



## **United International University**

**Group No : 07 (Tittle: AgroLand E-Commerce Application)**

**Course Code: CSE 3422                      Section : B**

**Course Name: Software Engineering**

### **Submitted By**

<b>Junied Hossain</b>	<b>011191035</b>
<b>Mithun Saha</b>	<b>011221242</b>
<b>Md. Moshiur Rahman</b>	<b>011221245</b>
<b>Alvy Rahman Masum</b>	<b>011221413</b>
<b>Abu Sufian Robin</b>	<b>0112230721</b>

### **Submitted To**

**Samin Sharaf Somik**

***Lecturer Of CSE Dept.***

***United International University***

***Submission Date: 11 July, 2025***

# Table of Contents

<b>Table of Contents.....</b>	<b>2</b>
<b>1 Introduction.....</b>	<b>3</b>
1.1 Short Description of Project.....	3
1.2 Objectives.....	3
1.3 Motivation.....	4
<b>2 Project Description.....</b>	<b>5</b>
2.1 List of Features.....	5
2.2 Description of Each Feature.....	6
<b>3 Technology Used.....</b>	<b>14</b>
3.1 Description of the Usage of Different Technologies in AgroLand.....	14
3.1.1 Frontend.....	14
3.1.2 Backend.....	15
3.1.3 Database.....	16
3.1.4 AI Models.....	18
3.1.5 Authentication.....	18
3.1.6 APIs.....	19
3.1.7 Testing.....	21
3.1.8 GIT.....	22
<b>4 Future Work.....</b>	<b>23</b>
4.1 Problems Faced and Solutions.....	23
4.2 Upgradation and New Features.....	23
<b>5 Conclusion.....</b>	<b>24</b>
<b>6 Individual Contribution.....</b>	<b>24</b>

# AgroLand

## 1 Introduction

### 1.1 Short Description of Project

The AgroLand website project is an innovative web application designed to serve as a comprehensive digital platform for the agricultural sector. It aims to connect farmers, agribusinesses, and agricultural experts, providing them with essential tools, information, and services to foster smarter, more sustainable, and profitable farming practices. Built with a modern technology stack encompassing React.js for the frontend and **Node.js with Express.js** and **Google Firebase** for the backend, AgroLand offers a dynamic and user-friendly experience, integrating features from market price insights and expert consultancy to equipment and storehouse rental services.

### 1.2 Objectives

The primary objectives of the AgroLand project are to:

- i) **Empower Farmers:** Provide farmers with easy access to critical information such as real-time market prices, weather updates, and expert agricultural advice, enabling informed decision-making.
- ii) **Facilitate Resource Sharing:** Create a platform for the efficient rental and listing of farming equipment and storehouses, optimizing resource utilization within the agricultural community.
- iii) **Promote Knowledge Exchange:** Establish a vibrant blog and community forum where users can share experiences, tips, and insights, fostering a collaborative learning environment.
- iv) **Enhance Business Efficiency:** Streamline processes for product buying and selling, and offer professional agro consultancy services to improve overall farm productivity and profitability.

v) **Build a Scalable and Robust Platform:** Develop a technically sound and maintainable application capable of supporting a growing user base and future feature expansions.

## 1.3 Motivation

AgroLand aims to address challenges in the agricultural landscape, particularly in Bangladesh, such as information asymmetry, resource underutilization, and expert guidance. By creating a rental marketplace, it can maximize the utility of farming equipment, reduce capital expenditure, and provide expert advice. AgroLand also aims to be at the forefront of digital agriculture, leveraging technology to improve efficiency and sustainability. This centralized digital hub will benefit farmers, consumers, and the agricultural ecosystem.

## 2 Project Description

AgroLand is a comprehensive web application designed to empower farmers and agribusinesses with technology, connecting them with essential resources, market insights, and expert advice. Built with React.js for a dynamic frontend and supported by a Node.js, Express.js, and Google Firebase backend, the platform aims to revolutionize agriculture through smart, accessible solutions.

### 2.1 List of Features

The AgroLand website offers a range of features tailored for both guest users and logged-in members, enhancing their farming experience and business operations:

#### **Guest User Features:**

- |                    |                                 |
|--------------------|---------------------------------|
| i) Landing Page    | ii) Features Section            |
| iii) About Section | iv) Contact Us Section          |
| v) User Login      | vi) User Registration (Sign Up) |

#### **Logged-In User Features (Dashboard Access):**

- |                           |                               |
|---------------------------|-------------------------------|
| i) Personalized Dashboard | ii) Blog Management           |
| iii) Order Management     | iv) Agro Consultancy Services |
| v) Product Marketplace    | vi) Storehouse Rental         |
| vii) Equipment Rental     | viii) Profile Editing         |
| ix) AI Recommendations    | x) Help Bot 24/7              |
| xi) Weather Update        |                               |

## 2.2 Description of Each Feature

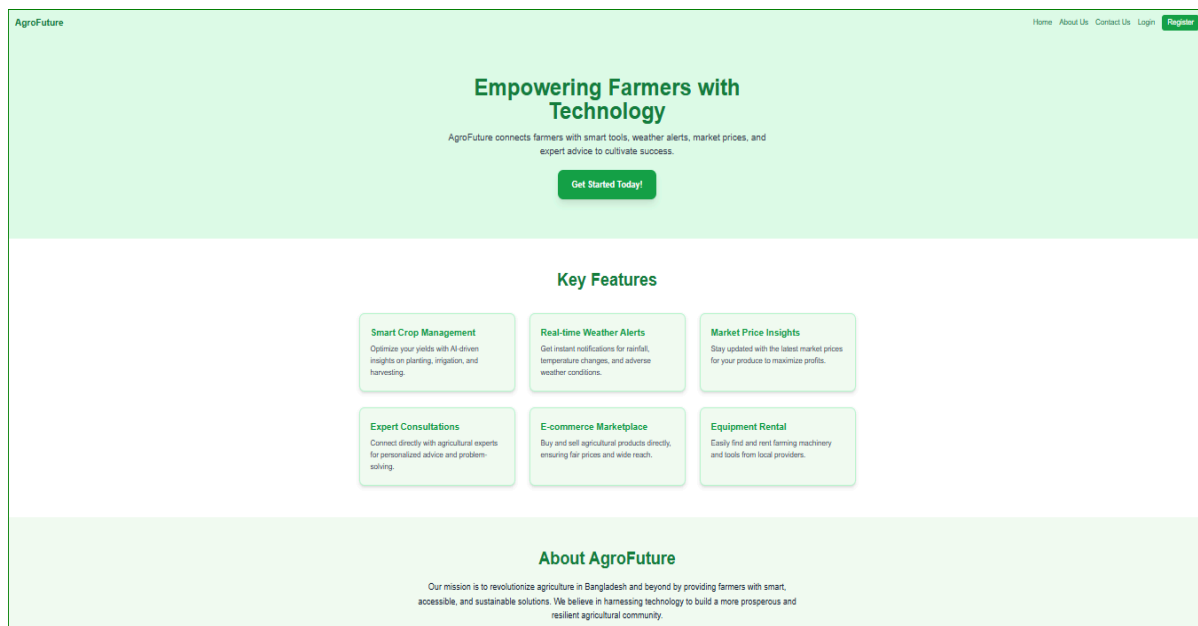
### Guest User Features:

**Landing Page:** The initial entry point for all visitors, designed to be visually appealing and informative. It provides an overview of AgroLand's mission and value proposition, encouraging new users to explore or sign up.

**Features Section:** Highlights the core functionalities and benefits of the AgroLand platform, such as market price access, expert tips, and rental services, to attract potential users.

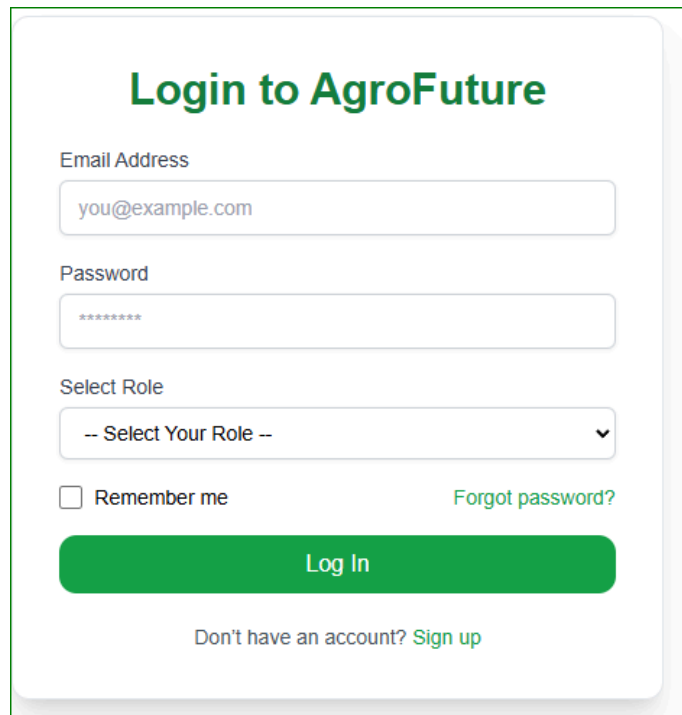
**About Section:** Details AgroLand's mission, vision, and commitment to sustainable agriculture, fostering trust and connection with the user base.

**Contact Us Section:** Provides essential contact information (email, phone, address) for users to reach out for support, inquiries, or partnerships.



*Fig 1: Guest User Features*

**User Login:** Allows existing users to securely access their personalized dashboards and features by entering their credentials and selecting their role (Agro Seller, Product Buyer, or Help-Seeker).



**Login to AgroFuture**

Email Address

Password

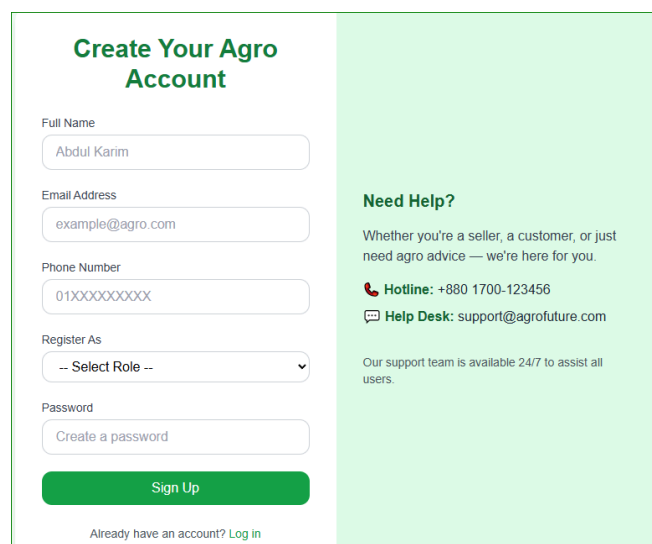
Select Role

☐ Remember me [Forgot password?](#)

Don't have an account? [Sign up](#)

*Fig 2: Login page for User Authentication*

**User Registration (Sign Up):** Enables new users to create an account by providing their full name, email, phone number, desired role, and a password, welcoming them into the AgroLand community.



**Create Your Agro Account**

Full Name

Email Address

Phone Number


Register As


Password

Already have an account? [Log in](#)

**Need Help?**

Whether you're a seller, a customer, or just need agro advice — we're here for you.

 **Hotline:** +880 1700-123456

 **Help Desk:** support@agrofutur.com

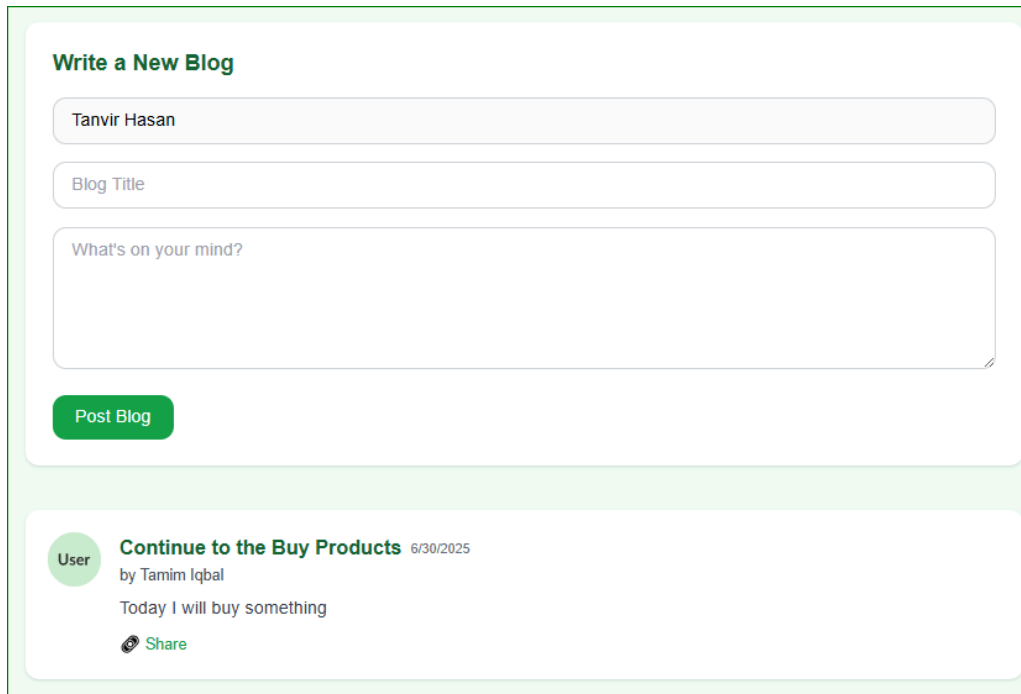
Our support team is available 24/7 to assist all users.

*Fig 3: User Sign Up Page*

### Logged-In User Features (Dashboard Access):

**Personalized Dashboard:** The central hub for logged-in users, displaying a personalized welcome message including their username. It provides quick access to various modules and an overview of relevant information.

**Blog Management:** A dedicated section where users can read, write, and share blog posts related to agriculture, fostering knowledge exchange and community engagement. Posts are persisted locally.



The image shows a UI mockup for a blog writing interface. It is divided into two main sections. The top section, titled "Write a New Blog", contains a text input field with the placeholder "Tanvir Hasan", another text input field labeled "Blog Title", and a larger text area with the placeholder "What's on your mind?". Below these fields is a green button labeled "Post Blog". The bottom section displays a sample blog post. It starts with a circular profile picture labeled "User", followed by the title "Continue to the Buy Products" and the date "6/30/2025". Below the title is the author "by Tamim Iqbal" and the text "Today I will buy something". At the bottom of the post is a "Share" button with a share icon.

*Fig 04: Blog Writing UI Page*

**Order Management:** (Placeholder) This feature will allow users to view and manage their product orders and service requests within the platform, providing a centralized transaction history.

**Agro Consultancy Services:** Offers access to professional agricultural consulting. It includes a visual cursor effect and a call-to-action for users to request expert guidance for smarter, sustainable farming practices.



**Product Marketplace:** (Placeholder) This section will serve as a marketplace for agricultural products, enabling users to browse, buy, and sell farming-related goods.


**Storehouse Rental:** Facilitates the rental of agricultural storehouses. Users can list their available storehouses or browse proposals from others, with data persisted locally.

The image shows a web interface for 'Storehouse Rental'. At the top, there's a header with a grid icon and the title 'Storehouse Rental Proposal'. Below this is a section titled 'Submit Your Proposal' containing a form with five input fields: 'Name' (pre-filled with 'Tanvir Hasan'), 'Storehouse Location', 'Size in sq ft', 'Monthly Rent (₳)', and a larger text area for 'Additional details (e.g., amenities, access)'. A green 'Submit Proposal' button is at the bottom of the form. Below the form is a section titled 'All Storehouse Proposals' displaying two proposal cards. Each card shows the location 'Dhaka', owner information, size, price, and a 'Contact Owner' button.

All Storehouse Proposals	
<b>Dhaka</b> Owner: asif1@gmail.com Size: 5000 sq ft Price: ₳25000/month Access	<b>Dhaka</b> Owner: Asif Akbar Size: 15000 sq ft Price: ₳8500/month Size is large

*Fig 05: Storehouse Rent Page*


**Equipment Rental:** Allows users to list their farming equipment for rent or find available equipment from other users. This feature helps optimize resource utilization within the farming community, with data persisted locally.


**Farming Equipment Rental**  
 Find, list, and rent farming tools easily

### List Your Equipment for Rent


Submit Listing

### Available Equipment for Rent




**Seeder**  
 Location: Rajshahi  
 Price: ₳6000/day  
 Owner: Mithun  
 Super Seeder Specifications Overall Length(mm) 1495 Overall Width(mm) 2360 Overall Height(mm) 1440 Weight(kg) 900 Tractor Power Required(HP) 45 And Above Maximum PTO power(KW) 36.3 NO. of Tynes 11 Nos Type of Blades CIL Types NO. of Blades 54 Furrow Openers Disc Type/Inverted T-tyres

Rent Now
 Contact Dealer



**Harvester**  
 Location: Naogaon  
 Price: ₳12500/day  
 Owner: Mithun  
 Yanmar Combine Harvester AG600GA Condition New Head-feeding Full Grain Tank type Water Cooled 4-stroke 4-cylinder diesel engine Provides REAPING BLOCK, SECOND WARNING, FULL PADDY, OBLIQUITY WARNING High Capacity Grain tank

Rent Now
 Contact Dealer



**Power Tiller**  
 Location: Narayanganj, bangladesh  
 Price: ₳1499.99/day  
 Owner: Mithun  
 Clutch cover made by nut bolt joining system instead of welding Fork made by Solid Cast Iron instead of welding 18 Tilling Blade New design Rotary Cover 8 Sprocket Iron wheel

Rent Now
 Contact Dealer

Fig 06: Equipment Rent Page

**Profile Editing:** Enables users to update their personal information, including full name, email address, and phone number, ensuring their account details are always current.

**AI Recommendations:** A future feature designed to provide personalized, AI-driven recommendations for crops, farming techniques, and market strategies based on user data and preferences.

## Crop Recommendation System

**Nitrogen**

**Phosphorus**

**Potassium**


**Temperature**

**Humidity**

**pH**

**Rainfall**

Get Recommendation

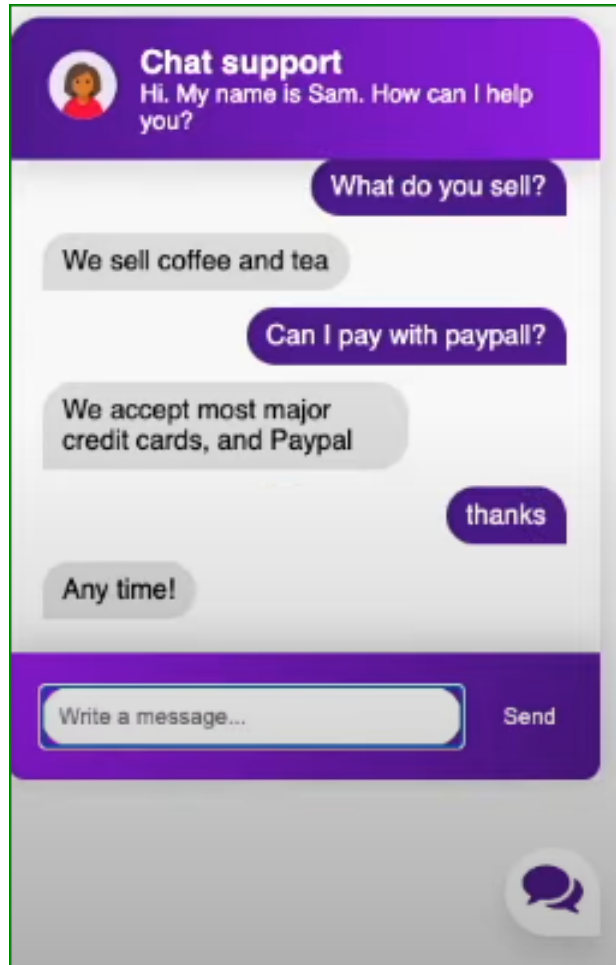


Recommend Crop for cultivation is:

Grapes is the best crop to be cultivated right there

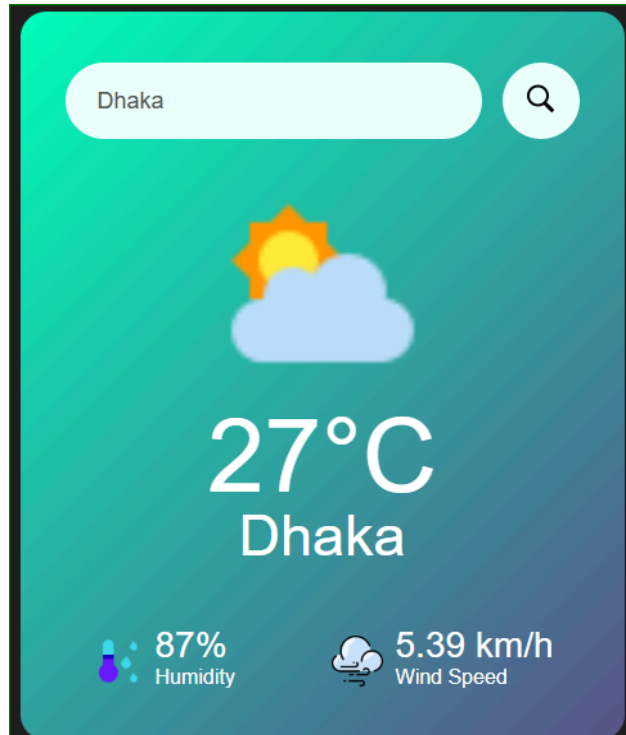
Fig 07: AI Crop Recommendation

**Help Bot:** A future feature that will offer an AI-powered chatbot for instant answers to farming queries, troubleshooting, and general guidance.



*Fig 08: ChatBot UI Page*

**Weather Update:** Located on the dashboard, this feature allows users to search for a city's weather information (currently simulated for "Dhaka" and "Chittagong"), providing crucial data for farming decisions.



*Fig 09: Weather Update Page*

This project aims to provide a robust and user-friendly platform that addresses the diverse needs of the agricultural sector in Bangladesh and beyond.

## 3 Technology Used

The AgroLand website project is built upon a modern and robust technology stack designed for scalability, performance, and a rich user experience. This section outlines the key technologies employed across the frontend, backend, and database layers, along with considerations for AI integration, authentication, APIs, testing, and version control.

### 3.1 Description of the Usage of Different Technologies in AgroLand

#### 3.1.1 Frontend

**React.js:** The entire user interface of AgroLand is developed using **React.js**, a declarative, component-based JavaScript library for building dynamic and interactive UIs.

**Component-Based Architecture:** The application is structured into reusable components (e.g., *Navbar*, *Sidebar*, *LoginPage*, *DashboardPage*), promoting modularity, maintainability, and reusability.

**State Management:** React's `useState` and `useContext` hooks are extensively used for managing component-level and global application states (e.g., *user authentication status*, *current page*, *form data*). The *AuthContext* provides a centralized way to manage user login/logout across the application.

**Client-Side Routing:** A light-weight, hash-based routing mechanism (using *window.location.hash* and *useEffect*) is implemented to navigate between different pages without full page reloads, providing a Single Page Application (SPA) experience.

**Tailwind CSS:** For styling and ensuring a consistent, responsive, and modern aesthetic, Tailwind CSS is utilized.

### ***index.css code Snippet:***

```
@tailwind base;  
@tailwind components;  
@tailwind utilities;
```

**Utility-First Approach:** Tailwind's utility classes (*flex, grid, p-4, text-xl, bg-green-500, rounded-xl, md:flex-row*) are directly applied in the JSX, allowing for rapid UI development and highly customizable designs without writing custom CSS.

**Responsiveness:** Tailwind's responsive prefixes (*sm:, md:, lg:*) are used to adapt layouts, spacing, and typography for optimal viewing across various device sizes (*mobile, tablet, desktop*).

**Material Icons:** Integrated via a CDN, Material Icons provide a rich set of vector-based icons used throughout the application for visual cues and enhanced user experience (*e.g., in the sidebar, quick action buttons*).

## 3.1.2 Backend

**Node.js:** The server-side logic for AgroLand is powered by Node.js, a JavaScript runtime built on Chrome's V8 JavaScript engine. It enables the development of fast, scalable network applications.

**Express.js:** As a minimal and flexible Node.js web application framework, Express.js is used to build the RESTful API for the backend.

**Routing:** Express handles defining routes (*/auth/signup, /auth/login*) that correspond to different functionalities.

**Middleware:** It supports middleware functions for tasks such as parsing request bodies (*express.json()*), enabling Cross-Origin Resource Sharing (*cors*), and handling authentication.

React JavaScript Code Snippet :

```
import React, { useEffect } from 'react';
import { Routes, Route } from 'react-router-dom';
```

### 3.1.3 Database

The AgroLand system utilizes a structured and scalable database system to manage diverse types of data, such as user information, rental bookings, product orders, blog content, and warehouse rental details. The backend database is designed using Firebase Realtime Database for dynamic data syncing, allowing real-time updates across devices. This ensures that changes in stock, rental availability, or user activity are immediately reflected in the system.

The database is structured to include key collections such as:

- i) Users: Storing authentication, personal information, and role (farmer, equipment owner, etc.)
- ii) Orders: Tracking agri-product orders, payment status, and delivery schedules.
- iii) Rentals: Maintaining availability and rental history for agricultural tools and storage units.
- iv) Blogs: Storing community posts and articles written by users on agricultural topics.
- v) Weather Logs: Caching weather data retrieved via external APIs for performance.

Data integrity and security are maintained using Firebase's built-in rules and validation, ensuring only authenticated users can read or write to the database as per their access level.

Firebase Data Store :

```
service cloud.firestore {

  match /databases/{database}/documents {
    // Existing rule for users collection
    match /users/{userId} {
      allow read, create: if request.auth != null;
      allow update, delete: if request.auth != null && request.auth.uid == userId;
    }
  }
}
```



```

}

// New rules for blogs
match /blogs/{blogId} {
  allow read: if true; // Anyone can read blogs
  allow create: if request.auth != null; // Only authenticated users can create
  allow update, delete: if request.auth != null && request.auth.uid == resource.data.userId; //
Only owner can update/delete
}

// New rules for instrumentRentals
match /instrumentRentals/{rentalId} {
  allow read: if true; // Anyone can read
  allow create: if request.auth != null; // Only authenticated users can create
  allow update, delete: if request.auth != null && request.auth.uid == resource.data.userId; //
Only owner can update/delete
}

// New rules for storehouseRentals
match /storehouseRentals/{proposalId} {
  allow read: if true; // Anyone can read
  allow create: if request.auth != null; // Only authenticated users can create
  allow update, delete: if request.auth != null && request.auth.uid == resource.data.userId; //
Only owner can update/delete
}
match /userActivities/{activityId} {
  allow read: if request.auth != null && request.auth.uid == resource.data.userId;
  allow create: if request.auth != null && request.auth.uid == request.resource.data.userId;
}
}
}

```

### 3.1.4 AI Models

The AgroLand project integrates Artificial Intelligence (AI) to enhance user experience and provide intelligent assistance, specifically through an **AI Chatbot** and a **Recommendation feature**.

i) **AI Chatbot:** For the interactive AI Chatbot feature, the project leverages the **Meta Llama 3.1 - 8B model**. This large language model enables the chatbot to understand natural language queries from users and generate coherent, contextually relevant responses related to agricultural topics, troubleshooting, and general inquiries. The *deployment* of this model is facilitated through a dedicated link integrated directly within the React.js code, allowing the frontend to interact with the model's capabilities to provide real-time conversational support to users.

ii) **Recommendation Feature:** To offer personalized insights and guidance, the project incorporates a Machine Learning model for its Recommendation feature. Specifically, **Random Forest sophisticated algorithm** is utilized. This model analyzes user data, preferences, and historical interactions (e.g., *crops viewed, articles read, equipment rented*) to provide tailored recommendations for optimal crops, farming techniques, market strategies, or relevant resources. Logistic Regression is chosen for its interpretability and effectiveness in classification tasks, making it suitable for predicting user interests and suggesting relevant content or actions.

### 3.1.5 Authentication

The AgroLand system employs **Firebase Authentication** for managing user login and signup processes. This choice provides a secure, robust, and scalable authentication solution, offloading the complexities of user management to a trusted third-party service.

i) **Secure User Management:** Firebase Authentication handles user registration, login, and session management, ensuring that user credentials are securely stored and processed. It supports various authentication methods, providing flexibility for future expansion.

ii) **Simplified Development:** By using Firebase Authentication, the project benefits from pre-built UI components and SDKs, significantly reducing the development time and effort required for implementing a secure authentication flow.

iii) **Integration with React:** The Firebase SDK is seamlessly integrated with the React.js frontend, allowing for straightforward implementation of user signup, login, and session persistence across the application. This ensures a smooth and secure user experience for accessing personalized features.

auth.js:

```
import { auth, db } from "../firebase"; // Import auth and db from firebase.js
import {
  createUserWithEmailAndPassword,
  signInWithEmailAndPassword,
  sendPasswordResetEmail,
  sendEmailVerification,
  updatePassword,
  signInWithPopup,
  GoogleAuthProvider,
  updateProfile, // Import updateProfile to set display name
} from "firebase/auth";
import { setDoc, doc, getDoc } from "firebase/firestore";
```

### 3.1.6 APIs

Application Programming Interfaces (APIs) are fundamental to the AgroLand project, enabling seamless communication between different components of the application and integrating external services.

i) **Custom RESTful APIs (Node.js/Express.js):** The core functionality of AgroLand relies on custom-built RESTful APIs developed using Node.js and the Express.js framework. These APIs serve as the communication layer between the React.js frontend and the MySQL database,

handling operations such as user registration and login, managing blog posts, and facilitating equipment and storehouse rental listings. These APIs define the endpoints and data structures for all internal data exchange.

ii) **Firestore API:** The Firestore API is crucial for interacting with Firebase Authentication. It provides the necessary methods and endpoints for the frontend to create new user accounts, authenticate existing users, manage user sessions, and handle password resets, ensuring a secure and reliable authentication flow.

firebase.js:

```
import { initializeApp } from "firebase/app";
import { getAuth } from "firebase/auth";
import { getFirestore } from "firebase/firestore"; // Import getFirestore

const firebaseConfig = {
  apiKey: // REPLACE WITH YOUR ACTUAL API KEY
  authDomain: "reactfirebase-b.firebaseio.com",
  projectId: "reactfirebase-b",
  storageBucket: "reactfirebase-b.firebaseio.com",
  messagingSenderId: // REPLACE WITH YOUR ACTUAL MESSAGING SENDER ID
  appId: // REPLACE WITH YOUR ACTUAL APP ID
};
```

iii) Hugging Face API: To deploy and interact with the Meta Llama 3.1 - 8B model for the AI Chatbot feature, the Hugging Face API is utilized. This API provides programmatic access to the hosted language model, allowing the React frontend to send user queries to the Llama model and receive generated responses, thereby powering the conversational AI capabilities.

*Hugging face Tokens: \*\*\*\*\*gvfg16;*

iv) Weather Map API: For providing real-time weather updates on the user dashboard, the project integrates with a Weather Map API. This external API is responsible for fetching current weather conditions, temperature, humidity, and wind information for specified city names, providing valuable environmental data for farmers to make informed decisions.

This combination of custom and third-party APIs ensures that AgroLand is a dynamic and data-rich application, capable of delivering a wide range of services and information to its users.

## WeatherUpdatesPage.jsx

```
const API_KEY = 'ada4e*****25a4ca6b1b9';
```

### 3.1.7 Testing

The AgroLand platform underwent rigorous **functional and non-functional** testing to ensure stability, performance, and user satisfaction across all modules.

i) **Unit Testing** was conducted for individual components such as the login system, blog posting feature, and equipment booking logic, ensuring each function performed as expected in isolation.

ii) **Integration Testing** was carried out to verify the correct interaction between modules—e.g., verifying that an authenticated user can make a booking, store it in the database, and receive real-time confirmation.

iii) **AI Model Testing** involved validating the performance and accuracy of the recommendation model using historical datasets. Additionally, the chatbot's responses were evaluated based on relevance, clarity, and user feedback.

iv) **API Testing** ensured that responses from third-party APIs (e.g., weather updates, model outputs from Hugging Face) were timely and correctly parsed into the UI. Fail-safe mechanisms were added to handle API downtime.

v) **Usability Testing** was conducted with a group of target users (farmers and equipment providers) to assess intuitiveness, user flow, and overall satisfaction. Feedback was used to refine UI/UX aspects and improve accessibility.

vi) **Security Testing** focused on authentication flows, data access rules, and input validation to prevent unauthorized access or data leaks.

All testing phases were documented, and issues were logged and resolved in an iterative process to maintain software quality and reliability.

### 3.1.8 GIT

To ensure efficient collaboration, version control, and code management throughout the development of the AgroLand platform, Git was used as the primary version control system. All source code, including frontend (React.js), backend logic, AI model integration scripts, and database rules, were managed through a [GitHub repository](#).

By leveraging Git, the development team was able to:

- i) **Track** changes to the codebase over time, making it easy to identify and revert specific updates when necessary.
- ii) **Collaborate** seamlessly, with each developer working on individual branches before merging updates into the main branch after thorough testing and code review.
- iii) **Maintain** project organization, using pull requests to propose, review, and integrate new features such as the AI chatbot interface, recommendation engine logic, weather API integration, and Firebase authentication flow.
- iv) **Document** contributions, allowing transparency in tracking which team member added or modified specific features.
- v) **Resolve** conflicts efficiently, especially when multiple developers worked on similar modules like order management or equipment rental.

The GitHub repository also includes comprehensive documentation, code structure explanations, and setup instructions to make the project maintainable and extensible for future contributors.

All final code used in deployment is pushed to the main branch of our private/public GitHub repository, ensuring centralized and secure storage of the AgroLand platform codebase.

## 4 Future Work

### 4.1 Problems Faced and Solutions

During the development of the AgroLand platform, several challenges emerged, especially in integrating real-time services and AI-based features. One of the primary issues was managing API rate limits and latency, particularly when fetching weather data from third-party services like OpenWeatherMap. This was addressed by implementing caching mechanisms and fallback alerts in case of API failures. Another challenge involved the integration of the **Meta LLaMA 3.1-8B** model for the chatbot feature, which required substantial compute resources and backend tuning. To resolve this, we utilized the **Hugging Face inference API** for optimized access and response handling.

On the front-end side, managing responsive UI across different devices was a recurring issue, especially when rendering real-time features like equipment booking or chat responses. This was improved by adopting a mobile-first design approach and testing extensively on multiple screen sizes. Additionally, Firebase Authentication presented occasional difficulties with **email verification and session timeout issues**, which were resolved through proper error handling, user *feedback messages*, and *persistent login status* using Firebase's session management.

### 4.2 Upgradation and New Features

Looking ahead, AgroLand has significant potential for growth through upgrades and additional features. One major area of focus is to enhance the AI recommendation system by incorporating more advanced machine learning models such as decision trees or ensemble learning to offer more accurate predictions tailored to user behavior, region, and seasonal trends. Another planned upgrade is to **develop a mobile application version of the platform, offering farmers easier access to services** and tools on the go. The mobile version would also support offline features, enabling farmers in remote areas with limited connectivity to use the platform. In terms of features, we aim to introduce live crop market pricing, integrating APIs that fetch real-time prices from local and national markets. This would help users make better selling or buying decisions. Moreover, we plan to **add payment gateway integration** to allow secure transactions for orders and equipment rentals directly within the platform.

Future upgrades also include ***multi-language support, allowing users from different regions of the country to interact with the system in their native language, and chatbot voice interaction, enabling users with limited literacy to engage through speech***. These new features and upgrades will make AgroLand more robust, user-friendly, and inclusive, paving the way for digital transformation in agriculture.

## 5 Conclusion

The development of the AgroLand platform marks a significant step toward modernizing agriculture through the integration of technology, AI, and user-centered services. By offering features such as agricultural equipment rental, weather forecasting, intelligent crop recommendations, and real-time chat support, the system addresses the key needs of farmers, equipment providers, and agriculture enthusiasts. The use of Firebase for authentication and database management ensures secure, real-time operations, while the inclusion of AI models like *Meta LLaMA 3.1-8B* and *random forest* enhances user interaction and decision-making.

## 6 Individual Contribution

Team Member	Completed Tasks
<b><i>Mithun Saha</i></b>	<i>UI/UX Design</i>
<b><i>Alvy Rahman Masum</i></b>	<i>Frontend Integration using React JS, Tailwind CSS.</i>
<b><i>Junied Hossain</i></b>	<i>Frontend Development and Feature Tester.</i>
<b><i>Md. Moshiur Rahman</i></b>	<i>Machine Learning (Random Forest for AI Recommendations ) Feature Implementation.</i>
<b><i>Abu Sufian Robin</i></b>	<i>Backend Integration using Firebase, Express JS, Node JS and AI ChatBot Feature Implementation.</i>