# Indian Institute of Technology Gandhinagar


# CS 432: Databases

## ASSIGNMENT-4
### DEPLOYING THE DBMS

*Under the guidance of*
Prof. Mayank Singh

15th April, 2023

***Submitted By: CDSLite***

## 3.1 Responsibility of G1:

### 1. Changes in UI:

**Summary of first feedback:**

- Stakeholder: Shailya Patel (Potential User)
- Some pictures from a survey [LINK]
- “Easy-to-use and intuitive UI with minor bugs”
- Products lack images
- From a user perspective, the UI could be more vibrant
- Lack of navigation ease from the cart

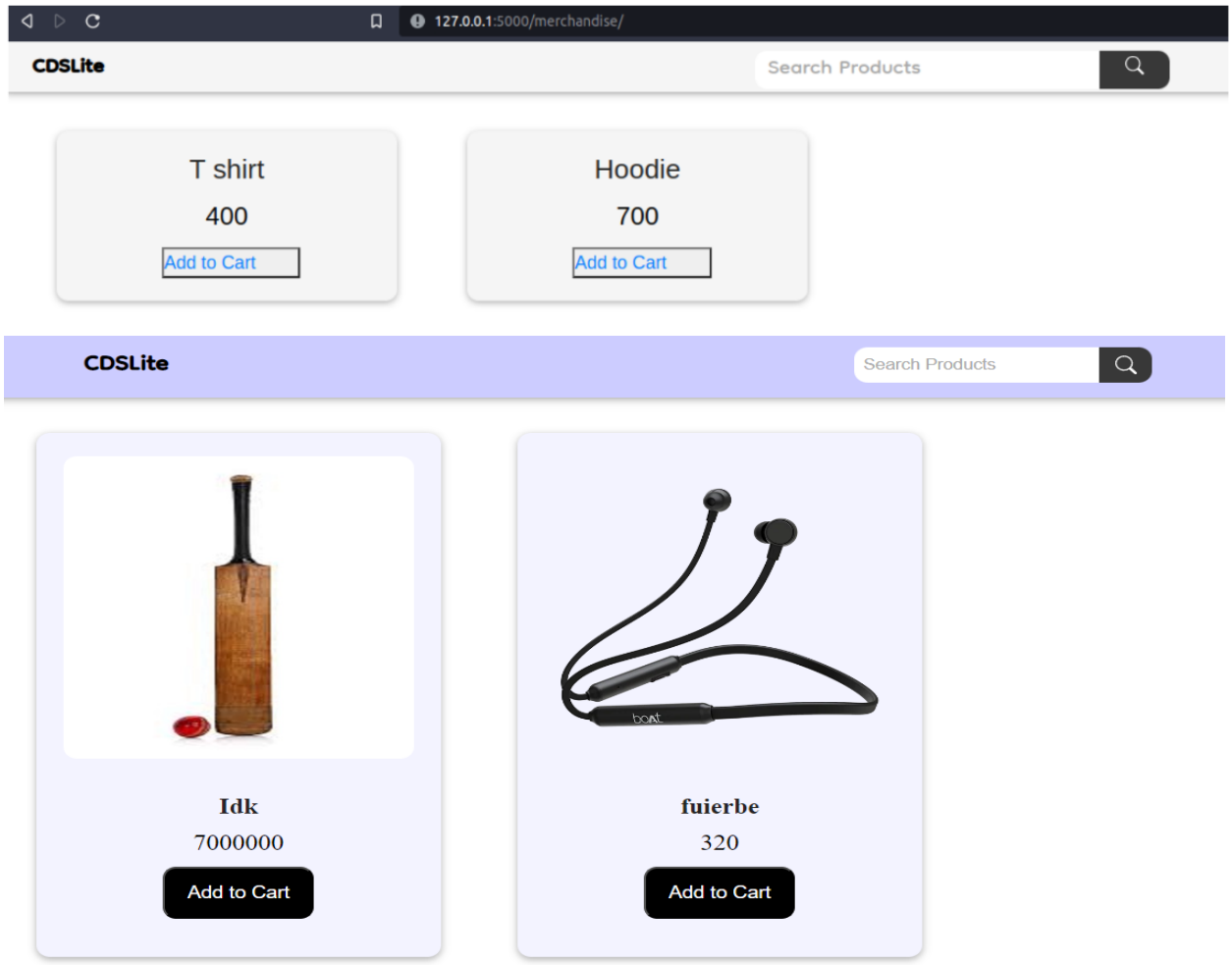



***Fig.-1**: Images were added to the product card.(Before v/s After 1st Feedback)*

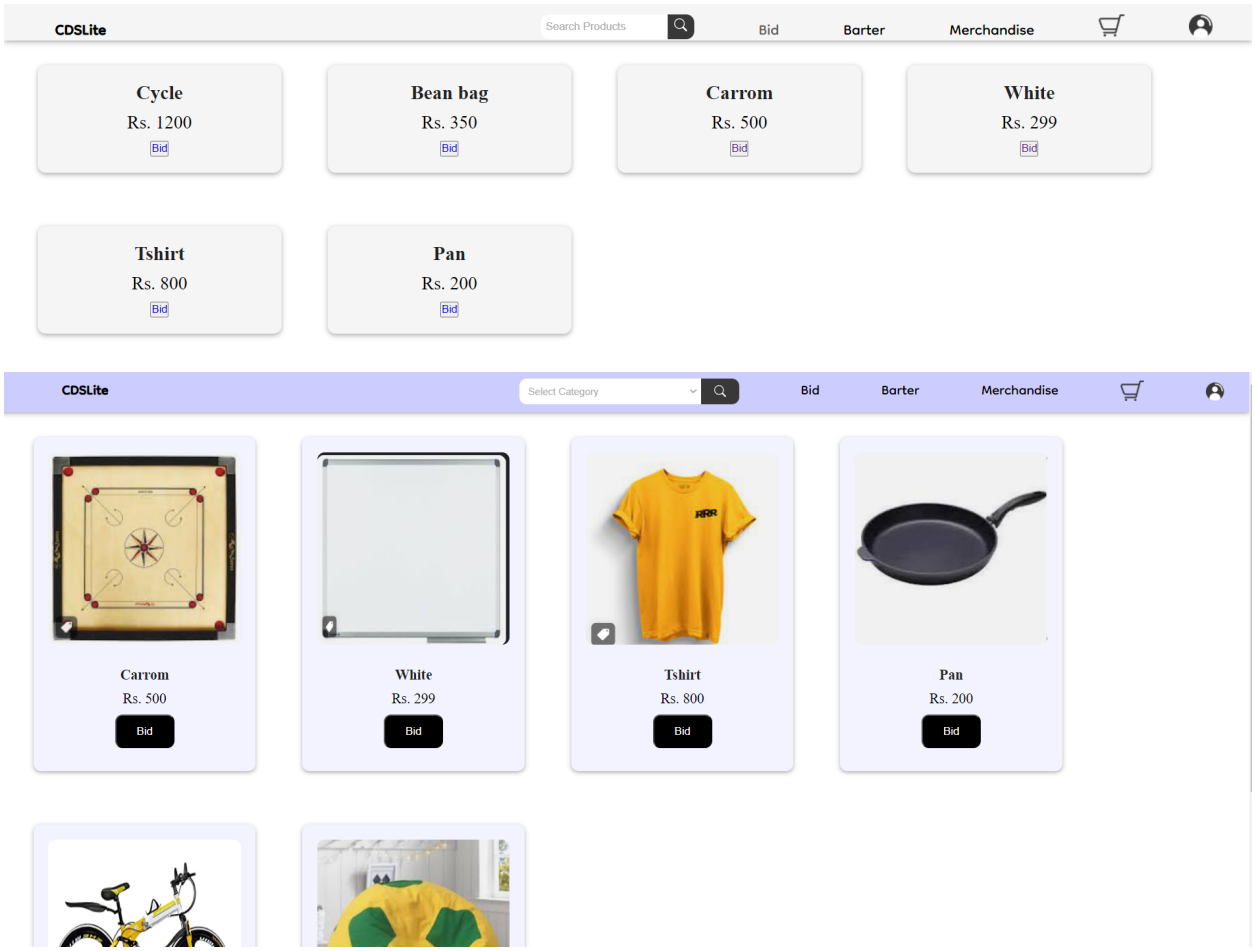



***Fig.-2**: UI was made vibrant.(Before v/s After 1st Feedback)*

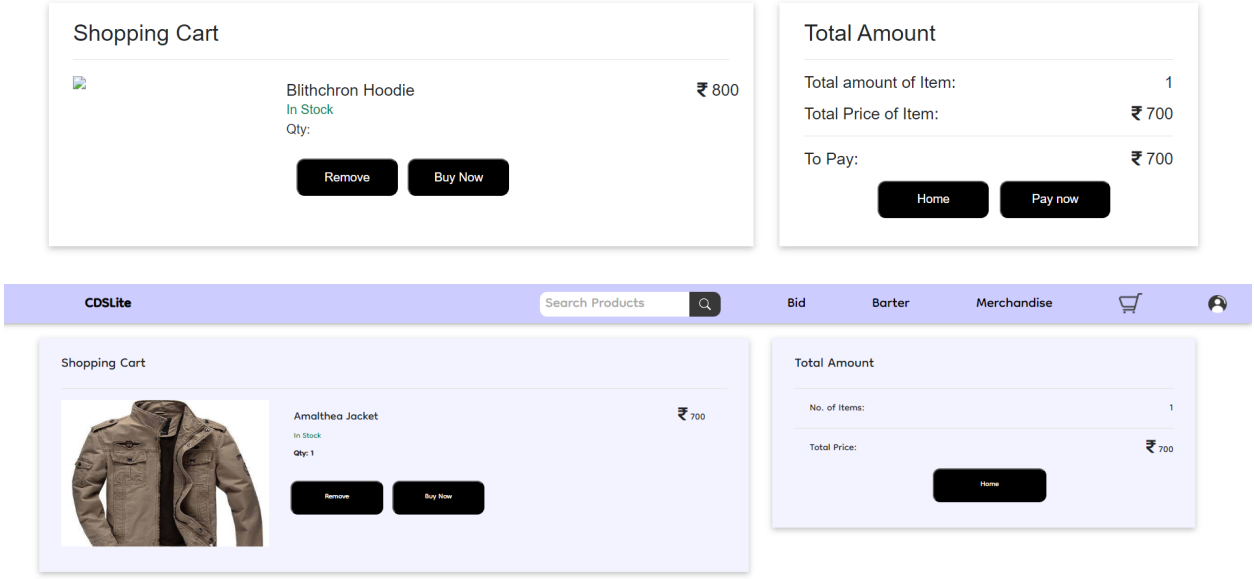



***Fig.-3**: Navigation in the shopping cart.(Before v/s After 1st Feedback)*

**Summary of second feedback:**

- Stakeholder: Adarsh Golait (Welfare Secretary and potential user)
- Audio Recording of Survey
- "Very useful application for the college, will be a great addition"
- Couldn't increase product quantity in cart.
- A single page Home page with all products will be more useful.
- For convenient product search, category search on such a home page.

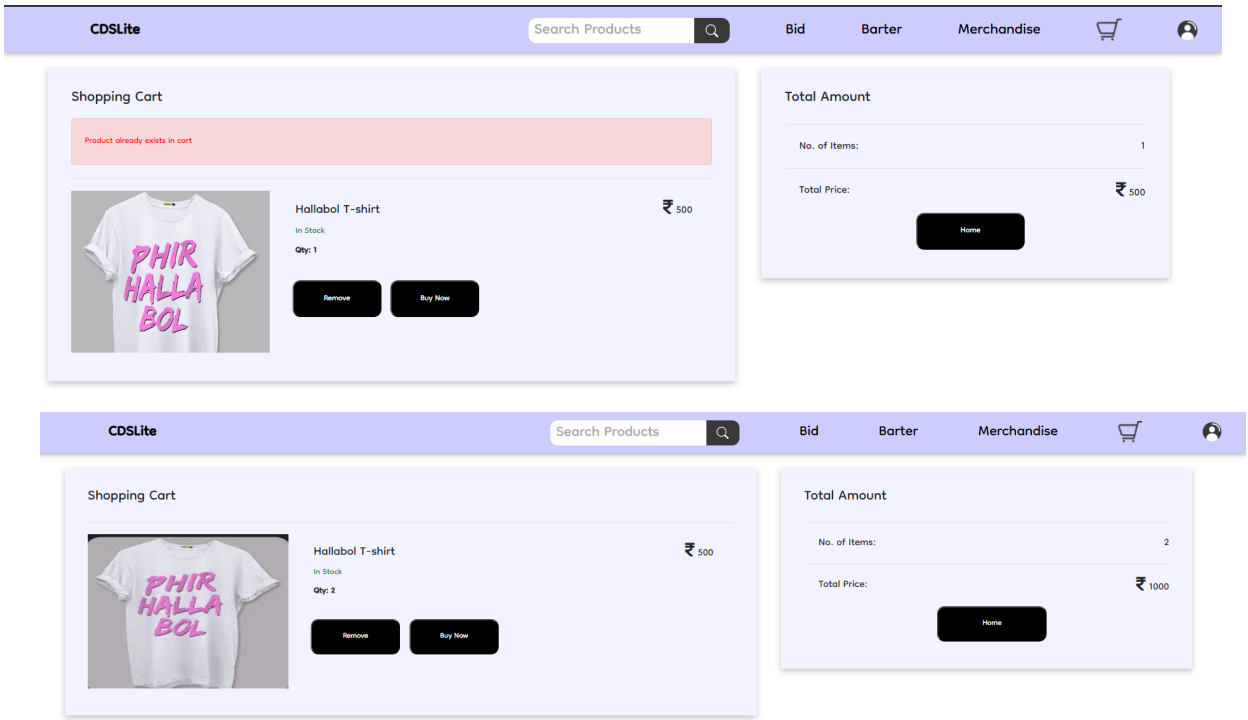



***Fig.-4**: Adding the same product to the cart multiple times, the quantity increases in the cart instead of flashing the message "Product already exists in cart"(After 1st Feedback v/s After 2nd Feedback)*

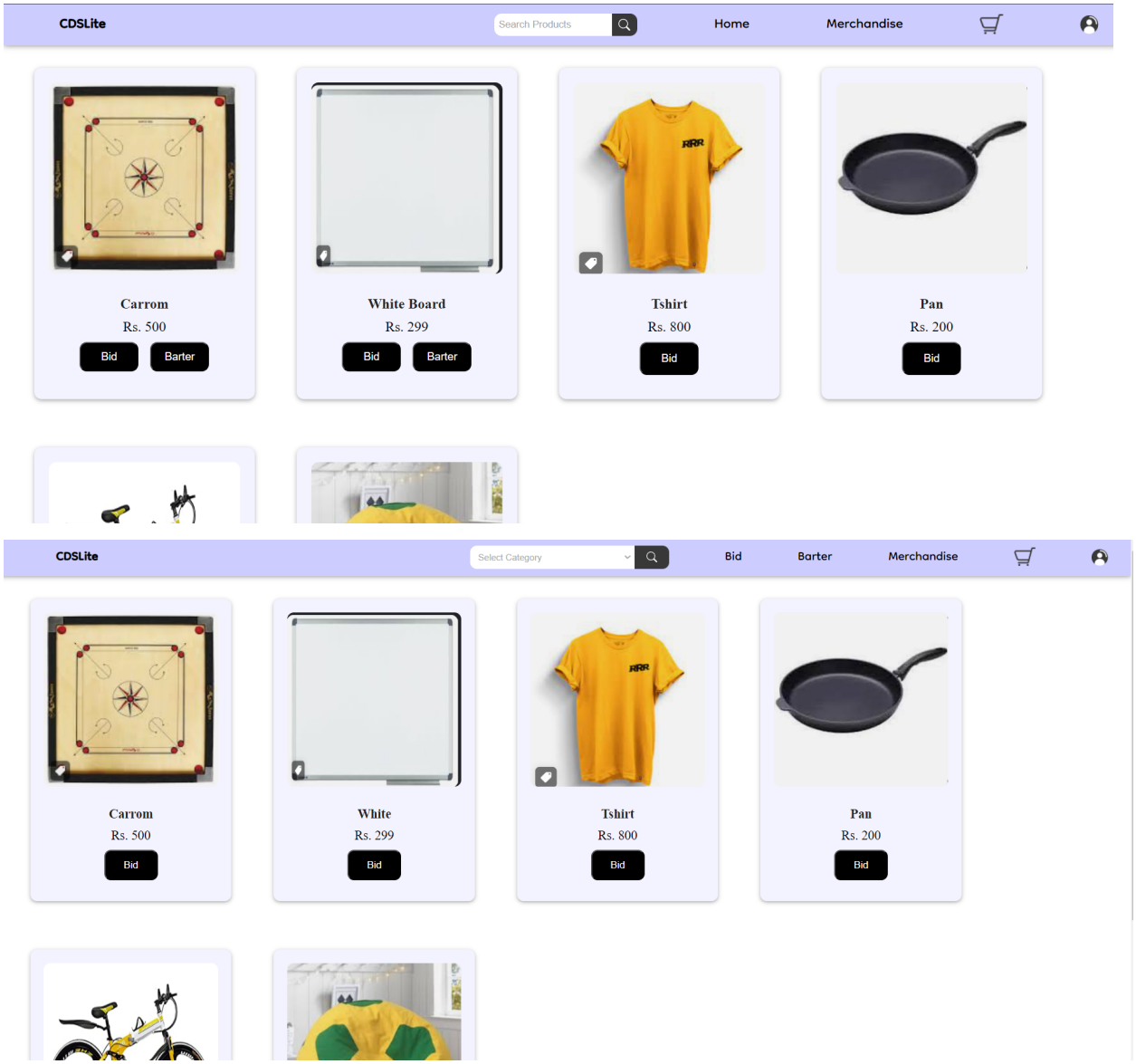



***Fig.-5**: Bid and Barter page combined into Home page.(After 1st Feedback v/s After 2nd Feedback)*

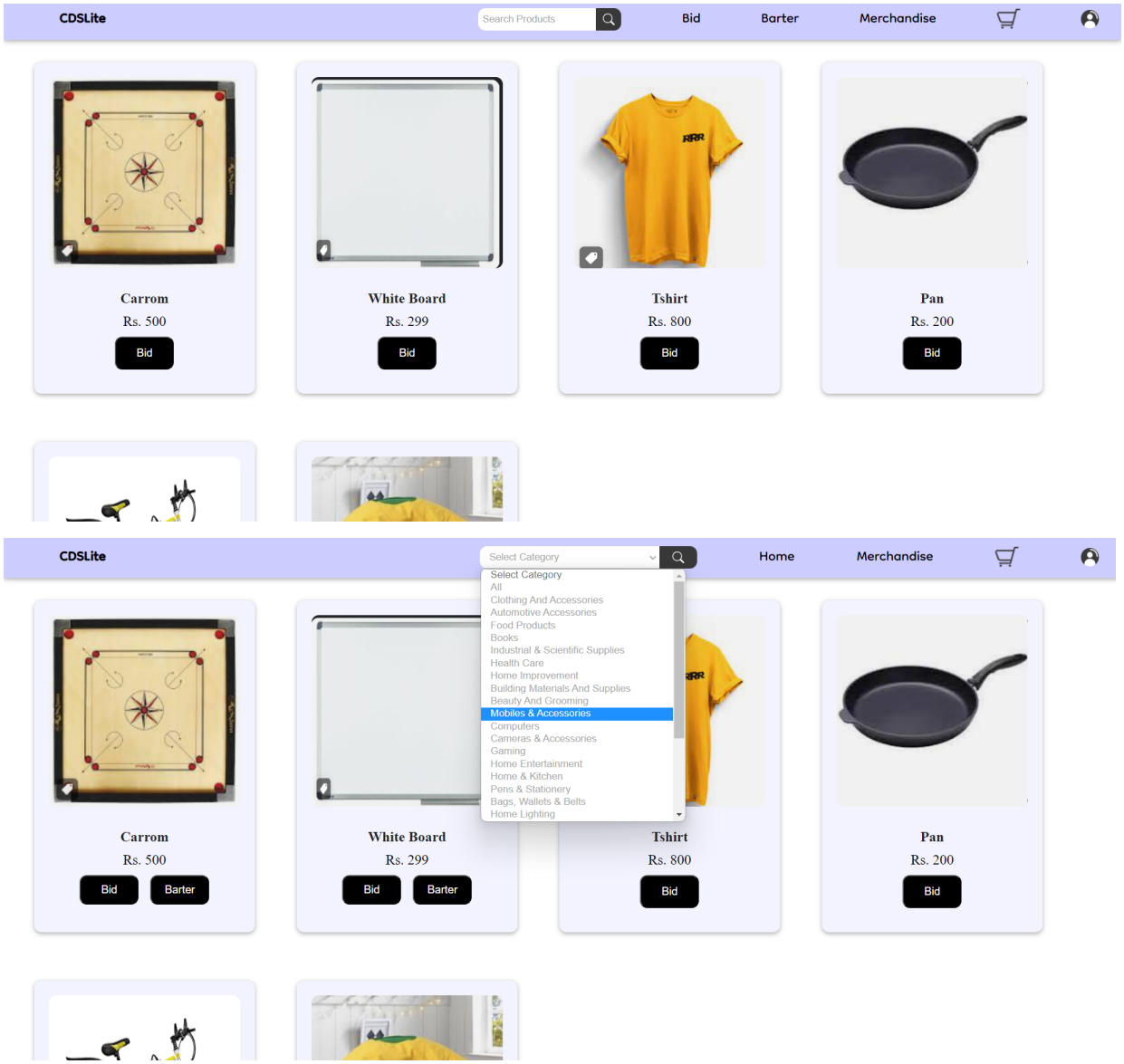
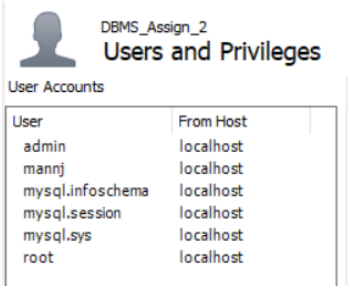



***Fig.-6**: Category search implemented in the Home page.(After 1st Feedback v/s After 2nd Feedback)*

## 2. Different Views:



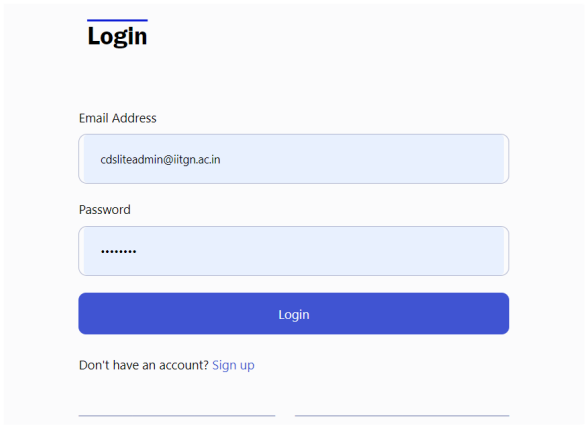

*the root is a normal user of the web page, and the admin is another class of user who has read-only access to the database*

```
create user 'admin'@'localhost' identified by 'Admin123';
GRANT SELECT ON * TO admin@localhost;
```

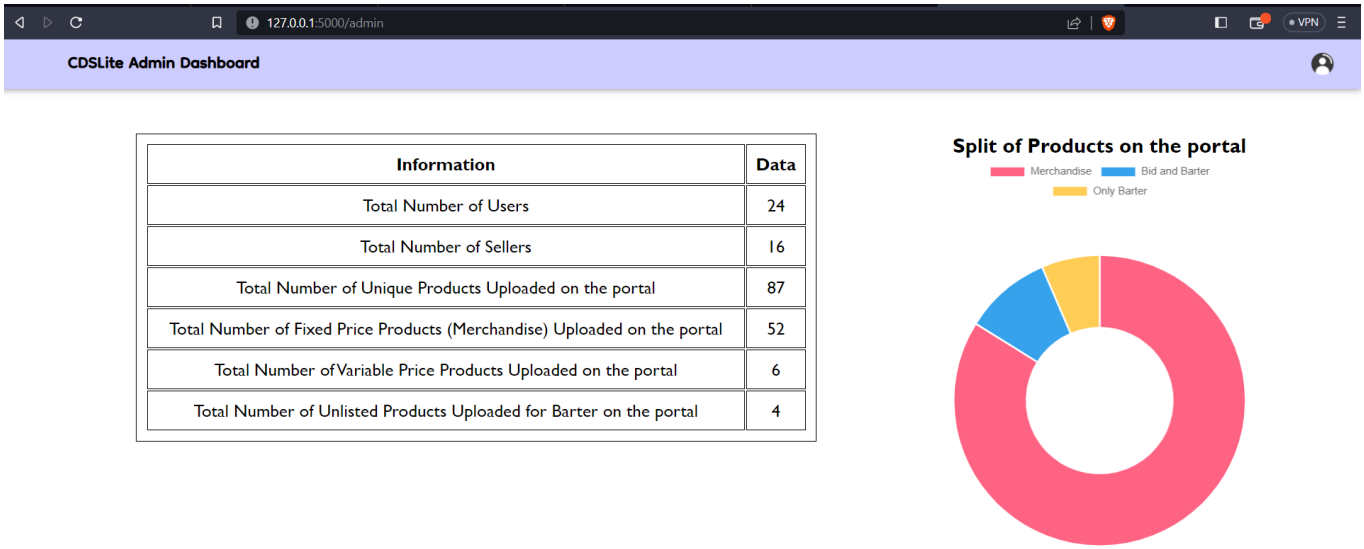
*User Admin created*

**Admin View:**

```
MYSQL_HOST = "localhost"
MYSQL_USER = "admin"
PASSWORD = "Admin123"
MYSQL_DATABASE = "CDSlite"
RAZORPAY_API_KEY = "rzp_test_idu4av7m0TTFkt"
RAZORPAY_API_SECRET = "Wsn0lWwoFG5XhWvgbDlXnhtN"
```

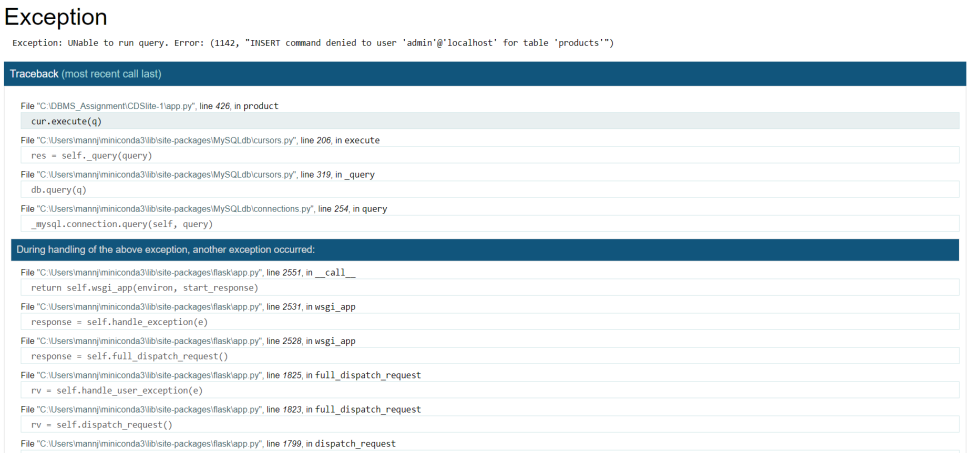



*Login with Admin credentials as present in the database*




*The Admin view on frontend*

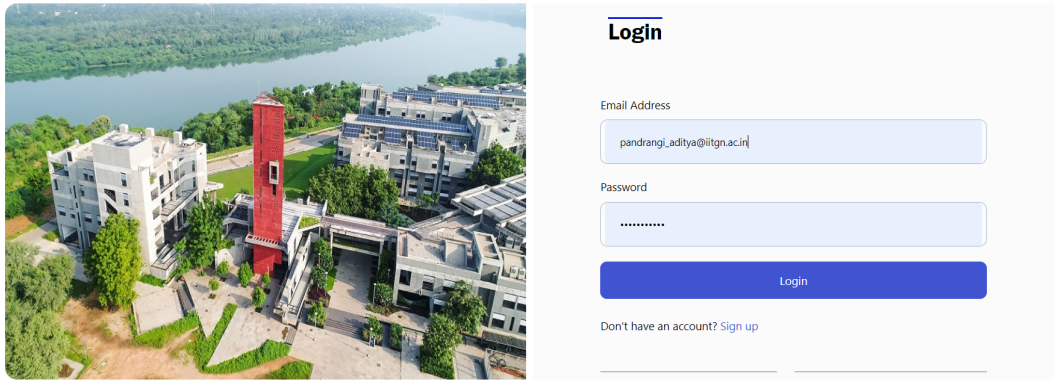
The above is the admin view of the website, where the admin can view the web portal's statistics created with a view query on the backend. The admin has been granted the Select privileges only on the database. The admin can view the details and passwords of all the registered users, the products being sold, etc.



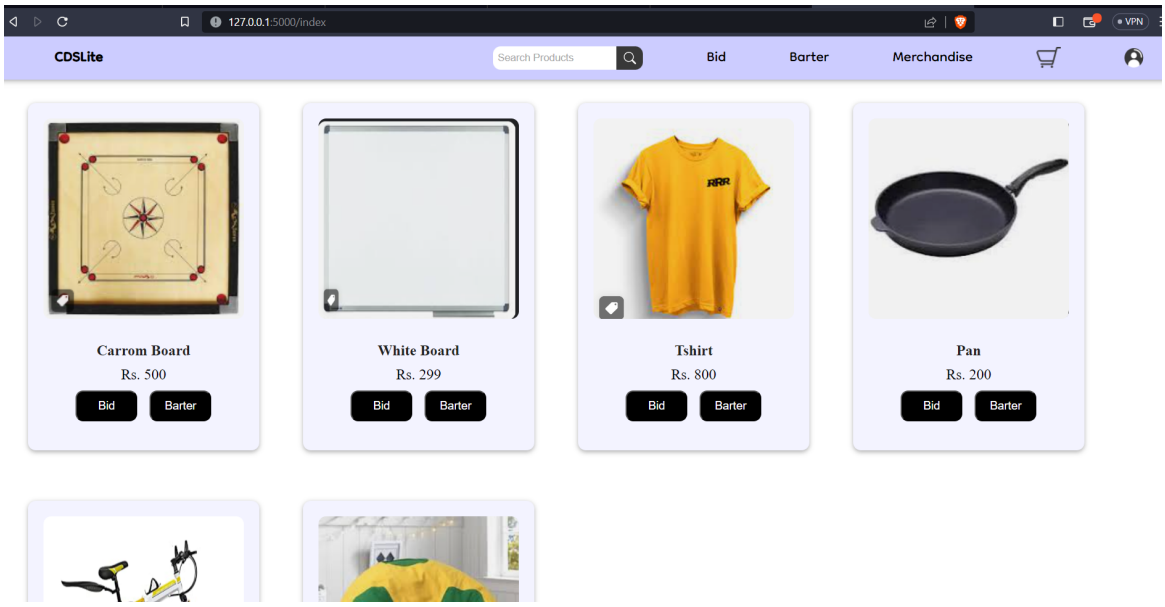

*Admin is not granted access to the insert privileges*

**User View:**

```
MYSQL_HOST = "localhost"
MYSQL_USER = "root"
PASSWORD = "password"
MYSQL_DATABASE = "CDSlite"
RAZORPAY_API_KEY = "rzp_test_idu4av7m0TTFkt"
RAZORPAY_API_SECRET = "Wsn0lWwoFG5XhWvgbDlXnhtN"
```



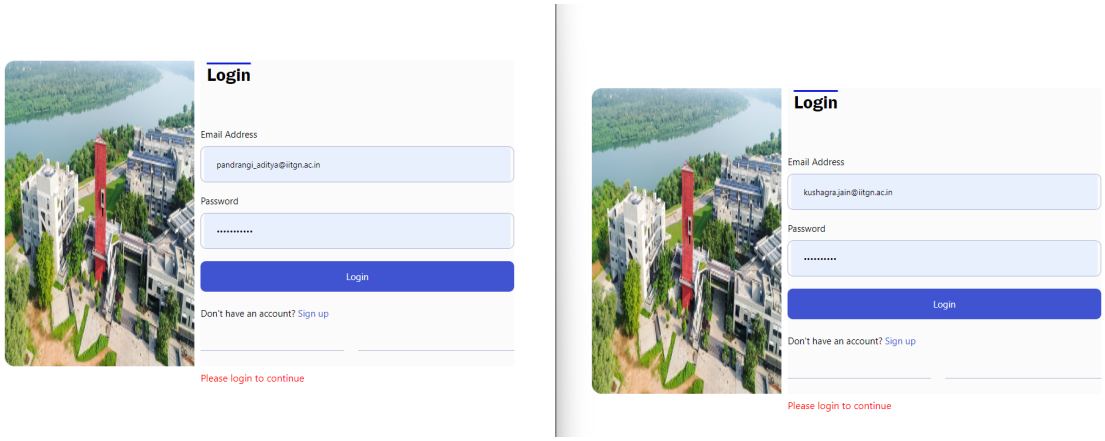

*Login with User credentials as present in the database*



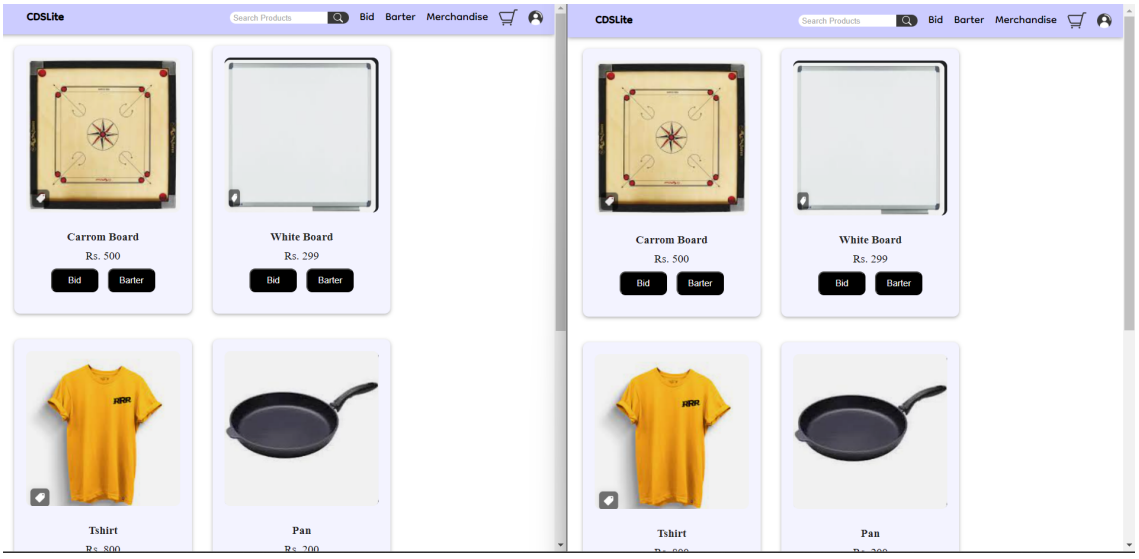

The User has all the privileges on the database for its own products. The user cannot view the passwords and other registered users as the view does not contain details of other users. Users can add their products as they have the privilege to insert them. Users have been granted all privileges on the database except the view for other registered users.

## 3.2 Responsibility of G2:

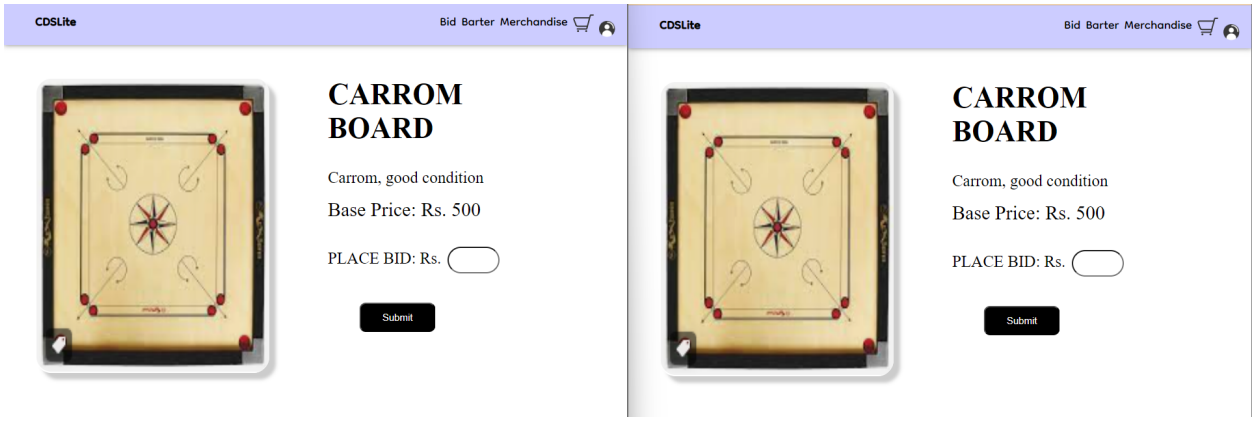
### 1. Concurrent Multi-user access:

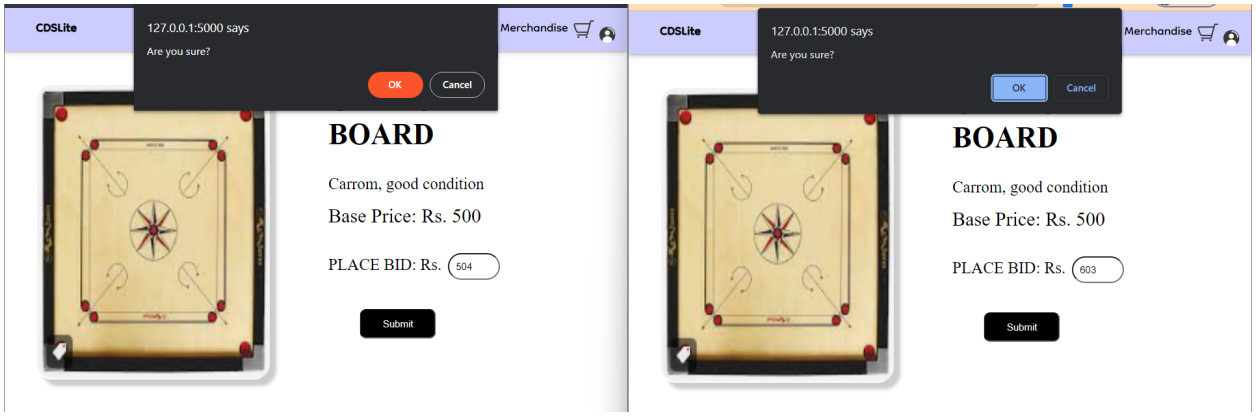



*Simultaneously logging in as two different users*

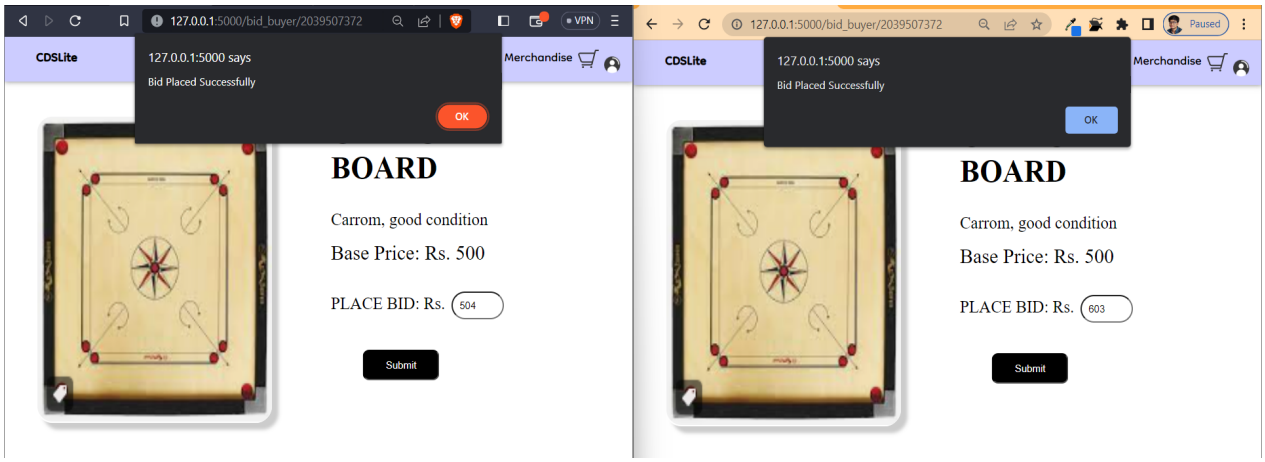



*Similar index views for two different users*






*Placing Bids on the same product simultaneously but as different users*




- **How concurrency is being maintained?**
  To achieve concurrency, locks are applied to a table. There are two types of locks: Shared locks(read lock) and Exclusive locks (Write Lock). The read lock gives access to read operations and restricts other transactions from updating it. The write lock gives access to both read and write operations on the data item and restricts other transactions from accessing that data item.
  In our case, a write lock is applied on the bid table in the database for an efficient insert query. The below example shows trying to insert bids in the bid table by two different users concurrently. Due to locks, the schedule happens concurrently, and we can see both the bids addition in the database without any errors.

- **Code segment for implementing locks →**

```python
@app.route('/bid_buyer/<id_>', methods=["GET", "POST"])
def bid_buyer(id_):
    if request.method == 'POST':
        q = f"LOCK Table BidTable Write"
        cur = mysql.connection.cursor()
        try:
            cur.execute(q)
        except Exception as e:
            raise Exception(f"UNable to run query. Error: {e}")
        bid_id = None
        user_id = session['uid']
        cur = mysql.connection.cursor()
        while True:
            bid_id = generate_uuid()
            query = f"SELECT * from BidTable WHERE BidTable.BidID='{bid_id}'"
            response = cur.execute(query)
            if response == 0:       You, 4 weeks ago • Bid on Products done …
                break
        form_details = request.form
        bid_price = form_details["bid"]
        q = f"INSERT INTO BidTable VALUES ('{bid_id}','{id_}','{user_id}',DEFAULT,{bid_price},'Pending')"
        try:
            cur.execute(q)
            mysql.connection.commit()
        except Exception as e:
            raise Exception(f"UNable to run query. Error: {e}")

        q = f"Unlock Tables"
        cur = mysql.connection.cursor()
        try:
            cur.execute(q)
        except Exception as e:
            raise Exception(f"UNable to run query. Error: {e}")

        return redirect(url_for('index'))
```

Result Grid | Filter Rows: | Edit: | Export/Import:

| BidID | ProductID | UserID | BidDate | BidPrice | BidStatus |
|---|---|---|---|---|---|
| 1504759028 | 12 | 1001 | 2023-03-30 15:07:09 | 6000 | Confirmed |
| 1732296900 | 12 | 1007 | 2023-04-14 16:22:54 | 1 | Pending |
| 1824317650 | 12 | 1001 | 2023-04-14 16:23:06 | 1 | Pending |
| 2719938481 | 11 | 1007 | 2023-04-14 16:32:08 | 150 | Pending |
| 2878501767 | 11 | 1001 | 2023-04-14 16:32:28 | 100 | Pending |
| 3266687519 | 2039507372 | 1007 | 2023-04-14 18:07:19 | 504 | Pending |
| 5016766166 | 2039507372 | 1007 | 2023-04-14 18:07:41 | 503 | Pending |
| 5039785171 | 1 | 1007 | 2023-04-14 16:34:38 | 1553 | Pending |
| 5834331217 | 1 | 1001 | 2023-04-14 16:34:48 | 1036 | Pending |
| 6535740667 | 12 | 1001 | 2023-04-14 16:13:28 | 2000 | Pending |
| 6633936051 | 12 | 1007 | 2023-04-14 16:13:29 | 1000 | Pending |
| 6915732371 | 2 | 27155... | 2023-03-30 16:02:42 | 5201 | Pending |
| 8258459967 | 2039507372 | 1007 | 2023-04-14 18:08:22 | 504 | Pending |
| 8975620063 | 2039507372 | 1001 | 2023-04-14 18:08:31 | 603 | Pending |
| B00001 | 5 | 1018 | 2023-01-09 16:08:42 | 850 | Declined |
| B00002 | 5 | 1001 | 2023-01-09 19:10:22 | 900 | Confirmed |
| B00003 | 5 | 1005 | 2023-01-08 11:42:07 | 750 | Declined |
| B00004 | 5 | 1015 | 2023-01-08 22:35:49 | 800 | Declined |

The highlighted entries show that the two bids have been inserted into the table concurrently with a time gap of approx. one minute. This is because of locks. The other user cannot access the bid table until the lock is released by the first user.

In general, MySQL relations follow the InnoDB engine, which allows concurrency. Locks show us a bigger picture of how the execution takes place for concurrency.

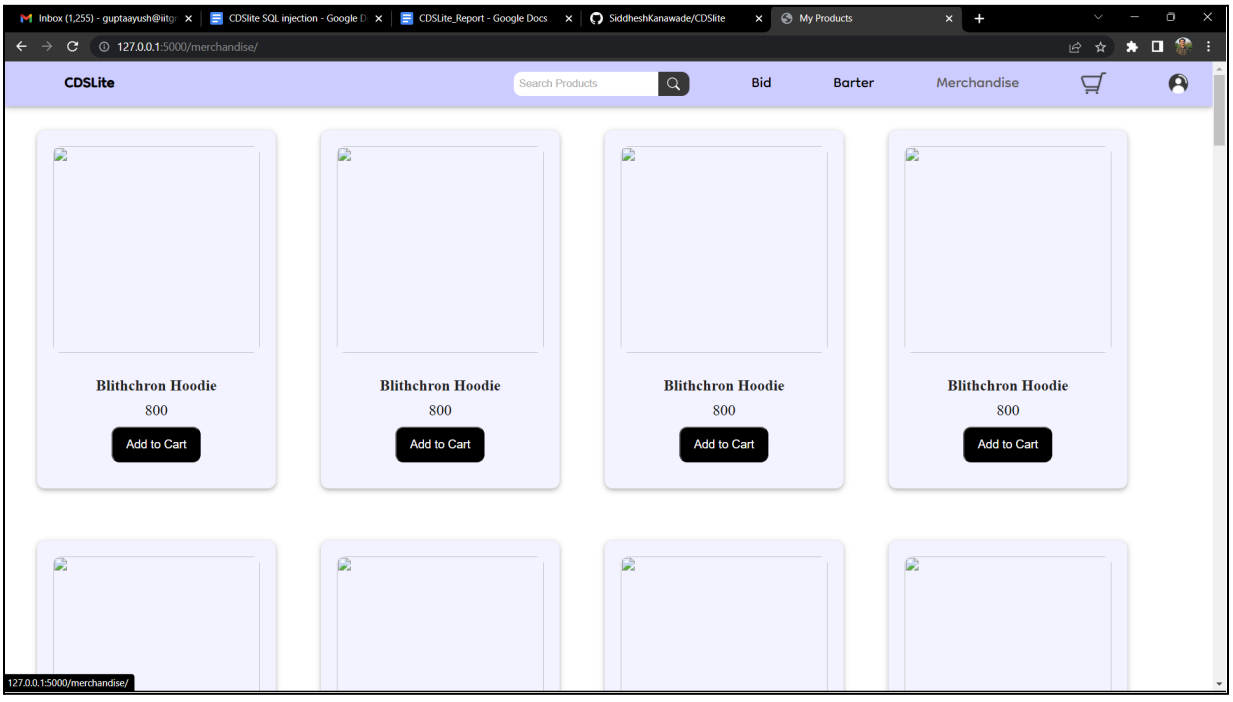
## 2. Changes in the Database:

### # Addition of Image Table -

The “Image” table has been added to the schema to add an image corresponding to a productID.The data type for the image is of “BLOB” type. Earlier, we used one image for every product from frontend. However, now it has been mapped to a particular productID. The image table stores the information on productID and image. It was valuable feedback from the stakeholder to map images to their products, and it was successfully integrated. Now, the application can dynamically insert, update, delete, and load images from and in the database.

```
create table Image (
ProductID varchar(15),
Img LONGBLOB NOT NULL,
foreign key(ProductID) references Products(ProductID) ON DELETE CASCADE,
primary key(ProductID)
);
```

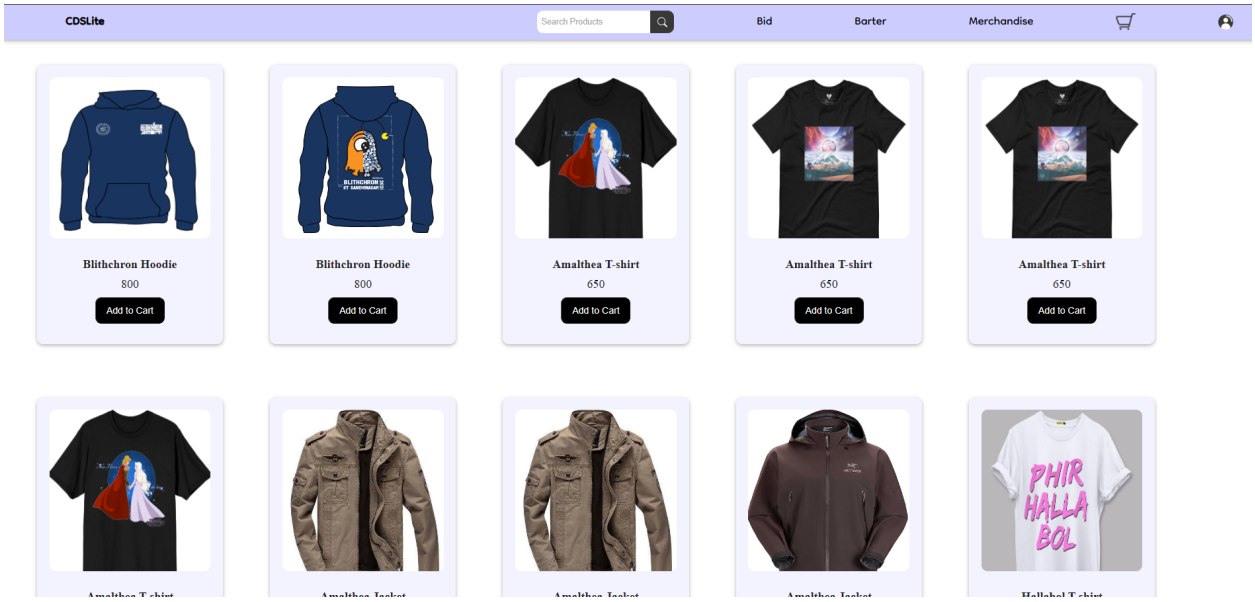
*Changes in the schema*




*Before the image table was included*

| | ProductID | Img |
|---|---|---|
| ▶ | 12 | BLOB |
| | 16 | BLOB |
| | 17 | BLOB |
| | 18 | BLOB |
| | 19 | BLOB |
| | 20 | BLOB |
| | 2039507372 | BLOB |
| | 21 | BLOB |
| | 2159921716 | BLOB |
| | 22 | BLOB |
| | 23 | BLOB |
| | 2349058646 | BLOB |
| | 24 | BLOB |
| | 2464503402 | BLOB |
| | 25 | BLOB |
| | 26 | BLOB |
| | 27 | BLOB |
| | 28 | BLOB |

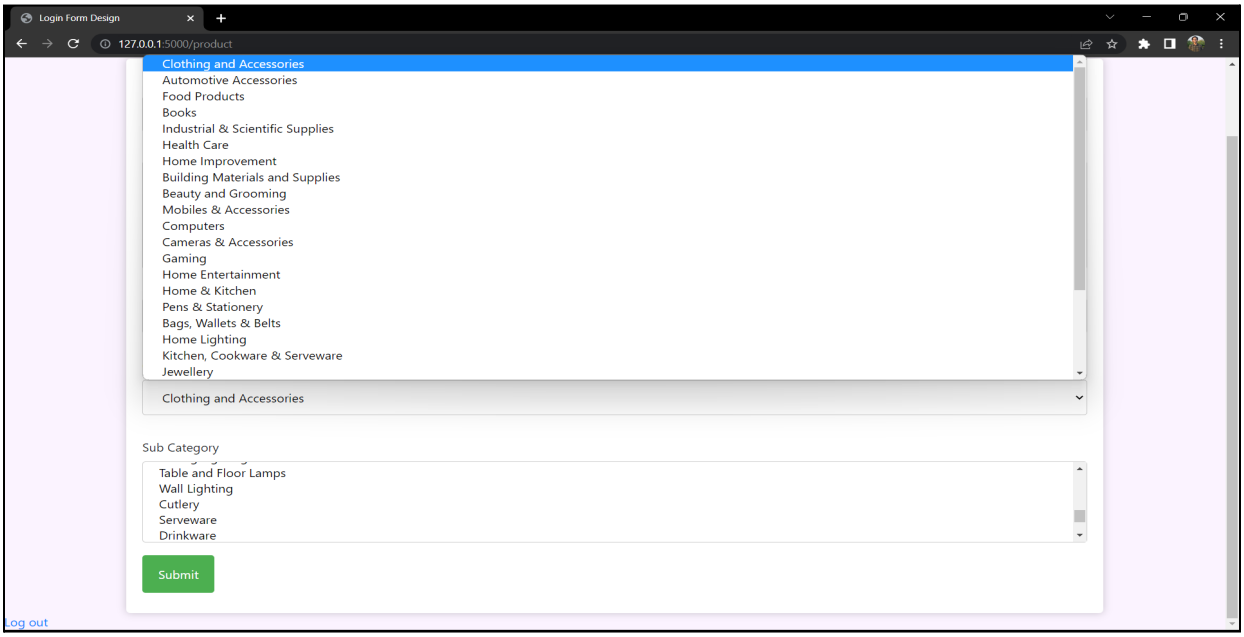
*The Image table*




*Application after integrating the Image Table*

**# Mapping Subcategories with Categories -**

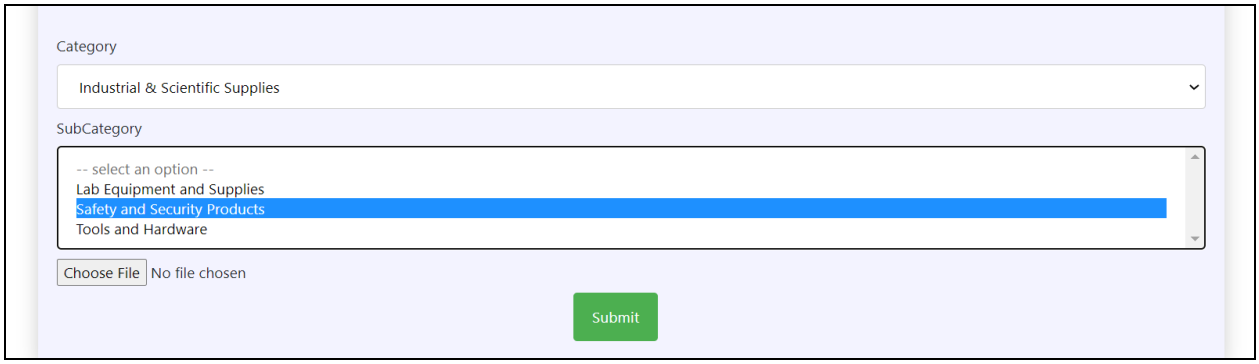
Earlier, the subcategories were not mapped with the categories table. In that case, a user can choose any subcategory for a category. For e.g., choosing the pet food subcategory in the books category does not make sense. We have created a “constrained” relation that maps subcategories to categories. Now, the user can select subcategories as per the category. The relations FPhasSubCat and VPhasSubCat were added and validated.




*Application before constraining subcategories and categories*

| CategoryID | SubCategoryID |
|---|---|
| C001 | SC0001 |
| C001 | SC0002 |
| C001 | SC0003 |
| C001 | SC0004 |
| C001 | SC0005 |
| C001 | SC0006 |
| C001 | SC0007 |
| C002 | SC0008 |
| C002 | SC0009 |
| C002 | SC0010 |
| C003 | SC0011 |
| C003 | SC0012 |
| C003 | SC0013 |
| C003 | SC0014 |
| C003 | SC0015 |
| C004 | SC0016 |
| C004 | SC0017 |
| C004 | SC0018 |

*The Constrained Table*




*After mapping the categories with subcategories*

**# Created Relation Products -**

The relation 'Products' is created to integrate all variable and fixed-price products in a single relation with unique IDs for each product. The other tables have foreign key references to this table for productID.

```
create table Products(
ProductID varchar(15) primary key,
SellerID varchar(15),
foreign key(SellerID) References Seller(SellerID) ON DELETE CASCADE
);
```

*Changes in schema*

| ProductID | SellerID |
|---|---|
| 67 | 1001 |
| 28 | 1002 |
| 29 | 1002 |
| 30 | 1002 |
| 31 | 1002 |
| 16 | 1003 |
| 17 | 1003 |
| 18 | 1003 |
| 19 | 1003 |
| 20 | 1004 |
| 21 | 1004 |
| 22 | 1004 |
| 23 | 1004 |
| 24 | 1004 |
| 25 | 1004 |
| 26 | 1004 |
| 27 | 1004 |

*Product Table mapping the products to different sellers*

**# Scrapped the Availability Attribute from FP_Products Table -**

The "FP_Products" relation has a column "Availability" along with "Quantity." The Availability column is redundant as Quantity = 0 resembles no availability. Thus, we dropped this column.

```
Alter table FP_Products
drop column Availability;
select * from Products;
```

| ProductID | ProductName | Description_ | Availability | MRP | Quantity | CreationDate | UpdationDate | CategoryID |
|---|---|---|---|---|---|---|---|---|
| 16 | Blithchron Hoodie | Dark Blue Colored Blithchron Hoodie with Blithch... | Yes | 800 | 20 | 2022-11-24 | 2022-11-26 | C001 |
| 17 | Blithchron Hoodie | Dark Blue Colored Blithchron Hoodie with Blithch... | Yes | 800 | 10 | 2022-11-24 | 2022-11-26 | C001 |
| 18 | Blithchron Hoodie | Dark Blue Colored Blithchron Hoodie with Blithch... | No | 800 | 0 | 2022-11-24 | 2022-11-26 | C001 |
| 19 | Blithchron Hoodie | Dark Blue Colored Blithchron Hoodie with Blithch... | No | 800 | 15 | 2022-11-24 | 2022-11-26 | C001 |
| 20 | Amalthea T-shirt | Black Colored Amalthea T-Shirt, Size S | Yes | 650 | 18 | 2023-02-06 | 2023-02-08 | C001 |
| 21 | Amalthea T-shirt | Black Colored Amalthea T-Shirt, Size M | Yes | 650 | 36 | 2023-02-06 | 2023-02-08 | C001 |
| 22 | Amalthea T-shirt | Black Colored Amalthea T-Shirt, Size L | Yes | 650 | 26 | 2023-02-06 | 2023-02-08 | C001 |
| 23 | Amalthea T-shirt | Black Colored Amalthea T-Shirt, Size XL | Yes | 650 | 9 | 2023-02-06 | 2023-02-08 | C001 |
| 24 | Amalthea Jacket | Black Colored Amalthea Winter Jacket with Amal... | No | 700 | 2 | 2022-10-12 | 2022-10-15 | C001 |
| 25 | Amalthea Jacket | Black Colored Amalthea Winter Jacket with Amal... | Yes | 700 | 1 | 2022-10-12 | 2022-10-15 | C001 |
| 26 | Amalthea Jacket | Black Colored Amalthea Winter Jacket with Amal... | Yes | 700 | 36 | 2022-10-12 | 2022-10-15 | C001 |
| 27 | Amalthea Jacket | Black Colored Amalthea Winter Jacket with Amal... | Yes | 700 | 26 | 2022-10-12 | 2022-10-15 | C001 |
| 28 | Hallabol T-shirt | Black Colored T-Shirt with Hallabol Logo, Size S | No | 500 | 4 | 2022-10-07 | 2022-10-07 | C001 |
| 29 | Hallabol T-shirt | Black Colored T-Shirt with Hallabol Logo, Size M | Yes | 500 | 5 | 2022-10-07 | 2022-10-07 | C001 |
| 30 | Hallabol T-shirt | Black Colored T-Shirt with Hallabol Logo, Size L | Yes | 500 | 0 | 2022-10-07 | 2022-10-07 | C001 |
| 31 | Hallabol T-shirt | Black Colored T-Shirt with Hallabol Logo, Size XL | No | 500 | 6 | 2022-10-07 | 2022-10-22 | C001 |
| 32 | IITGN Polo T-shirt | Grey Colored T-Shirt with Maroon Colored Collar... | Yes | 450 | 9 | 2022-07-28 | 2022-07-29 | C001 |
| 33 | IITGN Polo T-shirt | Grey Colored T-Shirt with Maroon Colored Collar... | Yes | 450 | 35 | 2022-07-28 | 2022-07-29 | C001 |

*FP_Products with Availability as an attribute*

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

| ProductID | ProductName | Description_ | MRP | Quantity | CreationDate | UpdationDate | CategoryID |
|---|---|---|---|---|---|---|---|
| 16 | Blithchron Hoodie | Dark Blue Colored Blithchron Hoodie with Blithch... | 800 | 20 | 2022-11-24 | 2022-11-26 | C001 |
| 17 | Blithchron Hoodie | Dark Blue Colored Blithchron Hoodie with Blithch... | 800 | 10 | 2022-11-24 | 2022-11-26 | C001 |
| 18 | Blithchron Hoodie | Dark Blue Colored Blithchron Hoodie with Blithch... | 800 | 0 | 2022-11-24 | 2022-11-26 | C001 |
| 19 | Blithchron Hoodie | Dark Blue Colored Blithchron Hoodie with Blithch... | 800 | 15 | 2022-11-24 | 2022-11-26 | C001 |
| 20 | Amalthea T-shirt | Black Colored Amalthea T-Shirt, Size S | 650 | 18 | 2023-02-06 | 2023-02-08 | C001 |
| 21 | Amalthea T-shirt | Black Colored Amalthea T-Shirt, Size M | 650 | 36 | 2023-02-06 | 2023-02-08 | C001 |
| 22 | Amalthea T-shirt | Black Colored Amalthea T-Shirt, Size L | 650 | 26 | 2023-02-06 | 2023-02-08 | C001 |
| 23 | Amalthea T-shirt | Black Colored Amalthea T-Shirt, Size XL | 650 | 9 | 2023-02-06 | 2023-02-08 | C001 |
| 24 | Amalthea Jacket | Black Colored Amalthea Winter Jacket with Amal... | 700 | 2 | 2022-10-12 | 2022-10-15 | C001 |
| 25 | Amalthea Jacket | Black Colored Amalthea Winter Jacket with Amal... | 700 | 1 | 2022-10-12 | 2022-10-15 | C001 |
| 26 | Amalthea Jacket | Black Colored Amalthea Winter Jacket with Amal... | 700 | 36 | 2022-10-12 | 2022-10-15 | C001 |
| 27 | Amalthea Jacket | Black Colored Amalthea Winter Jacket with Amal... | 700 | 26 | 2022-10-12 | 2022-10-15 | C001 |
| 28 | Hallabol T-shirt | Black Colored T-Shirt with Hallabol Logo, Size S | 500 | 4 | 2022-10-07 | 2022-10-07 | C001 |
| 29 | Hallabol T-shirt | Black Colored T-Shirt with Hallabol Logo, Size M | 500 | 5 | 2022-10-07 | 2022-10-07 | C001 |
| 30 | Hallabol T-shirt | Black Colored T-Shirt with Hallabol Logo, Size L | 500 | 0 | 2022-10-07 | 2022-10-07 | C001 |
| 31 | Hallabol T-shirt | Black Colored T-Shirt with Hallabol Logo, Size XL | 500 | 6 | 2022-10-07 | 2022-10-22 | C001 |
| 32 | IITGN Polo T-shirt | Grey Colored T-Shirt with Maroon Colored Collar... | 450 | 9 | 2022-07-28 | 2022-07-29 | C001 |
| 33 | IITGN Polo T-shirt | Grey Colored T-Shirt with Maroon Colored Collar... | 450 | 35 | 2022-07-28 | 2022-07-29 | C001 |

*FP_Products after removing Availability*

**# Dropped Unnecessary Tables from the database -**

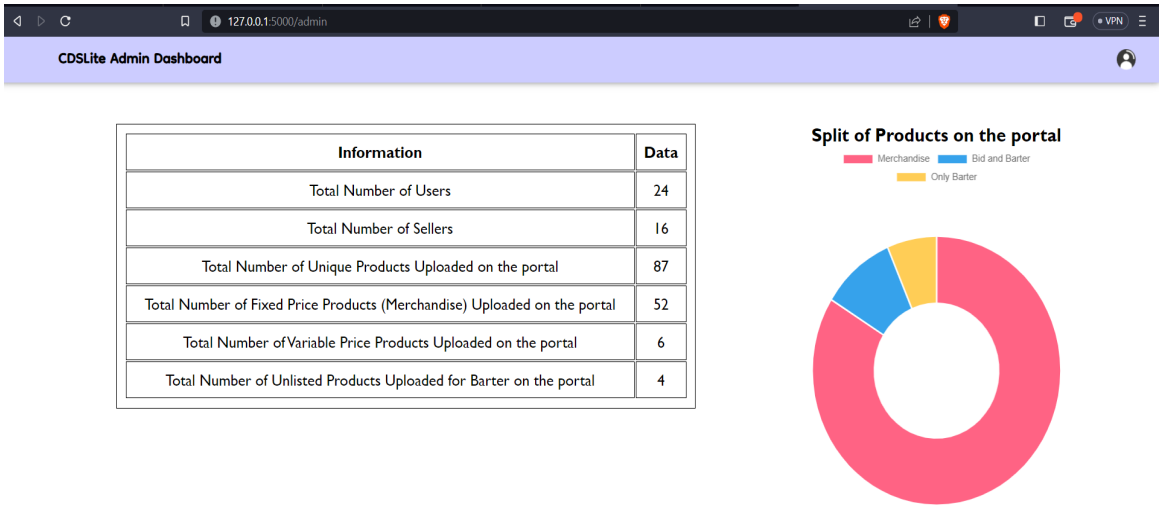
There were certain tables that were redundant in the schema. We deleted certain tables, which are as follows:

- isMerchant
- Variant
- Barter History

The above tables are redundant because we are directly using frontend values to map other relations. The isMerchant is a radio button in add product form. The variant relation was used for different variants of the same product in merchandise. However, we have used the variants as different products, and thus, this relation became redundant. The barter history relation is not required because products are deleted only by the user who uploaded them. Thus, all the details are in the Barter table only.

**# Created Admin view -**

Created a different user Admin in the database who has read-only access to the database. The admin view is created in the database, which dynamically fetches the number of users, sellers, products, etc., on the portal and presents the results to the admin on the admin dashboard.



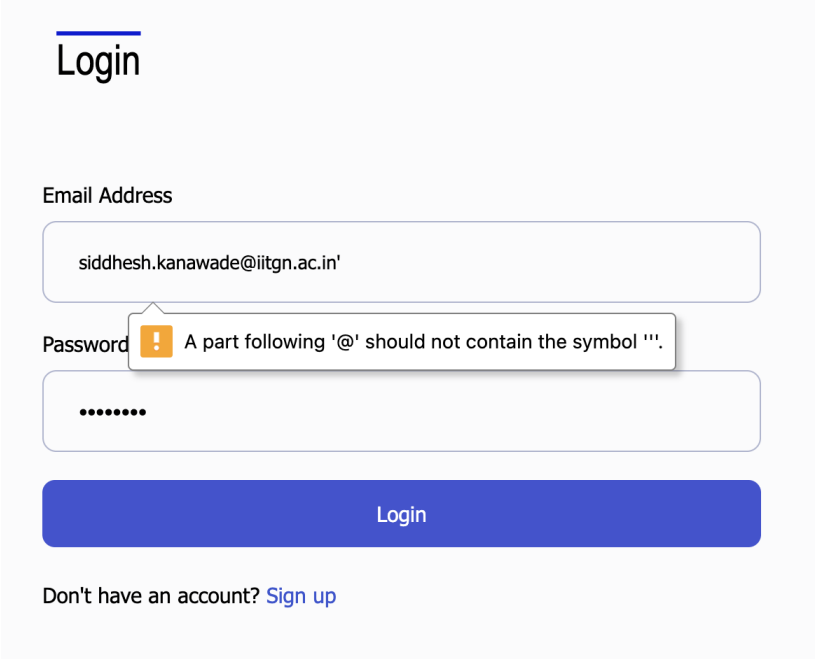

| Information | Data |
|---|---|
| Total Number of Users | 24 |
| Total Number of Sellers | 16 |
| Total Number of Unique Products Uploaded on the portal | 87 |
| Total Number of Fixed Price Products (Merchandise) Uploaded on the portal | 52 |
| Total Number of Variable Price Products Uploaded on the portal | 6 |
| Total Number of Unlisted Products Uploaded for Barter on the portal | 4 |

*The CDSLite Admin Dashboard*

## 3.3 Responsibility of G1 & G2:

### 1. Attacks & their Defenses -

#### → SQL Injection to Bypass Login -




It's not possible to perform SQL injection in the email address in the front-end as UI blocks it due to the specified input format.

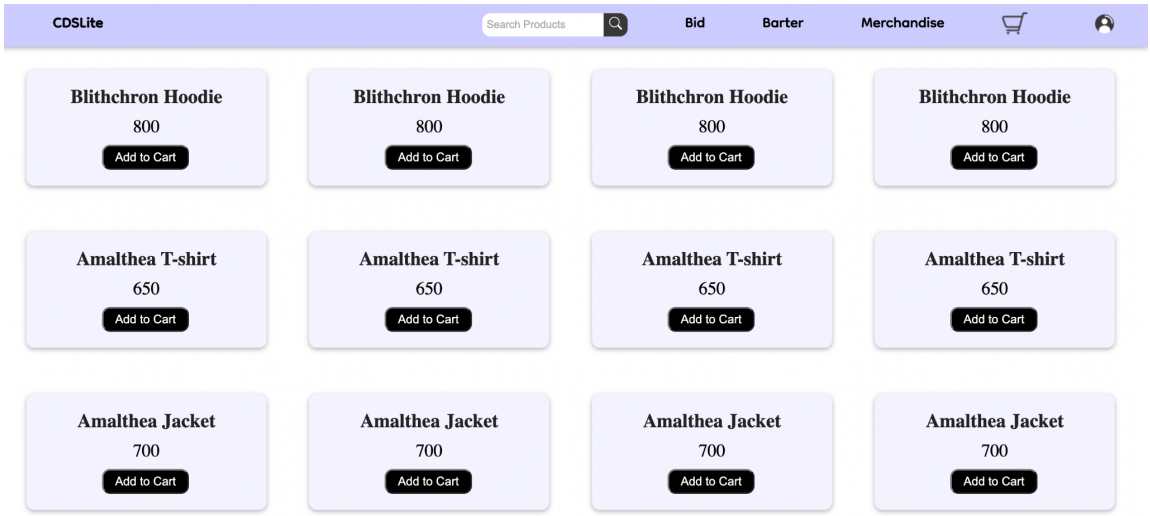
Thus we intercept the request in between using **BurpSuite**

Pretty Raw Hex \n

```
1  POST / HTTP/1.1
2  Host: 127.0.0.1:5000
3  Content-Length: 54
4  Cache-Control: max-age=0
5  sec-ch-ua: "Not:A-Brand";v="99", "Chromium";v="112"
6  sec-ch-ua-mobile: ?0
7  sec-ch-ua-platform: "macOS"
8  Upgrade-Insecure-Requests: 1
9  Origin: http://127.0.0.1:5000
10 Content-Type: application/x-www-form-urlencoded
11 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/112.0.5615.50 Safari/537.36
12 Accept:
   text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
13 Sec-Fetch-Site: same-origin
14 Sec-Fetch-Mode: navigate
15 Sec-Fetch-User: ?1
16 Sec-Fetch-Dest: document
17 Referer: http://127.0.0.1:5000/
18 Accept-Encoding: gzip, deflate
19 Accept-Language: en-GB,en-US;q=0.9,en;q=0.8
20 Cookie: session=eyJsb2dnZWRfaW4iOnRydWUsInNlc3Npb25fbmFtZSI6IlNpZGRoZXNoIiwidWlkIjoiMTM0MzY4NDUxNiJ9.ZDfiFw.UTp6V_Fbags_8BPmvuYBMLzm4rI
21 Connection: close
22
23 email=siddhesh.kanawade%40iitgn.ac.in' OR '1'='1&password=asdfsad
```

*We pass: **‘ OR ‘1’=’1** in the query*

***RESULT:***




We **bypassed** the login.

***SOLUTION:***

Current query:

```
query = f"SELECT * from User WHERE User.Email_ID='{email_id}' and
User.Password_='{password}'"

cur = mysql.connection.cursor()

try:
   cur.execute(query)
   mysql.connection.commit()
except Exception as e:
   raise Exception(f"UNable to run query. Error: {e}")
```

Updated query: Use parameterized queries

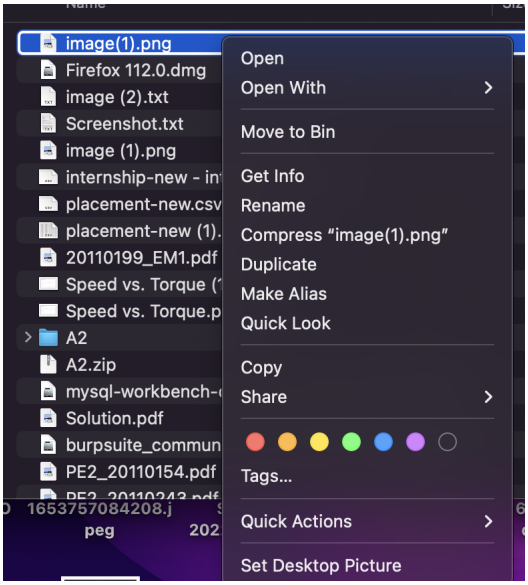
```
query = "SELECT * from User WHERE User.Email_ID=%s and
User.Password_=%s"
values = (email_id, password)

cur = mysql.connection.cursor()

try:
   cur.execute(query, values)
   mysql.connection.commit()
except Exception as e:
   raise Exception(f"UNable to run query. Error: {e}")
```

**→ Uploading Malicious Files to the Database (file upload vulnerability)**

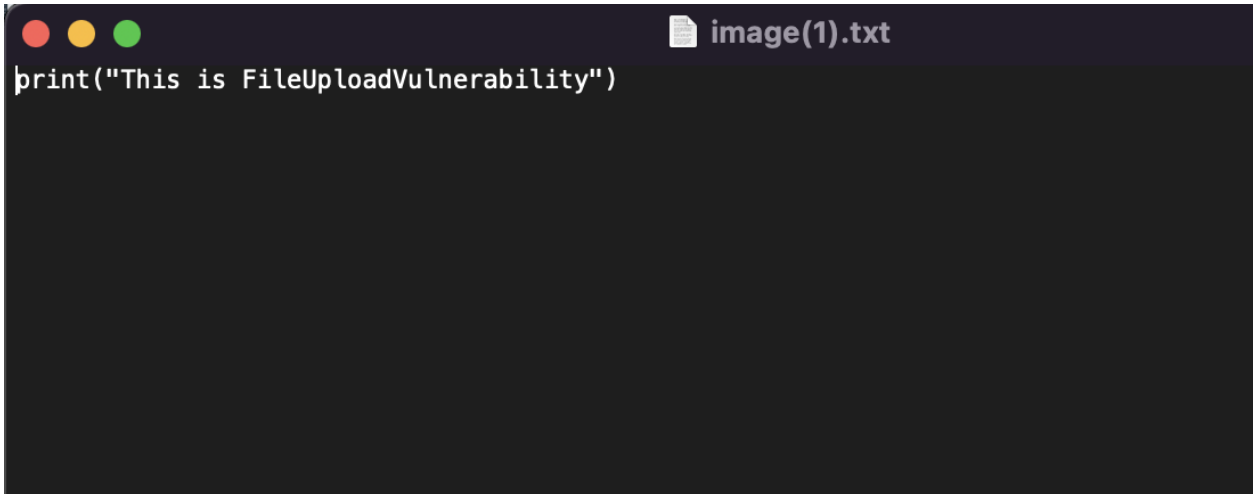
Choose file test.py

Front-end allows you to add any extension files.
Above vulnerability can be used to upload malicious files to the server, which may crash the application. The above can also be exploited to run custom scripts on the server side, which may be used to steal crucial user information.


Open the image in a new tab. This will download the image if the web app is hosted in mozilla firefox



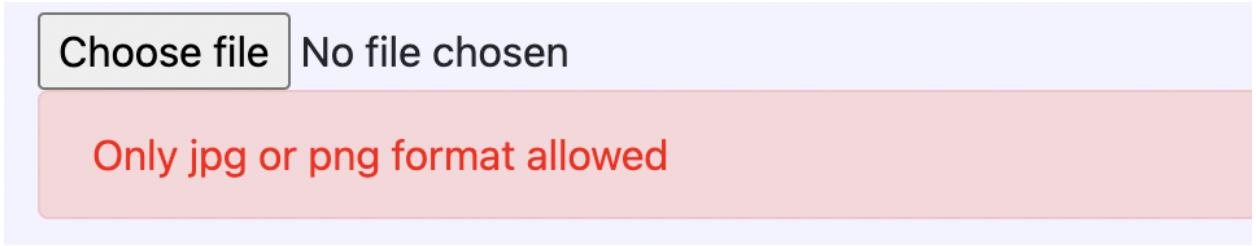

Rename the file extension to .txt




We see that we are successfully able to insert our script into the system. This attack can be used to insert malware into servers or systems.

***SOLUTION:***

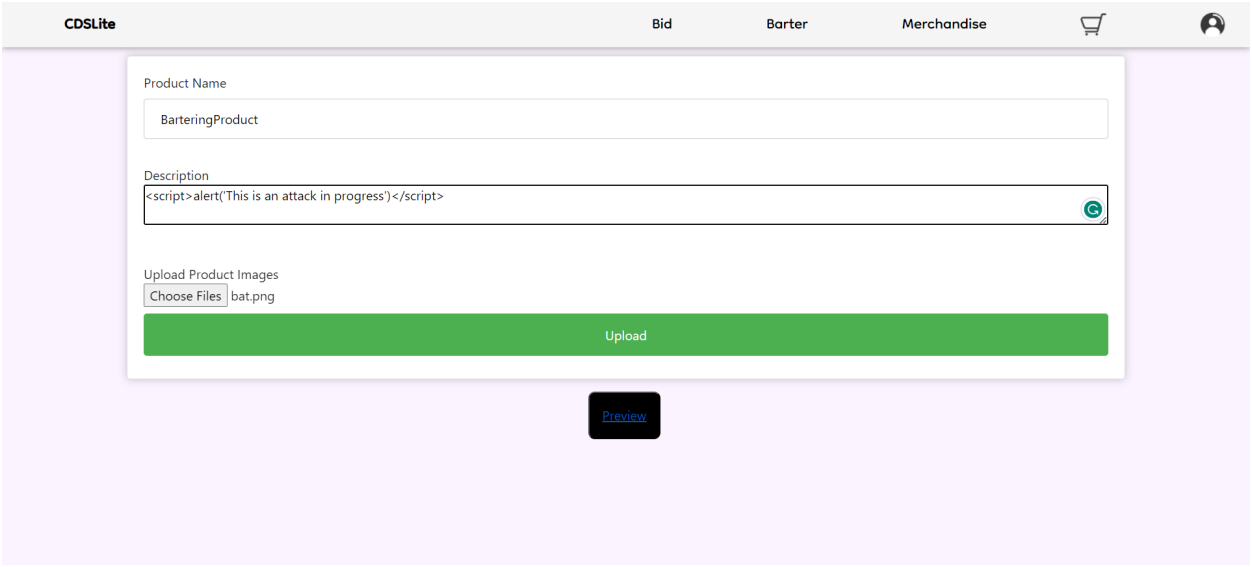
Check file type



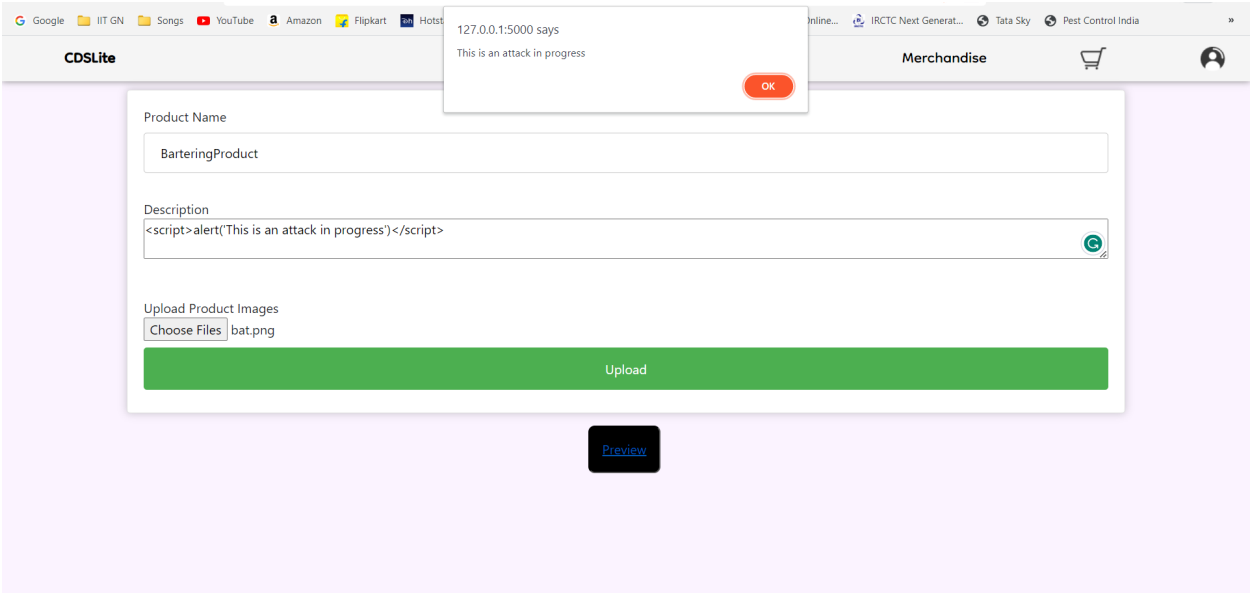

```
file_type = imghdr.what(None, image)
if not file_type=='png' or not file_type=='jpeg':
    flash("Only jpg or png format allowed")
    return redirect(url_for('product'))
```

**→ XSS Attack while adding Barter product on Barterpage-**

If I enter any script in the description field and then want to preview the Product Name and its description then instead of sanitizing the input and displaying it as plain text, the webpage instead runs the script.



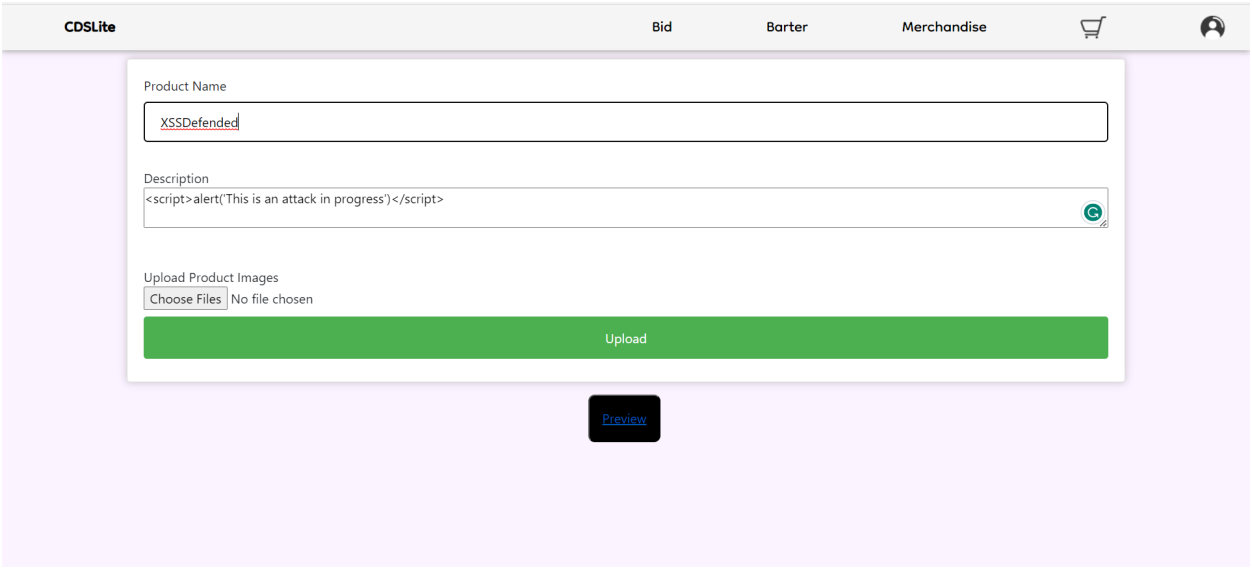

After clicking on preview,



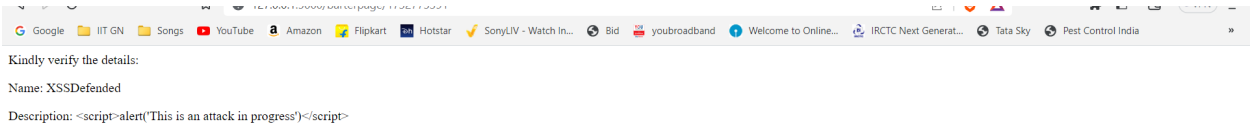

This can cause any user to run a malicious script to manipulate our data. So, we need to defend against this and sanitize the input before storing it in our variable.

***Solution*:**

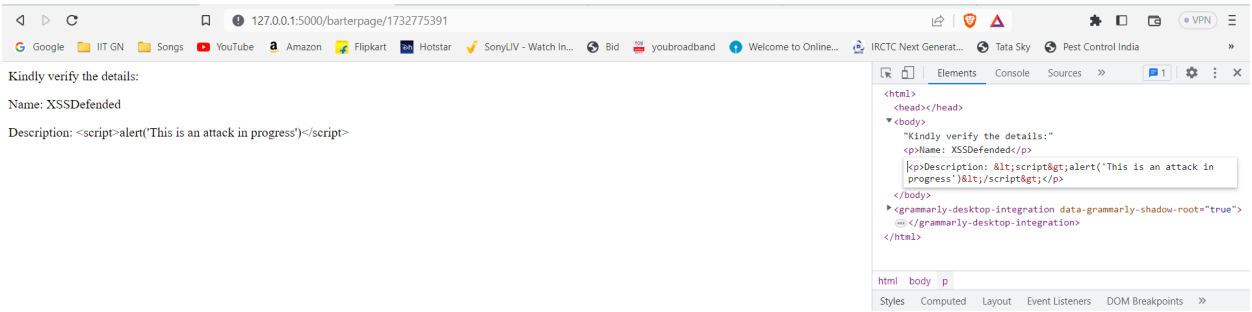
We sanitize the input using the sanitizeInput() function in javascript. This converts the special characters that are identified by javascript as tags to plain text while storing in the variable, which is then rendered on the website at the time of display into their respective symbols.




So, as we can see while previewing, it does show us symbols.

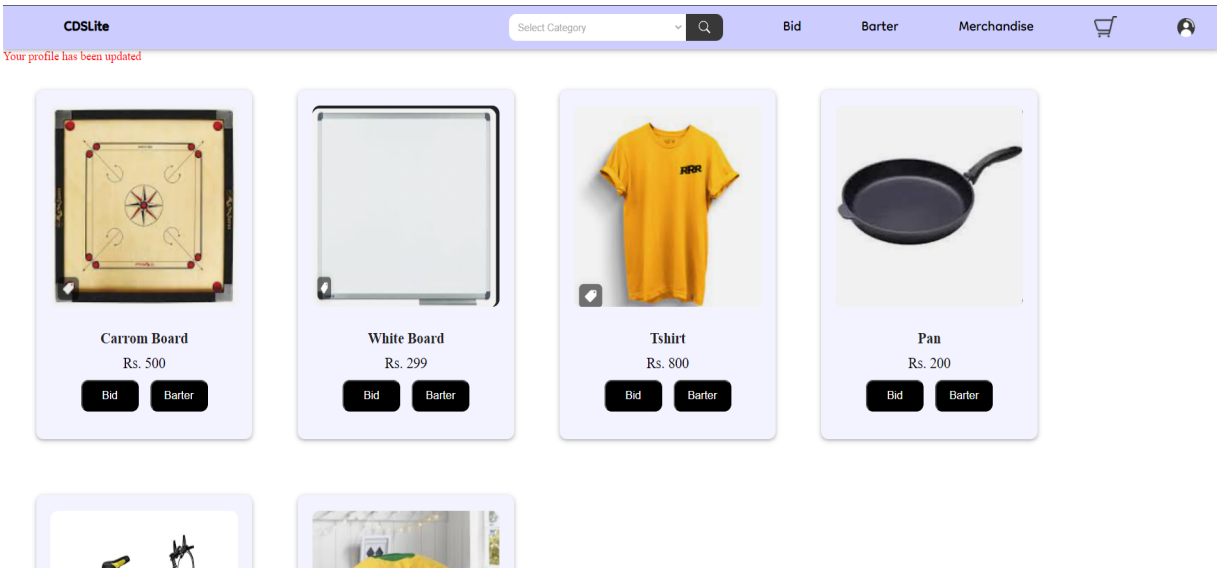


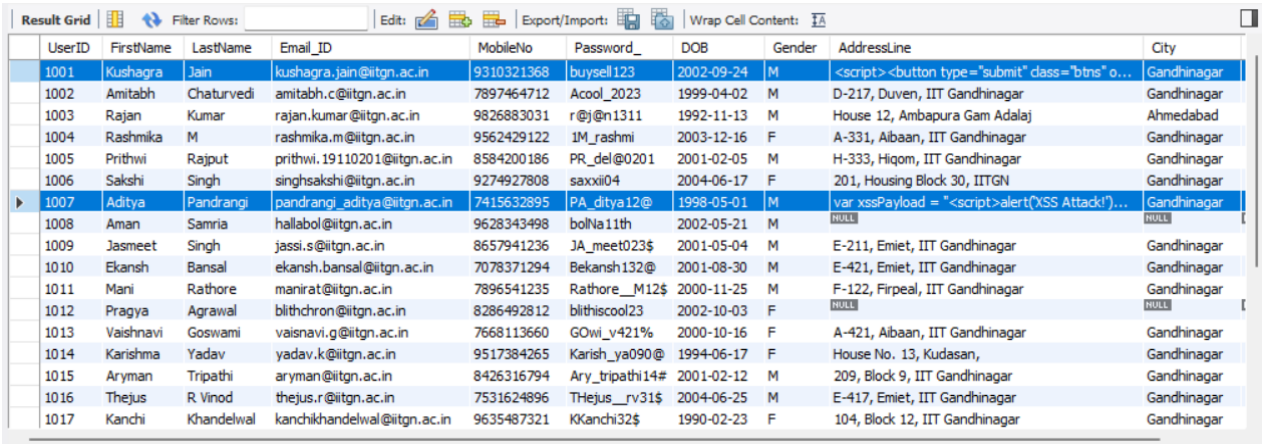

However, if we inspect the website, then we can see that it is stored as &lt, &gt, etc.



**→ XSS Attack while editing the profile details -**


The flash shows that the profile is updated, even on writing the script tag in the address line. We were able to defend against the XSS attack.





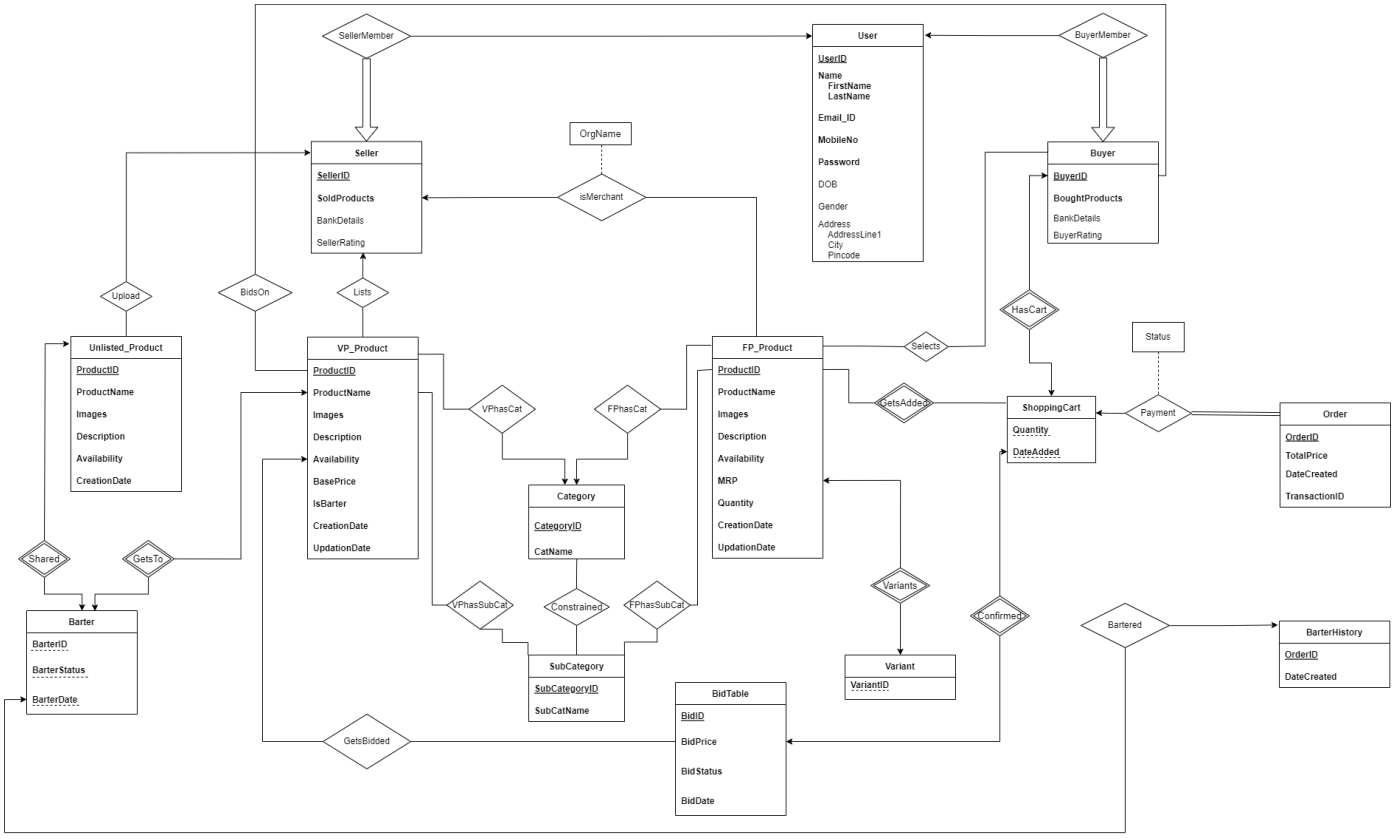
Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

| UserID | FirstName | LastName | Email_ID | MobileNo | Password_ | DOB | Gender | AddressLine | City |
|---|---|---|---|---|---|---|---|---|---|
| 1001 | Kushagra | Jain | kushagra.jain@iitgn.ac.in | 9310321368 | buysell123 | 2002-09-24 | M | <script><button type="submit" class="btns" o... | Gandhinagar |
| 1002 | Amitabh | Chaturvedi | amitabh.c@iitgn.ac.in | 7897464712 | Acool_2023 | 1999-04-02 | M | D-217, Duven, IIT Gandhinagar | Gandhinagar |
| 1003 | Rajan | Kumar | rajan.kumar@iitgn.ac.in | 9826883031 | r@j@n1311 | 1992-11-13 | M | House 12, Ambapura Gam Adalaj | Ahmedabad |
| 1004 | Rashmika | M | rashmika.m@iitgn.ac.in | 9562429122 | 1M_rashmi | 2003-12-16 | F | A-331, Aibaan, IIT Gandhinagar | Gandhinagar |
| 1005 | Prithwi | Rajput | prithwi.19110201@iitgn.ac.in | 8584200186 | PR_del@0201 | 2001-02-05 | M | H-333, Hiqom, IIT Gandhinagar | Gandhinagar |
| 1006 | Sakshi | Singh | singhsakshi@iitgn.ac.in | 9274927808 | saxxii04 | 2004-06-17 | F | 201, Housing Block 30, IITGN | Gandhinagar |
| 1007 | Aditya | Pandrangi | pandrangi_aditya@iitgn.ac.in | 7415632895 | PA_ditya12@ | 1998-05-01 | M | var xssPayload = "<script>alert('XSS Attack!')... | Gandhinagar |
| 1008 | Aman | Samria | hallabol@iitgn.ac.in | 9628343498 | bolNa11th | 2002-05-21 | M | NULL | NULL |
| 1009 | Jasmeet | Singh | jassi.s@iitgn.ac.in | 8657941236 | JA_meet023$ | 2001-05-04 | M | E-211, Emiet, IIT Gandhinagar | Gandhinagar |
| 1010 | Ekansh | Bansal | ekansh.bansal@iitgn.ac.in | 7078371294 | Bekansh132@ | 2001-08-30 | M | E-421, Emiet, IIT Gandhinagar | Gandhinagar |
| 1011 | Mani | Rathore | manirat@iitgn.ac.in | 7896541235 | Rathore__M12$ | 2000-11-25 | M | F-122, Firpeal, IIT Gandhinagar | Gandhinagar |
| 1012 | Pragya | Agrawal | blithchron@iitgn.ac.in | 8286492812 | blithiscool23 | 2002-10-03 | F | NULL | NULL |
| 1013 | Vaishnavi | Goswami | vaisnavi.g@iitgn.ac.in | 7668113660 | GOwi_v421% | 2000-10-16 | F | A-421, Aibaan, IIT Gandhinagar | Gandhinagar |
| 1014 | Karishma | Yadav | yadav.k@iitgn.ac.in | 9517384265 | Karish_ya090@ | 1994-06-17 | F | House No. 13, Kudasan, | Gandhinagar |
| 1015 | Aryman | Tripathi | aryman@iitgn.ac.in | 8426316794 | Ary_tripathi14# | 2001-02-12 | M | 209, Block 9, IIT Gandhinagar | Gandhinagar |
| 1016 | Thejus | R Vinod | thejus.r@iitgn.ac.in | 7531624896 | THejus__rv31$ | 2004-06-25 | M | E-417, Emiet, IIT Gandhinagar | Gandhinagar |
| 1017 | Kanchi | Khandelwal | kanchikhandelwal@iitgn.ac.in | 9635487321 | KKanchi32$ | 1990-02-23 | F | 104, Block 12, IIT Gandhinagar | Gandhinagar |

The highlighted entries show that the malicious scripts have been interpreted as text only and, therefore, have been inserted into the AddressLine attribute in the user relation.

## 2. Relations and Constraints -

Most of the relations and constraints which were finalized after the second feedback was present and valid as per the ER diagram in assignment 1. Although the relations "isMerchant," "BarterHistory," and "Variant" are deleted and thus not present in our database. The relations "products" and "images" were added in the schema; thus, they are not present in the ER diagram. All the existing constraints in the relations are valid as per the ER diagram.




The ER Diagram constructed in Assignment 1

From this ER Diagram, the redundant relationships were removed, including some of the unnecessary entities, namely BarterHistory, and variant. All of the remaining tables hold the constraints as described in the ER diagram. A generalization approach has been followed to remove the redundant relationships, where the primary key of higher-level entities was introduced as foreign keys in the low-level entities.

The current version of the schema contains the following relations described with all of their attributes, constraints, and default values:

| Table | Field | Type | Null | Key | Default | Extra |
|---|---|---|---|---|---|---|
| barter | BarterID | varchar(15) | NO | PRI | NULL | |
| barter | P1ID | varchar(15) | NO | MUL | NULL | |
| barter | P2ID | varchar(15) | NO | MUL | NULL | |
| barter | BarterStatus | enum('Accepted','Declined','Pending') | NO | | NULL | |
| barter | BarterDate | datetime | YES | | CURRENT_TIMESTAMP | DEFAULT_GENERATED |
| bidtable | BidID | varchar(15) | NO | PRI | NULL | |
| bidtable | ProductID | varchar(15) | YES | MUL | NULL | |
| bidtable | UserID | varchar(15) | YES | MUL | NULL | |
| bidtable | BidDate | datetime | YES | | CURRENT_TIMESTAMP | DEFAULT_GENERATED |
| bidtable | BidPrice | int | NO | | NULL | |
| bidtable | BidStatus | enum('Confirmed','Pending','Declined') | YES | | NULL | |
| category | categoryID | varchar(15) | NO | PRI | NULL | |
| category | catName | varchar(40) | NO | | NULL | |
| constrained | CategoryID | varchar(15) | NO | PRI | NULL | |
| constrained | SubCategoryID | varchar(15) | NO | PRI | NULL | |
| fp_products | ProductID | varchar(15) | NO | PRI | NULL | |
| fp_products | ProductName | varchar(30) | NO | | NULL | |
| fp_products | Description_ | longtext | YES | | NULL | |
| fp_products | MRP | int | NO | | NULL | |
| fp_products | Quantity | int | YES | | NULL | |
| fp_products | CreationDate | date | YES | | curdate() | DEFAULT_GENERATED |
| fp_products | UpdationDate | date | YES | | curdate() | DEFAULT_GENERATED |
| fp_products | CategoryID | varchar(15) | NO | MUL | NULL | |
| fphassubcat | ProductID | varchar(15) | NO | PRI | NULL | |
| fphassubcat | SubCategoryID | varchar(15) | NO | PRI | NULL | |

| | | | | | | |
|---|---|---|---|---|---|---|
| image | ProductID | varchar(15) | NO | PRI | NULL | |
| image | Img | longblob | NO | | NULL | |
| order_ | OrderID | varchar(15) | NO | PRI | NULL | |
| order_ | TotalPrice | int | YES | | NULL | |
| order_ | DateCreated | datetime | YES | | CURRENT_TIMESTAMP | DEFAULT_GENERATED |
| order_ | TransactionID | varchar(20) | NO | UNI | NULL | |
| payments | OrderID | varchar(15) | NO | PRI | NULL | |
| payments | UserID | varchar(15) | YES | MUL | NULL | |
| payments | ProductID | varchar(15) | NO | PRI | NULL | |
| payments | Status_ | enum('Successful','Failed') | YES | | NULL | |
| products | ProductID | varchar(15) | NO | PRI | NULL | |
| products | SellerID | varchar(15) | YES | MUL | NULL | |
| seller | SellerID | varchar(15) | NO | PRI | NULL | |
| seller | UserID | varchar(15) | YES | MUL | NULL | |
| seller | SoldProducts | int | YES | | 0 | |
| seller | BankName | varchar(25) | YES | | NULL | |
| seller | AccountNo | char(18) | YES | | NULL | |
| seller | IFSC_Code | char(11) | YES | | NULL | |
| shoppingcart | UserID | varchar(15) | NO | PRI | NULL | |
| shoppingcart | ProductID | varchar(15) | NO | PRI | NULL | |
| shoppingcart | Quantity | int | YES | | NULL | |
| shoppingcart | DateAdded | datetime | YES | | CURRENT_TIMESTAMP | DEFAULT_GENERATED |
| subcategory | SubcategoryID | varchar(15) | NO | PRI | NULL | |
| subcategory | subcatName | varchar(40) | NO | | NULL | |
| unlisted_products | ProductID | varchar(15) | NO | PRI | NULL | |
| unlisted_products | ProductName | varchar(30) | NO | | NULL | |
| unlisted_products | Description_ | longtext | YES | | NULL | |

| unlisted_products | CreationDate | date | YES | | curdate() | DEFAULT_GENERATED |
|---|---|---|---|---|---|---|
| user | UserID | varchar(15) | NO | PRI | NULL | |
| user | FirstName | varchar(25) | NO | | NULL | |
| user | LastName | varchar(25) | YES | | NULL | |
| user | Email_ID | varchar(30) | NO | UNI | NULL | |
| user | MobileNo | char(10) | NO | | NULL | |
| user | Password_ | varchar(30) | NO | | NULL | |
| user | DOB | date | YES | | NULL | |
| user | Gender | enum('M','F') | YES | | NULL | |
| user | AddressLine | varchar(35) | YES | | NULL | |
| user | City | varchar(15) | YES | | NULL | |
| user | PinCode | char(7) | YES | | NULL | |
| vp_products | ProductID | varchar(15) | NO | PRI | NULL | |
| vp_products | ProductName | varchar(30) | NO | | NULL | |
| vp_products | Description_ | longtext | YES | | NULL | |
| vp_products | Availability | enum('Yes','No') | NO | | NULL | |
| vp_products | BasePrice | int | NO | | NULL | |
| vp_products | isBarter | enum('Yes','No') | NO | | NULL | |
| vp_products | CreationDate | date | YES | | curdate() | DEFAULT_GENERATED |
| vp_products | UpdationDate | date | YES | | curdate() | DEFAULT_GENERATED |
| vp_products | CategoryID | varchar(15) | NO | MUL | NULL | |
| vphassubcat | ProductID | varchar(15) | NO | PRI | NULL | |
| vphassubcat | SubCategoryID | varchar(15) | NO | PRI | NULL | |

## Contribution

| Member | Roll Number | Group | Contribution |
|---|---|---|---|
| Vedang Chavan | 20110222 | G1 | ● Assisted in making relevant changes to the database based on the feedback<br>● Worked on revamping frontend based upon initial stakeholder feedback<br>● Assisted and worked on finding potential XSS vulnerabilities and implementing solution |
| Ayush Gupta | 20110031 | G1 | ● Documented stakeholder feedback<br>● Helped in PDF documentation in adding screenshots and adding answers. |
| Ayush Chaudhari | 20110042 | G1 | ● Helped to update frontend based on feedback<br>● Added screenshots and answers in PDF documentation |
| Ekansh Bansal | 20110243 | G1 | ● Assisted in making relevant changes to the database based on the feedback<br>● Worked on revamping front-end based upon initial and final stakeholder feedback |
| Kushagra Jain | 20110099 | G1 | ● Responsible for coordinating with stakeholders and collecting feedback<br>● Assisted in making relevant changes to the database based on the feedback<br>● Helped in documentation |
| Nesar Joshi | 20110120 | G1 | ● Assisted in collecting feedback from stakeholders<br>● Assisted in creating documentation and screenshots for the initial and final feedback. |
| Sakshi Jain | 20110181 | G1 | ● Documented stakeholder feedback<br>● Helped in PDF documentation in adding screenshots and adding answers. |
| Aditya Bhujbal | 20110040 | G2 | ● Assisted and worked on finding potential XSS vulnerabilities and implementing solution<br>● Tested multiple scripts to test implemented solution against XSS<br>● Helped in documentation |

| | | | |
|---|---|---|---|
| Mann Jain | 20110108 | G2 | • Responsible for ensuring concurrent multi-user access to the database and implementing locks in MySQL<br>• Assisted in making relevant changes to the database based on the feedback<br>• Implemented image upload and encoding and storage into MySQL database<br>• Assisted in creating documentation and screenshots for the attacks. |
| Piyush Dhirwani | 20110138 | G2 | • Tested multiple scripts to test implemented solution against SQL injections<br>• Helped in solving upload file issue<br>• Helped in documentation |
| Ronak Kalra | 20110171 | G2 | • Responsible for ensuring concurrent multi-user access to the database and implementing locks in MySQL<br>• Worked on revamping front-end based upon initial and final stakeholder feedback<br>• Worked and assisted in finding vulnerabilities and implementing solution<br>• Assisted in creating documentation and screenshots for the attacks. |
| Siddhesh Kanawade | 20110199 | G2 | • Worked on finding potential vulnerabilities on the platform by implementing SQL injections<br>• Developed potential solution against SQL injection and performed second assessment for same<br>• Assisted in creating documentation and screenshots for the attacks. |