'])

def signup():

```

```

if request.method == 'POST':

    Name = encrypt(request.form["name"])

    Email = encrypt(request.form["email"])

    user = db.find_one({"email": Email})

    salt = 'wqPDlMOPw5PChcOkwoHDn8OZw6I='

    psw = generate_password()

    print(psw)

    Password = hash2(

        hash1(hash0(encrypt(key="5b40117d08e+646606a733e=1f0c078c6b87",
clear=encrypt_password(psw + salt))))))

    if user:

        flash("Email already exists!!")

        return redirect(url_for("home"))

    else:

        msg = (f"Congratulation, Your account has been Created Successfully\n"

            f"Your account Password is : {psw}\n"

            f"Don't Share the Password with anyone.")

        subject = "Your account has been created successfully!"

        db.insert_one({

            "email": Email,

            "name": Name,

            "password": encrypt(Password)

```

```

    })

    mail(request.form['email'], msg, subject)

    flash("Your account has been created!")

    if current_user.is_authenticated:

        return redirect(url_for("home"))

    else:

        return redirect(url_for("home"))

    return render_template('signup.html')

@app.route(f'/{encrypt("logout")}', methods=['GET', 'POST'])
@login_required
def logout():

    logout_user()

    flash("You Have Been Logged Out!")

    return redirect(url_for('login'))

if __name__ == '__main__':

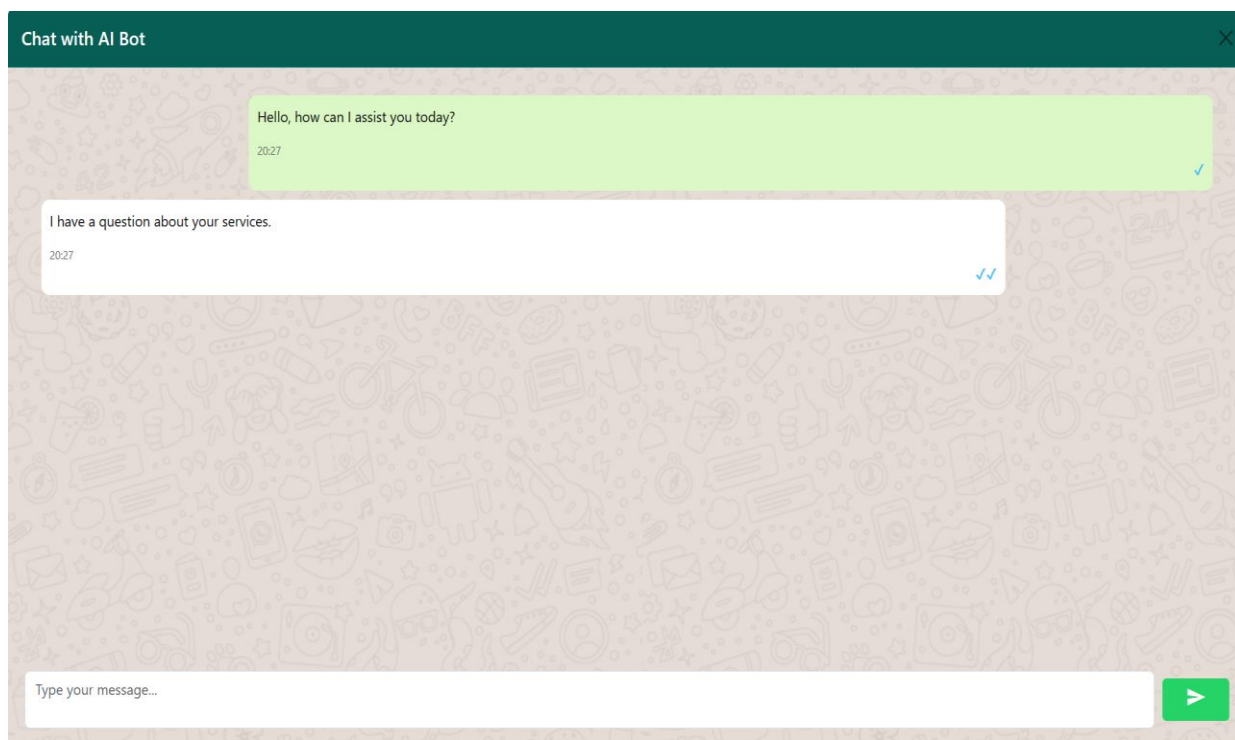
    app.run(debug=True)

```

## APENDIX 2

### SCREENSHOTS

#### CHATBOT



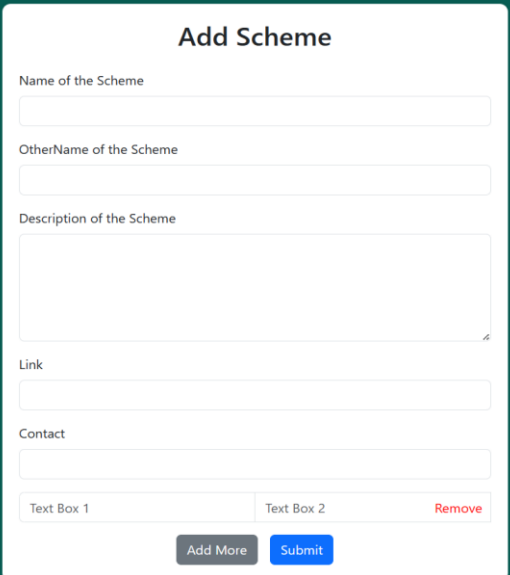
**Fig 1: Chatbot**

#### LIST SCHEMES

List of Schemes				
#	Scheme Name	View	Edit	Delete
1	Atal Pension Yojana	<a href="#">view</a>	<a href="#">Edit</a>	<a href="#">Delete</a>
2	Ayushman Bharat Yojana	<a href="#">view</a>	<a href="#">Edit</a>	<a href="#">Delete</a>
3	Antyodaya Anna Yojana	<a href="#">view</a>	<a href="#">Edit</a>	<a href="#">Delete</a>
4	Credit Guarantee Scheme for Startups (CGSS)	<a href="#">view</a>	<a href="#">Edit</a>	<a href="#">Delete</a>
5	Deen Dayal Upadhyaya Grameen Kaushalya Yojana	<a href="#">view</a>	<a href="#">Edit</a>	<a href="#">Delete</a>
6	Deen Dayal Antyodaya Yojana	<a href="#">view</a>	<a href="#">Edit</a>	<a href="#">Delete</a>
7	Pradhan Mantri Awaas Yojana-Gramin	<a href="#">view</a>	<a href="#">Edit</a>	<a href="#">Delete</a>
8	Pradhan Mantri Matritva Vandana Yojana	<a href="#">view</a>	<a href="#">Edit</a>	<a href="#">Delete</a>

**Fig 2: List of Schemes**

## ADD SCHEMES

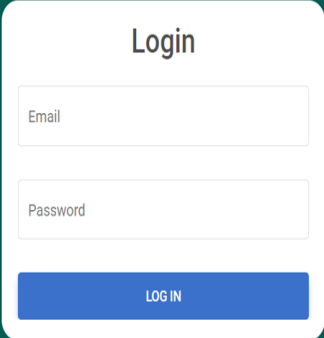


The 'Add Scheme' form is a white card with rounded corners centered on a dark teal background. It contains the following fields and controls:

- Name of the Scheme:** A single-line text input field.
- OtherName of the Scheme:** A single-line text input field.
- Description of the Scheme:** A multi-line text area.
- Link:** A single-line text input field.
- Contact:** A single-line text input field.
- Footer:** Two text boxes labeled 'Text Box 1' and 'Text Box 2' are positioned side-by-side. 'Text Box 2' has a red 'Remove' link to its right. Below these are two buttons: a grey 'Add More' button and a blue 'Submit' button.

**Fig 3: Add Schemes**

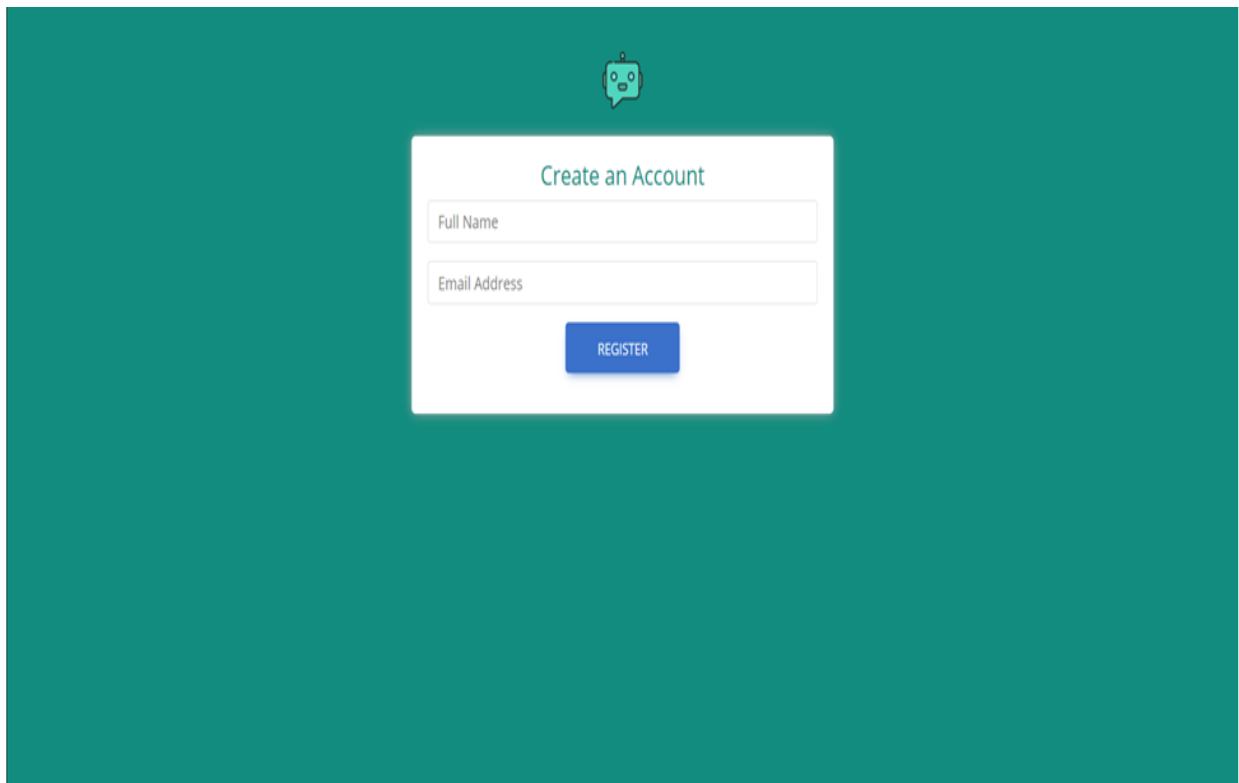
## ADMIN PROCESSES



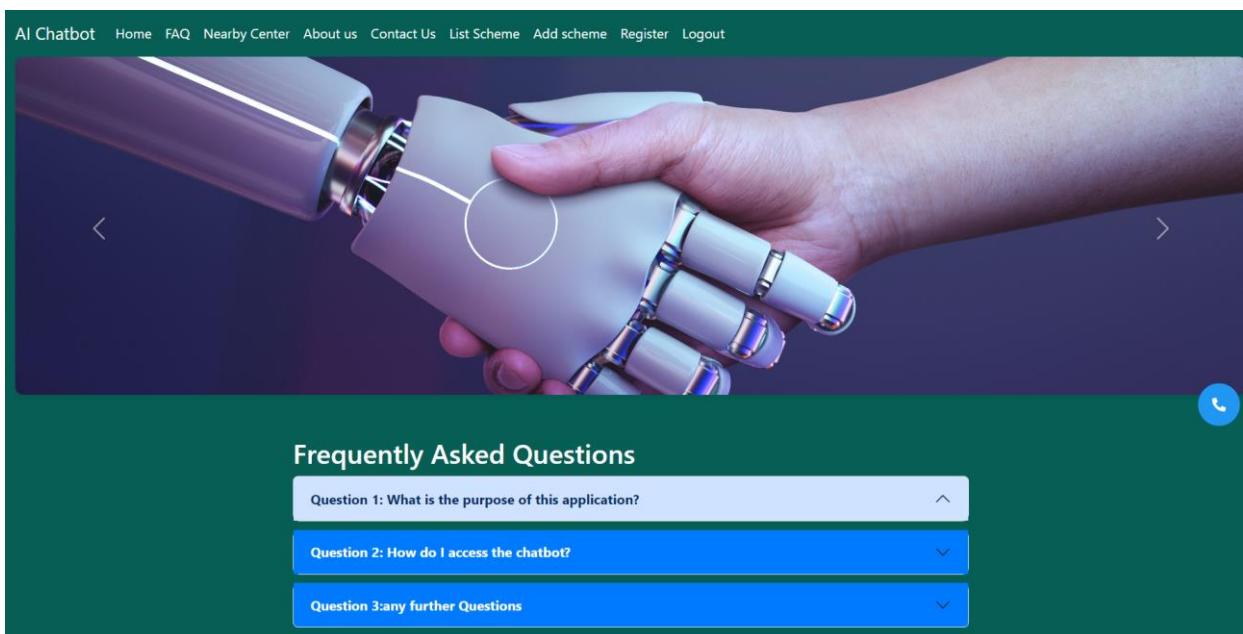
The 'Login' form is a white card with rounded corners centered on a dark teal background. It contains the following fields and controls:

- Email:** A single-line text input field.
- Password:** A single-line text input field.
- LOG IN:** A blue button with white text.

**Fig 4.1: Login**



**Fig 4.2: Register**



**Fig 4.3: Admin Interface**



## REFERENCES

- [1] Androutsopoulou, A. et al. (2019). Transforming citizen-government communication through AI chatbots. *Gov. Info. Q*, 36(2), 358-367.
- [2] Bhakta, P., & Ghosh, S. (2020). Evaluating government schemes' impact on rural development. *J. Rural Dev*, 39(2), 269-288.
- [3] Jain, M. et al. (2018). Evaluating and designing chatbots. *Proc. DIS '18*, 895-906.
- [4] Thakur, M., & Sharma, P. (2020). Awareness of government schemes among rural women. *Indian J. Ext. Educ*, 56(2), 82-86.
- [5] Tripathi, P., & Navlakha, S. (2021). Crowdsourcing government data: India's open data platform. *Proc. ICEGOV 2020*, 85-94.