Database Management Systems Project Report - CS 5200
Online Shopping Platform - InstaBuy

Group Name : AvadhaniSAyyaA
Group Members : Anirudh S Ayya
Sai Rakshith Avadhani M S

Introduction:

InstaBuy is a comprehensive database management system designed to streamline the operation of an online shopping platform that specializes in everyday essentials for the city of Boston. It offers a wide range of product categories tailored to meet daily needs.

README:

InstaBuy is built using Python and uses MySQL as its database backend. A python interpreter and a MySQL client and server must be installed on your computer. The application uses the following libraries:

1. Time
    - pip install TIME-python
2. Getpass
    - Part of Python standard library
3. Uuid
    - Pip install uuid
4. Pymysql
    - Pip install pymysql
5. Bcrypt
    - pip install bcrypt
6. Tabulate
    - Pip install tabulate
7. Csv
    - pip install csv
8. Os
    - Standard Python module
9. From decimal import decimal
    - Standard python module
10. Random
    - Part of standard library of python
11. From lxml import btree
    - pip install lxml

12. Requests
    - pip install requests
13. From bs4 import BeautifulSoup
    - pip install beautifulsoup4

To install the libraries, use the command in the terminal - “pip3 install library-name”

The MySQL client, server and the python interpreter can be installed in the default installation directories prompted during installation.

Link to install the latest version of python - https://www.python.org/downloads/release/python-396/
The python version we used to develop this application is **3.9.6**

Link to install the MySQL client - https://dev.mysql.com/downloads/workbench/
Link to install the MySQL server - https://dev.mysql.com/downloads/

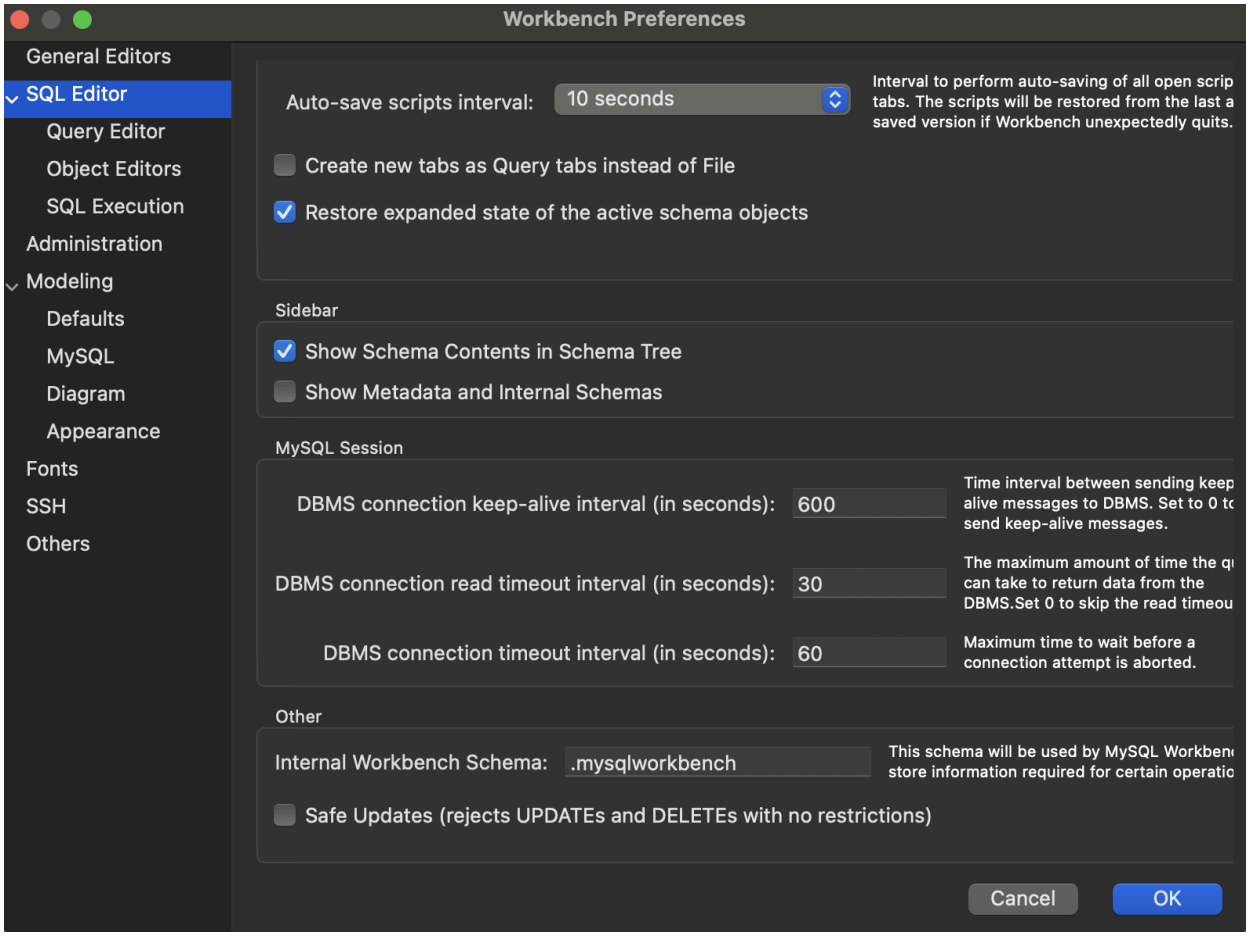
Note: MySQL server must be running at all times.

To run the program, navigate to the directory of the **InstaBuy.py** python file and type in the command - **python3 InstaBuy.py**

Commands:
cd path_to_directory_containing_instaBuy.py
python3 instaBuy.py

NOTE : PLEASE UNCHECK THE CHECKBOX “Safe Updates” UNDER THE “SQL Editor” OPTION IN SETTINGS BEFORE RUNNING THE APPLICATION.

PLEASE RESTART YOUR MYSQL CLIENT AND SERVER AFTER DOING THIS.



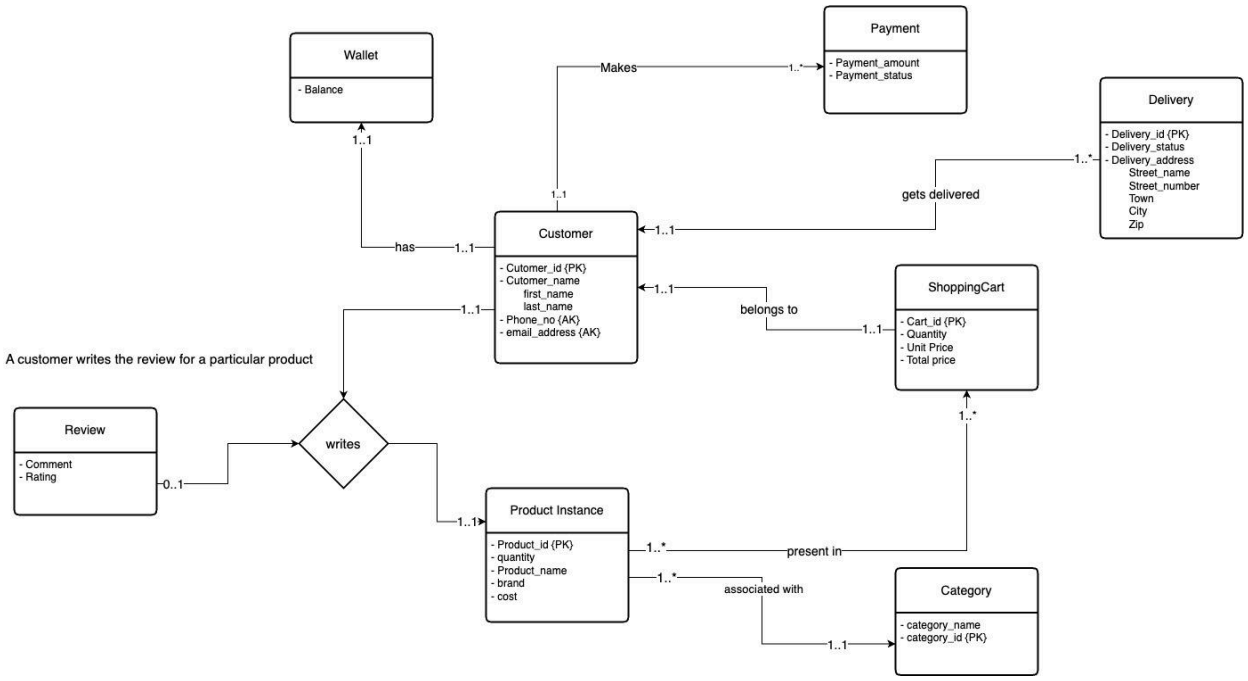
Note : To login to the admin account please create an account with the username "**admin**". The password can be given of your choice.

Technical specifications:

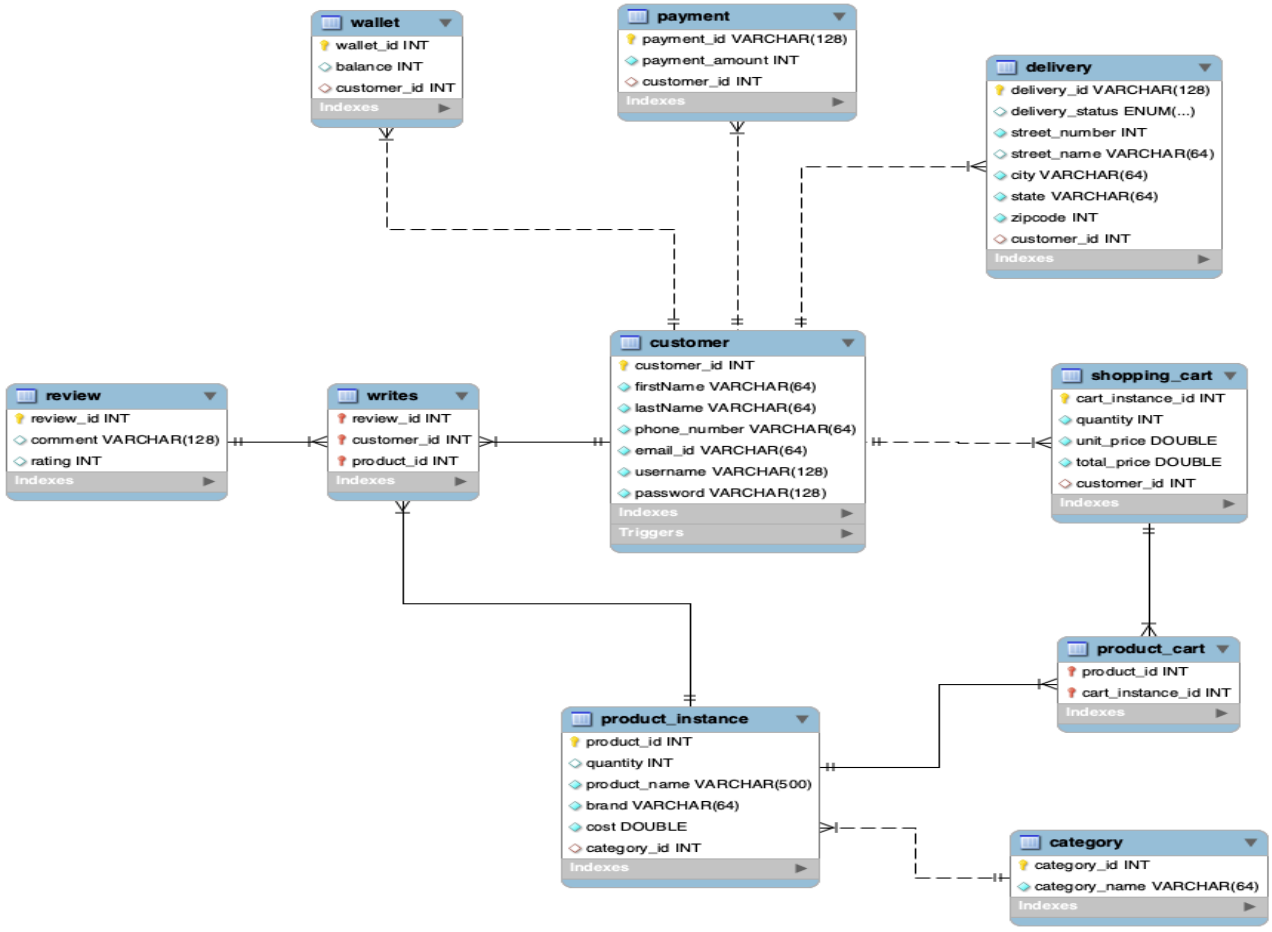
InstaBuy is built using Python and uses MySQL as its database backend.
To run the program, navigate to the directory of the **InstaBuy.py** python file and type in the command - **python3 InstaBuy.py**

Commands:
cd path_to_directory_containing_instaBuy.py
python3 instaBuy.py

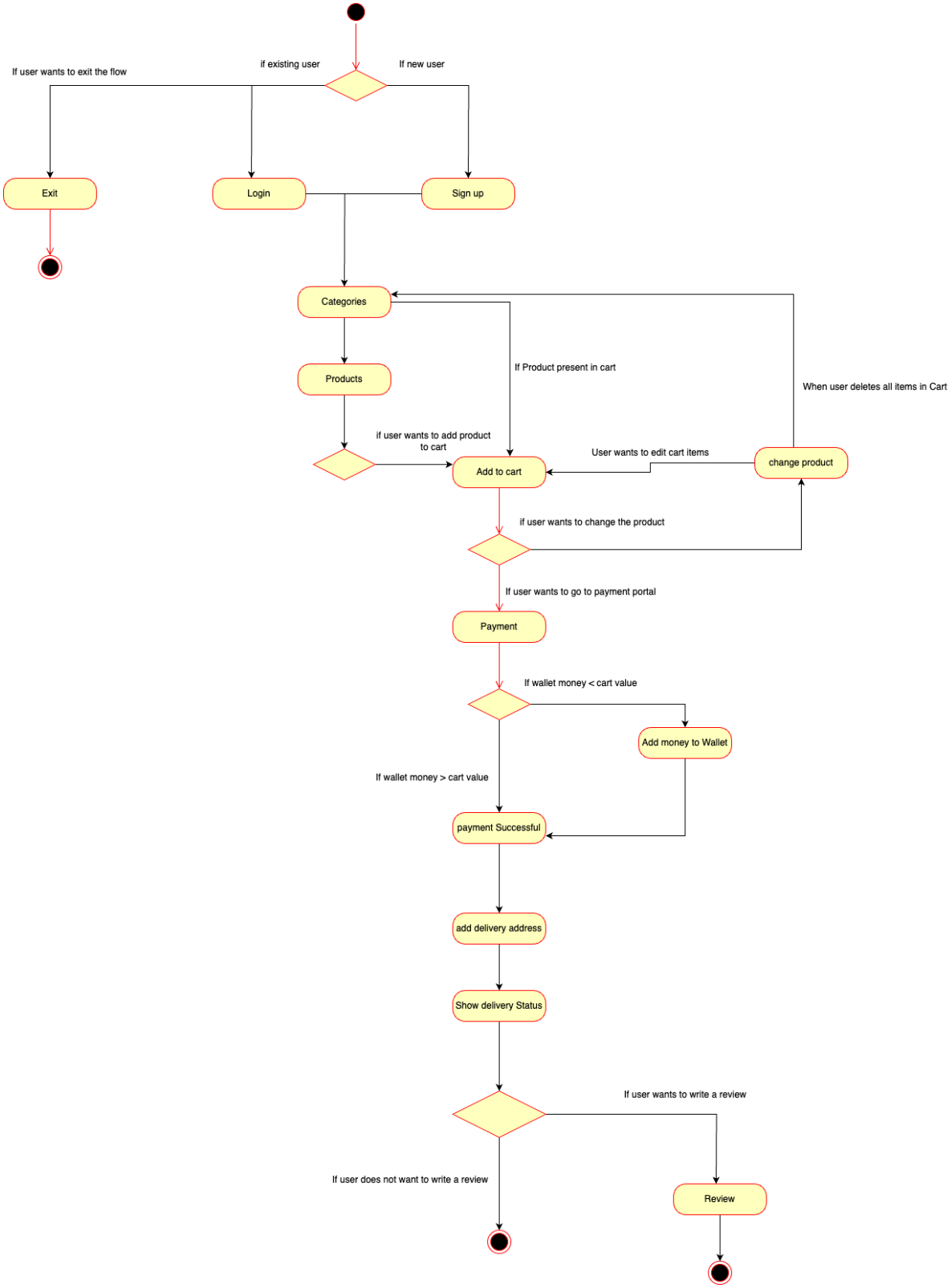
Conceptual Design:



Logical Design:



User Flow :


If user wants to exit the flow
if existing user
If new user
Exit
Login
Sign up
Categories
Products
If Product present in cart
When user deletes all items in Cart
if user wants to add product to cart
User wants to edit cart items
Add to cart
change product
if user wants to change the product
If user wants to go to payment portal
Payment
If wallet money < cart value
Add money to Wallet
If wallet money > cart value
payment Successful
add delivery address
Show delivery Status
If user wants to write a review
If user does not want to write a review
Review

List of commands :

The user **performs** the following commands and the respective methods are as follows:

1. Create account [create_account()]

2. Login [login()]

3. Chooses a category to select the products [displayCategoriesList()]

4. View products to add to cart from the chosen category [getProducts(category)]

5. View current cart items [fetchCartItems()]

6. View reviews for a product [showReviews(product)]

7. Enters the product to be added to cart and its quantity
   [addProductToCart(product_id, quantity)]

8. Updates the quantity of an existing cart item (if needed)
   [updateProductToCart(product, quantity)]

9. Deletes the item from the cart (if needed) [deleteCartItems(product)]

10. Moves to payment [checkWalletMoney(user), fetchTotalCartValue() happens in backend]

11. If wallet money is less than cart value, user is prompted to add money to wallet [UpdateWalletBalance(user, amount)

12. Adds delivery details [add_delivery_address()]

13. Writes reviews for purchased products (if needed) [writeReview(product)]

Lessons learned:

1. Technical expertise gained:
   a. Database design - Conceptual design using UML notation.
   b. Python - Database connection using pymysql, calling and making use of procedures, functions and SQL queries from the application.

c. MySQL - Creating tables using foreign keys, NOT NULL and UNIQUE constraints, relationship tables. Creating new procedures, functions and triggers.
d. Web scraping using BeautifulSoup and requests.

2.

a. Time management insights:
    i. We invested time upfront in designing the database schema and frontend architecture so that we don't have to make time-consuming revisions later.

    ii. We broke up the project into smaller milestones so that it is easier to track progress and allocate time effectively.

    iii. As database and frontend are interdependent, we worked on tasks in parallel where possible.

    iv. We followed a bottom up approach where we created the tables and procedures for the database before we wrote the frontend application calls. This saved us time while we developed the frontend application as we were already aware of what procedure had to be called for a particular functionality.

b. Data domain insights:
    i. In terms of data identification, we identified different domains of data such as
        1. Customer domain: User details, login credentials, addresses.
        2. Product domain: Product details, categories.
        3. Order domain: payments, delivery.
        4. Review domain: Customer review and ratings.

    ii. We gained insights on the data relationships between the different entities. Example, a shopping cart belongs to one customer and many products are present in a shopping cart.

    iii. We identified the constraints on the items in the data domains such as primary keys, foreign keys, unique and not null constraints. Example, each customer has a unique customer id.

iv. Through this project, we also gained data validation insights such as customer validation during login, storing the customer's password as a hashed value in the database, checking if the category and the product is present in the catalog before adding it to the cart, checking if enough balance is present in the wallet before processing the payment etc.

3. Alternative design/approaches :
    i. We had contemplated maintaining the total quantity of products in the shopping cart along with the cart id. The mapping between the product, customer and the cart would be maintained in the relationship table. But this doesn't allow us to identify the product uniquely for a customer. So in order to incorporate the contemplated changes we made a relationship from the customer to the cart and another relationship from the product to the cart table where the cart is uniquely identified by the cart id which is again unique to a particular customer.

Future Work

a. Planned uses of the database -
    ii. We plan on using the database for providing some comprehensive reports on Customer behaviour, customer trends and product analytics and try to give more insights on how the demand for a particular product changes over time.
    iii. We plan on integrating with other external systems like a CRM model to enhance the functionality and provide more varied data.

b. Potential areas for added functionality -
    iv. We plan on adding the orders table and try to build a strong and robust order management system which provides more support.
    v. Incorporate the GUI for the database thereby enhancing the user experience.
    vi. Integrate with AI by helping in recommending products to users based on their past purchases made on the application.
    vii. Can enhance to handle concurrent user demands by making use of sharding.