VISVESVARAYA TECHNOLOGICAL UNIVERSITY

JNANA SANGAMA, BELAGAVI – 590 018

A Mini Project Report on

MOBILE STORE SYSTEM

Submitted in partial fulfillment of the requirements as a part of the DBMS Lab for the V Semester of degree of ***Bachelor of Engineering in Information Science and Engineering*** *of Visvesvaraya Technological University, Belagavi*

Submitted by

Karthik raj R
1RN19IS068

Chinmay Hegde
1RN19IS053

Under the Guidance of

Faculty Incharge
Dr. R Rajkumar
Asso. Professor
Dept. of ISE, RNSIT

Lab Incharge
Dr. R Rajkumar
Asso. Professor
Dept. of ISE, RNSIT

Department of Information Science and Engineering

RNS Institute of Technology

Channasandra, Dr. Vishnuvardhan Road, RR Nagar Post,
Bengaluru – 560 098

2021 – 2022

RNS Institute of Technology
Channasandra, Dr.Vishnuvardhan Road, RR Nagar Post,
Bengaluru – 560 098

DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING

ESTD:2001
An Institute with a Difference

CERTIFICATE

This is to certify that the Mini project report entitled ***MOBILE STORE SYSTEM*** has been successfully completed by **KARTHIK RAJ R** bearing USN **1RN19IS068** and **CHINMAY HEGDE** bearing USN **1RN19IS053**, presently V semester student of **RNS Institute of Technology** in partial fulfillment of the requirements as a part of the DBMS Laboratory for the award of the degree ***Bachelor of Engineering in Information Science and Engineering*** under **Visvesvaraya Technological University, Belagavi** during academic year 2021 – 2022. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the report deposited in the departmental library. The mini project report has been approved as it satisfies the academic requirements as a part of DBMS Laboratory for the said degree.

________________	____________________
Dr R Rajkumar	**Dr. Suresh L**
Faculty & Guide Incharge	Professor and HOD
Asso. Prof, Dept. of ISE	Dept. of ISE

External Viva

Name of the Examiners	**Signature with date**
1. ________________	________________
2. ________________	________________

DECLARATION

We, **CHINMAY HEGDE [USN: 1RN19IS053]** and **KARTHIK RAJ R [USN: 1RN19IS068]** students of V Semester BE, in Information Science and Engineering, RNS Institute of Technology hereby declare that the Mini project entitled ***Mobile Store System has*** been carried out by us and submitted in partial fulfillment of the requirements for the *V Semester degree of* ***Bachelor of Engineering in Information Science and Engineering*** *of Visvesvaraya Technological University, Belagavi* during academic year 2021-2022.

Place: Bengaluru

Date :

CHINMAY HEGDE (1RN19IS068)

KARTHIK RAJ R (1RN19IS068)

ABSTRACT

The project Sales & and purchase management system for mobile is to develop software based information of Mobile shopping. In global business market the mobile is most important accessory in real life. Visualizing the huge opportunity, this is an effort to maximize the business through the development of this software and keeping the data and thus increasing the customer base from the local as well as global markets around the world.

This project is a software application because nowadays software is a prominent tool of marketing. With the advent of the software technologies, world has become a global village. Every year, millions more people around the world are added to the existing customer base. So considering a big hike in the revenue in this booming sector and one of the successful businesses through this software one should be proud to have such a technical deal.

ACKNOWLEDGMENT

The fulfillment and rapture that go with the fruitful finishing of any assignment would be inadequate without the specifying the people who made it conceivable, whose steady direction and support delegated the endeavors with success.

We would like to profoundly thank **Management** of **RNS Institute of Technology** for providing such a healthy environment to carry out this Project work.

We would like to express our thanks to our Principal **Dr. M K Venkatesha** for his support and inspired me towards the attainment of knowledge.

We wish to place on record our words of gratitude to **Dr. Suresh L,** Professor and Head of the Department, Information Science and Engineering, for being the enzyme and master mind behind our Project work.

We would like to express our profound and cordial gratitude to our faculty & guide incharge **Dr. R Rajkumar**, Associate Professor, Department of Information Science and Engineering for his valuable guidance in preparing Project report.

We would like to thank all other teaching and non-teaching staff of Information Science & Engineering who have directly or indirectly helped me to carry out the project work.

And lastly,we would hereby acknowledge and thank our parents who have been a source of inspiration and also instrumental in carrying out this Project work.

KARTHIK RAJ R
USN: 1RN19IS068

CHINMAY HEGDE
USN: 1RN19IS053

TABLE OF CONTENTS

LIST OF FIGURES

ABBREVIATIONS

API	-	Application Programming Interface
CRS	-	Computer Reservation System
CSS	-	Cascading style sheets
DBMS	-	Database Management System
ER	-	Entity Relationship
HTML	-	Hypertext Markup Language
HTTP	-	Hypertext Transfer Protocol
ID	-	Identification
JS	-	JavaScript
MVD	-	Multi Valued Dependency
SQL	-	Structured Query Language

Chapter 1

INTRODUCTION

1.1 Background

A **database** is an organized collection of data, generally stored and accessed electronically from a computer system. Where databases are more complex they are often developed using formal design and modeling techniques.

The database management system (DBMS) is the software that interacts with end users, applications, the database itself to capture and analyze the data and provides facilities to administer the database. The sum total of the database, the DBMS and the associated applications can be referred to as a "database system". Often the term "database" is also used to loosely refer to any of the DBMS, the database system or an application associated with the database. The DBMS manages three important things: the data, the database engine that allows data to be accessed, locked and modified and the database schema, which defines the database's logical structure. These three foundational elements help provide concurrency, security, data integrity and uniform administration procedures. Typical database administration tasks supported by the DBMS include change management, performance monitoring/tuning and backup and recovery. Many database management systems are also responsible for automated rollbacks, restarts and recovery as well as the logging and auditing of activity.

1.2 Introduction to Mobile Store System

The Mobile Store Management System is developed for desktop systems to facilitate mobile shop owners management of customer details and inventory data, which includes mobile phones. It can be used efficiently for physically separated shops in different locations.

This software will provide in a simple and easy to operate user interface, which can be managed by any store employee without having prior in-depth knowledge of the computer system. One can use this software to get order reports. Administrators can pull data, from any location from the database. This software is a complete package for stores which will allow them to keep track of their sales and inventory, and provide a computerized billing system.

There are various applications with more complex implementation and features available in the market, but they are generally very expensive.

Therefore, creating an application with the basic requirement of low cost is essential for small stores. This application will allow stores to manage customer details, keep inventory of all products and purchase information, in a very simple way, using a state-of the-art software application. It will automatically generate invoices and update inventory.

In today’s market, retailers and wholesale outlets should quickly adapt to the ever changing technology to minimize overhead, lower cost of operation, and help to stay competitive. Everybody needs software, which can facilitate store operations and make their day-to-day lives much easier.

Mobile Store Management System is application software designed to take advantage of today’s technology and reduce or avoid the burden of storing data on paper and in files. This facilitates moving purchase, sales, and customer information, as well as supplier and company data, from paper to digital media on a secured server.

Sales and purchase bills can be generated as needed.Each store has an option to store their data on one database server..

Chapter 2

E R DIAGRAM AND RELATIONAL SCHEMA DIAGRAM

2.1 Description of ER Diagram in SQL

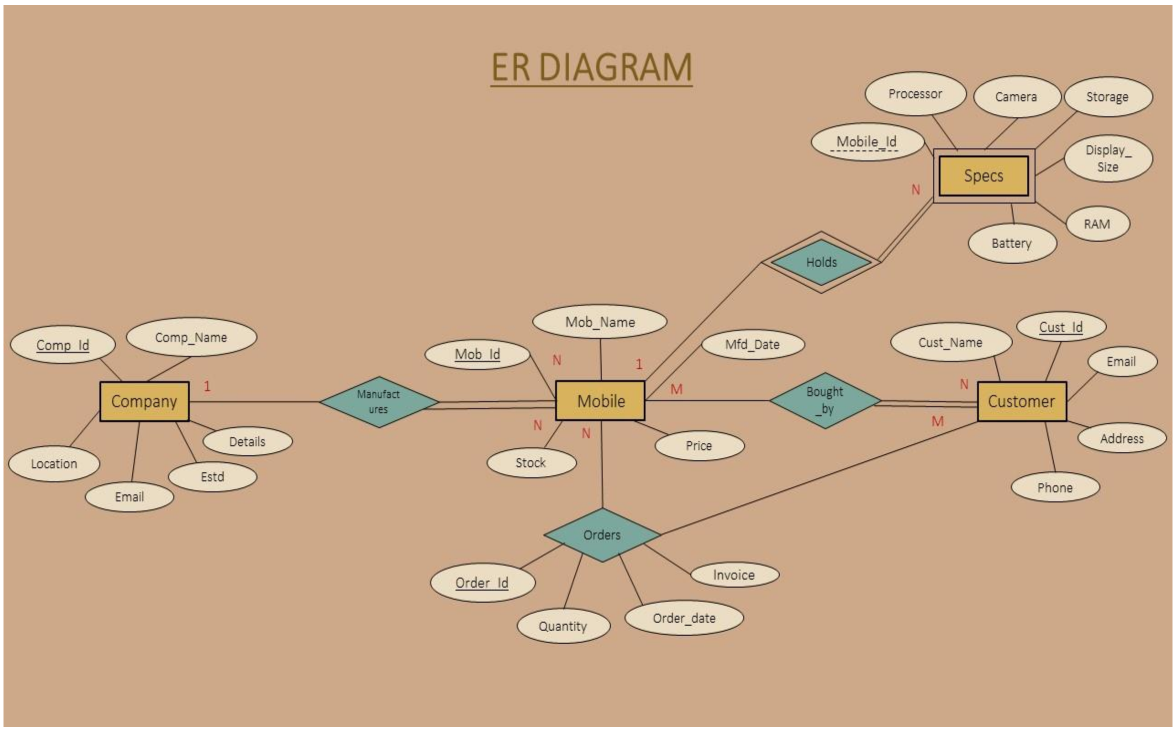


Figure 2.1: E-R Diagram for Mobile Store System

Entity relationship diagram displays the relationships of entity set stored in a database. In other words, we can say that ER diagrams help you to explain the logical structure of databases. At first look, an ER diagram looks very similar to the flowchart. However, ER Diagram includes many specialized symbols, and its meanings make this model unique.

2.1.1 Components of Mobile Store System, E-R Diagram

Entity types like **COMPANY , MOBILE , CUSTOMER** and **SPECS** are in rectangular boxes.

1. Relationships like **ORDERS** are in diamond boxes, attached to entity types with straight lines.
2. Attributes are shown in ovals like **Mob_name** and **Comp_Name**, each attached by a straight line to entity or relationship type.
3. Key attributes like **Comp_ID, Order_ID ,Cust_ID** and **Mob_ID_code** are underlined.
4. Weak entity types like **SPECS** are in double rectangular boxes and Identifying relations like **Holds** are in double diamond boxes
5. Multivalued attributes like in **Phone,** are represented using double layered oval.

2.1.2 E-R Diagram Relationships Description

1. **COMPANY:MOBILE** is of cardinality 1:N as one company can possibly have more than one mobile and therefore connected by an **MANUFACTURES** relationship. There is total participation from MOBILE as all mobiles belong to the same company, but partial participation from COMPANY as all mobiles are not manufactured by same company .
2. **MOBILE:CUSTOMER** is of cardinality M:N as many mobiles can be brought by many customers and therefore connected by an **BROUGHT_BY** relationship. There is total participation from CUSTOMER as all customer don’t by all the mobiles, but partial participation from MOBILE as all mobiles are not by customer.
3. **MOBILE:SPECS** is of cardinality of 1:N as each mobile can have many specifications and therefore connected by an **HOLDS** relationship. There is total participation from SPECS as all the specifications belong the mobile itself but partial participation from MOBILE as all mobiles do not have all the specifications .
4. **CUSTOMER:MOBILE** is of cardinality M:N as many customers can order many mobiles from the store and therefore connected by an **ORDERS** relationship . There is partial participation from both the relationships as all customer do not order all the mobiles **.**

2.1.3 MONGO DATABASE

MongoDB, the most popular NoSQL database, is an open-source document-oriented database. The term 'NoSQL' means 'non-relational'. It means that MongoDB isn't based on the table-like relational database structure but provides an altogether different mechanism for storage and retrieval of data. This format of storage is called BSON (similar to JSON format).
SQL databases store data in tabular format. This data is stored in a predefined data model which is not very much flexible for today's real-world highly growing applications. **Modern applications are more networked, social and interactive than ever**. Applications are storing more and more data and are accessing it at higher rates.
Relational Database Management System(RDBMS) i**s not the correct choice when it comes to handling big data by the virtue of their design since they are not horizontally scalable**. If the database runs on a single server, then it will reach a scaling limit. NoSQL databases are more scalable and provide superior performance. MongoDB is such a NoSQL database that scales by adding more and more servers and increases productivity with its flexible document model.
Without indexing, a database would have to scan every document of a collection to select those that match the query which would be inefficient. So, for efficient searching Indexing is a must and MongoDB uses it to process huge volumes of data in very less time.

2.1.4 COLLECTIONS IN MONGODB OF MOBILE STORE SYSTEM

USERS : This collection consists of a **Username , Email and Password** attributes which are used for the login authentication of employees in the Store and then using Mongo and Passport the password in the collection is hashed and additional salt is added to the password for security purposes**.**

COMPLAINT: This collection consists of a **Mobile_ID , Cust_ID, Rating and Complaint Details(body)** which is used in the store to get the customer complaints for a particular mobile so that the store can addresss to the company to fix the issues in the mobile. Hence it will be also helpful for the employess of the store to recommend mobiles to the customer.

2.2 Description of Relational Schema Diagram

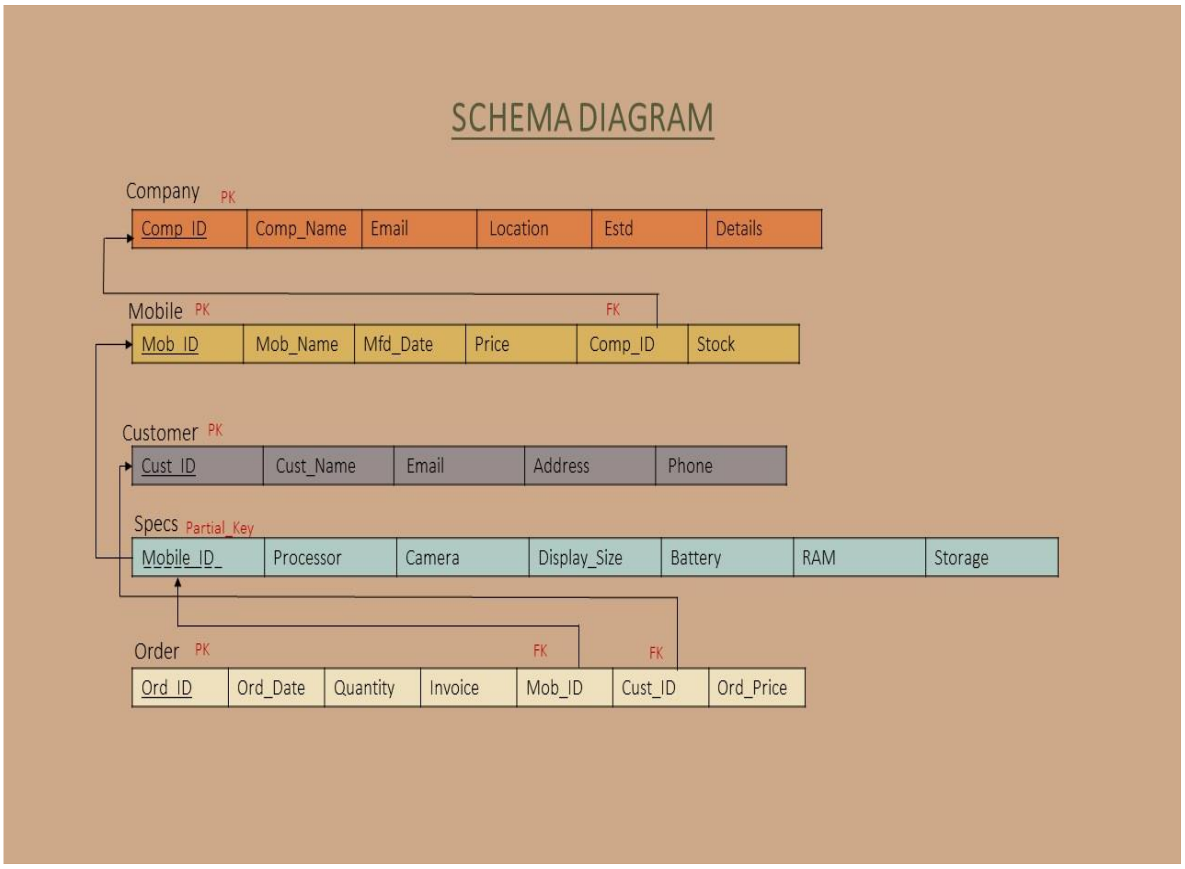


Figure 2.2 Relational Schema Diagram for Mobile Store System

2.2.1 General Constraints

1. **NULL Constraint**: Attributes that are under NOT NULL constraints have to be filled compulsorily.
2. **Entity Integrity Constraint**: This constraint makes sure that no primary key can have a NULL value assigned to it. The primary keys involved in the project include:
    - Comp_ID
    - Mob_ID
    - Customer_ID
    - Ord_ID

3. **Referential Integrity Constraints**: A table in the back end of the project may have references pointing to an attribute in another table. For example: COMP_ID in the MOBILE table refers to COMP_ID in COMPANY table. The various tables are also linked with multiple foreign keys which are all set to cascade or set null on any update or delete operation on the attribute in the main table. The various Foreign Key attributes are:
    - Comp_ID
    - Mob_ID
    - Cust_ID

2.2.2 Schema Description of SQL

The above Figure.2.2 shows the relational schema of Mobile Store System. It has the following entities.

1. **COMPANY:** This table contains the details like Comp_ID, Comp_Name , Email, Location, Estd and Details . Comp_ID is the primary key.
2. **MOBILE:** This table contains the details of all the mobiles in the store . It has 6 columns, Mob_ID ,Mob_Name ,Mfd_Date ,Price, Stock and Comp_ID referring to entry in previous table. Mob_ID is the primary key.
3. **CUSTOMER:** This table consists the list of customers who have ordered mobiles from the store . It has 5 columns Cust_ID, Cust_Name, Email,Address and Phone. Cust_ID is the primary key.
4. **SPECS:** This table consists of all the specifications for a particular mobile. It has 7 columns Mobile_ID, Processor, Camera, Display_Size, Ram, Storage and Battery. Here Mobile_ID is a partial key.
5. **ORDERS:** This table consists of list of all the orders from a customer to a particular mobile .It has 7 columns Ord_ID, Ord_Date, Quantity,Ord_Price, Invoice no,Mob_ID and Cust_ID. Here Ord_ID is the primary key and Cust_ID and Mob_ID are foreign keys respectively.

2.2.3 Schema Description of Mongo Database

1. **USERS:** This collection contains the details like Username , Email and Password . In MongoDB it generates id for each record of values it is represented as _id.
2. **COMPLAINT:** This collection contains the details of all the complaints of mobiles from a particular customer . It consists of Cust_ID, Mobile_ID, Complaint Details(body) and Rating.

THE SCHEMA FOR BOTH COLLECTION ARE GIVEN BELOW :-

1. **USERS:**

```
const passportLocalMongoose = require('passport-local-mongoose');
const UserSchema = new Schema({
   email: {
      type: String,
      required: true,
      unique: true
   }
});
```

   Here passport takes username and password automatically from its modules automatically.

2. **COMPLAINT:**

```
const complaintSchema = new Schema({
   body: String,
   mob_id: Number,
   cust_id: Number,
   rating: Number
});
```

Chapter 3

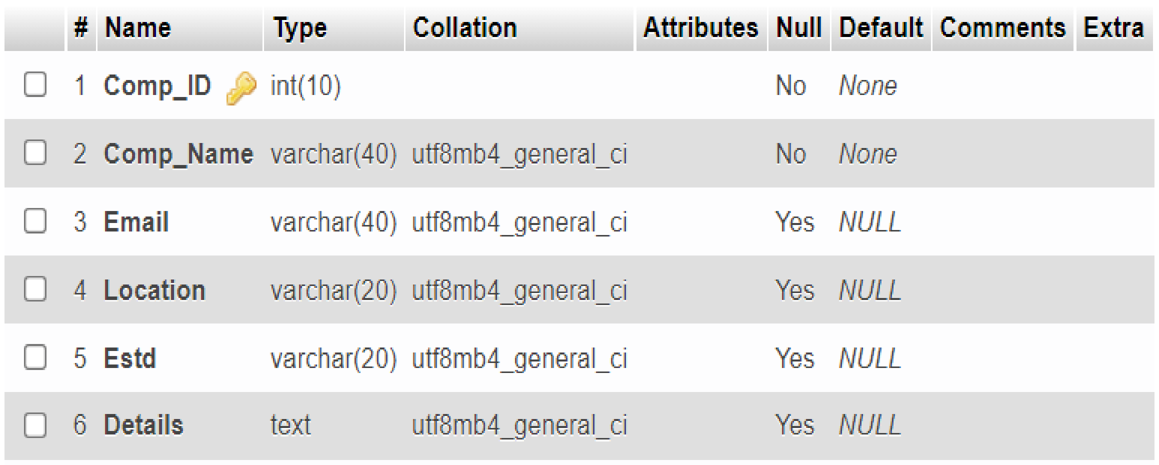
SYSTEM DESIGN

3.1.1 Table Description for SQL

There are total tables implemented in this project. The tables are:

Following are the contents of the table**:**

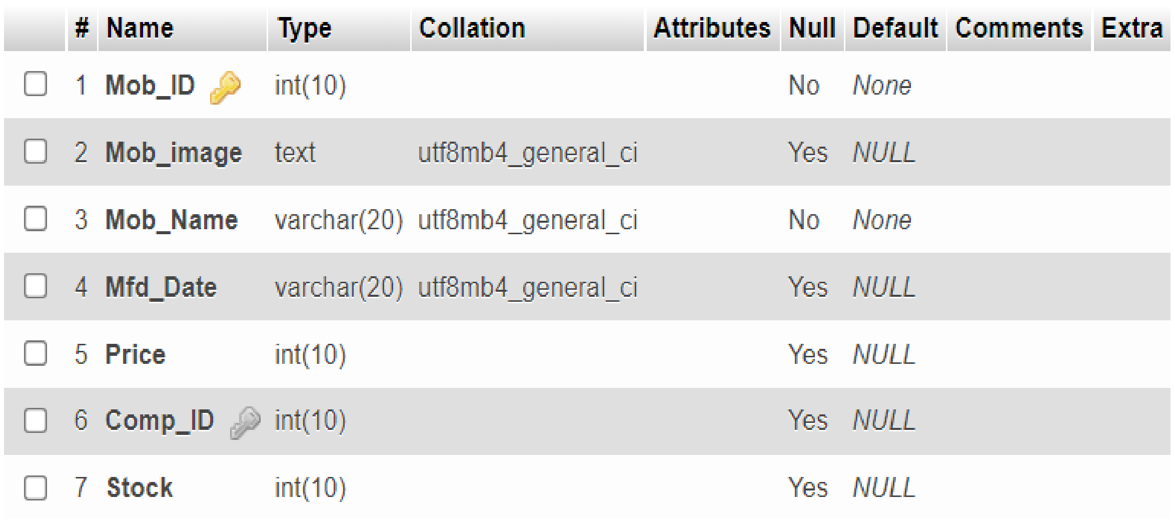
COMPANY:



	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
☐	1	Comp_ID	int(10)			No	*None*		
☐	2	Comp_Name	varchar(40)	utf8mb4_general_ci		No	*None*		
☐	3	Email	varchar(40)	utf8mb4_general_ci		Yes	*NULL*		
☐	4	Location	varchar(20)	utf8mb4_general_ci		Yes	*NULL*		
☐	5	Estd	varchar(20)	utf8mb4_general_ci		Yes	*NULL*		
☐	6	Details	text	utf8mb4_general_ci		Yes	*NULL*		

Table 3.1 Company

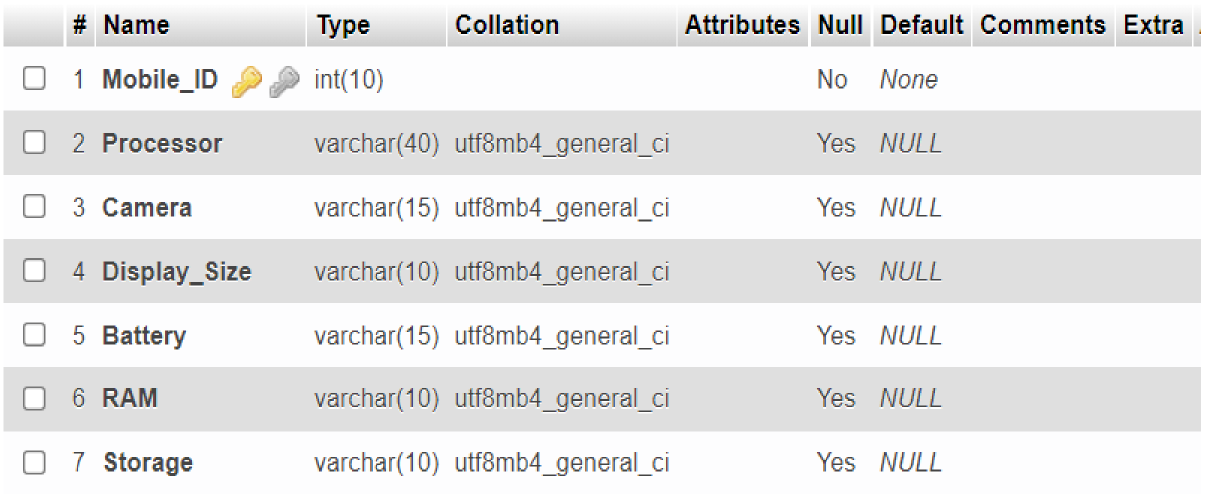
MOBILE:



	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
☐	1	Mob_ID	int(10)			No	*None*		
☐	2	Mob_image	text	utf8mb4_general_ci		Yes	*NULL*		
☐	3	Mob_Name	varchar(20)	utf8mb4_general_ci		No	*None*		
☐	4	Mfd_Date	varchar(20)	utf8mb4_general_ci		Yes	*NULL*		
☐	5	Price	int(10)			Yes	*NULL*		
☐	6	Comp_ID	int(10)			Yes	*NULL*		
☐	7	Stock	int(10)			Yes	*NULL*		

Table 3.2 Mobile

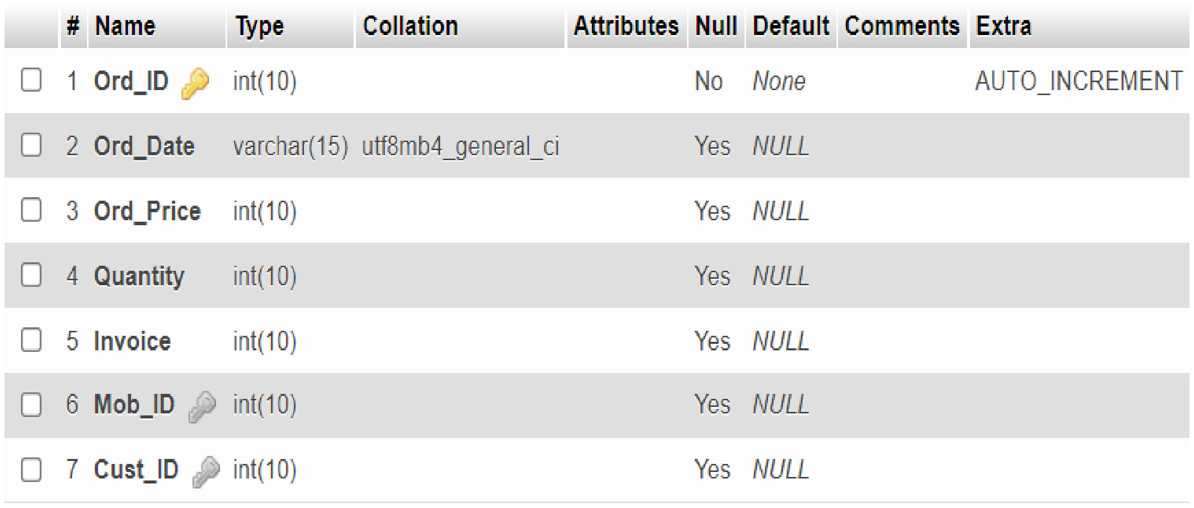
SPECS:



#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	Mobile_ID	int(10)			No	*None*		
2	Processor	varchar(40)	utf8mb4_general_ci		Yes	*NULL*		
3	Camera	varchar(15)	utf8mb4_general_ci		Yes	*NULL*		
4	Display_Size	varchar(10)	utf8mb4_general_ci		Yes	*NULL*		
5	Battery	varchar(15)	utf8mb4_general_ci		Yes	*NULL*		
6	RAM	varchar(10)	utf8mb4_general_ci		Yes	*NULL*		
7	Storage	varchar(10)	utf8mb4_general_ci		Yes	*NULL*		

Table 3.3 Specs

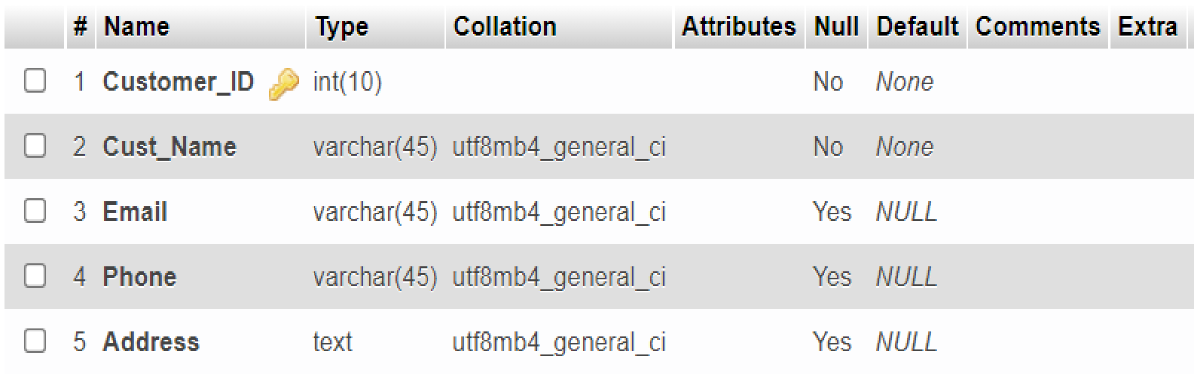
ORDER:



#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	Ord_ID	int(10)			No	*None*		AUTO_INCREMENT
2	Ord_Date	varchar(15)	utf8mb4_general_ci		Yes	*NULL*		
3	Ord_Price	int(10)			Yes	*NULL*		
4	Quantity	int(10)			Yes	*NULL*		
5	Invoice	int(10)			Yes	*NULL*		
6	Mob_ID	int(10)			Yes	*NULL*		
7	Cust_ID	int(10)			Yes	*NULL*		

Table 3.4 Order

CUSTOMER:



#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	Customer_ID	int(10)			No	*None*		
2	Cust_Name	varchar(45)	utf8mb4_general_ci		No	*None*		
3	Email	varchar(45)	utf8mb4_general_ci		Yes	*NULL*		
4	Phone	varchar(45)	utf8mb4_general_ci		Yes	*NULL*		
5	Address	text	utf8mb4_general_ci		Yes	*NULL*		

Table 3.5 Customer

3.1.1 Table Description for MONGODB

USERS:

_NAME	TYPE
_id	**Int32**
username	**string**
email	**string**
password	**string**

This collection is used to store the login details of the employees of the store.

COMPLAINT:

NAME	TYPE
_id	**Int32**
Mob_ID	**number**
Cust_ID	**number**
Rating	**number**
Complaint Details (body)	**string**

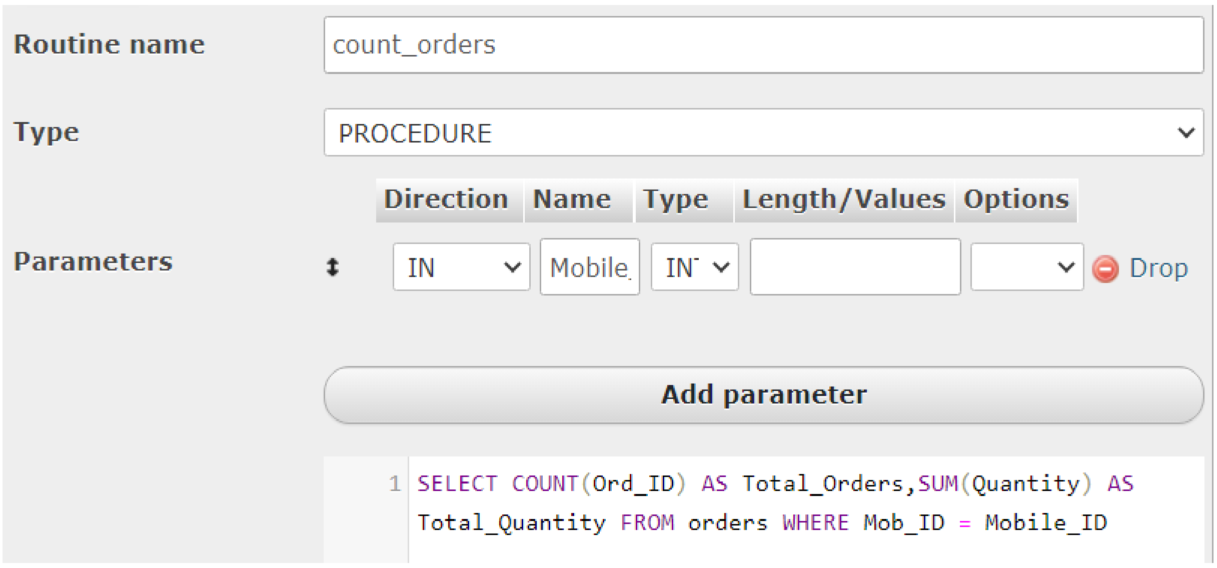
This collection is used to store the complaint description of a customer for a particular mobile.

3.2 Stored Procedures

This procedure is used to get the total number of orders for a particular mobile and total quantity of the mobiles are ordered from the store. The name of the procedure is **count_orders** where is operates on the orders table. This procedure takes one input parameter Mob_ID from the user.

CODE:

CREATE DEFINER=`root`@`localhost` PROCEDURE `count_orders` (IN `Mobile_ID` INT) SELECT COUNT(Ord_ID) AS Total_Orders,SUM(Quantity) AS Total_Quantity FROM orders WHERE Mob_ID = Mobile_ID$$


Routine name: count_orders

Type: PROCEDURE

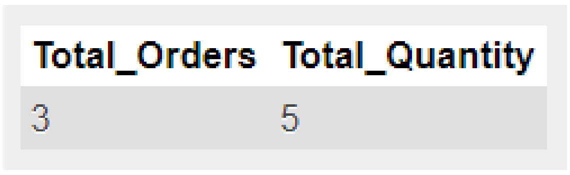
Parameters:

Direction	Name	Type	Length/Values	Options
IN	Mobile_	IN		

```
SELECT COUNT(Ord_ID) AS Total_Orders,SUM(Quantity) AS
Total_Quantity FROM orders WHERE Mob_ID = Mobile_ID
```

EXAMPLE:

To find the number of orders and total quantity for a particular mobile which has Mob_ID = 1



Total_Orders	Total_Quantity
3	5

3.3 Trigger

A trigger is a stored procedure in database which automatically invokes whenever a special event in the database occurs. For example, a trigger can be invoked when a row is inserted into a specified table or when certain table columns are being updated.

1) In this project a trigger is added to reduce the total stock of a mobile left after a mobile is ordered from the store.

CODE:

```
CREATE TRIGGER `stock_update_1`
    AFTER INSERT ON `orders`
            FOR EACH ROW
                    UPDATE mobile SET Stock=Stock-NEW.Quantity
                            WHERE mobile.Mob_ID=NEW.Mob_ID
```

The above trigger is called whenever an mobile is order from the store depending on the quantity of the order for a mobile and remaining stock is of the mobile is updated.

2) Another set of triggers is used in the project to maintain the LOGS of the insertion,updation and deletion of mobiles with the DATE and TIME of when the operation is done.

CODE:

- For insert operation

```
CREATE TRIGGER `mobile_insert`
  AFTER INSERT ON `mobile`
    FOR EACH ROW
         INSERT INTO trig
         VALUES(null,NEW.Mob_ID,NEW.Mob_name,NEW.Mfd_Date
         ,NEW.Price,NEW.Comp_ID,NEW.Stock,'INSERTED',
         CURRENT_DATE(),CURRENT_TIME())
```

- For update operation

```
CREATE TRIGGER `mobile_update`
  AFTER UPDATE ON `mobile`
     FOR EACH ROW
          INSERT INTO trig
          VALUES(null,NEW.Mob_ID,NEW.Mob_name,NEW.Mfd_Date,NEW.Price
          ,NEW.Comp_ID,NEW.Stock,'UPDATED',
          CURRENT_DATE(),CURRENT_TIME())
```

- For Deletion operation

```
CREATE TRIGGER `mobile_delete`
   BEFORE DELETE ON `mobile`
     FOR EACH ROW
          INSERT INTO trig
          VALUES(null,OLD.Mob_ID,OLD.Mob_name,OLD.Mfd_Date,OLD.Price,O
          LD.Comp_ID,OLD.Stock,'DELETED',
          CURRENT_DATE(),CURRENT_TIME())
```

These three triggers helps to maintain the logs of insertion, updation and deletion and the Date and Time of the event.

3)Another trigger is used which stops the insertion of order if the Quantity entered is more than the stock available.

CODE:

```
CREATE TRIGGER `stock_update_2`
        BEFORE INSERT ON `orders`
                FOR EACH ROW
                BEGIN
                IF((SELECT mobile.Stock from mobile WHERE
                mobile.Mob_ID=NEW.Mob_ID)<NEW.Quantity) THEN
                        SIGNAL sqlstate '45001' set message_text = "Quantity is more than
                        Stock !";
                END if;
                END
```

Chapter 4

IMPLEMENTATION

4.1 Front-end Development

The front-end is built using a combination of technologies such as Hypertext Markup Language (HTML), JavaScript and Cascading Style Sheets (CSS). Front-end developers design and construct the user experience elements on the web page or app including buttons, menus, pages, links, graphics and more.

4.1.1 Hypertext Markup Language

HTML is a computer language devised to allow website creation. These websites can then be viewed by anyone else connected to the Internet. It is relatively easy to learn, with the basics being accessible to most people in one sitting; and quite powerful in what it allows you to create. HTML is the standard markup language for creating Web pages. It stands for Hyper Text Markup Language. It describes the structure of a Web page. It consists of a series of elements. It elements tell the browser how to display the content. It elements are represented by tags. HTML tags label pieces of content such as "heading", "paragraph", "table", and so on. Browsers do not display the HTML tags, but use them to render the content of the page.

4.1.2 Cascading style sheets

CSS stands for Cascading Style Sheets. It is a style sheet language which is used to describe the look and formatting of a document written in markup language. It provides an additional feature to HTML. It is generally used with HTML to change the style of web pages and user interfaces. CSS is used along with HTML and JavaScript in most websites to create user interfaces for web applications. Before CSS, tags like font, color, background style, element alignments, border and size had to be repeated on every web page. This was a very long process. CSS solved that issue. CSS style definitions are saved in external CSS files so it is possible to change the entire website by changing just one file. CSS provides more detailed attributes than plain HTML to define the look and feel of the website.

4.1.3 JavaScript

JavaScript is a dynamic computer programming language. It is lightweight and most commonly used as a part of web pages, whose implementations allow client-side script to interact with the user and make dynamic pages. It is an interpreted programming language with object-oriented capabilities. Client-side JavaScript is the most common form of the language. The script should be included in or referenced by an HTML document for the code to be interpreted by the browser. It means that a web page need not be a static HTML, but can include programs that interact with the user, control the browser, and dynamically create HTML content. The JavaScript client-side mechanism provides many advantages over traditional CGI server-side scripts. The JavaScript code is executed when the user submits the form, and only if all the entries are valid, they would be submitted to the Web Server. JavaScript can be used to trap user-initiated events such as button clicks, link navigation, and other actions that the user initiates explicitly or implicitly. Advantages are: **Less server interaction, immediate feedback to the visitors, increased interactivity** and **richer interfaces.**

4.2 Back-end Development

Backend is server side of the website. It stores and arranges data, and also makes sure everything on the client-side of the website works fine. It is the part of the website that you cannot see and interact with. It is the portion of software that does not come in direct contact with the users. The parts and characteristics developed by backend designers are indirectly accessed by users through a front-end application. Activities, like writing APIs, creating libraries, and working with system components without user interfaces or even systems of scientific programming, are also included in the backend.

4.2.1 Backend – Node.js (Runtime environment)

Node.js is an open-source and cross-platform JavaScript runtime environment. It is a popular tool for almost any kind of project!

Node.js runs the V8 JavaScript engine, the core of Google Chrome, outside of the browser. This allows Node.js to be very performant.

A Node.js app runs in a single process, without creating a new thread for every request. Node.js provides a set of asynchronous I/O primitives in its standard library that prevent JavaScript code from blocking and generally, libraries in Node.js are written using non-blocking paradigms, making blocking behavior the exception rather than the norm.

When Node.js performs an I/O operation, like reading from the network, accessing a database or the filesystem, instead of blocking the thread and wasting CPU cycles waiting, Node.js will resume the operations when the response comes back.

4.2.2 Backend Web Framework -Express.js

EXPRESS is the most popular *Node* web framework, and is the underlying library for a number of other popular **Node web frameworks**. It provides mechanisms to:

- Write handlers for requests with different HTTP verbs at different URL paths (routes).
- Integrate with "view" rendering engines in order to generate responses by inserting data into templates.
- Set common web application settings like the port to use for connecting, and the location of templates that are used for rendering the response.
- Add additional request processing "middleware" at any point within the request handling pipeline.

While *Express* itself is fairly minimalist, developers have created compatible middleware packages to address almost any web development problem. There are libraries to work with cookies, sessions, user logins, URL parameters, POST data, security headers, and *many* more. You can find a list of middleware packages maintained by the Express team at **Express Middleware** (along with a list of some popular 3rd party packages).

4.2.3 Web Server – Apache

Apache is an open-source and free web server software that powers around 46% of websites around the world. The official name is Apache HTTP Server, and it's maintained and developed by the Apache Software Foundation. It allows website owners to serve content on the web — hence the name "web server". Although we call Apache a web server, it is not a physical server, but rather a software that runs on a server.

Its job is to establish a connection between a server and the browsers of website visitors (Firefox, Google Chrome, Safari, etc.) while delivering files back and forth between them (client-server structure). Apache is a cross-platform software, therefore it works on both UNIX and Windows servers.

4.2.4 Database – MySQL

MySQL is a fast, easy-to-use RDBMS being used for many small and big businesses. It is developed, marketed and supported by MySQL AB, which is a Swedish company. It is released under an open-source license. So you have nothing to pay to use it. It is a very powerful program in its own right. It handles a large subset of the functionality of the most expensive and powerful database packages. It uses a standard form of the well-known SQL data language. It works on many operating systems and with many languages including PHP, PERL, C, C++, JAVA, etc. It works very quickly and works well even with large data sets. It is very friendly to PHP, the most appreciated language for web development. MySQL supports large databases, up to 50 million rows or more in a table. The default file size limit for a table is 4GB, but you can increase this (if your operating system can handle it) to a theoretical limit of 8 million terabytes (TB). It is customizable. The open-source GPL license allows programmers to modify the MySQL software to fit their own specific environments

4.2.4 Database – Mongo

MongoDB, the most popular NoSQL database, is an open-source document-oriented database. The term 'NoSQL' means 'non-relational'. It means that MongoDB isn't based on the table-like relational database structure but provides an altogether different mechanism for storage and retrieval of data. This format of storage is called BSON (similar to JSON format).
SQL databases store data in tabular format. This data is stored in a predefined data model which is not very much flexible for today's real-world highly growing applications. **Modern applications are more networked, social and interactive than ever**. Applications are storing more and more data and are accessing it at higher rates.

4.3 Data Flow Diagram

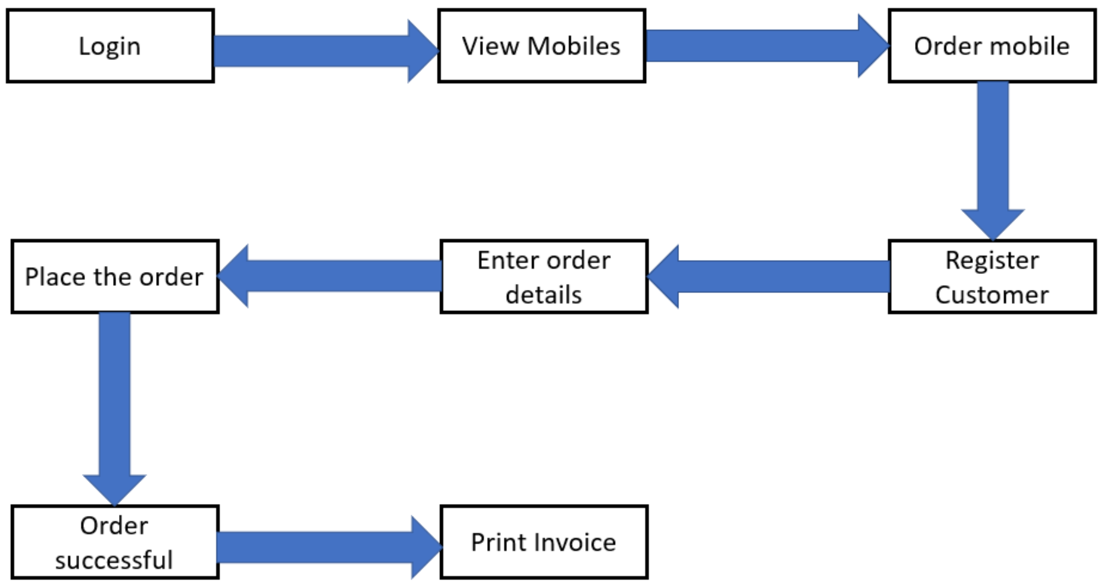


Figure 4.1 Data flow diagram

The above Figure 4.1 shows the Data flow diagram for a successful ordering. The store employee logs in, finds the mobile which customer wants, chooses the mobile from the available list and clicks on order, then he registers the customer, and enters the order details and confirms the order, on payment successful he is allowed to take the print of invoice receipt.

4.4 Discussion of Code Segment

This section talks about the important code sections and modules that are implemented in the Mobile Store System project. These modules add logic to the complete system, and make it function the way it is supposed to. It also talks about the integration between the front end HTML code and the back end MySQL database and also MongoDB.

4.4.1 Login Module

Pseudocode Login:

```
username := input()
password := input()
if(submit.click()) then
    if (validate(username, password) then
        make post request to /login
```

```
          passport.authenicate({username,password})
          if(passport.success) then
             session['_id'] := _id
             session['username'] := username
             print(true)
          else
             print(false)
          end if;
       if true then
          print("Welcome back!")
       end if;
    else
       print("Incorrect Username or password")
    end if;
 end if;
```

4.4.2 Register module

Pseudocode Signup:

```
 username := input()
 email_id:= input()
 password := input()

    if(submit.click()) then
       if(validate(username,email,password)) then
```

```
          make post request to /register
             const user = new User({email,username})
             const registerUser = await User.register(user,password)
             if(registerUser) then
                print(true)
             else
                print(false)
             end if;
          if true then
             print('Register Successful')
             redirect('/mobile')
          end if;
       else
          print("Incorrect Username or password")
       end if;
    end if;
```

4.4.3 Add Mobile module

Pseudocode Add_mobiles:

```
details := array()
details[Mobile_name] := input()
details[Mfd_Date] := input()
details[Price] := input()
details[Comp_ID] := input()
details[Stock] := input()
if(validate(details)) then
   make post request to /mobile/addmobile
   insert details into mobile
```

4.4.4 Order Mobile module

Pseudocode Order_Mobile:

```
orderdetails := array ()
orderdetails := input ()
if (already_existing_customer) then
            print (true)
else
    customerdetails := array ()
    customerdetails := input ()
     insert customerdetails into customer
end if;
if true then
        insert orderdetails into order
        Call trigger 'stock_update'
        Print("Order added successfully")
        Print invoice
else
        print("register the customer")
        Redirect /customer/addcustomer
end if;
```

4.5 Discussion of Results

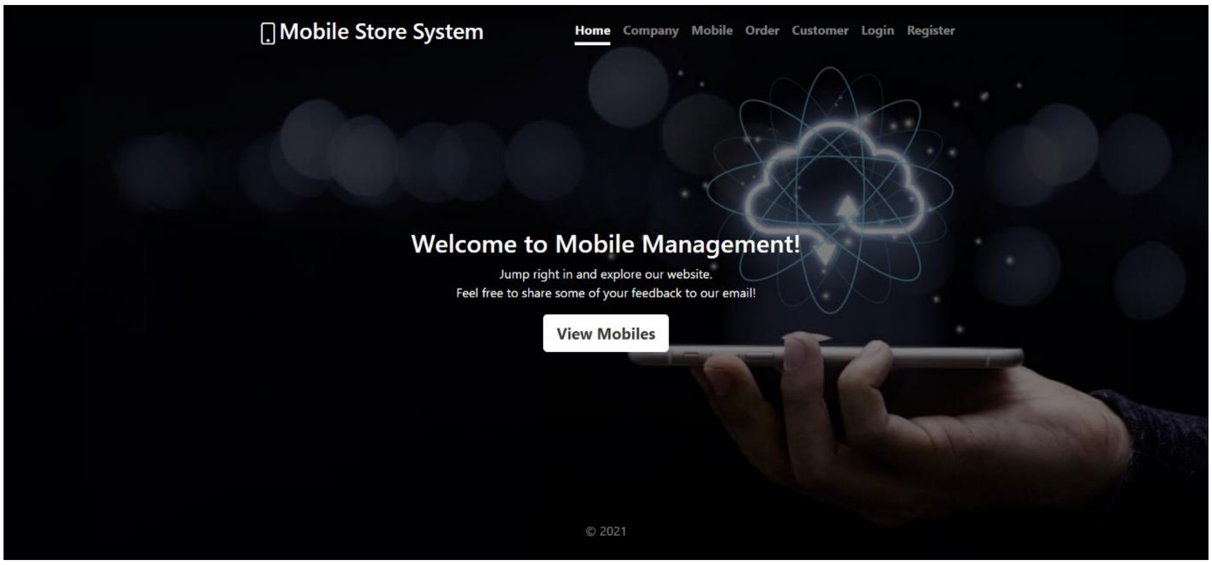

Mobile Store System
Home
Company
Mobile
Order
Customer
Login
Register
Welcome to Mobile Management!
Jump right in and explore our website.
Feel free to share some of your feedback to our email!
View Mobiles
© 2021

Figure 4.2 Homepage

The above Figure 4.2 is the snapshot of the homepage with the navigation bar on top. If the store employee has already registered with the website, he can login using username which he used to register along with the valid password also which he set during registration.

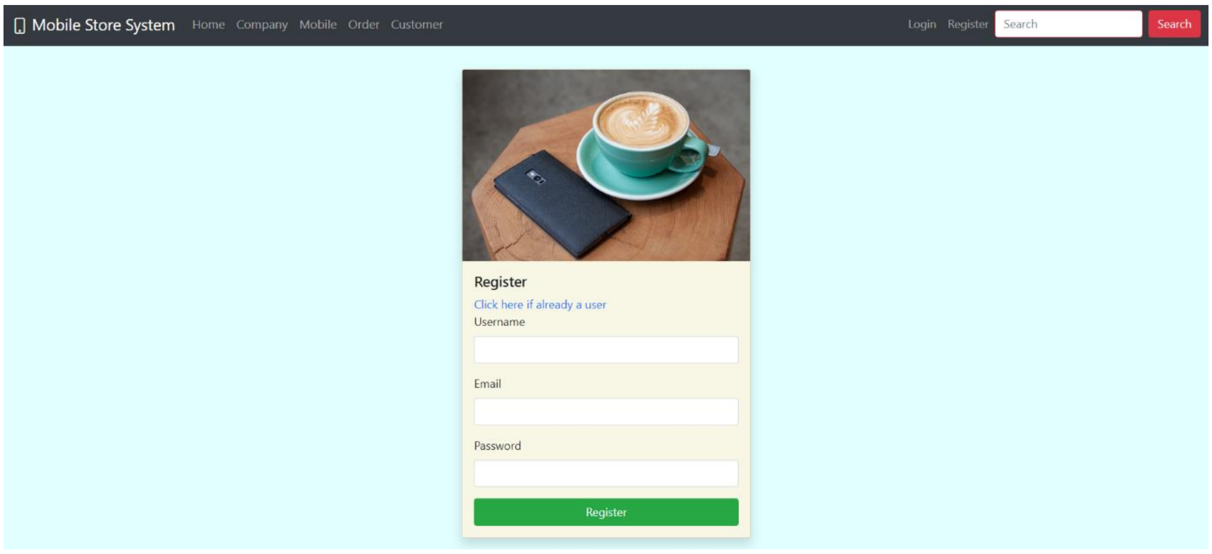


Figure 4.3 Register Page

This is the register page, if a store employee wants to register with the website, he can use this form to register. It accepts the username, email ID and password .

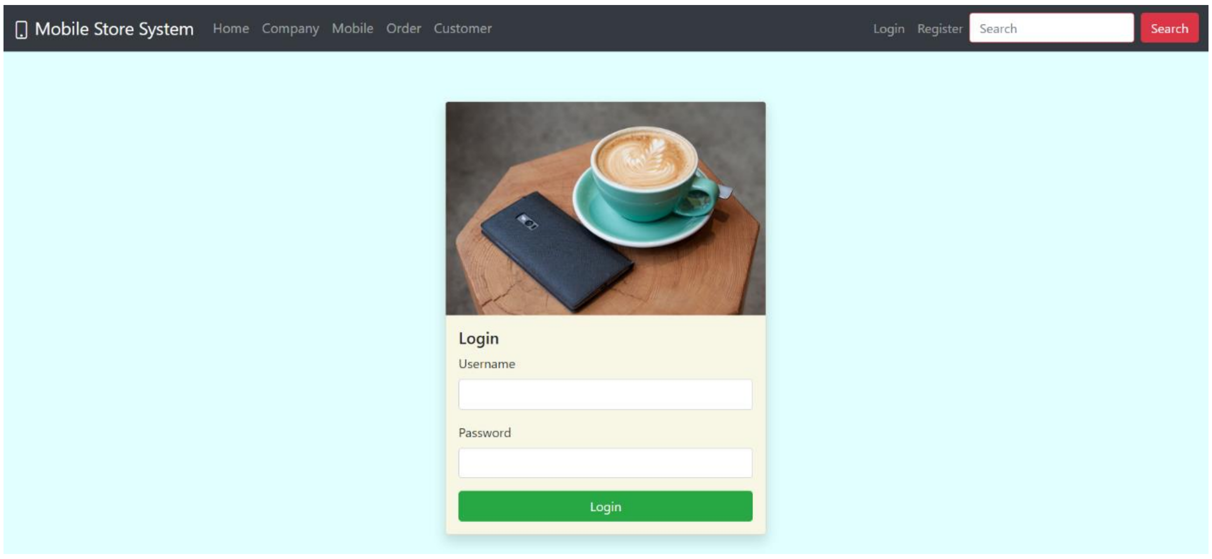
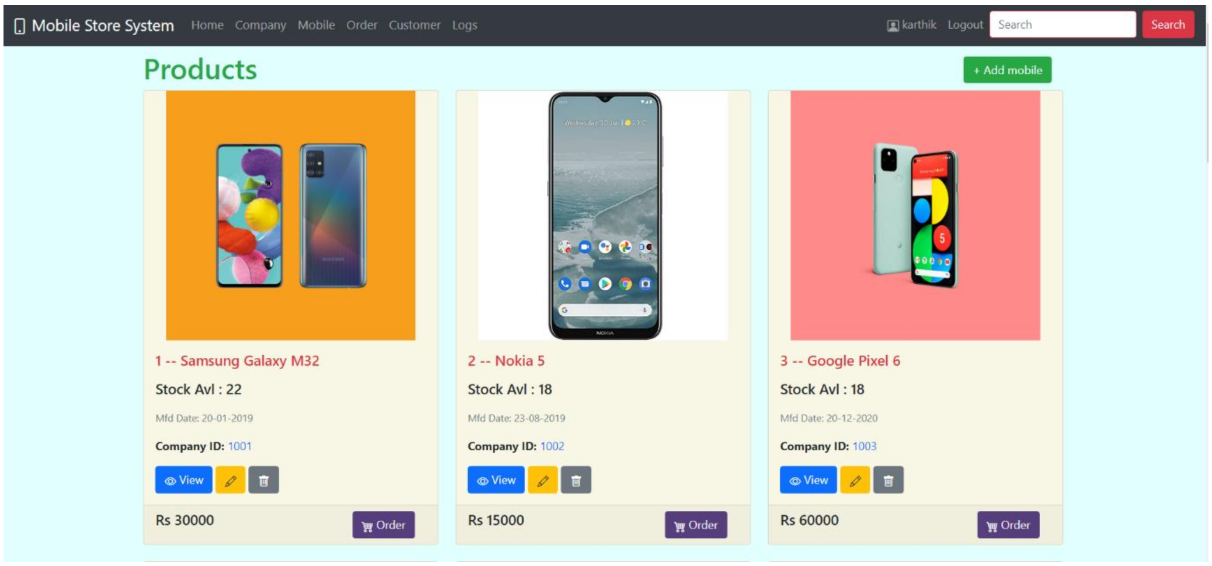


Figure 4.4 Login page

This is the login page using which the store employee can login into the website using his registered username and password.



1 -- Samsung Galaxy M32
Stock Avl : 22
Mfd Date: 20-01-2019
Company ID: 1001
Rs 30000

2 -- Nokia 5
Stock Avl : 18
Mfd Date: 23-08-2019
Company ID: 1002
Rs 15000

3 -- Google Pixel 6
Stock Avl : 18
Mfd Date: 20-12-2020
Company ID: 1003
Rs 60000

Figure 4.5 All mobiles page

The above figure 4.5 shows the all the mobiles available in the store
It also shows the Mobile Name , price of the mobile its manufactured date and to which company the mobile belongs too.
Using the displayed buttons the employee can view the mobile details , edit the details , delete and order the mobile for a customer. Using the search button he can search a particular mobile.

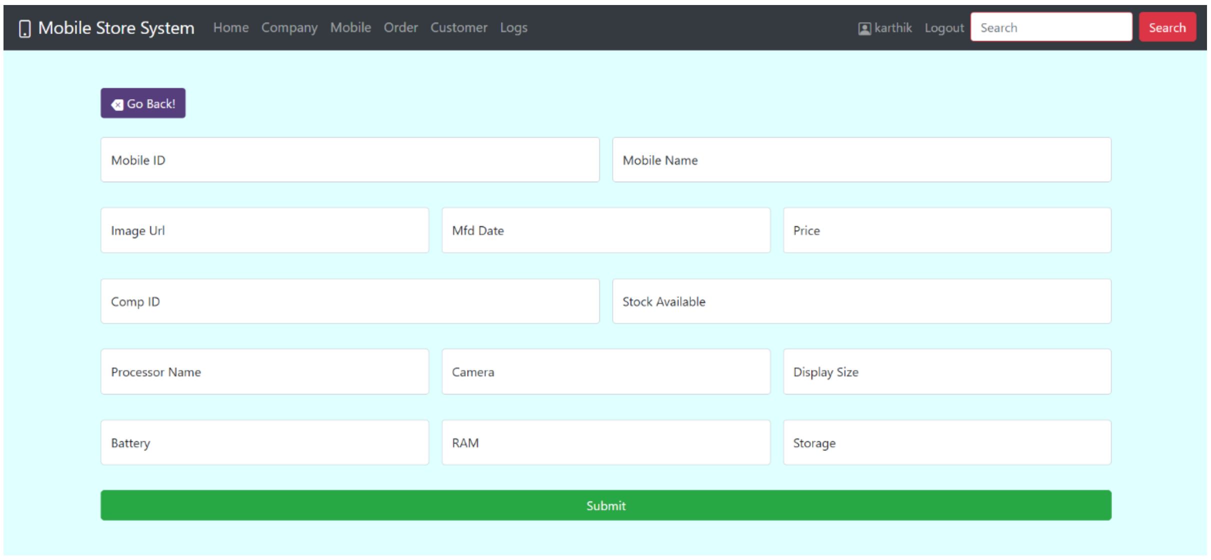
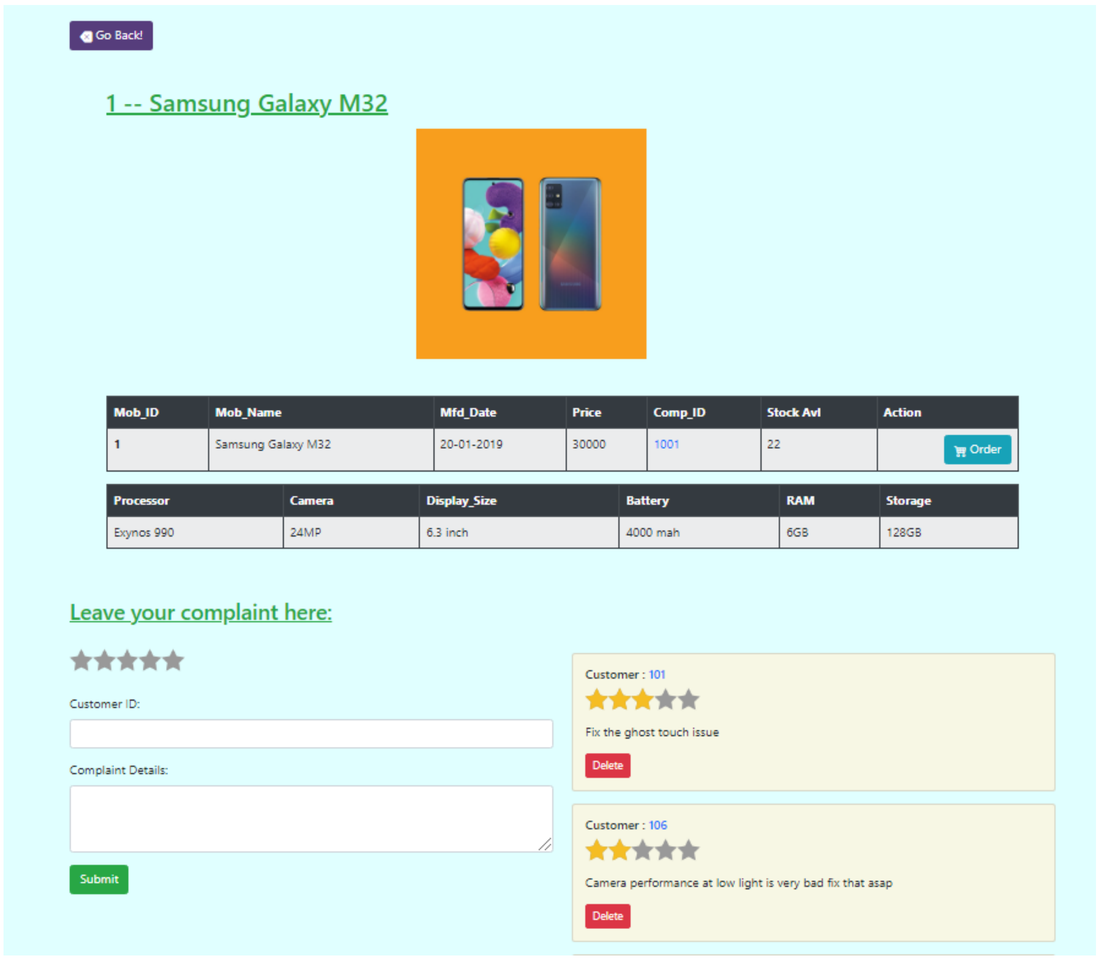


Figure 4.6 form to add mobile

This page shown in the above Figure 4.6 is used to add a new mobile to a store. The store employee after logging in can add mobiles to his store.



Mob_ID	Mob_Name	Mfd_Date	Price	Comp_ID	Stock Avl	Action
1	Samsung Galaxy M32	20-01-2019	30000	1001	22	Order

Processor	Camera	Display_Size	Battery	RAM	Storage
Exynos 990	24MP	6.3 inch	4000 mah	6GB	128GB

Customer : 101

Fix the ghost touch issue

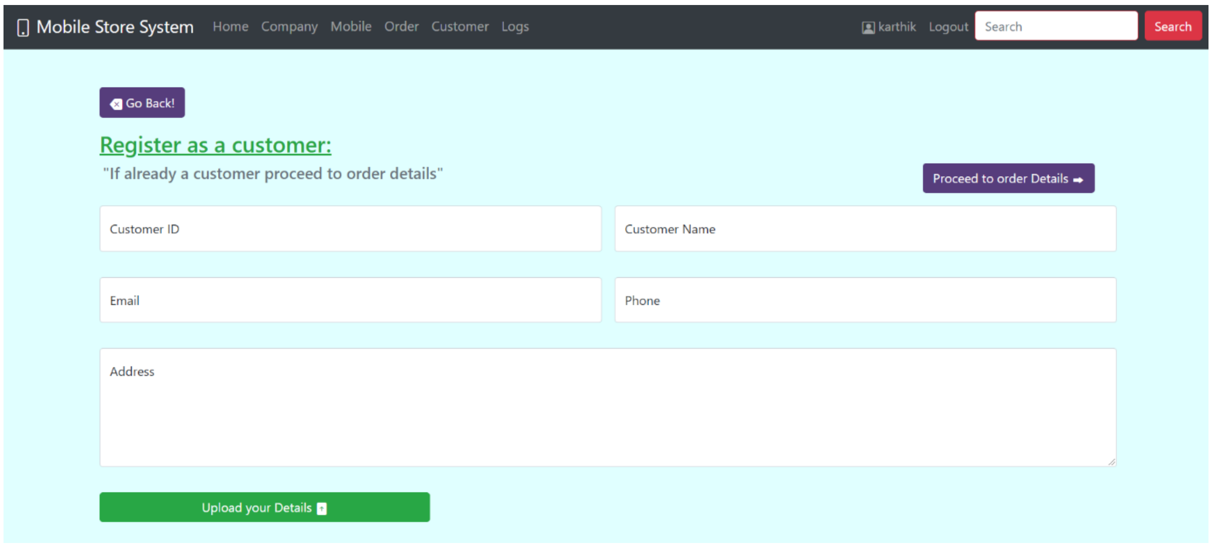
Customer : 106

Camera performance at low light is very bad fix that asap

Figure 4.7 view mobile

The above Figure 4.7 shows the details of a particular mobile .

The employee of the store can take complaints from the customer using the form below the mobile details and can delete complaints as well.



Register as a customer:

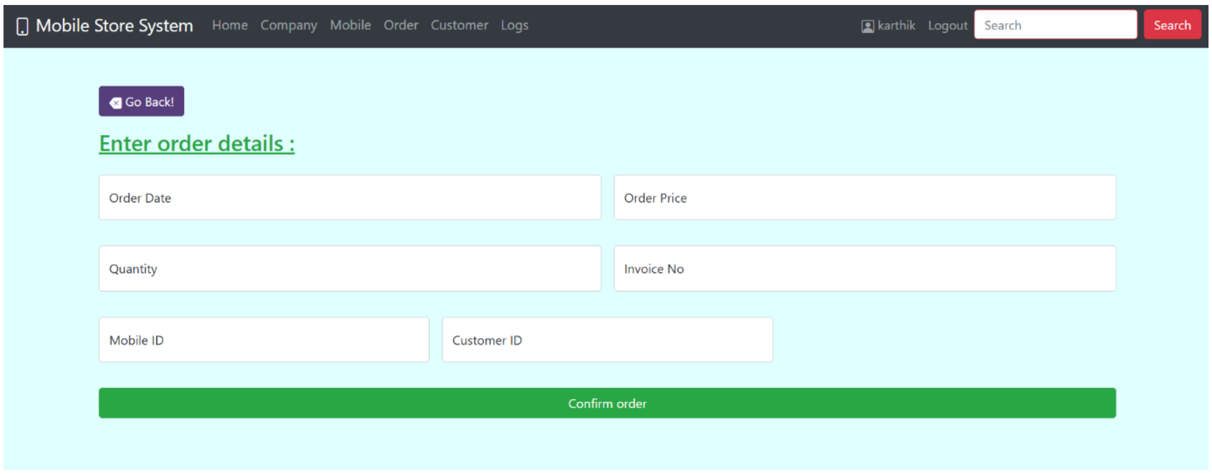
"If already a customer proceed to order details"

Customer ID	Customer Name
Email	Phone
Address	

Figure 4.8 customer form

The above figure 4.8 shows the customer form where employee can register the customer details of the store .

If the customer is a existing customer he can proceed to the order details.

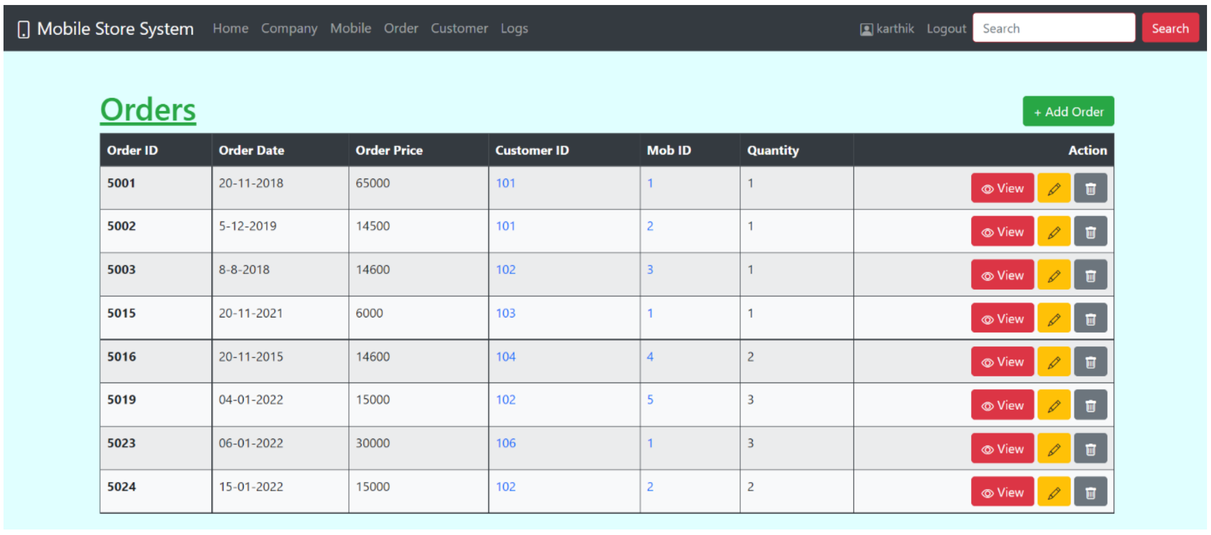


Enter order details :

Order Date	Order Price
Quantity	Invoice No
Mobile ID	Customer ID

Figure 4.9 order form

The above figure 4.9 shows the order form where the store employee can take the order details from the customer like which mobile , the quantity of the order and the customers ID.



Orders

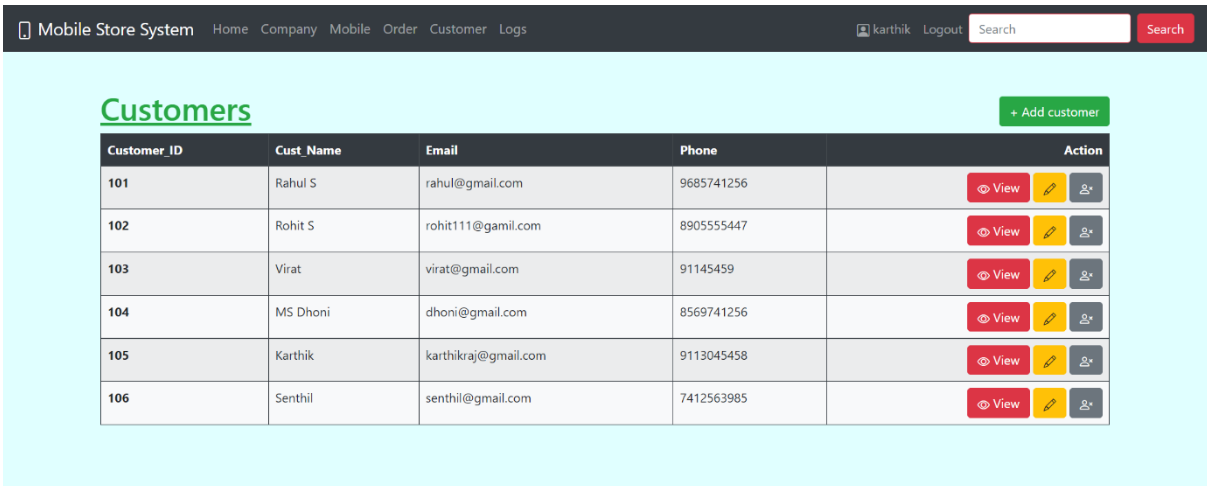
Order ID	Order Date	Order Price	Customer ID	Mob ID	Quantity	Action
5001	20-11-2018	65000	101	1	1	View
5002	5-12-2019	14500	101	2	1	View
5003	8-8-2018	14600	102	3	1	View
5015	20-11-2021	6000	103	1	1	View
5016	20-11-2015	14600	104	4	2	View
5019	04-01-2022	15000	102	5	3	View
5023	06-01-2022	30000	106	1	3	View
5024	15-01-2022	15000	102	2	2	View

Figure 4.10 order lists

The above Figure 4.10 shows the list of orders for mobiles in the store .It displays the order details like order date , price , quantity etc…

By clicking on the view button we can get the invoice of the particular order.

The employee can edit the order details or even delete the order as well.



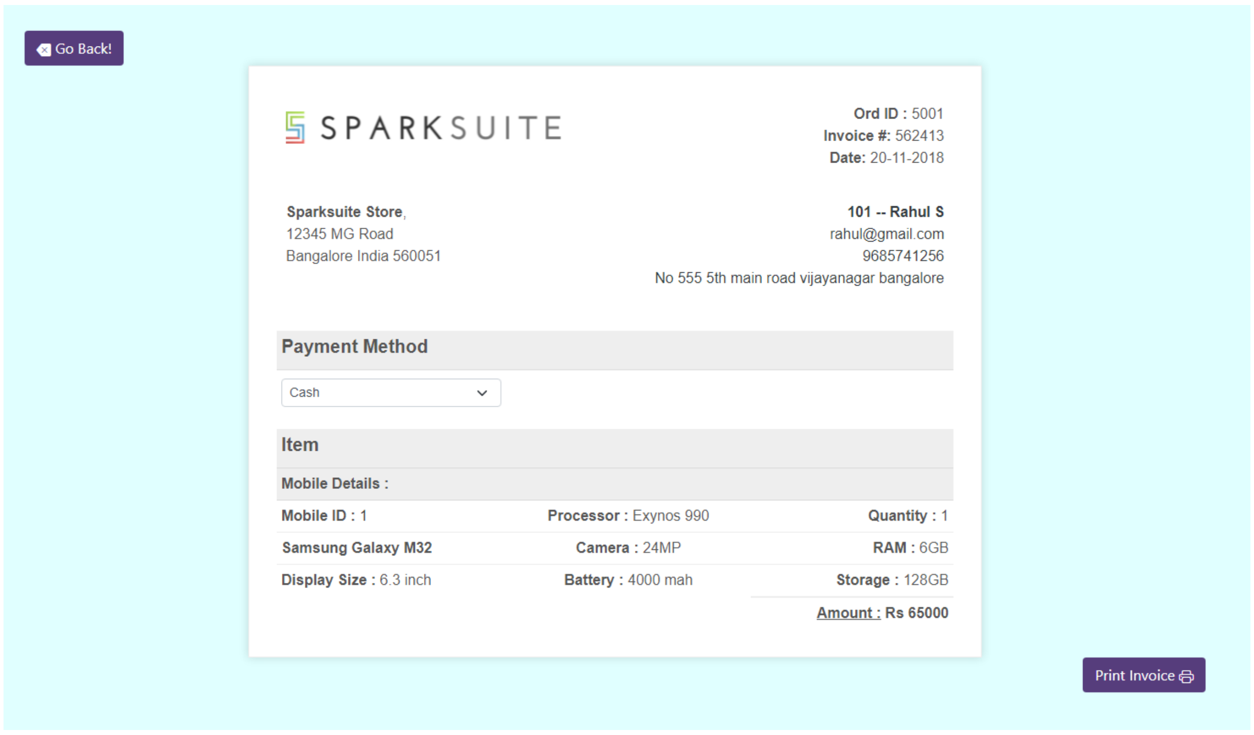
Customers

Customer_ID	Cust_Name	Email	Phone	Action
101	Rahul S	rahul@gmail.com	9685741256	View
102	Rohit S	rohit111@gamil.com	8905555447	View
103	Virat	virat@gmail.com	91145459	View
104	MS Dhoni	dhoni@gmail.com	8569741256	View
105	Karthik	karthikraj@gmail.com	9113045458	View
106	Senthil	senthil@gmail.com	7412563985	View

Figure 4.11 Customer list

The above Figure 4.11 shows the webpage which has the list of customers who has registered in the store .

The store employee can edit the customer details and delete customer as well.



Ord ID : 5001
Invoice #: 562413
Date: 20-11-2018

Sparksuite Store,
12345 MG Road
Bangalore India 560051

101 -- Rahul S
rahul@gmail.com
9685741256
No 555 5th main road vijayanagar bangalore

Payment Method

Cash

Item

Mobile Details :		
Mobile ID : 1	**Processor :** Exynos 990	**Quantity : 1**
Samsung Galaxy M32	**Camera :** 24MP	**RAM :** 6GB
Display Size : 6.3 inch	**Battery :** 4000 mah	**Storage :** 128GB
		Amount : Rs 65000

Figure 4.12 order invoice

Once the payment is made from any of the payment method, the store employee can take the printout of the invoice receipt from the print button.

4.6 Application of project

These types of Mobile Store Systems are examples of technology has had on the world. These help store to maintain the mobiles in the store, the orders of the mobiles , the customer who have bought mobiles in that store and from which companies the store gets the mobiles. They are easy to use and get the job done in very few steps. The store can have database of all the orders for mobile so that store can approach the company and address the demands of a mobile. Also it takes the complaints from the customers for a mobile so that store can address the company to fix those issues.

Major applications of Mobile Store Systems are:

1. All the mobiles and company details are maintained in the database.
2. Easy access to search mobile and order or view its details .
3. A database of all the orders are created for use if needed later.
4. The logs of updation, insertion of mobile details are maintained and invoice receipt for a order can be taken.

Chapter 5

CONCLUSION AND FUTURE ENHANCEMENT

5.1 Conclusion

This project was an attempt to make the structure and working of an Mobile Store system simpler . This is a working example similar to the real world implementation. In this scenario, all the undertakings of the Mobile Store System was achieved in a constructive manner. Given the right guidance and support, its applications and availability can be enhanced.

5.2 Future Enhancements

1. The project supports mobile store for single store , in the future this can be an addition to make the system to connect the databse to multiple stores.
2. Email and messages could be sent to the customer on successful ordering of the mobile.
3. Hosting it and developing further as e-commerce website.
4. Integrating and making order tracking facility when the website gets published as an e-commerce site.
5. Collecting and analyzing mobile prices and offering offers to the mobiles.

REFERENCES

[1] Raghu Ramakrishnan and Johannes Gehrke, Database Management Systems, McGRAW HILL, 3rd Edition.

[2] Ramez Elmasri and Shamkant B. Navathe, Fundamentals of Database Systems, Pearson, 7th Edition.

[3] https://stackoverflow.com/questions/42824740/how-to-use-a-handlebar-template#:~:text=The%20handlebar%20part%20is%20converted,js).

[4] https://www.npmjs.com/package/mysql

[5] https://www.sitepoint.com/using-node-mysql-javascript-client/

[6] https://docs.mongodb.com/manual/

[7] https://nodejs.org/en/docs/

[8] https://expressjs.com/en/5x/api.html