**VISVESVARAYA TECHNOLOGICAL UNIVERSITY**

**“Jnana Sangama”, Belagavi-560018, Karnataka**

****

**DATABASE MANAGEMENT SYSTEM - 18IS5CDBM**

REPORT

**On**

### “E-commerce”

#### BACHELOR OF ENGINEERING

#### In

**INFORMATION SCIENCE AND ENGINEERING**

**Submitted by:**

**Samriddhi Jain(1DS18IS056)**

#### UNDER THE GUIDANCE OF:

Mrs. Bhavani K

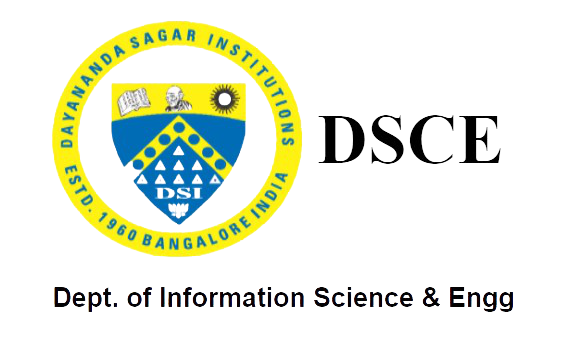
**Department Of Information Science and Engineering**

**DAYANANDA SAGAR COLLEGE OF ENGINEERING**

**SHAVIGE MALLESHWARA HILLS, KUMARASWAMY LAYOUT**

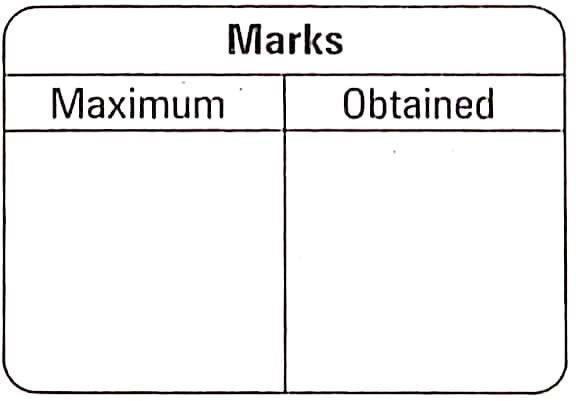
**Bengaluru-560078**

**Department of Information Science & Engineering**

****

**CERTIFICATE**

This is to certify that Management and Entrepreneurship, a report entitled **“ECOMMERCE”** is a bonafide work carried out by **Samriddhi Jain** with **USN 1DS18IS056** in the partial fulfillment for the **5th semester** of Bachelor of Engineering in **Information Science and Engineering** of the Visvesvaraya Technological University, Belagavi during the year 2020-21.



Signature of Faculty in-charge:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

[Mrs. VANI K A]

# **E-COMMERCE**

**ABSTRACT**

Electronic commerce, also known as ecommerce is a type of industry where buying and selling of a product is conducted over electronic systems such as the internet. The purpose of this application is to bring knowledge to students about ecommerce and how an interactive ecommerce application can be designed from scratch using client-side languages, such as JavaScript and HTML, combined with the server-side Java language through Java Server Faces.

The server side, mostly Java, contains all the implementation related to setting up the database, creating session models for joining different user-interface (UI) pages, calculating the shipping costs and sales tax, etc. It is responsible for taking information from the database and making it available to the UI by mapping the category or item ID to the respective IDs stored in the database.

The client side is responsible for showing the entire user interface, containing the CSS, HTML, and JavaScript.

Our website in particular aims to purport merchandise to different fandoms- cult classics like Harry Potter and FRIENDS as well as newer but just as impressive fanbases, such as that of Game of Thrones. Our website displays hoodies, sweaters, T-shirts etc bearing logos, slogans and quotes of the shows whose fans we’re servicing.

That’s why we decided to call our website and project “Fantom”; it’s a play on the word “Fandom”, the customer base that we’re targeting.

**CONTENTS :**

1. **INTRODUCTION**

**BACKGROUND**

**INTRODUCTION TO E-COMMERCE DBMS**

1. **E R DIAGRAM AND RELATIONAL SCHEMA DIAGRAM**

**DESCRIPTION OF E R DIAGRAM**

**DESCRIPTION OF RELATIONAL SCHEMA DIAGRAM**

1. **SYSTEM DESIGN**

**TABLE DESCRIPTION**

1. **IMPLEMENTATION**

**FRONT END DEVELOPMENT**

**BACK END DEVELOPMENT**

**INSERTION OF FEW RECORDS IN THE DATABASE**

**NORMALIZATION**

1. **STORED PROCEDURE**
2. **TRIGGERS**
3. **GENERATING REPORTS**
4. **DISCUSSION OF CODE SEGMENT**

**INTRODUCTION**

**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 modelling techniques.

The database management system (DBMS) is the software that interacts with end users, applications, the database itself to capture and analyse 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.

**INTRODUCTION TO E-COMMERCE DATABASE:**

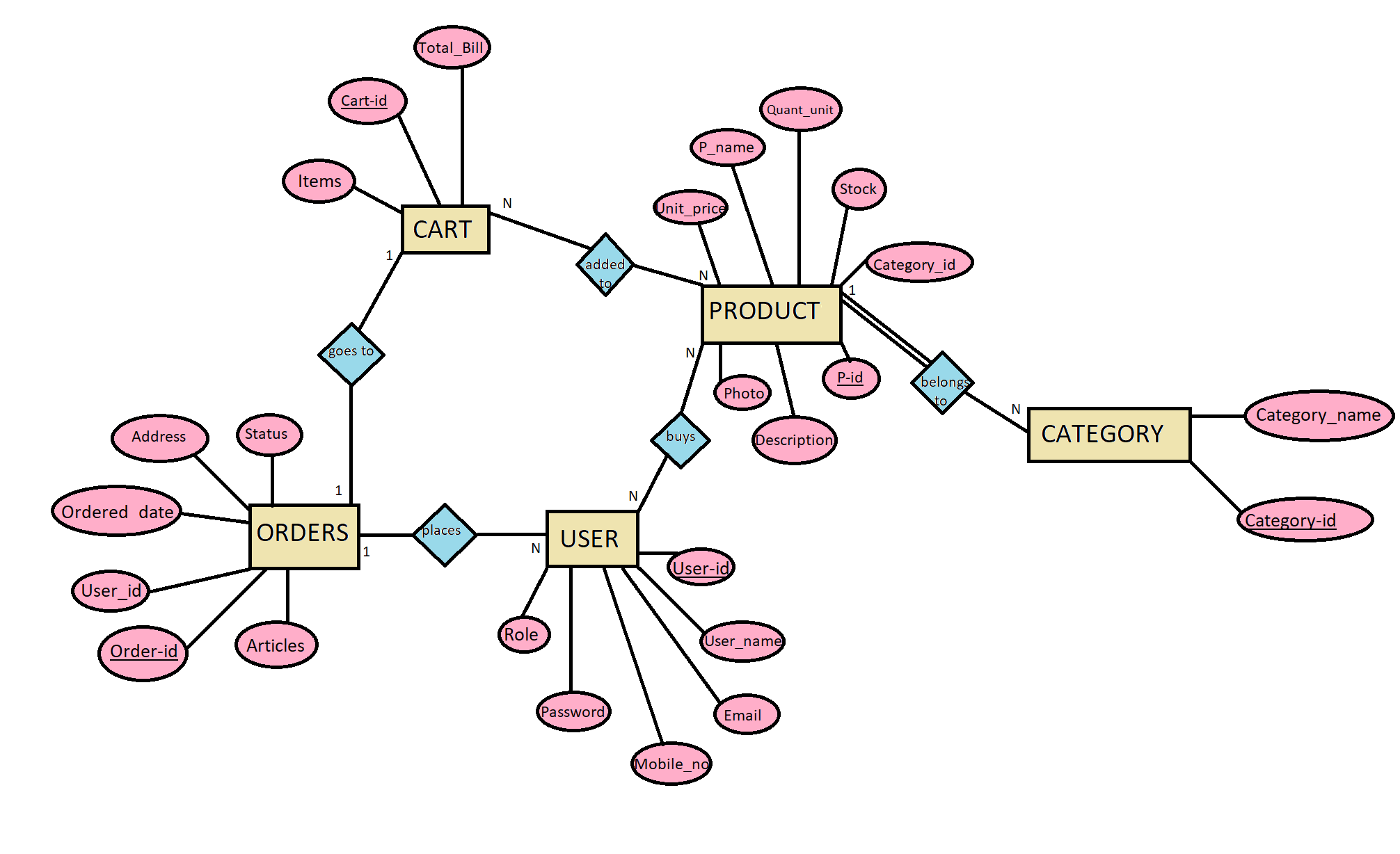
The two interface types found in the online shopping-cart application are as follows:

1. **User Interface**: Users are able to view the home page of the shopping-cart application, browse the different categories, browse and add any number of items from any categories in the shopping cart, look for information about each product, delete the items in the shopping cart, save the cart for later viewing, check out or continue shopping after adding the item to the cart, and check out the items by completing the required information in the order form.

2. **Admin Interface**: The administrator is able to view the users’ information that was entered during checkout in the database, can update the information, price, shipping costs of the items, add or remove items from the main display.

**E R DIAGRAM AND RELATIONAL SCHEMA DIAGRAM**

**DESCRIPTION OF E R DIAGRAM:**



The ER diagram describes the entities, attributes and relationships.

• Entity types like CART, ORDERS,CATEGORY,USER and PRODUCT are in rectangular boxes.

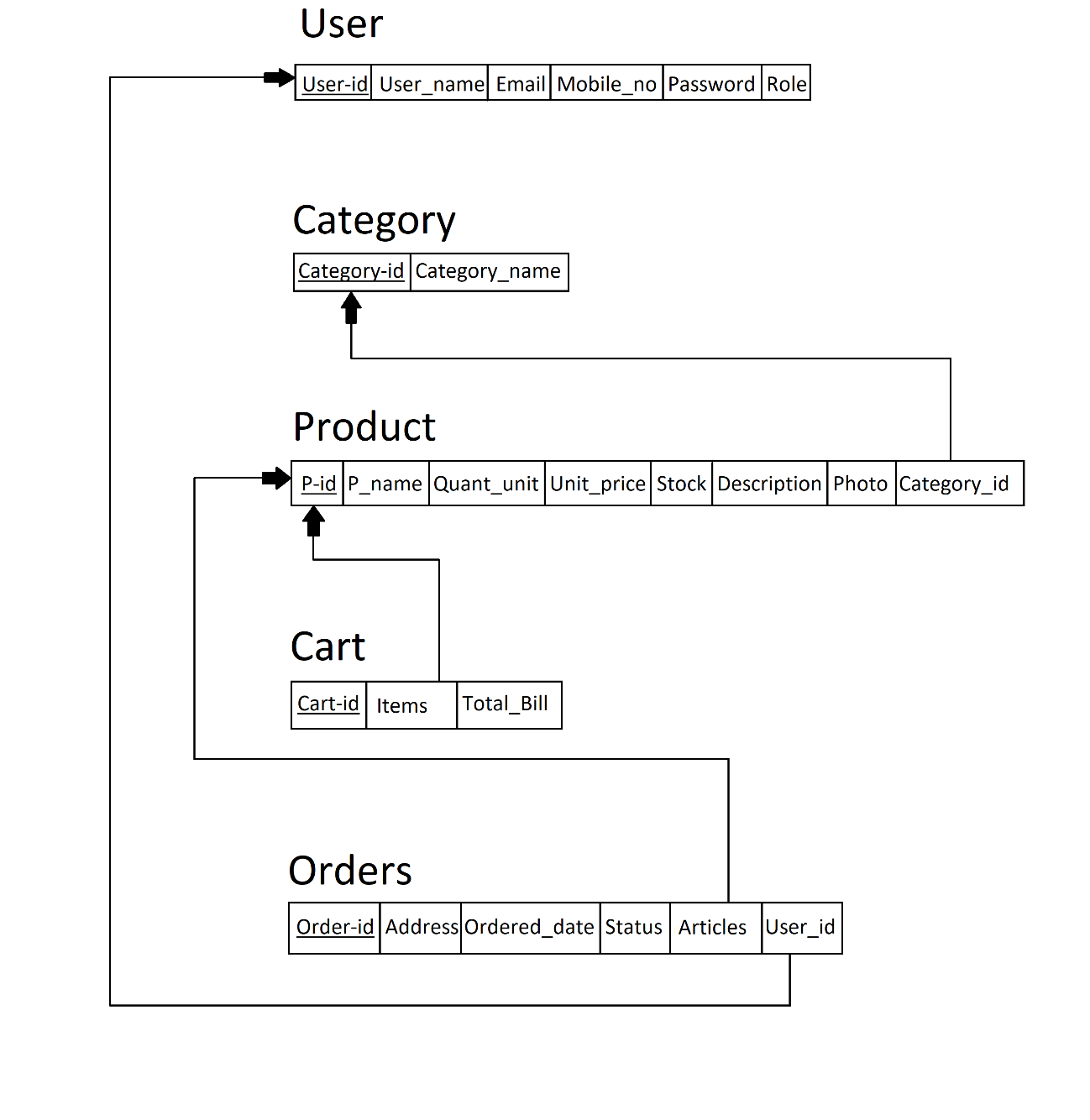
• Relationships like PLACES, GOES TO, ADDED TO, BUYS, BELONGS TO are in diamond boxes, attached to entity types with straight lines.

• Attributes are shown in ovals, each attached by a straight line to the entity or Relationship types.

• Key attributes like USER\_ID,ORDER\_ID are underlined.

• Participation of all entities in relationships are partial apart from that of PRODUCT with the relationship “belongs to”, which is total to indicate that every product *has* to belong to a category in our database. Cardinality is displayed in the E R Diagram. There are no weak entities present.

**DESCRIPTION OF RELATIONAL SCHEMA:**

****

The term database schema refers to the description of database that includes the database structure and various constraints on the database.

The schema diagram is in turn an illustrative display of the database schema.

The primary keys are underlined and the referential integrity constraints are depicted by arrows pointing to the keys they reference.

The above figure shows the relational schema of our e-commerce system. It has the following entities.

1. CART: It is a table that contains details of the cart of the user. It consists of Items, Cart-id, Total\_Bill.
2. ORDERS: This table consists of all details related to a customer’s orders. It consists of Ordered\_date, Address, Status, User\_id, Order-id and Articles.
3. USER: This table consists of details related to the user. It has the following attributes: User-id, User-name, Email, Mobile\_no, Password, Role.
4. PRODUCT: This table consists of details related to the Products. That is P-id, P\_name, Unit\_price, Description, Photo, Stock, Quant\_unit and Category\_id.
5. CATEGORY: This table consists of details related to the Categories. It consists of Category\_name and Category-id.

**GENERAL CONSTRAINTS :**

1. NULL Constraint**:** Attributes that are under NOT NULL constraints have to be filled compulsorily. Almost all the attributes in the project are under NOT NULL constraint.

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:

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: Category\_id in the PRODUCT table refers to Category\_id in CATEGORY table. The various tables are also linked with multiple foreign keys which are all set to cascade any update or delete operation on the attribute in the main table.

Some of the foreign key attributes in our schema are: category\_id in PRODUCTS table and user\_id in ORDERS table.

**SYSTEM DESIGN**

**CATEGORY:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| FIELD | TYPE | NULL? | KEY | DEFAULT |
| Id | Varchar(40) | NOT NULL | Primary Key | None |
| Name | Varchar(45) | NOT NULL |  | None |

The table contains the details of the categories. It consists of Category-id and Category\_name. Here Category-id is the primary key for CATEGORY and foreign key for PRODUCT entities.

**USER:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| FIELD | TYPE | NULL? | KEY | DEFAULT |
| Id | VARCHAR(40) | NOT NULL | PRIMARY KEY | NONE |
| Name | VARCHAR(40) | NOT NULL |  | NONE |
| Email | VARCHAR(40) | NOT NULL |  | NONE |
| Mobile\_no | BIGINT UNSIGNED(10) | NOT NULL |  | NONE |
| Password | VARCHAR(40) | NOT NULL |  | NONE |
| Role | INT(1) | NOT NULL |  | 0 |

The table contains the details of the users. It consists of User-id, User\_name, Email, Mobile\_no, Password and Role. Email and Mobile\_no are declared unique here. User-id is the primary key for USER entity and the foreign key for the ORDERS entity. The ‘Role’ attribute is kept as default 0 to indicate that any new user who signs up is recorded as a user and not an admin.

**ORDERS:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| FIELD | TYPE | NULL? | KEY | DEFAULT |
| Order-id | VARCHAR(40) | NOT NULL | PRIMARY KEY | NONE |
| User\_id | VARCHAR(40) | NOT NULL | FOREIGN KEY | NONE |
| Articles | VARCHAR(40) | NOT NULL | FOREIGN KEY | NONE |
| Address | VARCHAR(450) | NOT NULL |  | NONE |
| Status | VARCHAR(40) | NOT NULL |  | NONE |
| Ordered-date | DATE | NOT NULL |  | NONE |

The table contains the details of the ORDERS. It consists of Order-id, User\_id, Articles, Address, Status and Ordered-date . Here Order-id is the primary key for ORDER entity. User-id and Articles are referenced from USER and PRODUCT respectively.

**PRODUCT:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| FIELD | TYPE | NULL? | KEY | DEFAULT |
| P\_Id | VARCHAR(40) | NOT NULL | PRIMARY KEY | NONE |
| Category\_Id | VARCHAR(40) | NOT NULL | FOREIGN KEY | NONE |
| P\_name | VARCHAR(40) | NOT NULL |  | NONE |
| Unit\_price | INT(40) | NOT NULL |  | NONE |
| Stock | INT(10) | NOT NULL |  | NONE |
| Description | VARCHAR(40) | NOT NULL |  | NONE |
| Photo | VARCHAR(40) | NOT NULL |  | NONE |

The table contains the details of the PRODUCTS. It consists of P\_id, Category\_id, P\_name, Unit\_price, Stock, Description and Photo . Here Product-id is the primary key for PRODUCT entity.

**IMPLEMENTATION**

**FRONT-END DEVELOPMENT**

The front-end is built using a combination of technologies such as React js, 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.

**React Js**

To give you a gentle introduction, React is an open-source JavaScript library used for frontend development, which was developed by Facebook. Its component-based library lets you build high-quality user-interfaces for web apps. This library allows you to place HTML code inside JavaScript and it works with Virtual DOM**.**

**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. Its elements tell the browser how to display the content. Its 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.

**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. SS 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.

**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.

**BACKEND 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.

**Node.js**

Node.js is an open source and cross-platform runtime environment for executing JavaScript code outside of a browser. You need to remember that NodeJS is not a framework and it’s not a programming language. Most of the people are confused and understand it’s a framework or a programming language. We often use Node.js for building back-end services like APIs like Web App or Mobile App. It’s used in production by large companies such as Paypal, Uber, Netflix, WallMart and so on.

**Express.js**

Express is a small framework that sits on top of Node.js’s web server functionality to simplify its APIs and add helpful new features. It makes it easier to organize your application’s functionality with middle ware and routing. It adds helpful utilities to Node.js’s HTTP objects. It facilitates the rendering of dynamic HTTP objects.

|  |  |  |
| --- | --- | --- |
| **Feature** | **Express.js** | **Node.js** |
| Usage | It is used to build web-apps using approaches and principles of Node.js. | It is used to build server-side, input-output, event-driven apps. |
| Level of features | More features than Node.js. | Fewer features. |
| Building Block | It is built on Node.js. | It is built on Google’s V8 engine. |
| Written in | JavaScript | C, C++, JavaScript |
| Framework/Platform | Framework based on Node.js. | Run-time platform or environment designed for server-side execution of JavaScript. |
| Controllers | Controllers are provided. | Controllers are not provided. |
| Routing | Routing is provided. | Routing is not provided. |
| Middleware | Uses middleware for the arrangement of functions systematically server-side. | Doesn’t use such a provision. |
| Coding time | It requires less coding time. | It requires more coding time. |

**Web Server –** APACHE

**WAMP** stands for "Windows, Apache, MySQL, and PHP." WAMP is a variation of LAMP for Windows systems and is often installed as a software bundle (Apache, MySQL, and PHP). It is often used for web development and internal testing, but may also be used to serve live websites.

The most important part of the WAMP package is Apache (or "Apache HTTP Server") which is used run the web server within Windows. By running a local Apache web server on a Windows machine, a web developer can test webpages in a web browser without publishing them live on the Internet. AMP also includes MySQL and PHP, which are two of the most common technologies used for creating dynamic websites. MySQL is a high-speed database, while PHP is a scripting language that can be used to access data from the database. By installing these two components locally, a developer can build and test a dynamic website before publishing it to a public web server.

**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.

**INSERTION OF FEW RECORDS IN THE DATABASE**

**CATEGORY:**

|  |  |
| --- | --- |
| CATEGORY\_ID | CATEGORY\_NAME |
| 100 | Game of Thrones |
| 200 | Harry Potter |
| 300 | F.R.I.E.N.D.S |

**USER:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| USER\_ID | USER\_NAME | EMAIL | PASSWORD | MOBILE\_NO | ROLE |
| 1000 | Sanjana S | sanj.xp@gmail.com | punnypuns%45 | 9900551232 | 1 |
| 1001 | Samriddhi Jain | sambam@gmail.com | coldplayparadise1 | 9900149624 | 1 |
| 1002 | Chaitanya Bhat | chaitanya.bhat@gmail.com | $farmersrock$ | 9990134567 | 1 |
| 1003 | Anagha H | raoanagha@gmail.com | budisthebest04 | 9990045671 | 1 |

**ORDERS:**

|  |  |  |  |
| --- | --- | --- | --- |
| ORDER\_ID | ADDRESS | ORDERED\_DATE | USER\_ID |
| 6000 | #67 Gruha,JP Nagar 5th Phase,Bangalore | 2020-12-31 | 1000 |
| 6001 | 512 S Block,Trinity Apartments,7th Main,Rajajinagar,Bangalore | 2020-12-25 | 1001 |
| 6002 | No.32 Nialya,KP Road,Malleshwaram,Bangalore | 2020-12-25 | 1002 |
| 6003 | #26 Rukmini Apartments,5th Main,Jayanagar,Bangalore | 2021-01-02 | 1003 |

**PRODUCT:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| P-id | P\_name | Unit\_price | Quant\_unit | Stock | Description | Photo | Category\_id |
| 101 | GOT T-shirt | 350 | 7 | 10 | A black ,comfortable cotton tee that displays the Iron Throne, the symbol of GoT fans everywhere | https://pin.it/7GI3Q4O | 100 |
| 102 | GOT Sweatshirt | 500 | 4 | 10 | A black and grey sweatshirt with the iconic Stark sigil and motto | https://pin.it/24vquCg | 100 |
| 103 | Harry Potter Sweater | 435 | 5 | 10 | A red, cozy sweater that sports glasses and a thunderbolt scar | https://pin.it/2br2z37 | 200 |

**NORMALIZATION**

**Third normal form** (**3NF**) is a normal form that is used in normalizing a database design to reduce the duplication of data and ensure referential integrity by ensuring that:

1. The entity is in second normal form.

2. No non-prime (non-key) attribute is transitively dependent on any key i.e. no non-prime attribute depends on other non-prime attributes. All the non-prime attributes must depend only on the candidate keys.

**STORED PROCEDURES**

A stored procedure is a set of Structured Query Language (SQL) statements with an assigned name, which are stored in a relational database management system as a group, so it can be reused and shared by multiple programs.

The following is the code for a stored procedure to show products by category:

DELIMITER //

create procedure ProductByid

(in id varchar (20))

begin

SELECT \* FROM product p natural join category c WHERE p.p\_id=id;

end//

DELIMITER ;

select \* from product;

call productByid('2eh9b0Ul3B');

The applications table has many entries and needs to be updated on a regular basis. All declined applications and accepted applications can be dropped from the database after a period of time. To do this automatically, we add a stored procedure.

**TRIGGERS**

A database trigger is procedural code that is automatically executed in response to certain events on a particular table or view in a database.

1.Trigger for updating quant\_unit whenever a user places an order:

delimiter $$

create if not existstrigger insert\_on\_order

after insert

on orders for each row

begin

declare productList json ;

declare i int default 0 ;

set @productList = (new.products) ;

set @i =0;

while @i<json\_length(@productList) do

update product

set quant\_unit = quant\_unit + json\_extract(@productList, concat('$[',@i,'].count'))

where p\_id = json\_extract(@productList, concat('$[',@i,'].p\_id')) ;

set @i = @i + 1;

end while ;

end ;

delimiter ;

2.Trigger for tracking of dates on which the orders are placed:

delimiter $$

create trigger set\_placed\_on

before insert on orders

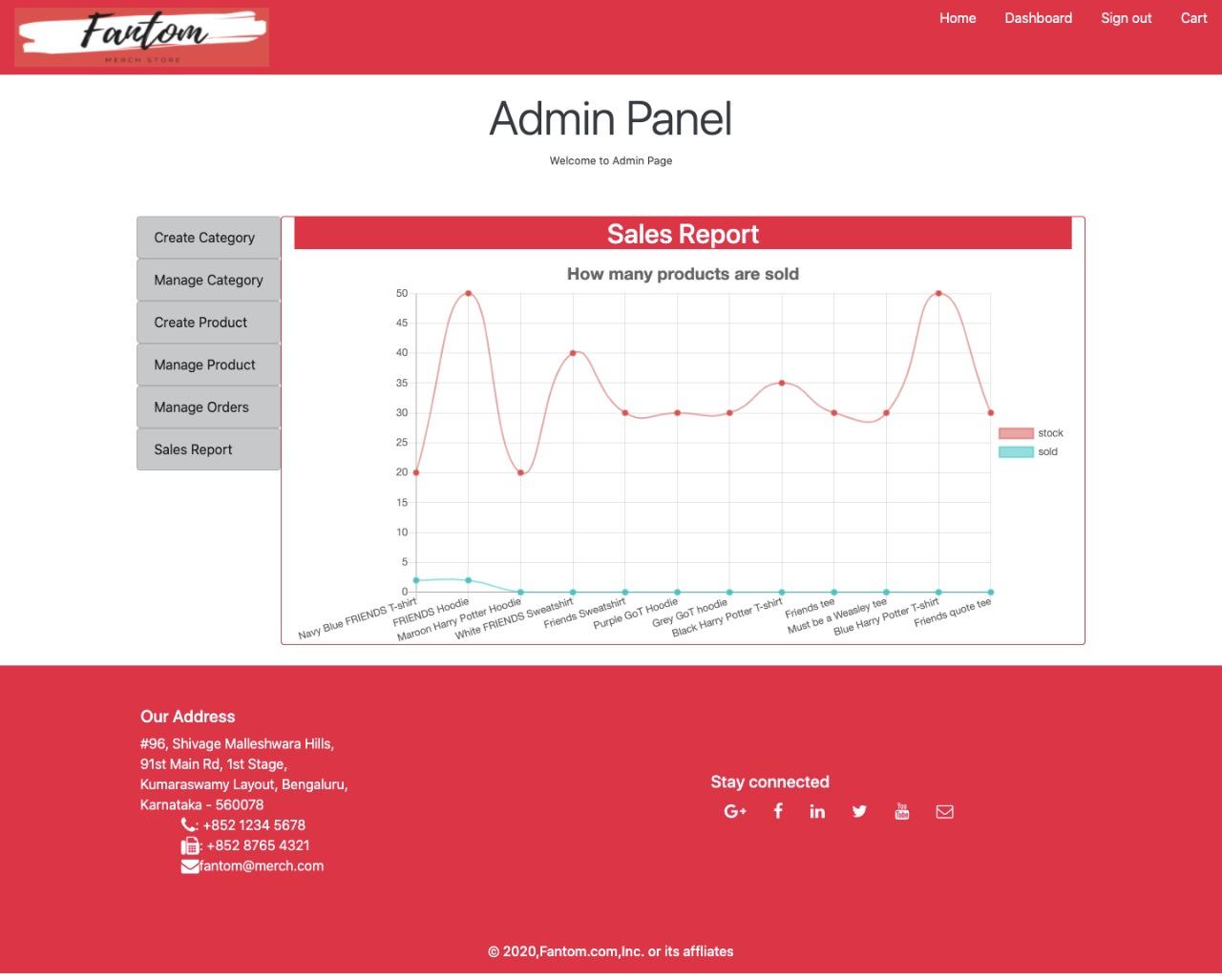
for each row

begin

set new.placed\_on = now();

end ;

**GENERATING REPORTS**

****

This is the report generation which is visible in the admin panel of our website. It is dynamically updated with changes in the database.

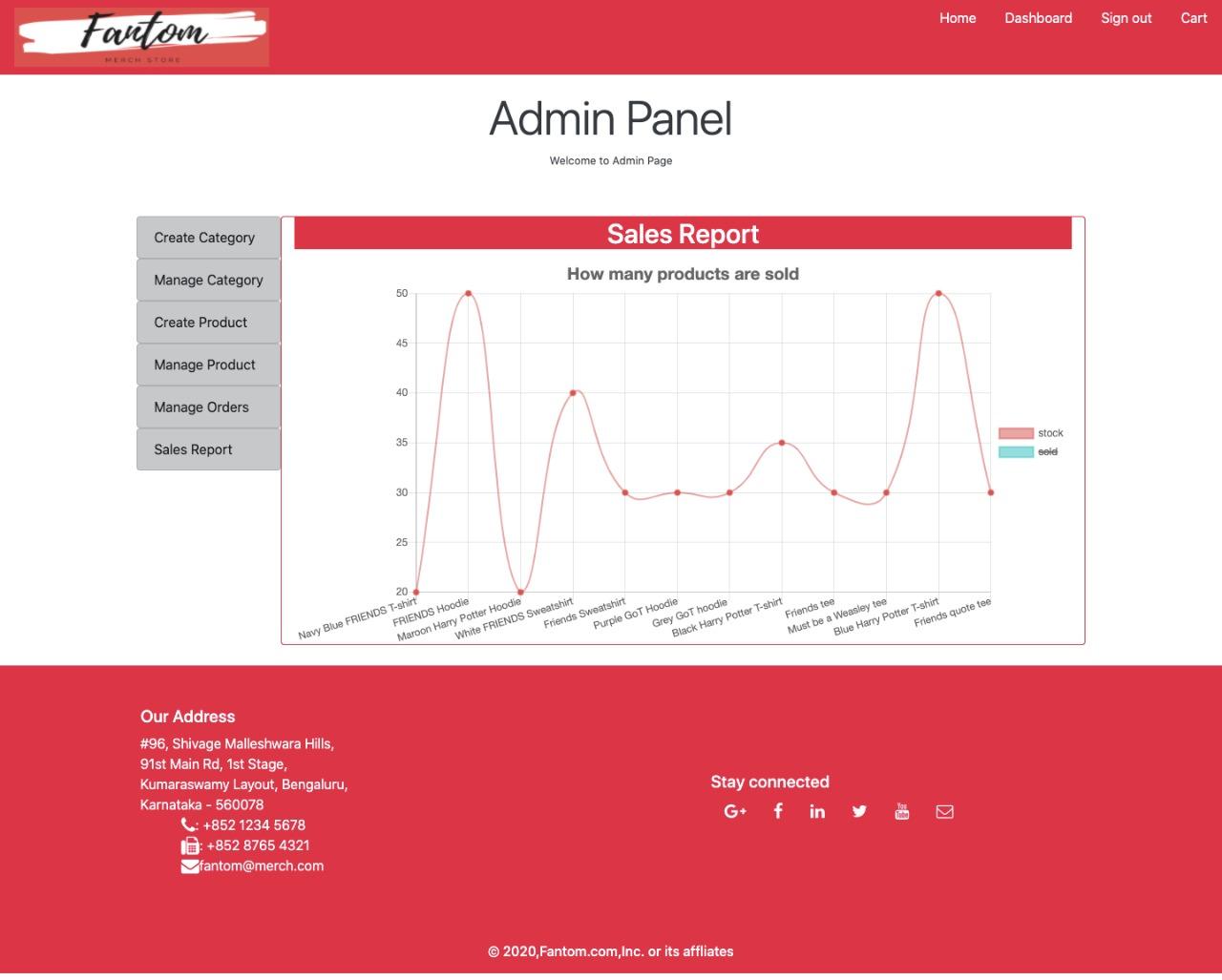
The two differently coloured lines represent the two reports:

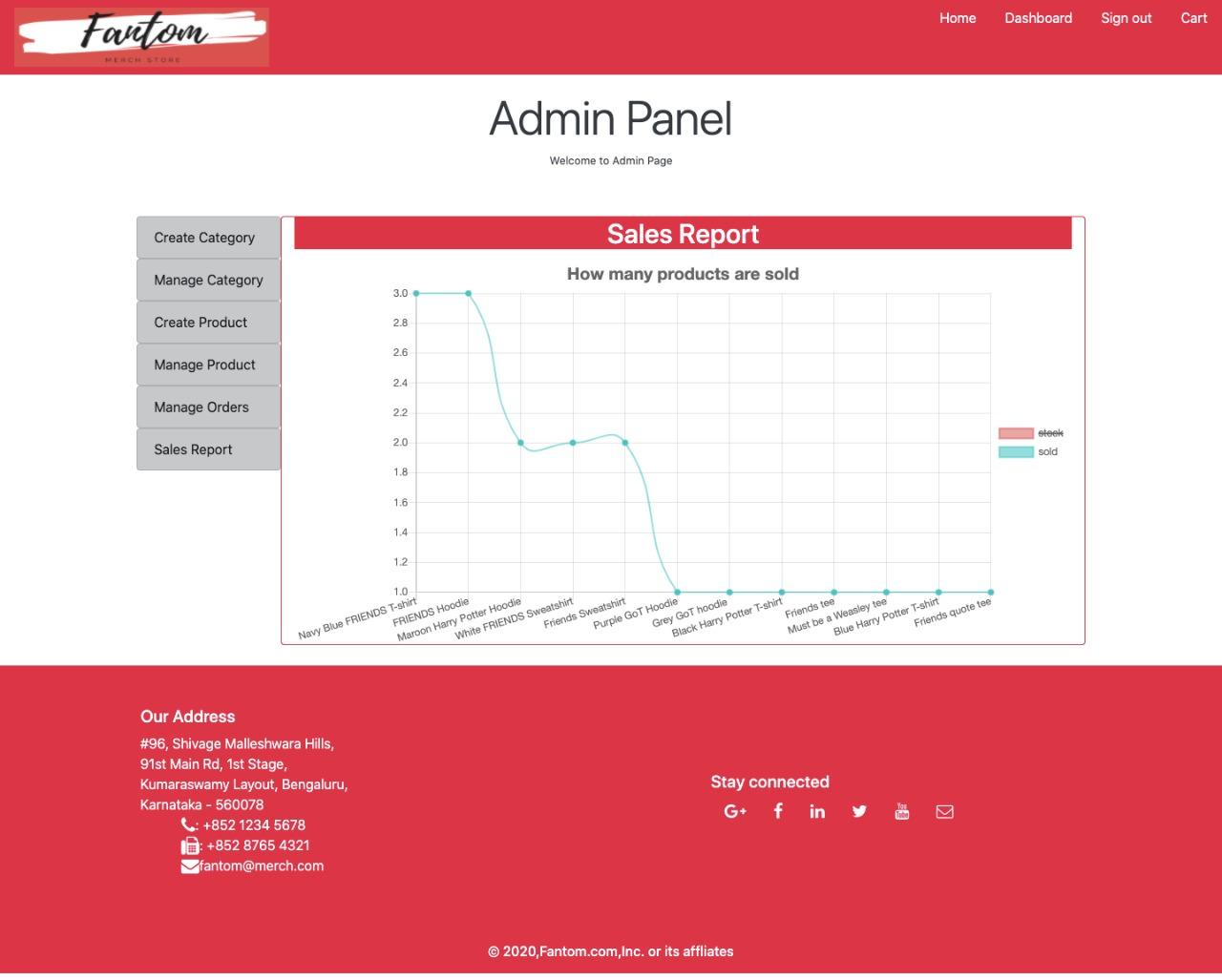
The first report with the graph lines in RED depicts the stock of all the products logged by Fantom at any given point of time, and is dynamically updated.

It is maintained by the stock attribute of the PRODUCT table.

The second report in with the graph lines in BLUE shows how many of the products were sold at that given instant in time.

It is maintained by the Quant-unit attribute of the PRODUCT table.

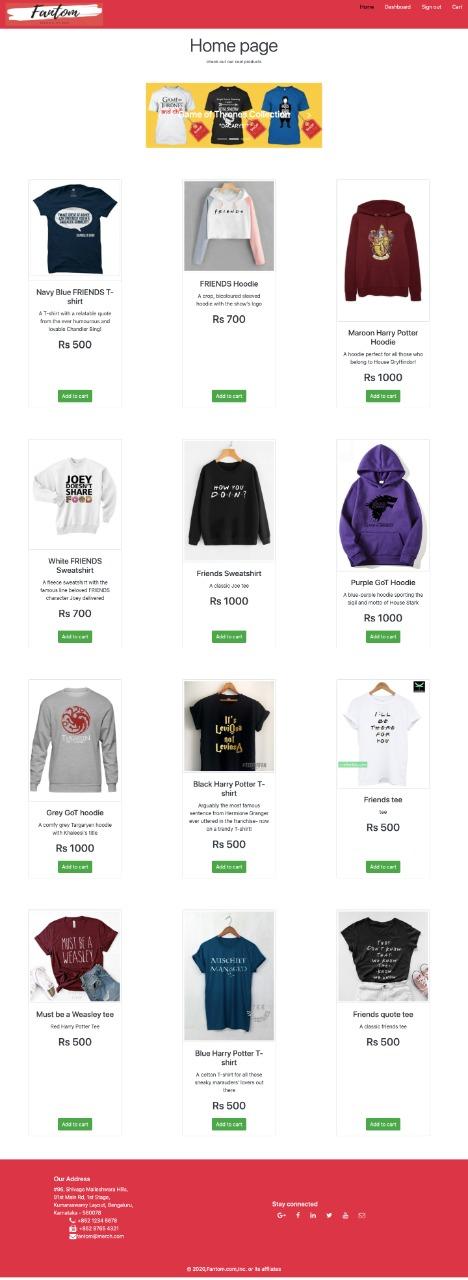
****

****

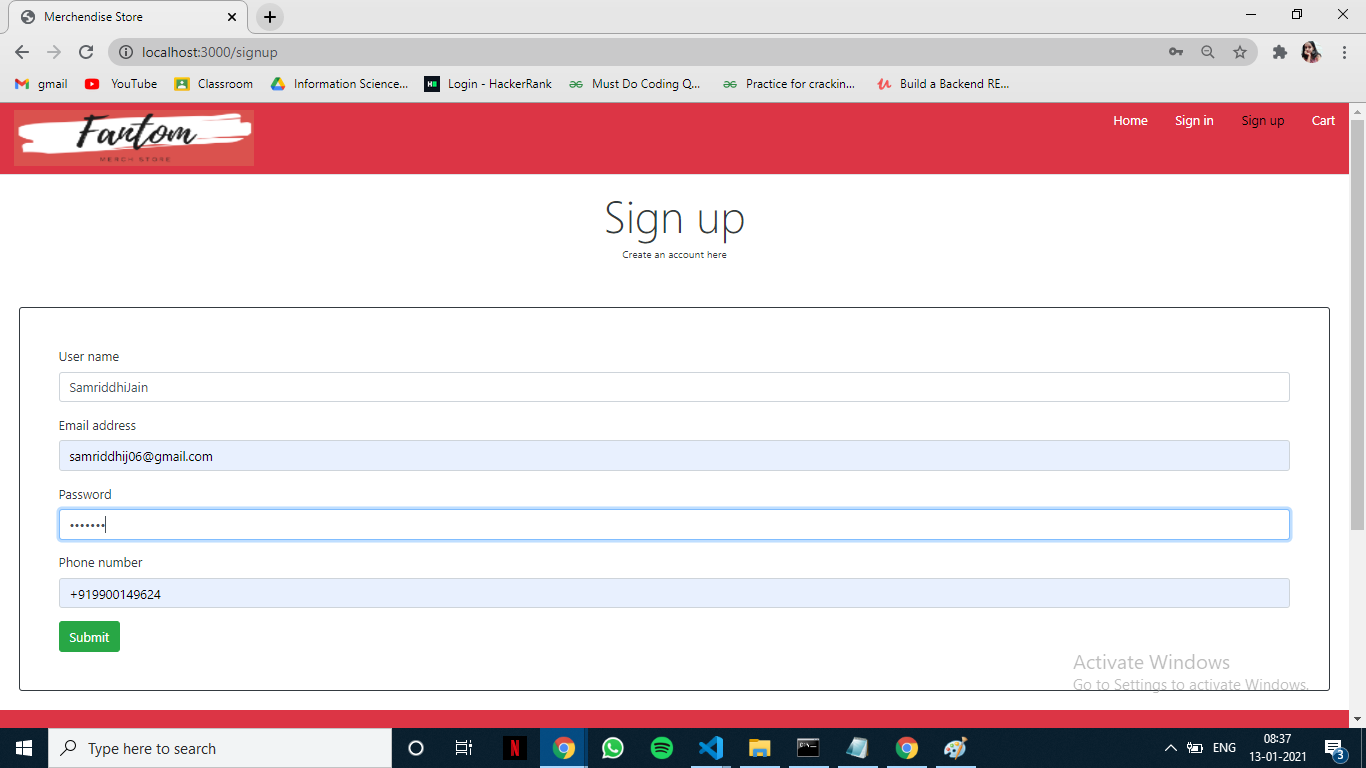
**CODE REVIEW**

**FRONTEND:**

**HOME PAGE OF THE WEBSITE:**



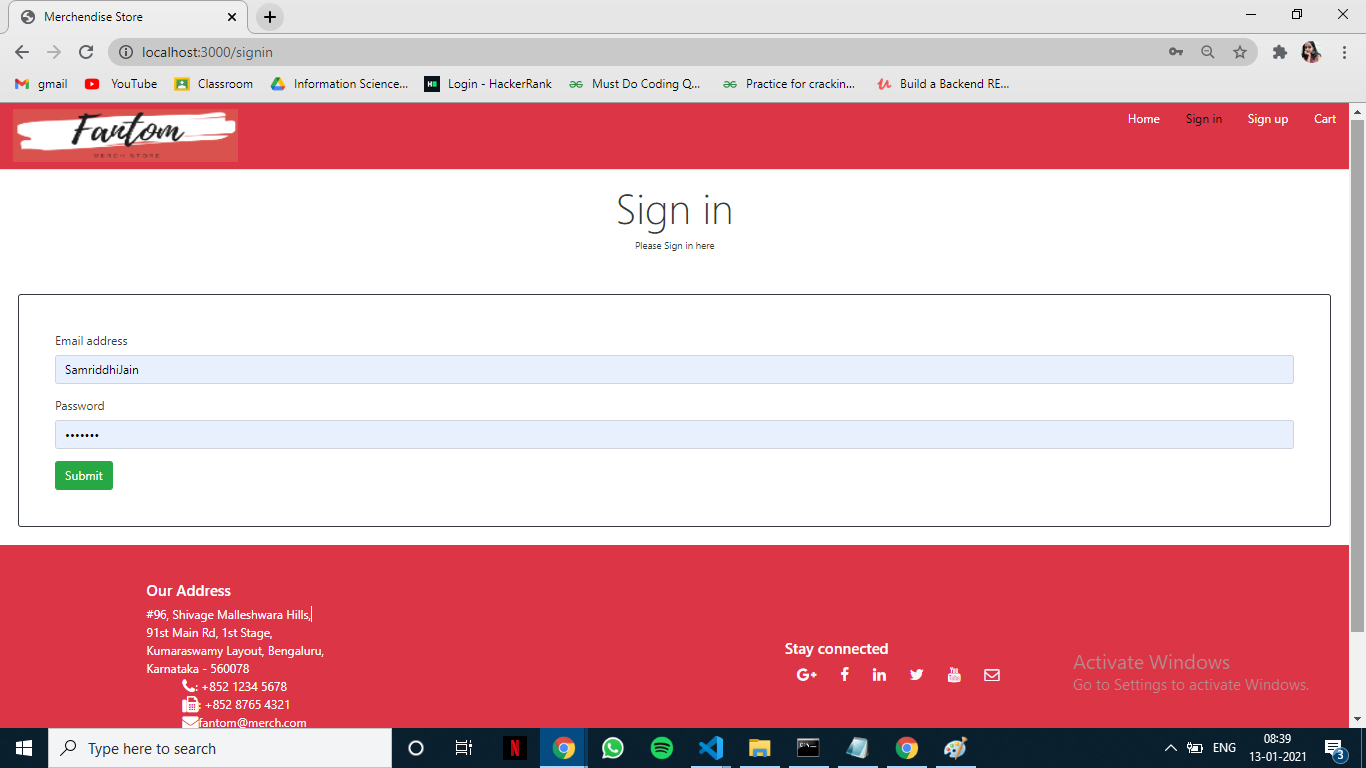
Displays all the products in Fantom’s inventory with their description, price and the option to add that particular product to the cart.

**SIGNUP PAGE FOR USERS:**

Asks for a first-time user to input their name, email-id, password and phone number. Their user-id is automatically generated in the backend and the minimum number of characters for the password is 3 and user\_name is 5.

Since the email and mobile keys are declared as unique, a user with the same ones cannot register twice.

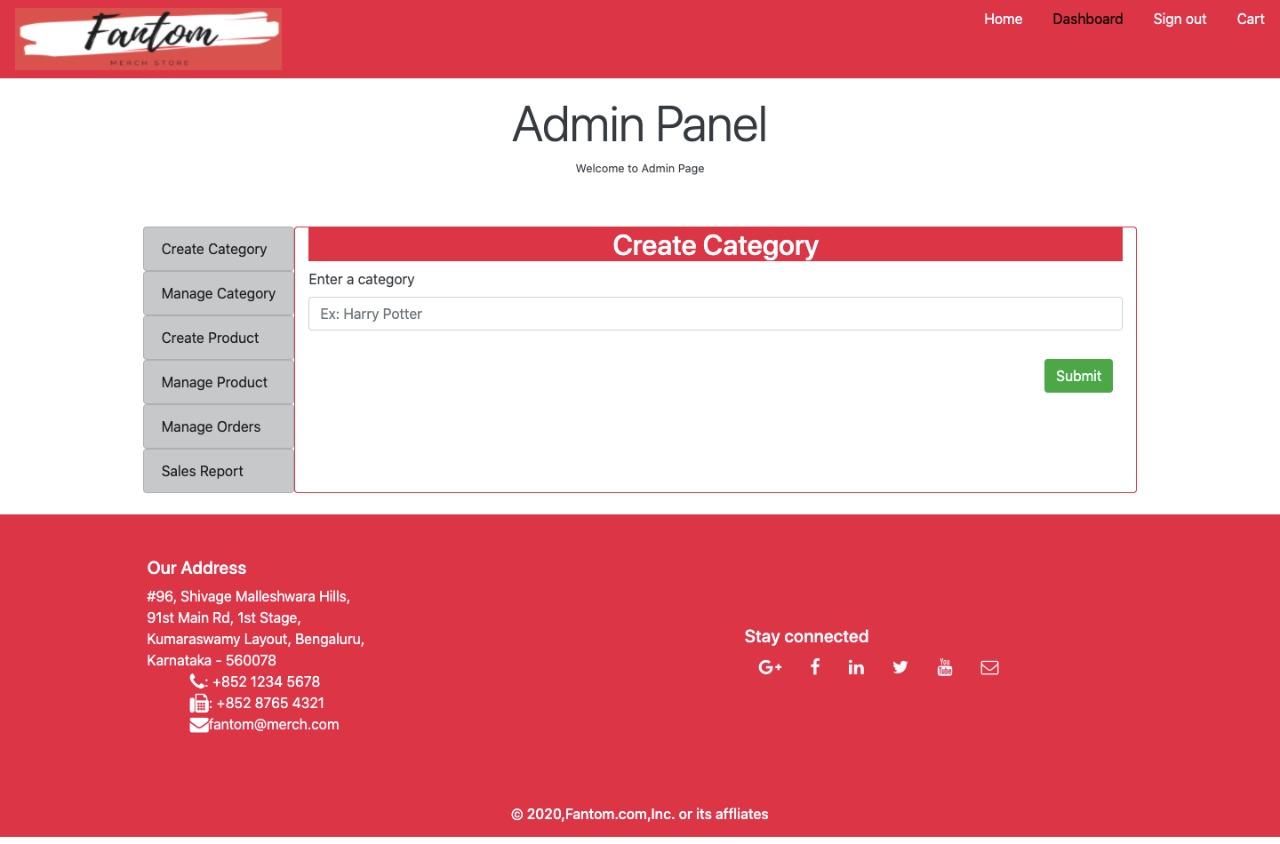
**SIGN IN PAGE FOR USERS:**



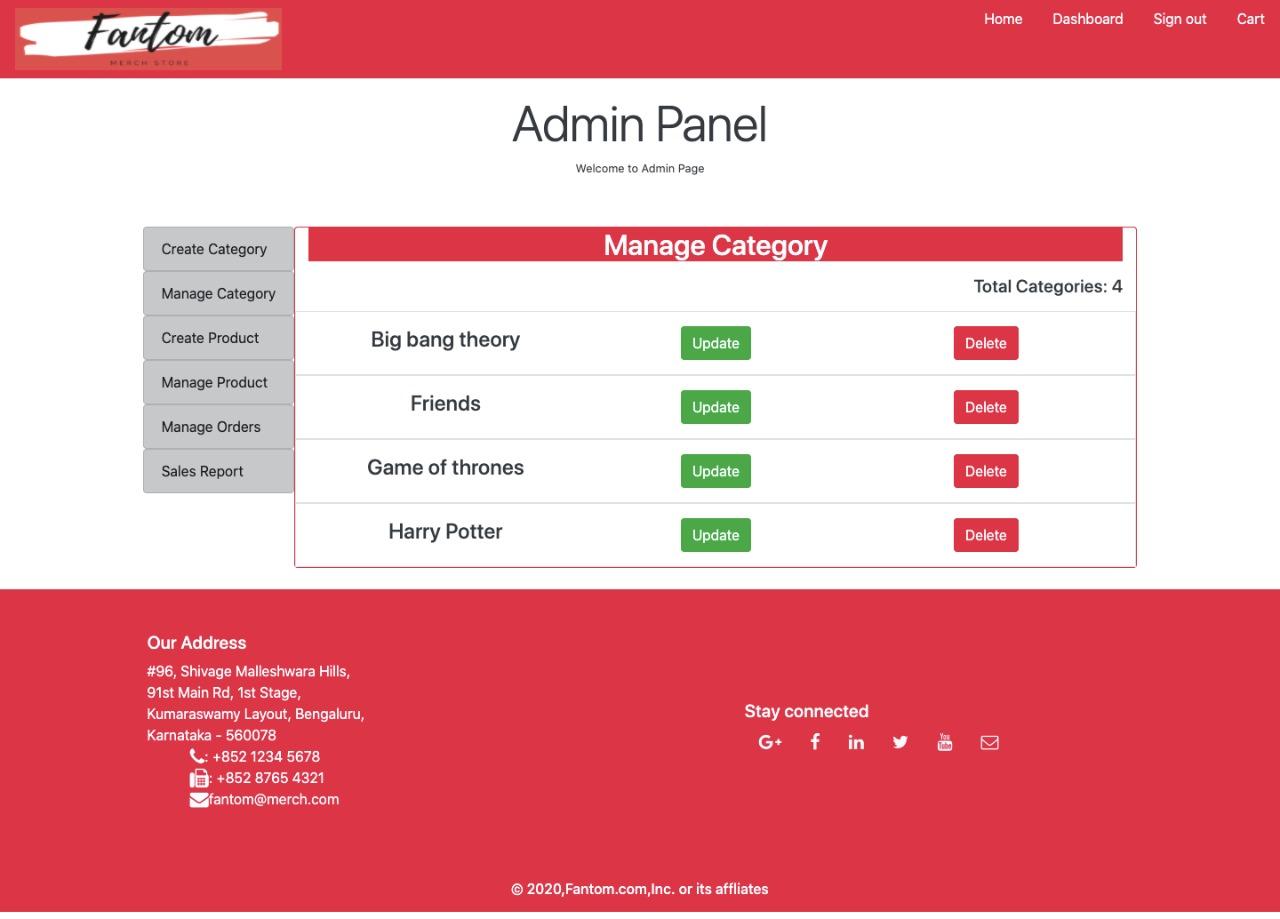
An already registered user can log in to Fantom by entering their email and respective password.

**ADMIN SIDE:**

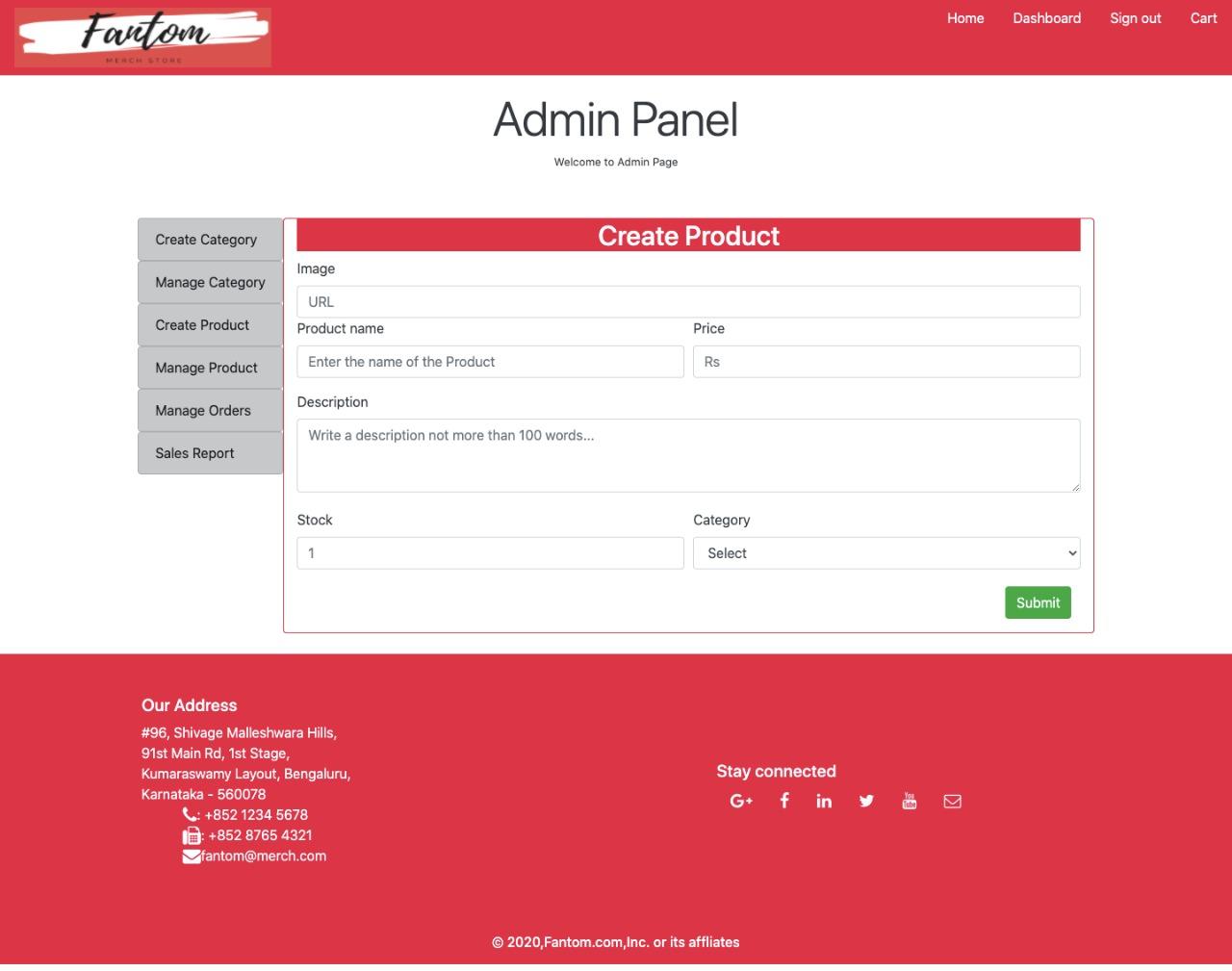
**CREATING CATEGORY:**

****

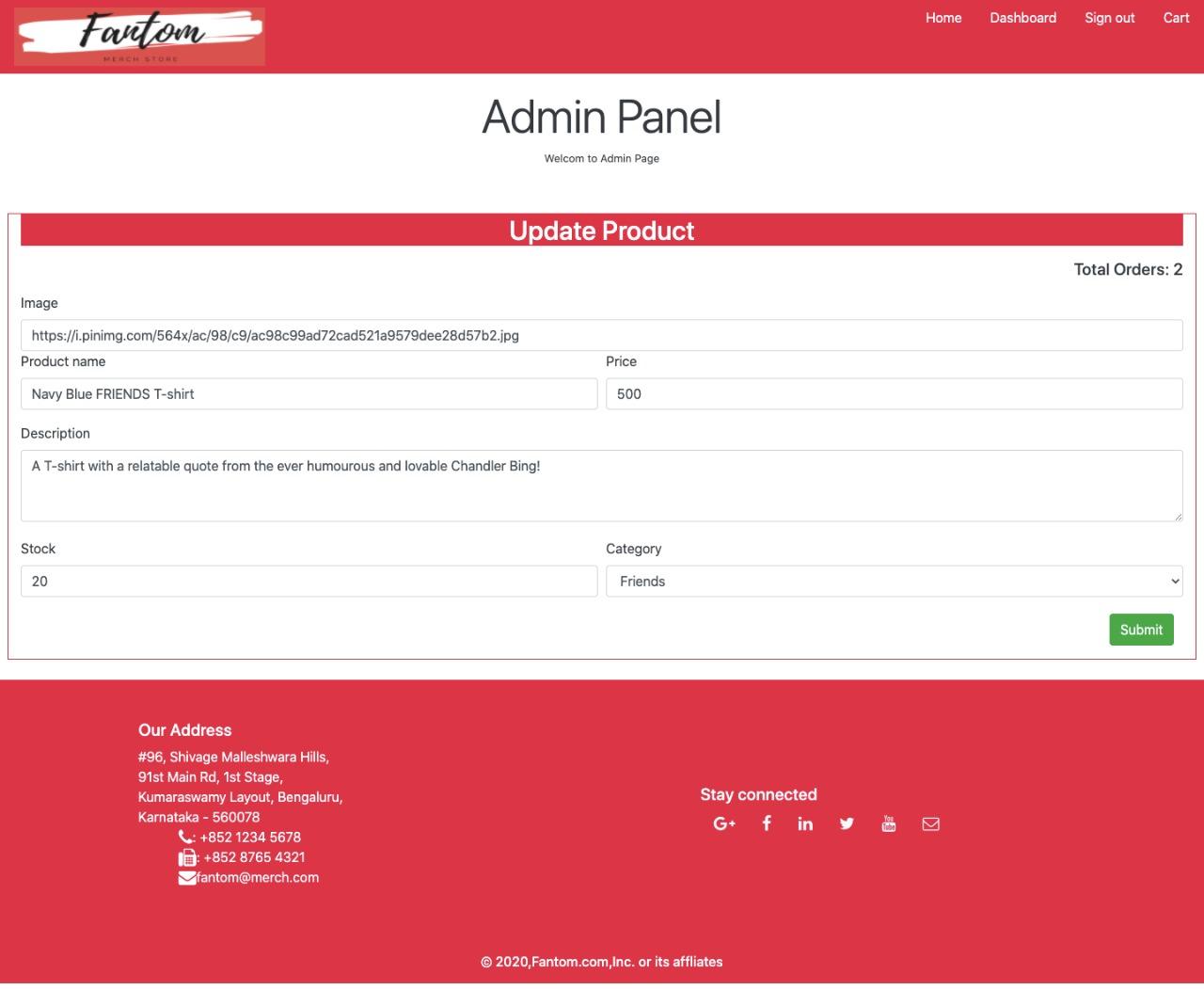
**MANAGING CATEGORIES:**

****

The admin can create, update and delete categories as shown in the above 2 images,

**CREATING PRODUCTS:**

**UPDATE PRODUCTS:**

****

**MANAGING PRODUCTS:**

****

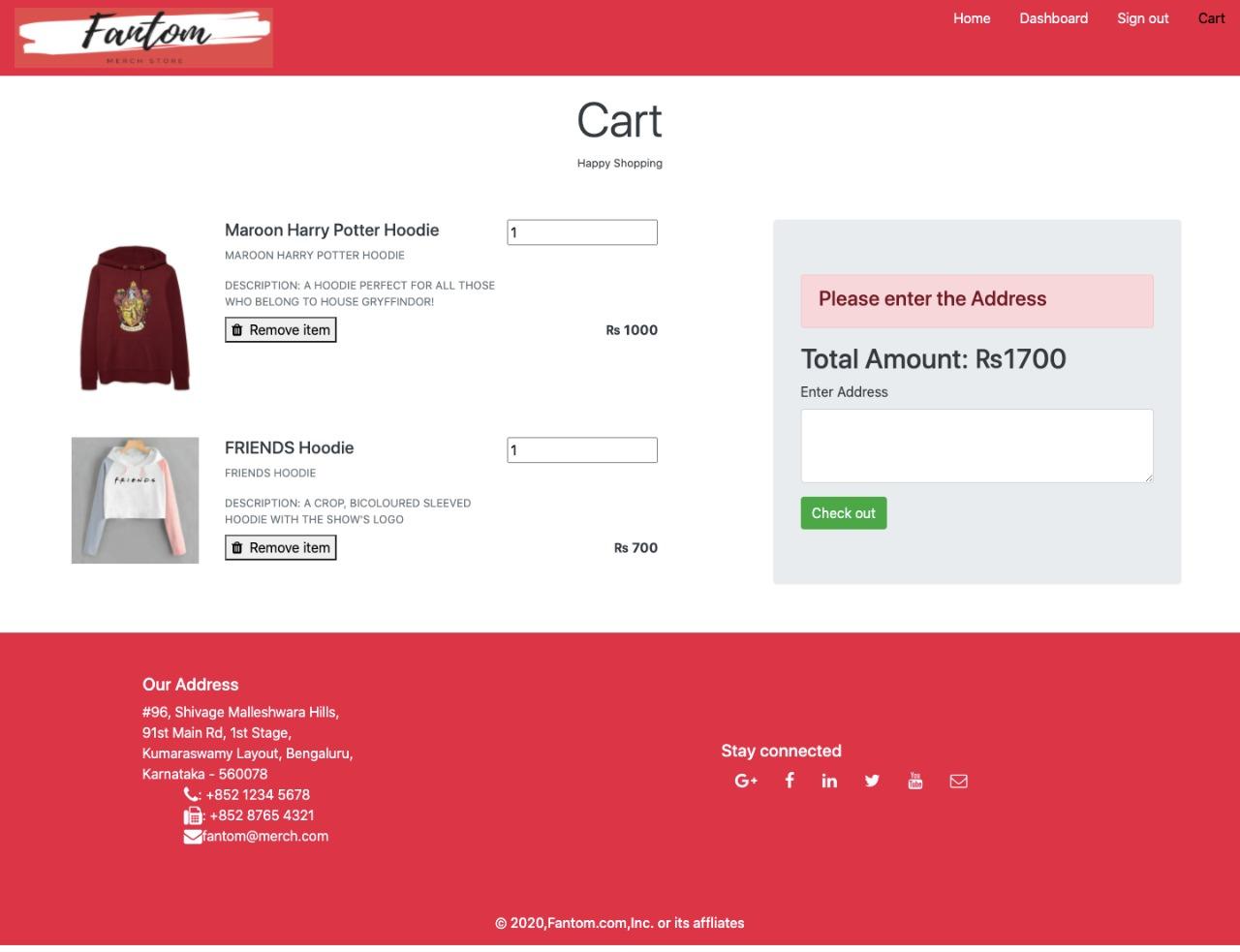
The above three pictures show the different actions an admin can perform with respect to the products in Fantom.

They can create a product by uploading the URL of the product’s image, giving it a name, description, initializing its stock and selecting with fandom category it belongs to.

They can update the same if necessary later on.

Lastly, they can manage the products at large with the option to update or delete them.

**CART:**

****

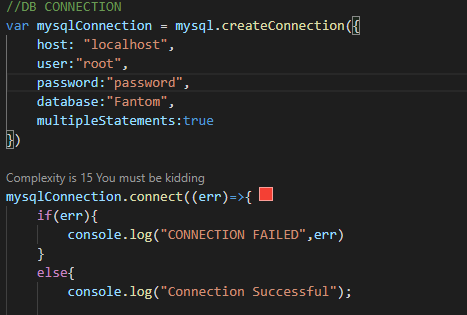
The products that a customer has picked out show up in the cart, where the description and number of that product is displayed, and can be edited for any last minute changes.

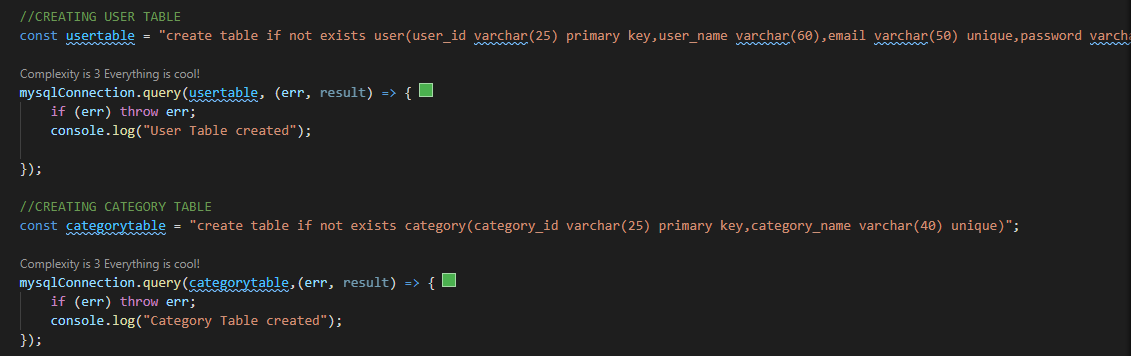
The total amount is visible on the right and below it is a box where the user can enter the required delivery address, then proceed to check out.

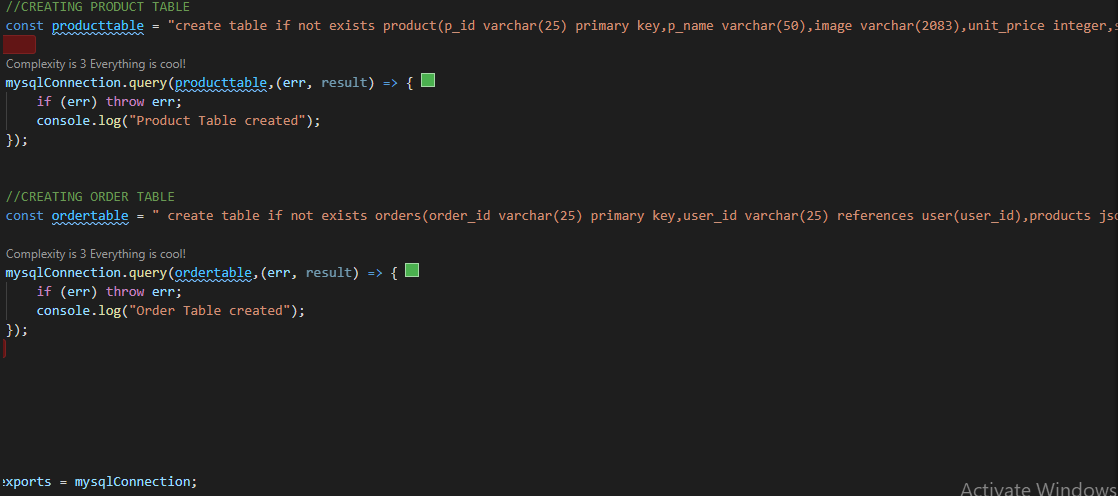
When this is done, the order is confirmed and entered In the database.

**DATABASE CONNECTION:**

This section talks about the backend developed in MYSQL which queries the database. This is the part which connects the frontend HTML code and the backend database.







**BACKEND CONNECTION:**

