Project-I Proposal Report

On

Stable Market Access through Assured Contract Farming: A Digital Solution

Submitted By:

Anmol Purohit -

Lalit Gaur - ROLL No.

Yash Bhandari - 22EJICS164

Submitted To:

Department of Computer Science & Engineering

In partial fulfillment of requirement for the degree of

Bachelor of Technology

Jodhpur Institute of Engineering & Technology

(An Autonomous Institution affiliated with Bikaner Technical University, Bikaner)

Date of Submission:

September 19, 2025

Contents

Chapter 1

Introduction

1.1 Background of the Study

Agriculture remains a critical sector in India, employing a significant portion of the population and contributing substantially to the national economy. Despite its importance, farmers often face structural constraints such as fragmented markets, dependency on intermediaries, price volatility, and limited access to reliable buyers. These issues lead to uncertain incomes and discourage investment in improved farming practices.

Digital platforms and contract farming models present opportunities to reduce these inefficiencies by connecting producers directly with buyers, ensuring agreed prices or supply commitments, and improving transparency. However, many existing solutions lack legally enforceable agreements, easy-to-use interfaces for low-digital-literacy users, or integrated payment and dispute-resolution mechanisms. This project proposes an **Assured Contract Farming System** — a farmer-centric digital solution designed to provide stable market access, transparent contract management, and secure transactions.

1.2 Problem Statement

Small and marginal farmers frequently encounter unstable market access, sudden price drops, and exploitation by intermediaries. Buyers similarly face challenges in sourcing consistent, quality produce from reliable suppliers. The absence of a streamlined, legally-backed digital mechanism for negotiating, creating, and enforcing purchase contracts exacerbates these problems. Consequently, farmers lack income predictability and buyers lack assured supply, which together reduce the efficiency and fairness of agricultural markets.

Therefore, this project aims to design and implement a digital system that:

- Enables direct connections between farmers and buyers through legally-assured contracts,
- Provides price assurance and contract enforcement mechanisms,
- Integrates secure payment processing and basic dispute resolution,
- Is accessible to farmers with limited digital literacy.

1.3 Objectives of the Project

The primary objectives of this project are:

1. To design and develop a farmer-friendly digital marketplace for assured contract farming.
2. To implement contract creation, negotiation, storage, and management features with legal traceability.
3. To integrate secure payment gateways and transaction records to ensure timely and verifiable settlements.
4. To provide user interfaces and workflows that are usable by farmers with low digital literacy (mobile-first design).
5. To include basic analytics and notifications for contract milestones, delivery scheduling, and alerts.
6. To evaluate the solution through a pilot test (functional and usability testing) and collect feedback for improvement.

1.4 Scope of the Project

This project focuses on a Minimum Viable Product (MVP) for an Assured Contract Farming System with the following scope items:

- Development of a mobile-first web application for farmers and buyers.
- Support for contract lifecycle: request, negotiation, acceptance, delivery scheduling, and payment settlement.

- Integration with at least one payment gateway for secure transfers (gateway selection and sandbox integration).
- Administrative dashboard for basic monitoring, dispute logging, and user management.
- Documentation of functional and non-functional requirements, system design artifacts (architecture, ER diagrams, DFD), and test cases.

Out of scope for the MVP:

- Full-scale logistics integration with multiple carriers (may be planned as future work).
- Advanced AI price prediction or complex blockchain-based immutable ledgers (not included in initial MVP; can be future enhancements).
- Multi-country or multi-state legal compliance beyond a generic contract template (localization to be addressed in future iterations).

1.5 Significance of the Study

The proposed system aims to achieve practical benefits for key stakeholders:

- **Farmers:** Increased income predictability, reduced reliance on middlemen, and improved bargaining power via access to direct contracts.
- **Buyers:** Reliable sourcing, clearer supply scheduling, and traceable transaction records.
- **Market Efficiency:** Reduced information asymmetry, lower transactional friction, and potential uplift in quality and supply consistency.
- **Academia/Industry:** A reference implementation and documentation for further research or real-world pilots in contract farming platforms.

The study also contributes to a user-centered design approach for agricultural digital platforms, prioritizing accessibility and legal enforceability.

1.6 Methodology Overview

This project follows a structured methodology combining standard software engineering practices with domain-specific research:

1. **Requirement Gathering:** Conduct surveys and interviews with potential users (farmers and buyers), review literature and existing platforms (e.g., e-NAM, AgriBazaar), and finalize the Software Requirements Specification (SRS).

2. **System Design:** Create high-level architecture, ER diagrams, DFD (Level-0 / Level-1), and UML diagrams (use cases, class diagrams, sequence diagrams) to plan the system components and interactions.

3. **Development:** Implement the application using an appropriate stack (e.g., React/Flutter frontend, Flask/Django/Node backend, relational database). Follow modular development practices and version control (Git).

4. **Integration:** Integrate payment gateway sandbox, implement contract storage and basic legal templates, and build notification modules (email/SMS/WhatsApp/Push).

5. **Testing:** Perform unit testing, integration testing, usability testing with target users, and security checks (basic authentication, input validation).

6. **Deployment and Pilot:** Deploy the MVP to a cloud or hosting environment and run a small pilot to collect user feedback and measure key metrics (contract completion rate, time to payment, user satisfaction).

7. **Documentation and Reporting:** Prepare project report chapters, user manuals, and test reports for CIE submission and future reference.

Chapter 2

Literature Survey

2.1 Introduction

The agricultural sector has been the subject of extensive research aimed at improving farmer income, ensuring fair pricing, and minimizing dependence on intermediaries. Traditional agricultural markets in India often expose farmers to uncertainties of price fluctuations, exploitation by middlemen, and limited access to reliable buyers. Literature suggests that technology-driven solutions, particularly digital platforms, can mitigate these challenges by enhancing transparency, improving bargaining power, and enabling assured market access through contractual agreements.

2.2 Contract Farming Approaches

Contract farming has long been recognized as a method to reduce risks associated with agricultural marketing. According to several studies, contract farming arrangements ensure farmers receive pre-determined prices and guaranteed buyers, thereby stabilizing their incomes. Research highlights the following benefits:

- Reduction of price uncertainty by fixing contracts prior to cultivation.
- Assurance of timely procurement for buyers and stable income for farmers.
- Promotion of high-value crops through buyer–farmer agreements.

However, drawbacks such as disputes over quality, delayed payments, and lack of legal enforcement have limited its effectiveness in practice.

2.3 Existing Digital Agriculture Platforms

Several digital platforms have been introduced to modernize agricultural marketing in India:

- **e-NAM (Electronic National Agriculture Market):** A government initiative to create a unified national market for agricultural commodities. While promising, adoption remains low due to technical barriers and lack of farmer training.
- **AgriBazaar:** A private digital marketplace enabling farmers to sell directly to buyers. Despite its potential, issues such as trust, contract enforcement, and language barriers hinder mass adoption.
- **Reliance Fresh and ITC e-Choupal:** Early private initiatives that attempted to link farmers with markets. These initiatives provided localized success but faced scalability challenges.

These platforms highlight the role of technology but also underscore the need for legally binding, transparent, and accessible systems for contract farming.

2.4 Technology Integration in Agriculture

Advancements in technology have enabled new possibilities in agriculture:

- **Mobile Applications:** Mobile-first applications have been developed to provide market prices, weather updates, and advisory services to farmers.
- **Payment Gateways:** Secure digital payment solutions ensure timely and traceable transactions, reducing disputes between farmers and buyers.
- **Blockchain:** Studies suggest blockchain can enhance transparency and trust in agricultural supply chains by providing immutable records of contracts and transactions.
- **AI-based Market Prediction:** Machine learning models can predict commodity prices and demand trends, enabling better decision-making for farmers.

2.5 Challenges in Existing Systems

Despite various digital initiatives, key challenges remain unresolved:

- Limited adoption due to low digital literacy among farmers.

- Lack of multilingual support restricting accessibility in rural areas.
- Absence of legally assured and enforceable contracts in most digital platforms.
- Trust deficit between farmers and buyers, often leading to contract disputes.
- Weak dispute resolution mechanisms and delays in payment settlement.

2.6 Research Gap Identified

The literature highlights the gap between existing digital marketplaces and the needs of small and marginal farmers. While platforms exist, they often fail to provide:

- Legally assured contracts that guarantee both pricing and procurement.
- Transparent negotiation mechanisms to reduce exploitation by intermediaries.
- Integrated, secure payment gateways tailored for low-literacy users.
- Additional farmer support services such as advisory, reminders, and dispute resolution.

This research gap forms the basis of the proposed project, which seeks to build a farmer-centric digital platform for stable market access through assured contract farming.

Chapter 3

Requirement Specification & Methodology

3.1 Software Development Life Cycle (SDLC) Model

The development of the Assured Contract Farming System follows the **Iterative and Incremental SDLC model**. This model was chosen because it allows progressive development of the system while incorporating user feedback from farmers and buyers at each stage. The key phases are:

1. **Requirement Analysis:** Gather user requirements through surveys, interviews, and analysis of existing digital agriculture platforms.
2. **System Design:** Prepare architectural diagrams, ER diagrams, and Data Flow Diagrams (DFDs) to define the system's structure.
3. **Implementation:** Develop the system in modules (Farmer Module, Buyer Module, Admin Module, Payment Gateway).
4. **Testing:** Conduct unit testing, integration testing, usability testing, and security testing.
5. **Deployment:** Deploy the system in a pilot phase with limited farmers and buyers.
6. **Maintenance:** Gather feedback, fix bugs, and plan enhancements such as multilingual support and AI-based market prediction.

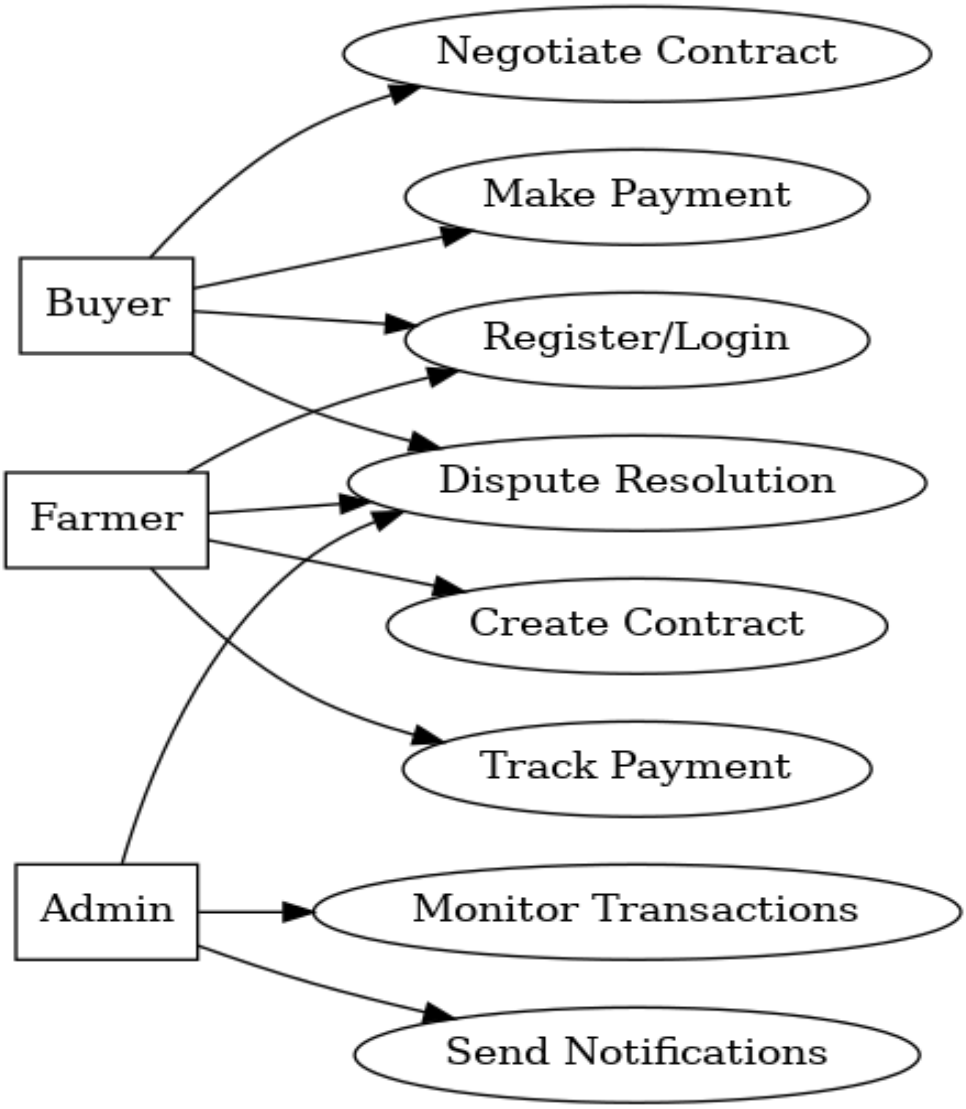
This model provides flexibility to refine requirements during development, ensuring the final product aligns with real-world user needs.

3.2 Use Case Diagrams

The system has three main actors: **Farmer**, **Buyer**, and **Admin**. Key use cases are:

- Farmer registers, creates/view contracts, receives payments, and raises disputes.
- Buyer registers, negotiates contracts, makes payments, and confirms deliveries.
- Admin monitors transactions, manages disputes, and maintains system integrity.

Note: The actual Use Case Diagram should be created using tools like StarUML, Lucidchart, or draw.io and inserted as a figure here. Example placeholder:



3.3 Interfaces

The system requires multiple interfaces to ensure smooth communication between software components, hardware, and users.

3.3.1 Software Interfaces

- **Frontend:** Web and mobile-first interface developed using React.js/Flutter for farmers and buyers.
- **Backend:** REST API developed using Node.js/Flask/Django to handle core business logic.
- **Database:** Relational database (MySQL/PostgreSQL) for storing contracts, user profiles, and transaction records.
- **Payment Gateway:** Integration with services like Razorpay/Paytm for secure digital payments.
- **Authentication:** Secure login using JWT/OAuth2.

3.3.2 Hardware Interfaces

- **Client Devices:** Smartphones (Android/iOS) or computers with internet connectivity.
- **Server:** Cloud-hosted virtual machine or server with minimum configuration: Quad-core CPU, 8GB RAM, and scalable storage.

3.3.3 Communication Interfaces

- **Internet Protocols:** HTTPS for secure communication between client and server.
- **Notification Services:** SMS gateway and email service for contract updates and reminders.
- **Third-Party APIs:** Optional integration with weather APIs or advisory systems for additional farmer support.

3.4 Methodology

The methodology adopted for this project ensures systematic development and testing of the proposed system:

1. **Requirement Gathering:** Identify challenges faced by farmers and buyers and translate them into functional and non-functional requirements.
2. **System Design:** Define architecture, use cases, and data models.
3. **Module Development:** Build independent modules (Farmer, Buyer, Admin, Payment).
4. **Integration:** Connect modules and third-party services (payment, notifications).
5. **Testing and Validation:** Conduct functional, usability, and security testing.
6. **Deployment:** Launch the pilot system and collect user feedback.

This iterative methodology ensures that user needs remain central throughout the project lifecycle.

Chapter 4

Work Distribution

4.1 Task Allocation

The project has been undertaken by a team of three students: **Anmol Purohit**, **Lalit Gaur**, and **Yash Bhandari**. To ensure systematic progress, tasks were distributed among the team members according to their skills and areas of interest.

- **Anmol Purohit:** Focused on frontend interface development, use case documentation, project documentation and testing.
- **Lalit Gaur:** Handled requirement analysis, report writing, and coordination of modules
- **Yash Bhandari:** Responsible for database design, backend development, and system integration.

This distribution ensures parallel development of system components and timely integration.

4.2 Module Details

The proposed system is divided into distinct modules for clarity and better management:

1. **Farmer Module:** Registration, contract creation, viewing available contracts, payment tracking.
2. **Buyer Module:** Registration, searching for farmers, negotiating contracts, making payments.
3. **Admin Module:** Monitoring transactions, resolving disputes, and system maintenance.
4. **Payment Module:** Secure payment gateway integration (Razorpay/Paytm).

5. **Notification Module:** SMS/email alerts for contract milestones, delivery reminders, and payment confirmations.

4.3 Remark

The team adopted a collaborative approach where members contributed to each other's domains when required. Regular team meetings were held to review progress, resolve issues, and ensure alignment with project goals. This cooperative work distribution facilitated timely completion of the initial project phases and will continue throughout the development lifecycle.

Chapter 5

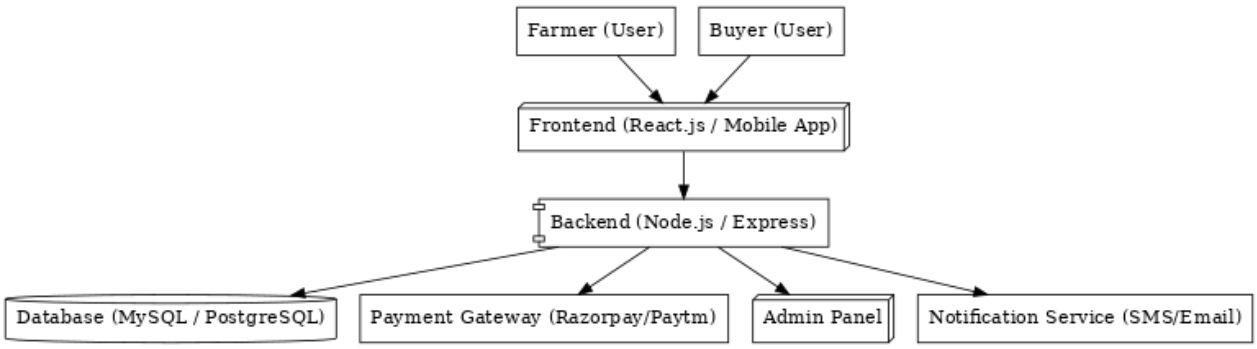
Design Document

This chapter presents the detailed design of the proposed system. It includes the functional description, functional partitions, data organization, user interface design, module specifications, and process flow representations. The design ensures that the system is modular, scalable, and user-friendly.

5.1 Functional Description

The Assured Contract Farming System is designed to connect farmers and buyers through a digital platform that ensures transparent and legally valid contracts. The system supports the following core functions:

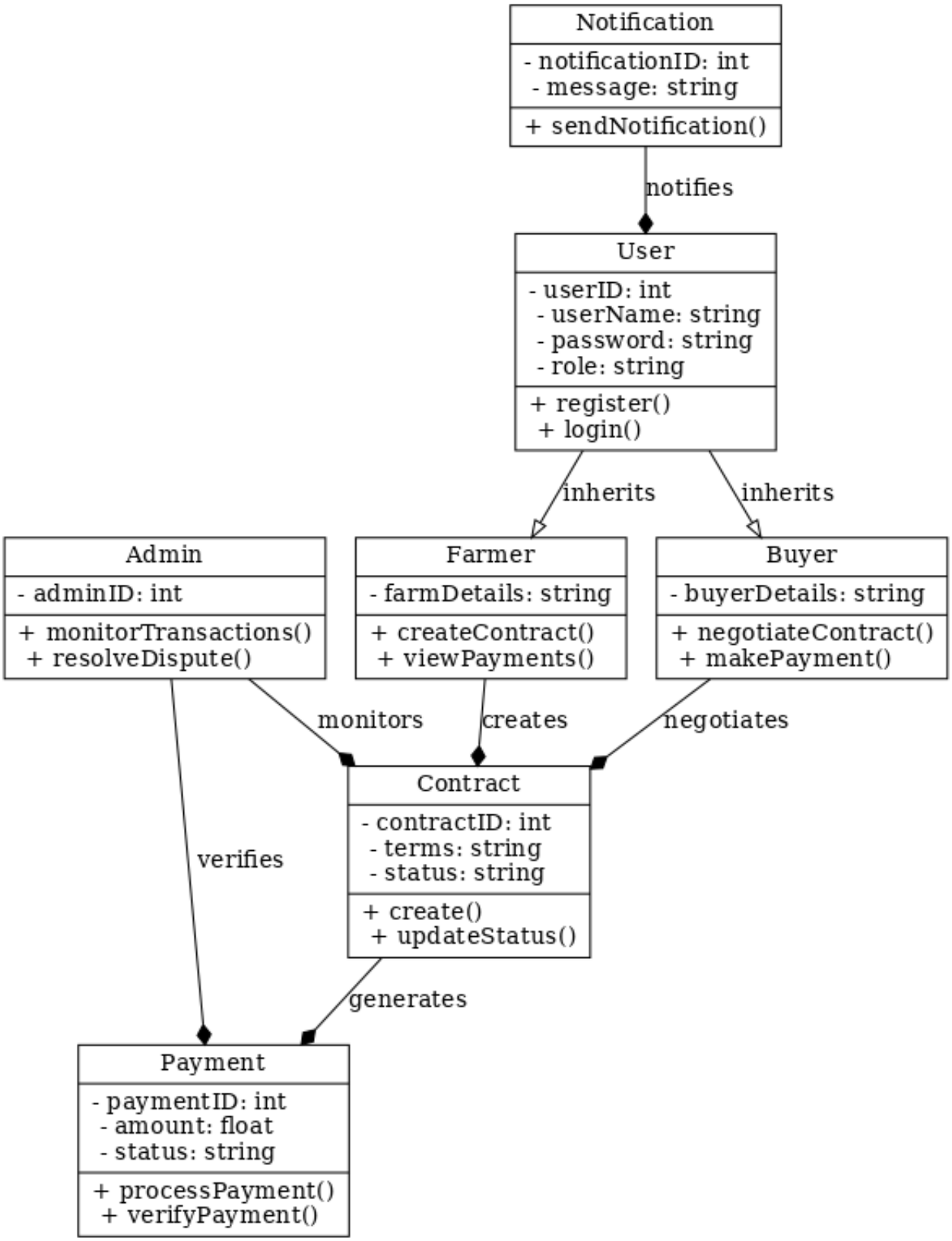
- **Farmer Functions:** Registration, contract creation, negotiation, payment tracking, and dispute raising.
- **Buyer Functions:** Registration, contract negotiation, payment processing, and delivery confirmation.
- **Admin Functions:** Monitoring transactions, dispute resolution, and maintaining system security.
- **Payment Functions:** Secure digital transactions via integrated payment gateways.
- **Notification Functions:** Automated SMS/email alerts for contract milestones and payment confirmations.



5.2 Functional Partitions

The system is divided into logical modules, each performing a specific set of tasks:

1. **Farmer Module:** Registration, contract management, and payment receipt.
2. **Buyer Module:** Product search, contract negotiation, and secure payments.
3. **Admin Module:** System monitoring, user management, and dispute resolution.
4. **Payment Module:** Handles secure integration with third-party payment gateways.
5. **Notification Module:** Sends reminders and alerts regarding contracts and payments.


Notification
- notificationID: int
- message: string
+ sendNotification()
notifies
User
- userID: int
- userName: string
- password: string
- role: string
+ register()
+ login()
inherits
inherits
Admin
- adminID: int
+ monitorTransactions()
+ resolveDispute()
Farmer
- farmDetails: string
+ createContract()
+ viewPayments()
Buyer
- buyerDetails: string
+ negotiateContract()
+ makePayment()
monitors
creates
negotiates
verifies
Contract
- contractID: int
- terms: string
- status: string
+ create()
+ updateStatus()
generates
Payment
- paymentID: int
- amount: float
- status: string
+ processPayment()
+ verifyPayment()

5.3 Data Description

The database schema is designed to support contract management, user authentication, and transaction tracking.

Table 5.1: Sample Database Schema

Field Name	**Data Type**	**Constraints**	**Description**
UserID	INT	Primary Key	Unique user identifier
UserName	VARCHAR(50)	Not Null	Name of the user
Password	VARCHAR(100)	Not Null	Encrypted user password
Role	VARCHAR(20)	Default 'User'	Defines user type (Admin/Farmer/Buyer)
ContractID	INT	Primary Key	Unique contract identifier
PaymentID	INT	Foreign Key	Links to payment transactions

Figure 5.3: Entity Relationship Diagram of the Database

Figure 5.4: Data Flow Diagram (Level 1) of the System

5.4 User Interface Design

The system is designed as a mobile-first platform with simple navigation and multilingual support. The interface provides:

- Easy registration and login.
- Dashboard for farmers and buyers to track contracts and payments.
- Notifications section for alerts and reminders.
- Support section for dispute resolution.

Figure 5.5: Sample User Interface Mockup

5.5 Module Description

Module 1: Farmer Module Inputs: Farmer registration data, contract details. Processing: Validation, contract creation, payment record update. Outputs: Stored contract, payment confirmation.

Module 2: Buyer Module Inputs: Buyer registration data, contract requests. Processing: Contract negotiation, payment initiation. Outputs: Buyer confirmation, payment record.

Module 3: Admin Module Inputs: User reports, disputes, system logs. Processing: Issue resolution, monitoring, user control. Outputs: Updated system records, dispute reports.

Module 4: Payment Module Inputs: Payment request. Processing: API call to gateway, verification, transaction logging. Outputs: Payment receipt, status update.

Module 5: Notification Module Inputs: Contract milestones, payment events. Processing: Generate alerts. Outputs: SMS/email notifications to users.

5.6 Process Flow Representation

The execution of the system follows structured workflows. Flowcharts and sequence diagrams illustrate system interactions.

Figure 5.6: Flowchart of the System Process

Figure 5.7: Sequence Diagram of User Interaction with System

5.7 Deployment View

The system is deployed on a cloud-hosted server with the following structure:

- Client devices: Smartphones or PCs with internet access.
- Application server: Hosts backend, APIs, and databases.
- Payment gateway: Integrated third-party service for secure transactions.

Figure 5.8: Deployment Diagram of the System