Crop-Chat : Pioneering Agricultural Paradigms

****

Submitted By:

Anantha Krishnan S: AM.EN.U4CSE20208 Hrishika Dayan: AM.EN.U4CSE 20234 Keerthana A : AM.EN.U4CSE20241

Gopika Menon: AM.EN.U4CSE20245

**AMRITA SCHOOL OF COMPUTING**

**(Estd. U/S 3 of the UGC Act 1956)**

**AMRITAPURI CAMPUS KOLLAM -690525**

**JUNE 2024**

**Abstract**

India is the second-largest producer of agricultural output in the world. Despite this, many farmers still face major challenges when it comes to getting timely information about in farming practices, weather conditions, and market prices. In today’s digital world, there are many tools available, but most of them require reading and typing skills, which not all farmers are comfortable with. Our project, **CropChat**, aims to bridge this gap by providing a voice-based solution using Natural Language Processing (NLP). CropChat is a smart assistant that allows farmers to ask questions using their own voice, in their preferred language. The system understands the voice input, processes it using NLP techniques, and gives back answers in a simple and natural way. The assistant supports over 95 languages, which helps make it more inclusive and user-friendly for farmers from different regions of India. Through this project, we applied various software engineering principles such as system design using UML diagrams, modular development, and iterative planning. We also integrated external data sources to provide real-time weather forecasts and market price updates. For issues like pest attacks or crop diseases, the system allows farmers to report problems, which can then be forwarded to an agricultural expert for advice. We followed a structured development approach and used testing methods to make sure the system works reliably. This project not only helped us learn about how software systems are built and tested but also gave us the opportunity to solve a meaningful real-world problem using technology. In short, CropChat is a practical and farmer-friendly tool that uses speech and NLP to make important farming information more accessible and easier to understand.

**Table of contents**

1. Communications 4
2. Planning 11
3. Roles 13
4. Deadline and Budget 14
5. Design phase 15
6. Modules 17
7. Coding 19
8. Testing 24
9. Conclusion 25

## CHAPTER 1 COMMUNICATION

1. Client Information

The main users (clients) of our CropChat system are farmers who need help with farming- related information in a simple and easy way. Many farmers are not comfortable with reading or typing, so this voice-based assistant helps them get the information they need just by speaking in their local language.

We also considered a few other people and groups as part of the system’s users:

1. Farmers (Main Users)
   * Small and medium-level farmers from rural areas
   * Need help with crop suggestions, weather updates, and market prices
2. Agricultural Experts
   * Can give advice when a farmer reports a crop issue or problem
   * Help improve the quality of suggestions given by the system
3. Government and NGOs (Optional)
   * May use this tool to reach more farmers with important information
   * Can support large-scale farming advisory programs

Our goal was to make a system that is easy to use, works in many Indian languages, and helps farmers get timely help without needing to read or write. CropChat supports their day-to-day farming decisions using simple voice interaction.

### Meeting Minutes with Project Owner

Attendees

1. Anantha Krishnan S Team lead
2. Hrishika Dayan Business Analyst
3. Keerthana A Scrum Master
4. Gopika Menon Developer
5. Anjali T Project Owner

***Main meetings***

Meeting 1

Date: February 14, 2023

Agenda: Problem Understanding and Research Points Discussed:

We discussed the main problems faced by farmers, especially related to access to crop advice, market rates, and weather information. We explored how voice-based systems can help non-literate users. We also reviewed a few similar tools available and their limitations. The goal was to better understand how our project can be helpful and unique.

Meeting 2

Date: February 21, 2023

Agenda: Planning the System and Feature Listing Points Discussed:

We listed out the key features needed for CropChat, such as voice input, local language support, NLP for intent recognition, and market/weather info. We divided tasks among team members and created a rough plan for designing and implementing the system in phases.

Meeting 3

Date: February 28, 2023 Agenda: Design and Diagrams Points Discussed:

We discussed how the system should work internally and created use case diagrams, activity diagrams, class diagrams, and data flow diagrams. We also thought about how the interface should look and how a farmer will interact with the system easily

Meeting 4

Date: March 15, 2023

Agenda: Development Phase Start Points Discussed:

We started implementing different parts of the system. We worked on voice input, speech-to- text conversion, and began integrating NLP for query understanding. Team members were assigned specific development tasks.

Meeting 5

Date: April 29, 2023 Agenda: Testing the System Points Discussed:

We discussed how to test the voice input, the accuracy of NLP results, and the response given back to users. We checked for errors and confirmed the system could handle different languages and accents. We also tested the weather and market modules.

Meeting 6

Date: May 7, 2023

Agenda: Improvements Based on Feedback Points Discussed:

We took feedback from classmates and faculty on how the system works. Based on that, we made improvements in how responses are shown and fixed minor issues. We also made the voice response clearer and simplified the interface.

Meeting 7

Date: May 10, 2023

Agenda: Final Deployment and Documentation Points Discussed:

We discussed how to prepare for the final project submission. Tasks like cleaning up the code, preparing the final report, and documenting diagrams were planned. We also ensured the project was ready for demo and viva.

1. **Problem Statement**

Many Indian farmers lack access to accurate farming information due to language and literacy barriers. Existing digital tools are not user-friendly for rural, non-literate users. This leads to poor decisions based on unreliable sources. CropChat aims to solve this with a voice-based, multilingual assistant using NLP.

# CHAPTER-2 PLANNING

### Deadline

The deadlines for each phase of the project were meticulously set by considering the workload and manpower required for each phase. The following table summarizes the scheduled deadlines for developing the Cropchat

|  |  |  |  |
| --- | --- | --- | --- |
| Phase | Description | Start Date | End Date |
| Initiation Meeting | Problem statement discussion and our approach to the solution | 14-02-2023 | 17-02-2023 |
| Discussion and Planning | Decide on the methodology and architecture that will be used for development | 21-02-2023 | 27-02-2023 |
| Related Work | Refer to similar research, tools, and limitations for farming advisory systems | 28-02-2023 | 06-03-2023 |
| Prototype Development | Develop a prototype to visualize the software | 07-03-2023 | 13-03-2023 |
| Design and User Interface Discussion | Discuss how the software should look and feel to the farmer | 14-03-2023 | 20-03-2023 |
| Software Development | Develop the final version of the software | 21-03-2023 | 27-03-2023 |
| Testing and Validation | Test and validate the software with farmers and collect feedback | 28-03-2023 | 04-04-2023 |
| Refinement and Optimization | Modify the software based on feedback to improve performance and usability | 05-04-2023 | 19-04-2023 |
| Deployment Preparation | Prepare for deployment and document the project | 20-04-2023 | 04-05-2023 |

### About The Team

* + 1. Anantha Krishnan S : Team lead
    2. Hrishika Dayan : Business Analyst
    3. Keerthana A : Scrum Master
    4. Gopika Menon: : Developer
    5. Anjali T : Project Owner

### Anantha Krishnan S (Team Lead & Develpoer):

* + - * **Role**: As the Team Lead, B. Medhovarsh is responsible for overseeing the project's development, coordinating tasks among team members, and ensuring project milestones are met. Anantha Krishnan S brings leadership and technical expertise to guide the team towards achieving project goals efficiently.

### Hrishika Dayan (Business Analyst & Developer):

* + - * **Role**: Hrishika Dayan serves as a Business Analyst, specializing in analyzing business requirements, defining project scopes, and bridging communication between stakeholders and the development team. His role focuses on aligning project objectives with business goals to ensure successful outcomes.

### Keerthana A(Scrum Master&Developer):

* + - * **Role**: Keerthana A functions as the Scrum Master, facilitating Agile methodologies within the team. Keerthana A responsibilities include organizing daily stand-ups, removing obstacles that hinder progress, and promoting a collaborative environment that fosters productivity and efficiency.

### Gopika Menon (Business Analyst& Developer):

* + - * **Role**: M. Gopika collaborates closely with stakeholders and the development team as a Business Analyst. Her role involves gathering and analyzing project requirements, conducting market research, and ensuring that project deliverables meet business needs and expectations.

### Anjali Mam (Faculty):

* + - * **Role**: Anjali Mam plays a crucial advisory role, bringing academic expertise and guidance to the team. As a faculty member, her insights contribute to the project's educational and research aspects, ensuring alignment with industry standards and best practices.

# CHAPTER-3 DESIGN

**Modeling and Design Phase**

The design phase is a crucial step in software engineering that translates the system requirements into a structured solution. It acts as a bridge between the requirement analysis and actual coding. In this phase, the entire CropChat system is planned, visualized, and broken down into logical modules to ensure that the final implementation is efficient, maintainable, and scalable.

For the CropChat project, the design phase helped us understand how different components like voice input, Natural Language Processing (NLP), and data APIs (for market and weather info) should work together. We created several diagrams, including UML and Data Flow Diagrams (DFDs), to represent the system’s structure, user interactions, and internal processes.

This structured approach made it easier for our team to divide tasks, identify challenges early, and avoid confusion during development. It also ensured that every team member had a clear understanding of the system’s behavior and interactions, leading to smoother implementation and testing

### System Architecture

The architecture of the CropChat system is designed to enable natural, voice-based interaction for farmers, helping them access agricultural information like crop guidance, weather forecasts, and market prices. The system is modular and follows a pipeline structure, with each component responsible for a specific task. The architecture ensures that the voice input is processed intelligently and converted into accurate, helpful responses in the farmer's native language.

Below is a detailed explanation of each component as shown in the architecture diagram:

1. Crop Data

This module holds the background data required for the chatbot to provide meaningful responses. It includes datasets related to crops, diseases, fertilizers, weather information, and market prices. This data is used throughout the system to provide real-time, relevant answers to user queries.

1. Speech Recognition

This component captures the farmer’s spoken input and converts it into text. It uses speech- to-text (STT) technology, which is crucial since many users may not be comfortable typing or reading.

1. NLU (Natural Language Understanding)

After the input is converted to text, the NLU module processes it to understand what the user is asking. It identifies the intent (e.g., "What crop should I grow?") and entities (e.g., "March", "wheat", "Delhi").

1. Dialog Management

This component manages the flow of conversation. It keeps track of the context and determines what should happen next—whether to ask a follow-up question, fetch data, or end the interaction.

1. NLG (Natural Language Generation)

This module prepares the final response in a human-friendly way. It converts structured data or information into natural sentences that the user can understand easily.

1. Integration

Here, various components are brought together. The system integrates responses from APIs (for weather, market rates, etc.), internal crop data, and expert knowledge if needed.

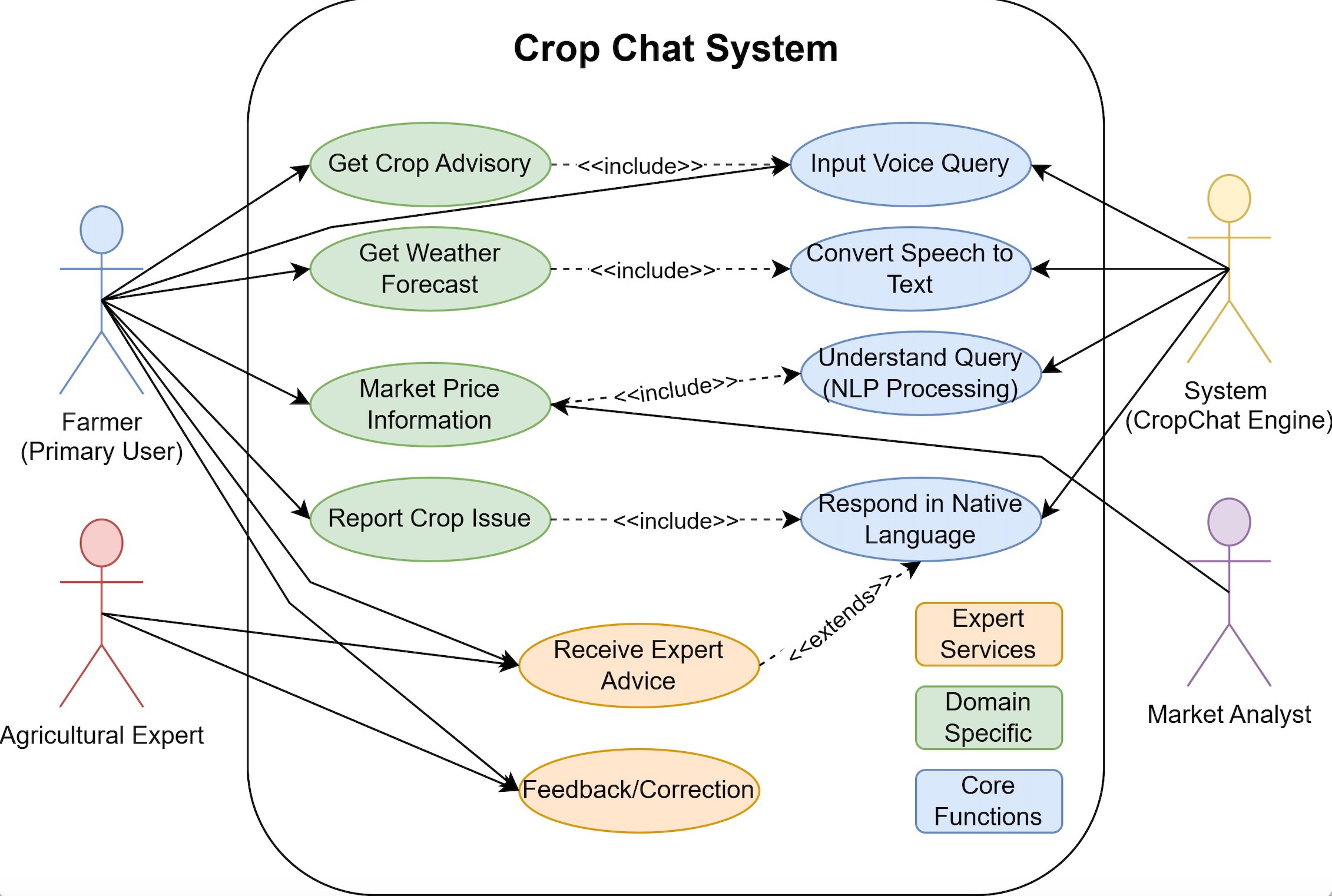
1. Testing

Once the system is integrated, testing is conducted to check for errors, bugs, and ensure proper understanding and responses. Both unit testing and user testing are performed here.

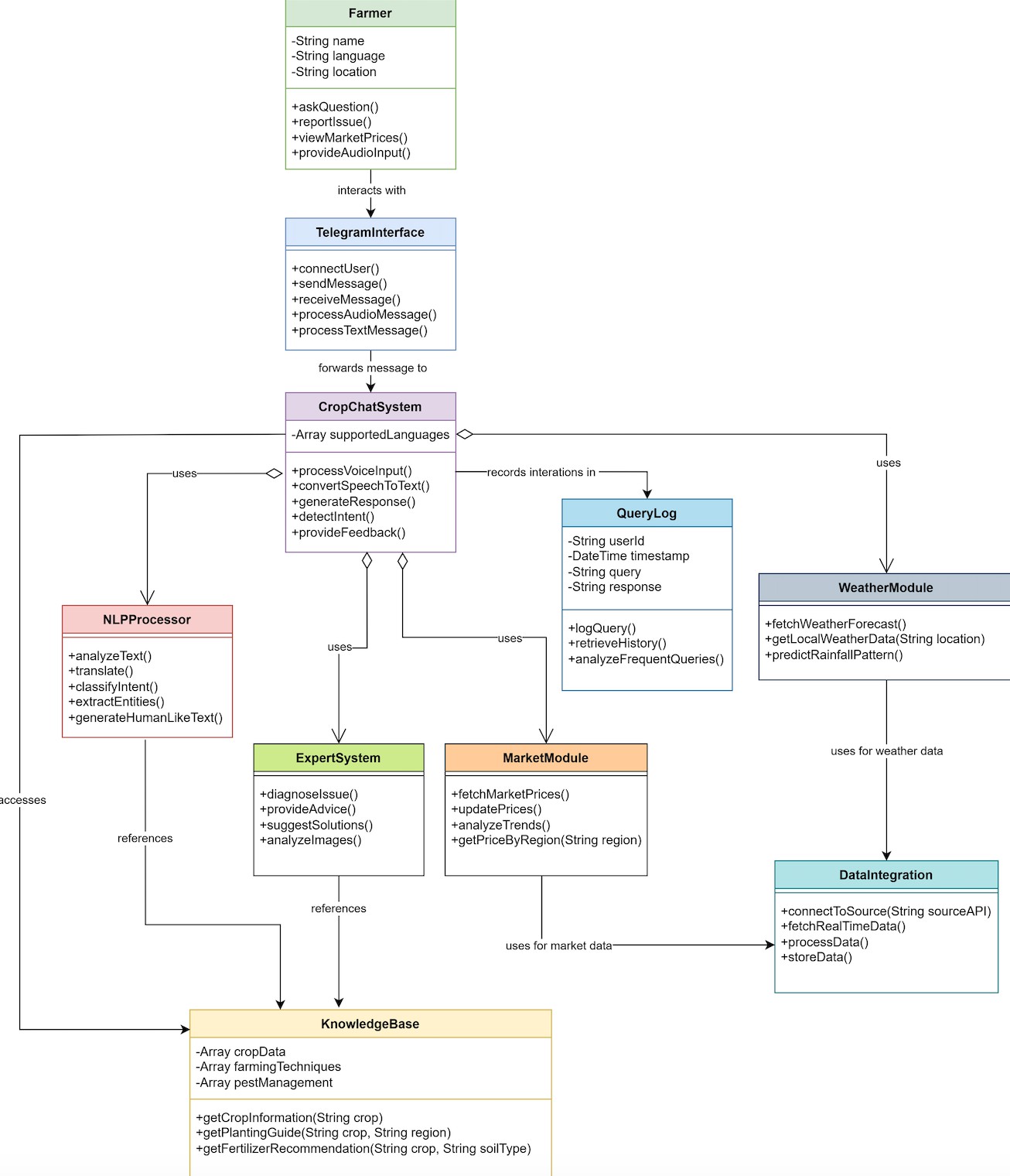
1. Deployment

The final version of CropChat is deployed for actual use. The deployed system is able to receive real queries from farmers and give them helpful voice-based responses.

**UML Use Case Diagram**

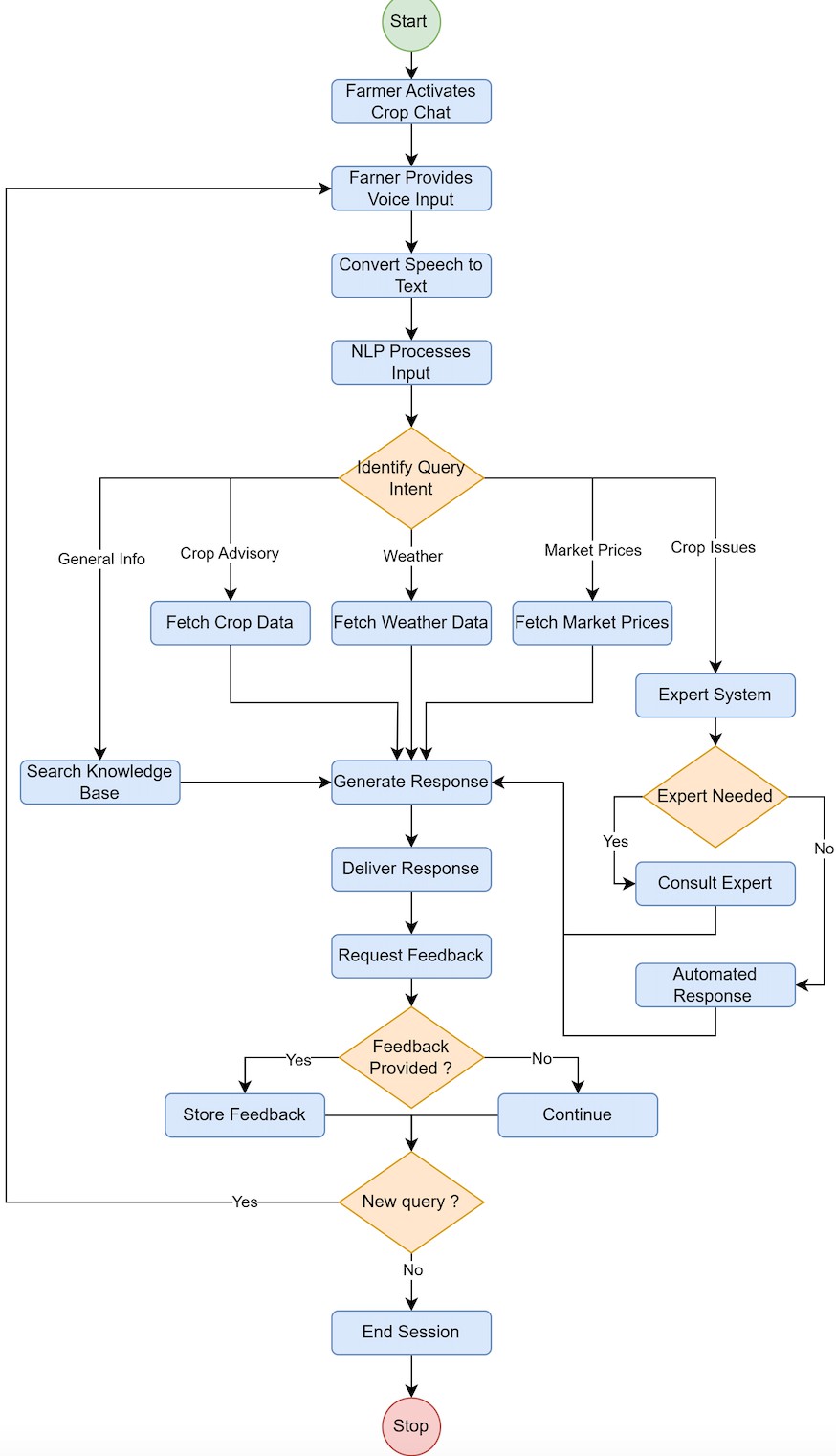
****

**Class Diagram**

****

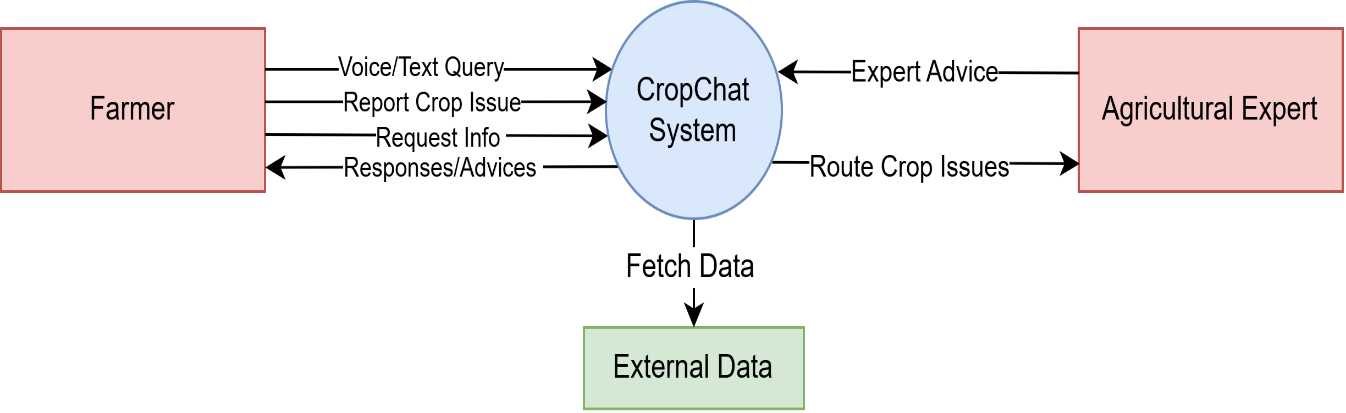
**Activity Diagram**

Activity diagram: Illustrates user interactions and administrative tasks within an AI Chatbot SaaS platform, highlighting sequential actions from registration and login to chatbot interaction and system management.



### Data Flow Diagram Level 0:

This diagram represents the Context-Level Data Flow Diagram (DFD) of the CropChat system. It shows how the main entities—Farmer, Agricultural Expert, and External Data Sources—interact with the system. Farmers can send voice or text queries, request information, or report crop issues. CropChat processes these inputs, fetches relevant data from external sources (like weather or market APIs), and forwards unresolved issues to experts who return advice, which is then delivered back to the farmer.



### Level 1:

This is a Level 1 Data Flow Diagram (DFD) of the CropChat system, which shows how the system handles a farmer’s voice input step by step.

The process starts when the farmer speaks a query. The Speech Processing module converts the voice into text. This text is passed to the NLP & Intent Detection module, which understands what the farmer is asking. If it’s a general question (like weather or price), the system fetches information from External Data sources and sends it back through Advisory & Info Services.

If the question is complex (like a crop disease or issue), the system forwards it to the Expert System Interface, which passes it on to an Agricultural Expert. The expert’s feedback is returned and shared with the farmer. All queries and responses are stored in the Query Logs Database for future reference or learning.

This flow ensures that farmers get quick and accurate help—either automatically or from a real expert when needed.

A diagram of a process

AI-generated content may be incorrect.

Level 2

This diagram shows a Level 2 Data Flow Diagram (DFD) for the CropChat system, focusing in detail on how a farmer’s voice query is processed and how the system responds with helpful agricultural information.

The process starts when a farmer speaks a question, which goes through Audio Processing and is converted into text. The Language Detection module identifies the language of the input, and then Intent Analysis figures out what the farmer is asking—such as weather, market price, or crop advice.

Next, the system uses Entity Extraction to identify specific details in the query. Depending on the query type, it routes the request to the appropriate module:

* Weather Information (for weather-related questions)
* Market Analysis (for market price queries)
* Crop Advisory (for farming-related guidance)

If the system detects a complex issue, it passes the query to Issue Routing, which forwards it to an Agricultural Expert. The expert provides advice, which is stored in the Query & Response Log DB and possibly added to the Knowledge Base DB for future use.

Finally, the system sends the weather info, market data, or crop advice back to the farmer in a clear and simple format.

This diagram explains how different modules work together to ensure farmers receive accurate, timely, and useful responses.

A diagram of a process

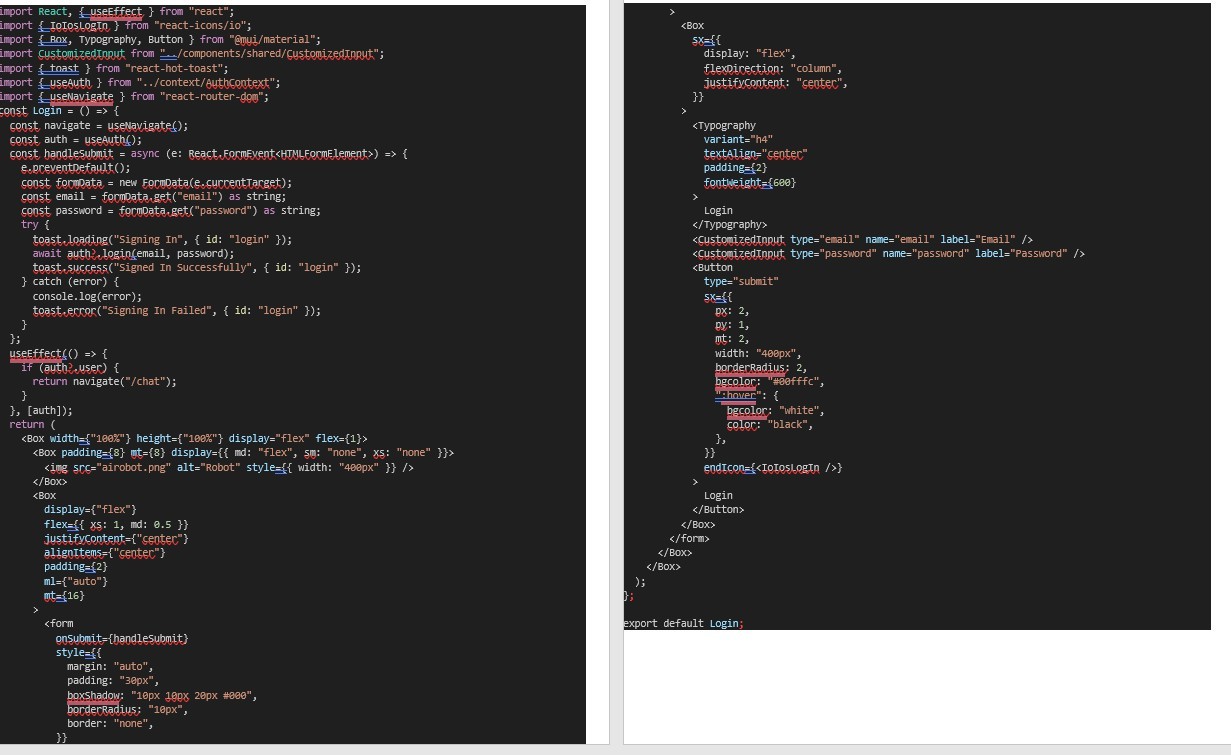
AI-generated content may be incorrect.

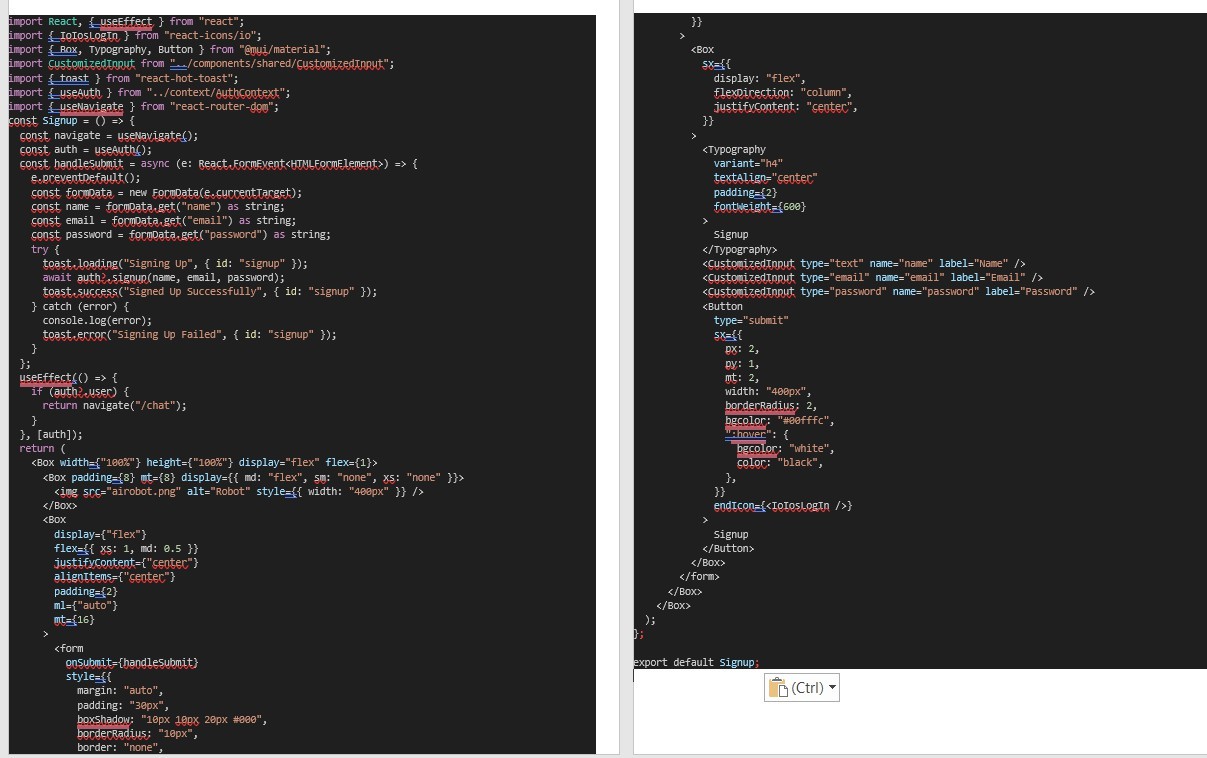
**CHAPTER-4 IMPLEMENTATION**

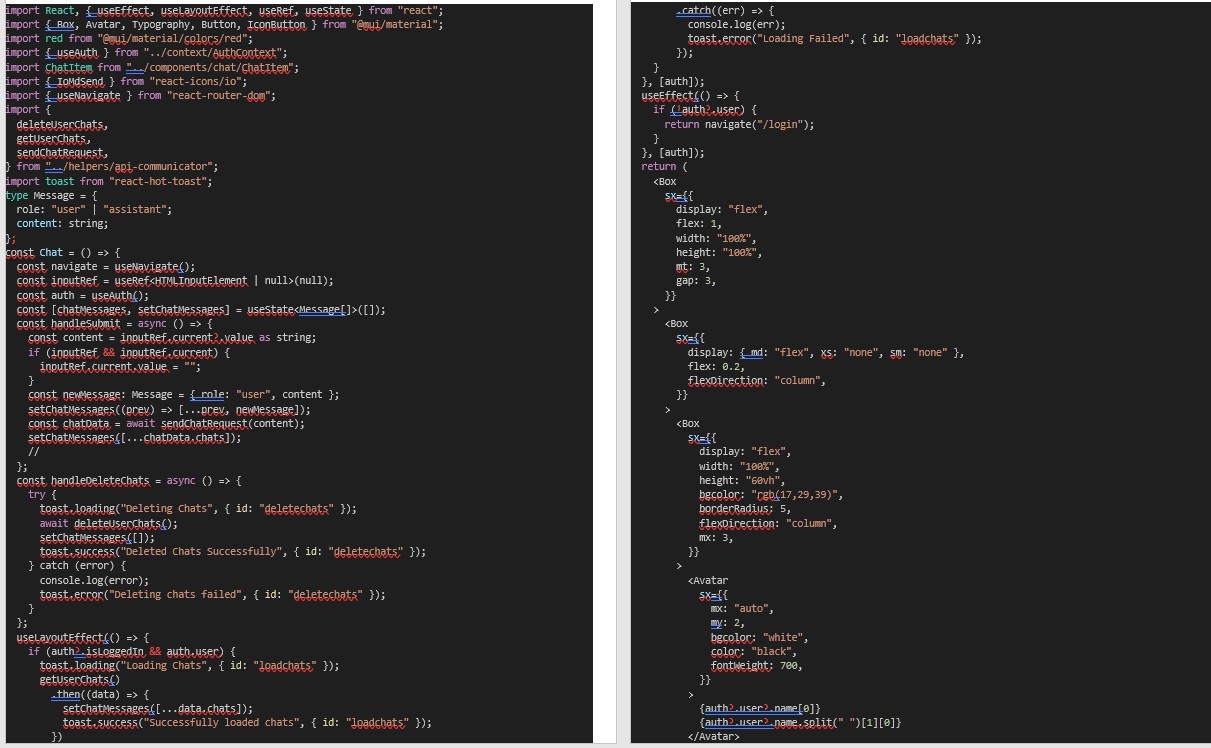
**IMPLEMENTATION**

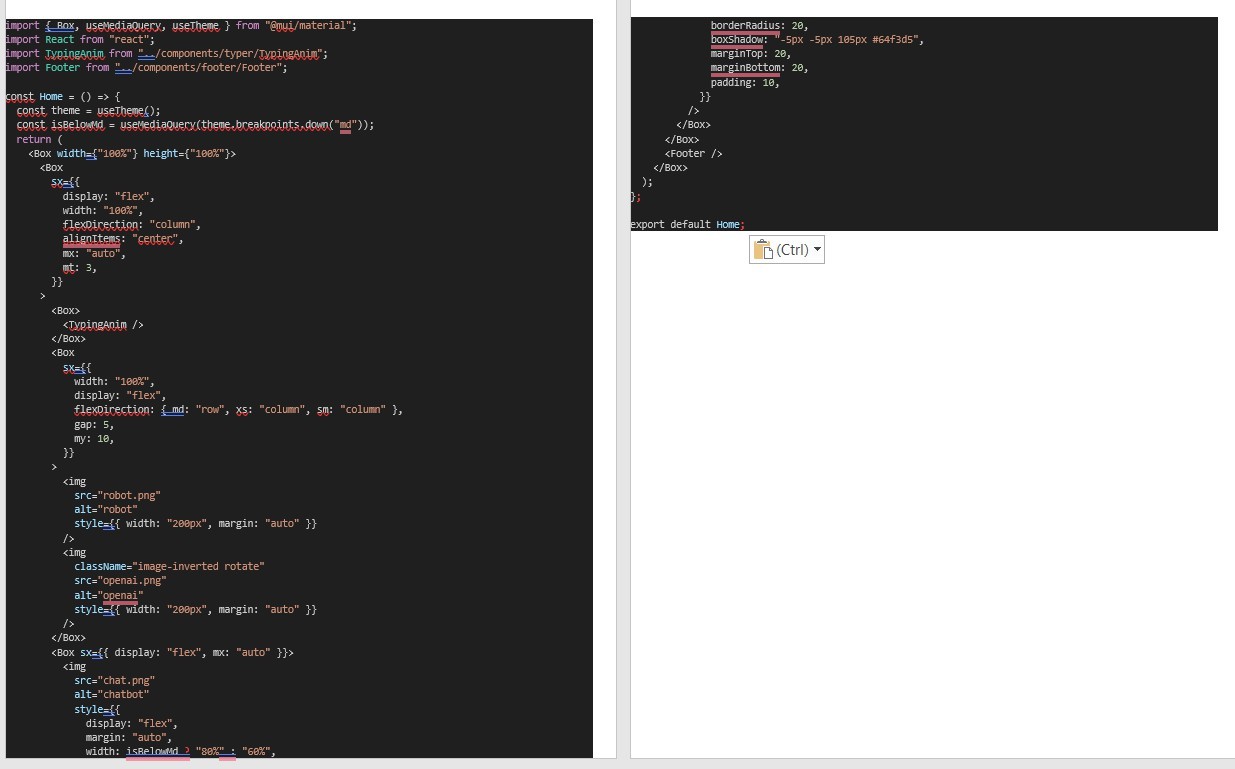
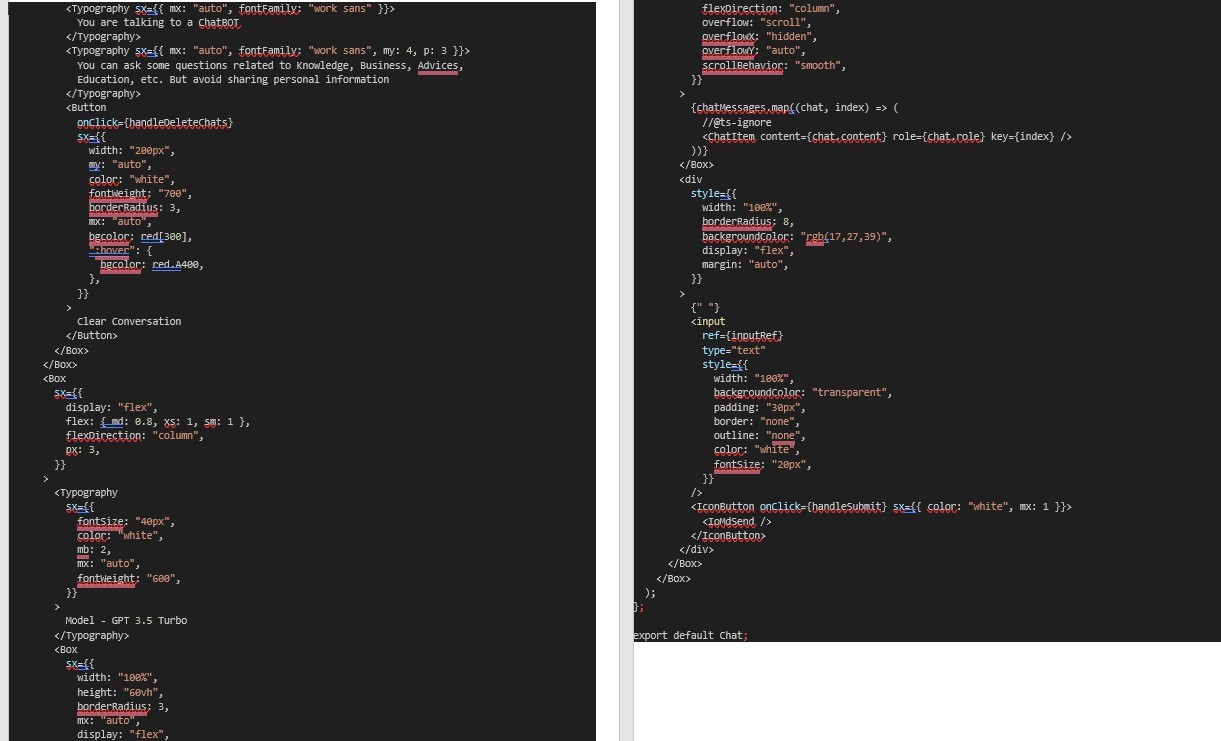
In the implementation phase of the CropChat project, the designed system is developed into a functional voice-based, multilingual assistant. This involves integrating Natural Language Processing (NLP) technologies to recognize and process farmer queries spoken in various regional languages. Speech-to-text and text-to-speech modules are implemented to handle voice input and output. The backend connects to databases and APIs to fetch real-time data on crop guidance, weather updates, pest control, and market prices. The system is then deployed on mobile platforms with a simple user interface to ensure accessibility for non-literate users. Rigorous testing is conducted to ensure the assistant responds accurately and consistently in different languages and dialects.

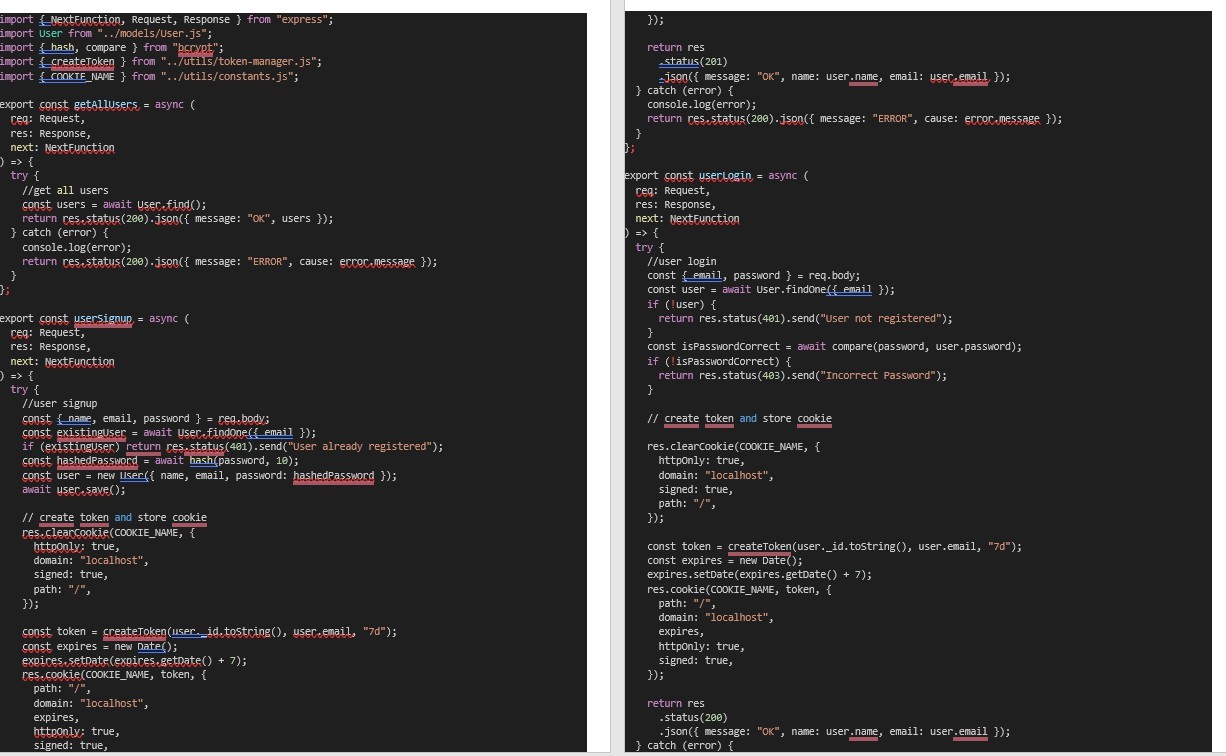
*S****creenshots***

******



******





******

**CHAPTER-5 TESTING**

## Testing

Testing was a very important phase in our CropChat project. Since our system involved multiple components like speech input, natural language processing (NLP), and real-time data fetching (like weather and market prices), we had to make sure everything worked smoothly together. Our goal was to ensure the system was reliable, easy to use, and gave correct responses to farmers’ queries.

Testing Was Important in CropChat

* To check if the voice input was accurately converted to text
* To ensure that user queries were understood correctly
* To test if data like crop info, weather, and prices were fetched and shown properly
* To make sure that the user interface was simple and functional
* And finally, to confirm that adding new features didn’t break the old ones

***Types of Testing we performed***

1. Unit Testing

We tested small parts like the speech-to-text module, the NLP logic, and response formatting. This helped us catch basic issues early before connecting everything together.

1. Integration Testing

Once individual parts worked, we tested them together. For example, we checked if a spoken query went from audio → text → NLP → database → correct reply. This showed us if the system behaved properly end-to-end.

1. Smoke Testing

Every time we made updates, we did a quick check to make sure major things like the mic, API responses, and language detection were still working. It saved us time and helped us avoid big mistakes later.

1. Regression Testing

If we changed or improved one part, we always re-tested the old features to make sure they still worked. For example, after adding support for a new language, we checked if existing queries were still giving the right answers.

1. Black Box Testing

Here, we tested the system like a real user would—without looking at the code. We asked questions like “When should I plant groundnut?” and “What is the price of rice in Coimbatore?” to see how well the chatbot handled them.

1. White Box Testing

We also checked inside the code to see how the data flowed, how errors were handled, and whether responses were being generated the way we designed. This helped us clean up logic and fix some small bugs.

1. Performance Testing (Basic)

Though our project was small, we tried sending multiple queries quickly to see how fast the system responded and whether it slowed down. This gave us some idea of how it might perform if used by more users in the future.

The development of the CropChat system was a valuable and practical experience for our team. The main objective of the project was to support farmers by providing agricultural information in a simple, accessible way using voice-based interaction in their local language. With many farmers facing challenges related to digital literacy, language barriers, and limited access to expert advice, CropChat was designed to bridge that gap using modern technologies like speech recognition and natural language processing.

Through this project, we were able to apply the core principles of software engineering, including requirement analysis, system modeling, modular design, implementation, and testing. Each stage of the development process taught us how to approach real-world problems systematically. Creating use case diagrams, DFDs, and architecture plans helped us organize our ideas clearly.

We also learned the importance of thorough testing to ensure accuracy, reliability, and usability. Integration with APIs for real-time weather and market price data helped make the system more practical and helpful.

In conclusion, CropChat gave us hands-on experience in building a socially impactful solution while improving our technical, planning, and teamwork skills. The project not only met its goals but also inspired us to explore how technology can truly empower rural communities.

## Conclusion

The development of the CropChat system was a valuable and practical experience for our team. The main objective of the project was to support farmers by providing agricultural information in a simple, accessible way using voice-based interaction in their local language. With many farmers facing challenges related to digital literacy, language barriers, and limited access to expert advice, CropChat was designed to bridge that gap using modern technologies like speech recognition and natural language processing.

Through this project, we were able to apply the core principles of software engineering, including requirement analysis, system modeling, modular design, implementation, and testing. Each stage of the development process taught us how to approach real-world problems systematically. Creating use case diagrams, DFDs, and architecture plans helped us organize our ideas clearly.

We also learned the importance of thorough testing to ensure accuracy, reliability, and usability. Integration with APIs for real-time weather and market price data helped make the system more practical and helpful.

The CropChat project offers a practical and inclusive solution to bridge the information gap faced by Indian farmers. By leveraging voice technology and multilingual NLP, it empowers farmers to access critical agricultural information easily, regardless of their literacy or technical skills. This user-friendly system enhances decision-making, improves productivity, and supports sustainable farming practices, ultimately contributing to better livelihoods and rural development.