



**SCHOOL OF BUSINESS AND ECONOMICS  
DEPARTMENT OF BUSINESS TECHNOLOGY  
BIT DEPARTMENT**

**Level 2**

**Group1**

**SYSTEM ENGINEERING**

**Project Name:**

**MARKETING MOBILE APPLICATION FOR  
FARMERS**

**Student name: NYIRANEZA Jacqueline**

**Reg No: 222008848**

**Submitted to: Dr. BUGINGO Emmanuel**

**Date: 8/ Mar 2024**

## Table of Contents

LIST OF FIGURES .....	II
INTRODUCTION .....	1
PROBLEM STATEMENT AND SOLUTION .....	1
OBJECTIVES OF THE PROJECT .....	2
GENERAL OBJECTIVES.....	2
SPECIFIC OBJECTIVES.....	2
SYSTEM REQUIREMENTS.....	2
FUNCTIONAL REQUIREMENT OF THE PROPOSED SYSTEM .....	2
Users LOGIN .....	2
Add new farmer .....	2
Add new crop .....	3
Approve Crops.....	3
Reject Crops .....	3
NONFUNCTIONAL REQUIREMENTS.....	3
USE CASE DIAGRAM .....	4
DATA FLOW DIAGRAM .....	5
SEQUENCE DIAGRAM .....	8
CLASS DIAGRAM .....	9
FEASIBILITY STUDY .....	10
TECHNICAL FEASIBILITY .....	10
FINANCIAL FEASIBILITY.....	11
MARKET FEASIBILITY .....	12
ECONOMIC FEASIBILITY.....	13
LEGAL AND REGULATORY FEASIBILITY .....	14
OPERATIONAL FEASIBILITY .....	15
DESCRIPTION OF SELECTED PROCESS MODEL .....	16
CONCLUSION .....	17
REFERENCES .....	18

## LIST OF FIGURES

Figure 1: Use Case Diagram .....	4
Figure 2: Level 0 .....	5
Figure 3: Level 1 .....	6
Figure 4: Level 2 .....	7
Figure 5: Sequence diagram.....	8
Figure 6: Class diagram .....	9

## INTRODUCTION

Farming is the act or process of working the ground, planting seeds, and growing edible plants. You can also describe raising animals for milk or meat as farming.

The major agricultural products can be broadly grouped into Food classes include cereals (grains), vegetables, fruits, oils, meat, milk, fungi and eggs. Over one-third of the world's workers are employed in agriculture, second only to the service sector.

Now technology is increasing very fast in the world today. Marketing application for farmers is mobile application, which is based on a powerful tool known as smart phone. The most people around the world use it; it connects buyer and sellers for the purpose of business. In Rwanda, information technology has increased but in Farming still has problems.

Marketing application is an application designed for joining farmers and customers easily for the purposes of view the available products and predicted products. The users with modern smart phone are able to access this application through the mobile application, which is designed specifically for mobile devices not desktop.

New technology can implement this marketing application for farmers in order to facilitate farmers to sell their products to the customers in Rwanda only.

## PROBLEM STATEMENT AND SOLUTION

Currently IGIRE Cooperative is facing problems related to the time spend by farmers to find the market of the productions. Farmers spent money while transporting to reach to the market without specified customers. Marketing application for farmers is developed to remove problem found in the existing system.

In order to accomplish our work, the following research question was formulated:

“How marketing application for farmers can be a solution to the problem related to the farmers to find the market with specified customers”

## OBJECTIVES OF THE PROJECT

### GENERAL OBJECTIVES

The main purpose of our project is to implement a marketing application for farmers. That handles the problems of cooperative Igire Such as to search customers of their productions.

This mobile application connects farmers with customers easily.

### SPECIFIC OBJECTIVES

Specific Objectives of this marketing application for farmers are the following:

- To develop a mobile application that facilitates the customers to get available products, predicted products; and send orders to the cooperative manager. Therefore, manager view orders and send response to customers.
- To develop a mobile application, to help manager to manage orders where customer send unpaid orders or upload fake receipt, approved/ rejected order, undelivered and delivered orders.
- To develop a mobile application that allows manager to get report to the shareholders, stock, payment and cooperative income.

## SYSTEM REQUIREMENTS

### FUNCTIONAL REQUIREMENT OF THE PROPOSED SYSTEM

#### Users LOGIN

DESCRIPTION: The system provides facility to login into the system.

INPUT: Enter username and password OUTPUT:

Home page.

PROCESSING: The system will check the input of user and if valid then login is done. Otherwise user will be asked to re-enter username and password.

#### Add new farmer

DESCRIPTION: The system provides facility to register a new farmer.

INPUT: Enter farmer related information.

OUTPUT: Farmers list.

PROCESSING: The system will save the provided data to the database.

### Add new crop

DESCRIPTION: The system provides facility to register a new crop.

INPUT: Enter crop related information.

OUTPUT: Crops list.

PROCESSING: The system will save the provided data to the database.

### Approve Crops

DESCRIPTION: The system provides facility to approve a new crops.

INPUT: Enter crops approval related information.

OUTPUT: Report and generation of notification.

PROCESSING: The system will save the provided data to the database

### Reject Crops

DESCRIPTION: The system provides facility to reject a new crops.

INPUT: Enter crops rejection related information.

OUTPUT: Report and generation of notification.

PROCESSING: The system will save the provided data to the database

## NONFUNCTIONAL REQUIREMENTS

The software was designed to fulfill the following non-functional requirements.

- Performance Requirements: the system will run on high performance with quick response when the user has got a good working internet connection.
- Portability: the system will run with a centralized warehouse of data and accessible over the World Wide Web, this makes it also to run in various number of operating system systems as it will be supported on windows (7,8,10), Mac OS accessible through different browsers including Mozilla Firefox, Google Chrome, Opera Mini, Safari, Microsoft Edge as well as UC Browser
- Security: a strong password will be required before entering the system and use it.
- Usability: the system will be user friendly
- Adaptability: the finished software will support new user types without needing to be written or recompiled.

- Reliability: the system will be available to use if the user has internet. The lower the internet functions, the higher the response time will be.

## USE CASE DIAGRAM

A use case diagram is a graphic depiction of the interactions among the elements of a system.

A use case is a methodology used in system analysis to identify, clarify, and organize system requirements. In this context, the term "system" refers to something being developed or operated, such as a mail-order product sales and service Web site. Use case diagrams are employed in UML (Unified Modeling Language), a standard notation for the modeling of real-world objects and systems. (web programming notes, 2014)

### USE CASE DIAGRAM



Figure 1: Use Case Diagram

## DATA FLOW DIAGRAM

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system, modelling its Process

Aspects. A DFD is often used as a preliminary step to create an overview of the system, which can later be elaborated. DFDs can also be used for the visualization of data processing (structured design). A DFD shows what kind of information will be input to and output from the system, where the data will come from and go to, and where the data will be stored. It does not show information about the timing of process or information about whether processes will operate in sequence or in parallel.

### CONTEXT DIAGRAM (LEVEL 0) DFD

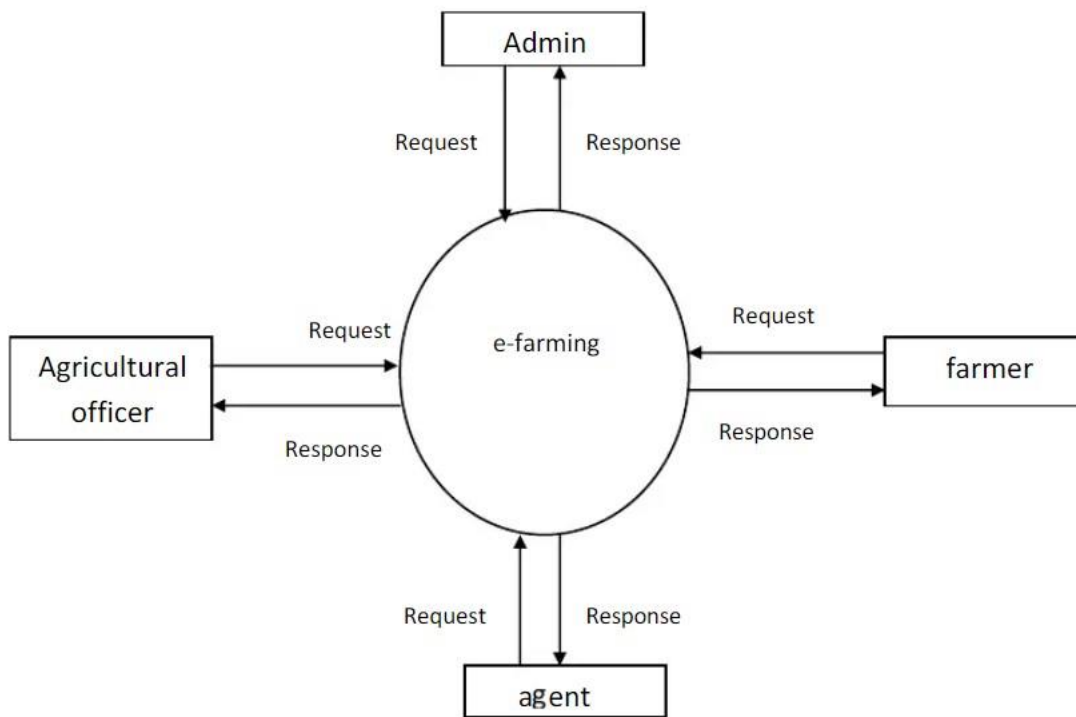


Figure 2: Level 0



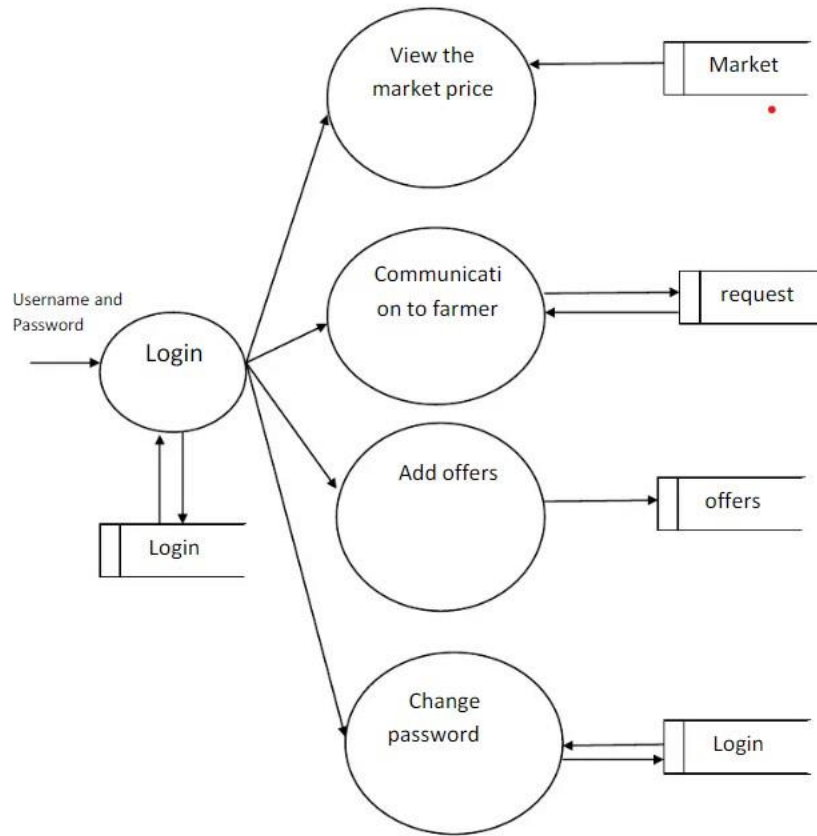


Figure 3: Level 1

# LEVEL 1 DIAGRAM FOR FARMER

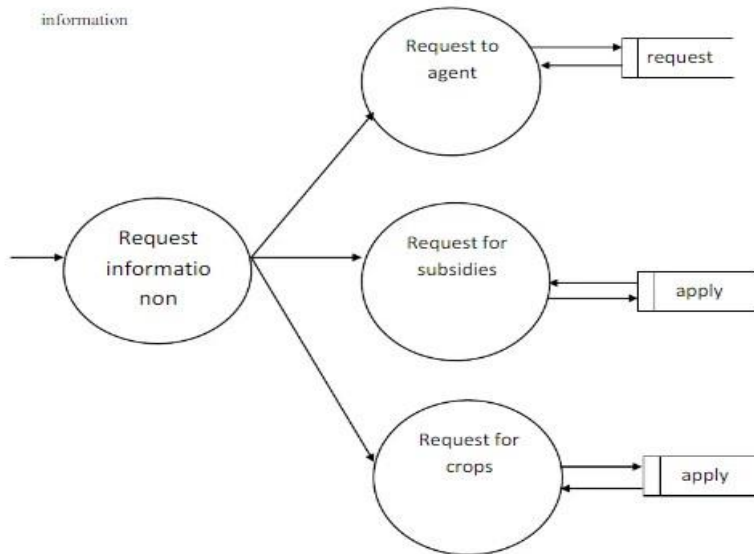
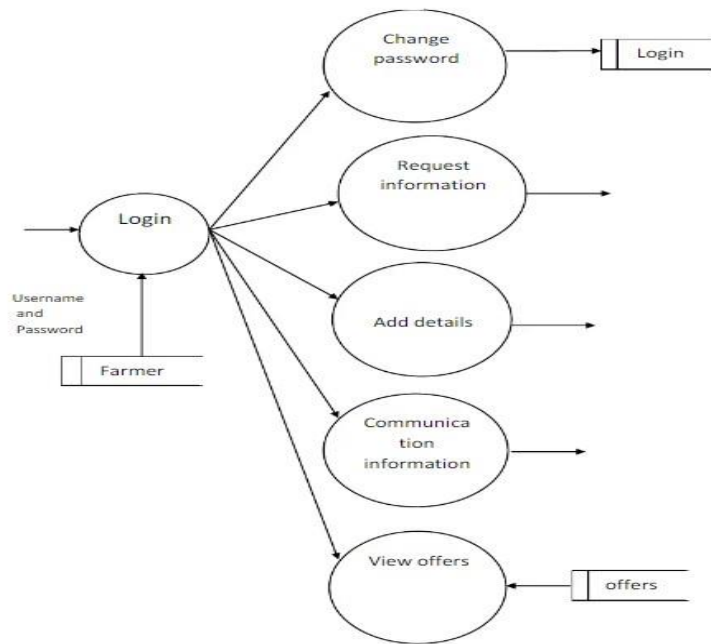


Figure 4: Level 2

## SEQUENCE DIAGRAM

Sequence diagrams describe interactions among classes in terms of an exchange of messages over time. They're also called event diagrams. A sequence diagram is a good way to visualize and validate various runtime scenarios. These can help to predict how a system will behave and to discover responsibilities a class may need to have in the process of modeling a new system.

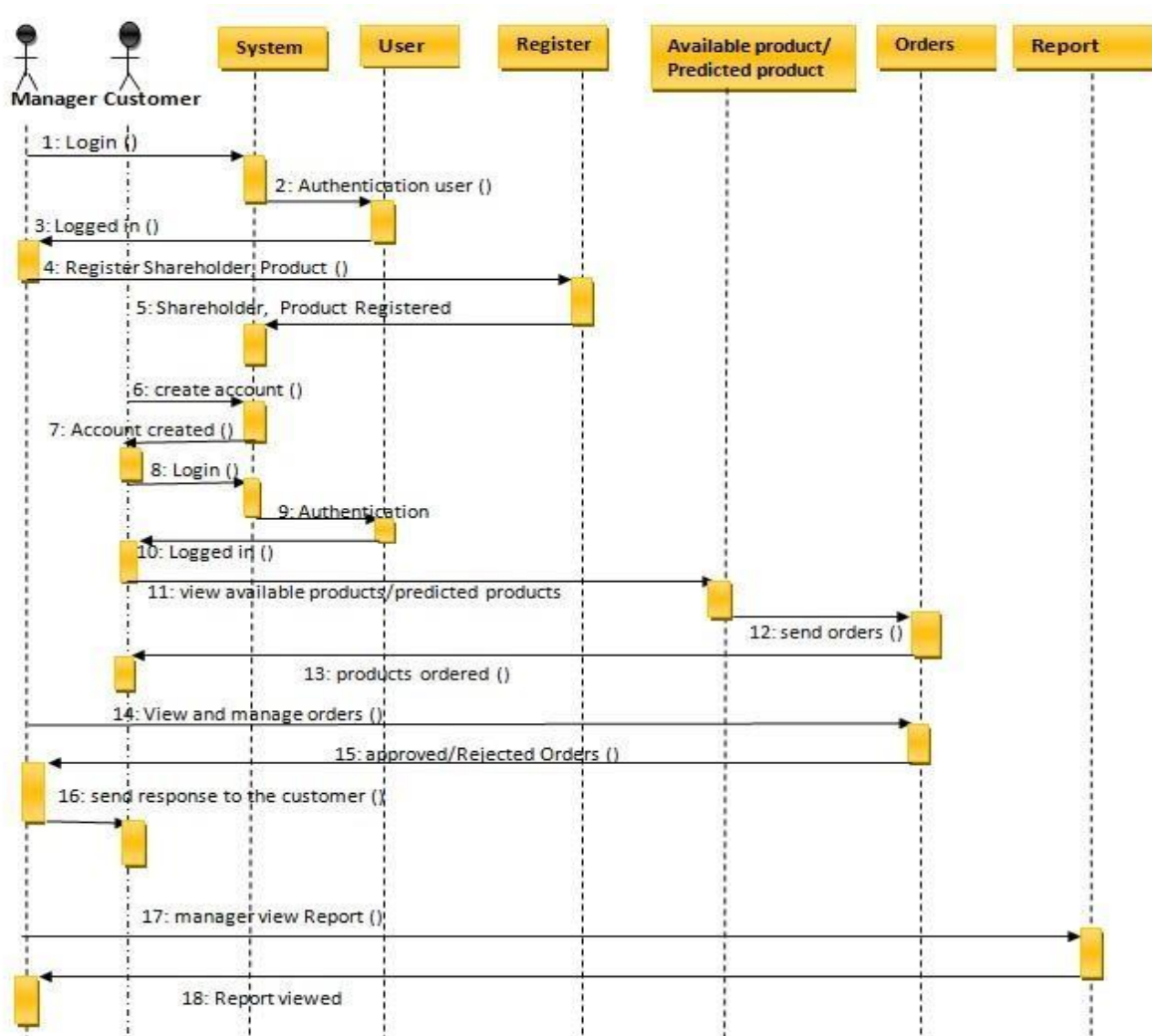


Figure 5: Sequence diagram

## CLASS DIAGRAM

According to Laurie Williams (2004) Class diagrams are used in both the analysis and the design phases. During the analysis phase, a very high-level conceptual design is created. At this time, a class diagram might be created with only the class names shown or possibly some pseudo code like phrases may be added to describe the responsibilities of the class. The class diagram created during the analysis phase is used to describe the classes and relationships in the problem domain, but it does not suggest how the system is implemented. By the end of the design phase, class diagrams that describe how the system to be implemented should be developed. The class diagram created after the design phase has detailed implementation information, including the class names, the methods and attributes of the classes, and the relationships among classes.

The below class diagram shows a collection of classes, interfaces, associations, collaborations and constraints of our system.

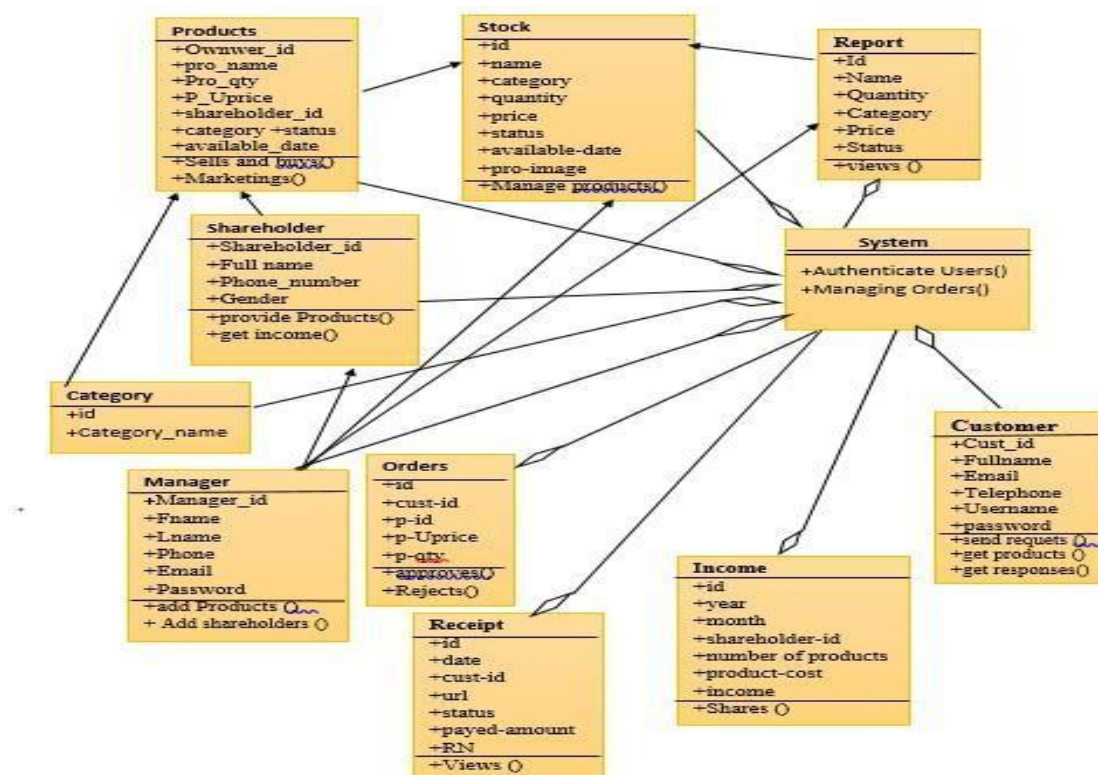


Figure 6: Class diagram

## FEASIBILITY STUDY

### TECHNICAL FEASIBILITY

The technical feasibility of marketing a mobile application for farmers would depend on several factors. Here are some key considerations:

1. **Platform compatibility:** You would need to ensure that the mobile application is compatible with the major mobile operating systems, such as iOS and Android. This would require developing separate versions of the app or using cross-platform frameworks to build a single app that works on multiple platforms.
2. **Internet connectivity:** Farmers may not always have access to a stable internet connection, especially in rural areas. The app should be designed to handle intermittent or low-bandwidth connections and offer offline functionality whenever possible. Additionally, you may need to consider providing support for SMS or USSD-based communication for users with limited internet access.
3. **User interface and user experience:** The app should have an intuitive and user-friendly interface that is easy for farmers to navigate and understand. Consider the technological literacy of the target audience and design the app accordingly, keeping it simple and straightforward.
4. **Data management and storage:** The app may need to handle a significant amount of data related to farmers' profiles, products, transactions, and other relevant information. You would need to plan for efficient data management, including database design, scalability, and security measures to protect users' data.
5. **Integration with existing systems:** Farmers might already be using various tools or systems for managing their farming activities. It would be beneficial to integrate your app with these existing systems, such as agricultural databases, weather services, or marketplaces, to provide a comprehensive solution and streamline their workflows.
6. **Notifications and alerts:** The app can leverage push notifications or SMS alerts to keep farmers informed about important updates, market trends, weather conditions, or relevant news. Ensuring timely and accurate delivery of notifications would be crucial for the app's effectiveness.
7. **Backend infrastructure:** Building and maintaining the required backend infrastructure to support the app's functionalities, such as user authentication, data synchronization, and analytics, is essential. You would need to consider the scalability and reliability of your server architecture to handle potential user growth.
8. **Security and privacy:** Farmers may be concerned about the privacy and security of their data. Implement robust security measures, including encryption, secure data transfer, and user authentication, to protect sensitive information and gain users' trust.
9. **Continuous updates and support:** Mobile app development is an ongoing process, and you should be prepared to release regular updates to address bug fixes, add new features, and adapt to evolving user needs. Additionally, providing responsive customer support channels would be beneficial for user satisfaction.

## FINANCIAL FEASIBILITY

The financial feasibility of marketing a mobile application for farmers depends on several factors, including the market potential, revenue streams, cost structure, and competition. Here are some considerations for assessing the financial feasibility of such a project:

1. **Market Potential:** Evaluate the size of the target market for the mobile application. Research the number of farmers who would benefit from the app and their willingness to adopt technology. Assessing the market potential helps determine the revenue generation capacity.
2. **Revenue Streams:** Identify potential revenue streams for the mobile application. This could include subscription fees, in-app purchases, advertising, or partnerships with agricultural input suppliers or equipment manufacturers. Estimate the revenue each stream can generate and evaluate their feasibility.
3. **Cost Structure:** Determine the development and maintenance costs of the mobile application. Consider factors like app design, software development, hosting, updates, and customer support. Assess the costs involved in marketing and promoting the app to reach the target audience effectively.
4. **Competitor Analysis:** Identify existing mobile applications or digital solutions catering to farmers. Analyze their offerings, pricing models, and user feedback. Assess the competitiveness of your mobile app and identify unique features that differentiate it from competitors.
5. **User Acquisition and Retention:** Evaluate the strategies and costs associated with acquiring new users for the mobile application. Consider the marketing channels, user acquisition costs, and user retention tactics such as regular updates, customer support, and engaging content. Assess the expected user growth rate over time.
6. **Monetization Timeline:** Determine the expected time it will take for the mobile application to generate revenue and potentially break even. Consider the investment required upfront and the projected revenue growth rate. It's crucial to have a clear understanding of when the project is expected to become financially viable.
7. **Financial Projections:** Develop financial projections that incorporate revenue and expense estimates. Consider variables such as user adoption rates, pricing, operating costs, and customer acquisition costs. Conduct sensitivity analysis to assess the impact of changes in key assumptions.
8. **Risk Assessment:** Identify potential risks and challenges that could impact the financial feasibility of the project. These could include factors like market saturation, regulatory changes, technological advancements, or shifts in farmer preferences. Develop contingency plans to mitigate these risks.

## MARKET FEASIBILITY

The feasibility of marketing a mobile application for farmers depends on several factors. Here are some key considerations to assess the market feasibility of such a project:

1. **Target Audience:** Evaluate the size and characteristics of the target audience (farmers) who would benefit from the mobile application. Consider factors such as the number of farmers, their technological adoption, and their willingness to use mobile apps.
2. **Market Demand:** Research the existing market demand for mobile applications catering to farmers. Identify any gaps or unmet needs that your application can fulfill. Analyze the competition and determine if there is room for a new entrant in the market.
3. **Value Proposition:** Clearly define the value proposition of your mobile application. Identify the specific problems or challenges faced by farmers that your app can solve. Highlight the unique features, benefits, and advantages of your application over existing solutions.
4. **Technological Infrastructure:** Assess the technological infrastructure available in the target market. Consider factors such as internet connectivity, smartphone penetration, and compatibility of your application with different devices and operating systems.
5. **Monetization Strategy:** Develop a viable monetization strategy for your mobile application. Explore different revenue models such as paid downloads, in-app purchases, subscriptions, or advertising. Analyze the potential revenue streams and determine if they align with the market dynamics.
6. **Regulatory and Legal Considerations:** Investigate any regulatory or legal requirements related to the agricultural sector and mobile applications in the target market. Ensure that your application complies with relevant regulations and laws.
7. **Marketing and Distribution Channels:** Define an effective marketing and distribution strategy for your mobile application. Identify the most suitable channels to reach farmers, such as agricultural events, farmer associations, digital marketing, or partnerships with agricultural input providers.
8. **Financial Viability:** Conduct a financial analysis to assess the economic feasibility of your project. Estimate the development costs, operational expenses, expected revenue, and projected return on investment. Evaluate the financial sustainability of the project in the long term.
9. **User Feedback and Validation:** Gather feedback from potential users and conduct pilot tests or focus groups to validate the concept and usability of your mobile application. Incorporate user feedback to enhance the application's features and functionality.
10. **Scalability and Growth Potential:** Evaluate the scalability and growth potential of your project. Assess if the mobile application can be expanded to serve a larger market, cater to different types of farmers, or offer additional services over time.

## ECONOMIC FEASIBILITY

To assess the economic feasibility of marketing a mobile application for farmers, several factors need to be considered. Here are some key aspects to evaluate:

1. **Target Market Size:** Determine the size of the target market, including the number of farmers who could potentially benefit from the mobile application. Consider the region, type of farming, and the availability of smartphones among the target audience.
2. **Market Demand:** Assess the demand for such a mobile application among farmers. Conduct surveys, interviews, or market research to gather insights on farmers' pain points, needs, and their willingness to adopt new technology.
3. **Competitive Analysis:** Identify existing mobile applications or digital solutions catering to farmers. Evaluate their features, pricing models, market share, and customer satisfaction. This analysis will help you understand the competitive landscape and position your application accordingly.
4. **Revenue Model:** Define a revenue model for the mobile application. Possible approaches include charging a one-time download fee, offering a subscription-based model, or generating revenue through in-app purchases or advertisements. Consider what pricing strategy would be viable for your target market.
5. **Development and Maintenance Costs:** Estimate the costs associated with developing, launching, and maintaining the mobile application. This includes expenses for app development, hosting, ongoing updates, customer support, and marketing efforts.
6. **Return on Investment (ROI):** Calculate the potential return on investment by estimating the revenue generated from the application and subtracting the development and maintenance costs. Consider the projected time it will take to recoup the initial investment and start generating profits.
7. **Marketing and Distribution:** Determine the marketing and distribution channels to reach the target audience effectively. This could involve digital marketing, partnering with agricultural associations or cooperatives, attending trade shows or farmer-focused events, and utilizing social media platforms.
8. **Value Proposition:** Clearly define the unique value proposition of your mobile application. Identify the key features and benefits it offers to farmers, such as real-time market prices, weather updates, pest management information, or crop advisory services. Highlighting these advantages will help attract and retain users.
9. **Adoption Rate and User Retention:** Assess the potential adoption rate of the mobile application and consider strategies to improve user retention. This could involve offering regular updates, personalized recommendations, user-friendly interfaces, and prompt customer support.
10. **Risk Assessment:** Evaluate potential risks and challenges associated with marketing a mobile application for farmers. These may include technological limitations, resistance to change, regulatory barriers, or the need for continuous updates to stay relevant.



## LEGAL AND REGULATORY FEASIBILITY

The legal and regulatory feasibility of marketing a mobile application for farmers can vary depending on the jurisdiction and specific regulations in place. However, I can provide you with a general overview of some key considerations that may apply in many regions. Please note that I'm not a legal expert, and it's always advisable to consult with a legal professional who is familiar with the laws and regulations in your specific area before launching any marketing initiatives.

1. **Privacy and Data Protection:** Mobile applications often collect and process personal data from users. You need to ensure that your app complies with applicable data protection laws, such as the General Data Protection Regulation (GDPR) in the European Union or the California Consumer Privacy Act (CCPA) in the United States. This may involve obtaining user consent for data collection and implementing appropriate security measures to protect the data.
2. **Intellectual Property:** Make sure that the content and features of your mobile application do not infringe on any existing patents, copyrights, or trademarks. Conduct a thorough search to identify any potential intellectual property conflicts and take necessary steps to mitigate them.
3. **Agricultural Regulations:** Depending on the region, there may be specific regulations governing the marketing and sale of agricultural products or services. Ensure that your mobile application and any associated services comply with these regulations, such as licensing requirements or restrictions on advertising certain products or making health claims.
4. **Advertising and Marketing Regulations:** Be aware of advertising and marketing regulations that may apply to your mobile application. This includes regulations related to false or misleading advertising, unfair competition, and consumer protection. Ensure that any claims made in your marketing materials are accurate, substantiated, and comply with relevant regulations.
5. **Financial Regulations:** If your mobile application involves financial transactions, such as facilitating payments between farmers and buyers, you may need to comply with financial regulations, such as anti-money laundering (AML) and know-your-customer (KYC) requirements. Depending on the jurisdiction, you may also need to obtain appropriate licenses or work with licensed financial institutions.
6. **Terms of Service and End-User Agreements:** Draft clear and comprehensive terms of service and end-user agreements for your mobile application. These agreements should outline the rights and responsibilities of both the users and your company. It's important to have users agree to these terms before they can use your application.
7. **Consumer Protection:** Ensure that your mobile application and marketing practices comply with consumer protection laws. This includes being transparent about the features, costs, and limitations of your application, as well as providing clear information about refunds, warranties, and any applicable guarantees.

## OPERATIONAL FEASIBILITY

The operational feasibility of marketing a mobile application for farmers depends on several factors. Here are some key considerations:

1. **User Adoption:** The success of a mobile application relies on its adoption by the target users, in this case, farmers. It's important to assess whether farmers are technologically inclined and open to using mobile applications. Conducting surveys or interviews with farmers can help gauge their willingness to adopt such technology.
2. **Access to Smartphones and Internet:** Since it's a mobile application, farmers must have access to smartphones and a reliable internet connection. Evaluate the availability of these resources among the target farmer population. If access is limited, it might be necessary to explore alternative approaches such as SMS-based solutions or partnerships with local organizations to provide access.
3. **User Interface and Ease of Use:** The mobile application should have a user-friendly interface that is easy for farmers to navigate and understand. Consider the level of technical literacy among the target users and design the application accordingly. Conduct usability testing to identify and address any potential usability issues.
4. **Local Language and Localization:** Farmers may have varying levels of proficiency in the dominant language or prefer to use applications in their local language. Ensure that the mobile application supports localization and provides content in languages commonly spoken by the farmers in the target region.
5. **Training and Support:** Providing adequate training and support to farmers is crucial for successful adoption. Develop user guides, conduct training sessions, and establish channels for farmers to seek assistance or report issues. This support system will help farmers overcome any initial challenges and build confidence in using the application.
6. **Integration with Existing Systems:** Evaluate whether the mobile application needs to integrate with existing systems or platforms used by farmers, such as marketplaces or weather forecasting services. Seamless integration can enhance the utility of the application and provide a holistic solution for farmers.
7. **Scalability and Maintenance:** Consider the long-term scalability and maintenance requirements of the mobile application. As the user base grows, the application should be able to handle increased traffic and provide timely updates. Plan for regular maintenance and bug fixes to ensure the application remains functional and up-to-date.
8. **Marketing and Promotion:** To ensure the success of the mobile application, create a comprehensive marketing and promotion strategy. Identify the most effective channels to reach farmers, such as agricultural fairs, local community organizations, or online platforms. Collaborate with relevant stakeholders, including agricultural associations, NGOs, and government agencies, to leverage their networks and support in promoting the application.

## DESCRIPTION OF SELECTED PROCESS MODEL

One selected process model for marketing a mobile application for farmers could be the Agile methodology, specifically the Scrum framework. Agile methodologies are often used in software development projects, including mobile application development, due to their iterative and flexible nature. Here's a description of how the Scrum framework can be applied to the marketing process for a mobile application targeting farmers:

1. **Product Backlog:** The marketing team creates a product backlog, which is a prioritized list of marketing tasks, features, and goals for the mobile application. This includes defining the target audience, identifying key messaging, determining marketing channels, and setting performance metrics.
2. **Sprint Planning:** The marketing team, along with other stakeholders such as developers, designers, and product managers, conducts sprint planning meetings to define the objectives for the upcoming sprint. The team selects a set of marketing tasks from the product backlog that can be accomplished within the sprint's time frame (usually 1-4 weeks).
3. **Sprint Execution:** The marketing team works on the selected tasks during the sprint. This can include activities such as creating marketing materials (e.g., website content, social media posts, videos), optimizing app store listings, conducting user research, running advertising campaigns, and organizing promotional events.
4. **Daily Scrum:** The marketing team holds brief daily meetings to synchronize their work, discuss progress, identify any obstacles, and adjust their plans accordingly. This ensures that everyone is aligned and can address any issues or changes in a timely manner.
5. **Sprint Review:** At the end of each sprint, the marketing team presents their accomplishments to the stakeholders, which can include completed marketing tasks, campaign results, and any insights or feedback gathered during the sprint. This allows for continuous feedback and collaboration, ensuring that the marketing efforts are aligned with the overall project goals.
6. **Sprint Retrospective:** After the sprint review, the marketing team holds a retrospective meeting to reflect on the sprint process and identify areas for improvement. This can involve discussing what went well, what could be improved, and any adjustments needed for future sprints. It helps the team refine their marketing strategies and optimize their approach.
7. **Repeat:** The process continues with subsequent sprints, iterating and refining the marketing efforts based on feedback and insights gained from each sprint.

## CONCLUSION

The implementation of marketing mobile application for farmers was the research that put in practice the development of an automated information system to allow Cooperative and shareholders to get advantages from the usage of information technology. This work helped us to increase our knowledge gained from University of Rwanda Huye Campus in Business Information Technology. According to the time that we had, we wished to focus only to the application of marketing mobile application for farmers in order to facilitate shareholders meet with customers.

The objectives of this research were to help cooperative Manager to view the all requests from customers and enabling the cooperative to get information of shareholder without asking him/her documents, also allow customers to send request and receive response from cooperative after checked by manager. So the developed software proved the achievement of the objectives.

## REFERENCES

- David and John. (2010). *assistant lect David* . chicago: David and John.
- Philippe le Hegaret. ( 2016). *fundamental of web*. chicago: web designer.
- Web developer's notes. (2007). *Web developer's notes*. new york: Web developer's .
- (Sun Microsystem, 2009). (2009). *Sun Microsystem*. new york: Sun Microsystem.
- Burton, Richard Antony. (2014). *Burton, Richard Antony*. britich: Richard.
- INGOZIRARUSHYA Dominic. (2019). *manager*. Rutsiro: cooperative.
- Jacobs, Ian; Walsh, Norman. ( 2009). *developer*. new york: developer.
- java notes. ( 2011). *Haverbeke*. new york: Haverbeke. Web design.
- (2006). *Web designer*. new york: Web designer.
- web programming notes. (2014). *web developer*. randon: web developer.

## Chapter2; DATA BASE DESIGN

### DATA BASE OF THE SYSTEM

#### INTRODUCTION

In this chapter we will be describing database of the system tables inside that database and the way those tables were created in, addition on that in this chapter we will show table views from original tables, the relationship between created tables that are in this database. In database of this system be ready to look at operation used on some table in this database so that you can know what to and what not to do on given entity. Note that to develop this system database there are some material which were used so that we can get on final output of system as was described in chapter one "system analyses". Among these the important one is xampp saver MySQL, so now let together navigate this database system

#### 2.2 Section1;

##### 2.2.1 ENTITIES

1. Describe all the entities and their corresponding attributes.

Here are some entities and their corresponding attributes that might be in a database for a "Marketing Mobile Application for Farmers".

**1. Farmers: this table are include**

This is table is table that will be only created by admin and will hold other system users apart from admin ti will give them email and password that they will use to login.;

- Farmer ID
- First Name
- Last Name
- Contact Number
- Email Address
- Farm Location (GPS coordinates or address)
- Farm Size
- Crop(s) Grown
- Livestock (if applicable)
- Membership Status (e.g., free user, premium user)

**2. Products: this table of product are include;**

- Product ID
- Product Name
- Category (e.g., crops, livestock, equipment)
- Description
- Price
- Unit of Measurement (e.g., kg, liters, units)
- Availability (in stock/out of stock)
- Farmer ID (seller)

**3. Orders: this table of orders are include;**

- Order ID
- Farmer ID (buyer)
- Product ID
- Quantity
- Total Price
- Order Date
- Order Status (e.g., placed, shipped, delivered)

**4. Reviews: this table are include;**

- Review ID
- Product ID
- Farmer ID (reviewer)
- Rating (e.g., 1-5 stars)
- Comment/Feedback

**5. Messaging: this table are include;**

- Message ID
- Sender ID
- Receiver ID
- Message Content
- Timestamp
- Read Status

#### **6. Payment Transactions:**

- Transaction ID
- Order ID
- Payment Method (e.g., credit card, mobile money)
- Amount
- Payment Status (e.g., pending, completed)
- Payment Date

#### **7. Farmers' Associations:**

- Association ID
- Association Name
- Location
- Contact Person
- Contact Email
- Contact Phone

#### **8. Promotions and Discounts:**

- Promotion ID
- Product ID (related to the promotion)
- Discount Percentage
- Start Date
- End Date

#### **9. User Accounts:**

- User ID
- Username
- Password
- Account Type (e.g., farmer, admin)
- Profile Picture (URL)
- Last Login Date

#### **10. Analytics and Usage Data:**

- User ID
- Date
- Number of Products Viewed
- Number of Products Purchased
- Search Queries
- App Usage Statistics

### **11. Geolocation Data:**

- Location ID
- Latitude
- Longitude
- Location Name (e.g., nearby markets, weather updates)

### **12. Feedback and Support:**

- Feedback ID
- User ID
- Feedback Type (e.g., bug report, feature request, general feedback)
- Description
- Status (e.g., open, resolved)

2.2.2 this is how Create an LDM (Logical Data Model).

### **Entities and Attributes:**

**In this data Base model they have entities that facility to know entity relations in my data base and to know primary and foreign key**

#### **1. Farmers:**

- FarmerID (Primary Key)
- FirstName
- LastName
- ContactNumber
- Email
- FarmLocation
- FarmSize
- MembershipStatus

#### **2. Products:**

- ProductID (Primary Key)
- ProductName
- Category
- Description
- Price
- UnitOfMeasurement
- Availability
- FarmerID (Foreign Key)

#### **3. Orders:**

- OrderID (Primary Key)
- FarmerID (Foreign Key)



- ProductID (Foreign Key)
  - Quantity
  - TotalPrice
  - OrderDate
  - OrderStatus
4. **Reviews:**
- ReviewID (Primary Key)
  - ProductID (Foreign Key)
  - FarmerID (Foreign Key)
  - Rating
  - Comment
5. **Messaging:**
- MessageID (Primary Key)
  - SenderID (Foreign Key)
  - ReceiverID (Foreign Key)
  - MessageContent
  - Timestamp
  - ReadStatus
6. **PaymentTransactions:**
- TransactionID (Primary Key)
  - OrderID (Foreign Key)
  - PaymentMethod
  - Amount
  - PaymentStatus
  - PaymentDate
7. **FarmersAssociations:**
- AssociationID (Primary Key)
  - AssociationName
  - Location
  - ContactPerson
  - ContactEmail
  - ContactPhone
8. **PromotionsDiscounts:**
- PromotionID (Primary Key)
  - ProductID (Foreign Key)
  - DiscountPercentage
  - StartDate
  - EndDate
9. **UserAccounts:**
- UserID (Primary Key)

- Username
- Password
- AccountType
- ProfilePictureURL
- LastLoginDate

**10. AnalyticsUsageData:**

- AnalyticsID (Primary Key)
- UserID (Foreign Key)
- Date
- ProductsViewed
- ProductsPurchased
- SearchQueries
- AppUsageStatistics

**11. GeolocationData:**

- LocationID (Primary Key)
- Latitude
- Longitude
- LocationName

**12. FeedbackSupport:**

- FeedbackID (Primary Key)
- UserID (Foreign Key)
- FeedbackType
- Description
- Status

**Relationships:**

- Farmers (1) - (N) Products
- Farmers (1) - (N) Orders
- Farmers (1) - (N) Reviews
- Farmers (1) - (N) Messaging (assuming farmers can message each other)
- Farmers (1) - (N) PaymentTransactions
- Farmers (N) - (1) FarmersAssociations
- Products (1) - (N) PromotionsDiscounts
- Users (1) - (N) AnalyticsUsageData
- Users (1) - (N) FeedbackSupport

2.2.3 This how to Create an ERD (Entity-Relationship Diagram).

### **Entities:**

#### **1. Farmers**

- Attributes: FarmerID (Primary Key), FirstName, LastName, ContactNumber, Email, FarmLocation, FarmSize, MembershipStatus
- Relationships:
  - One-to-Many with Products (A farmer can have multiple products listed)
  - One-to-Many with Orders (A farmer can place multiple orders)
  - One-to-Many with Reviews (A farmer can write multiple reviews)
  - One-to-Many with Messaging (A farmer can send/receive multiple messages)
  - One-to-Many with PaymentTransactions (A farmer can have multiple payment transactions)
  - Many-to-One with FarmersAssociations (A farmer can belong to one association)

#### **2. Products**

- Attributes: ProductID (Primary Key), ProductName, Category, Description, Price, UnitOfMeasurement, Availability
- Relationships:
  - Many-to-One with Farmers (Each product is listed by one farmer)
  - One-to-Many with Orders (A product can be in multiple orders)
  - One-to-Many with Reviews (A product can have multiple reviews)
  - One-to-Many with PromotionsDiscounts (A product can have multiple discounts)

#### **3. Orders**

- Attributes: OrderID (Primary Key), Quantity, TotalPrice, OrderDate, OrderStatus
- Relationships:
  - Many-to-One with Farmers (Each order is placed by one farmer)
  - Many-to-One with Products (Each order can include multiple products)
  - Many-to-One with PaymentTransactions (Each order can have multiple payment transactions)

#### **4. Reviews**

- Attributes: ReviewID (Primary Key), Rating, Comment
- Relationships:
  - Many-to-One with Products (Each review is about one product)
  - Many-to-One with Farmers (Each review is written by one farmer)

#### **5. Messaging**

- Attributes: MessageID (Primary Key), MessageContent, Timestamp, ReadStatus

- Relationships:
  - Many-to-One with Farmers (Each message is sent by one farmer and received by another farmer)
- 6. **PaymentTransactions**
  - Attributes: TransactionID (Primary Key), PaymentMethod, Amount, PaymentStatus, PaymentDate
  - Relationships:
    - Many-to-One with Farmers (Each transaction is associated with one farmer)
    - Many-to-One with Orders (Each transaction is associated with one order)
- 7. **FarmersAssociations**
  - Attributes: AssociationID (Primary Key), AssociationName, Location, ContactPerson, ContactEmail, ContactPhone
  - Relationships:
    - One-to-Many with Farmers (An association can have multiple member farmers)
- 8. **PromotionsDiscounts**
  - Attributes: PromotionID (Primary Key), DiscountPercentage, StartDate, EndDate
  - Relationships:
    - Many-to-One with Products (Each promotion is associated with one product)
- 9. **Users**
  - Attributes: UserID (Primary Key), Username, Password, AccountType, ProfilePictureURL, LastLoginDate
  - Relationships:
    - One-to-Many with AnalyticsUsageData (A user can have multiple usage data records)
    - One-to-Many with FeedbackSupport (A user can submit multiple feedback/support requests)
- 10. **AnalyticsUsageData**
  - Attributes: AnalyticsID (Primary Key), Date, ProductsViewed, ProductsPurchased, SearchQueries, AppUsageStatistics
  - Relationships:
    - Many-to-One with Users (Each usage data record is associated with one user)
- 11. **GeolocationData**
  - Attributes: LocationID (Primary Key), Latitude, Longitude, LocationName
- 12. **FeedbackSupport**
  - Attributes: FeedbackID (Primary Key), FeedbackType, Description, Status
  - Relationships:

- Many-to-One with Users (Each feedback/support request is submitted by one user)

## 2.3 Section2:SQL

In this sub unit we will be describing techniques especially SQL queries used to create, delete tables as well as inserting and deleting data in the tables in that database.

### 1. Create the database of your system

```
MariaDB [(none)]> CREATE DATABASE FarmersAppDatabase;
Query OK, 1 row affected (0.002 sec)
```

```
MariaDB [(none)]> USE FarmersAppDatabase;
Database changed
```

### 2. queries to create all the tables and relationships of your system

```
MariaDB [FarmersAppDatabase]> CREATE TABLE Farmers (FarmerID INT
PRIMARY KEY AUTO_INCREMENT, FirstName VARCHAR(50), VARCHAR(50),
ContactNumber VARCHAR(20), Email VARCHAR(100), FarmLocation
VARCHAR(255), FarmSize DECIMAL(10, 2), MembershipStatus
VARCHAR(20));
Query OK, 0 rows affected (0.018 sec)
```

```
MariaDB [FarmersAppDatabase]> CREATE TABLE Products (ProductID
INT PRIMARY KEY AUTO_INCREMENT, ProductName VARCHAR(100),
Category VARCHAR(50), Description TEXT, Price DECIMAL(10, 2),
UnitOfMeasurement VARCHAR(20), Availability BOOLEAN, FarmerID
INT, FOREIGN KEY (FarmerID) REFERENCES Farmers(FarmerID));
Query OK, 0 rows affected (0.049 sec)
```

```
MariaDB [FarmersAppDatabase]> CREATE TABLE Orders (OrderID INT
PRIMARY KEY AUTO_INCREMENT, Quantity INT, TotalPrice DECIMAL(10,
2), OrderDate DATE, OrderStatus VARCHAR(20), FarmerID INT,
FOREIGN KEY (FarmerID) REFERENCES Farmers(FarmerID));
Query OK, 0 rows affected (0.031 sec)
```

```
MariaDB [FarmersAppDatabase]> CREATE TABLE Reviews (ReviewID INT
PRIMARY KEY AUTO_INCREMENT, Rating INT, Comment TEXT, ProductID
INT, FarmerID INT, FOREIGN KEY (ProductID) REFERENCES
Products(ProductID), FOREIGN KEY (FarmerID) REFERENCES
Farmers(FarmerID));
Query OK, 0 rows affected (0.023 sec)
```

```
MariaDB [FarmersAppDatabase]> CREATE TABLE Messaging (MessageID
INT PRIMARY KEY AUTO_INCREMENT, MessageContent TEXT, Timestamp
TIMESTAMP, ReadStatus BOOLEAN, SenderID INT, ReceiverID INT,
```

```
FOREIGN KEY (SenderID) REFERENCES Farmers(FarmerID), FOREIGN KEY  
(ReceiverID) REFERENCES Farmers(FarmerID));  
Query OK, 0 rows affected (0.038 sec)
```

```
MariaDB [FarmersAppDatabase]> CREATE TABLE PaymentTransactions  
(TransactionID INT PRIMARY KEY AUTO_INCREMENT, PaymentMethod  
VARCHAR(50), Amount DECIMAL(10, 2), PaymentStatus VARCHAR(20),  
PaymentDate DATE, OrderID INT, FarmerID INT, FOREIGN KEY (OrderID)  
REFERENCES Orders(OrderID), FOREIGN KEY (FarmerID) REFERENCES  
Farmers(FarmerID));  
Query OK, 0 rows affected (0.032 sec)
```

```
MariaDB [FarmersAppDatabase]> CREATE TABLE FarmersAssociations  
(AssociationID INT PRIMARY KEY AUTO_INCREMENT, AssociationName  
VARCHAR(100), Location VARCHAR(255), ContactPerson VARCHAR(100),  
ContactEmail VARCHAR(100), ContactPhone VARCHAR(20));  
Query OK, 0 rows affected (0.011 sec)
```

```
MariaDB [FarmersAppDatabase]> CREATE TABLE PromotionsDiscounts  
(PromotionID INT PRIMARY KEY AUTO_INCREMENT, DiscountPercentage  
DECIMAL(5, 2), StartDate DATE, EndDate DATE, ProductID INT,  
FOREIGN KEY (ProductID) REFERENCES Products(ProductID));  
Query OK, 0 rows affected (0.030 sec)
```

```
MariaDB [FarmersAppDatabase]> CREATE TABLE Users (UserID INT  
PRIMARY KEY AUTO_INCREMENT, Username VARCHAR(50), Password  
VARCHAR(255), AccountType VARCHAR(20), ProfilePictureURL  
VARCHAR(255), LastLoginDate TIMESTAMP);  
Query OK, 0 rows affected (0.013 sec)
```

```
MariaDB [FarmersAppDatabase]> CREATE TABLE AnalyticsUsageData  
(AnalyticsID INT PRIMARY KEY AUTO_INCREMENT, Date DATE,  
ProductsViewed INT, ProductsPurchased INT, SearchQueries INT,  
AppUsageStatistics TEXT, UserID INT, FOREIGN KEY (UserID)  
REFERENCES Users(UserID));  
Query OK, 0 rows affected (0.032 sec)
```

```
MariaDB [FarmersAppDatabase]> CREATE TABLE GeolocationData  
(LocationID INT PRIMARY KEY AUTO_INCREMENT, Latitude DECIMAL(9,  
6), Longitude DECIMAL(9, 6), LocationName VARCHAR(255));  
Query OK, 0 rows affected (0.013 sec)
```

```
MariaDB [FarmersAppDatabase]> CREATE TABLE FeedbackSupport  
(FeedbackID INT PRIMARY KEY AUTO_INCREMENT, FeedbackType  
VARCHAR(50), Description TEXT, Status VARCHAR(20), UserID INT,  
FOREIGN KEY (UserID) REFERENCES Users(UserID));  
Query OK, 0 rows affected (0.031 sec)
```

### 3. queries to insert data into your tables

```
MariaDB [FarmersAppDatabase]> INSERT INTO Farmers (FirstName,
LastName, ContactNumber, Email, FarmLocation, FarmSize,
MembershipStatus) VALUES ('John', 'Doe', '123-456-7890',
'john@example.com', '123 Main St', 50.5, 'Premium');
Query OK, 1 row affected (0.058 sec)
```

```
MariaDB [FarmersAppDatabase]> INSERT INTO Products (ProductName,
Category, Description, Price, UnitOfMeasurement, Availability,
FarmerID) VALUES ('Wheat', 'Crops', 'High-quality wheat grains',
10.99, 'kg', true, 1);
Query OK, 1 row affected (0.003 sec)
```

```
MariaDB [FarmersAppDatabase]> INSERT INTO Orders (Quantity,
TotalPrice, OrderDate, OrderStatus, FarmerID) VALUES (100,
1099.00, '2023-09-15', 'Completed', 1);
Query OK, 1 row affected (0.003 sec)
```

```
MariaDB [FarmersAppDatabase]> INSERT INTO Reviews (Rating,
Comment, ProductID, FarmerID) VALUES (5, 'Excellent product!', 1,
1);
Query OK, 1 row affected (0.003 sec)
```

```
MariaDB [FarmersAppDatabase]> INSERT INTO PaymentTransactions
(PaymentMethod, Amount, PaymentStatus, PaymentDate, OrderID,
FarmerID) VALUES ('Credit Card', 1099.00, 'Paid', '2023-09-15',
1, 1);
Query OK, 1 row affected (0.002 sec)
```

```
MariaDB [FarmersAppDatabase]> INSERT INTO FarmersAssociations
(AssociationName, Location, ContactPerson, ContactEmail,
ContactPhone) VALUES ('Farmers United', '456 Oak St', 'Jane
Smith', 'jane@example.com', '987-654-3210');
Query OK, 1 row affected (0.003 sec)
```

### 4. queries to display all the information in your tables

```
MariaDB [FarmersAppDatabase]> SELECT * FROM Farmers;
+-----+-----+-----+-----+-----+
| FarmerID | FirstName | LastName | ContactNumber | Email |
| FarmLocation | FarmSize | MembershipStatus |
+-----+-----+-----+-----+-----+
| 1 | John | Doe | 123-456-7890 | john@example.com | 123 Main St | 50.50 | Premium |
```

```
+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+
1 row in set (0.001 sec)
```

```
MariaDB [FarmersAppDatabase]> SELECT * FROM Products;
```

```
+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+
| ProductID | ProductName | Category | Description
| Price | UnitOfMeasurement | Availability | FarmerID |
+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+
|          1 | Wheat       | Crops    | High-quality wheat grains
| 10.99 | kg          |          |          1 |          1 |
+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+
1 row in set (0.000 sec)
```

##### 5. query to update information in any of the two tables of your system

```
MariaDB [FarmersAppDatabase]> UPDATE Farmers SET Email =
'newemail@example.com' WHERE FarmerID = 1;
Query OK, 1 row affected (0.010 sec)
Rows matched: 1 Changed: 1 Warnings: 0
```

## Section III

### 2.3.1 VIEW, TRIGGERS AND PROCEDURES

#### 1. view to insert data into your tables.

```
MariaDB [FarmersAppDatabase]> CREATE VIEW InsertIntoFarmers AS
SELECT * FROM Farmers;
Query OK, 0 rows affected (0.012 sec)
```

```
MariaDB [FarmersAppDatabase]> INSERT INTO InsertIntoFarmers
(FirstName, LastName, ContactNumber, Email, FarmLocation,
FarmSize, MembershipStatus) VALUES ('Alice', 'Johnson', '987-654-
3210', 'alice@example.com', '456 Elm St', 75.5, 'Free');
Query OK, 1 row affected (0.007 sec)
```

#### 2. view to display all the information in your tables.

```
MariaDB [FarmersAppDatabase]> CREATE VIEW AllFarmersInfo AS
SELECT * FROM Farmers;
```



Query OK, 0 rows affected (0.013 sec)

```
MariaDB [FarmersAppDatabase]> SELECT * FROM AllFarmersInfo;
+-----+-----+-----+-----+-----+
+
+
+ FarmerID | FirstName | LastName | ContactNumber | Email
+ FarmLocation | FarmSize | MembershipStatus |
+-----+-----+-----+-----+-----+
+
+
+          1 | John      | Doe      | 123-456-7890 |
newemail@example.com | 123 Main St |      50.50 | Premium
+
+          2 | Alice     | Johnson  | 987-654-3210 |
alice@example.com    | 456 Elm St  |      75.50 | Free
+
+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+
+
+
2 rows in set (0.013 sec)
```

### 3. view to update information in any of the two tables of your system.

```
MariaDB [FarmersAppDatabase]> CREATE VIEW UpdateFarmerInfo AS SELECT
FarmerID, FirstName, LastName, EmailFROM Farmers;
```

Query OK, 0 rows affected (0.013 sec)

```
MariaDB [FarmersAppDatabase]> UPDATE UpdateFarmerInfo SET Email =
'updated_email@example.com'WHERE FarmerID = 1;
```

Query OK, 1 row affected (0.009 sec)

Rows matched: 1 Changed: 1 Warnings: 0

4.view to delete data in any two of your tables according to any simple condition of your choice.

```
MariaDB [FarmersAppDatabase]> CREATE VIEW DeleteOrdersByStatus
ASSELECT OrderID FROM Orders WHERE OrderStatus = 'Cancelled';
```

Query OK, 0 rows affected (0.007 sec)

```
MariaDB [FarmersAppDatabase]> DELETE FROM DeleteOrdersByStatus;
Query OK, 0 rows affected (0.003 sec)
```

5. In your database, create one view of your choice that considers sub-query.

```
MariaDB [FarmersAppDatabase]> CREATE VIEW
FarmersWithHighestFarmSizes AS SELECT FarmerID, FirstName,
LastName, FarmSize FROM Farmers WHERE FarmSize = (SELECT
MAX(FarmSize) FROM Farmers);
Query OK, 0 rows affected (0.014 sec)
```

## Section IV

1. Create a stored procedure to insert data into your tables.

```
MariaDB [FarmersAppDatabase]> DELIMITER //
MariaDB [FarmersAppDatabase]> CREATE PROCEDURE
InsertFarmer(IN p_FirstName VARCHAR(50), IN p_LastName
VARCHAR(50), IN p_ContactNumber VARCHAR(20), IN p_Email
VARCHAR(100), IN p_FarmLocation VARCHAR(255), IN p_FarmSize
DECIMAL(10, 2), IN p_MembershipStatus VARCHAR(20)) BEGIN
INSERT INTO Farmers (FirstName, LastName, ContactNumber,
Email, FarmLocation, FarmSize, MembershipStatus) VALUES
(p_FirstName, p_LastName, p_ContactNumber, p_Email,
p_FarmLocation, p_FarmSize, p_MembershipStatus); END//
Query OK, 0 rows affected (0.043 sec)

MariaDB [FarmersAppDatabase]> DELIMITER ;

MariaDB [FarmersAppDatabase]> CALL InsertFarmer('Alice',
'Johnson', '987-654-3210', 'alice@example.com', '456 Elm
St', 75.5, 'Free');
Query OK, 1 row affected (0.013 sec)
```

2. stored procedure to display all the information in your tables.

```
MariaDB [FarmersAppDatabase]> DELIMITER //
MariaDB [FarmersAppDatabase]> CREATE PROCEDURE
DisplayAllFarmersInfo() BEGIN SELECT * FROM Farmers; END//
Query OK, 0 rows affected (0.011 sec)
MariaDB [FarmersAppDatabase]> DELIMITER ;
MariaDB [FarmersAppDatabase]> CALL DisplayAllFarmersInfo();
```

```

+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+
-----+
| FarmerID | FirstName | LastName | ContactNumber | Email
| FarmLocation | FarmSize | MembershipStatus |
+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+
-----+
|          1 | John      | Doe      | 123-456-7890 |
updated_email@example.com | 123 Main St | 50.50 |
Premium      |
|          2 | Alice     | Johnson  | 987-654-3210 |
alice@example.com          | 456 Elm St | 75.50 | Free
|
|          3 | Alice     | Johnson  | 987-654-3210 |
alice@example.com          | 456 Elm St | 75.50 | Free
|
+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+
-----+
3 rows in set (0.002 sec)

```

Query OK, 0 rows affected (0.099 sec)

### 3. stored procedure to update information in any of the two tables of your

system.

```
MariaDB [FarmersAppDatabase]> DELIMITER //
```

```
MariaDB [FarmersAppDatabase]> CREATE PROCEDURE
UpdateFarmerEmail(IN p_FarmerID INT, IN p_NewEmail VARCHAR(100))
BEGINUPDATE FarmersSET Email = p_NewEmailWHERE FarmerID =
p_FarmerID; END//
```

Query OK, 0 rows affected (0.017 sec)

```
MariaDB [FarmersAppDatabase]> DELIMITER ;
```

```
MariaDB [FarmersAppDatabase]> CALL UpdateFarmerEmail(1,
'updated_email@example.com');
```

Query OK, 0 rows affected (0.002 sec)

**4. stored procedure to delete data in any two of your tables according to any simple condition of your choice.**

```
MariaDB [FarmersAppDatabase]> DELIMITER //
```

```
MariaDB [FarmersAppDatabase]> CREATE PROCEDURE
DeleteOrdersByStatus(IN p_OrderStatus VARCHAR(20)) BEGINDELETE
FROM OrdersWHERE OrderStatus = p_OrderStatus; END//
```

```
Query OK, 0 rows affected (0.021 sec)
```

```
MariaDB [FarmersAppDatabase]> DELIMITER ;
```

```
MariaDB [FarmersAppDatabase]> CALL
DeleteOrdersByStatus('Cancelled');
```

```
Query OK, 0 rows affected (0.001 sec)
```

**5. In your database, stored the procedure view of your choice that considers sub-query.**

```
MariaDB [FarmersAppDatabase]> DELIMITER //
```

```
MariaDB [FarmersAppDatabase]> CREATE PROCEDURE
FarmersWithHighestFarmSizes() BEGIN SELECT FarmerID, FirstName,
LastName, FarmSizeFROM FarmersWHERE FarmSize = (SELECT
MAX(FarmSize) FROM Farmers); END//
```

```
Query OK, 0 rows affected (0.016 sec)
```

```
MariaDB [FarmersAppDatabase]> DELIMITER ;
```

```
MariaDB [FarmersAppDatabase]> CALL
FarmersWithHighestFarmSizes();
```

FarmerID	FirstName	LastName	FarmSize
2	Alice	Johnson	75.50
3	Alice	Johnson	75.50

```
2 rows in set (0.001 sec)
```

Query OK, 0 rows affected (0.028 sec)

## Section V

1. Create after inserting triggers for any two tables of your choice.

```
MariaDB [FarmersAppDatabase]> DELIMITER //
MariaDB [FarmersAppDatabase]> CREATE TRIGGER
AfterInsertFarmer
AFTER INSERT
ON Farmers FOR EACH ROW
BEGIN
INSERT INTO FarmerAudit (Action, FarmerID,
RegistrationDate)
VALUES ('Insert', NEW.FarmerID, NOW());
END//
Query OK, 0 rows affected (0.016 sec)
MariaDB [FarmersAppDatabase]> DELIMITER ;
MariaDB [FarmersAppDatabase]> DELIMITER //
MariaDB [FarmersAppDatabase]> CREATE TRIGGER
AfterInsertProduct
AFTER INSERT
ON Products FOR EACH ROW
BEGIN
UPDATE Farmers
SET ProductCount = ProductCount + 1
WHERE FarmerID = NEW.FarmerID;
END//
Query OK, 0 rows affected (0.009 sec)
MariaDB [FarmersAppDatabase]> DELIMITER ;
```

2. Create after-update triggers for any two tables of your choice.

```
MariaDB [FarmersAppDatabase]> DELIMITER //
MariaDB [FarmersAppDatabase]> CREATE TRIGGER
AfterUpdateFarmer
AFTER UPDATE
ON Farmers FOR EACH ROW
BEGIN
INSERT INTO FarmerAudit (Action, FarmerID, FieldChanged,
OldValue, NewValue, ChangeDate)
```

```

VALUES ('Update', NEW.FarmerID, 'Email', OLD.Email,
NEW.Email, NOW());
END//
Query OK, 0 rows affected (0.021 sec)
MariaDB [FarmersAppDatabase]> DELIMITER ;
MariaDB [FarmersAppDatabase]> DELIMITER //
MariaDB [FarmersAppDatabase]> CREATE TRIGGER
AfterUpdateOrder
AFTER UPDATE
ON Orders FOR EACH ROW
BEGIN
SET NEW.TotalPrice = NEW.Quantity * (SELECT Price FROM
Products WHERE ProductID = NEW.ProductID);
END//
MariaDB [FarmersAppDatabase]> DELIMITER ;

```

**3. after deleting triggers for any two tables of your choice.**

```

MariaDB [FarmersAppDatabase]> DELIMITER //
MariaDB [FarmersAppDatabase]> CREATE TRIGGER
AfterDeleteFarmer
AFTER DELETE
ON Farmers FOR EACH ROW
BEGIN
DELETE FROM Orders WHERE FarmerID = OLD.FarmerID;
END//
Query OK, 0 rows affected (0.022 sec)
MariaDB [FarmersAppDatabase]> DELIMITER ;

MariaDB [FarmersAppDatabase]> DELIMITER //
MariaDB [FarmersAppDatabase]> CREATE TRIGGER
AfterDeleteProduct
AFTER DELETE
ON Products FOR EACH ROW
BEGIN
UPDATE Farmers
SET ProductCount = ProductCount - 1
WHERE FarmerID = OLD.FarmerID;
END//
Query OK, 0 rows affected (0.031 sec)
MariaDB [FarmersAppDatabase]> DELIMITER ;

```

## Section VI

1. user with your name as username and your student number as password and grant all privileges to the created user.

```
MariaDB [FarmersAppDatabase]> CREATE USER 'jacky'@'localhost' IDENTIFIED BY '222008848';
```

```
Query OK, 0 rows affected (0.011 sec)
```

```
MariaDB [FarmersAppDatabase]> GRANT ALL PRIVILEGES ON *.* TO 'jacky'@'localhost';
```

```
Query OK, 0 rows affected (0.005 sec)
```

2. user with your "names\_semi" as username and your student number as password and give him insert, update, and delete privileges to the created user.

```
MariaDB [FarmersAppDatabase]> CREATE USER 'nyiraneza'@'localhost' IDENTIFIED BY '222008848';
```

```
Query OK, 0 rows affected (0.07sec)
```

```
MariaDB [FarmersAppDatabase]> GRANT INSERT, UPDATE, DELETE ON *.* TO 'nyiraneza'@'localhost';
```

```
Query OK, 0 rows affected (0.011 sec)
```

4. Revoke insert privileges to the last user you created.

```
MariaDB [FarmersAppDatabase]> REVOKE INSERT ON *.* FROM 'nyiraneza'@'localhost';
```

### 2.4 CONCLUSION

By concluding on this database we have achieved more on this system using MySQL and the system in terms of keeping data is working well with different operation we have settled on it in some tables like revoking some activities, triggers are created in this database and relationship of much of these tables in this database are there to link them one by one. So this will lead us to

creating the link between data store which is database and the final user of the system which will be focusing more on user interface and connectivity.

### Cpter 3; JAVA PROGRAMMING

#### 3.1 INTRODUCTION

In this chapter I will be describing how powerful generalpurpose programming language was used to create the analyzed system. Under this chapter I will undergo full detail of how everything will function together with database that have been describe above and how it cope with full analyzed system.

3.2 Tools used to develop this system in java programming: Eclipse IDE: an integrated development environment used in computer programming. It contains a base workspace and an extensible plug-in system for customizing the environment. It is the second-most-popular IDE for Java development, and, until 2016, was the most popular.

#### 3.3 Forms description

So let look together how the system will function one by one from the beginning up to the end;

Names ;

Reg Number :

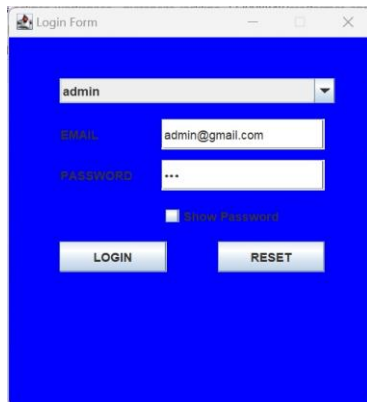
simple description of my project !!

---

my project was done in java with mysql database !  
this project is farmer online/management where used to  
reserve different product in different category like  
crops,livestock,equipment.

this system used by two main users admin and farmer  
ADIMIN LOGIN;

As we have described in above page login form will direct as on this admin login paga.



FARMER LOGIN;



With this page we will be informed what tasks can system farmer do in this system like adding new farmer for those farmer who van not access the system or who do not have knowledge to use the system



A screenshot of a web browser window titled "Login Form". The background is blue. At the top, there is a dropdown menu with "farmer" selected. Below it are two input fields: "EMAIL" with the value "nepo@gmail.com" and "PASSWORD" with four asterisks. A checkbox labeled "Show Password" is below the password field. At the bottom are two buttons: "LOGIN" and "RESET".

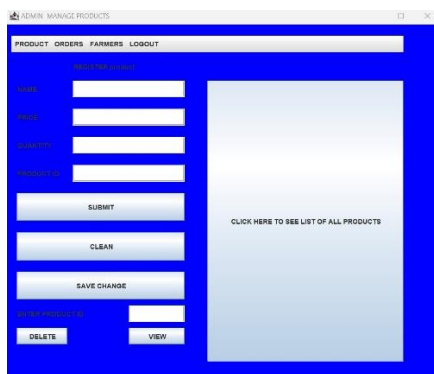
Here are same activities of admin in my project;

admin is able to add and modify different farmers



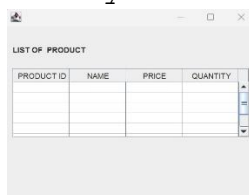
A screenshot of a web browser window titled "ADMIN: MANAGE FARMERS". The background is blue. At the top is a navigation bar with "PRODUCT", "ORDERS", "FARMERS", and "LOGOUT". Below it is a form with fields for "NAME", "EMAIL", "PHONE", "ADDRESS", and "CITY". There are buttons for "SUBMIT", "CLEAN", and "SAVE CHANGE". Below the form is a large light blue area with the text "CLICK HERE TO DISPLAY FARMERS".

admin is able to add and modify different products



A screenshot of a web browser window titled "ADMIN: MANAGE PRODUCTS". The background is blue. At the top is a navigation bar with "PRODUCT", "ORDERS", "FARMERS", and "LOGOUT". Below it is a form with fields for "NAME", "PRICE", "QUANTITY", and "DESCRIPTION". There are buttons for "SUBMIT", "CLEAN", and "SAVE CHANGE". Below the form is a large light blue area with the text "CLICK HERE TO SEE LIST OF ALL PRODUCTS".

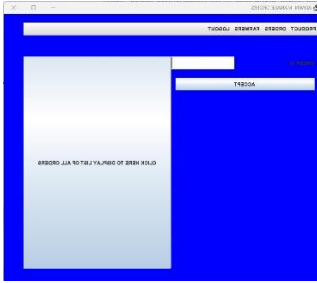
When you do login you can see list of prouduct



A screenshot of a web browser window titled "LIST OF PRODUCT". It displays a table with the following structure:

PRODUCT ID	NAME	PRICE	QUANTITY

admin is able to view and approve different reserves reserved by farmers



second user is farmer where

farmer login in system and reserve to different products and able to view if his reserve approved or rejected by admin



12)users cridentios

This the page on which new created user will use given username and password that are recorded in database called in table called users to login so that he/she can access the system features he/she is allowed to;

admin

email:admin@gmail.com password:000

former

email:nepo@gmail.com password:1234

## how to run my project !

database must be imported for name as it is  
(farmersappdatabase)

project folder must be imported correctly (FarmerApp)

## CONCLUSION

By concluding this chapter concerns with java programming especially in my developed system, we can say that I have final product that I was expecting to have it, the manipulation of data is going well the design is there with special appearance, but there much to go on and that need to be improved will be gained from external view apart from system developer

