

# **POULTRY MANAGEMENT SYSTEM**

*Project Report Submitted By*

**JINTA SUSAN MATHEW**

**Reg. No:AJC16MCA-I32**

*In Partial fulfillment for the Award of the Degree Of*

**INTEGRATED MASTER OF COMPUTER APPLICATIONS  
(INMCA)**

**APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY**



**AMAL JYOTHI COLLEGE OF ENGINEERING  
KANJIRAPPALLY**

[Affiliated to APJ Abdul Kalam Technological University, Kerala. Approved by AICTE, Accredited by NAAC with „A” grade. Koovappally, Kanjirappally, Kottayam, Kerala – 686518]

**2017-2022**

**DEPARTMENT OF COMPUTER APPLICATIONS**  
**AMAL JYOTHI COLLEGE OF ENGINEERING**  
**KANJIRAPPALLY**



**CERTIFICATE**

This is to certify that the Project report, “**Poultry Management System**” is the bonafide work of **Jinta Susan Mathew (Reg.No:AJC16MCA-I32)** in partial fulfillment of the requirements for the award of the Degree of Integrated Master of Computer Applications under APJ Abdul Kalam Technological University during the year 2017-22.

**Ms. Sruthimol Kurian**  
**Internal Guide**

**Ms. Meera Rose Mathew**  
**Coordinator**

**Rev. Fr. Dr. Rubin Thottupurathu Jose**  
**Head of the Department**

**External Examiner**

## **DECLARATION**

I hereby declare that the project report “**POULTRY MANAGEMENT SYSTEM**” is a bonafided work done at Amal Jyothi College of Engineering, towards the partial fulfillment of the requirements for the award of the Degree of Integrated Master of Computer Applications (MCA) from APJ Abdul Kalam Technological University, during the academic year 2017-2022.

**Date:**

**KANJIRAPPALLY**

**JINTA SUSAN MATHEW**

**Reg. No: AJC16MCA-I32**

## ACKNOWLEDGEMENT

First and foremost, I thank God almighty for his eternal love and protection throughout the project. I take this opportunity to express my gratitude to all who helped me in completing this project successfully. It has been said that gratitude is the memory of the heart. I wish to express my sincere gratitude to our manager **Rev. Fr. Dr. Mathew Paikatt** and Principal **Dr. Lillykutty Jacob** for providing good faculty for guidance.

I owe a great depth of gratitude towards our Head of the Department **Rev.Fr. Dr. Rubin Thottupurathu Jose** for helping us. I extend my whole hearted thanks to the **Ms. Meera Rose Mathew** for her valuable suggestions and for overwhelming concern and guidance from the beginning to the end of the project. I would also like to express sincere gratitude to my guide, **Ms. Sruthimol Kurian** for her inspiration and helping hand.

I thank our beloved teachers for their cooperation and suggestions that helped me throughout the project. I express my thanks to all my friends and classmates for their interest, dedication, and encouragement shown towards the project. I convey my hearty thanks to my family for the moral support, suggestions, and encouragement to make this venture a success.

JINTA SUSAN MATHEW

## ABSTRACT

**Poultry Management System** is a desktop application developed in PHP platform. It is a simple and informative system in PHP that manages all the activities of a poultry farm including the breed management, egg production rate, cages for farming, mortality rate, proper feeding and adequate medications. Proposed system is to automate poultry management so that users can track their livestock effectively. If anyone needs to purchase bulky products like hen, eggs or cages by online for setting their own farm, this poultry management system will help them to save their time and cost. Without any intermediate the customer can directly enquire and purchase from the owner. Availability is a major concern in poultry management. So, implementing poultry management system as a website, users can immediately get the whole information about their needs. Like, what breeds they are looking for, when the chicks get vaccinated, what type of feeds are given to chicks and who are the distributors etc. After enquiry one can easily book products by online by paying through online platforms. It will allow the long-distance customers to easily tackle their needs without any time lapse. Also, online booking helps the distributor to schedule the supply time. Poultry Management System comes with wide range of features and functionalities. So, customers can enter by registering. Once registered, they can sign in with credentials and view their page. Also, admin can add and update the details of the cock or chicken and get the notification. Moreover, this is completely safe and fast as it has fully featured software with a user friendly which manages poultry efficiently.

# CONTENT

Sl. No	Topic	Page No
<b>1</b>	<b>INTRODUCTION</b>	<b>1</b>
<b>1.1</b>	<b>PROJECT OVERVIEW</b>	<b>2</b>
<b>1.2</b>	<b>PROJECT SPECIFICATION</b>	<b>2</b>
<b>2</b>	<b>SYSTEM STUDY</b>	<b>4</b>
<b>2.1</b>	<b>INTRODUCTION</b>	<b>5</b>
<b>2.2</b>	<b>EXISTING SYSTEM</b>	<b>6</b>
<b>2.3</b>	<b>DRAWBACKS OF EXISTING SYSTEM</b>	<b>6</b>
<b>2.4</b>	<b>PROPOSED SYSTEM</b>	<b>6</b>
<b>2.5</b>	<b>ADVANTAGES OF PROPOSED SYSTEM</b>	<b>7</b>
<b>3</b>	<b>REQUIREMENT ANALYSIS</b>	<b>9</b>
<b>3.1</b>	<b>FEASIBILITY STUDY</b>	<b>10</b>
<b>3.1.1</b>	<b>ECONOMICAL FEASIBILITY</b>	<b>10</b>
<b>3.1.2</b>	<b>TECHNICAL FEASIBILITY</b>	<b>11</b>
<b>3.1.3</b>	<b>BEHAVIORAL FEASIBILITY</b>	<b>11</b>
<b>3.2</b>	<b>SYSTEM SPECIFICATION</b>	<b>12</b>
<b>3.2.1</b>	<b>HARDWARE SPECIFICATION</b>	<b>12</b>
<b>3.2.2</b>	<b>SOFTWARE SPECIFICATION</b>	<b>12</b>
<b>3.3</b>	<b>SOFTWARE DESCRIPTION</b>	<b>12</b>
<b>3.3.1</b>	<b>PHP</b>	<b>12</b>
<b>3.3.2</b>	<b>MYSQL</b>	<b>13</b>
<b>4</b>	<b>SYSTEM DESIGN</b>	<b>16</b>
<b>4.1</b>	<b>INTRODUCTION</b>	<b>17</b>
<b>4.2</b>	<b>UML DIAGRAM</b>	<b>17</b>
<b>4.2.1</b>	<b>USE CASE DIAGRAM</b>	<b>18</b>
<b>4.2.2</b>	<b>SEQUENCE DIAGRAM</b>	<b>19</b>
<b>4.2.3</b>	<b>CLASS DIAGRAM</b>	<b>23</b>
<b>4.2.4</b>	<b>OBJECT DIAGRAM</b>	<b>24</b>
<b>4.2.5</b>	<b>COLLABORATION DIAGRAM</b>	<b>24</b>
<b>4.2.6</b>	<b>ACTIVITY DIAGRAM</b>	<b>26</b>
<b>4.2.7</b>	<b>STATE CHART DIAGRAM</b>	<b>27</b>
<b>4.2.8</b>	<b>DEPLOYMENT DIAGRAM</b>	<b>28</b>

<b>4.2.9</b>	<b>COMPONENT DIAGRAM</b>	<b>30</b>
<b>4.3</b>	<b>USER INTERFACE DESIGN</b>	<b>32</b>
<b>4.4</b>	<b>DATA BASE DESIGN</b>	<b>35</b>
<b>5</b>	<b>SYSTEM TESTING</b>	<b>47</b>
<b>5.1</b>	<b>INTRODUCTION</b>	<b>48</b>
<b>5.2.1</b>	<b>UNIT TESTING</b>	<b>50</b>
<b>5.2.2</b>	<b>INTEGRATION TESTING</b>	<b>57</b>
<b>5.2.3</b>	<b>VALIDATION TESTING</b>	<b>57</b>
<b>5.2.4</b>	<b>USER ACCEPTANCE TESTING</b>	<b>58</b>
<b>6</b>	<b>IMPLEMENTATION</b>	<b>59</b>
<b>6.1</b>	<b>INTRODUCTION</b>	<b>60</b>
<b>6.2</b>	<b>IMPLEMENTATION PROCEDURES</b>	<b>60</b>
<b>6.2.1</b>	<b>USER TRAINING</b>	<b>61</b>
<b>6.2.2</b>	<b>TRAINING ON APPLICATION SOFTWARE</b>	<b>61</b>
<b>6.2.3</b>	<b>SYSTEM MAINTENANCE</b>	<b>62</b>
<b>7</b>	<b>CONCLUSION &amp; FUTURE SCOPE</b>	<b>63</b>
<b>7.1</b>	<b>CONCLUSION</b>	<b>64</b>
<b>7.2</b>	<b>FUTURE SCOPE</b>	<b>64</b>
<b>8</b>	<b>BIBLIOGRAPHY</b>	<b>65</b>
<b>9</b>	<b>APPENDIX</b>	<b>67</b>
<b>9.1</b>	<b>SAMPLE CODE</b>	<b>68</b>
<b>9.2</b>	<b>SCREEN SHOTS</b>	<b>80</b>

## **List of Abbreviation**

IDE	-	Integrated Development Environment
HTML	-	Hyper Text Markup Language.
CSS	-	Cascading Style Sheet
SQL	-	Structured Query Language
UML	-	Unified Modeling Language



# **CHAPTER 1**

## **INTRODUCTION**

## 1.1 PROJECT OVERVIEW

“POULTRY MANAGEMENT SYSTEM” is a window based application for maintaining and management of the poultry farm. Proposed system is to automate poultry control so that customers can tune their farm animals effectively. If everyone needs to buy cumbersome merchandise like bird, eggs or cages by on line for setting their personal farm, this fowl management gadget will help them to save their time and cost. with none intermediate the customer can without delay enquire and purchase from the proprietor. Availability is a main subject in poultry control. So, enforcing hen control machine as a website, customers can immediately get the whole records approximately their needs. Like what breeds they may be seeking out, whilst the chicks get vaccinated, what sort of feeds is given to chicks and who are the distributors and so forth? After enquiry you can still without difficulty eBook products by using on-line via paying through on-line platforms. it's going to permit the lengthy-distance clients to without difficulty tackle their needs without any time lapse. also, online reserving enables the distributor to agenda the deliver time. poultry control system comes with extensive range of capabilities and functionalities. So, customers can enter by way of registering. as soon as registered, they could sign in with credentials and examine their web page. Additionally, admin can upload and replace the details of the cock or bird and get the notification. furthermore, that is absolutely secure and fast because it has absolutely featured software program with a user- pleasant interface which manages rooster efficaciously.

## 1.2 PROJECT SPECIFICATION

The proposed system helps to automate everything and get notification of whatever they are looking for. However, customer has an option for advanced booking for a batch of chicken. It has been computerized to run the farm easily. Records are maintained in a database. Any query related to requests is generated immediately. Most of the manual works are reduced.

The system includes 7 modules. They are:

**1. Admin Module**

Admin must have a login into this system. He has the overall control of the system. Admin can add, update and even products, categories etc. Admin can view all the registered customer details by adding or rejecting them. Also admin can add staff and view them. Stock Updation is the main functionality of the admin module. ie Admin can increase or decrease stock (product's quantity) with respect to customers' ordering.

**2. Customer Module**

Customer can register and they can view their products, purchase and do secure online payment through payment gateway. Also they have a functionality to track their orders. If a product is out of stock they have an option to notify it. When stock gets updated by admin, customer will get notifications with respect to it.

**3. Distributor Module**

Distributors can register and they can see all the details of the order requests by the customers. They have an option for update the order status. So that, the distributor can study the plan details and requirements of the customer and distribute/deliver products they are looking for.

**4. Farmer Module**

Farmer can order breeds from hatchery and feeds from supplier. When order become success, they pay and have an option for bill generation. They can add birds. Also can view order requests from wholesalers.

**5. Wholesaler Module**

They can order birds from farmer through payment. They can view their updates and also able to download bill.

**6. Supplier Module**

They can add feeds per quantity. Farmers order feeds from supplier. They can view order requests thereby approving, pending/rejecting it.

**7. Hatchery Module**

They can add birds. Farmer can order bulk of birds from them. They can view order requests from farmer.

## **CHAPTER 2**

### **SYSTEM STUDY**

## 2.1 INTRODUCTION

System analysis is a process of gathering and interpreting facts, diagnosing problems and the information to recommend improvements on the system. It is a problem solving activity that requires intensive communication between the system users and system developers. System analysis or study is an important phase of any system development process. The system is studied to the minutes detail and analyzed. The system analyst plays the role of the interrogator and dwells deep into the working of the present system. The system is viewed as a whole and the input to the system are identified. The outputs from the organizations are traced to the various processes. System analysis is concerned with becoming aware of the problem, identifying the relevant and decisional variables, analyzing and synthesizing the various factors and determining an optimal or at least a satisfactory solution or program of action.

A detailed study of the process must be made by various techniques like interviews, questionnaires etc. The data collected by these sources must be scrutinized to arrive to a conclusion. The conclusion is an understanding of how the system functions. This system is called the existing system. Now the existing system is subjected to close study and problem areas are identified. The designer now functions as a problem solver and tries to sort out the difficulties that the enterprise faces. The solutions are given as proposals. The proposal is then weighed with the existing system analytically and the best one is selected. The proposal is presented to the user for an endorsement by the user. The proposal is reviewed on user request and suitable changes are made. This is loop that ends as soon as the user is satisfied with proposal.

Preliminary study is the process of gathering and interpreting facts, using the information for further studies on the system. Preliminary study is problem solving activity that requires intensive communication between the system users and system developers. It does various feasibility studies. In these studies, a rough figure of the system activities can be obtained, from which the decision about the strategies to be followed for effective system study and analysis can be taken.

## 2.2 EXISTING SYSTEM

Existing system is not a fully automated system. In existing system, it was found to be completely manual i.e customers has to approach directly to the shop owner or purchase directly those who are in long distances. It seems to be difficult for them. The proposed system rectifies the drawbacks of the present system.

It is necessary to modify the existing system in order to include additional information and make the system efficient, flexible and secure. The error in one or another way has the nearest possibility to face any problem. Several activities are done in an unordered manner. The different type of record such as feed detail, bird files, medicine details, expenditure detail, labor detail etc are all related to each other. So it is therefore many problems arise during the manipulation of such record in the existing system.

## 2.3 DRAWBACKS OF EXISTING SYSTEM

- Project often delayed with no progress visibility.
- The system is slow, tedious and limited processing of data.
- Cannot ensure availability of products.
- Completely Manual
- Human effort is needed.
- Difficult in report generating
- Time consuming
- Lot of paper work

## 2.4 PROPOSED SYSTEM

The proposed system is defined to meet all the disadvantages of the existing system. It is necessary to have a system that is more user friendly and user attractive for business growth; on such consideration the system is proposed. In our proposed system there is admin who can view all the distributors and customers. It allows customers to view their products and do their transactions by using online payment method. Users of this proposed system are admin, distributor and customer. The aim of proposed system is to develop a system of improved facilities.

The system provides proper security and reduces the manual work. The interfaces for the new system were implemented using HTML, Bootstrap and Java Script. MYSQL was also used for implementing the system database while PHP was used to create interactivity with the database. After the implementation, the new system was then tested and validated. When developing the system, the focus was on making the whole process of Information management in the poultry farm faster, more convenient and efficient for the company. This system automates the current poultry farm management information system in the organization.

## 2.5 ADVANTAGES OF PROPOSED SYSTEM

The system is very simple in design and to implement. The system requires very low system resources and the system will work in almost all configurations. It has got following features:

➤ **User friendly:-**

The purposed system is user friendly because the retrieval and storing of data is fast and data is maintained efficiently. More over the graphical user is interface is provided in the purposed system, which provide user to deal with the system easily.

➤ **Report are easily generated:-**

Report is easily generated in the purposed system so user can generated the report as per the requirement (monthly) or in the middle of the session.

➤ **Very less paper work:-**

The purposed system required very less paper work. All the data is field into the computer immediately and report can be generated through computer.

➤ **Computer operator control:-**

Computer operator control will be there so less chance of error. Moreover storing and retrieving of information is easy. So work can be done spending and in time.

➤ **Speed:-**

Manual system is always time consuming. Because humans are employed to do necessary action to generate report and store records, it always is slow. In case of computerized system the great processing speed of the computer system will surely



help in quickly performing the process.

➤ **Less space requirement:-**

In the traditional system, there is a need to maintain voluminous paper file. They occupy quite a large amount of space. Moreover, their maintenance is extremely difficult. Our system will replace paper files and store the files electrically in the computer's hard disk or in any other electronic storage media. These files are easy to maintain.

➤ **Backup facility:-**

The traditional system offers no backup mechanism. Natural disasters such as flood or fire may cause paper files to be lost once all. However, our computerized model offers proper backup mechanism. Files stored electrically in the hard disk can be backed up in secondary storage devices such as floppy disk or CD-ROM.

➤ **Better security: -**

For data to remain secure, measures must be taken to prevent unauthorized access. Security means that data are protected from various forms of destruction. The system security problem can be divided into four related issues: security, integrity, privacy and confidentiality. Username and password requirement to sign in ensures security. It will also provide data security as we are using the secured databases for maintaining the documents.

➤ **Ensure data accuracy: -**

The proposed system eliminates the manual errors while entering the details of the users during the registration.

➤ **Better service: -**

The system will avoid the burden of hard copy storage. We can also conserve the time and human resources for doing the same task. The data can be maintained for a longer period with no loss of data.

## **CHAPTER 3**

### **REQUIREMENT ANALYSIS**

### 3.1 FEASIBILITY STUDY

Feasibility study is made to see if the project on completion will serve the purpose of the organization for the amount of work, effort and the time that spend on it. Feasibility study lets the developer foresee the future of the project and the usefulness. A feasibility study of a system proposal is according to its workability, which is the impact on the organization, ability to meet their user needs and effective use of resources. Thus, when a new application is proposed it normally goes through a feasibility study before it is approved for development.

The document provides the feasibility of the project that is being designed and lists various areas that were considered very carefully during the feasibility study of this project such as Technical, Economic and Operational feasibilities. The following are its features: -

#### 3.1.1 Economical Feasibility

The developing system must be justified by cost and benefit. Criteria to ensure that effort is concentrated on project, which will give best, return at the earliest. One of the factors, which affect the development of a new system, is the cost it would require.

The following are some of the important financial questions asked during preliminary investigation:

- The costs conduct a full system investigation.
- The cost of the hardware and software.
- The benefits in the form of reduced costs or fewer costly errors.

The proposed system is developed as part of project work, there is no manual cost to spend for the proposed system. Also all the resources are already available, it give an indication of the system is economically possible for development.

The cost of project, POULTRY MANAGEMENT SYSTEM was divided according to the system used, its development cost and cost for hosting the project. According to all the calculations the project was developed in a low cost. As it is completely developed using open source software.

### 3.1.2 Technical Feasibility

The system must be evaluated from the technical point of view first. The assessment of this feasibility must be based on an outline design of the system requirement in the terms of input, output, programs and procedures. Having identified an outline system, the investigation must go on to suggest the type of equipment, required method developing the system, of running the system once it has been designed.

Technical issues raised during the investigation are:

- Does the existing technology sufficient for the suggested one?
- Can the system expand if developed?

The project should be developed such that the necessary functions and performance are achieved within the constraints. The project requires High Resolution Scanning device and utilizes Cryptographic techniques. Through the technology may become obsolete after some period of time, due to the fact that newer version of same software supports older versions, the system may still be used. So there are minimal constraints involved with this project. The system has been developed using PHP in front end and MySQL in server in back end, the project is technically feasible for development. The System used was also of good performance of Processor Intel i5 core; RAM 8GB and, Hard disk 1TB.

### 3.1.3 Behavioral Feasibility

The proposed system includes the following questions:

- Is there sufficient support for the users?
- Will the proposed system cause harm?

The project would be beneficial because it satisfies the objectives when developed and installed. All behavioral aspects are considered carefully and conclude that the project is behaviorally feasible.

POULTRY MANAGEMENT SYSTEM, GUI is simple so that users can easily use it. POULTRY MANAGEMENT SYSTEM is simple enough so that no training is needed.

### 3.2 SYSTEM SPECIFICATION

#### 3.2.1 Hardware Specification

Processor	- Intel core i5
RAM	- 8 GB
Hard disk	- 1 TB

#### 3.2.2 Software Specification

Front End	-	HTML, CSS
Backend	-	MYSQL
Client on PC	-	Windows 7 and above.
Technologies used	-	JS, HTML5, AJAX, JQuery, PHP, CSS

### 3.3 SOFTWARE DESCRIPTION

#### 3.3.1 PHP

PHP is a server-side scripting language designed for web development but also used as a general-purpose programming language. PHP is now installed on more than 244 million websites and

2.1 million web servers. Originally created by Rasmus Ledorf in 1995, the reference implementation of PHP is now produced by the PHP group. While PHP originally stood for personal home page, it now stands for PHP: Hypertext Preprocessor, a recursive acronym. PHP code is interpreted by a web server with a PHP processor module which generates the resulting web page. PHP commands can be embedded directly into a HTML source document rather than calling an external file to process data. It has also evolved to include a command-line interface capability and can be used in standalone incompatible with the GNU General Public License (GPL) due to restrictions on the usage of the term PHP. PHP can be deployed on most web servers and also as a standalone shell on almost every operating system and platform, free of charge.

### 3.3.2 MySQL

MySQL, the most popular Open-Source SQL database management system, is developed, distributed, and supported by Oracle Corporation. The MySQL Web site provides the latest information about MySQL software.

- **MySQL is a database management system.**

A database is a structured collection of data. It may be anything from a simple shopping list to a picture gallery or the vast amounts of information in a corporate network. To add, access, and process data stored in a computer database, you need a database management system such as MySQL Server. Since computers are very good at handling large amounts of data, database management systems play a central role in computing, as standalone utilities, or as parts of other applications.

- **MySQL databases are relational.**

A relational database stores data in separate tables rather than putting all the data in one big storeroom. The database structures are organized into physical files optimized for speed. The logical model, with objects such as databases, tables, views, rows, and columns, offers a flexible programming environment. You set up rules governing the relationships between different data fields, such as one-to-one, one-to-many, unique, required or optional, and “pointers” between different tables. The database enforces these rules, so that with a well- designed database, your application never sees inconsistent, duplicate, orphan, out-of-date, or missing data. The SQL part of “MySQL” stands for “Structured Query Language”. SQL is the most common standardized language used to access databases. Depending on your programming environment, you might enter SQL directly (for example, to generate reports), embed SQL statements into code written in another language, or use a language-specific API that hides the SQL syntax. SQL is defined by the ANSI/ISO SQL Standard. The SQL standard has been evolving since 1986 and several versions exist. In this manual, “SQL92” refers to the standard released in 1992, “SQL: 1999” refers to the standard released in 1999, and “SQL: 2003” refers to the current version of the standard. We use the phrase “the SQL standard” to mean the current version of the SQL Standard at any time.

- **MySQL software is Open Source.**

Open-Source means that it is possible for anyone to use and modify the software. Anybody can download the MySQL software from the Internet and use it without paying anything. If you wish, you may study the source code and change it to suit your needs. The MySQL software uses the GPL (GNU General Public License), to define what you may and may not do with the software in different situations. If you feel uncomfortable with the GPL or need to embed MySQL code into a commercial application, you can buy a commercially licensed version from us. See the MySQL Licensing Overview for more information.

- **The MySQL Database Server is very fast, reliable, scalable, and easy to use.**

If that is what you are looking for, you should give it a try. MySQL Server can run comfortably on a desktop or laptop, alongside your other applications, web servers, and so on, requiring little or no attention. If you dedicate an entire machine to MySQL, you can adjust the settings to take advantage of all the memory, CPU power, and I/O capacity available.

- **MySQL Server works in client/server or embedded systems.**

The MySQL Database Software is a client/server system that consists of a multi-threaded SQL server that supports different back ends, several different client programs and libraries, administrative tools, and a wide range of application programming interfaces (APIs). We also provide MySQL Server as an embedded multi-threaded library that you can link into your application to get a smaller, faster, easier-to-manage standalone product.

## **CHAPTER 4**

### **SYSTEM DESIGN**



## 4.1 INTRODUCTION

Design is the first step into the development phase for any engineered product or system. Design is a creative process. A good design is the key to effective system. The term “design” is defined as “the process of applying various techniques and principles for the purpose of defining a process or a system in sufficient detail to permit its physical realization”. It may be defined as a process of applying various techniques and principles for the purpose of defining a device, a process or a system in sufficient detail to permit its physical realization. Software design sits at the technical kernel of the software engineering process and is applied regardless of the development paradigm that is used. The system design develops the architectural detail required to build a system or product. As in the case of any systematic approach, this software too has undergone the best possible design phase fine tuning all efficiency, performance and accuracy levels. The design phase is a transition from a user-oriented document to a document to the programmers or database personnel. System design goes through two phases of development: Logical and Physical Design.

## 4.2 UML DIAGRAM

UML is a standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems. UML was created by the Object Management Group (OMG) and UML 1.0 specification draft was proposed to the OMG in January 1997.

UML stands for Unified **Modeling Language**. UML is different from the other common programming languages such as C++, Java, COBOL, etc. UML is a pictorial language used to make software blueprints. UML can be described as a general-purpose visual modeling language to visualize, specify, construct, and document software system. Although UML is generally used to model software systems, it is not limited within this boundary. It is also used to model non- software systems as well. For example, the process flows in a manufacturing unit, etc. UML is not a programming language but tools can be used to generate code in various languages using UML diagrams. UML has a direct relation with object-oriented analysis and design. After some

standardization, UML has become an OMG standard. All the elements, relationships are used to make a complete UML diagram and the diagram represents a system. The visual effect of the UML diagram is the most important part of the entire process. All the other elements are used to make it complete. UML includes the following nine diagrams.

- Class diagram
- Object diagram
- Use case diagram
- Sequence diagram
- Collaboration diagram
- Activity diagram
- State chart diagram
- Deployment diagram
- Component diagram

#### 4.2.1 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 modeling of real-world objects and systems.

System objectives can include planning overall requirements, validating a hardware design, testing and debugging a software product under development, creating an online help reference, or performing a consumer-service-oriented task. For example, use cases in a product sales environment would include item ordering, catalog updating, payment processing, and customer relations. A use case diagram contains four components.

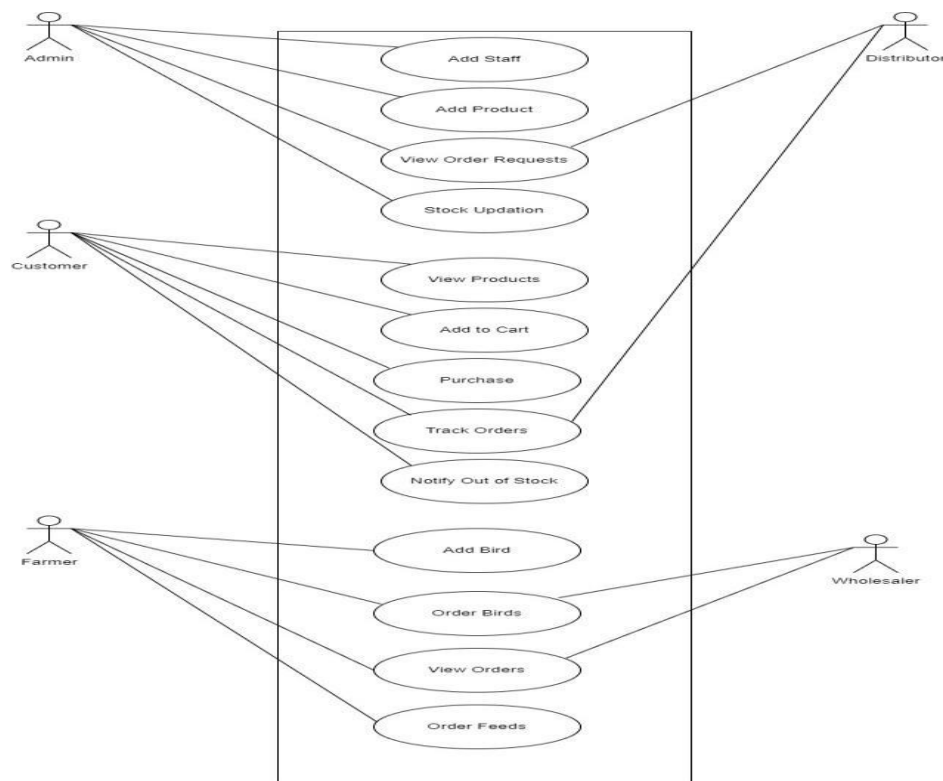
- The boundary, which defines the system of interest in relation to the world around it.
- The actors, usually individuals involved with the system defined according to their roles.

- The use cases, which are the specific roles, are played by the actors within and around the system.
- The relationships between and among the actors and the use cases.

Use case diagrams are drawn to capture the functional requirements of a system. After identifying the above items, we have to use the following guidelines to draw an efficient use case diagram.

- The name of a use case is very important. The name should be chosen in such a way so that it can identify the functionalities performed.
- Give a suitable name for actors.
- Show relationships and dependencies clearly in the diagram.
- Do not try to include all types of relationships, as the main purpose of the diagram is to identify the requirements.
- Use notes whenever required to clarify some important points.

Fig 1: Use case diagram for Poultry Management System



### 4.2.2 SEQUENCE DIAGRAM

A sequence diagram simply depicts interaction between objects in a sequential order i.e., the order in which these interactions take place. We can also use the terms event diagrams or event scenarios to refer to a sequence diagram. Sequence diagrams describe how and in what order the objects in a system function. These diagrams are widely used by businessmen and software developers to document and understand requirements for new and existing systems.

#### Sequence Diagram Notations –

- i. **Actors** – An actor in a UML diagram represents a type of role where it interacts with the system and its objects. It is important to note here that an actor is always outside the scope of the system we aim to model using the UML diagram. We use actors to depict various roles including human users and other external subjects. We represent an actor in a UML diagram using a stick person notation. We can have multiple actors in a sequence diagram.
- ii. **Lifelines** – A lifeline is a named element which depicts an individual participant in a sequence diagram. So basically, each instance in a sequence diagram is represented by a lifeline. Lifeline elements are located at the top in a sequence diagram.
- iii. **Messages** – Communication between objects is depicted using messages. The messages appear in a sequential order on the lifeline. We represent messages using arrows. Lifelines and messages form the core of a sequence diagram.

Messages can be broadly classified into the following categories:

- Synchronous messages
- Asynchronous Messages
- Create message
- Delete Message
- Self-Message
- Reply Message
- Found Message
- Lost Message

**iv. Guards** – To model conditions we use guards in UML. They are used when we need to restrict the flow of messages on the pretext of a condition being met. Guards play an important role in letting software developers know the constraints attached to a system or a particular process.

#### Uses of sequence diagrams –

- Used to model and visualize the logic behind a sophisticated function, operation or procedure.
- They are also used to show details of UML use case diagrams.
- Used to understand the detailed functionality of current or future systems.
- Visualize how messages and tasks move between objects or components in a system.

Fig 2.1: Sequence diagram of Admin for Poultry Management System

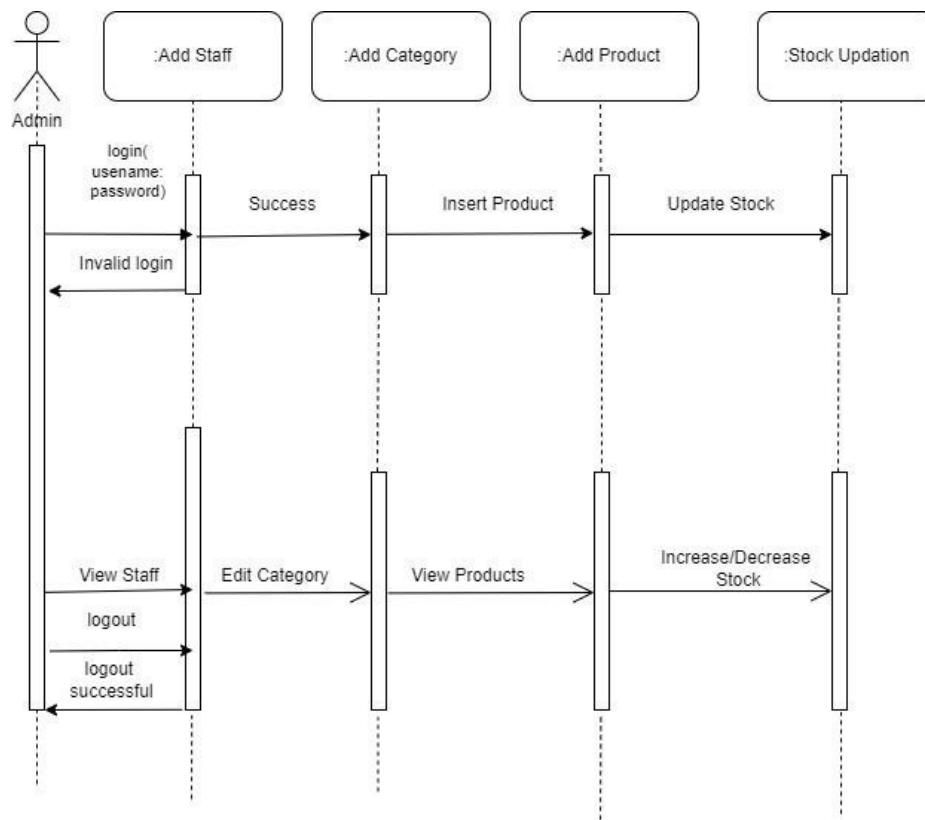


Fig 2.2: Sequence diagram of Customer for Poultry Management System

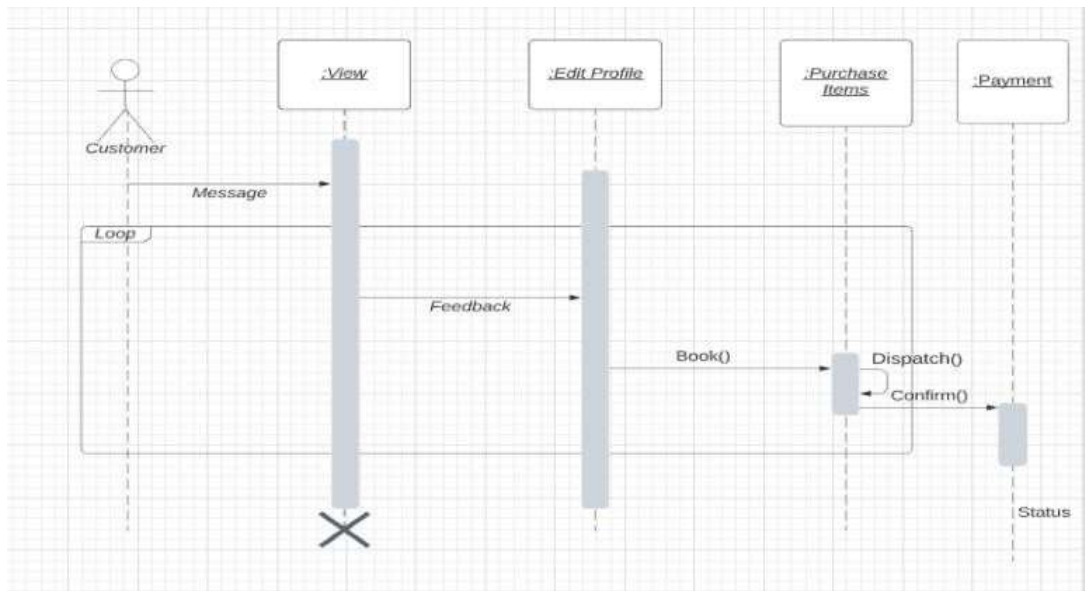
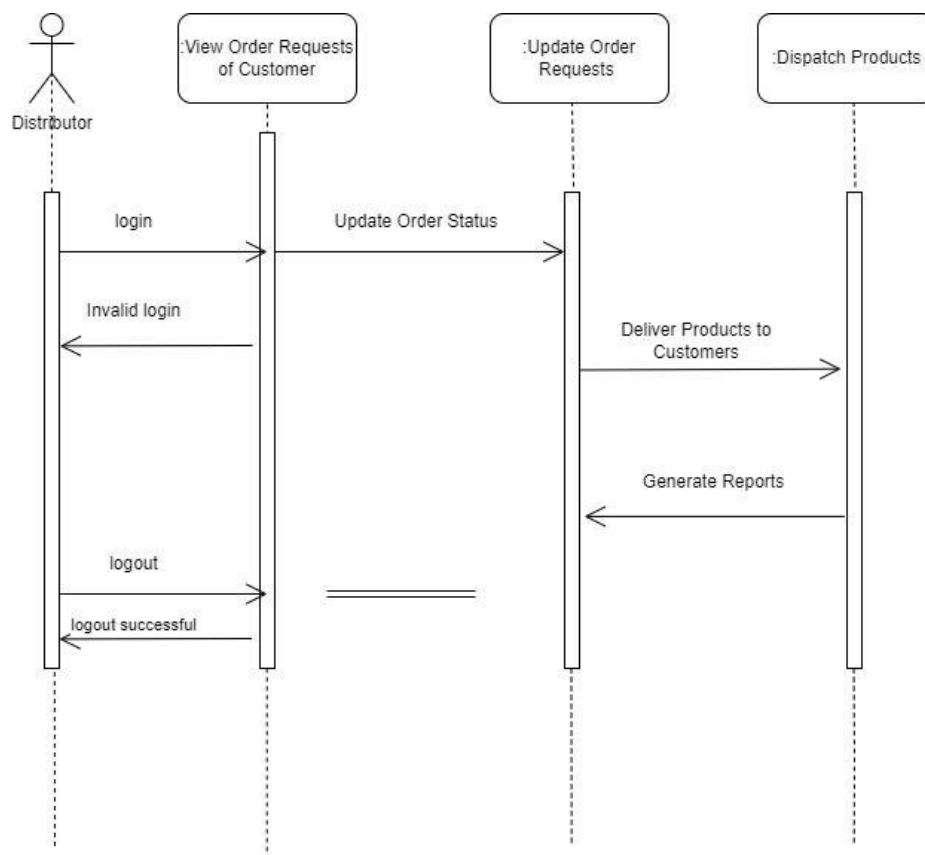


Fig 2.3: Sequence diagram of Distributor for Poultry Management System



### 4.2.3 CLASS DIAGRAM

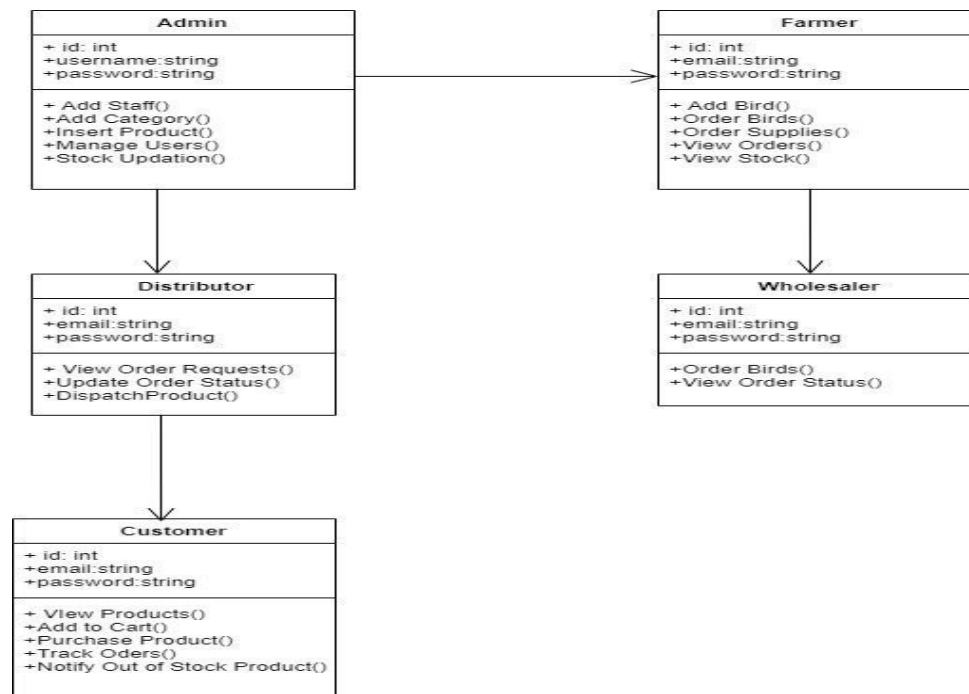
Class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application.

Class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modeling of object-oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages. Class diagram shows a collection of classes, interfaces, associations, collaborations, and constraints. It is also known as a structural diagram.

The purpose of class diagram is to model the static view of an application. Class diagrams are the only diagrams which can be directly mapped with object-oriented languages and thus widely used at the time of construction.

UML diagrams like activity diagram, sequence diagram can only give the sequence flow of the application, however class diagram is a bit different. It is the most popular UML diagram in the coder community.

Fig 3: Class diagram for Poultry Management System



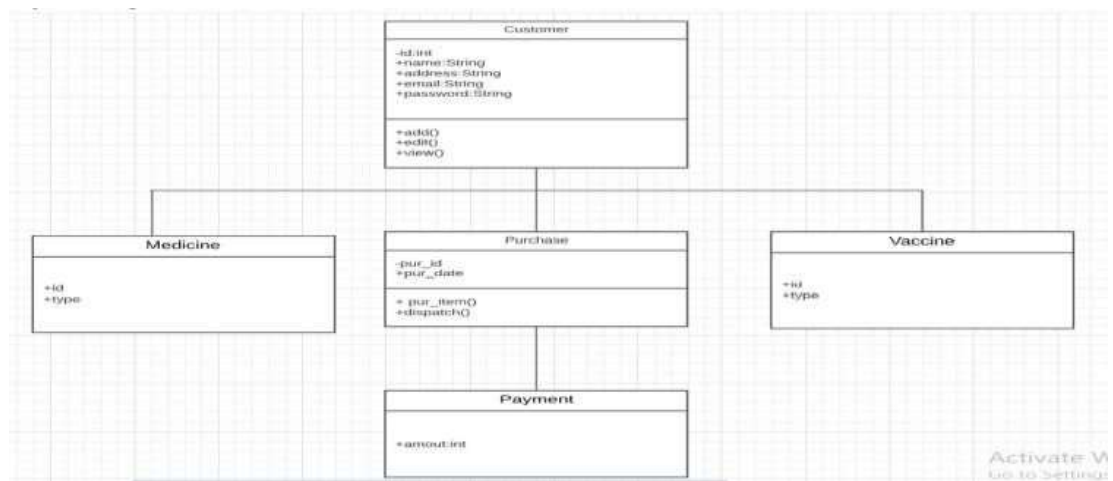
#### 4.2.4 OBJECT DIAGRAM

Object diagrams are derived from class diagrams so object diagrams are dependent upon class diagrams. Object diagrams represent an instance of a class diagram. The basic concepts are similar for class diagrams and object diagrams. Object diagrams also represent the static view of a system but this static view is a snapshot of the system at a particular moment. Object diagrams are used to render a set of objects and their relationships as an instance.

The purpose of a diagram should be understood clearly to implement it practically. The purposes of object diagrams are similar to class diagrams.

The difference is that a class diagram represents an abstract model consisting of classes and their relationships. However, an object diagram represents an instance at a particular moment, which is concrete in nature.

Fig 4: Object diagram for Poultry Management System



#### 4.2.5 COLLABORATION DIAGRAM

A collaboration diagram, also known as a communication diagram, is an illustration of the relationships and interactions among software objects in the Unified Modeling Language (UML).



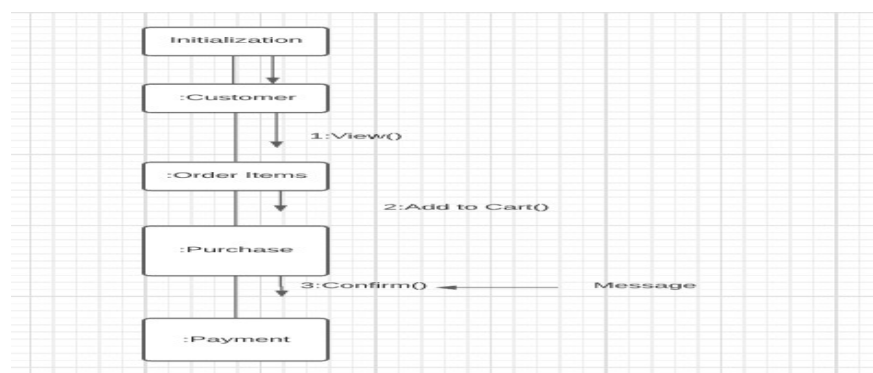
These diagrams can be used to portray the dynamic behavior of a particular use case and define the role of each object.

Collaboration diagrams are created by first identifying the structural elements required to carry out the functionality of an interaction. A model is then built using the relationships between those elements. Several vendors offer software for creating and editing collaboration diagrams.

A collaboration diagram resembles a flowchart that portrays the roles, functionality and behavior of individual objects as well as the overall operation of the system in real time. The four major components of a collaboration diagram are:

1. **Objects**- Objects are shown as rectangles with naming labels inside. The naming label follows the convention of object name: class name. If an object has a property or state that specifically influences the collaboration, this should also be noted.
2. **Actors**- Actors are instances that invoke the interaction in the diagram. Each actor has a name and a role, with one actor initiating the entire use case.
3. **Links**- Links connect objects with actors and are depicted using a solid line between two elements. Each link is an instance where messages can be sent.
4. **Messages**- Messages between objects are shown as a labeled arrow placed near a link. These messages are communications between objects that convey information about the activity and can include the sequence number.

Fig 5: Object diagram for Poultry Management System

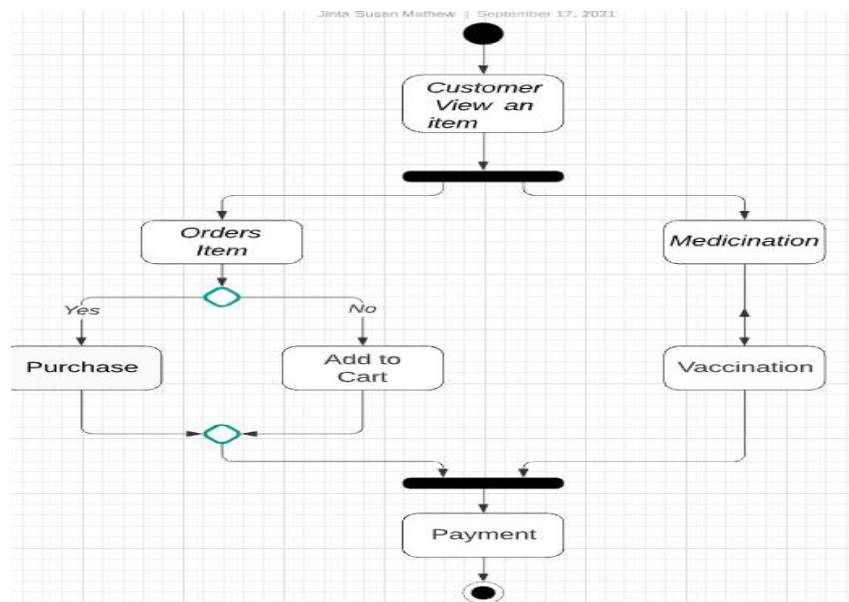


#### 4.2.6 ACTIVITY DIAGRAM

An activity diagram is a **behavioral diagram** i.e., it depicts the behavior of a system. An activity diagram portrays the control flow from a start point to a finish point showing the various decision paths that exist while the activity is being executed. We can depict both sequential processing and concurrent processing of activities using an activity diagram. They are used in business and process modeling where their primary use is to depict the dynamic aspects of a system.

An activity diagram is very similar to a flowchart. So let us understand if an activity diagrams or flowcharts are any different. An activity diagram is used to model the workflow depicting conditions, constraints, sequential and concurrent activities. An activity diagram is used by developers to understand the flow of programs on a high level. It also enables them to figure out constraints and conditions that cause particular events. A flow chart converges into being an activity diagram if complex decisions are being made.

Fig 6: Activity diagram for Poultry Management System



#### 4.2.7 STATE CHART DIAGRAM

A State chart diagram describes a state machine. State machine can be defined as a machine which defines different states of an object and these states are controlled by external or internal events. It describes different states of a component in a system. The states are specific to a component/object of a system. State chart diagram is one of the five UML diagrams used to model the dynamic nature of a system. They define different states of an object during its lifetime and these states are changed by events. State chart diagrams are useful to model the reactive systems. Reactive systems can be defined as a system that responds to external or internal events.

State chart diagram describes the flow of control from one state to another state. States are defined as a condition in which an object exists and it changes when some event is triggered. The most important purpose of State chart diagram is to model lifetime of an object from creation to termination. State chart diagrams are also used for forward and reverse engineering of a system. However, the main purpose is to model the reactive system.

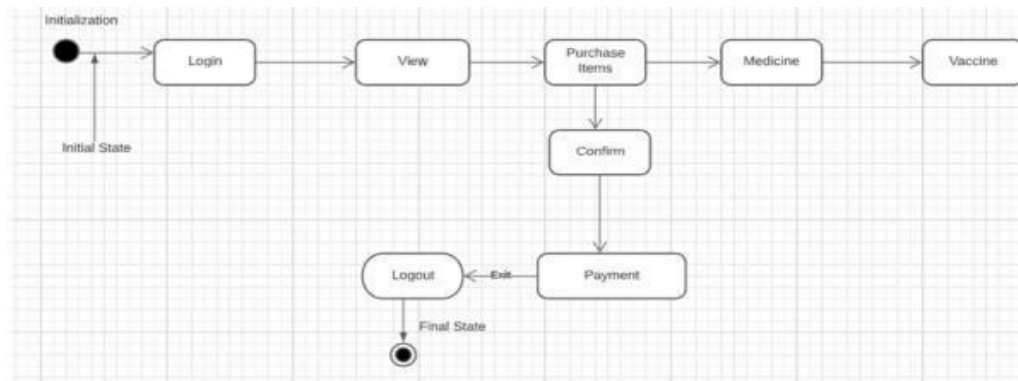
Following are the main purposes of using State chart diagrams –

- To model the dynamic aspect of a system.
- To model the life time of a reactive system.
- To describe different states of an object during its life time.
- Define a state machine to model the states of an object.

Before drawing a state chart diagram, we should clarify the following points –

- Identify the important objects to be analyzed.
- Identify the states.
- Identify the events.

Fig 7: State chart diagram for Poultry Management System



#### 4.2.8 DEPLOYMENT DIAGRAM

A deployment diagram is a UML diagram type that shows the execution architecture of a system, including nodes such as hardware or software execution environments, and the middleware connecting them. Deployment diagrams are typically used to visualize the physical hardware and software of a system. Using it you can understand how the system will be physically deployed on the hardware. Deployment diagrams help model the hardware topology of a system compared to other UML diagram types which mostly outline the logical components of a system.

##### Deployment Diagram Notations -

In order to draw a deployment diagram, you need to first become familiar with the following deployment diagram notations and deployment diagram elements.

- I. **Nodes** - A node, represented as a cube, is a physical entity that executes one or more components, subsystems or executables. A node could be a hardware or software element.
  - i. **Artifacts** - Artifacts are concrete elements that are caused by a development process. Examples of artifacts are libraries, archives, configuration files, executable files etc.
  - ii. **Communication Association** - This is represented by a solid line between two nodes. It shows the path of communication between nodes.

- iv. **Devices** - A device is a node that is used to represent a physical computational resource in a system. An example of a device is an application server.
- v. **Deployment Specifications** - Deployment specifications are a configuration file, such as a text file or an XML document. It describes how an artifact is deployed on a node.

Follow are the simple steps below to draw a deployment diagram.

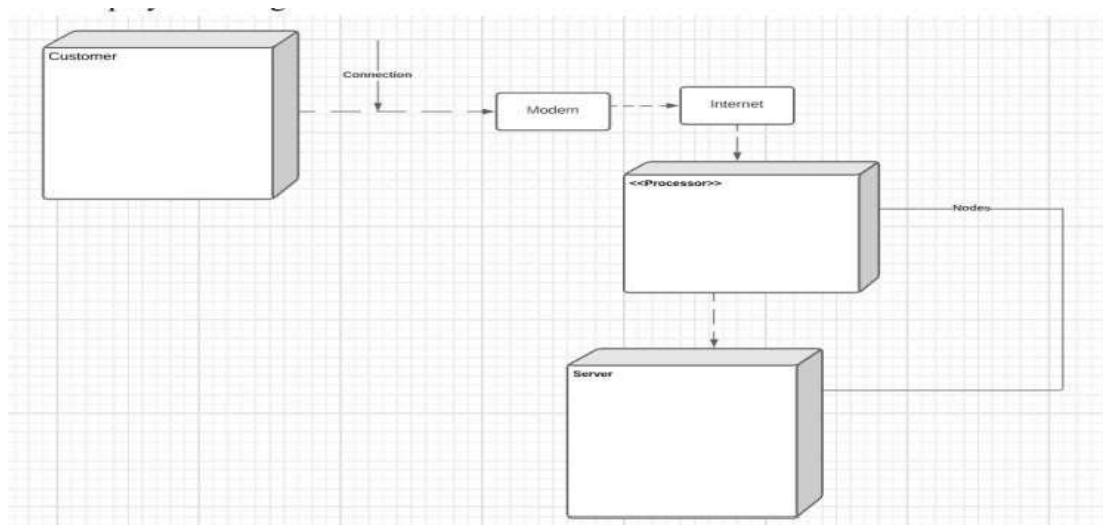
**Step 1:** Identify the purpose of your deployment diagram. And to do so, you need to identify the nodes and devices within the system you'll be visualizing with the diagram.

**Step 2:** Figure out the relationships between the nodes and devices. Once you know how they are connected, proceed to add the communication associations to the diagram.

**Step 3:** Identify what other elements like components, active objects you need to add to complete the diagram.

**Step 4:** Add dependencies between components and objects as required.

Fig 8: Deployment diagram for Poultry Management system



#### 4.2.9 COMPONENT DIAGRAM

Component diagram is a special kind of diagram in UML. The purpose is also different from all other diagrams discussed so far. It does not describe the functionality of the system but it describes the components used to make those functionalities. Thus, from that point of view, component diagrams are used to visualize the physical components in a system. These components are libraries, packages, files, etc.

Component diagrams can also be described as a static implementation view of a system. Static implementation represents the organization of the components at a particular moment. A single component diagram cannot represent the entire system but a collection of diagrams is used to represent the whole.

The purpose of the component diagram can be summarized as –

- Visualize the components of a system.
- Construct executables by using forward and reverse engineering.
- Describe the organization and relationships of the components.

Component diagrams are used to describe the physical artifacts of a system. This artifact includes files, executables, libraries, etc. The purpose of this diagram is different. Component diagrams are used during the implementation phase of an application. However, it is prepared well in advance to visualize the implementation details.

Initially, the system is designed using different UML diagrams and then when the artifacts are ready, component diagrams are used to get an idea of the implementation. This diagram is very important as without it the application cannot be implemented efficiently. A well- prepared component diagram is also important for other aspects such as application performance, maintenance, etc.

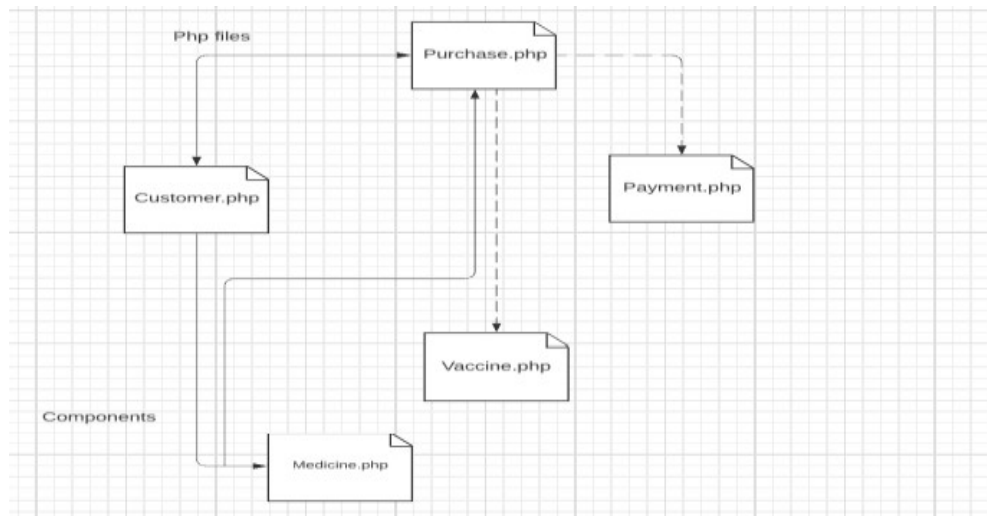
Before drawing a component diagram, the following artifacts are to be identified clearly –

- Files used in the system.
- Libraries and other artifacts relevant to the application.
- Relationships among the artifacts.

After identifying the artifacts, the following points need to be kept in mind.

- Use a meaningful name to identify the component for which the diagram is to be drawn.
- Prepare a mental layout before producing the using tools.
- Use notes for clarifying important points.

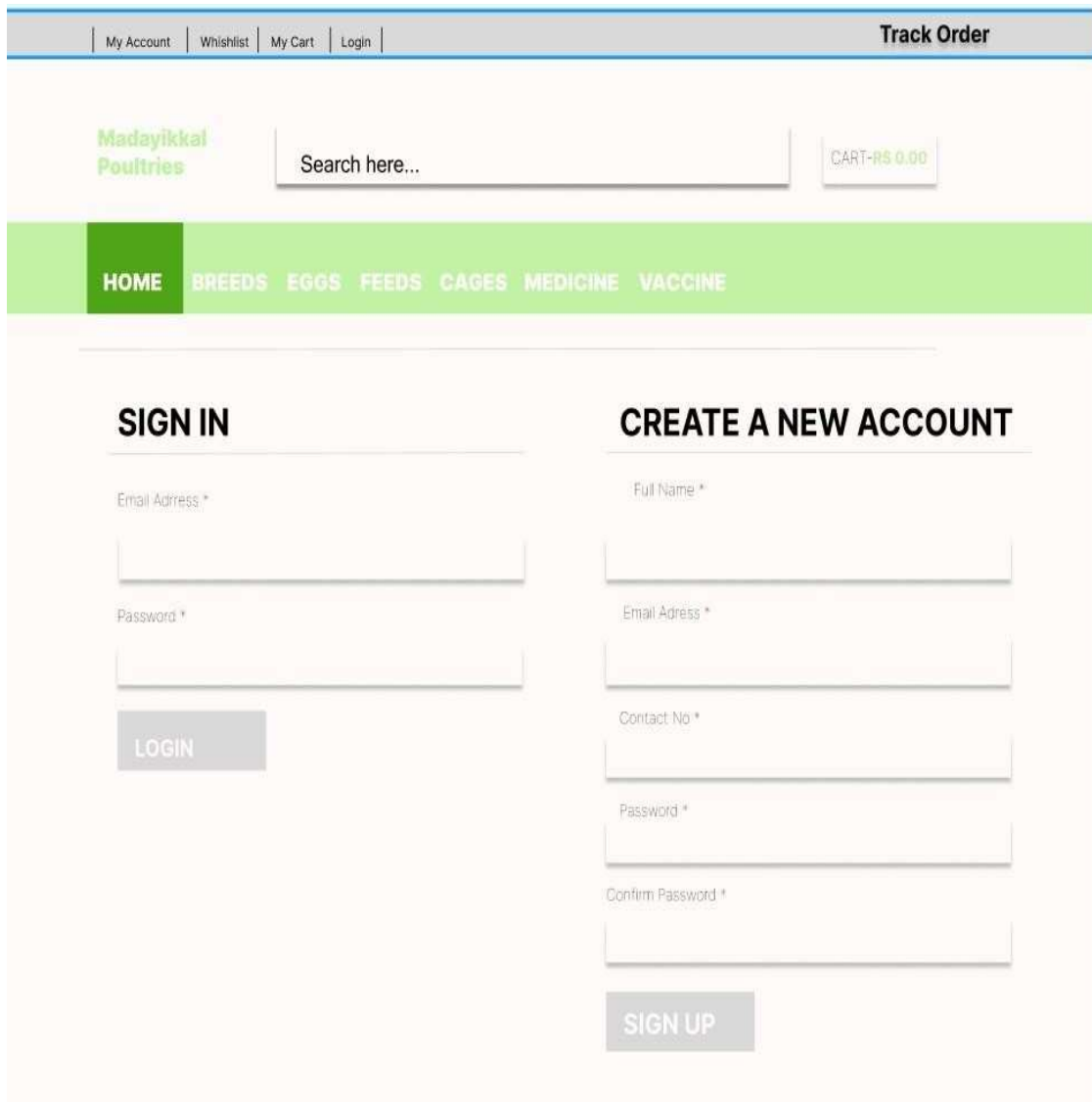
Fig 9: Component diagram for Poultry Management System



### 4.3 USER INTERFACE DESIGN USING FIGMA

#### 4.3.1-INPUT DESIGN

Form Name: Customer Registration & Login



The image displays a user interface design for a poultry management system, specifically for customer registration and login. The layout is as follows:

- Header:** A grey bar at the top contains links: "My Account", "Whishlist", "My Cart", and "Login". On the right side of this bar is a "Track Order" button.
- Logo and Search:** Below the header, on the left, is the logo "Madayikkal Poultries". To its right is a search bar with the placeholder text "Search here...". Further right is a cart icon showing "CART-R\$ 0.00".
- Navigation Menu:** A green horizontal bar contains the following menu items: "HOME", "BREEDS", "EGGS", "FEEDS", "CAGES", "MEDICINE", and "VACCINE".
- Sign In Section:** Titled "SIGN IN", it includes two input fields: "Email Address \*" and "Password \*". Below these fields is a grey "LOGIN" button.
- Create a New Account Section:** Titled "CREATE A NEW ACCOUNT", it includes four input fields: "Full Name \*", "Email Address \*", "Contact No \*", and "Password \*". Below these fields is a "Confirm Password \*" field. At the bottom of this section is a grey "SIGN UP" button.



Form Name: Customer Edit Profile

The screenshot displays the 'Customer Edit Profile' form within the Madayikkal Poultries web application. The interface includes a top navigation bar with links for 'My Account', 'Whishlist', 'My Cart', and 'Login', alongside a 'Track Order' button. Below this is a search bar and a cart status indicator showing 'CART-RS 0.00'. A green navigation menu contains links for 'HOME', 'BREEDS', 'EGGS', 'FEEDS', 'CAGES', 'MEDICINE', and 'VACCINE'. The main content area is titled '1 MY PROFILE' and contains a 'Personal info' section with input fields for 'Name', 'Email Address', and 'Phone No'. An 'UPDATE' button is positioned at the bottom of this section.

ame 1 | My Account | Whishlist | My Cart | Login | **Track Order**

Madayikkal Poultries

Search here...

CART-RS 0.00

**HOME** BREEDS EGGS FEEDS CAGES MEDICINE VACCINE

**1 MY PROFILE**

Personal info

Name

Email Address

Phone No

UPDATE

## 4.3.2 OUTPUT DESIGN

### User Login & Register

My Account Wishlist My Cart Login Track Order

Madayikkal Poultry

Search here...

CART - ₹5,00.00

HOME BREEDS EGGS FEEDS CAGES MEDICINE VACCINE

Home / Authentication

### SIGN IN

Hello, Welcome to your account.

You have successfully logout

Email Address \*

Password \*

### CREATE A NEW ACCOUNT

Create your own account.

Full Name \*

Email Address \*

Contact No. \*

### Customer Edit Profile

← → localhost/MAIN%20-%20Copy/myaccount.php

HOME BREEDS EGGS FEEDS CAGES MEDICINE VACCINE

Home / Checkout

### 1 MY PROFILE

Personal info

Name\*

Basil

Email Address\*

basilp@gmail.com

Phone No. \*

9747471432

UPDATE

### YOUR CHECKOUT PROGRESS

My Account

Shipping / Billing Address

Order History

Payment Pending Order

## 4.4 DATABASE DESIGN

A database is an organized mechanism that has the capability of storing information through which a user can retrieve stored information in an effective and efficient manner. The data is the purpose of any database and must be protected.

The database design is a two-level process. In the first step, user requirements are gathered together and a database is designed which will meet these requirements as clearly as possible. This step is called Information Level Design and it is taken independent of any individual DBMS.

In the second step, this Information level design is transferred into a design for the specific DBMS that will be used to implement the system in question. This step is called Physical Level Design, concerned with the characteristics of the specific DBMS that will be used. A database design runs parallel with the system design. The organization of the data in the database is aimed to achieve the following two major objectives.

### 4.4.1 Data Integrity

### 4.4.2 Data independence

### 4.6.1 Relational Database Management System (RDBMS)

A relational model represents the database as a collection of relations. Each relation resembles a table of values or file of records. In formal relational model terminology, a row is called a tuple, a column header is called an attribute and the table is called a relation. A relational database consists of a collection of tables, each of which is assigned unique name. A row in a table represents a set of related values.

### Relations, Domains & Attributes

A table is a relation. The rows in a table are called tuples. A tuple is an ordered set of  $n$  elements. Columns are referred to as attributes. Relationships have been set between every table in the database. This ensures both Referential and Entity Relationship Integrity. A domain  $D$  is a set of atomic values. A common method of specifying a domain is to specify a data type from which the data values forming the domain are drawn. It is also useful to specify a name for the domain to help in interpreting its values.

Every value in a relation is atomic, that is not decomposable.

**Relationships**

- Table relationships are established using Key. The two main keys of prime importance are Primary Key & Foreign Key. Entity Integrity and Referential Integrity Relationships can be established with these keys.
- Entity Integrity enforces that no Primary Key can have null values.
- Referential Integrity enforces that no Primary Key can have null values.
- Referential Integrity for each distinct Foreign Key value, there must exist a matching Primary Key value in the same domain. Other key are Super Key and Candidate Keys.

**4.6.2 Normalization**

Data are grouped together in the simplest way so that later changes can be made with minimum impact on data structures. Normalization is formal process of data structures in manners that eliminates redundancy and promotes integrity. Normalization is a technique of separating redundant fields and breaking up a large table into a smaller one. It is also used to avoid insertion, deletion, and updating anomalies. Normal form in data modeling use two concepts, keys and relationships. A key uniquely identifies a row in a table. There are two types of keys, primary key and foreign key. A primary key is an element or a combination of elements in a table whose purpose is to identify records from the same table. A foreign key is a column in a table that uniquely identifies record from a different table. All the tables have been normalized up to the third normal form.

As the name implies, it denotes putting things in the normal form. The application developer via normalization tries to achieve a sensible organization of data into proper tables and columns and where names can be easily correlated to the data by the user. Normalization eliminates repeating groups at data and thereby avoids data redundancy which proves to be a great burden on the computer resources. These include:

- ✓ Normalize the data.
- ✓ Choose proper names for the tables and columns.
- ✓ Choose the proper name for the data.

**First Normal Form**

The First Normal Form states that the domain of an attribute must include only atomic values and that the value of any attribute in a tuple must be a single value from the domain of that attribute. In other words, 1NF disallows “relations within relations” or “relations as attribute values within tuples”. The only attribute values permitted by 1NF are single atomic or indivisible values. The first step is to put the data into First Normal Form. This can be done by moving data into separate tables where the data is of similar type in each table. Each table is given a Primary Key or Foreign Key as per requirement of the project. In this we form new relations for each non-atomic attribute or nested relation. This eliminated repeating groups of data. A relation is said to be in first normal form if only if it satisfies the constraints that contain the primary key only.

**Second Normal Form**

According to Second Normal Form, for relations where primary key contains multiple attributes, no non-key attribute should be functionally dependent on a part of the primary key. In this we decompose and setup a new relation for each partial key with its dependent attributes. Make sure to keep a relation with the original primary key and any attributes that are fully functionally dependent on it. This step helps in taking out data that is only dependent on a part of the key. A relation is said to be in second normal form if and only if it satisfies all the first normal form conditions for the primary key and every non-primary key attributes of the relation is fully dependent on its primary key alone.

**Third Normal Form**

According to Third Normal Form, Relation should not have a non-key attribute functionally determined by another non-key attribute or by a set of non-key attributes. That is, there should be no transitive dependency on the primary key. In this we decompose and set up relation that includes the non-key attributes that functionally determines other non-key attributes. This step is taken to get rid of anything that does not depend entirely on the Primary Key. A relation is said to be in third normal form if only if it is in second normal form and more over the non key attributes of the relation should not be depend on other non-key attribute.

**TABLE DESIGN**

**Table No        01**

**Table Name     : admin**

**Primary Key    : id**

**Table Description : To store admin details**

<b>Name</b>	<b>Type</b>	<b>Key</b>	<b>Description</b>
<b>id</b>	<b>Int</b>	<b>Primary</b>	<b>Id</b>
<b>username</b>	<b>varchar</b>		<b>Admin's username</b>
<b>password</b>	<b>varchar</b>		<b>Password</b>

**Table No        02**

**Table Name     : tbl\_addbird**

**Primary Key    : id**

**Table Description: To store bird details**

<b>Name</b>	<b>Type</b>	<b>Key</b>	<b>Description</b>
<b>Bird_Id</b>	<b>Int</b>	<b>Primary</b>	<b>Bird's Id</b>
<b>BirdCount</b>	<b>Int</b>		<b>Bird's Count</b>
<b>Breed</b>	<b>Int</b>		<b>Bird Type</b>
<b>BirdDate</b>	<b>Date</b>		<b>Date in which breed added</b>

**Table No        03**

**Table Name     : tbl\_breed**

**Primary Key    : Breed\_Id**

**Table Description : To store breed details**

Name	Type	Key	Description
Breed_Id	int	Primary	Breed's Id
Breed_Type	int		Breed Type
Status	tinyint		Breed's Status

**Table No           04**

**Table Name       : tbl\_category**

**Primary Key       : Category\_Id**

**Table Description : To store category details**

Name	Type	Key	Description
Category_Id	int	Primary	Id
Category_name	varchar		Name
Status	boolean		0 or 1

**Table No           05**

**Table Name       : tbl\_chickrate**

**Primary Key       : Rate\_Id**

**Table Description: To store chick rate details**

Name	Type	Key	Description
Rate_Id	int	Primary	Id
Rate_Date	date		Date in which rate Added
Rate	int		Chick's rate

**Table No           06**

**Table Name       : tbl\_events**

**Primary Key       : Rate\_Id**

**Table Description: To store calendar event details**

Name	Type	Key	Description
Id	Int	Primary	Id
title	varchar		Title of the event
start	datetime		Event added
end	datetime		End date of the event

**Table No           07**

**Table Name       : tbl\_feed**

**Primary Key      : Feed\_Id**

**Table Description: To store feed details**

Name	Type	Key	Description
Feed_Id	Int	Primary	Id
Feed_Type	Varchar		Feed Type
Status	Tinyint		0 or 1

**Table No           08**

**Table Name       : tbl\_feedsupplier**

**Primary Key       : SFeed\_Id**

**Table Description : To store feedsupplier details**

Name	Type	Key	Description
SFeed_Id	Int	Primary	Id
SFeed_Type	Int		Feed Type Id
Feed_Quantity	Int		Count in Kg
Feed_EDate	Date		Feed Expiry Date
Feed_Price	Int		Price of Feed in Rs
Status	tinyint		0 or 1
SReg_Id	Int		Staff Id

**Table No           09**

**Table Name       : tbl\_hatchery**

**Primary Key       : Hatchery\_Id**

**Table Description : To store hatchery details**

Name	Type	Key	Description
Hatchery_Id	Int	Primary	Id
Hatchery_Count	Int		Count
Hatchery_Date	date		Date in which birds Added
Hatchery_Price	Int		Price in Rs

**Table No           10**

**Table Name       : tbl\_hatcherybill**

**Primary Key       : HBill\_Id**



**Table Description : To store hatchery bill details**

Name	Type	Key	Description
Hbill_Id	int	Primary	Id of Bill
HOrder_Id	int		Bird Order Id
HBill_Date	date		Bill Date
HPrice	int		Price of bird in Rs
Status	tinyint		0 or 1

**Table No            11****Table Name        : tbl\_orderbirdshatchery****Primary Key        : HOrder\_Id****Table Description : To store bird's order from hatchery details**

Name	Type	Key	Description
HOrder_Id	int	Primary	Hatchery Order Id
HHatchery_Id	int		Staff Reg_Id
Hatchery_Id	int		Id of Hatchery
HCount	int		Count of bird
HOrderDate	date		Date in which bird ordered
Status	tinyint		0 or 1

**Table No            12****Table Name        : tbl\_orderchickswholesaler****Primary Key        : WOrder\_Id****Table Description : To store bird's order from wholesaler side details**

Name	Type	Key	Description
WOrder_Id	int	Primary	Wholesaler Order Id
Farmer_Id	int		Staff Reg_Id(Farmer)
Bird_Id	int		Id of Bird
Type_Id	int		Type Id
WCount	int		Count of bird
WOrderDate	date		Date in which bird ordered
Status	tinyint		0 or 1

**Table No            13**

**Table Name        : tbl\_orderfeedsupplier**

**Primary Key        : SOrder\_Id**

**Table Description: To store feed's order from supplier details**

Name	Type	Key	Description
SOrder_Id	Int	Primary	Supplier Order Id
Supplier_Id	Int		Staff Reg_Id(Supplier)
SFeed_Id	Int		Id of feed
SType_Id	Int		Type Id
SQuantity	Int		Count of feed in Kg
SOrderDate	Date		Date in which feed ordered
Status	Tinyint		0 or 1

**Table No            14**

**Table Name        : tbl\_orders**

**Primary Key        : Order\_Id**

**Table Description: To store product order details**

Name	Type	Key	Description
Order_Id	Int	Primary	Order Id
User_Id	Int		Customer_Id
Product_Id	Int		Id of product
Quantity	Int		Count
OrderDate	Date		Date in which product ordered
PaymentMethod	Varchar		Mode of payment – cash,dr/cr etc
OrderStatus	Varchar		Status of Order
Status	Varchar		0 or 1
Reference_Number	Varchar		Order no

**Table No**           **15**

**Table Name**       **: tbl\_products**

**Primary Key**       **: id**

**Table Description** : To store product details

Name	Type	Key	Description
Id	Int	Primary	Product Id
Category_Id	Int		Category Id
Name	Varchar		Product Name
productDescription	Varchar		Product Decsription
Code	Varchar		Product code
Image	Varchar		Product Image
productPriceBeforeDiscount	Int		Price before Discount
productShippingcharge	Int		Shipping charge allotted for Product
productPrice	Int		Product Price
Quantity	Int		Count
productAvailability	Varchar		Product Stock
Status	Tinyint		0 or 1

**Table No**           **16**

**Table Name**       **: tbl\_staffregister**

**Primary Key**       **: StaffReg\_Id**

**Table Description:** To store staff details

Name	Type	Key	Description
StaffReg_Id	int	Primary	Staff Id
Type_Id	int		Type Id
Name	varchar		Staff name
Address	varchar		Staff Address
Email	varchar		Staff Email
Phone_No	bigint		Staff Phone Number
Password	varchar		Password
Status	tinyint		0 or 1

**Table No**            **17**

**Table Name**        **: tbl\_supplierbill**

**Primary Key**        **: SOrder\_Id**

**Table Description: To store supplier bill details**

<b>Name</b>	<b>Type</b>	<b>Key</b>	<b>Description</b>
SBill_Id	Int	Primary	Supplier Bill Id
SOrder_Id	Int		Order Id
SBill_Date	Date		Date in which bill added
SPrice	Int		Amount of feed
Status	Tinyint		0 or 1

**Table No**            **18**

**Table Name**        **: tbl\_type**

**Primary Key**        **: Type\_Id**

**Table Description: To store category details of staff**

<b>Name</b>	<b>Type</b>	<b>Key</b>	<b>Description</b>
Type_Id	Int	Primary	Type Id
Type	Varchar		Staff Category
Status	Tinyint		0 or 1

**Table No**            **19**

**Table Name**        **: tbl\_userlog**

**Primary Key**        **: Userlog\_Id**

**Table Description: To store customer entry details**

<b>Name</b>	<b>Type</b>	<b>Key</b>	<b>Description</b>
Userlog_Id	Int	Primary	Supplier Bill Id
UserEmail	varchar		Customer Log Id
UserIp	Binary		Customer EmailId
LoginTime	timestamp	Primary	Customer IP Address
Logout	varchar		Customer logout time
Status	Tinyint		0 or 1

**Table No**            **20**

**Table Name**        **: tbl\_users**

**Primary Key**        **: Userlog\_Id**

**Table Description : To store customer entry details**

<b>Name</b>	<b>Type</b>	<b>Key</b>	<b>Description</b>
User_Id	int	Primary	Customer Id
Name	varchar		Customer Name
Email	varchar		Customer Email Id
Phone_No	bigint		Customer Contact No
Password	varchar		Password
shippingAddress	varchar		Customer shippingAddress
shippingCity	varchar		Customer City
shippingState	varchar		State
shippingPincode	int		Pincode
billingAddress	varchar		Customer billingAddress
billingCity	varchar		City
billingState	varchar		State
billingPincode	int		Pincode
RegDate	timestamp		Customer Registration Date
UpdationDate	timestamp		Customer Updation time

**Table No**            **21**

**Table Name**        **: tbl\_wishlist**

**Primary Key**        **: Wishlist\_Id**

**Table Description : To store customer product wishlist details**

<b>Name</b>	<b>Type</b>	<b>Key</b>	<b>Description</b>
Wishlist_Id	int	Primary	Customer Wishlist Id
User_Id	int		Customer Id
Product_Id	int		Customer Product Id
Posting_Date	timestamp		Customer Posting Date

**Table No**            **22**

**Table Name**        **: tbl\_notify**

**Primary Key**        **: Id**

**Table Description : To store customer product out of stock notification details**

<b>Name</b>	<b>Type</b>	<b>Key</b>	<b>Description</b>
id	int	Primary	Notification Id
Product_Id	int		To store Product Id
User_Id	int		Customer Id
Update_status	varchar		To store updated notification status
status	int		To store status

## **CHAPTER 5**

### **SYSTEM TESTING**

## 5.1 INTRODUCTION

Software Testing is the process of executing software in a controlled manner, in order to answer the question - Does the software behave as specified? Software testing is often used in association with the terms verification and validation. Validation is the checking or testing of items, includes software, for conformance and consistency with an associated specification. Software testing is just one kind of verification, which also uses techniques such as reviews, analysis, inspections, and walkthroughs. Validation is the process of checking that what has been specified is what the user actually wanted.

Other activities which are often associated with software testing are static analysis and dynamic analysis. Static analysis investigates the source code of software, looking for problems and gathering metrics without actually executing the code. Dynamic analysis looks at the behavior of software while it is executing, to provide information such as execution traces, timing profiles, and test coverage information.

Testing is a set of activity that can be planned in advanced and conducted systematically. Testing begins at the module level and work towards the integration of entire computers based system. Nothing is complete without testing, as it vital success of the system testing objectives, there are several rules that can serve as testing objectives. They are:

Testing is a process of executing a program with the intent of finding an error.

- A good test case is one that has high possibility of finding an undiscovered error.
- A successful test is one that uncovers an undiscovered error.

If a testing is conducted successfully according to the objectives as stated above, it would uncover errors in the software. Also testing demonstrate that the software function appear to be working according to the specification, that performance requirement appear to have been met.

There are three ways to test program.

- For correctness
- For implementation efficiency
- For computational complexity



For computational complexity Test for correctness are supposed to verify that a program does exactly what it was designed to do. This is much more difficult than it may at first appear, especially for large programs.

## 5.2 TEST PLAN

A test plan implies a series of desired course of action to be followed in accomplishing various testing methods. The Test Plan acts as a blue print for the action that is to be followed. The software engineers create a computer program, its documentation and related data structures. The software developers are always responsible for testing the individual units of the programs, ensuring that each performs the function for which it was designed. There is an independent test group (ITG) which is to remove the inherent problems associated with letting the builder to test the thing that has been built. The specific objectives of testing should be stated in measurable terms. So that the mean time to failure, the cost to find and fix the defects, remaining defect density or frequency of occurrence and test work-hours per regression test all should be stated within the test plan.

The levels of testing include:

- ❖ Unit testing
- ❖ Integration Testing
- ❖ Data validation Testing
- ❖ Output Testing

### 5.2.1 Unit Testing

Unit testing focuses verification effort on the smallest unit of software design – the software component or module. Using the component level design description as a guide, important control paths are tested to uncover errors within the boundary of the module. The relative complexity of tests and uncovered scope established for unit testing. The unit testing is white-box oriented, and step can be conducted in parallel for multiple components. The modular interface is tested to ensure that information properly flows into and out of the program unit under test. The local data structure is examined to ensure that data stored temporarily maintains its integrity during all steps in an algorithm's execution.

Boundary conditions are tested to ensure that all statements in a module have been executed at least once. Finally, all error handling paths are tested.

Tests of data flow across a module interface are required before any other test is initiated. If data do not enter and exit properly, all other tests are moot. Selective testing of execution paths is an essential task during the unit test. Good design dictates that error conditions be anticipated and error handling paths set up to reroute or cleanly terminate processing when an error does occur. Boundary testing is the last task of unit testing step. Software often fails at its boundaries.

Unit testing was done in Sell-Soft System by treating each module as separate entity and testing each one of them with a wide spectrum of test inputs. Some flaws in the internal logic of the modules were found and were rectified. After coding each module is tested and run individually. All unnecessary code were removed and ensured that all modules are working, and gives the expected result.

#### **5.2.1.1 Selenium**

Selenium is one of the most widely used open-source Web UI (User Interface) automation testing suite. It was originally developed by Jason Huggins in 2004 as an internal tool at Thought Works. Selenium supports automation across different browsers, platforms and programming languages.

Selenium can be easily deployed on platforms such as Windows, Linux, Solaris and Macintosh. Moreover, it supports OS (Operating System) for mobile applications like iOS, windows mobile and android.

Selenium supports a variety of programming languages through the use of drivers specific to each language. Languages supported by Selenium include C#, Java, Perl, PHP, Python and Ruby. Currently, Selenium Web driver is most popular with Java and C#. Selenium test scripts can be coded in any of the supported programming languages and can be run directly in most modern web browsers. Browsers supported by Selenium include Internet Explorer, Mozilla Firefox, Google Chrome and Safari.

Selenium Features:

- Selenium is an open source and portable Web testing Framework.
- Selenium IDE provides a playback and record feature for authoring tests without the need to learn a test scripting language.
- It can be considered as the leading cloud-based testing platform which helps testers to record their actions and export them as a reusable script with a simple-to-understand and easy-to-use interface.
- Selenium supports various operating systems, browsers and programming languages. It also supports parallel test execution which reduces time and increases the efficiency of tests.
- Selenium can be integrated with frameworks like Ant and Maven for source code compilation.
- Selenium can also be integrated with testing frameworks like TestNG for application testing and generating reports.
- Selenium requires fewer resources as compared to other automation test tools.
- WebDriver API has been indulged in selenium which is one of the most important modifications done to selenium.
- Selenium web driver does not require server installation, test scripts interact directly with the browser.
- Selenium commands are categorized in terms of different classes which make it easier to understand and implement.
- Selenium Remote Control (RC) in conjunction with WebDriver API is known as Selenium 2.0. This version was built to support the vibrant web pages and Ajax.

## 5.2.1.2 Test Case

Test Case 1					
Project Name: Poultry Management System					
Login Test Case					
Test Case ID: Fun_1			Test Designed By: Jinta Susan Mathew		
Test Priority (Low/Medium/High): High			Test Designed Date: 22-05-2022		
Module Name: Login Screen			Test Executed By: Ms. Sruthimol Kurian		
Test Title : Verify login with valid email and password			Test Execution Date: 22-05-2022		
Description: Test the Login Page					
Pre-Condition :User has valid email id and password					
Step	Test Step	Test Data	Expected Result	Actual Result	Status(Pass/Fail)
1	Navigation to Login Page		Login Page should be	Login page displayed	Pass
			displayed		
2	Provide Valid Email Id	Email: basilp@gmail.com	User should be able to Login	User Logged in and navigated to my-cart with records	Pass
3	Provide Valid Password	Password: basil@123			
4	Click on Sign In button				

5	Provide Invalid Email Id or password	Email Id: user@gmail.Com Password: User1234	User should not be able to Login	Message for enter valid email id or password displayed	Pass
6	Provide Null Email Id or Password	Email Id: null Password: null			
7	Click on Sign In button				
<b>Post-Condition:</b> User is validated with database and successfully login into account. The Account session details are logged in database					

**Code**

```
package testcases;
```

```
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import Chromedriver.DriverSetup;
```

```
public class login{
    public static WebDriver driver;
```

```
    public static void main(String[] args) {
        //TODO Auto-generated method stub
```

```
        driver = DriverSetup.getWebDriver("http://localhost/MAIN%20-%20Copy/login.php");
```

```
        driver.findElement(By.name("email")).sendKeys("basilp@gmail.com");
        driver.findElement(By.name("password")).sendKeys("basil@123");
        driver.findElement(By.name("login")).click();
```

```
        String actualUrl = "http://localhost/MAIN%20-%20Copy/my-cart.php";
        String expectedUrl = driver.getCurrentUrl();
        if(actualUrl.equalsIgnoreCase(expectedUrl)) {
            System.out.println("Test passed");
        }
    }
}
```

```

else {
    System.out.println("Test failed");
}

```

```

driver.quit();

```

```

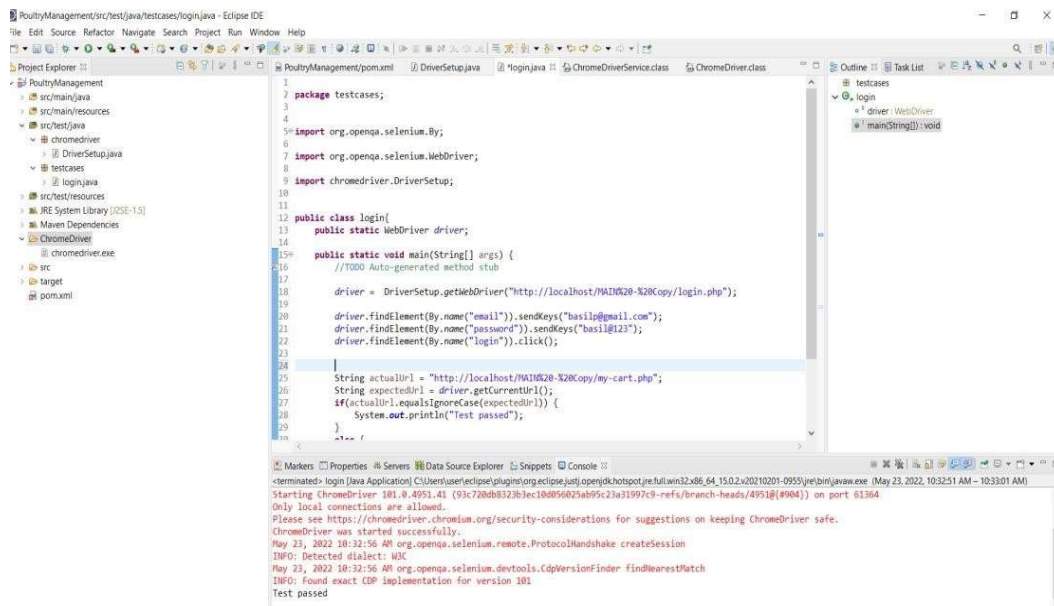
}

```

```

}

```



Test Case 2					
Project Name: Poultry Management System					
Change Password Test Case					
Test Case ID: Fun_2			Test Designed By: Jinta Susan Mathew		
Test Priority(Low/Medium/High): High			Test Designed Date: 22-05-2022		
Module Name: Add Category Test			Test Executed By : Ms.Sruthimol Kurian		
Test Title : Verify login with valid email and password and navigate to Category by Admin			Test Execution Date: 22-05-2022		
Description: Test the Add Category Page of Admin					
Pre-Condition :User has valid email id and password					
Step	Test Step	Test Data	Expected Result	Actual Result	Status(Pass/Fail)
1	Navigation to Login Page		Login Page should be displayed	Login page displayed	Pass
2	Provide Valid Username	Email: admin	Admin should be able to Login	Admin Logged in and navigated to Category with records	Pass
3	Provide Valid Password	Password: admin@123			
4	Click on Sign In button				
5	Provide Category name	Category name: Breed	Category name entered correctly	Category name added	Pass
6	Click on Submit button				
Post-Condition: Admin is validated with database and successfully login into account. The Account session details are logged in database and added Category name successfully					

**Code**

```
package testcases;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;

import chromedriver.DriverSetup;

public class adminlogintest {
    public static WebDriver driver;

    public static void main(String[] args) {
        //TODO Auto-generated method stub

        driver = DriverSetup.getWebDriver("http://localhost/MAIN%20-%20Copy/admin/index.php");

        driver.findElement(By.name("username")).sendKeys("admin");
        driver.findElement(By.name("password")).sendKeys("admin@123");
        driver.findElement(By.name("submit")).click();

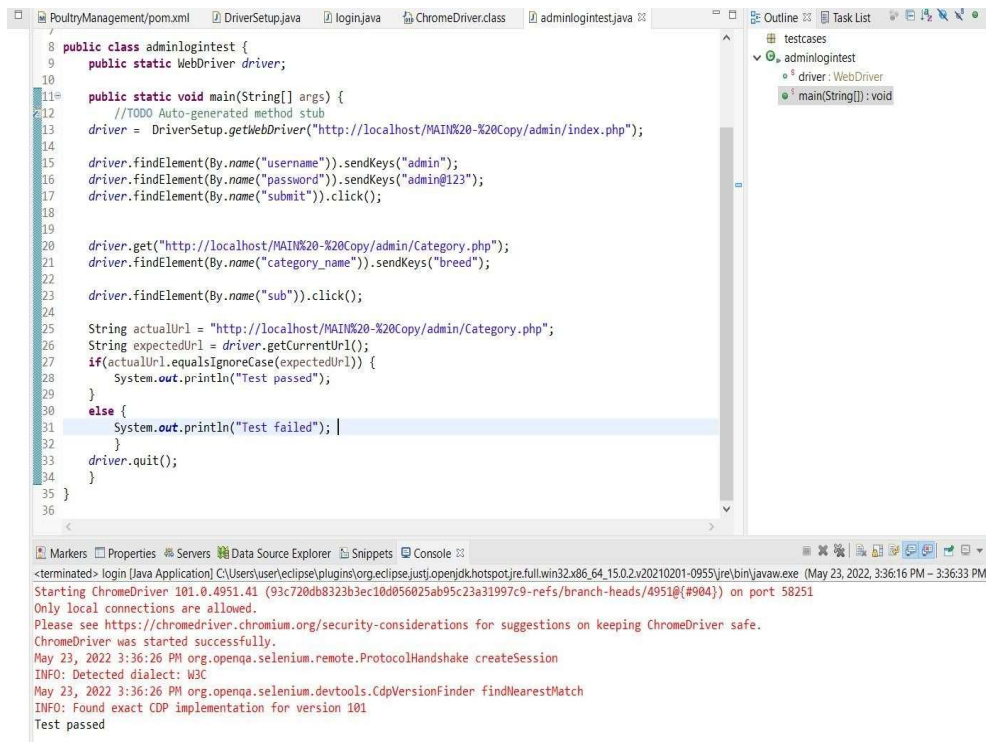
        driver.get("http://localhost/MAIN%20-%20Copy/admin/Category.php");
        driver.findElement(By.name("category_name")).sendKeys("breed");

        driver.findElement(By.name("sub")).click();

        String actualUrl = "http://localhost/MAIN%20-%20Copy/admin/Category.php"; String
        expectedUrl = driver.getCurrentUrl();
        if(actualUrl.equalsIgnoreCase(expectedUrl)) {
            System.out.println("Test passed");
        }
        else {
            System.out.println("Test failed");
        }

        driver.quit();
    }
}
```





```
8 public class adminlogintest {
9     public static WebDriver driver;
10
11     public static void main(String[] args) {
12         //TODO Auto-generated method stub
13         driver = DriverSetup.getWebDriver("http://localhost/MAIN%20-%20Copy/admin/index.php");
14
15         driver.findElement(By.name("username")).sendKeys("admin");
16         driver.findElement(By.name("password")).sendKeys("admin@123");
17         driver.findElement(By.name("submit")).click();
18
19
20         driver.get("http://localhost/MAIN%20-%20Copy/admin/Category.php");
21         driver.findElement(By.name("category_name")).sendKeys("breed");
22
23         driver.findElement(By.name("sub")).click();
24
25         String actualUrl = "http://localhost/MAIN%20-%20Copy/admin/Category.php";
26         String expectedUrl = driver.getCurrentUrl();
27         if(actualUrl.equalsIgnoreCase(expectedUrl)) {
28             System.out.println("Test passed");
29         }
30         else {
31             System.out.println("Test failed");
32         }
33         driver.quit();
34     }
35 }
36
```

Markers Properties Servers Data Source Explorer Snippets Console

<terminated> login [Java Application] C:\Users\user\workspace\poultrymanagement\src\test\java\org\openqa\selenium\remote\ProtocolHandshake createSession  
Starting ChromeDriver 101.0.4951.41 (93c720db8323b3ec10d056025ab95c23a31997c9-refs/branch-heads/4951@#904) on port 58251  
Only local connections are allowed.  
Please see https://chromedriver.chromium.org/security-considerations for suggestions on keeping ChromeDriver safe.  
ChromeDriver was started successfully.  
May 23, 2022 3:36:26 PM org.openqa.selenium.remote.ProtocolHandshake createSession  
INFO: Detected dialect: W3C  
May 23, 2022 3:36:26 PM org.openqa.selenium.devtools.CdpVersionFinder findNearestMatch  
INFO: Found exact CDP implementation for version 101  
Test passed

### 5.2.2 Integration Testing

Integration testing is systematic technique for constructing the program structure while at the same time conducting tests to uncover errors associated with interfacing. The objective is to take unit tested components and build a program structure that has been dictated by design. The entire program is tested as whole. Correction is difficult because isolation of causes is complicated by vast expanse of entire program. Once these errors are corrected, new ones appear and the process continues in a seemingly endless loop. After performing unit testing in the System all the modules were integrated to test for any inconsistencies in the interfaces. Moreover differences in program structures were removed and a unique program structure was evolved.

### 5.2.3 Validation Testing or System Testing

This is the final step in testing. In this the entire system was tested as a whole with all forms, code, modules and class modules. This form of testing is popularly known as Black Box testing or System tests.

Black Box testing method focuses on the functional requirements of the software. That is, Black Box testing enables the software engineer to derive sets of input conditions that will fully exercise all functional requirements for a program.

Black Box testing attempts to find errors in the following categories; incorrect or missing functions, interface errors, errors in data structures or external data access, performance errors and initialization errors and termination errors.

#### **5.2.4 Output Testing or User Acceptance Testing**

The system considered is tested for user acceptance; here it should satisfy the firm's need. The software should keep in touch with perspective system; user at the time of developing and making changes whenever required. This done with respect to the following points:

- Input Screen Designs,
- Output Screen Designs,

The above testing is done taking various kinds of test data. Preparation of test data plays a vital role in the system testing. After preparing the test data, the system under study is tested using that test data. While testing the system by which test data errors are again uncovered and corrected by using above testing steps and corrections are also noted for future use.

## **CHAPTER 6**

### **IMPLEMENTATION**

## 6.1 INTRODUCTION

Implementation is the stage of the project where the theoretical design is turned into a working system. It can be considered to be the most crucial stage in achieving a successful new system gaining the users confidence that the new system will work and will be effective and accurate. It is primarily concerned with user training and documentation. Conversion usually takes place about the same time the user is being trained or later. Implementation simply means convening a new system design into operation, which is the process of converting a new revised system design into an operational one. At this stage the main work load, the greatest upheaval and the major impact on the existing system shifts to the user department. If the implementation is not carefully planned or controlled, it can create chaos and confusion.

Implementation includes all those activities that take place to convert from the existing system to the new system. The new system may be a totally new, replacing an existing manual or automated system or it may be a modification to an existing system. Proper implementation is essential to provide a reliable system to meet organization requirements. The process of putting the developed system in actual use is called system implementation. This includes all those activities that take place to convert from the old system to the new system. The system can be implemented only after thorough testing is done and if it is found to be working according to the specifications. The system personnel check the feasibility of the system. The more complex the system being implemented, the more involved will be the system analysis and design effort required to implement the three main aspects: education and training, system testing and changeover.

The implementation state involves the following tasks:

- Careful planning.
- Investigation of system and constraints.
- Design of methods to achieve the changeover.

## 6.2 IMPLEMENTATION PROCEDURES

Implementation of software refers to the final installation of the package in its real environment, to the satisfaction of the intended uses and the operation of the system.

In many organizations someone who will not be operating it, will commission the software development project. In the initial stage people doubt about the software but we have to ensure that the resistance does not build up, as one has to make sure that:

- ☐ The active user must be aware of the benefits of using the new system.
- ☐ Their confidence in the software is built up.
- ☐ Proper guidance is imparted to the user so that he is comfortable in using the application.

Before going ahead and viewing the system, the user must know that for viewing the result, the server program should be running in the server. If the server object is not up running on the server, the actual process won't take place.

### **6.2.1 User Training**

User training is designed to prepare the user for testing and converting the system. To achieve the objective and benefits expected from computer based system, it is essential for the people who will be involved to be confident of their role in the new system. As system becomes more complex, the need for training is more important. By user training the user comes to know how to enter data, respond to error messages, interrogate the database and call up routine that will produce reports and perform other necessary functions.

### **6.2.2 Training on the Application Software**

After providing the necessary basic training on computer awareness the user will have to be trained on the new application software. This will give the underlying philosophy of the use of the new system such as the screen flow, screen design type of help on the screen, type of errors while entering the data, the corresponding validation check at each entry and the ways to correct the date entered. It should then cover information needed by the specific user/group to use the system or part of the system while imparting the training of the program on the application. This training may be different across different user groups and across different levels of hierarchy.

### 6.2.3 System Maintenance

Maintenance is the enigma of system development. The maintenance phase of the software cycle is the time in which a software product performs useful work. After a system is successfully implemented, it should be maintained in a proper manner. System maintenance is an important aspect in the software development life cycle. The need for system maintenance is for it to make adaptable to the changes in the system environment. Software maintenance is of course, far more than "Finding Mistakes".

## **CHAPTER 7**

### **CONCLUSION AND FUTURE SCOPE**

## 7.1 CONCLUSION

The current system working technology is old fashioned and there is no usage of commonly used technologies like internet, digital money. The proposed system introduces facility for customer to view products. Provides lots of advantages like search products; add items to shopping cart, enhanced user interface, payment options, add feedback, track orders and many more.

## 7.2 FUTURE SCOPE

- Admin can add a functionality regarding the purchase of any breed/feed by the customer by showing nearest distributor.
- After purchase, customer get updated (including delivery date) about transportation using GPS tracking.
- Admin can manage income and expenses.
- Directing Medical consulting with nearby Veterinary clinics



## **CHAPTER 8**

## **BIBLIOGRAPHY**

**REFERENCES:**

- Gary B. Shelly, Harry J. Rosenblatt, “*System Analysis and Design*”, 2009.
- Roger S Pressman, “*Software Engineering*”, 1994.
- PankajJalote, “*Software engineering: a precise approach*”, 2006.
- James lee and Brent ware Addison, “Open source web development with LAMP”, 2003
- IEEE Std 1016 Recommended Practice for Software Design Descriptions.

**WEBSITES:**

- [www.w3schools.com](http://www.w3schools.com)
- [www.jquery.com](http://www.jquery.com)
- <https://www.lovelycoding.org/construction-work-management-system/>
- <https://www.slideshare.net/chiragbarasiya/construction-management-system-final-year-report>
- <https://app.diagrams.net>
- <http://homepages.dcc.ufmg.br/~rodolfo/es-1-03/IEEE-Std-830-1998.pdf>
- [www.agilemodeling.com/artifacts/useCaseDiagram.html](http://www.agilemodeling.com/artifacts/useCaseDiagram.html)

## **CHAPTER 9**

## **APPENDIX**

## 9.1 Sample Code

### Shopping Cart

```
<?php
session_start();
error_reporting(0);
include('DbConnect.php');
if(isset($_POST['submit'])){
if(!empty($_SESSION['cart'])){
    foreach($_POST['quantity'] as $key => $val){
        if($val==0){
            unset($_SESSION['cart'][$key]);
        }
        else{
            $_SESSION['cart'][$key]['quantity']=$val;
        }
    }
}

    echo "<script>alert('Your Cart has been Updated');</script>";
}
}

// Code for Remove a Product from Cart
if(isset($_POST['remove_code']))
{

    if(!empty($_SESSION['cart'])){
        foreach($_POST['remove_code'] as $key){
            unset($_SESSION['cart'][$key]);
        }
        echo "<script>alert('Your Cart has been Updated');</script>";
    }
}

// code for insert product in order table

if(isset($_POST['ordersubmit']))
{

    if(strlen($_SESSION['Email'])==0)
    {
        header('location:login.php');
    }
}
```

```
else{

    $quantity=$_POST['quantity'];
    $pdd=$_SESSION['pid'];
    $value=array_combine($pdd,$quantity);

    foreach($value as $qty=> $val34){

        mysqli_query($con,"insert into tbl_orders(User_Id,Product_Id,Quantity)
        values('".$_SESSION['userid']."','$qty','$val34')"); header('location:payment-
        method.php');
    }

}

}

// code for billing address updation
if(isset($_POST['update']))
{

    $baddress=$_POST['billingaddress'];
    $bstate=$_POST['bilingstate'];
    $bcity=$_POST['billingcity'];
    $bpincode=$_POST['billingpincode'];

    $query=mysqli_query($con,"update tbl_users set billingAddress='$baddress',billingCity
    ='$bcity',billingState='$bstate',billi ngPincode='$bpincode' where User_Id
    ='".$_SESSION['userid']."'");

    if($query)
    {
        echo "<script>alert('Billing Address has been updated');</script>";
    }
}

// code for Shipping address updation
if(isset($_POST['shipupdate']))
{

    $saddress=$_POST['shippingaddress'];
    $sstate=$_POST['shippingstate'];
    $scity=$_POST['shippingcity'];
    $spincode=$_POST['shippingpincode'];

    $query=mysqli_query($con,"update tbl_users set shippingAddress = '$saddress',
    shippingCity='$scity',shippingState='$sstate', shippingPincode='$spincode' where
```

```
User_Id="$_SESSION['userid'].");

if($query)
{
    echo "<script>alert('Shipping Address has been updated');</script>";
}
}

?>

<!DOCTYPE html>
<html lang="en">
<head>
<!-- Meta -->
<meta charset="utf-8">
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0, user-scalable=no">
<meta name="description" content="">
<meta name="author" content="">
<meta name="keywords" content="MediaCenter, Template, eCommerce">
<meta name="robots" content="all">

<title>My Cart</title>
<link rel="stylesheet" href="project/assets/css/bootstrap.min.css">
<link rel="stylesheet" href="project/assets/css/main.css">
<link rel="stylesheet" href="project/assets/css/green.css">
<link rel="stylesheet" href="project/assets/css/owl.carousel.css">
<link rel="stylesheet" href="project/assets/css/owl.transitions.css">
<link href="project/assets/css/lightbox.css" rel="stylesheet">
<link rel="stylesheet" href="project/assets/css/animate.min.css">
<link rel="stylesheet" href="project/assets/css/rateit.css">
<link rel="stylesheet" href="project/assets/css/bootstrap-select.min.css">

<!-- Demo Purpose Only. Should be removed in production -->

<link rel="stylesheet" href="project/assets/css/config.css">

<link href="project/assets/css/green.css" rel="alternate stylesheet" title="Green color">
<link href="project/assets/css/blue.css" rel="alternate stylesheet" title="Blue color">
<link href="project/assets/css/red.css" rel="alternate stylesheet" title="Red color">

<link href="project/assets/css/orange.css" rel="alternate stylesheet" title="Orange color">

<link href="project/assets/css/dark-green.css" rel="alternate stylesheet" title="Darkgreen color">
```

```
<!-- Demo Purpose Only. Should be removed in production : END -->

<!-- Icons/Glyphs -->
<link rel="stylesheet" href="project/assets/css/font-awesome.min.css">

<!-- Fonts -->

<link href='http://fonts.googleapis.com/css?family=Roboto:300,400,500,700' rel='stylesheet'
  type='text/css'>

<!-- Favicon -->
<link rel="shortcut icon" href="project/assets/images/favicon.ico">

<!-- HTML5 elements and media queries Support for IE8 : HTML5 shim and Respond.js -->

</head>
<body class="cnt-home">

<!-- ===== HEADER
===== -->
<header class="header-style-1">
<?php include('project/includes/top-header.php');?>
<?php include('project/includes/main-header.php');?>
<?php include('project/includes/menu-bar.php');?>
</header>
<!-- ===== HEADER : END
===== -->
<div class="breadcrumb">
  <div class="container">
    <div class="breadcrumb-inner">
      <ul class="list-inline list-unstyled">
        <li><a href="Homeindex.php">Home</a></li>

        <li class='active'>Shopping Cart</li>

      </ul>
    </div><!-- /.breadcrumb-inner -->

  </div><!-- /.container -->
</div><!-- /.breadcrumb -->
```

```

<div class="body-content outer-top-xs">

    <div class="container">
        <div class="row inner-bottom-sm">
            <div class="shopping-cart">
                <div class="col-md-12 col-sm-12 shopping-cart-table ">
                    <div class="table-responsive">
                        <form name="cart" method="post">
                            <?php
                                if(!empty($_SESSION['cart'])) {
                                    ?>
                                    <table class="table table-bordered">
                                        <thead>
                                            <tr>
                                                <th class="cart-remove item">Remove</th>
                                                <th class="cart-description item">Image</th>
                                                <th class="cart-product-name item">Product Name</th>

                                                <th class="cart-qty item">Quantity</th>
                                                <th class="cart-sub-total item">Price Per unit</th>
                                                <th class="cart-sub-total item">Shipping Charge</th>
                                                <th class="cart-total last-item">Grandtotal</th>
                                            </tr>
                                        </thead>
                                        <tfoot>
                                            <tr>
                                                <td colspan="7">
                                                    <div class="shopping-cart-btn">
                                                        <span class="">
                                                            <a href="Homeindex.php" class="btn btn-upper btn- primary outer-
                                                                left-xs">Continue Shopping</a>
                                                            <input type="submit" name="submit" value="Update shopping cart"
                                                                class="btn btn-upper btn-primary pull-right outer-right-xs">
                                                        </span>
                                                    </div><!-- /.shopping-cart-btn -->
                                                </td>
                                            </tr>
                                        </tfoot>
                                    <tbody>

                            <?php

                                $pdtid=array();

```



```

sql = "SELECT * FROM tbl_products WHERE id IN(";

foreach($_SESSION['cart'] as $id => $value){
    $sql .= $id. ",";
}
$sql=substr($sql,0,-1) . ") ORDER BY id ASC";

$query = mysqli_query($con,$sql);
$totalprice=0;
$totalqunty=0;

if(!empty($query)){
    while($row = mysqli_fetch_array($query)){
        $quantity=$_SESSION['cart'][$row['id']]['quantity'];
        $subtotal= $_SESSION['cart'][$row['id']]['quantity']*$row['productPrice']+
            $row['productShippingcharge'];

        $totalprice += $subtotal;
        $_SESSION['qnty']=$totalqunty+=$quantity;

        array_push($pdtid,$row['id']);
    }
    //print_r($_SESSION['pid'])=$pdtid;exit;
    ?>

<tr>
    <td class="romove-item"><input type="checkbox" name="remove_code[]"
    value="<?php echo htmlentities($row['id']);?>" /></td>
    <td class="cart-image">
        <a class="entry-thumbnail" href="product-details.php">
            
        </a>
    </td>
    <td class="cart-product-name-info">
        <h4 class="cart-product-description"><a href="product-
        details.php?pid=<?php echo htmlentities($pd=$row['id']);?>" >
        <?php echo $row['name'];

        $_SESSION['sid']=$pd;
        ?></a></h4>

    <div class="row">
        <div class="col-sm-4">

```

```

        <div class="rating rateit-small"></div>
    </div>
    <div class="col-sm-8">
        <?php $rt=mysqli_query($con,"select * from tbl_productreviews where
        Product_Id='$pd'");
        $num=mysqli_num_rows($rt);
        {
        ?>

    <div class="reviews">
        ( <?php echo htmlentities($num);?> Reviews )
    </div>
    <?php } ?>

</div>
</div><!-- /.row -->

</td>

    <td class="cart-product-quantity">
        <div class="quant-input">
            <div class="arrows">
                <div class="arrow plus gradient"><span class="ir">
                    <i class="icon fa fa-sort-asc"></i></span></div>
                <div class="arrow minus gradient"><span class="ir">
                    <i class="icon fa fa-sort-desc"></i></span></div>
            </div>
            <input type="text" value="<?php echo $_SESSION['cart'][$row['id']]['quantity']; ?>"
            name="quantity[<?php echo $row['id'];
            ?>]">

        </div>
    </td>
    <td class="cart-product-sub-total"><span class="cart-sub-total- price">
        <?php echo "Rs"." ".$row['productPrice']; ?>.00</span></td>
    <td class="cart-product-sub-total"><span class="cart-sub-total-price">
        <?php echo "Rs"." ".$row['productShippingcharge']; ?>.00</span></td>

    <td class="cart-product-grand-total"><span class="cart-grand-total- price">
        <?php echo ($_SESSION['cart'][$row['id']]['quantity']*$ row['productPrice'] +
        $row['productShippingc harge']); ?>.00</span></td>
</tr>

<?php } }

```

```

$_SESSION['pid']=$pdtid;
?>

</tbody><!-- /tbody -->
</table><!-- /table -->

</div>
</div><!-- /.shopping-cart-table -->
<div class="col-md-4 col-sm-12 estimate-ship-tax">

<table class="table table-bordered">
<thead>
<tr>
<th>
<span class="estimate-title">Shipping Address</span>
</th>
</tr>
</thead>

<tbody>
<tr>
<td>
<div class="form-group">

<?php
$query=mysqli_query($con,"select * from tbl_users where User_Id='".$_$_SESSION['userid']."'");
while($row=mysqli_fetch_array($query))
{
?>

<div class="form-group">
<label class="info-title" for="Billing Address">Billing Address<span>*</span></label>
<textarea class="form-control unicast-form-control text-input" name="billingaddress"
required="required"><?php echo
$row['billingAddress'];?></textarea>
</div>

<div class="form-group">
<label class="info-title" for="Billing State ">Billing State <span>*</span></label>

<input type="text" class="form-control unicast-form-control text-input" id="bilingstate"
name="bilingstate" value="<?php echo $row['billingState'];?>" required>
</div>

```

```

<div class="form-group">
<label class="info-title" for="Billing City">Billing City <span>*</span></label>

<input type="text" class="form-control unicast-form-control text-input"
id="billingcity" name="billingcity" required="required" value="<?php echo

$row['billingCity'];?>" >
</div>
<div class="form-group">
<label class="info-title" for="Billing Pincode">Billing Pincode
<span>*</span></label>
<input type="text" class="form-control unicast-form-control text-input"
id="billingpincode" name="billingpincode"

required="required" value="<?php echo $row['billingPincode'];?>" >
</div>
<?php

<button type="submit" name="update" class="btn-upper btn btn-primary
checkout-page-button">Update</button>

<?php } ?>

</div>

</td>
</tr>
</tbody><!-- /tbody -->
</table><!-- /table -->
</div>

<div class="col-md-4 col-sm-12 estimate-ship-tax">
<table class="table table-bordered">
<thead>
<tr>
<th>
<span class="estimate-title">Billing Address</span>
</th>
</tr>
</thead>
<tbody>
<tr>
<td>

```

```

        <div class="form-group">
        <?php
$query=mysqli_query($con,"select * from tbl_users where User_Id='".$$_SESSION['userid']."'");
while($row=mysqli_fetch_array($query))
{
?>

        <div class="form-group">
        <label class="info-title" for="Shipping Address">Shipping Address<span>*</span></label>

        <textarea class="form-control unicast-form-control text-input" name="shippingaddress"
        required="required"><?php echo
        $row['shippingAddress'];?></textarea>
        </div>

        <div class="form-group">
        <label class="info-title" for="Billing State ">Shipping State <span>*</span></label>

        <input type="text" class="form-control unicast-form-control text-input"
        id="shippingstate" name="shippingstate" value="<?php echo $row['shippingState'];?>"
        required>
        </div>

        <div class="form-group">
        <label class="info-title" for="Billing City">Shipping City <span>*</span></label>

        <input type="text" class="form-control unicast-form-control text-input"
        id="shippingcity" name="shippingcity" required="required" value="<?php echo
        $row['shippingCity'];?>" >
        </div>
        <div class="form-group">
        <label class="info-title" for="Billing Pincode">Shipping Pincode <span>*</span></label>

        <input type="text" class="form-control unicast-form-control text-input"
        id="shippingpincode" name="shippingpincode" required="required" value="<?php
        echo $row['shippingPincode'];?>" >
        </div>

        <button type="submit" name="shipupdate" class="btn-upper btn btn- primary checkout-page-
        button">Update</button>

        <?php } ?>

    </div>

```

```

</td>
</tr>
</tbody><!-- /tbody -->
</table><!-- /table -->
</div>
<div class="col-md-4 col-sm-12 cart-shopping-total">
<table class="table table-bordered">
<thead>
<tr>

<th>

<div class="cart-grand-total">
Grand Total<span class="inner-left-md"><?php echo
$_SESSION['tp']="$totalprice". ".00"; ?></span>
</div>
</th>
</tr>
</thead><!-- /thead -->
<tbody>
<tr>
<td>

<div class="cart-checkout-btn pull-right">
<button type="submit" name="ordersubmit" class="btn btn- primary">PROCEED TO
CHECKOUT</button>

</div>
</td>
</tr>
</tbody><!-- /tbody -->
</table>
<?php } else {
    echo "Your shopping Cart is empty";
    }?>
</div>
</div>
</div>
</form>
<?php echo include('project/includes/brands-slider.php');?>
</div>
</div>
<?php include('project/includes/footer.php');?>

```

```
<script src="project/assets/js/jquery-1.11.1.min.js"></script>
<script src="project/assets/js/bootstrap.min.js"></script>
<script src="project/assets/js/bootstrap-hoverdropdown.min.js"></script>
<script src="project/assets/js/owl.carousel.min.js"></script>

<script src="project/assets/js/echo.min.js"></script>
<script src="project/assets/js/jquery.easing-1.3.min.js"></script>
<script src="project/assets/js/bootstrap-slider.min.js"></script>
<script src="project/assets/js/jquery.rateit.min.js"></script>
<script type="text/javascript" src="project/assets/js/lightbox.min.js"></script>

<script src="project/assets/js/bootstrap-select.min.js"></script>
<script src="project/assets/js/wow.min.js"></script>
<script src="project/assets/js/scripts.js"></script>

<!-- For demo purposes – can be removed on production -->

<script src="switchstylesheet/switchstylesheet.js"></script>

<script>
    $(document).ready(function() {
        $(".changecolor").switchstylesheet( { seperator:"color" } );
        $('.show-theme-options').click(function() {
            $(this).parent().toggleClass('open'); return
            false;
        });

    });

    $(window).bind("load", function() {
        $('.show-theme-options').delay(2000).trigger('click');
    });
</script>

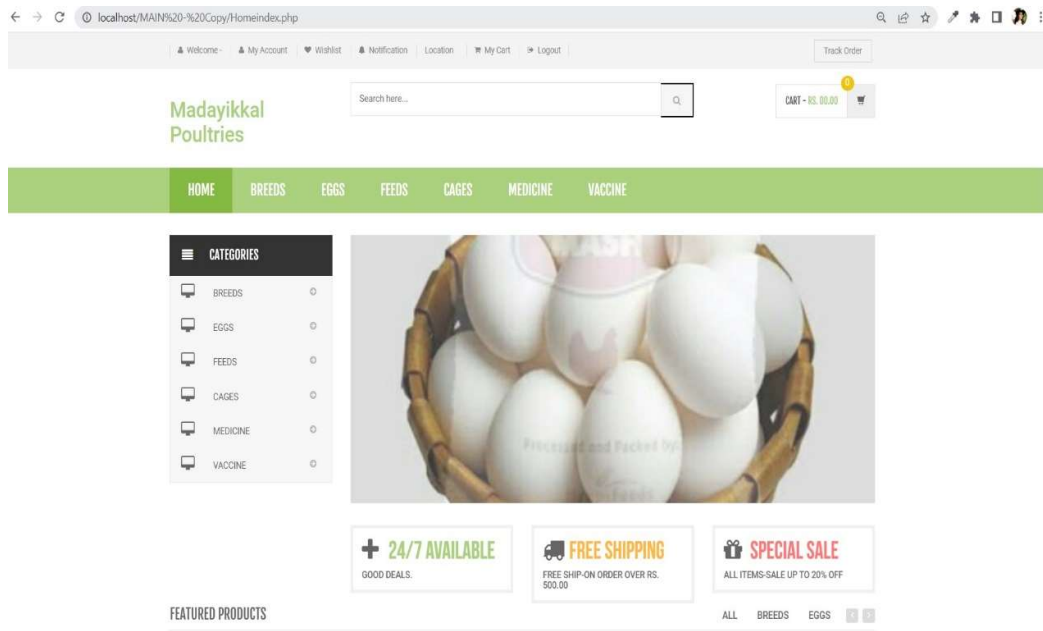
<!-- For demo purposes – can be removed on production : End -->
</body>
</html>
```

---

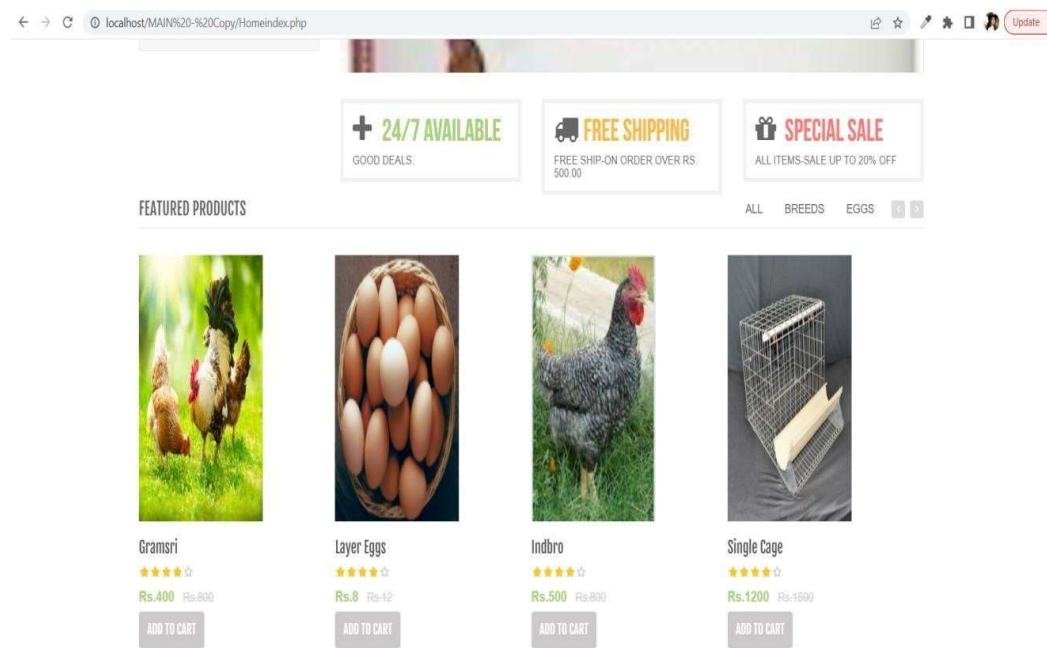
## 9.2 Screen Shots

### CUSTOMER PAGES

#### Customer Home page

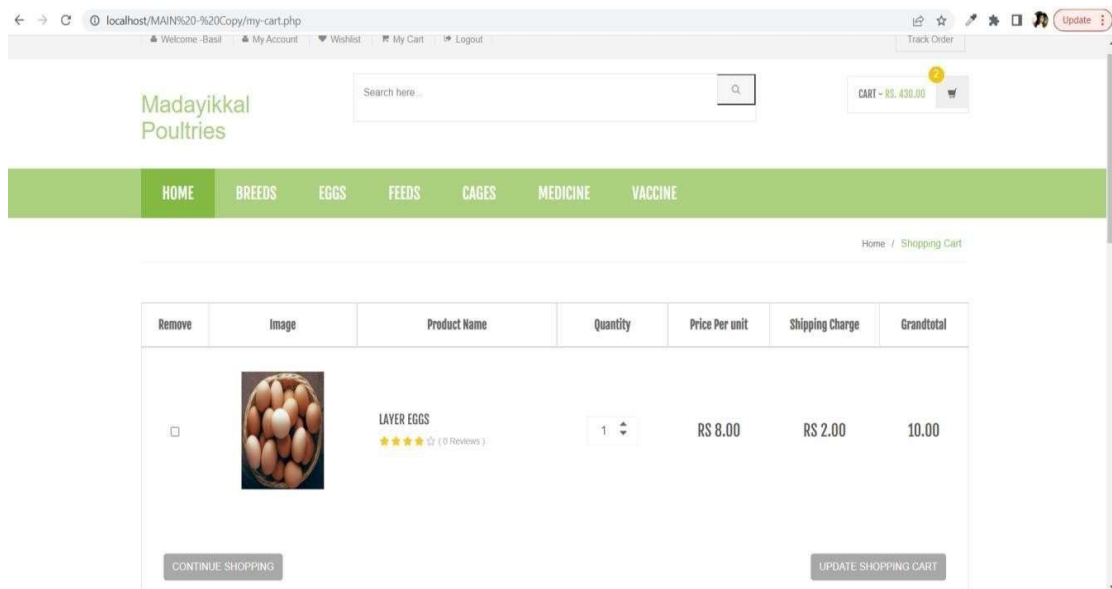


#### Customer View Product page

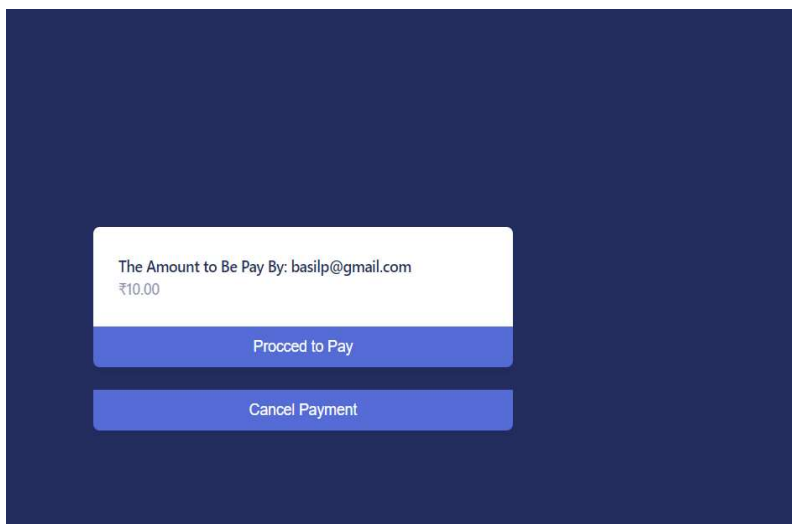




## Shopping Cart page



## Purchase page



## ADMIN PAGES

### View all registered customers

localhost/MAIN%20-%20Copy/admin/manage-users.php

Madayikkal Poultries | Admin

Admin

- Dashboard
- Calendar
- Add Chick Rate
- Manage users
- Staff Management
- Farm Settings
- Create Category
- Insert Product
- Manage Products
- Stock Updation

#### Manage Users

Name	Email	Contact no	Shipping Address/City/State/Pincode	Billing Address/City/State/Pincode	Reg. Date
Jinta Susan Mathew	jintasusanmathew10@gmail.com	9845632102	Pullockkal,Pathanmthi,Kerala-689643	Madayikkal House, KalathuVayal PO, Ambalavayal,Way,Ambalavaya,Kerala-673593	2022-05-14 10:16:16.076660
Basil	basilp@gmail.com	9747471432	bcdn sv, cdsn,cvd-984563	nvmf,bvfn,bvfh-895412	2022-05-11 15:21:07.294822

### Admin Add Categories

localhost/MINI/Category.php

#### CATEGORIES

Category Name

Submit

#	Category Name	Edit	Status	Toggle
1	Breeds	<a href="#">Edit</a>	Inactive	<a href="#">Activate</a>
2	Eggs	<a href="#">Edit</a>	Inactive	<a href="#">Activate</a>
3	Feeds	<a href="#">Edit</a>	Inactive	<a href="#">Activate</a>
4	Cages	<a href="#">Edit</a>	Inactive	<a href="#">Activate</a>
5	Medicine	<a href="#">Edit</a>	Inactive	<a href="#">Activate</a>
6	Vaccine	<a href="#">Edit</a>	Inactive	<a href="#">Activate</a>

## Admin Add Products

← → ↻ localhost/MINI/addproduct.php Update

**DASHBOARD**

**PRODUCTS**

ORDERS

CUSTOMERS

STOCK UPDATION

STATISTICS

REVIEWS

TRANSACTIONS

APPEARANCE

SETTINGS

LOGOUT

### ADD PRODUCT

Product Category: --Select--

Product Name:

Product Description:

Product Code:

Image:  No file chosen

180 x 180

Powered by HTML.COM

Product Price:

Quantity:

Show 10 entries

Search:

Product	Price	Category	Edit	Status	Toggle
Cage	1500.00	Cages	<a href="#">Edit</a>	Active	<a href="#">Deactivate</a>
Crumble	1500.00	Feeds	<a href="#">Edit</a>	Active	<a href="#">Deactivate</a>
Eccoli Medicine	800.00	Medicine	<a href="#">Edit</a>	Active	<a href="#">Deactivate</a>
Giriraja	500.00	Breeds	<a href="#">Edit</a>	Inactive	<a href="#">Activate</a>
Gramasri	500.00	Breeds	<a href="#">Edit</a>	Active	<a href="#">Deactivate</a>
Layereggs	100.00	Eggs	<a href="#">Edit</a>	Active	<a href="#">Deactivate</a>
Vaccine Pro Vita	800.00	Vaccine	<a href="#">Edit</a>	Active	<a href="#">Deactivate</a>

Showing 1 to 7 of 7 entries

Previous 1 Next

## DISTRIBUTOR PAGES

### View Order List

← → ↻ localhost/MAIN%20-%20Copy/Distributor/order\_list.php Welcome basilp@gmail.com

Madayikkal Poultries | Distributor

**Dashboard**

**Order Management**

Reports

Calendar

Reference Number	Customer Name	Product Name	Quantity	Amount	Payment Method	Ordered Date	Status	Action
881684764118	Jinta Susan Mathew	Single Cage	1	1200		2022-05-23 07:00:00.315618	Shipped	<a href="#">View</a> <a href="#">Delete</a>
271619186244	Basil	Indbro	1	520		2022-05-23 06:59:50.142820	Delivered	<a href="#">View</a> <a href="#">Delete</a>
572740657351	Basil	Indbro	1	520		2022-05-23 06:58:54.769044	In-Transit	<a href="#">View</a> <a href="#">Delete</a>
863011247036	Basil	Layer Eggs	1	10		2022-05-23 07:04:02.414028	Delivered	<a href="#">View</a> <a href="#">Delete</a>