

CHAPTER 1

INTRODUCTION

In the modern agricultural sector, characterized by its complexity and the critical need for efficiency, the introduction of innovative digital solutions is essential for improving both market dynamics and farmer welfare. This project presents an advanced online platform designed to address the persistent challenges faced by farmers and enhance their overall productivity and profitability. Traditional agricultural markets are often burdened by lengthy supply chains and multiple intermediaries, which not only inflate costs but also diminish the direct relationship between farmers and consumers. By leveraging digital technology, this platform aims to transform these traditional paradigms. It facilitates a direct marketplace where farmers can present and sell their produce straight to consumers, bypassing intermediaries. This direct-to-consumer model empowers farmers to set competitive prices for their products and engage in fair trade practices, thus ensuring they receive a fair share of the revenue generated from their produce. Moreover, this approach fosters greater market transparency and efficiency, which benefits both farmers and consumers by providing clearer pricing and a more direct connection between supply and demand.

In addition to revolutionizing market transactions, the platform addresses another significant challenge faced by many farmers: land management. Access to suitable and efficiently managed land is crucial for maximizing agricultural output, yet many farmers struggle with land availability and optimal utilization. To address this, the platform integrates a land leasing feature that connects farmers with landowners. This feature simplifies the process of leasing land, offering flexible and efficient solutions for farmers looking to expand their operations or enhance their land use. By streamlining the land leasing process, the platform enables farmers to better manage their resources and increase their agricultural productivity.

The combination of these features within a single digital platform represents a substantial advancement in agricultural commerce. It not only enhances market efficiency and supports fair trade but also provides practical solutions for land management challenges. This holistic approach aims to create a more resilient and equitable agricultural market, where farmers can thrive economically and sustainably. Consumers, in turn, benefit from greater access to fresh, high-quality produce, resulting in a more connected and efficient food supply chain. By addressing key issues such as market transparency, pricing, and land use, the platform is poised to make a significant impact on the agricultural sector, driving positive change and supporting the long-term success of farming operations.

1.1 PROBLEM STATEMENT

The agricultural sector is hindered by inefficient supply chains, high costs due to intermediaries, and challenges in land management. Farmers often struggle with setting fair prices and accessing suitable land, which impacts their profitability and productivity. Consumers face difficulties accessing fresh, high-quality produce directly from producers. There is a need for an integrated solution that streamlines direct transactions between farmers and consumers, enhances pricing transparency, and simplifies land leasing to optimize land use and improve overall agricultural efficiency.

1.2 OVERVIEW

This report outlines the development of an innovative online platform for agricultural commerce, built using React.js for the frontend and supported by Spring Boot and MySQL for backend connectivity. The platform enables farmers to market their produce directly to consumers, set competitive prices, and manage land leasing efficiently. It aims to streamline agricultural transactions and enhance productivity through a seamless, integrated approach.

1.3 OBJECTIVE

The objective of the agricultural commerce platform is to provide farmers with a comprehensive tool to directly market their produce and manage their agricultural operations more effectively. The platform is designed to allow farmers to set competitive prices for their products, engage in transparent trade with consumers, and optimize land use through a streamlined land leasing feature. Additionally, the platform will offer access to market insights, reviews, and expert advice to help farmers make informed decisions. By integrating these features, the platform aims to enhance market efficiency, improve farmer welfare, and support sustainable agricultural practices.

CHAPTER 2

LITERATURE SURVEY

2.1 RELATED WORKS

[1] "Digital Platforms for Agriculture: A Review of Current Trends and Future Prospects" by H. Chen, Y. Liu, and X. Wang.

This paper provides a comprehensive review of digital platforms in agriculture, focusing on their role in enhancing market efficiency and farmer welfare. The authors analyze current trends in digital agricultural platforms, discussing their potential to transform agricultural practices through direct marketing, data-driven decision-making, and improved resource management. They explore the benefits and challenges associated with these platforms, including issues of accessibility, scalability, and integration with existing agricultural systems.

[2] "Connecting Farmers and Consumers: A Review of Online Agricultural Marketplaces" by R. Tucker and S. Nair. The authors review literature on online agricultural marketplaces that connect farmers directly with consumers. This paper discusses the advantages of direct-to-consumer models in agriculture, such as increased transparency, fair pricing, and enhanced market access for farmers. It also examines the challenges faced by such platforms, including technological barriers, user adoption, and the need for robust logistical support.

[3] "Land Leasing and Agricultural Productivity: Insights from Digital Solutions" by K. Deininger, S. Jin, and T. Kato. This paper explores the impact of digital solutions on land leasing practices in agriculture. The authors provide an overview of how digital platforms facilitate land leasing, offering insights into their benefits for optimizing

land use and enhancing agricultural productivity. They discuss the theoretical and empirical evidence on the effectiveness of these solutions in improving access to land and managing land resources efficiently.

[4] "Enhancing Farmer Decision-Making with Digital Tools: A Literature Review" by A. Kumar and P. Sharma. This review examines the role of digital tools in enhancing decision-making for farmers. The paper discusses various digital platforms and applications that assist farmers in making informed decisions about crop management, resource allocation, and market strategies. It highlights the benefits of these tools in improving operational efficiency and economic outcomes for farmers, as well as the challenges of integrating digital solutions into traditional farming practices.

[5] "The Impact of Digital Platforms on Agricultural Supply Chains: A Comprehensive Review" by L. Smith, T. Jones, and M. Green.] This paper provides a detailed analysis of the impact of digital platforms on agricultural supply chains. The authors review how these platforms contribute to streamlining supply chain processes, reducing transaction costs, and enhancing transparency and traceability. They discuss the theoretical and empirical evidence on the effects of digital platforms on supply chain efficiency and farmer engagement, as well as the challenges of implementing such technologies in diverse agricultural contexts.

CHAPTER 3

SYSTEM SPECIFICATION

In this chapter, we are going to see the software that we have used to build the website. This chapter gives you a small description about the software used in the project.

3.1 VS CODE

Visual Studio Code is a source code editor developed by Microsoft for Windows, Linux, and macOS. It includes support for debugging, embedded Git control, syntax highlighting, intelligent code completion, snippets, and code refactoring. It is also customizable, so users can change the editor's theme, keyboard shortcuts, and preferences.

VS Code is an excellent code editor for React projects. It is lightweight, customizable, and has a wide range of features that make it ideal for React development. It has built-in support for JavaScript, JSX, and TypeScript, and enables developers to quickly move between files and view detailed type definitions. It also has a built-in terminal for running tasks. Additionally, VS Code has an extensive library of extensions that allow developers to quickly add features like code snippets, debugging tools, and linting support to their projects.

3.2 REACT

React is a JavaScript library created by Facebook for building user interfaces. It is a component-based, declarative, and highly efficient library that is used to develop interactive UIs (user interfaces) for single page web applications. React uses a virtual DOM (Document Object Model) that makes it faster and easier to manipulate the DOM

elements. It also provides declarative components that allow developers to write code that is easy to read and maintain. React also offers an extensive library of tools and components that make it easier to develop complex user interfaces.

3.3 ROUTERS IN REACT

Routers are important components in React applications. They provide the ability to navigate between different views or components of the application. React Router is the most popular library to handle routing in React applications. It provides the ability to define routes, set up links, and render components based on the current route. It also provides features like data fetching, code-splitting, and server-side rendering.

3.4 LOCAL STORAGE

Local storage is a type of web storage for storing data on the client side of a web browser. It allows websites to store data on a user's computer, which can then be accessed by the website again when the user returns. Local storage is a more secure alternative to cookies because it allows websites to store data without having to send it back and forth with each request. Local storage is a key-value pair storage mechanism, meaning it stores data in the form of a key and corresponding value. It is similar to a database table in that it stores data in columns and rows, except that local storage stores the data in the browser rather than in a database. Local storage is often used to store user information such as preferences and settings, or to store data that is not meant to be shared with other websites. It is also used to cache data to improve the performance of a website. Local storage is supported by all modern web browsers, including Chrome,

Firefox, Safari, and Edge. It is accessible through the browser's JavaScript API. Local storage is a powerful tool for websites to store data on the client side. It is secure, efficient, and can be used to store data that does not need to be shared with other websites.

Local Storage is a great way to improve the performance of a website by caching data. Local storage in web browsers allows website data to be stored locally on the user's computer. It is a way of persistently storing data on the client side, which is not sent to the server with each request. This allows users to store data such as preferences, login information, and form data without needing to send it to a server. It is typically stored in a browser's cookie file, but it can also be stored in other locations such as HTML5 Local Storage and Indexed DB. The data stored in local storage is persistent and can be accessed by the website even if the user closes the browser or navigates to another page. It is a great way for websites to store user-specific data, as it is secure, reliable, and fast. It is also a great way for developers to store data that does not need to be sent to the server with each request.

One of the key benefits of using local storage is its reliability. Unlike server-side storage, which can be affected by network outages or other server issues, local storage is stored locally on the user's machine, and so is not affected by these issues. Another advantage of local storage is its speed. Because the data is stored locally, it is accessed quickly, as there is no need to send requests to a server. This makes it ideal for storing data that needs to be accessed quickly, such as user preferences or session data. Local storage is also secure, as the data is stored on the user's machine and not on a server. This means that the data is not accessible by anyone other than the user, making it a good choice for storing sensitive information.

CHAPTER 4

PROPOSED SYSTEM

This chapter gives a small description about the proposed idea behind the development of our website

4.1 PROPOSED SYSTEM

The proposed FarmAssist website aims to revolutionize agricultural commerce by offering a streamlined, intuitive platform that directly connects farmers with vendors and landowners. The site is designed to be fully optimized for both desktop and mobile devices, ensuring a seamless user experience across all platforms. Central to FarmAssist is its direct connection feature, which allows farmers to market their produce straight to vendors and consumers, bypassing traditional intermediaries. This direct interaction not only facilitates more efficient transactions but also empowers farmers to set competitive prices and engage in transparent trade, enhancing both market efficiency and fairness.

Additionally, FarmAssist includes a robust land leasing service that connects farmers with landowners to simplify the process of leasing land. This feature provides a user-friendly interface for farmers to explore and secure additional land, offering flexible leasing options that can be tailored to their specific needs. By streamlining land leasing, the platform helps optimize land use, supports better resource management, and enables farmers to expand their operations and improve productivity. The integration of these features within FarmAssist is designed to create a more efficient and equitable agricultural market, promoting sustainable farming practices and providing valuable resources to enhance farmer welfare and operational success.

4.2 ADVANTAGES

- **Direct Market Access:** FarmAssist allows farmers to connect directly with vendors and consumers, eliminating the need for intermediaries. This direct access results in better pricing control, improved transparency, and increased market efficiency, benefiting both farmers and buyers.
- **Enhanced Pricing Flexibility:** By removing middlemen, farmers can set competitive prices for their produce based on real-time market conditions and their own cost structures. This flexibility helps farmers maximize their revenue and respond more effectively to market demand.
- **Improved Trade Transparency:** The platform fosters transparent trade practices by enabling direct communication between farmers and buyers. This transparency builds trust and ensures fair transactions, as both parties have clear insights into pricing, quality, and delivery terms.
- **Efficient Land Leasing:** FarmAssist simplifies the process of land leasing by connecting farmers with landowners through a streamlined interface. This efficiency facilitates better land utilization and resource management, allowing farmers to expand their operations or access additional land as needed.
- **Increased Market Reach:** The platform enables farmers to reach a broader audience of consumers and vendors beyond their local markets. This expanded reach opens up new opportunities for sales and growth, helping farmers access larger and potentially more profitable markets.
- **User-Friendly Interface:** Designed for ease of use, FarmAssist provides a modern and intuitive interface that simplifies navigation and interaction. Both farmers and vendors benefit from a straightforward user experience, making it easier to manage listings, transactions, and land leases.

CHAPTER 5

METHODOLOGIES

This chapter gives a small description about how our system works.

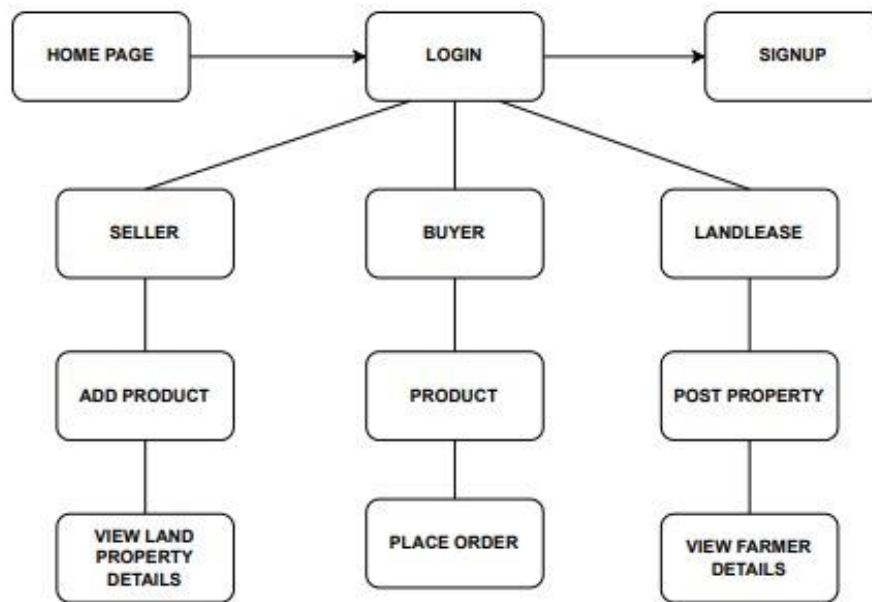


Fig 5.1 PROCESS FLOW DIAGRAM

5.1 LOGIN

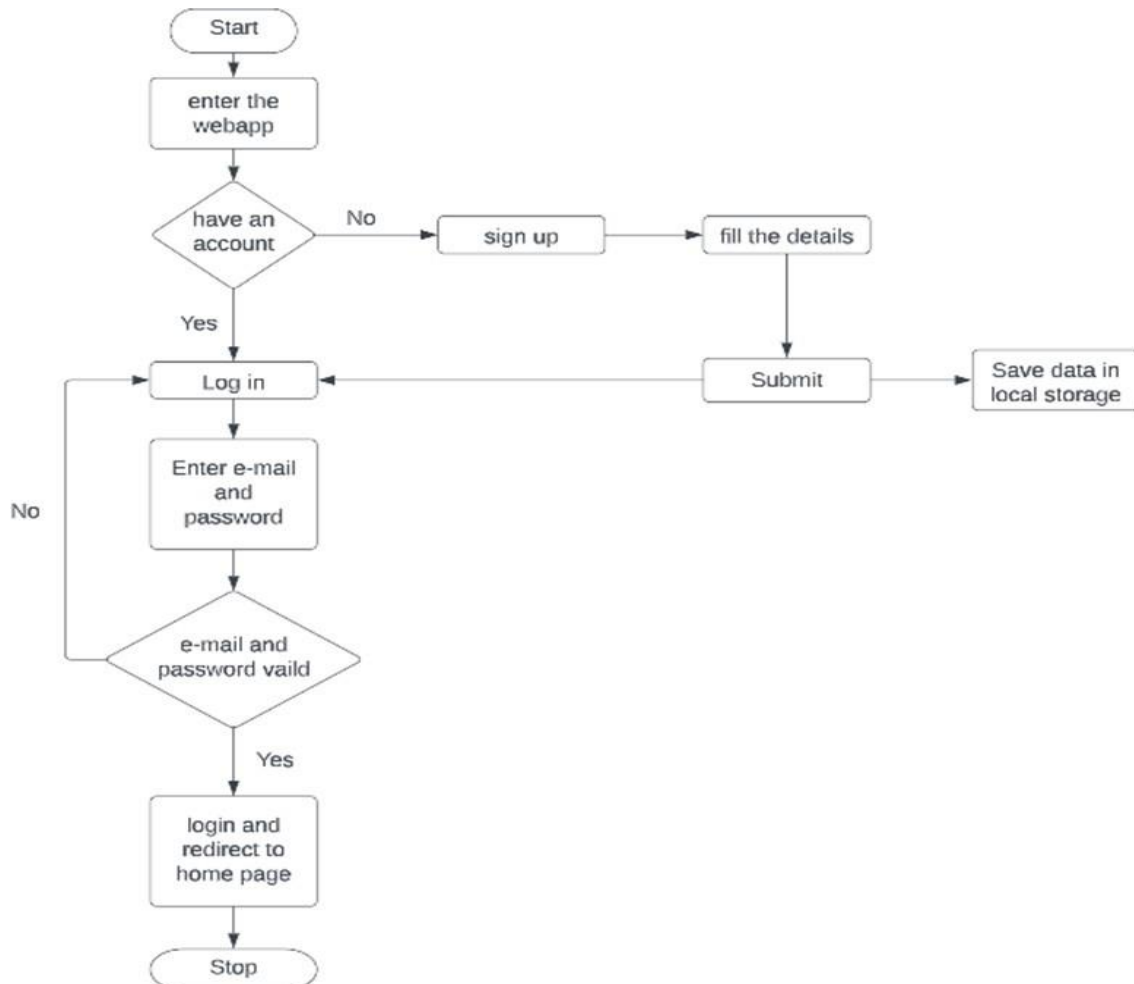


Fig 5.2 LOGIN PAGE FLOWCHART

In this page we will be asking about the username and password of the user. Firstly the website validates the user inputs. It verifies the username and password by checking it with the usernames and passwords stored in the local storage when the user creates an account in the website.

5.2 SIGNUP PAGE

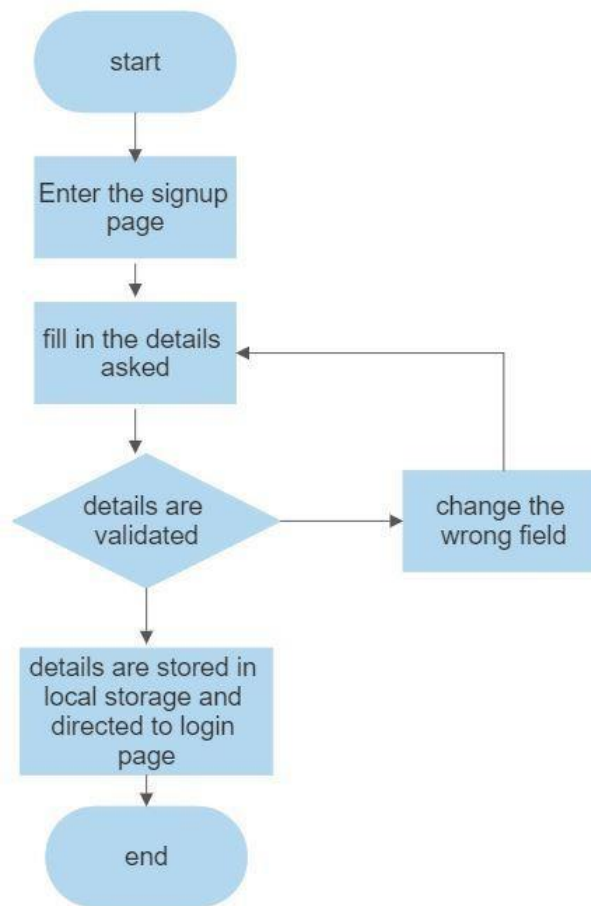


Fig 5.3 SIGNUP PAGE FLOWCHART

This page asks users about the basic details of the user to create an account. This page asks for details like username, password ,email id, phone number. After the user enters the details, these details are then validated by our code. If all details are correct then the user is then directed to the login page

5.3 SELLER PAGE

This page displays the list of sellers available on the FarmAssist platform. The information is dynamically populated based on the data maintained in the system's backend. This approach allows for easy updates and management of seller details. To add or modify seller information, updates can be made directly through the backend management system, ensuring that changes are seamlessly reflected on the website. This system allows for efficient handling of seller profiles and product listings, providing a streamlined and flexible solution for maintaining accurate and up-to-date seller information.

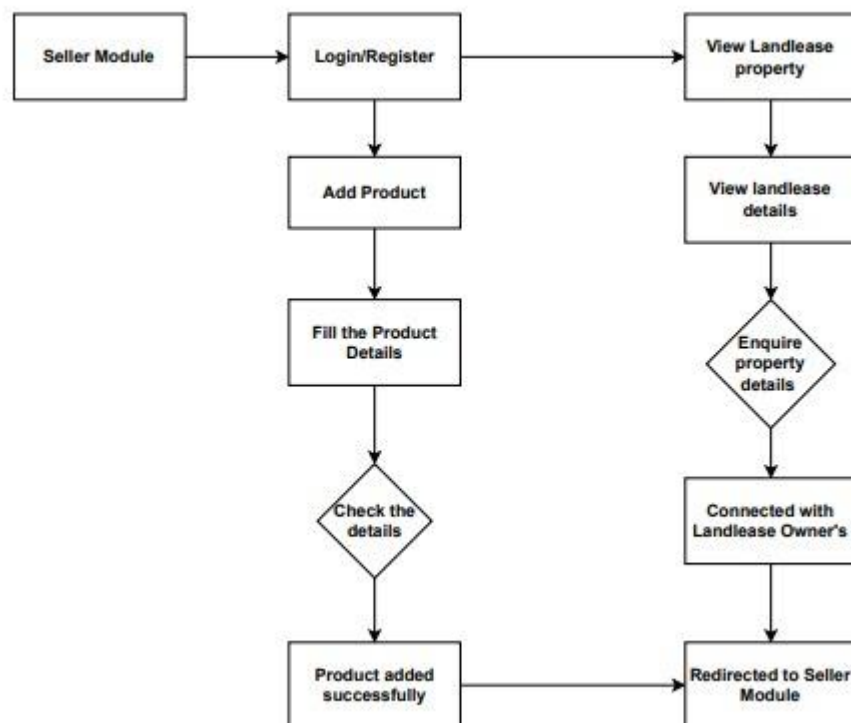


Fig 5.4 SELLER PAGE FLOWCHART

5.4 BUYER PAGE

This page allows buyers to browse through available products on the FarmAssist platform, viewing details such as product names, descriptions, prices, and quantities. Buyers can easily add items to their cart and specify quantities. Additionally, the page features an order request form where buyers can fill in their product selections and any special instructions. After completing the form, buyers can submit their order directly through the platform. If no products are added or no orders are placed, the page will display a message indicating an empty cart or inactive order status.

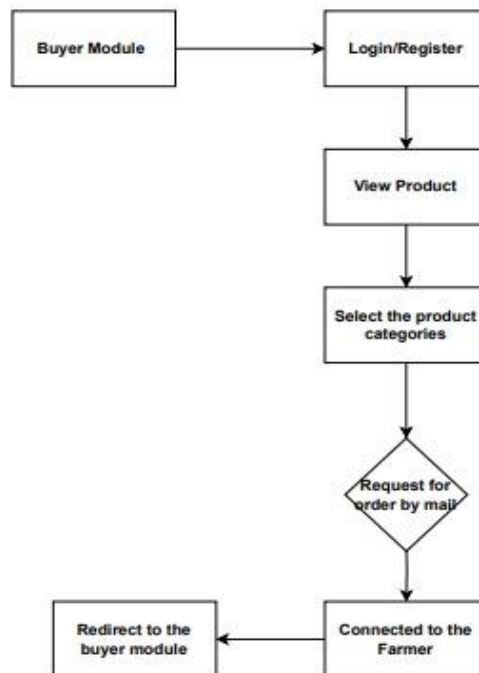


Fig 5.5 BUYER PAGE FLOWCHART

5.5 LAND TENANCY PAGE

This page enables landowners to post available properties for lease, including details such as location, size, and lease terms. Farmers can browse these listings, view property details, and contact landowners directly to express interest.

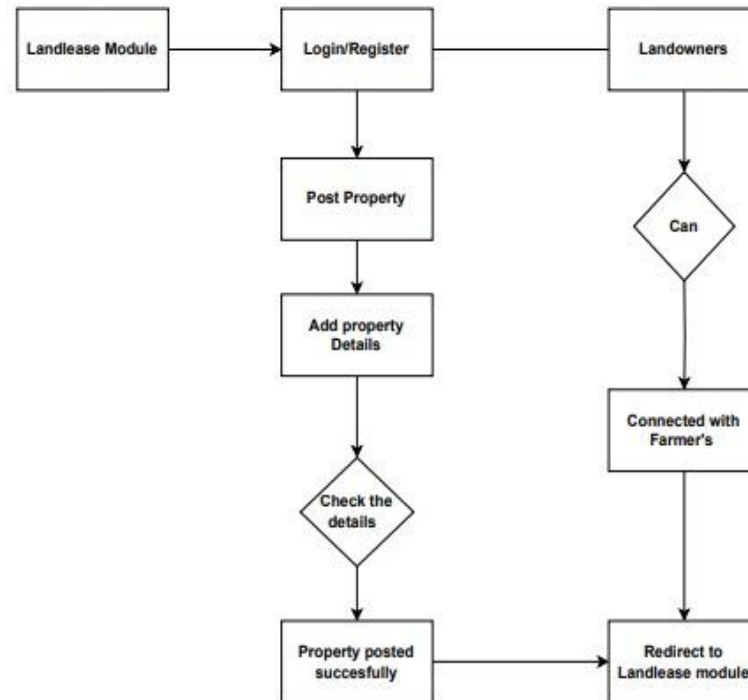


Fig 5.6 LAND TENANCY PAGE FLOWCHART

CHAPTER 6

IMPLEMENTATION AND RESULT

This chapter gives a description about the output that we produced by developing the website of our idea.

6.1 HOME PAGE

When users enter the website, they are greeted with an overview of the platform's features and services. They can navigate through various sections, such as product listings, land tenancy options, and user accounts. The home page provides easy access to key functionalities, including browsing available products, viewing land lease opportunities, and accessing personal account information. If users are not logged in, they will be prompted to enter their login details, such as email and password, which will be verified against the credentials provided during account creation.

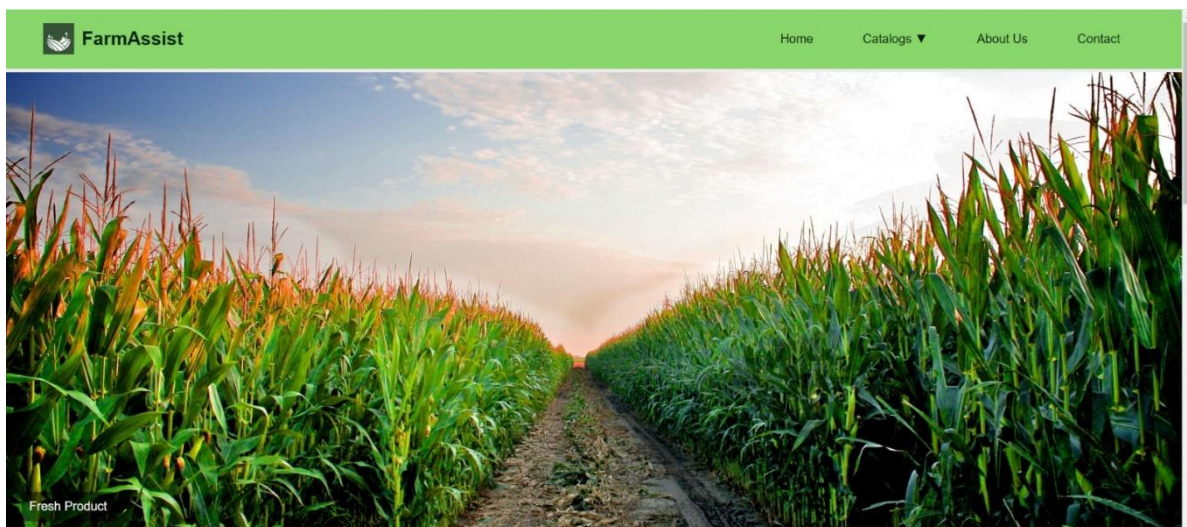


Fig 6.1. HOME PAGE

6.2 FARMER HOME PAGE

When farmers visit the FarmAssist home page, they are welcomed with an overview of key features tailored to their needs. The page provides easy access to manage their product listings, explore land leasing opportunities, and view market insights. Farmers can quickly navigate to their dashboard, update their profile, and connect with landowners. If not logged in, they will be prompted to enter their login details to access personalized features and services.

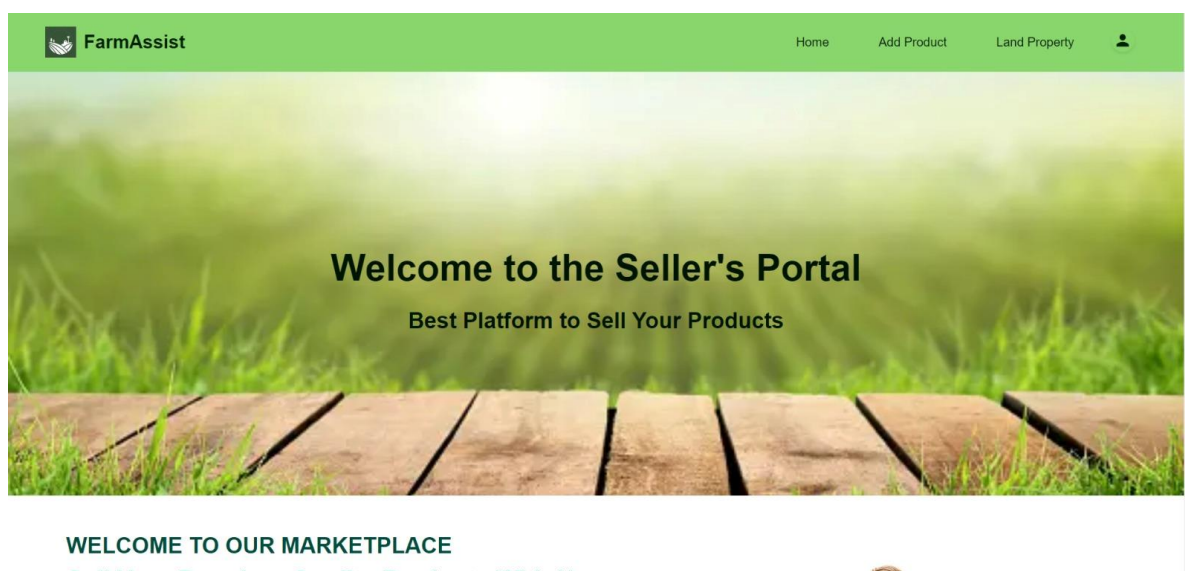
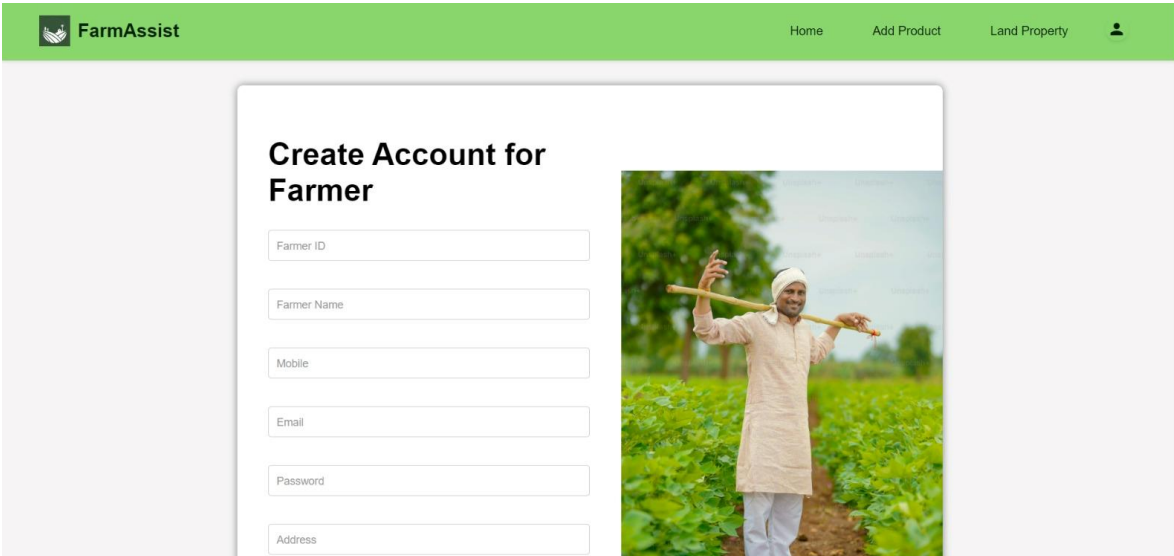


Fig 6.2 FARMER HOME PAGE

6.2.1 Farmer registration page

On the Farmer Registration page, new users can create an account by entering essential details such as their name, email address, and a secure password. Additional fields include farm location, type of produce, and contact information. After completing the registration form and submitting it, users will be redirected to the login page.

Upon successful registration, they are prompted to log in with their new credentials to access and utilize the platform's features. If there are any errors or missing information, the page will prompt users to correct the details before proceeding.

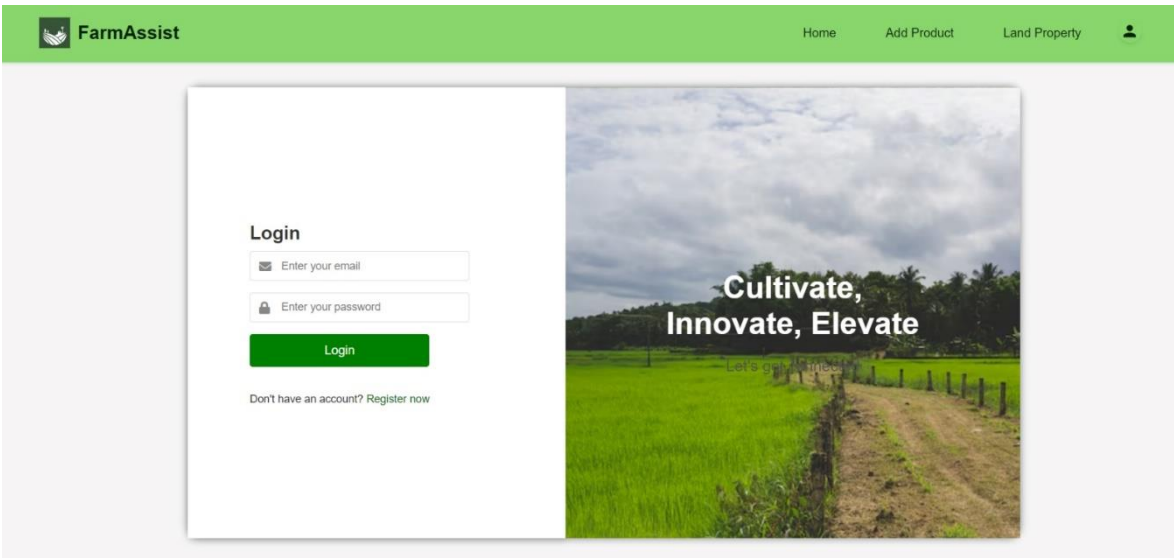


The screenshot shows the 'Create Account for Farmer' page of the FarmAssist application. The page has a green header with the FarmAssist logo and navigation links: Home, Add Product, Land Property, and a user icon. The main content area is white and contains a registration form with the following fields: Farmer ID, Farmer Name, Mobile, Email, Password, and Address. To the right of the form is a large image of a farmer in a field, holding a wooden staff.

Fig 6.3 FARMER REGISTRATION PAGE

6.2.2 Farmer login page

On the Login page, users enter their email address and password. The system verifies these credentials and, if correct, redirects them to their dashboard. If the credentials are incorrect, users will be prompted to re-enter their details.



The screenshot shows the 'Login' page of the FarmAssist application. The page has a green header with the FarmAssist logo and navigation links: Home, Add Product, Land Property, and a user icon. The main content area is white and contains a login form with the following fields: Enter your email, Enter your password, and a green Login button. Below the button is a link: Don't have an account? Register now. To the right of the form is a large image of a green field with a dirt path, with the text 'Cultivate, Innovate, Elevate' and 'Let's get started' overlaid.

Fig 6.4 FARMER LOGIN PAGE

6.2.3 Farmer add product page

On the Add Product page, users can enter details about their new product, including the name, description, price, and quantity. They can also upload images and specify any relevant categories or tags. Once all information is provided, users submit the form to add the product to their listings.

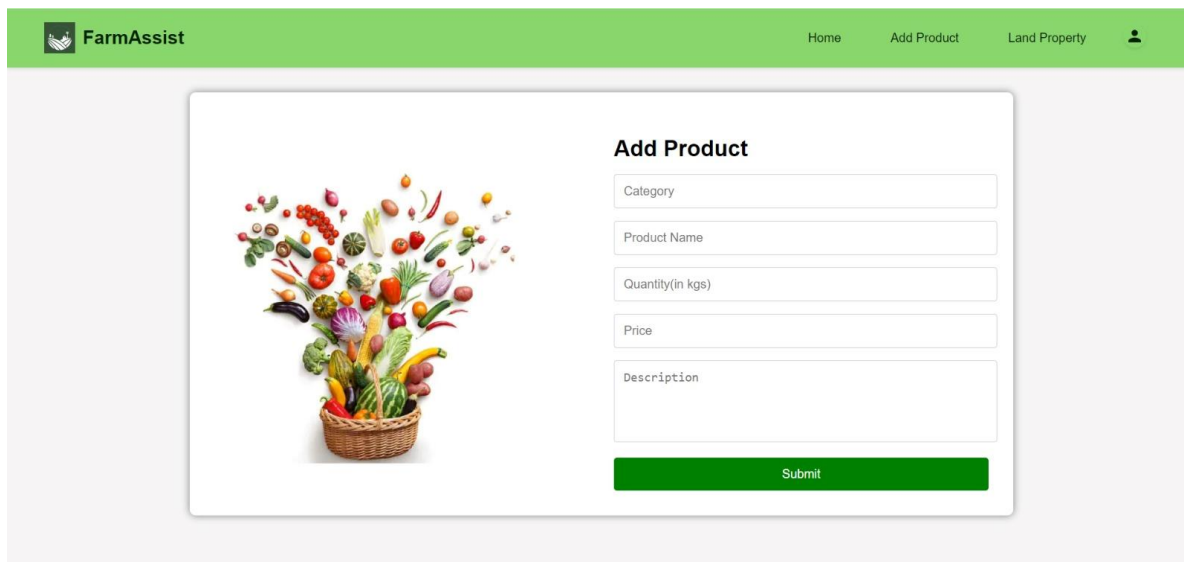


Fig 6.5 ADD PRODUCT PAGE

6.2.4 Product listing page

On the Product Listing page, once a farmer has added a product, it will appear with details such as the product name, description, price, and quantity. The page allows farmers to view .

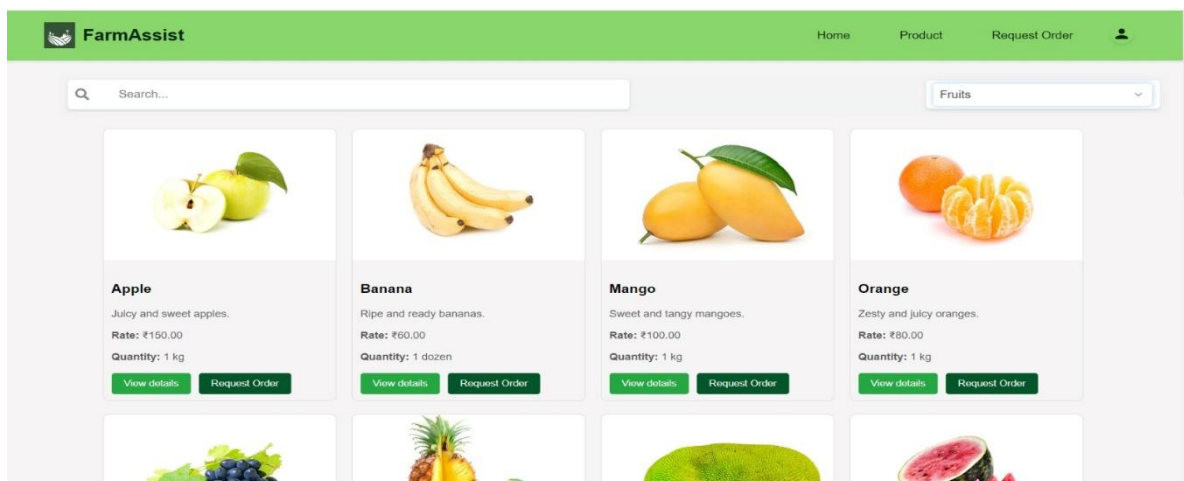


Fig 6.6 PRODUCT LISTING PAGE

6.3 BUYER PAGE

Buyers can view available products and place order requests to farmers. If not logged in, buyers are prompted to log in to access these features.

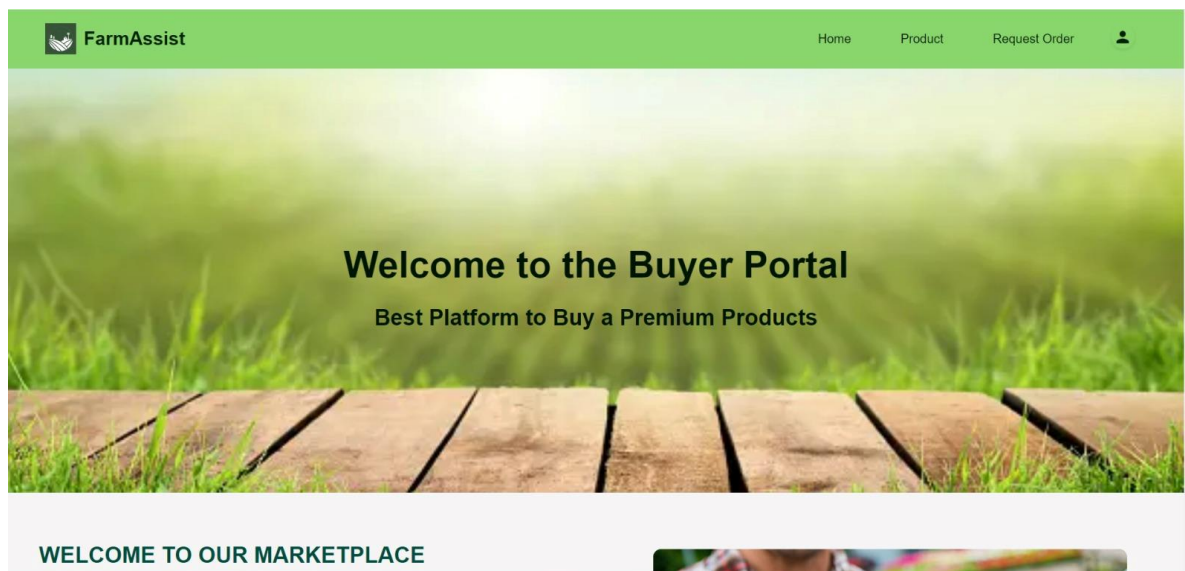


Fig 6.7 BUYER PAGE

6.3.1 Buyer registration page

Register to create your account and access available products. Once registered, you'll be directed to log in to place order requests with farmers.

 The screenshot displays the FarmAssist Buyer Registration page. It has the same green navigation bar as the previous page. The main content area is a light gray background with a white registration form centered on the page. The form is titled "Create Account for Buyer" in bold black text. It contains six input fields: "Company Name", "Buyer Name", "Mobile No", "Your Email", "Password", and "Confirm Password". Below these fields is a green "Register" button. To the right of the form is a colorful illustration of a farmer in a blue shirt and straw hat standing next to a wooden fruit and vegetable stand. The stand has a sign that says "FRUITS & VEGETABLES" and is filled with various produce like apples, bananas, and pumpkins.

Fig 6.8 BUYER REGISTRATION PAGE

6.3.2 Buyer login page

Log in to view available products and place order requests with farmers. If you don't have an account, please register first.

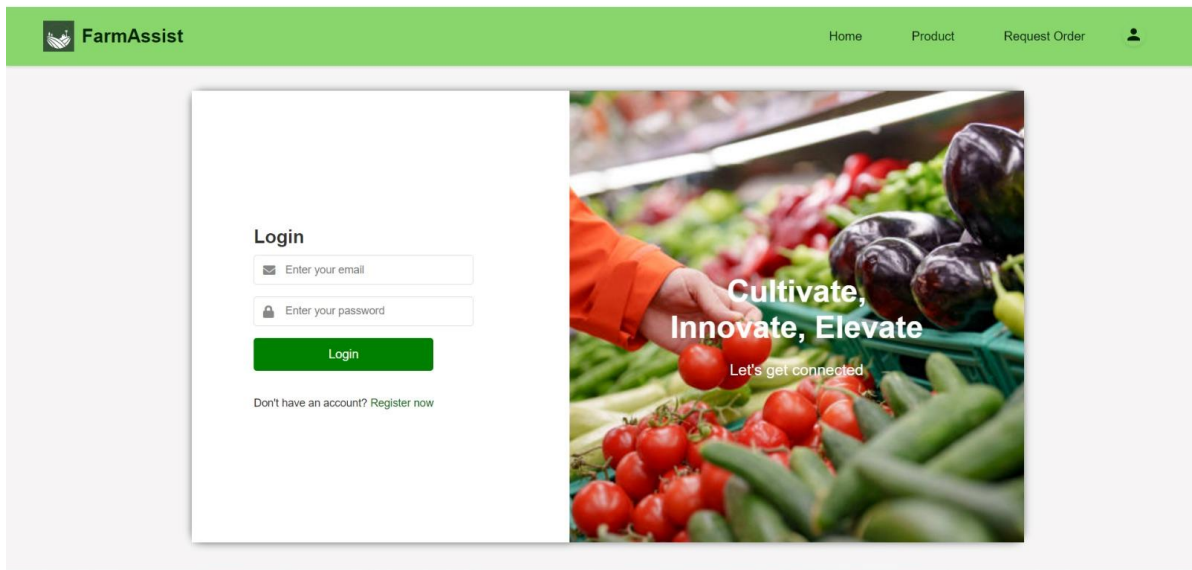


Fig 6.9 BUYER LOGIN PAGE

6.3.3 Product listing page

On the View Product page, users can see details about each product, including name, description, price, and quantity. Users can also view images and relevant tags. To request an order, users fill out an order request form to connect with the farmer.

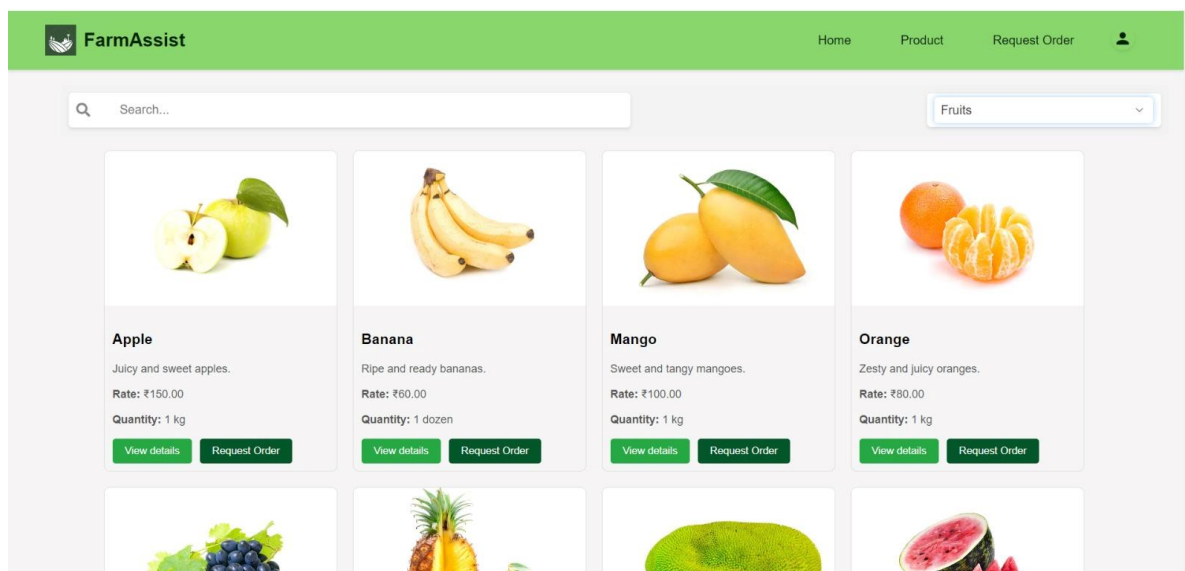


Fig 6.10 PRODUCT LISTING PAGE

6.3.4 Order request form

On the Order Request page, buyers can enter details such as the product name, quantity, and preferred delivery date. They can also add any special instructions or requests. Once the form is completed, submit it to send the order request directly to the farmer.

Fig 6.11 ORDER REQUEST FORM

6.4 LAND TENANCY

Explore available land for lease in our Land Tenancy section. Landowners can list their properties with details such as location, size, and lease terms. Farmers can use search filters to find land that fits their needs. Easily connect with landowners to discuss lease agreements and explore new agricultural opportunities. Start by browsing the listings or posting your property.

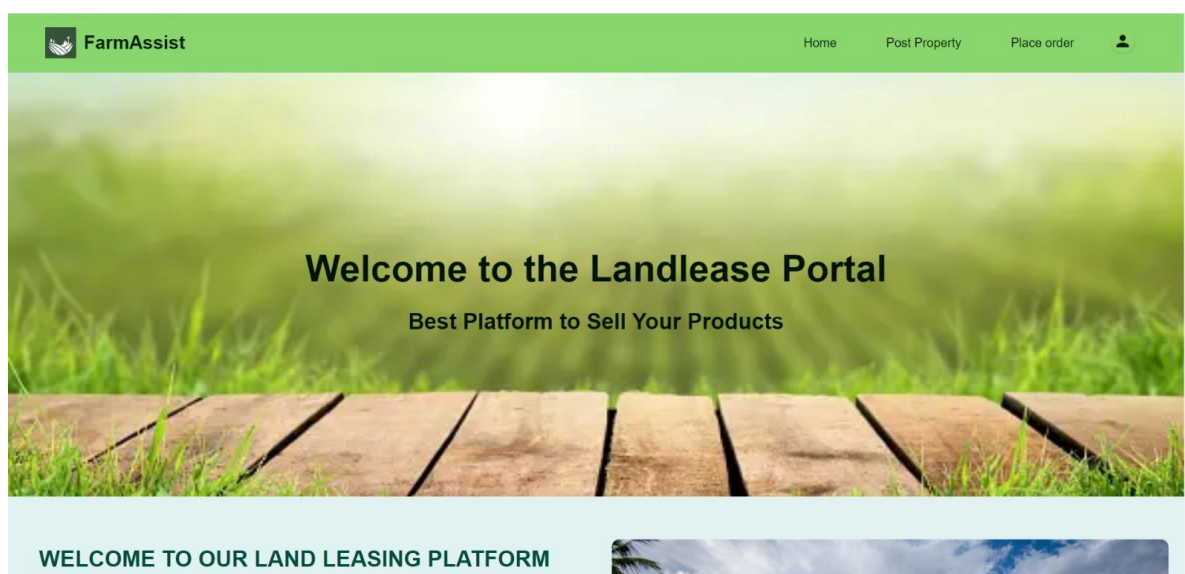
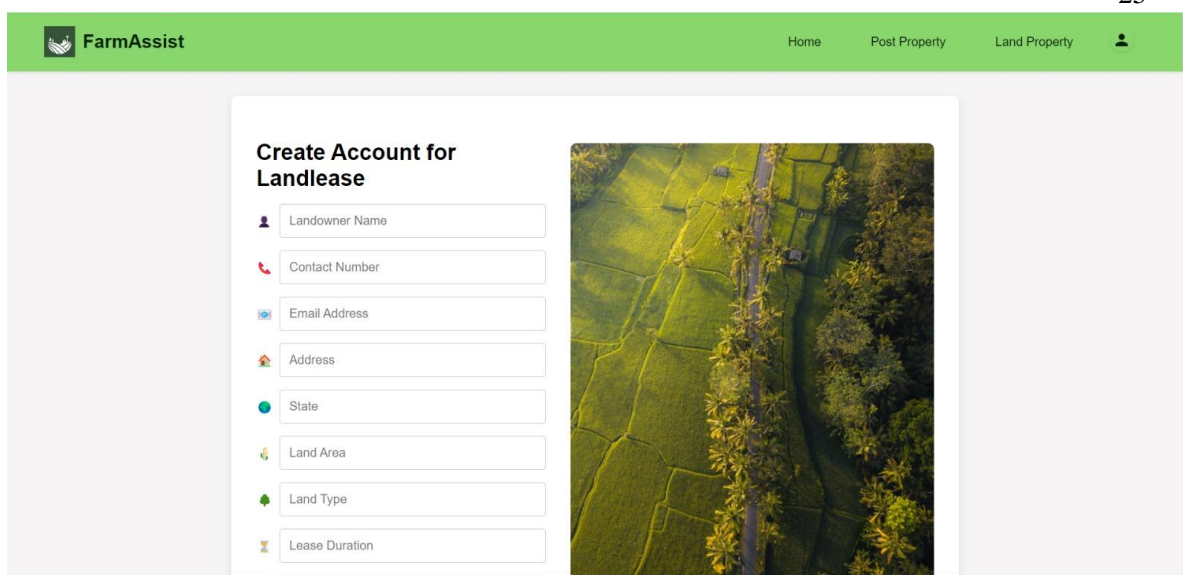


Fig 6.12 LAND TENANCY HOME PAGE

6.4.1 Landowner registration page

Register as a landowner to list your property for lease. After registration, you'll be directed to the login page where you can access your account and post your property details. Manage your listings and connect with farmers seeking new land opportunities. Start by registering to begin the process.

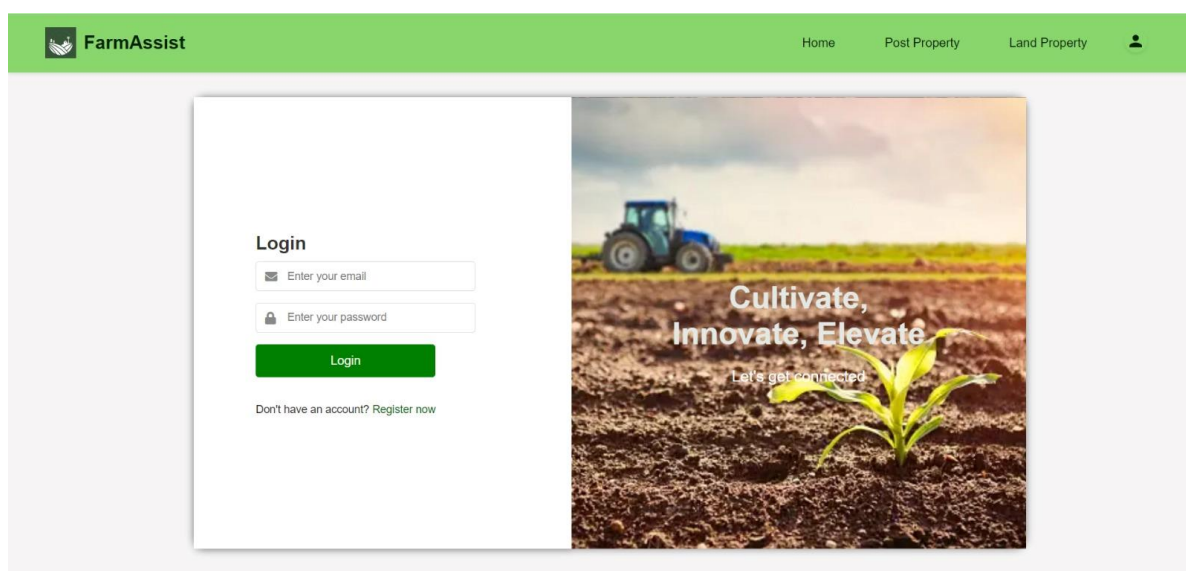


The screenshot shows the 'Create Account for Landlease' form on the FarmAssist website. The form is located on the left side of the page, with a green header bar at the top containing the FarmAssist logo and navigation links: Home, Post Property, Land Property, and a user icon. The form itself has a white background and a title 'Create Account for Landlease'. It contains several input fields, each with a small icon to its left: a person icon for 'Landowner Name', a telephone icon for 'Contact Number', an email icon for 'Email Address', a house icon for 'Address', a location pin icon for 'State', a leaf icon for 'Land Area', a tree icon for 'Land Type', and a clock icon for 'Lease Duration'. To the right of the form is a large, vertical image of a lush green field with a path and trees.

Fig 6.13 LANDOWNER REGISTRATION FORM

6.4.2 Landowner login page

Log in using your email and password to access your landowner account. If your credentials are correct, you will be able to post property details and manage your listings.



The screenshot shows the 'Login' page on the FarmAssist website. The page has a green header bar with the FarmAssist logo and navigation links: Home, Post Property, Land Property, and a user icon. The main content area is white and features a 'Login' form on the left and a large, vertical image of a blue tractor in a field on the right. The form has a title 'Login' and two input fields: 'Enter your email' and 'Enter your password'. Below the fields is a green 'Login' button. At the bottom of the form, there is a link that says 'Don't have an account? Register now'. The image on the right shows a blue tractor in a field with a young plant in the foreground. The text 'Cultivate, Innovate, Elevate' and 'Let's get connected' is overlaid on the image.

Fig 6.14 LANDOWNER LOGIN PAGE

6.4.3 Post property page

On the Post Property page, enter details about your land, including location, size, lease terms, and any additional features. Upload relevant images and provide a clear description to attract potential tenants. Once all information is completed, submit the form to list your property and make it available for lease to farmers.

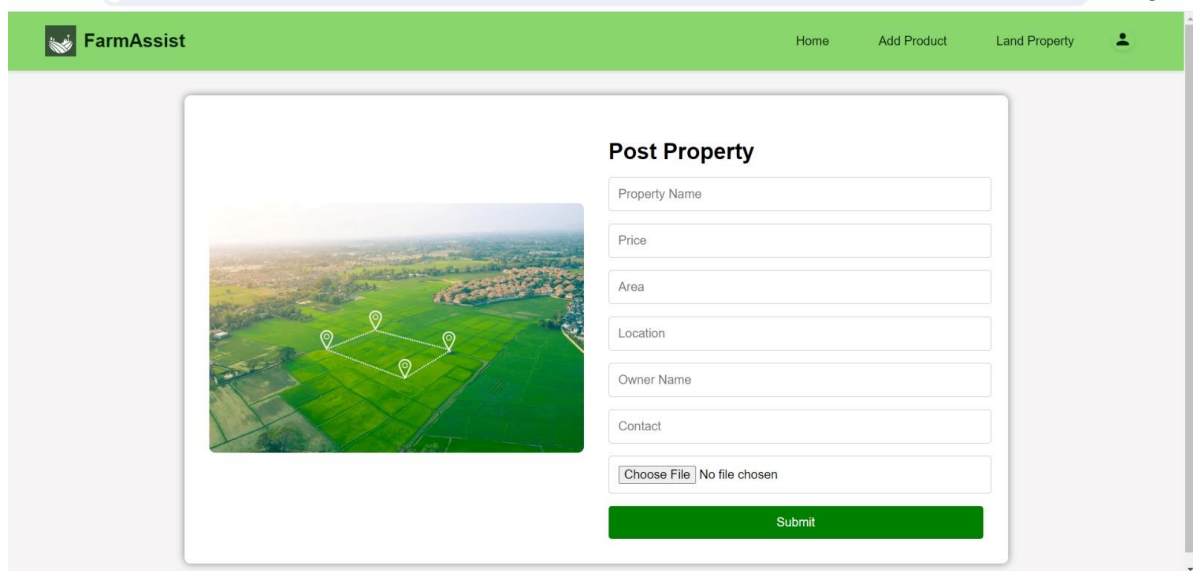
The screenshot shows the 'Post Property' form on the FarmAssist website. The form is titled 'Post Property' and is located on a page with a green header. The header contains the 'FarmAssist' logo, a 'Home' link, an 'Add Product' link, a 'Land Property' link, and a user profile icon. The form itself is a white box with a green border. It contains a large image placeholder on the left showing a green field with location pins. To the right of the image are several input fields: 'Property Name', 'Price', 'Area', 'Location', 'Owner Name', and 'Contact'. Below these fields is a file upload section with a 'Choose File' button and the text 'No file chosen'. At the bottom of the form is a green 'Submit' button.

Fig 6.15 POST PROPERTY PAGE

6.4.4 Property listing page

On the Property Listing page, browse available land for lease, view detailed information including location, size, and lease terms, and see images of each property. Use search filters to narrow down options based on your preferences. Click on individual listings to learn more and contact landowners directly to discuss lease opportunities.

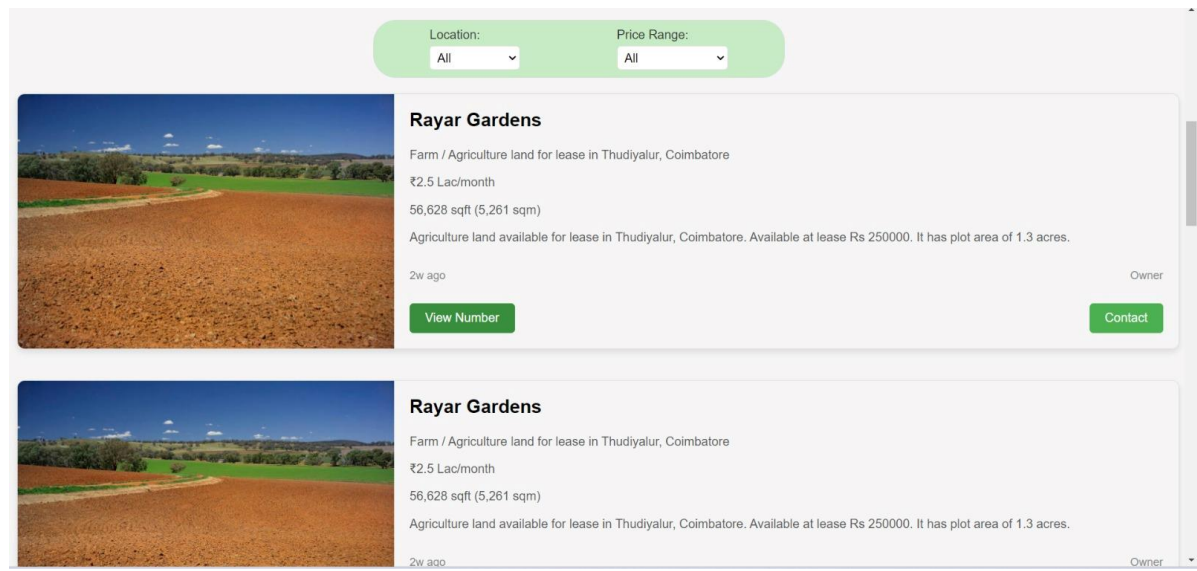


Fig 6.16 LAND LISTING PAGE

CODING

6.5.1 FRONTEND

HOME PAGE:

```
import React from 'react';
import { useNavigate } from 'react-router-dom';
import { Link } from 'react-router-dom';
import { Carousel } from 'react-responsive-carousel';
import "react-responsive-carousel/lib/styles/carousel.min.css";
import './Homepage.css';
const Homepage = () => {
  const navigate = useNavigate();
  const handleButtonClick = () => {
    navigate('/Register-seller');
  };
  return (
    <div>
```

```

<div className="carousel-container">
  <Carousel showArrows={true} autoPlay={true} infiniteLoop={true}>
    <div className="carousel-slide slide1">
      <div className="carousel-text">
        <h3>From Farm Fields to Market Shelves</h3>
      </div>
    </div>
    <div className="carousel-slide slide2">
      <div className="carousel-text">
        <h3>Your Land, Their Future: A Harvest of Opportunities</h3>
      </div>
    </div>
    <div className="carousel-slide slide3">
      <div className="carousel-text">
        <h3>Where Farming Meets Profitability</h3>
      </div>
    </div>
  </Carousel>
</div>
<div className="goals">
  <h3><strong>OUR GOALS</strong></h3>
  <p>We connect fresh produce farmers to businesses through a collaborative
approach.</p>
<div className="grid-text-center">

```

```
<div className="cont">
```

```
  <div className="count-box">01</div>
```

```
  <h3>Vision</h3>
```

```
  <p>The mission of the FarmAssist is to empower farmers by creating a direct link between them and vendors.
```

```
    We aim to eliminate transportation costs and other intermediaries, allowing farmers to sell their goods directly.
```

```
    Additionally, we support landless farmers by helping them find available land for lease, connecting them directly with landowners.
```

```
    Our mission extends to financial assistance, where donations from people are transformed into essential agricultural products for farmers.</p>
```

```
</div>
```

```
<div className="cont">
```

```
  <div className="count-box">02</div>
```

```
  <h3>Mission</h3>
```

```
  <p>Our mission is to empower small farmers to produce healthy, nutritious crops for their local communities and beyond.
```

```
    We aim to create sustainable initiatives and long-term partnerships with our farmers, providing constant support and assistance, both financially and technically.
```

```
</p>
```

```
</div>
```

```
<div className="cont">
```

```
  <div className="count-box">03</div>
```

```
  <h3>Values</h3>
```

```
  <p>
```

```
    The FarmAssist operates with a clear purpose to empower farmers and enhance their livelihoods.
```

```
    Our core values drive our actions every day. We believe in equitable access to fresh, locally sourced food.
```

We champion economic, social, and environmental justice.

Compassion and support guide our interactions, while transparency and accountability underpin our operations.

We foster collaboration within our community and continuously seek innovative solutions.

Together, we're building a thriving ecosystem.

</p>

</div>

</div>

</div>

<div className="content-section">

<h1 className="content-title">Faster growth starts with FarmAssist</h1>

<div className="content-body">

<div className="text-content farmers-box">

<h2 className="sub-title">FOR FARMERS</h2>

<p className="content-text">

Earn more by growing cash crops to meet market demand and our buyer specifications.

Get expert agronomic and managerial support, access inputs and farm services, and receive reliable data to help you farm better. </p>

<button className="rounded-button" onClick={handleButtonClick}>Farm the Better Way!</button>

</div>

<div className="image-content1" />

</div>

<div className="content-body buyers-content">

```

<div className="text-content farmers-box">
  <h2 className="sub-title">FOR BUYERS</h2>
  <p className="content-text">
    Get everything you need for seamless crop procurement from order to
    fulfilment all in one place.
    Gain access to a network of qualified farmers, uncovering new business
    opportunities.
  </p>
  <Link to="/BuyerRegister" className="rounded-button">Source quality
  crops</Link> { /* Rounded button */ }
</div>
<div className="image-content2" />
</div>
<div className="content-body buyers-content">
  <div className="text-content farmers-box">
    <h2 className="sub-title">LAND TENANCY</h2>
    <p className="content-text">
      Secure the ideal land for your farming operations with ease. Our platform
      simplifies the land leasing process,
      providing you with a range of options that match your specific requirements.
      Connect directly with landowners and explore diverse land opportunities to
      enhance your farming capabilities and boost productivity.
    </p>
    <Link to="/land" className="rounded-button">Land Connect</Link>
  </div>

```

```

<div className="image-content3" />
    </div>
  </div>
</>
);
};
export default Homepage;

```

Farmer Registration page:

```

import React, { useState } from 'react';
import { useNavigate, Link } from 'react-router-dom';
import axios from 'axios';
import './RegisterSeller.css';
const RegisterSeller = () => {
  const [formData, setFormData] = useState({
    name: "",
    mobile: "",
    aadhar: "",
    email: "",
    address: "",
    city: "",
    district: "",
    state: "",
    createPassword: ""
  });
  const [error, setError] = useState("");

```



```
const navigate = useNavigate();
const handleChange = (e) => {
  const { name, value } = e.target;
  setFormData({
    ...formData,
    [name]: value
  });
};
const handleSubmit = async (e) => {
  e.preventDefault();
  setError("");
  try {
    const response = await axios.post('http://localhost:9001/api/sellers/signup', {
      name: formData.name,
      mobile: formData.mobile,
      aadhar: formData.aadhar,
      email: formData.email,
      address: formData.address,
      city: formData.city,
      district: formData.district,
      state: formData.state,
      password: formData.createPassword
    });
    console.log('User registered:', response.data);
    navigate('/Login');
  } catch (error) {
```

```

console.error('There was an error registering the user!', error);
    if (error.response && error.response.data) {
        const errorMsg = error.response.data.error || 'Registration failed. Please try
again.';
        setError(errorMsg);
    } else {
        setError('Registration failed. Please try again.');
```

```

    }
}
};

```

```

return (

```

```

    <div className="register-seller-page">
        <h1>Create Account to get started</h1>
        <form onSubmit={handleSubmit} className="register-seller-form">
            <div className="form-group">
                <h3>Seller Registration Form</h3>
                <div className="login-link">
                    <span>Already have an account? </span>
                    <Link to="/login">Login</Link>
                </div>
            </div>
            </div>
            { /* Input fields */ }
            { Object.keys(formData).map((key) => (
                <div className="form-group" key={key}>
                    <label htmlFor={key}>{key.charAt(0).toUpperCase() +
key.slice(1)}</label>

```

```

<input>
  type={key === 'createPassword' ? 'password' : 'text'}
  id={key}
  name={key}
  value={formData[key]}
  onChange={handleChange}
  required
/>
</div>
))}
{error && <p className="error-message">{error}</p>}
<button type="submit" className="submit-button">Register</button>
</form>
</div>
);
};

export default RegisterSeller;

```

FARMER LOGIN PAGE:

```

import React, { useState } from 'react';
import { useNavigate } from 'react-router-dom';
import axios from 'axios';
import './Login.css';
const Login = () => {
  const [formData, setFormData] = useState({

```

```

email: ",
    password: "
  });
const [errorMessage, setErrorMessage] = useState("");
const navigate = useNavigate();
const handleChange = (e) => {
  const { name, value } = e.target;
  setFormData({
    ...formData,
    [name]: value
  });
};
const handleSubmit = async (e) => {
  e.preventDefault();
  try {
    const response = await axios.post('http://localhost:9001/api/sellers/login',
    formData);
    console.log('Login successful:', response.data);
    navigate('/Categories'); // Navigate to the landing page on successful login
  } catch (error) {
    if (error.response && error.response.data) {
      // Set specific error message from the server
      setErrorMessage(error.response.data.error || 'Login failed');
    } else {

```

```

setErrorMessage('An error occurred');
    }
    }
};

return (
  <div className="login-page">
    <div className="left-container">
      { /* Add any additional content or image here */ }
    </div>
    <div className="right-container">
      <div className="header-text">
        <h2>Farm from anywhere.</h2>
        <p>Everything you need in one place.</p>
      </div>
      <div className="form-wrapper">
        <h1>Login</h1>
        { errorMessage && <div className="error-message">{ typeof errorMessage
=== 'object' ? JSON.stringify(errorMessage) : errorMessage } </div> }
        <form onSubmit={handleSubmit}>
          <div className="form-group">
            <label htmlFor="email">Email</label>
            <input
              type="email"
              id="email"
              name="email"

```

```

    value={formData.email}
      onChange={handleChange}
      required
    />
  </div>
  <div className="form-group">
    <label htmlFor="password">Password</label>
    <input
      type="password"
      id="password"
      name="password"
      value={formData.password}
      onChange={handleChange}
      required
    />
  </div>
  <button type="submit" className="submit-button">Login</button>
</form>
</div>
</div>
</div>
);
};

```

export default Login

ADD PRODUCT PAGE:

```

import React, { useState } from 'react';
import { useParams, useNavigate } from 'react-router-dom';

```

```
import axios from 'axios';
import './AddProduct.css';
const AddProduct = () => {
  const { category } = useParams();
  const navigate = useNavigate();
  const [formData, setFormData] = useState({
    category: category || 'vegetables',
    productName: "",
    quantity: "",
    price: "",
    description: "",
    imageUrl: ""
  });
  const handleChange = (e) => {
    setFormData({
      ...formData,
      [e.target.name]: e.target.value
    });
  };
  const handleSubmit = (e) => {
    e.preventDefault();
    axios.post('http://localhost:9001/api/AddProduct', formData)
      .then((response) => {
        navigate('/ProductDashboard');
      })
      .catch((error) => {
```

```

console.error("There was an error adding the product!", error);
    });
};
return (
    <div className="add-products-form-page">
        <h1>Add Product Details</h1>
        <form onSubmit={handleSubmit} className="add-products-form">
            <div className="form-group">
                <label htmlFor="category">Category</label>
                <select id="category" name="category" value={formData.category}
onChange={handleChange} required>
                    <option value="vegetables">Vegetables</option>
                    <option value="fruits">Fruits</option>
                    <option value="cereals">Cereals</option>
                </select>
            </div>
            <div className="form-group">
                <label htmlFor="product-name">Product Name</label>
                <input type="text" id="product-name" name="productName"
value={formData.productName} onChange={handleChange} required />
            </div>
            <div className="form-group">
                <label htmlFor="quantity">Quantity (in kgs)</label>
                <input type="number" id="quantity" name="quantity"
value={formData.quantity} onChange={handleChange} step="1" required />
            </div>
            <div className="form-group">

```



```

<label htmlFor="price">Price</label>
    <input type="number" id="price" name="price" value={formData.price}
onChange={handleChange} required />
</div>
<div className="form-group">
    <label htmlFor="description">Description</label>
    <textarea id="description" name="description"
value={formData.description} onChange={handleChange} required></textarea>
</div>
<div className="form-group">
    <label htmlFor="image-url">Image URL</label>
    <input type="text" id="image-url" name="imageUrl"
value={formData.imageUrl} onChange={handleChange} required />
</div>
<button type="submit" className="submit-button">Add Product</button>
</form>
</div>
);
};
export default AddProduct;

```

PRODUCT LISTING PAGE

```

import React, { useState, useEffect } from 'react';
import axios from 'axios';
import './ProductListPage.css'; // Add your styles
const ProductListPage = () => {
    const [products, setProducts] = useState([]);

```

```

useEffect(() => {
  // Fetch products from backend
  axios.get('http://localhost:9001/api/products') // Adjusted to match your
backend port
  .then(response => {
    setProducts(response.data);
  })
  .catch(error => {
    console.error("There was an error fetching the products!", error);
  });
}, []);
return (
  <div className="product-list-page">
    <h1>Product List</h1>
    <div className="product-list-container">
      {products.map(product => (
        <div key={product.id} className="product-card">
          <img src={http://localhost:9001/uploads/${product.imageUrl}}
alt={product.productName} />
          <h2>{product.productName}</h2>
          <p>Category: {product.category}</p>
          <p>Price: ${product.price}</p>
          <p>Quantity: {product.quantity}</p>
          <p>Description: {product.description}</p>
          <p>Seller: {product.farmerName}</p>
          <p>Contact: {product.contact}</p>
        </div>

```

```

    ))}
  </div>
</div>

);
};
export default ProductListPage;

```

6.5.2 BACKEND

APPLICATION.PROPERTIES

```

spring.application.name=FarmAssistProj
server.port=9001
spring.datasource.url=jdbc:mysql://localhost:3306/FarmAssistProject
spring.datasource.username=root
spring.datasource.password=dhanushree1234
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true
spring.servlet.multipart.enabled=true
spring.servlet.multipart.max-file-size=10MB
spring.servlet.multipart.max-request-size=10MB
spring.mvc.static-path-pattern=/uploads/**
spring.resources.static-
locations=file:/C:/Users/Dhanushree/OneDrive/Desktop/demo/my-
app/src/images/

```

MODEL

```
package com.FarmAssistProj.FarmAssistProj.model;

import jakarta.persistence.Entity;
import jakarta.persistence.GeneratedValue;
import jakarta.persistence.GenerationType;
import jakarta.persistence.Id;
import jakarta.persistence.Table;

@Entity
@Table(name = "Add_Product_details")
public class AddProductFormModel {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    private String productName;
    private int quantity;
    private double price;
    private String description;
    private String imageUrl;
    private String category;
    private String farmerName;
    private String contact; //

    public String getFarmerName() {
        return farmerName; }

    public void setFarmerName(String farmerName) {
        this.farmerName = farmerName; }

    public String getContact() {
        return contact; }
```

```
public void setContact(String contact) {  
    this.contact = contact; }  
  
public Long getId() {  
    return id; }  
  
public void setId(Long id) {  
    this.id = id; }  
  
public String getCategory() {  
    return category; }  
  
public void setCategory(String category) {  
    this.category = category; }  
  
public String getProductName() {  
    return productName; }  
  
public void setProductName(String productName) {  
    this.productName = productName; }  
  
public int getQuantity() {  
    return quantity; }  
  
public void setQuantity(int quantity) {  
    this.quantity = quantity; }  
  
public double getPrice() {  
    return price; }  
  
public void setPrice(double price) {  
    this.price = price; }  
  
public String getDescription() {  
    return description; }  
  
public void setDescription(String description) {  
    this.description = description; }
```

```

public String getImageUrl() {
    return imageUrl; }

public void setImageUrl(String imageUrl) {
    this.imageUrl = imageUrl;
}
}

```

CONTROLLER

```

package com.FarmAssistProj.FarmAssistProj.controller;
import com.FarmAssistProj.FarmAssistProj.model.AddProductFormModel;
import com.FarmAssistProj.FarmAssistProj.service.AddProductFormService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.*;
import org.springframework.web.multipart.MultipartFile;
import java.util.List;

```

```
@RestController
```

```
@CrossOrigin(origins = "http://localhost:3000")
```

```
@RequestMapping("/api")
```

```
public class AddProductFormController {
```

```
    @Autowired
```

```
    private AddProductFormService addProductFormService;
```

```
    @PostMapping("/AddProduct")
```

```
    public ResponseEntity<AddProductFormModel> addProduct(
```

```
        @RequestParam("category") String category,
```

```
        @RequestParam("productName") String productName,
```

```

        @RequestParam("quantity") int quantity,
        @RequestParam("price") double price,
        @RequestParam("description") String description,
        @RequestParam("imageFile") MultipartFile imageFile,
        @RequestParam("farmerName") String farmerName,
        @RequestParam("contact") String contact) {
    AddProductFormModel product = addProductFormService.saveProduct(
        category, productName, quantity, price, description, imageFile,
        farmerName, contact);
    return ResponseEntity.ok(product);
}

@GetMapping("/products")
public ResponseEntity<List<AddProductFormModel>> getAllProducts() {
    List<AddProductFormModel> products = addProductFormService.findAll();
    return ResponseEntity.ok(products);
}
}

```

REPOSITORY

```

package com.FarmAssistProj.FarmAssistProj.repository;

import com.FarmAssistProj.FarmAssistProj.model.AddProductFormModel;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.stereotype.Repository;

@Repository
public interface AddProductFormInterface extends
    JpaRepository<AddProductFormModel, Long> {
}

```

CHAPTER 7

CONCLUSION

This chapter tells about the conclusion that anyone can drive from the project and the learning we learnt by taking over this project.

7.1 CONCLUSION

The completion of the FarmAssist website represents a major advancement in connecting farmers directly with end users, including wholesalers, retailers, food processing industries, and cloud kitchens. This platform allows for the seamless purchase of fresh produce, enhancing farmer profitability through increased transparency and the removal of intermediaries.

FarmAssist facilitates a straightforward and efficient marketplace where buyers can access a variety of high-quality products directly from farmers. By cutting out middlemen, we ensure that farmers receive a fairer share of the profits, leading to improved income and sustainability for their operations. Additionally, the website offers a unique feature for landowners, allowing them to list available properties for lease. This functionality enables farmers to find suitable land for their needs, fostering new opportunities for agricultural expansion and development.

We have designed FarmAssist with a focus on user-friendliness and security, incorporating the latest technologies and standards to provide a reliable and intuitive experience for all users. We are excited about the upcoming launch and confident that FarmAssist will drive significant benefits to the agricultural community, improving connections and opportunities for both farmers and end users.

7.2 FUTURE SCOPE

Looking ahead, FarmAssist has several exciting opportunities for growth and enhancement. We plan to expand our market reach by integrating features that cater to international markets, allowing farmers to connect with a broader range of buyers. Advanced analytics will be introduced to provide users with valuable insights into market trends and pricing strategies. We will also enhance connectivity by adding real-time communication tools, such as chat and video calls, to facilitate better interactions between farmers and buyers. Sustainability will be a key focus, with the addition of metrics and certifications to promote eco-friendly practices. Additionally, a mobile app will be developed to offer users greater accessibility and convenience. Finally, we will improve land leasing options with advanced search filters and provide farm management tools to support operational optimization. These advancements will strengthen FarmAssist's role as a leading platform in the agricultural sector.

REFERENCES

- FreeCodeCamp, “The Beginner’s Guide to React”
<https://www.freecodecamp.org/news/the-beginners-guide-to-react-9be65f50a55c/#:~:text=React%20is%20a%20JavaScript%20library,many%2C%20many%20other%20software%20companies.>
- Baeldung, “Spring Boot with MySQL”
<https://www.baeldung.com/spring-boot-mysql-tls>
- Spring.io, “Spring Boot Reference Documentation”
<https://docs.spring.io/spring-boot/docs/2.7.0/reference/htmlsingle/>
- Tutorialspoint “How to Build an E-commerce Website with React.js”
<https://www.tutorialspoint.com/build-ecommerce-website-like-amazon-react-amp-node-amp-mongodb/index.asp>
- MKisan, “Government Portal for Farmer Services”
<https://mkisan.gov.in/Home/Downloads>