Database Design And Implementation For E-Commerce

Sourabh Aggarwal (111601025), Nikhil Kumar Yadav (111601013)

February 11, 2019

Contents

1 Contribution

As of now, we did everything together.

2 Introduction

2.1 Requirements

Following is the list of requirements.

1. Company maintains the details of stock like their id, name, quantity, rating etc.
2. Company maintains the details of users like their id, name, address, phone number, ewallet.
3. Only users which have purchased the product can leave rating and review to product, they can also give rating to the seller. Thus users should also be able to see their past puchases.
4. Users can add balance to their ewallet.
5. Company maintains the details of its suppliers like their id, name, address, phone number and rating. Each supplier has at least some stock for some item. (Suppliers can add new product (stock) and mention its quantity which he/she has.)
6. When users browses for a product, suppliers will be listed based on the quantity user wants.

3 Entity Relation Diagram

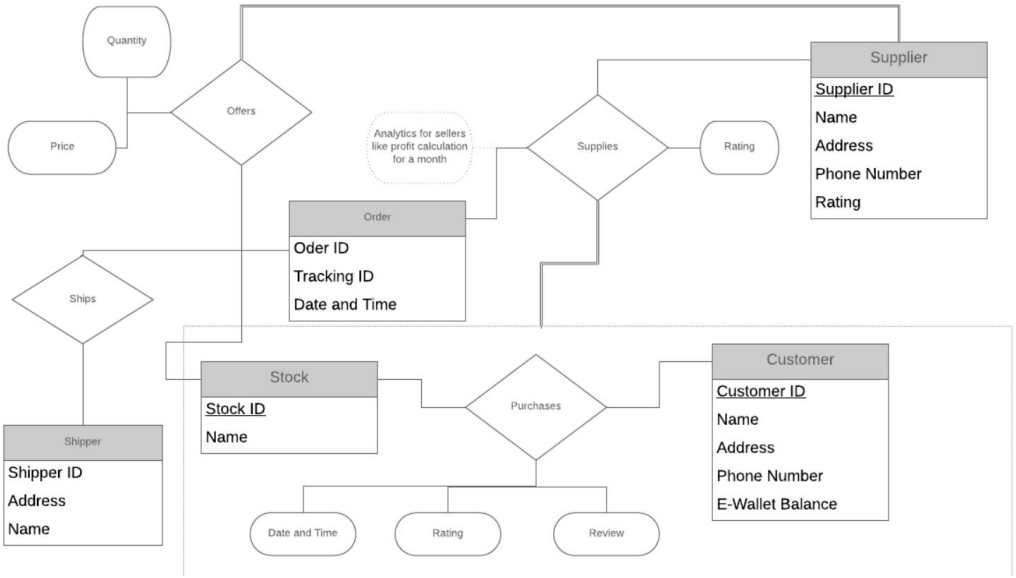

Figure 1: Entity Relationship Diagram for e-commerce

4 Database Schema

```
create table Users (
  username VARCHAR (20) primary key not null,
  passcode VARCHAR (20) not null,
  role VARCHAR (20) not null
);

create table customer (
  customer_id VARCHAR (20) primary key not null,
  name VARCHAR (20) not null,
  address VARCHAR (60) not null,
  phone_number DECIMAL (10) UNSIGNED not null,
  email_id VARCHAR (20) not null,
  foreign key (customer_id) references Users(username) on
  delete cascade
);

create table payment (
```

```
  payment_id VARCHAR (20) primary key not null,
  credit_card_number VARCHAR (20) not null,
  date_ timestamp,
  billing_address varchar(60) not null
);

create table order_ (
  order_id VARCHAR (20) primary key not null,
  customer_id VARCHAR (20),
  shipping_address varchar(60) not null,
  payment_id VARCHAR (20),
  foreign key (customer_id) references customer (customer_id)
  on delete set null,
  foreign key (payment_id) references payment (payment_id) on
  delete set null
);

create table supplier (
  supplier_id varchar (20) primary key not null,
  name varchar (20) not null,
  address varchar (60) not null,
  phone_number decimal (10) UNSIGNED NOT NULL,
  email_id VARCHAR (20) not null,
  foreign key (supplier_id) references Users(username) on
  delete cascade
);

create table shipper (
  shipper_id varchar (20) primary key not null,
  name varchar (20) not null,
  head_quarters varchar (60) not null,
  phone_number decimal (10) UNSIGNED not null,
  email_id VARCHAR (20) not null,
  foreign key (shipper_id) references Users(username) on
  delete cascade
);

create table track (
  index_ INT AUTO_INCREMENT primary key not null,
  shipper_id varchar (20),
  tracking_id varchar (20),
```

```
  foreign key (shipper_id) references shipper (shipper_id) on
  delete set null
);

create table product (
  product_id varchar (20) not null,
  supplier_id varchar (20) not null,
  price float not NULL,
  total_stock int,
  description varchar (60),
  foreign key (supplier_id) references supplier (supplier_id)
  on delete cascade,
  primary key (product_id, supplier_id)
);

create table product_order (
  product_id varchar(20) not null,
  order_id varchar (20) not null,
  supplier_id varchar (20),
  product_rating int check (product_rating in (1, 2, 3, 4,
  5)),
  supplier_rating int check (supplier_rating in (1, 2, 3, 4,
  5)),
  ship_index int,
  product_review varchar (60),
  supplier_review varchar (60),
  quantity int,
  primary key (product_id, order_id),
  foreign key (product_id) references product (product_id) on
  delete cascade,
  foreign key (order_id) references order_ (order_id) on
  delete cascade,
  foreign key (supplier_id) references supplier (supplier_id)
  on delete set null,
  foreign key (ship_index) references track (index_) on
  delete set null
);
```

5 Roles, Triggers, Views

5.1 Views

- A view to allow a customer to check his personal previous orders/order history.
- A view to check the current status of a particular package.
- A view to check the spendings done by the customer per month.
- A view for supplier to check his pending (not shipped) packages.
- A view for supplier to check his previously processed packages.
- A view for supplier to check various important statistics like sale per month.
- A view for shipper to check his not yet delivered packages.
- A view for shipper to know his past delivered packages.
- A view for shipper to know various statistics like sales per month, sales associated with particular supplier, etc.

5.2 Roles

- A role for database administrator.
- A role for customer.
- A role for supplier.
- A role for shipper.

5.3 Triggers

- Trigger to notify addition of a new customer.
- Trigger to notify addition of a new supplier.
- Trigger to notify addition of a new shipper.
- Trigger to notify addition of new item.

- A trigger to add a tuple in track relation before an insertion into *product_order* relation.
- Trigger to notify cutomer of successful order.
- Trigger to notify supplier about successful dispatch.
- Trigger to notify when the stock goes below a specific amount.
- Trigger to delete corresponding entries in various tables if the stock decreases to 0.
- Trigger to notify customer that the package has been delivered.
- Trigger to notify supplier that the package has been recieved.
- Trigger to notify supplier that a customer has left a review.

6 Use Cases

Lets for this case assume that we a supplier. We first want to register and then add products to sell.

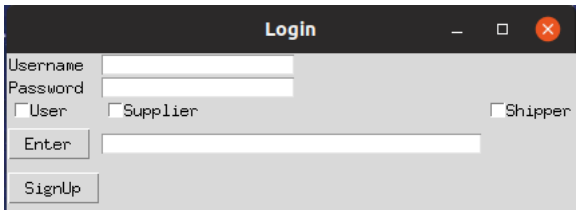

Figure 2: Login page.

So at the login page we will select SignUp. It will then open up the SignUp page.

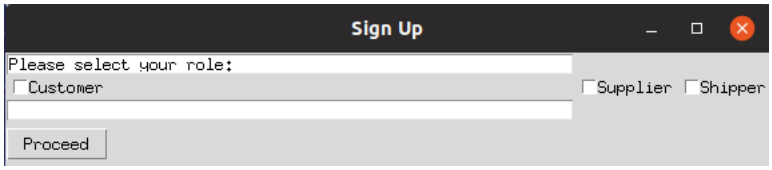

Figure 3: SignUp page(Selecting role).

Since we are Supplier we will select supplier option and then proceed.

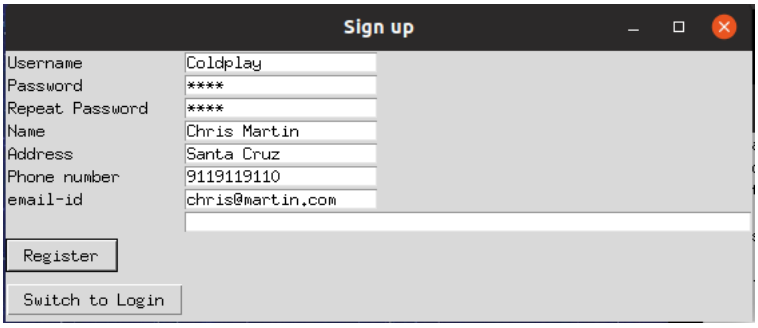

Figure 4: SignUp page(Entering Details).

After entering the details press the Register button. It will display that the user is successfully created.

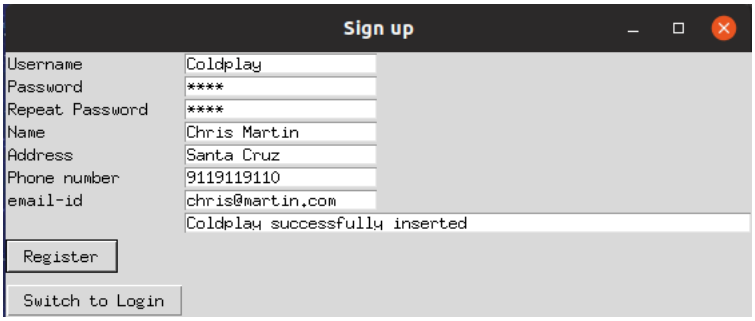

Figure 5: SignUp page(Conformation).

After registering goto the login page by pressing the Switch to Login button.

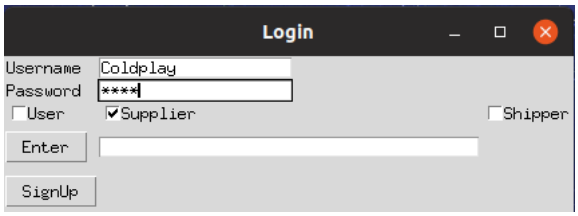

Figure 6: Login page(Entering Details).

After login you will be taken to a welcome page which will have two options for you

- Either to add new products in the market.
- Or to change the quantites or price of existing products.

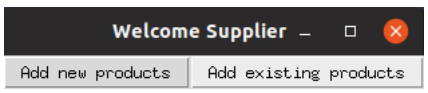

Figure 7: Welcome page for Supplier.

So lets add a new product. You will be taken to a page which will ask to fill the details of the product.

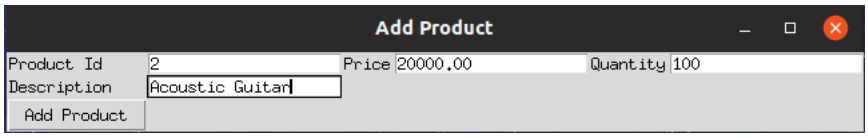

Figure 8: Add new product.

Now the product has been added successfully.

7 Useful links

Hosted on github with love - Link
GUI source code - Link

8 Logs

For Developers Use only

- $11^{th} Feb 2019$ -

  1. Table Schema Modified.
  2. Basic Application Interface Completed.
  3. Added Use Cases in Report.
  4. And yes we are calling it AmaKart.