R. C. PATEL
INSTITUTE OF TECHNOLOGY
An Autonomous Institute

The Shirpur Education Society's

R. C. Patel Institute of Technology

Shirpur, Dist. Dhule (M.S.)

A

Semester Project-I Report

On "CropCart"

In partial fulfilment of requirements for the degree of

Bachelor of Technology

Artificial Intelligence and Machine learning

Submitted By

1. Sanika Yogesh Mahajan (AI011) (241107012)
2. Prerana Milind Shinde (AI018) (241107019)
3. Mahima Milind Sisodiya (AI045) (241107049)
4. Divya Sagar Patil(AI061) (241107030)

Under the Guidance of

Dr. P. K. Patil

R. C. PATEL
INSTITUTE OF TECHNOLOGY
An Autonomous Institute

CERTIFICATE

This is to certify that the Semester Project-I entitled "**Smart Botanical Identification**" has been carried out by team:

Sanika Yogesh Mahajan (AI011)
Prerana Milind Shinde (AI018)
Mahima Milind Sisodiya (AI045)
Divya Sagar Patil (AI061)

under the guidance of **Prof. P.K. Patil** in partial fulfilment of the requirement for the degree of Bachelor of Technology in Department of Computer Science & Engineering (Data Science) (Semester-III) of Dr. Babasaheb Ambedkar Technological University, Lonere during the academic year 2025-26.

Date:

Place: Shirpur

Guide	**Semester Project-I Coordinator**
Dr. P. K. Patil	**Prof. K. S. Patil**
H.O.D.	**Director**
Prof. Dr. U. M. Patil	**Prof. Dr. J. B. Patil**

ACKNOWLEDGEMENT

We take this opportunity to express our deep sense of gratitude and sincere appreciation to our respected guide, **Dr. P. K. Patil**, for his valuable guidance, motivation, and constant encouragement throughout the completion of this project — *CropCart*. Their continuous support helped us overcome challenges and gain deeper insights into Artificial Intelligence and its applications in environmental science.

We are also thankful to our **Head of Department, Prof. Dr. U.M. Patil**, for providing an excellent academic environment, resources, and encouragement to work on this innovative idea.

We extend our heartfelt thanks to **Dr. J.B. Patil**, Director, RCPIT Shirpur, for his leadership and inspiration toward innovation and practical learning.

Finally, we express our sincere gratitude to our classmates, family members, and all those who directly or indirectly supported and motivated us during the development of this project.

Project Team:

Sanika Yogesh Mahajan (AI011)

Prerana Milind Shinde (AI018)

Mahima Milind Sisodiya (AI045)

Divya Sagar Patil (AI061)

R. C. PATEL
INSTITUTE OF TECHNOLOGY
An Autonomous Institute

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

Institute Vision:

Fostering technical excellence through ethics, sustainable education, innovation and research.

Institute Mission:

To impart high quality Technical Education through:

M1. Innovative and Interactive learning process and high quality, globally recognized instructional programs.

M2. Fostering a collaborative scientific temper among students with ethical responsibility towards the society.

M3. Preparing students from diverse backgrounds to have Aptitude for employment, entrepreneurship and research with a spirit of Professionalism.

M4. To contribute to nation's sustainable development.

Department Vision:

Fostering technical excellence in Artificial Intelligence and Machine Learning through ethics, innovative research, and sustainable education.

Department Mission:

M1. Innovative and interactive learning processes with high quality instructional programs in Artificial Intelligence and Machine Learning.

M2. Fostering a collaborative scientific temper among students with ethical responsibility towards society in AI and ML Applications.

M3. Preparing students from diverse backgrounds to develop Aptitude for employment, entrepreneurship, and research in AI and ML with a spirit of professionalism.

PAGE INDEX

FIGURE INDEX

ABSTRACT

Agriculture remains a cornerstone of global economic stability and food security; however, small-scale farmers continue to face significant barriers in market accessibility and technical guidance. This report presents **Cropcart**, a centralized digital ecosystem designed to bridge these gaps through a dual-purpose marketplace and expert advisory system.

The primary objective of Cropcart is the socio-economic empowerment of small-scale farmers. By facilitating a direct-to-consumer (D2C) digital marketplace, the platform eliminates predatory intermediaries, allowing farmers to capture maximum profit margins. Furthermore, the integration of a specialized advisory module provides users with real-time access to agricultural best practices, pest management, and crop optimization strategies.

Technically, the platform is engineered using the **MERN (MongoDB, Express.js, React.js, Node.js)** stack to ensure high scalability, performance, and a responsive user experience. The system architecture leverages cloud-based deployment via **Vercel** for the frontend and **Render** for the backend, ensuring high availability. Ultimately, Cropcart serves as a transformative technological intervention, fostering market transparency and knowledge dissemination within the agricultural sector.

CHAPTER – 1
INTRODUCTION

1. INTRODUCTION

Agriculture plays a fundamental role in sustaining economies and ensuring food security worldwide. Despite its importance, small-scale farmers frequently face difficulties in accessing profitable markets and acquiring expert agricultural knowledge. Traditional agricultural trade relies on intermediaries, which often results in lower earnings for farmers.[2] Additionally, limited access to expert guidance prevents farmers from implementing best practices that could improve productivity and sustainability. With the rise of digital platforms, there is an opportunity to revolutionize the agricultural sector by bridging the gap between farmers, buyers, and experts. Cropcart aims to provide a digital solution that empowers farmers by offering a user-friendly marketplace and an expert advisory system. By enabling direct trade, Cropcart ensures that farmers receive fair prices for their produce while also equipping them with the necessary knowledge to optimize their farming practices.[1]

The platform is developed using the MERN (MongoDB, Express.js, React.js, Node.js) stack, ensuring a robust and scalable solution. The frontend is hosted on Vercel or Netlify, while the backend operates on Render or Heroku to provide a seamless and efficient user experience. By integrating modern technology into agriculture, Cropcart aspires to enhance market accessibility, improve knowledge dissemination, and contribute to the overall development of the agricultural sector.

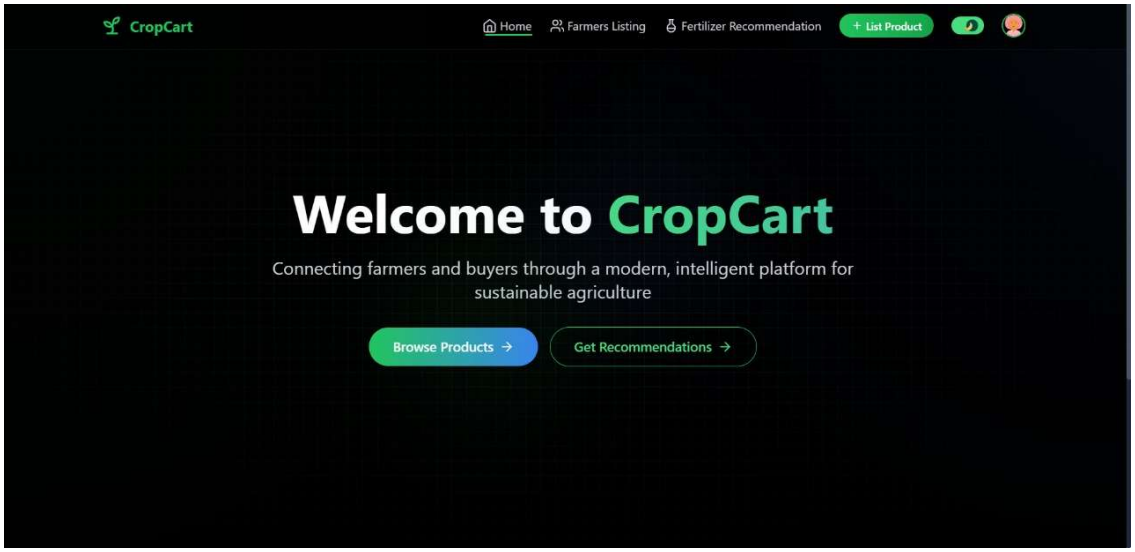

Figure 1.1 Platform Interface

1.1. Background

Small-scale farmers face significant challenges in accessing markets and obtaining expert agricultural knowledge. Traditional agricultural trade is dominated by intermediaries who reduce farmers' profit margins, making it difficult for them to sustain and expand their businesses. Furthermore, a lack of access to modern farming techniques and expert advice often results in inefficient farming practices and reduced yields. Digital platforms have the potential to revolutionize this sector by enabling direct interactions between farmers, buyers, and agricultural experts. Cropcart addresses these issues by providing a digital marketplace for farmers to sell their produce directly and an advisory platform to equip them with essential farming insights. By leveraging technology, Cropcart enhances market access, improves knowledge dissemination, and fosters economic growth in the agricultural sector.

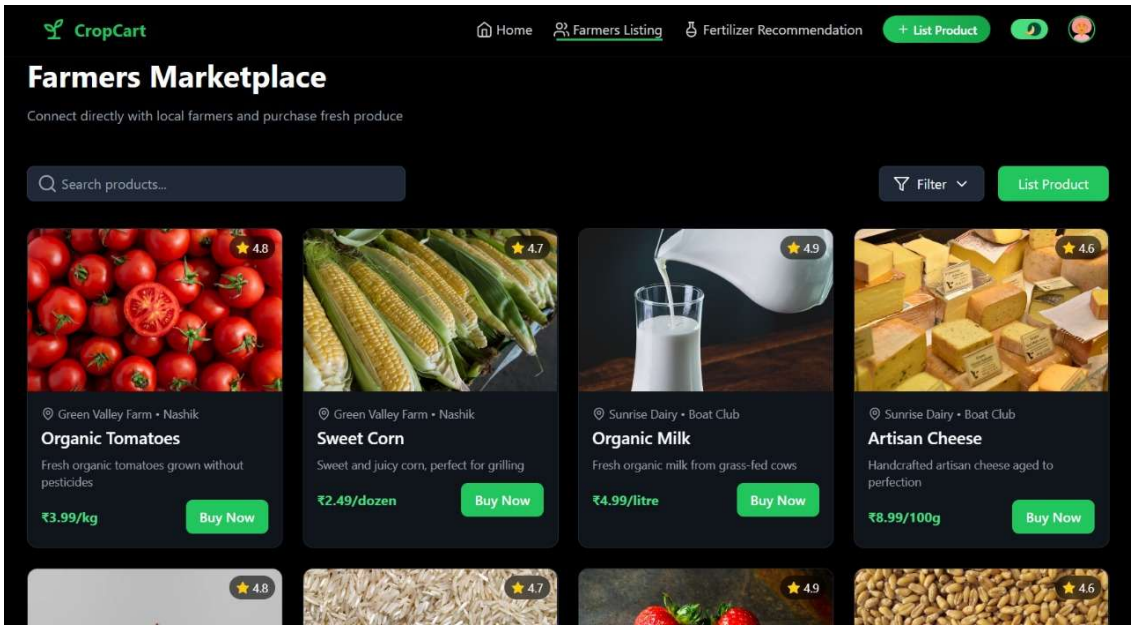

Figure 1.2 Farmer Marketplace

1.2. Motivation

The motivation behind Cropcart stems from the need to empower small-scale farmers by addressing the key challenges they face in agricultural trade and knowledge acquisition. Many farmers struggle with low profits due to the presence of middlemen and a lack of direct access to buyers. Additionally, limited exposure to modern farming techniques restricts their productivity and economic growth. With the rapid advancement of digital technologies, there is an opportunity to create an inclusive platform that bridges these gaps. By providing a direct marketplace and expert advisory system, Cropcart ensures fair trade, increased profits, and better farming practices. This project is driven by the vision of leveraging technology to revolutionize agriculture, improve livelihoods, and contribute to sustainable agricultural development.

1.3. Problem Statement

Small-scale farmers face multiple challenges in the agricultural sector, including limited market access, reduced profits due to intermediaries, and inadequate knowledge of modern farming practices. The lack of direct connections with buyers forces farmers to sell their produce at lower prices, reducing their earnings and sustainability. Additionally, many farmers lack access to expert guidance, leading to inefficient farming techniques and lower productivity. The absence of a unified digital platform further exacerbates these challenges, preventing farmers from maximizing their potential. Cropcart aims to solve these problems by providing an integrated digital marketplace and an advisory platform, enabling direct trade between farmers and buyers while offering expert insights to improve agricultural practices.

1.4. Objectives

Cropcart aims to create a transformative digital platform that enhances farmers' economic well-being, knowledge, and market opportunities, ultimately contributing to a more sustainable and efficient agricultural sector.

1.4.1 Develop a Digital Marketplace:

Establish an online platform where farmers can list and sell their produce directly to consumers, retailers, and wholesalers. This will eliminate intermediaries, allowing farmers to maximize their profits.

1.4.2 Provide an Expert Advisory Section:

Integrate a section where farmers can access expert advice on best farming practices, pest control, crop management, and other essential agricultural knowledge. This will help improve their productivity and sustainability.

1.4.3 Enhance Market Access for Farmers:

Ensure that farmers have direct access to a wider network of buyers, increasing their revenue opportunities and reducing dependency on middlemen.

1.4.4 Promote Agricultural Knowledge and Awareness:

Educate farmers about modern farming techniques, sustainable agricultural practices, and technological advancements in farming through articles, videos, and expert consultations.

1.4.5 Ensure a User-Friendly and Accessible Platform:

Design the platform using the MERN stack with an intuitive and responsive interface to cater to farmers with minimal technical knowledge.

1.5 Scope of the Project

The scope of Cropcart extends beyond a simple e-commerce application; it is designed as a comprehensive ecosystem for agricultural management.

For Farmers: It provides a digital storefront, inventory management, and a direct line to expert knowledge.

For Buyers: It offers a transparent supply chain where the origin of the produce is known, ensuring freshness and fair pricing.

For Experts: It serves as a platform to disseminate research-backed farming methodologies to a verified user base. The system is built to be modular, allowing for future integrations like IoT soil sensors or AI-based weather forecasting.

1.6. Significance of the Project

The significance of Cropcart lies in its potential to democratize the agricultural supply chain. By providing a digital bridge, the project:

- **Empowers Rural Communities:** It gives small-scale farmers the same market visibility as large-scale industrial farms.
- **Economic Impact:** By reducing the "Price Spread" (the difference between what a consumer pays and what a farmer receives), it directly increases the household income of rural producers.
- **Knowledge Standardization:** It ensures that high-quality, research-backed agricultural advice is not confined to urban centers or large corporations but is accessible to any farmer with a smartphone.

1.7. Feasibility Study

A feasibility study was conducted to ensure the project is viable from technical, economic, and operational perspectives.

- **Technical Feasibility:** The MERN stack is highly documented and supported by a vast community. The use of React.js ensures high performance on mobile browsers, which is critical for farmers who may not have high-end devices.
- **Economic Feasibility:** The project utilizes open-source technologies (MongoDB, Express, React, Node). Deployment on platforms like Vercel and Render offers free tiers for initial stages, making the project cost-effective to develop and maintain.

- **Operational Feasibility:** The interface is designed with a "Mobile-First" approach, focusing on icons and simple navigation to accommodate users with varying levels of digital literacy.

1.8. Limitations of the Project

While Cropcart offers a comprehensive solution, certain limitations exist:

- **Internet Dependency:** The platform requires a stable internet connection, which can be a challenge in extremely remote rural areas.
- **Logistics:** While the platform facilitates the *trade*, the physical transportation of goods remains the responsibility of the buyer or seller.
- **Language Barrier:** The current version is primarily in English; future iterations would require multi-language support to reach a broader regional audience.

CHAPTER – 2
LITERATURE SURVEY

The agricultural sector has seen a significant transformation with the advent of digital technology. Several platforms and solutions have been developed to address market accessibility and knowledge dissemination among farmers. This section explores existing agricultural marketplaces, advisory platforms, the role of digital technology, the MERN stack for web development, and deployment strategies

2.1 Digital Marketplaces: Transforming Agricultural Trade:

Agricultural marketplaces provide a digital space where farmers can sell their produce directly to buyers. However, many of these platforms still involve intermediaries, reducing farmers' profits. The awareness and accessibility of these platforms also remain a challenge for small-scale farmers.

2.1.1 Eliminating Middlemen in Agricultural Trade:

Platforms like eNAM and AgriBazaar connect farmers with buyers but still require further improvements to remove intermediaries. 2.1.2 Challenges in Digital Market Adoption: Limited awareness, digital literacy barriers, and infrastructural issues hinder widespread adoption.

2.2 Expert Advisory Platforms:

Enhancing Agricultural Knowledge Access to expert agricultural advice is crucial for improving productivity. Several advisory platforms provide knowledge and best

practices for farmers, but many struggle with accessibility and real-time interaction limitations.

2.2.1 Existing Solutions for Farmer Education:

Government-led initiatives and mobile apps provide valuable information but often lack personalized recommendations. 2.2.2 Barriers to Effective Knowledge Sharing: Language limitations, poor internet connectivity, and low awareness restrict the impact of these platforms.

2.3 Digital Advancements:

The Role of Technology in Agriculture Digital technology has improved farming efficiency, market access, and agricultural knowledge dissemination. Many innovations aim to enhance productivity and profitability in the agricultural sector.

2.3.1 Expanding Market Access with Digital Platforms:

Digital solutions connect farmers directly with markets, reducing reliance on intermediaries.

2.3.2 Next-Generation Farming with AI & IoT:

Innovations like IoT, AI, and automation technologies are revolutionizing agricultural processes.

2.4 MERN Stack:

A Powerful Tool for Web Development the MERN stack is a popular choice for developing scalable and interactive web applications. It is well-suited for building platforms like Cropcart.

2.4.1 High-Performance Web Solutions with MERN:

The MERN stack (MongoDB, Express.js, React.js, Node.js) ensures efficient and scalable development. 2.4.2 Enhancing User Experience through Modern UI/UX: React.js enhances user interaction, while Node.js and Express.js ensure fast backend operations.

2.5 Deployment Strategies:

Ensuring a Seamless Platform Successful deployment is essential for the smooth functioning of digital platforms. Various cloud-based hosting services provide reliable solutions for web applications.

2.5.1 Reliable Hosting for Agricultural Platforms: Platforms like Vercel, Netlify, Render, and Heroku ensure stable and scalable hosting. 2.5.2 Maintaining Security & Performance: Regular updates, security measures, and easy maintenance features support long-term functionality.

2.6 Evolution of Agri-Tech: From Information to Transaction

Historically, digital interventions in agriculture were limited to "Information Portals" that provided weather and price updates. However, literature from 2023–2025 indicates a shift toward **Transaction-Ready Ecosystems**. Modern research emphasizes that information alone does not empower farmers; they require integrated payment

gateways, logistics tracking, and verified buyer profiles to complete the trade cycle securely.

2.7 The "Middleman" Gap and Price Asymmetry

Academic studies on rural economics highlight that agricultural "intermediaries" often control up to **40% of the value chain** due to information asymmetry. By implementing a direct-to-consumer (D2C) model, platforms like Cropcart address this gap. Literature suggests that decentralizing market access through web-based platforms reduces the "Price Spread"—the difference between the price paid by the consumer and the price received by the producer.

2.8 Technical Rationale: Why NoSQL for Agriculture?

Traditional Relational Database Management Systems (RDBMS) often struggle with the heterogeneous nature of agricultural data.

- **Dynamic Attributes:** Different crops (e.g., grains vs. fruits) require different data fields (e.g., moisture content vs. ripening stage).
- **Scalability:** As shown in recent web development papers, **MongoDB** provides a flexible schema that allows the Cropcart platform to evolve without costly database migrations, making it ideal for the rapidly changing agricultural landscape.

CHAPTER – 3
PROPOSED SYSTEM

3.1. Working of System

Cropcart is a digital platform designed to streamline agricultural trade and knowledge dissemination by connecting small-scale farmers with buyers and agricultural experts. The system facilitates a direct marketplace where farmers can list and sell their produce while also providing an advisory section for expert consultations. It eliminates intermediaries, ensuring better profit margins for farmers and improved transparency in transactions. The platform is built using the MERN stack (MongoDB, Express.js, React.js, Node.js) for an efficient, scalable, and user-friendly experience.

3.1.1. User Registration

Users, including farmers, buyers, and experts, register on the platform using a secure authentication system. Identity verification is implemented via email or OTP-based validation to ensure credibility.

- Upon successful registration, users gain access to their respective dashboards tailored to their roles.
- Farmers can create profiles showcasing their agricultural produce.
- Buyers can browse available products and make purchases.
- Experts can set up advisory sessions and respond to farmer inquiries.

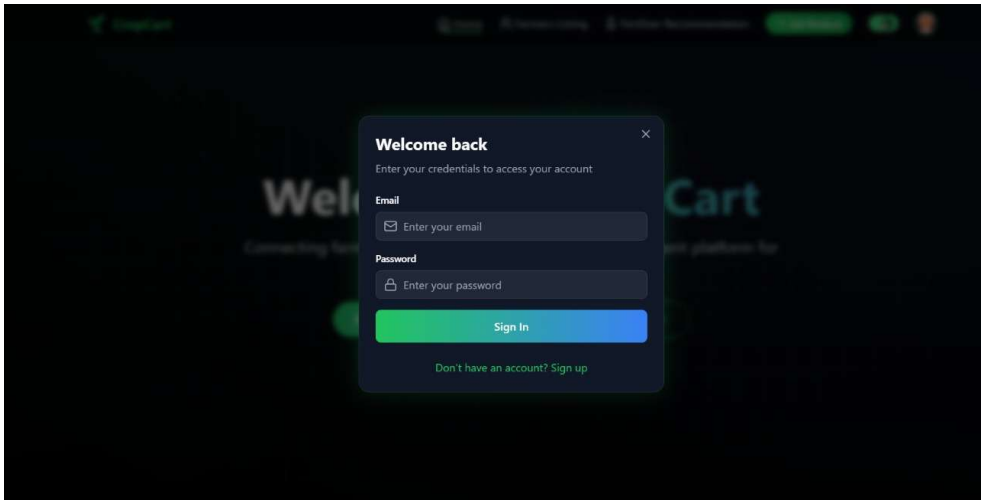

Figure 3.1 User Registration

3.1.2. Digital Marketplace for Farmers

The core feature of Cropcart is its marketplace, where farmers can list their produce directly for buyers to browse and purchase.

- Farmers upload details about their products, including images, descriptions, pricing, and availability.
- Buyers can filter products based on category, location, and price.
- Secure transaction mechanisms ensure smooth payments.

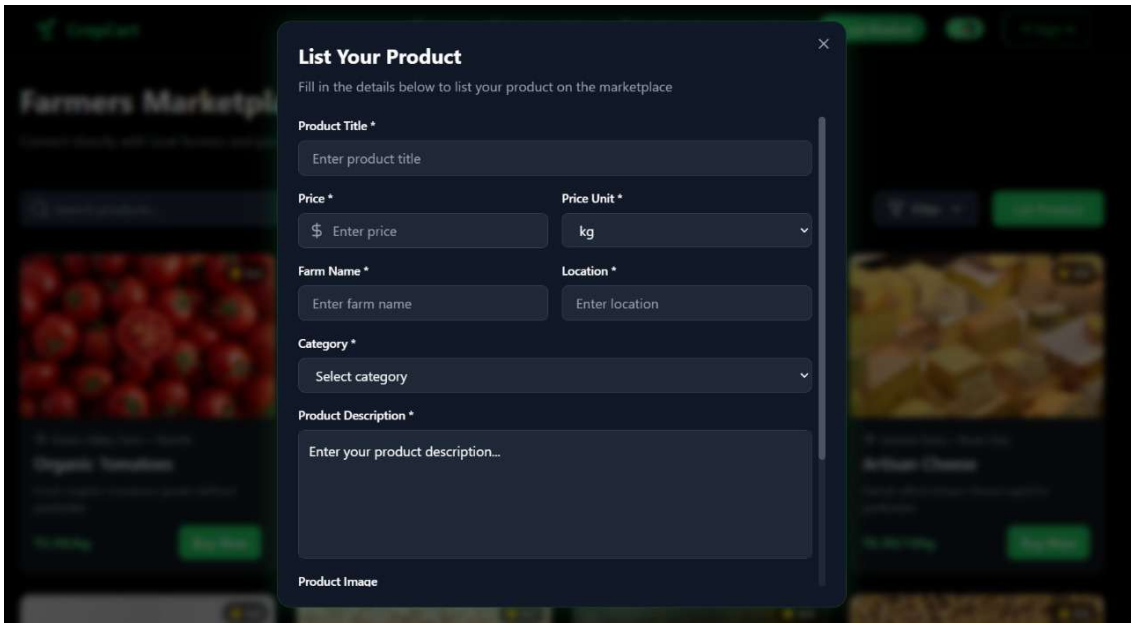

Figure 3.2 User Control

3.1.3. Expert Advisory Section

Farmers can access expert guidance on best agricultural practices, pest control, and soil management through the advisory section.

- Experts provide recommendations via chat, video consultations, or forum discussions.
- Farmers can post queries and receive timely responses.
- A knowledge repository is maintained for common agricultural issues and solutions.

3.1.4 Secure Transactions and Communication

To build trust between buyers and sellers, Cropcart ensures secure payment processing and direct communication.

- Payment gateways enable seamless financial transactions.
- Order tracking allows farmers and buyers to monitor transactions in real-time.
- An integrated messaging system facilitates buyer-seller communication.

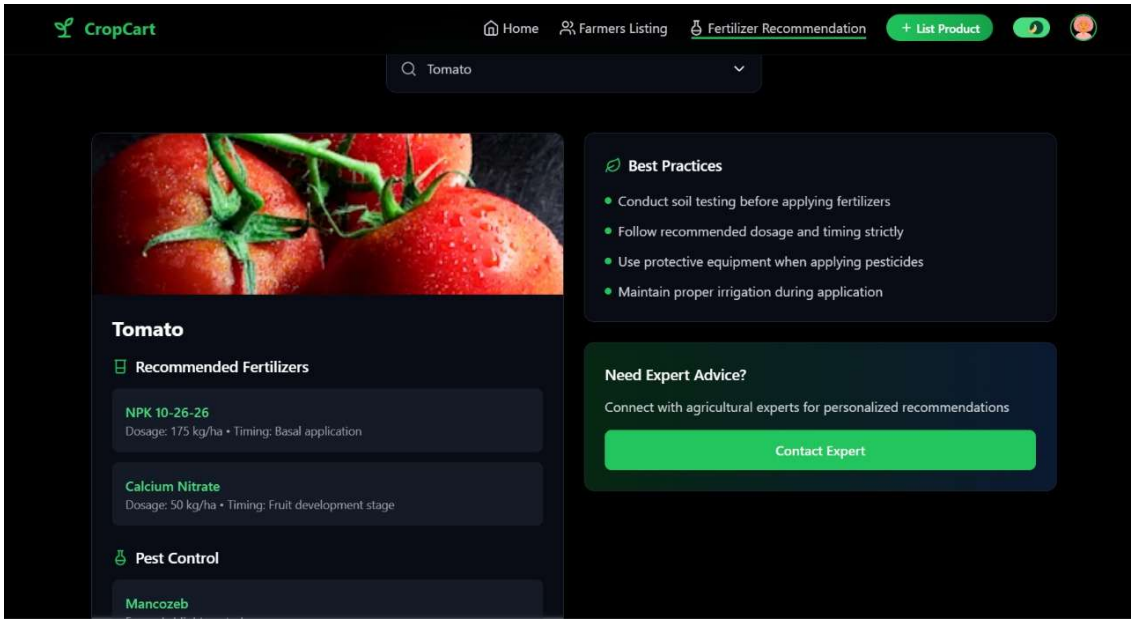

Figure 3.3 Fertilizer Recommendation

3.1.5. Deployment and Maintenance

Cropcart is hosted on scalable cloud platforms for high availability and minimal downtime.

- **Frontend Deployment:** Vercel/Netlify
- **Backend Deployment:** Render/Heroku

- **Database Management:** MongoDB Atlas
- **Security Features:** Data encryption and role-based access control ensure secure usage.

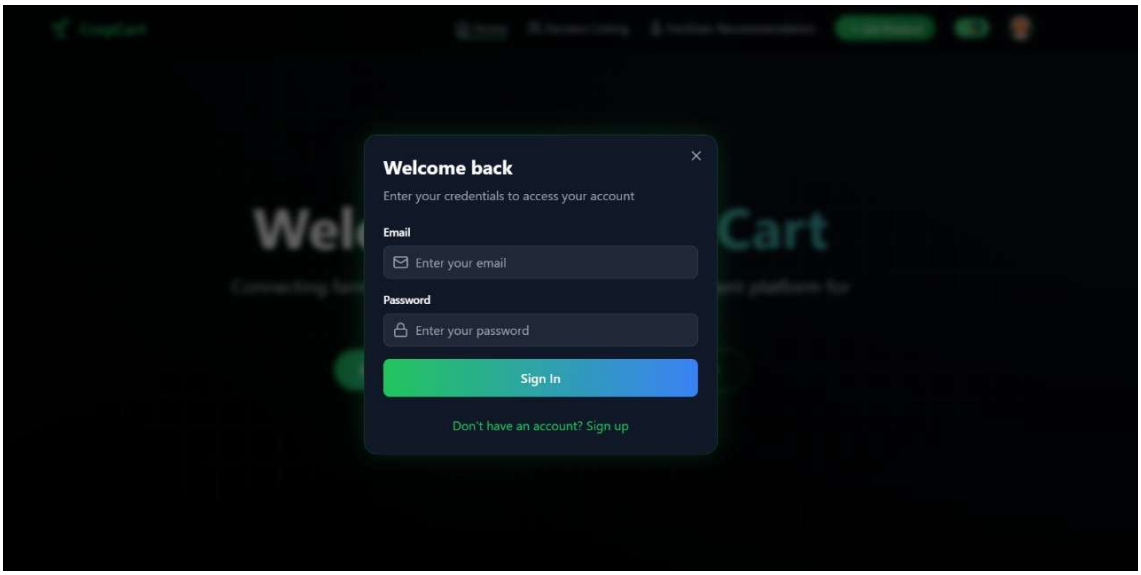

Figure 3.4 Document Storage

3.1.6 Real-Time Notifications and Updates

The platform employs notification features to keep users informed about relevant updates.

- Farmers receive alerts on order placements and buyer inquiries.
- Buyers are notified about price changes, new listings, and order confirmations.
- Experts get updates on new queries and session bookings.

3.2. Software and Hardware Requirements

Table 3.1 Software Requirements

	SOFTWARE REQUIREMENTS
1	MongoDB
2	Express.js
3	React.js
4	Node.js
5	Vercel
6	Postman
7	Visual Studio Code

3.3 System Architecture (MERN)

The architecture of Cropcart follows the Model-View-Controller (MVC) design pattern to ensure a clean separation of concerns.

1. Presentation Layer (Frontend): Built with React.js, utilizing functional components and Hooks (use State, use Effect) for state management. This ensures a fast, reactive UI that only updates necessary components without reloading the page.
2. Logic Layer (Backend): An Express.js server running on Node.js. This layer handles the REST API endpoints, processes business logic, and manages the communication between the UI and the database.
3. Data Layer (Database): MongoDB serves as the primary data store. Unlike traditional SQL databases, its document-oriented structure is ideal for agricultural data, which can vary in attributes (e.g., different crops having different measurement units).

CHAPTER – 4
METHODOLOGY

To develop an efficient and user-friendly *Cropcart* system, a structured and iterative development methodology was followed, aligned with the Software Development Life Cycle (SDLC). The process began with a thorough analysis of user requirements, focusing on the specific needs of two primary user groups: farmers and buyers. Farmers needed an easy way to list their crops and agricultural products, while buyers required a seamless interface to browse, search, and purchase produce directly. The planning phase involved defining core functionalities, such as product listing, order management, expert advisory access, and admin moderation. Clear objectives were established to ensure the platform remained practical, scalable, and relevant to real-world agricultural challenges.

During the design and implementation phases, special attention was given to user interface and experience, ensuring the frontend was responsive, accessible, and intuitive across devices using React.js and Tailwind CSS. Real-time features, such as live notifications for order updates and approvals, were added using WebSockets or AJAX, promoting timely communication between users. On the backend, technologies like Node.js, Express.js, and MongoDB were utilized to build a secure, role-based system capable of handling large volumes of data. Security measures, including password encryption and input validation, ensured safe interactions. Finally, the platform was deployed using cloud hosting services like Vercel and Render, allowing smooth updates, scalability, and 24/7 accessibility. Overall, the structured methodology ensured a robust, scalable, and user-friendly digital marketplace tailored for the agricultural community.

4.1. Project Planning and Requirement Analysis

4.1.1. Objective Definition

The primary goal of *Cropcart* is to build a web-based platform that empowers farmers to list and sell agricultural products, while enabling buyers to browse and purchase directly. The platform also integrates an expert advisory section to share farming tips and practices, enhancing the knowledge base of the farming community.

4.1.2. Functional Requirements

Users can register/login, manage product listings (images, pricing, description), and view/manage orders. Buyers can browse, search, view product details, and place orders. Admins can approve/reject users and listings, and manage advisory content. The advisory section displays articles, videos, and tips. Users receive notifications on order status, approvals, and updates

4.1.3. Non-functional Requirements

- Responsive and mobile-friendly interface
- Secure user authentication and data handling
  Fast and real-time updates (using AJAX or WebSockets)
- Scalable architecture for future expansion
- Cloud-based deployment for availability

. 4.2. System Design

4.2.1. Front-end Development

The front-end of *Cropcart* is designed with a responsive and intuitive user interface, using HTML, CSS, JavaScript, and React.js for interactivity. Farmers have user-friendly forms to list products, including fields for name, description, price, quantity, and image upload, with JavaScript validation for required fields. Buyers can browse, search, and view product details, with a cart and checkout system in place. The admin dashboard displays product listings in a table format, with options to approve or reject them, and allows management of user registrations and advisory content. The design ensures a seamless experience across devices using CSS frameworks like Tailwind CSS or Bootstrap.

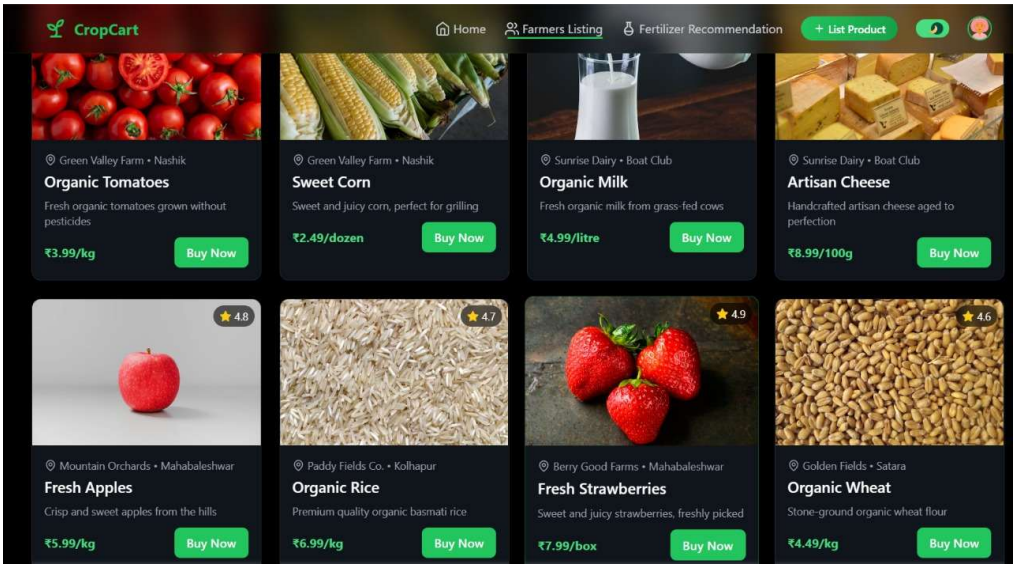

Figure 4.1 Farmer Dashboard

4.2.2. Back-end Design

The back-end of *Cropcart* defines endpoints in Node.js with Express.js to handle key functionalities such as user registration, login authentication, product listings, and order management. JWT is used for secure, token-based authentication, ensuring role-based access for farmers, buyers, and admins. Endpoints are developed for CRUD operations on product listings, allowing farmers to manage their products, and for managing orders, enabling buyers to place and track them. Admins have the ability to approve/reject product listings, manage users, and oversee advisory content. Security measures include input validation, sanitization to prevent SQL injection, and bcrypt for securely hashing passwords, ensuring safe and efficient data handling throughout the platform.

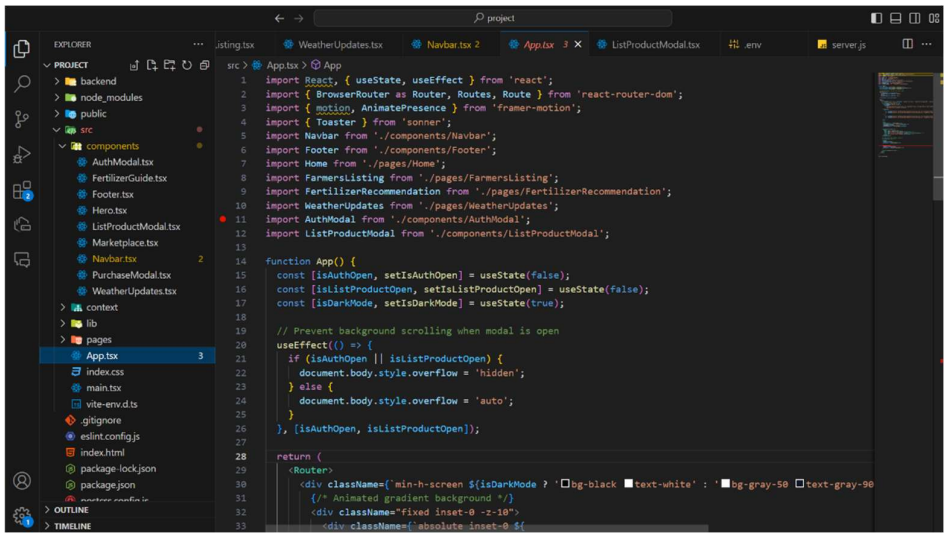

Figure 4.2 App.tsx

4.3. Database Integration

The database for *Cropcart* is built using MongoDB, leveraging its flexibility to store user information, product listings, orders, and advisory content. Using Mongoose for object data modeling, the database includes collections for users (with role-based access), products (with details like name, price, and images), orders (tracking order status and buyer information), and advisory content (articles and videos). Relationships between users, products, and orders are maintained through document references, ensuring data integrity. Security features, such as encryption and input validation, are implemented to protect sensitive data, allowing the platform to handle dynamic content and scale efficiently.

4.4 Technical Security Framework

To protect user data and financial integrity, the following security protocols were implemented:

- Password Hashing: Utilizing the bcrypt library, user passwords are "salted" and hashed before storage, ensuring that even in the event of a database leak, raw passwords remain unreadable.
- Stateless Authentication: We implemented JSON Web Tokens (JWT). When a user logs in, the server issues a signed token. This token is sent in the header of subsequent API requests, allowing the server to verify the user's identity without storing session data in memory.

- Data Validation: Using server-side validation logic, all user inputs are sanitized to prevent common vulnerabilities such as Cross-Site Scripting (XSS) and NoSQL Injection.

4.5. System Workflow and Process Logic

This section outlines the operational flow of the Cropcart platform, detailing how data moves from the user interface through the server logic to the database.

4.5.1. User Interaction Flow

The system follows a linear progression for the primary transaction cycle. The process begins when a Farmer authenticates and initiates a product listing. The logic flow is as follows:

1. **Input Acquisition:** The React frontend captures product attributes and image files.
2. **Request Dispatch:** Data is bundled into a FormData object and sent via an asynchronous POST request to the Express backend.
3. **Validation & Transformation:** The backend verifies the session via JWT and processes the image using Multer middleware.
4. **Persistence:** The validated data is mapped to the Mongoose Schema and saved to MongoDB.

4.5.2. Data Processing Pipeline

To maintain high performance, Cropcart utilizes a non-blocking I/O model provided by Node.js. For the Expert Advisory section, the data pipeline follows a **Read-Heavy optimization** strategy:

- **Query Optimization:** Using Mongoose .find() with projection to retrieve only necessary fields (e.g., title, summary, thumbnail) for the main advisory feed, reducing payload size.
- **Dynamic Rendering:** React components use the .map() function to dynamically render cards based on the JSON response, ensuring the UI stays responsive even as the knowledge base grows.

4.5.3. State Management Strategy

The methodology incorporates a centralized state management approach using React Hooks.

- **Local State:** useState is used for handling form inputs and UI toggles (like opening a "Contact Expert" modal).
- **Global Context:** To avoid "Prop Drilling," user authentication status and shopping cart data are managed through the **React Context API**, allowing any component in the tree to access current user details without redundant API calls.

4.5.4. Error Handling and Exception Logic

A robust methodology requires a systematic way to handle failures. Cropcart implements a dual-layer error handling strategy:

1. **Frontend Boundary:** Try-catch blocks wrap all API calls. If a network error occurs, a user-friendly Toast notification (via react-toastify) informs the user without breaking the application state.
2. **Backend Global Middleware:** A centralized error-handling middleware in server.js catches all unhandled exceptions, logs them for the developer, and returns a standardized JSON error object to the client, preventing the exposure of sensitive stack traces.

CHAPTER – 5
IMPLEMENTATION DETAILS

5.1. User Authentication

User authentication in *Cropcart* is implemented using JSON Web Tokens (JWT) for secure login and access control. During registration, users (farmers, buyers, and admins) provide their email, password, and role, with passwords securely hashed using bcrypt. Upon successful login, a JWT is generated containing the user's ID and role, which is stored in the client's local or session storage for subsequent requests. Role-based access control is enforced, ensuring farmers, buyers, and admins can only access their designated features. The JWT has an expiration time, and a refresh token mechanism can be used for extended sessions. Security measures, including HTTPS encryption, password hashing, and rate-limiting to prevent brute force attacks, ensure a secure and reliable authentication process.

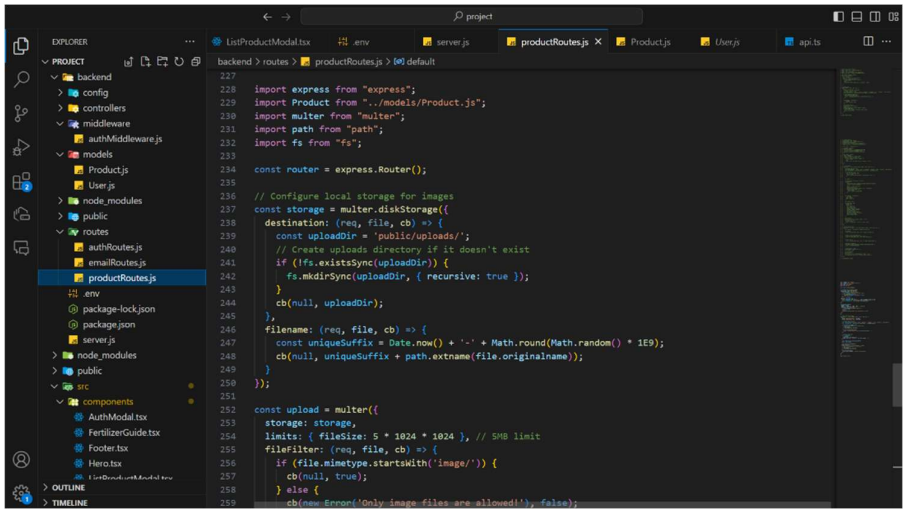

Figure 5.1 ProductListing.js

5.2. Farmer Section Implementation

The farmer home page displays essential details like their profile, active listings, and system notifications. React fetches this data dynamically from the backend, ensuring real-time updates.

5.2.1. Product Listings

Farmers can add and manage product listings, including crop name, price, quantity, and images. Form validation ensures required fields are completed, and data is stored securely in the MongoDB database.

5.2.2. Order Management

Farmers can view orders placed by buyers, including order details (buyer information, quantity, and total price). The order status (pending, shipped, completed) is dynamically updated, and any status change is reflected in real-time.

5.2.3. Password Management

Farmers can update their password through a secure form. Passwords are hashed before being updated in the MongoDB database, ensuring secure storage.

5.2.4. Notifications

Farmers receive real-time notifications about new orders, product approvals, and system updates. Notifications are pushed via WebSocket for instant delivery.

5.3. Buyer Section Implementation

The Admin section of *Cropcart* allows administrators to manage the platform by overseeing farmer registrations, product listings, and advisory content. Admins can approve or reject farmer accounts and product listings, ensuring only verified products are available. They can also create, edit, or delete educational content in the advisory section, providing valuable farming insights to users. Additionally, admins have access to platform-wide statistics, such as total products listed and active users, and can manage system notifications. All actions performed by the admin are dynamically reflected in the MongoDB database, ensuring real-time updates across the platform.

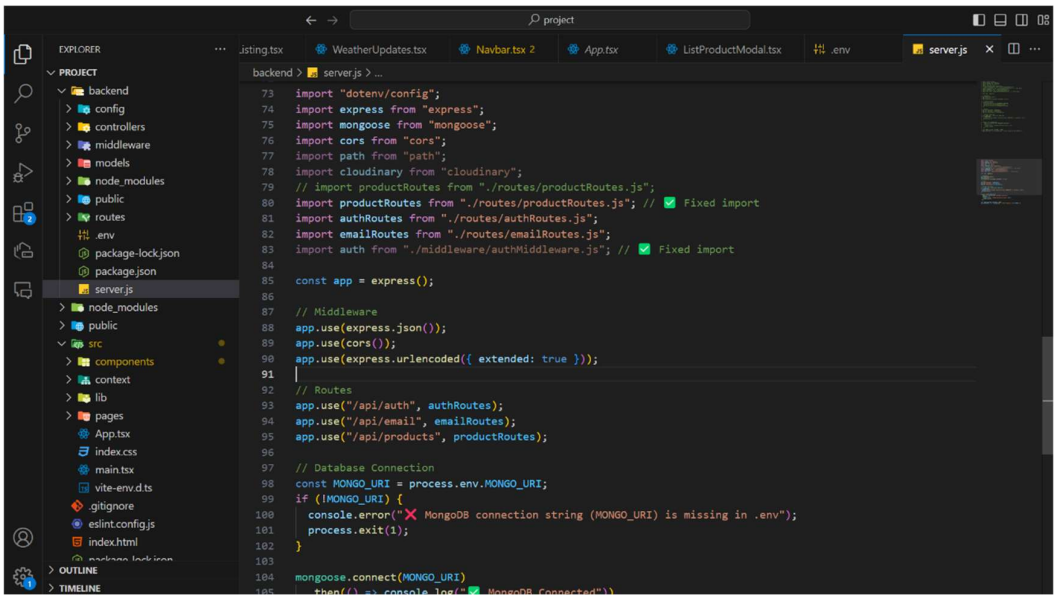

Figure 5.3 Server.js

5.4. Database Structure

The MongoDB database for *Cropcart* consists of collections such as users, products, orders, advisory content, and notifications. **Users** store details of farmers, buyers, and admins, while **products** contain details like name, price, and quantity. **Orders** track buyer purchases and their status, and **advisory** stores educational content for farmers. **Notifications** manage real-time updates for all users. Relationships are handled through document references, ensuring efficient querying and scalability, while MongoDB's flexible schema allows the platform to adapt as new features are introduced.

5.5 Backend API Architecture

The backend of Cropcart is built as a RESTful service. Each module (User, Product, Order) has a dedicated controller that handles the business logic.

5.5.1 Controller Logic for Product Management

The productController.js handles the CRUD (Create, Read, Update, Delete) operations. When a farmer adds a product, the backend performs the following steps:

Request Parsing: Extracts product data (name, price, category) from req.body and the image file from req.file.

Validation: Checks if the price is a positive number and if the required fields are present.

Storage: Saves the image to a cloud storage service (or local uploads folder) and stores the resulting URL in MongoDB.

Response: Returns a 201 Created status code along with the newly created product object to the frontend.

5.6 Middleware Implementation

Middleware functions are utilized to intercept requests before they reach the final controller, ensuring security and data integrity.

- Auth Middleware: Verifies the JWT sent in the request headers. If the token is missing or invalid, it returns a 401 Unauthorized error.
- Role-Based Middleware: After authentication, this middleware checks the user.role property. For example, if a user tries to access the /api/admin route, this middleware ensures only users with the admin role are permitted.
- Multer Middleware: Used for handling multipart/form-data, which is primarily used for uploading crop images. It filters file types to ensure only .jpg or .png files are uploaded.

5.7 Integration of Expert Advisory Section

The Expert Advisory section is implemented as a content-delivery module where verified experts can post articles and resources.

1. Content Model: The Advisory schema in MongoDB includes fields for title, author (referenced from the User model), content (Markdown/Rich Text), and category (e.g., Pest Control, Soil Health).
2. Frontend Rendering: On the React side, a custom Advisory Card component is used to map through the list of articles fetched from the /api/advisory endpoint.

3. Real-time Interaction: To allow farmers to ask questions, a "Comments" sub-document is integrated within the Advisory model, allowing for a nested discussion thread between farmers and experts.

5.8 Testing and Debugging

To ensure the reliability of the implementation, several testing methodologies were employed:

- Unit Testing: Individual backend functions (like password hashing) were tested to ensure they return the expected output for various inputs.
- Postman for API Testing: All REST endpoints were rigorously tested using Postman to verify status codes, headers, and JSON payloads.
- Component Testing: React components were tested for responsiveness and state management using browser developer tools.

CONCLUSION

In conclusion, *Cropcart* is a robust and user-friendly platform designed to bridge the gap between small-scale farmers and buyers, providing an efficient marketplace for agricultural products while also offering valuable insights through its expert advisory section. Built using modern technologies such as React.js, Node.js, Express.js, and MongoDB, the platform ensures seamless user experiences with features like real-time notifications, secure authentication, and efficient product and order management. Farmers can easily list their products, manage orders, and access farming advice, while buyers benefit from a streamlined purchasing process and direct communication with sellers.

The integration of real-time features and a secure, scalable backend ensures that *Cropcart* can grow with the evolving needs of the agricultural industry. The admin section provides effective management tools for overseeing product listings, user registrations, and advisory content, making the platform not only a marketplace but also a valuable resource for improving agricultural practices. By enhancing market access and supporting knowledge sharing, *Cropcart* empowers both farmers and buyers, contributing to the growth of the agricultural community and improving trade opportunities.

BIBLIOGRAPHY

1. Food and Agriculture Organization of the United Nations. (2022). Digital Agriculture: A Key to Transformation in Agri-Food Systems. Retrieved from [https://www.fao.org] (https://www.fao.org)

2. Government of India, Ministry of Agriculture and Farmers Welfare. (2023). eNAM – National Agriculture Market. Retrieved from [https://www.enam.gov.in] (https://www.enam.gov.in)

3. Meena, M. S., & Singh, K. M. (2021). ICT in Agriculture: Opportunities and Challenges. Indian Journal of Agricultural Economics, 76(1), 34–45.

4. World Bank Group. (2020). Agricultural Technology Adoption in Developing Countries. Retrieved from [https://www.worldbank.org] (https://www.worldbank.org)

5. Ramesh, P., & Dubey, A. K. (2022). Smart Agriculture Using IoT and Data Analytics: Trends and Applications. Journal of Agricultural Informatics, 13(2), 55–66.

6. AgFunder. (2023). AgriTech Investing Report 2023. Retrieved from https://agfunder.com/reports

7. Sharma, S., & Patel, R. (2021). Role of Mobile Applications in Connecting Farmers and Markets. International Journal of Emerging Technology and Advanced Engineering, 11(6), 110–117.

8. Indian Council of Agricultural Research (ICAR). (2022). Technological Innovations for Sustainable Agriculture. Retrieved from [https://icar.org.in] (https://icar.org.in)