­­

# **SYNOPSIS**

## Title of the project:

## 

## ARIMALA GENERAL STORE

## Introduction:

## In Hosangadi, there is a wholesale and retail establishment called Arimala General Store. There isn't yet an online delivery service in Hosangadi. Arimala General Store is a web application where it provides a platform for clients to acquire their home grocery items at their doorstep. The user will be able to use the app to find and buy the products they require. It is designed to be simple for customers to purchase groceries at ARIMALA GENERAL STORE. Customers can place online grocery orders from their homes, and the items can then be picked up from the store.

## Objective of the project:

* The main objective of the “Arimala General Store” application is to deliver grocery items which have been ordered through online.
* Here Customers can get free delivery with some conditions.
* Shop owner update all prices and stocks of grocery items daily.
* This application typically user-friendly interface that allows customer to browse and serach for grocery items and add to their cart.
* Customer can also post their review based on services.
* The application provide notification about the new offers.
* Pre-ordering is provided to the customer.

## Project Category:

Web-Based Application

## Languages to be used:

* + - * Front-end: HTML, CSS, JavaScript
      * Back-end: PYTHON
      * Database: Mysql

## Hardware Interface:

* + - Processors: Intel Pentium dual-core or above
    - RAM: 2GB and above
    - Hard disk Utilization: 40GB and above
    - Input Devices: Mouse, Keyboard

### Software Interface:

* + - Web Technology: Python
    - Web Components: HTML5/CSS/JavaScript/Django Framework
    - Software’s: XAMPP,VS Code editor
    - Database(Back-end): Mysql
    - WebServer: Apache
  1. **Stake Holders:**

Admin

Customers(retailers)

## Assumption:

* Direct delivery
* User friendly
* Time save
* Speed of working process

## 1.9 Modular description

Admin:

The Admin can login to the system using his/her unique credential. Here admin can manage the details of customer and Store.

* Login : In this module Admin can login by using their login credentials.
* Profile: This module shows the Admin ID,Email&Password.
* View customer: In this module admin can manage the customer details who are registered to the system.
* Manage Payments: In this module, the admin can manage all the payments made by the customers.
* Manage Locations: In this module, the admin can manage location details.
* Manage Category: In this module, the admin can manage the different category details.
* View Feedback: In this module, the admin can view feedback given by the customers.
* Manage Reports: In this module, the admin will manage all the report details of the system.
* Manage Product Details: In this module, the shop owner will add, delete and update the product details.
* Manage Order: In this module, the shop owner can view the orders made by the customers.
* Employee pay: In this module can check payment related to the employee.
* Employee attendance: This module helps the admin to check the employee attendance.

Customer:

* Register: The customer has to register by providing primary details before ordering.
* Login: In this module Customer can login by using their login credentials.
* Manage profile details: In this module customers can update his /her profile details.
* View product details: In this module, customers can view the respective shop details, the product details including the product description and specification.
* Add to cart: Customer can add the selected items to cart.
* Place an order: Customer can order items added to the cart.
* Track the delivery: Once the booking is made the customer can track the product delivery.
* Payment: Customer can pay for the orders through an online or offline method.
* Write Feedback: In this module, the Customer can write feedback.

## 1.11 User class and characteristics

**1.11.1 Administrators**

### The administrator can verify the payment activity of the Arimala General Store as well as alter the prices and stock levels. Additionally, they keep track of staff attendance and pay. All operations in a web application are managed by administrators.

**1.11.3 Customers**

## Customers can browse several product categories, choose a product, and add it to their shopping basket. Customers who needed to purchase certain things from Arimala General Store might also pre-order them. During a given timeframe, the ordered item will be delivered

## 1.12. Limitations

* The system is a web based application and therefore requires sturdy servers that can handle a lot of data.
* people who don’t know how to use application cannot make order.
* The information on website might be unreliable if not updated on a regular basis.

**1.13 Advantage/Benefit.**

* Even for the people who have basic knowledge about the application can use this website.
* Product will be delivered at your doorstep in short time. • Reduce your travelling time and cost.

## 1.14 Future Scope

* count to mobile application
* More categories of products can be added overtime
* Additional Enhanced features can be added overtime.

## 1.15 Team members

* Abdul Hakeem
* Mohammed Ubais M B

**2. SOFTWARE REQUIREMENT SPECIFICATION**

**2.1 Introduction**

All of the features and requirements of the "ARIMALA GENERAL STORE" system are fully described in this software requirement specification document. The application will be fully described by the (SRS). The software requirement specification defines the document's purpose and its range of application. The project's scope will specify how the programme will operate. The (SRS) will provide the user with a thorough explanation of how it functions.

**2.1 Purpose**

This document is developed for the project **“ARIMALA GENERAL STORE”** to purchase online grocery items for customers through online. The customers can search different grocery items and they can also search items based on different quality and their prices can be viewed. The purpose of this document is to describe the external requirement of the project. The main purpose is to translate the ideas in the mind of the client into a formal document. A software requirements specification (SRS) minimizes the time and effort required by developers to achieve desired goals and also minimizes the development cost. A good SRS defines how an application will interact with system hardware, other programs, and human users in a wide variety of real-world situations. The SRS provides critical information about development operations, quality assurance, and maintenance to the developers.

**2.1 Scope**

The main objective of the project “**ARIMALA GENERAL STORE”** system is to provide an online facility for customers to purchase grocery items from the shop.Today is the era of computers. This software project solves all the problems experienced in the present system. The proposed system provides the following features:

* All the details of items available at the shop can be found.
* It enables the customer to find the different quality of products.
* It monitors the customer details and employee details.
* Current prices of items are displayed.
* Provide a dynamic menu.

**2.1 Definition,Acronyms,Abbreviations**

GUI - Graphical User Interface

DBMS - Database Management System

RDBMS - Relational Database Management System

SRS - Software Requirement Specification

ADMIN -The Administrator.

CPU -Central processing unit

SQL -Structured Query Language.

HTML -Hyper Text Markup Language.

CSS -Cascading style sheet

Python

Django

**2.1 Reference**

www.w3schools.com

[www.tutorialsapoint.com](http://www.tutorialsapoint.com/)

**2.1 Overview**

The project “**ARIMALA GENERAL STORE**” is developed to Search and order grocery items through online, different quality of items and their prices can be compared and then order the product online. admin can update the product details to the website and they can get the customer online. Here admin manages the application. Here customers can compare products and they can place orders.

**2.1 Overall Description**

An overview of the entire system will be provided in this section. The system's context will be discussed in order to introduce its basic capabilities and demonstrate how it interacts with other systems. Additionally, it will outline the different user types that the system will cater to and the features that each kind will have access to. Finally, the system's limitations and presumptions will be discussed.

**2.1 Product Perspective**

The **ARIMALA GENERAL STORE** programme is a stand-alone item that does not function as a component of any other system. The system's users are divided into two groups: admins and customers. Each individual user has access to the system online. This method offers clients a simple way to get high-quality groceries at reasonable prices. This Project is complete in itself. For high requirements, it offers a straightforward database, and it offers both new and seasoned system users an excellent and user-friendly graphical user interface (GUI). The primary goal of this project is to find a replacement for the current system, which required people to visit stores in order to purchase groceries.

**2.1 Product Functions**

• The "ARIMALA GENERAL SHOP" concept was created to give customers access to an online platform. where clients can browse all types of grocery things and admin can add their products online and get orders online.

• It oversees all internet activities.

• See all of the information pertaining to the grocery goods.

• It makes it possible for customers to find products of various quality.

• It makes it possible to keep a sizable stock of groceries.

• It makes it simple for customers to follow their deliveries.

• It offers quick access to various supermarket products.

• It gives administrators the opportunity to increase their consumer base online.

• With a web-based user-interactive interface, it increases the system's flexibility.

**2.1 User Classes and Characteristics**

Admin:

The Admin can login to the system using his/her unique credential. Here admin can manage the details of customer and Store.

* Login
* Profile
* View customer
* Manage Payments
* Manage Locations
* Manage Category
* View Feedback
* Manage Reports
* Manage Product Details
* Manage Order
* Employee pay
* Employee attendance

Customer:

With their customer ID and password, customers can log in. The system will produce a customer ID and password.

* Register
* Login
* Manage profile details
* View product details
* Add to cart
* Place an order
* Track the delivery
* Payment
* Write Feedback

**2.1 General Constraints:**

* To use this programme, you must have a working internet connection.
* To proceed, you must register and obtain a customer identification number.
* The administrator will update the pricing and stock.
* Client information is well-protected No other access to the application
* The final system should support seamless device failure recovery without data loss.

**2.1 Assumption and Dependencies:**

* Both administrators and customers use the system.
* Customer ID will receive admin approval.
* Client information is correctly maintained
* All information entered will be accurate and current.
* The application maintains a database to collect and store the data.

**2.1 SPECIFIC REQUIREMENTS:**

**2.1 External Interface Requirements:**

Functional requirements include needs for external interfaces. In terms of embedded systems, they are crucial. Also, they describe how your product will interact with other parts.

**2.1 User Interface:**

A user interface is a place where people may communicate and interact with a system.

During design, we kept the following requirements in mind:

* Labels to display the information.
* Checkboxes.
* Textboxes to enter details.
* Buttons to add, delete, update and search.
* Combo boxes and list boxes.
* Grid box to display the information

**2.1 Software Requirements:**

* Operating system: Windows.
* Text editor: Sublime Text 3
* Language: Python
* Server: Apache
* User interface: HTML, CSS, JavaScript
* Database: MySQL
* Browser: Chrome, Mozilla Fire, fox, or any other browsing application.

**2.1 Hardware Requirements:**

* Processors: Intel Pentium dual-core or above
* RAM: 2GB and above
* Hard disk Utilization: 40GB and above
* Input Devices: Mouse, Keyboard

**2.1 Communication Interface:**

### This product needs HTTP protocol for communications, while TCP/IP protocol is used for internet communication.

**2.1 Functional Requirements:**

**2.1 Functional Requirements:**

**Admin:**

* Login : In this module Admin can login by using their login credentials.
* Profile: This module shows the Admin ID,Email&Password.
* View customer: In this module admin can manage the customer details who are registered to the system.
* Manage Payments: In this module, the admin can manage all the payments made by the customers.
* Manage Locations: In this module, the admin can manage location details.
* Manage Category: In this module, the admin can manage the different category details.
* View Feedback: In this module, the admin can view feedback given by the customers.
* Manage Reports: In this module, the admin will manage all the report details of the system.
* Manage Product Details: In this module, the shop owner will add, delete and update the product details.
* Manage Order: In this module, the shop owner can view the orders made by the customers.
* Employee pay: In this module can check payment related to the employee.
* Employee attendance: This module helps the admin to check the employee attendance.

**Customer:**

* Register: The customer has to register by providing primary details before ordering.
* Login: In this module Customer can login by using their login credentials.
* Manage profile details: In this module customers can update his /her profile details.
* View product details: In this module, customers can view the respective shop details, the product details including the product description and specification.
* Add to cart: Customer can add the selected items to cart.
* Place an order: Customer can order items added to the cart.
* Track the delivery: Once the booking is made the customer can track the product delivery.
* Payment: Customer can pay for the orders through an online or offline method.
* Write Feedback: In this module, the Customer can write feedback.

**2.1 Performance Requirements:**

**•** The server must be able to handle any amount of active consumer payments in other words, no payments should ever be lost.

• A page should load in under 40 seconds.

• It need to have plenty of memory.

• The server must be able to support an unlimited number of active orders in other words, orders must never be lost.

• It must be free of errors.

• A 1MB file ought to upload in 60 seconds.

**2.1 Design Constraints:**

• When a customer registers for a system, mandatory fields must be reviewed to ensure that the customer entered the correct information in those fields.

• When purchasing things from the cart, consumers must complete all required fields.

• When adding profile information, customers must complete all required fields.

• The system needs to be created so that it will be simple to use and accessible on most browsers.

**2.1 System Attributes:**

Users of the database will find it to be very user-friendly because of the high quality standards that are upheld for the database.

• Performance: Every web browser should be able to manage all users at once on the system.

• Reliability: To prevent inaccurate record storage, user inputs will be thoroughly validated.

• User-friendly Interfaces: It is assumed that screen layouts that are self-explanatory or simple to use would result in higher usage quality.

• Maintainability: The capacity of a system to alter with some ease is known as maintainability.

• Portability: This system needs to be portable, and switching the server is a breeze.

• Flexibility: The system will continuously update the data in accordance with the transactions that occur.

• Timing: The all operations will take only less time.

**2.1 Other Requirements:**

**2.1 Safety Requirements:**

• The **ARIMALA GENERAL STORE** has two user levels. A user log-in screen that requires a username and password will serve as protection against unauthorised access to the various subsystems. This provides several views and user-level functions that are accessible through the system.

• Client ID, which makes every user distinct and easily recognised, cannot be modified once registered with the system.

• Keeping backups ensures the database security of the system. In the event of an emergency, the system can be restored.

**2.1 Security Requirements:**

• The **"ARIMALA GENERAL STORE"** server will have security measures in place to guard against unauthorised write and delete access. There are no limitations on who can read.

• The website under consideration will be safe. Users can be divided into three groups: administrators, store owners, and customers.

• Access privileges are determined according to the type of use.

• The administrator has full access to all subsystems.

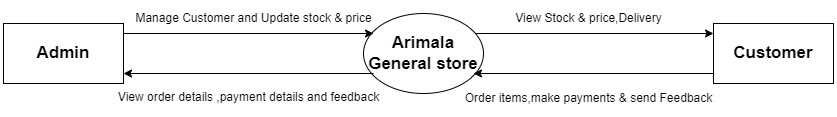
**3.SYSTEM DESIGN**

**3.1 Introduction:**

System design is the process of outlining every requirement for a system, including the architecture, modules, interface, and design. As a result, we can say that system design encompasses everything from determining the needs of the system to creating new products. The procedures, practises, and approaches used to design a system are modified through system creation or system modification. In order to manage the system requirements and design technique, a systematic strategy is required. It can be divided into logical design and physical design categories. The system's input and output operations are represented by the physical design, whilst the abstract dataflow is represented by the logical design. The primary structural elements of the system are under the control of the system design. It significantly affects the system's testability and modifiability. The architectural plan for the software system that will be created is the output.

**3.1 Context Flow Diagram:**

A top-level data flow diagram is a context flow diagram. There is only one process node that generalises how the entire system works in regard to outside things. The context diagram treats the entire system as a single process and shows all of its sources, sinks, inputs, and outputs.



**3.1 Data flow diagram:**

A data flow diagram represents a system or a part of a system graphically. The description is stored using symbols that are simple to grasp and includes data flows, processes, sources, and sinks.

In terms of modelling tools, DFD ranks among the most significant. The system, its components, and the ways in which data and information move through the system are all modelled using it.

The DFD illustrates the flow of information and how it is transformed throughout. Information movements from input or output are shown graphically using this technique.

The terms "DFD" and "data flow graphs" are also used to describe DFDs. Any level of abstraction can be utilised with DFD to portray the system. DFDs might fragment.

**3.1 Rules Regarding DFD Construction:**

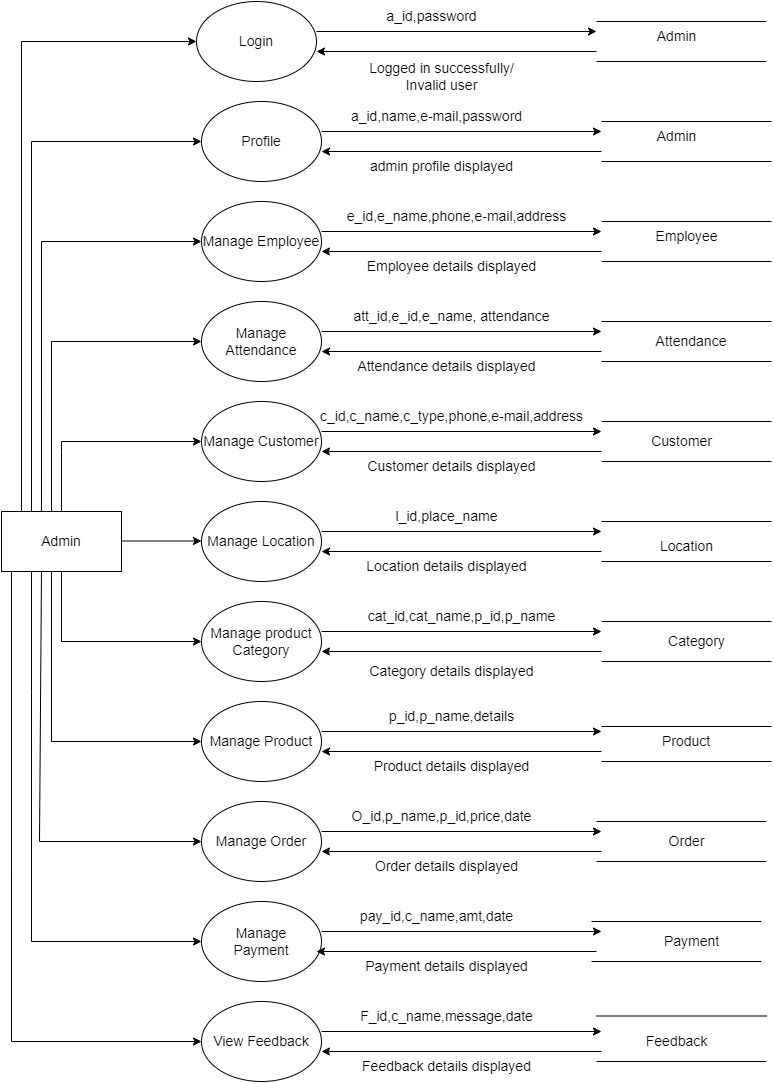
* A process cannot just provide results.
* Just inputs cannot be used in a process.
* The inputs to a process must be adequate to generate the process's outputs.
* At least one process must be linked to each data storage.
* Each and every data store needs to be joined to a source or sink.
* A data flow can only flow in one way. Separate arrows must be used to depict different data flows to and/or from the same process and data repository.
* A forked arrow should be used to indicate a situation in which the exact same data goes to two different shafts.
* Data cannot enter the process it just left and then immediately return. A noun phrase must be used to name all data flows.

## 

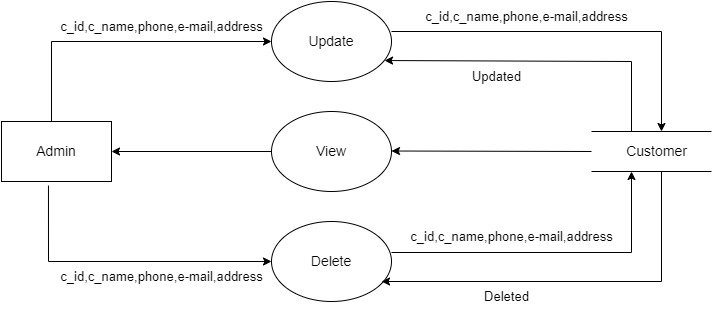
**3.1 DFD Symbols:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Name** | **Notation** | **Description** |  |  |  |
| **Process** |  | A process transforms incoming data flow into outgoing data flow. The processes are shown by named circles. |  |  |  |
| **Datastore** |  | Data stores are repositories of data in the system.  They are sometimes also referred to as files. |  |  |  |
| **Dataflows** |  | Data flows are pipelines through which packets of information flow. Label the arrows with the name of the data that moves through it. |  |  |  |
| **External Entity** |  | External entities are objects outside the system with which the system communicates. External Entities are sources and destinations of the system’s inputs and outputs |  |  |  |

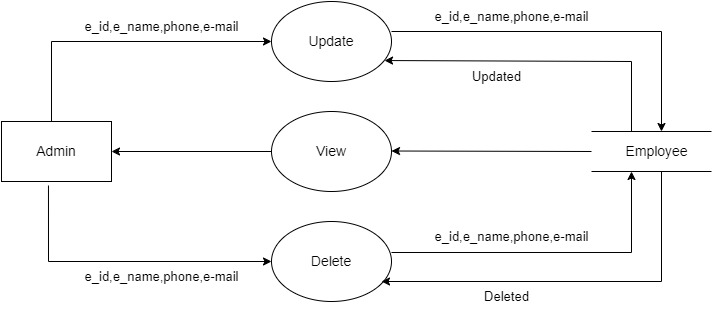
**3.1 DFD Level 1(Admin):**

****

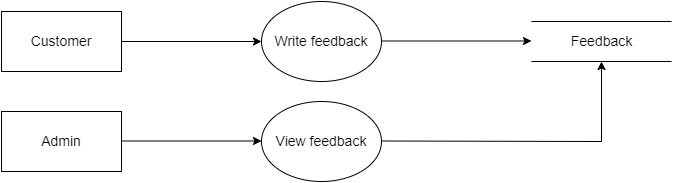
**3.1 DFD Level 2(MANAGE CUSTOMER):**

****

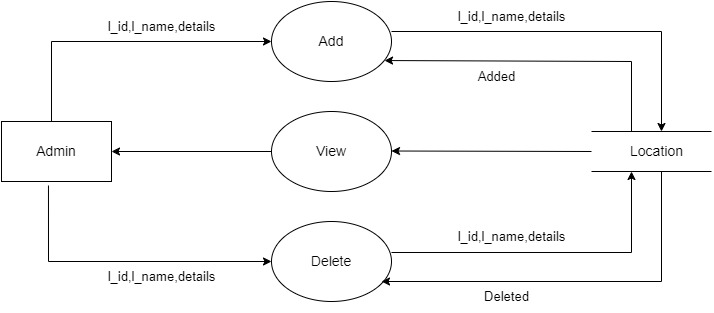
**3.1 DFD Level 2(MANAGE EMPLOYEE):**

****

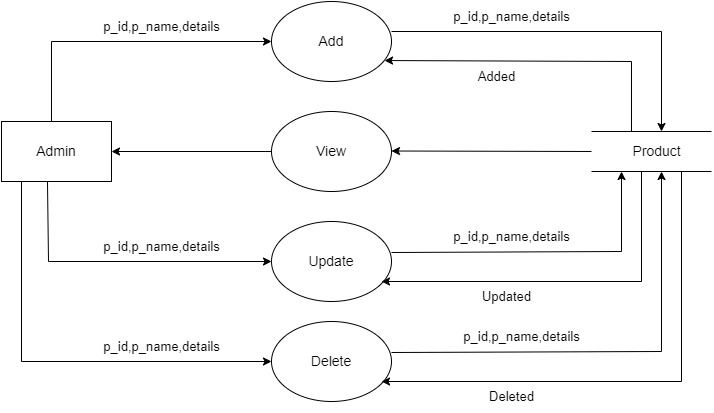
**3.1 DFD Level 2(VIEW FEEDBACK):**

****

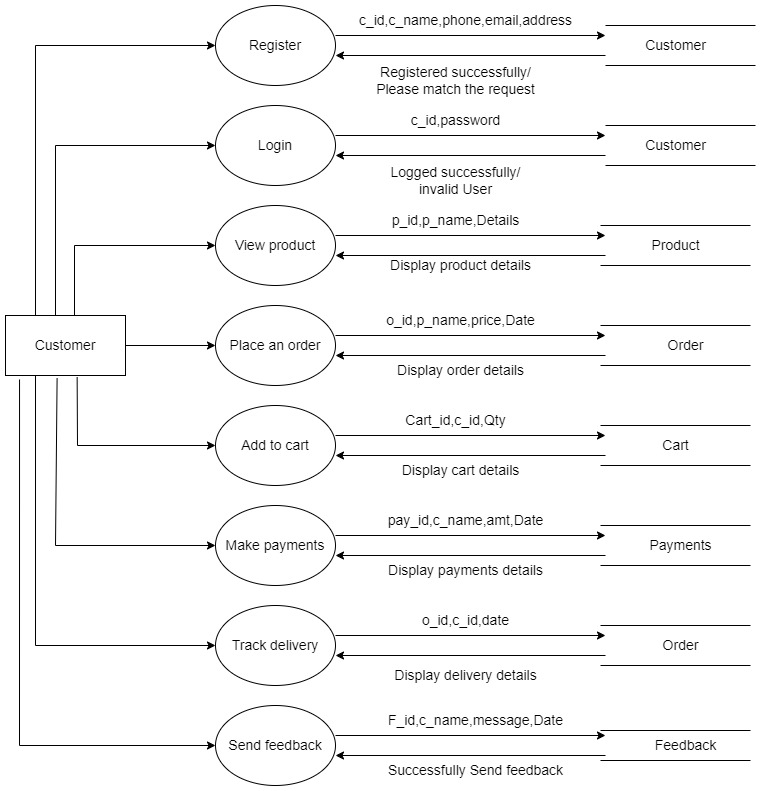
**3.1 DFD Level 2(MANAGE LOCATION):**

****

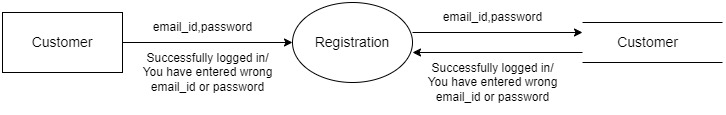
**3.1 DFD Level 2(MANAGE PRODUCT):**

****

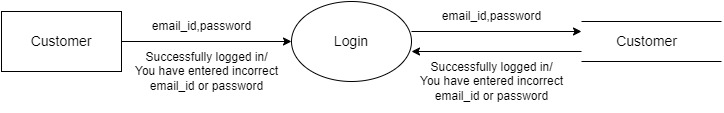
**3.1 DFD Level 2(CUSTOMER):**

****

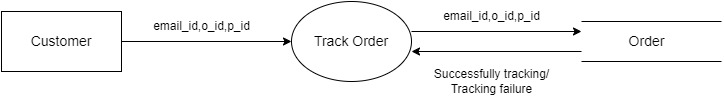
**3.1 DFD Level 2(REGISTRATION):**

****

**3.1 DFD Level 2(LOGIN):**

****

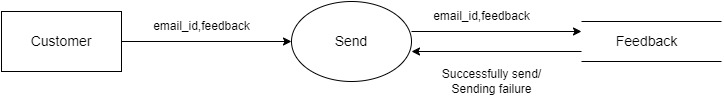
**3.1 DFD Level 2(TRACK ORDER):**

****

**3.1 DFD Level 2(PAYMENT):**

****

**3.1 DFD Level 2(FEEDBACK):**

****

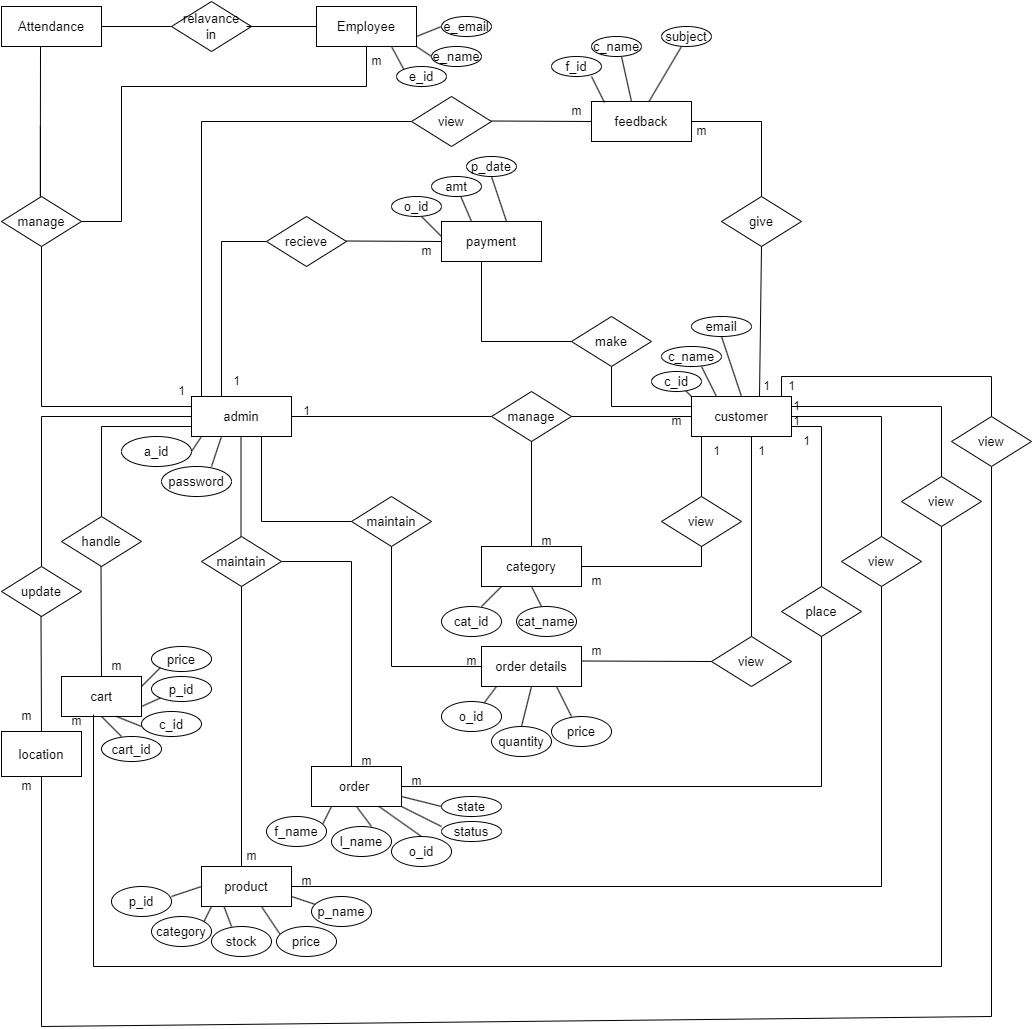
**3.1 Entity-Relationship Diagram:**

An entity-relationship (ER) diagram is a specialized graphic that illustrates the relationships between entities in a database. ER diagrams often use symbols to represent three different types of information. Boxes are commonly used to represent entities. Diamonds are normally used to represent relationships and ovals are used to represent attributes.

Symbols are shown in below table:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Name** | | **Notation** | | **Description** | |
| **Entity** | | Entity name | | It may be an object with the physical existence or conceptual existence. It is represented by a Rectangle. | |
| **Attribute** | | Attribute name | | The properties of the entity can be a attribute. It is represented by a Ellipse. | |
| **Relationship** | | Relation | | Whenever an attribute of one entity refers to another entity, some relationship exists. It is represented by a Diamond. | |
| **Link** | |  | | Lines link attributes to entity sets and entity sets to relation. | |
| **Derived Attribute** | |  | | Dashed ellipse denote derived attributes. | |
| **Key Attribute** | | Key    Attribute | | An entity type usually has an attribute whose values are distinct for each individual entry in the entity set. It is represented by aUnderlined word in ellipse. | |
| **Multivalued**  **Attribute** | | Multi  -  valued    attribute | | Attributes that have different numbers of values for a particular attribute.It is represented by a Double ellipse represents multi-valued attributes. | |
| **Cardinality Ratio** | | 1. 1:1 2. 1:M 3. M:1 4. M:M | | It specifies the maximum number of relationships instances that an entity can participate in. There are four cardinality ratios. | |

**3.1 ER-Diagram:**

****

**4. Database Design**

**3.1 Introduction:**

* **Database**: A Database is collection of related data, which can be of any size and complexity. By using the concept of Database, we can easily store and retrieve the data. The major purpose of a database is to provide the information, which utilizes it with the information’s that the system needs according to its own requirements.
* **Database Design:** Database design is done before building it to meet needs of endusers within a given information-system that the database is intended to support. The database design defines the needed data and data structures that such a database comprises The database is physically implemented using MySQL.

**4.1 The database for “Chickenfarm” is organized into 11 tables:**

* Admin
* Employee
* Attendance
* Customer
* Location
* Category
* Product
* Cart
* Order
* Payment
* Feedback

Each entity can be described as follows along with its attributes:

**4.3 Database Table Structure**

* + 1. **Structure of Table “admin”:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Field Type** | **Size** | **CONSTRAINTS** | **Description** |
| a\_id | int | 50 | PRIMARY KEY | Admin ID |
| a\_name | varchar | 50 | NOT NULL | Admin Name |
| email | varchar | 50 | UNIQUE | E-mail |
| password | Varchar | 50 | NOTNULL | Password |

* + 1. **Structure of Table “employee”:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Field Type** | **Size** | **CONSTRAINTS** | **Description** |
| E\_id | int | 50 | PRIMARY KEY | Employee ID |
| e\_name | varchar | 50 | NOT NULL | Employee Name |
| e\_image | varchar | 50 | NOT NULL | Employee Image |
| e\_phone | bigint | 11 | NOTNULL | Employee Phone No |
| email | varchar | 50 | UNIQUE | Employee E-mail |
| e\_address | varchar | 50 | NOT NULL | Employee Address |
| e\_doj | date | - | NOT NULL | Date of joining |

* + 1. **Structure of Table “Attendance”:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Field Type** | **Size** | **CONSTRAINTS** | **Description** |
| atten\_id | int | 50 | PRIMARY KEY | Attendance ID |
| e\_id | int | 50 | FOREIGN KEY | Employee ID |

**4.3.4 Structure of Table “Customer”:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Field Type** | **Size** | **CONSTRAINTS** | **Description** |
| c\_id | int | 50 | PRIMARY KEY | Customer ID |
| c\_name | varchar | 50 | NOT NULL | Customer Name |
| email | varchar | 50 | UNIQUE | Customer E-mail |
| address | varchar | 50 | NOT NULL | Customer Address |
| Phone no | bigint | 11 | NOT NULL | Customer Phone no |
| c\_type | varchar | 50 | NOT NULL | Customer type |
| password | Varchar | 50 | NOTNULL | Password |

**4.3.5 Structure of Table “Location”:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Field Type** | **Size** | **CONSTRAINTS** | **Description** |
| l\_id | int | 50 | PRIMARY KEY | Location ID |
| l\_name | varchar | 50 | NOT NULL | Location Name |

**4.3.6 Structure of Table “Category”:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Field Type** | **Size** | **CONSTRAINTS** | **Description** |
| cat\_id | int | 50 | PRIMARY KEY | Category ID |
| cat\_name | varchar | 50 | NOT NULL | Category Name |
| p\_id | varchar | 50 | FOREIGN KEY | Product ID |
| p\_name | varchar | 50 | FOREIGN KEY | Product Name |

**4.3.7 Structure of Table “Product”:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Field Type** | **Size** | **CONSTRAINTS** | **Description** |
| p\_id | int | 50 | PRIMARY KEY | Product id |
| p\_name | varchar | 50 | NOT NULL | Name of the product |
| stock | int | 50 | NOT NULL | Stock |
| price\_w | decimal | 50,2 | NOT NULL | Price for wholesale customer |
| price\_r | int | 11 | NOT NULL | Price for Customer |
| Image | varchar | 50 | NOT NULL | Image |
| o\_id | int | 50 | FOREIGN KEY | Order ID |

**4.3.8 Structure of Table “Cart”:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Field Type** | **Size** | **CONSTRAINTS** | **Description** |
| cart\_id | int | 50 | PRIMARY KEY | Cart ID |
| p\_id | varchar | 50 | FOREIGN KEY | Product ID |
| c\_id | varchar | 50 | FOREIGN KEY | Customer ID |
| c\_type | varchar | 50 | NOT NULL | Type of customer |
| t\_price | decimal | 50,2 | NOT NULL | Total price |

**4.3.9 Structure of Table “Order”:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Field Type** | **Size** | **CONSTRAINTS** | **Description** |
| o\_id | int | 50 | PRIMARY KEY | Order ID |
| cart\_id | int | 50 | FOREIGN KEY | Cart ID |
| c\_id | Int | 50 | FOREIGN KEY | Customer Id |
| sub\_amt | Decimal | 50,2 | NOT NULL | Original amount |
| psub\_amt | Decimal | 50,2 | NOT NULL | Discounted amount |
| total\_amt | Decimal | 50,2 | NOT NULL | Total Amount |
| p\_id | Int | 50 | FOREIGN KEY | Product ID |
| qty | Int | 50 | NOT NULL | Quantity |
| od\_id | Int | 50 | FOREIGN KEY | Order id |
| status | int | 11 | NOT NULL | Status |

**4.3.10 Structure of Table “Payment”:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Field Type** | **Size** | **CONSTRAINTS** | **Description** |
| pay\_id | int | 50 | PRIMARY KEY | Payment ID |
| c\_id | Int | 50 | FOREIGN KEY | Customer ID |
| o\_id | int | 50 | FOREIGN KEY | Order ID |
| amt | decimal | 50,2 | NOT NULL | Amount |
| date | date | - | NOT NULL | Date of payment |

**4.3.11 Structure of Table “Feedback”:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Field Type** | **Size** | **CONSTRAINTS** | **Description** |
| f\_id | Int | 50 | PRIMARY KEY | Feedback ID |
| c\_id | Int | 50 | FOREIGN KEY | Customer ID |
| message | varchar | 100 | NOT NULL | Message |
| date | date | - | NOT NULL | Date |

**5.** **Detailed Design**

**5.1. Introduction**

The purpose of preparing this document is to explain complete design details of Arimala General Store. This detailed design report will mainly contain the general definition and features of the project, design constraints, the overall system architecture and data architecture. Additionally, a brief explanation about our current progress and schedule of the project will be provided in related sections. Design of the system and subsystems/modules will be explained both verbally and visually by means of diagrams in order to help the programmer to understand all information stated in this document correctly and easily .