

# Database Design And Implementation For E-Commerce

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

March 28, 2019

## Contents

<b>1</b>	<b>Contribution</b>	<b>1</b>
<b>2</b>	<b>Introduction</b>	<b>1</b>
<b>3</b>	<b>Requirements</b>	<b>1</b>
<b>4</b>	<b>Entity Relation Diagram</b>	<b>3</b>
4.1	Enhanced ERD . . . . .	3
4.2	Simple ERD . . . . .	5
<b>5</b>	<b>Database Schema And Normalization</b>	<b>5</b>
<b>6</b>	<b>Roles, Views and Triggers</b>	<b>10</b>
6.1	Roles . . . . .	10
6.2	Views . . . . .	10
6.2.1	Customer Views . . . . .	10
6.2.2	Seller Views . . . . .	11
6.2.3	Shipper Views . . . . .	11
6.3	Triggers . . . . .	11
<b>7</b>	<b>Functions And Procedures</b>	<b>13</b>
7.1	Customer Procedures . . . . .	13
7.2	Seller Procedures . . . . .	15
7.3	Shipper Procedures . . . . .	16
<b>8</b>	<b>Tour Of GUI And Use Cases</b>	<b>16</b>

8.1	Login Window . . . . .	16
8.2	Sign Up . . . . .	17
8.3	Customer Window . . . . .	18
8.3.1	Previous Products . . . . .	18
8.3.2	Browse Products . . . . .	19
8.3.3	Cart . . . . .	20
8.3.4	Update Your Info . . . . .	20
8.3.5	View Your Profile . . . . .	20
8.4	Seller Window . . . . .	21
8.4.1	Add New Products . . . . .	21
8.4.2	Update Existing Product . . . . .	22
8.4.3	See Your Products . . . . .	22
8.4.4	See Past Sellings . . . . .	23
8.4.5	See Earnings Between Duration . . . . .	24
8.4.6	See Your Rating . . . . .	24
8.4.7	See Products Which Are Sold But Not Shipped . . . . .	24
8.4.8	Browse Shippers . . . . .	25
8.4.9	Update Your Info . . . . .	25
8.4.10	View Your Profile . . . . .	26
8.5	Shipper Window . . . . .	26
8.5.1	Update Your Info . . . . .	26
8.5.2	See Past Shipments . . . . .	26
8.5.3	View Your Profile . . . . .	27

## 9 Further Reading And Useful Links 29

# 1 Contribution

We both (Nikhil and Sourabh) agree with each other that we have contributed equally to this project.

We both sat and discussed the project outline and we divided the work among us equally. We regularly consulted each other throughout the development phase.

# 2 Introduction

In this project, our aim was to come up with a reasonably scalable database along with basic GUI for E-Commerce purpose.

We started by listing down various requirements presented in the next section. After that we proceeded on building an ER Diagram to fulfil the same. After that it was time to implement all this in SQL. In due time, we managed to put various important features provided by almost all E-Commerce site in our project.

So the following report touches on each of these aspects in brief and sequential manner.

# 3 Requirements

Following is the list of requirements.

1. Idea of roles. Model should have role for each of Customer, Seller and Shipper. Each user is thus assigned its role.
2. Company maintains the details of its users like their name, address, phone number and email-id.
3. Each user has security credentials like their username and password.
4. It should be possible for each user to update their information including their password.
5. Customer requirements
  - (a) Customers should be able to browse all products.

- (b) They should be able to add those products in their cart. Thus each Customer should possess a cart into which they can add products for purchasal.
- (c) Note that each product could be sold by different sellers, so customer should be able to select a product corresponding to the seller they wished to buy from.
- (d) Once all the items are added in cart, Customer should then be able to purchase all those items in cart at once.
- (e) For each of the product the Customer has bought, it should be possible for customer to pass the rating and review for both the product as well as seller.

#### 6. Seller requirements

- (a) Each Seller possesses a rating which they should be able to see. This rating is given and updated by the users of role Customer who purchased the product from Seller.
- (b) Seller should list down the products they are willing to sell. Each product should have:
  - i. Product Name
  - ii. Product Image
  - iii. Price
  - iv. Quantity which the seller possess of the product
  - v. Pickup address
  - vi. Description
  - vii. Rating (Product rating is again updated by the users of role Customer)
- (c) It should be possible for seller to add or update their products.
- (d) It should be possible for seller to see their past sellings.
- (e) It should be possible for seller to see the products which they has listed.
- (f) It should be possible for seller to see their earnings in a specific duration.

- (g) Seller should be able to browse Shippers so that they can decide and contact various Shippers.
  - (h) It should be possible for seller to see the latest purchases of their products which are pending to be shipped, thus seller should also have an interface to update customer with the sold product's shipment Tracking ID.
7. Shipper requirements
- (a) It should be possible for shipper to see their past shipments (using our interface) including source and destination of the product.

## 4 Entity Relation Diagram

### 4.1 Enhanced ERD

Enhanced Entity Relation Diagram can be represented in various notations, below shows the notation which we have followed.

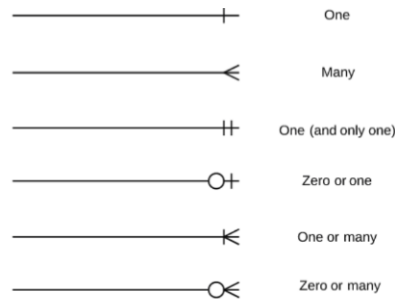


Figure 1: Entity Relation Diagram Notation

Figure 1: Entity Relation Diagram Notation

Following this notation, our ERD is represented below.

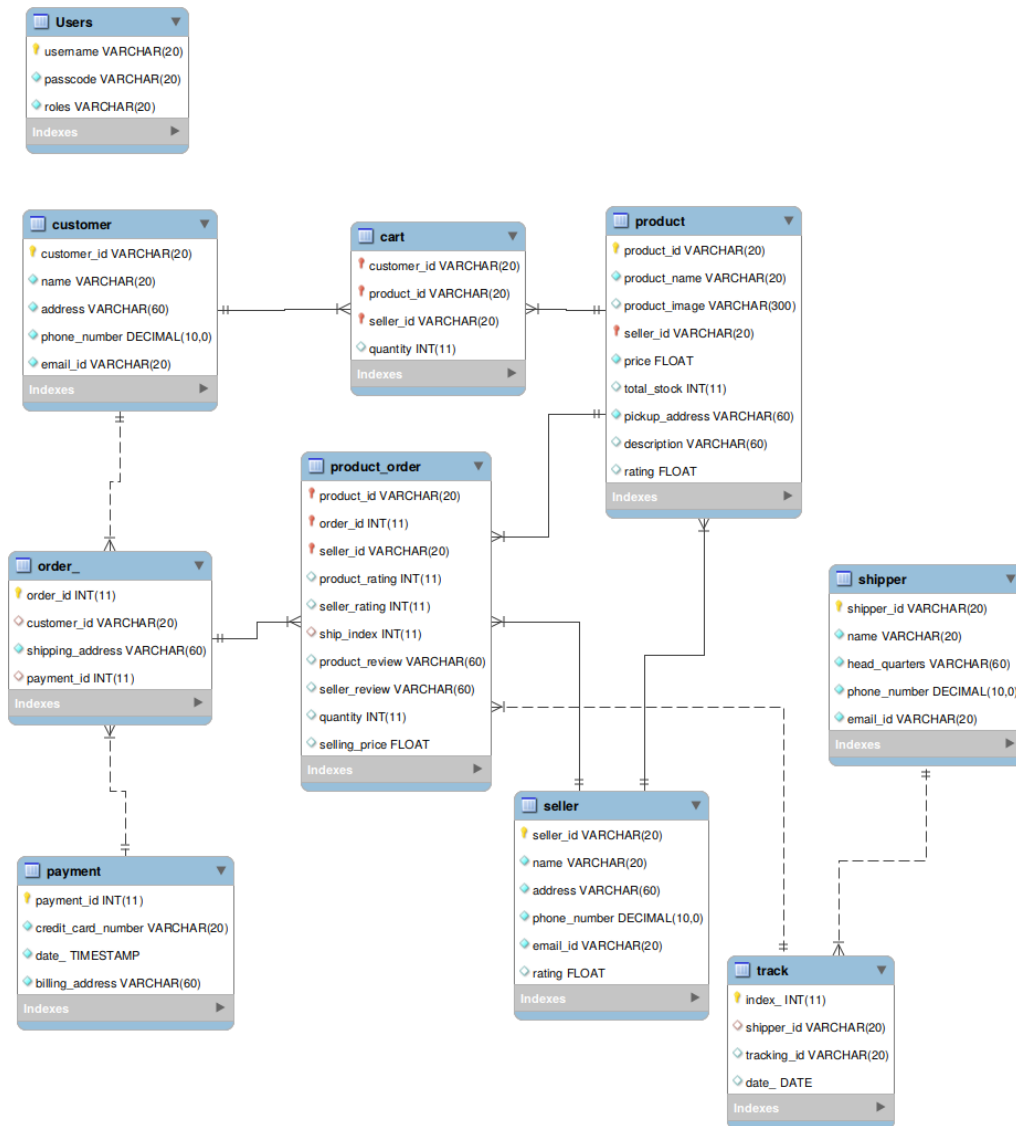


Figure 2: Entity Relationship Diagram for e-commerce

*Note* : The dotted line is to show that the foreign key used to draw that line is not primary key (or part of primary key) in one of the two tables.

## 4.2 Simple ERD

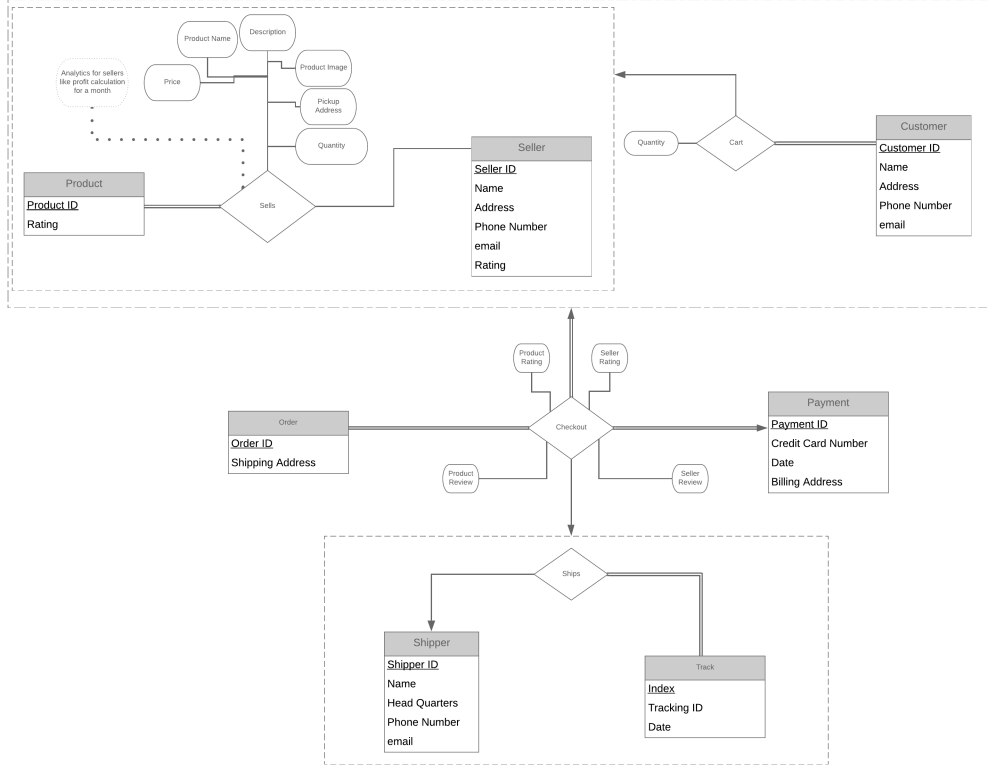


Figure 3: Simple Entity Relationship Diagram for e-commerce

## 5 Database Schema And Normalization

In this section we describe various aspects of our schema and also mention that it is indeed normalized (BCNF). All sections except this don't have code but we feel it is necessary to include code in this section for better understanding of various relations.

*Notation:* A dependency  $A \rightarrow B$  is called relevant if all other dependencies from  $A$  are of the form  $A \rightarrow C$  where  $C \subseteq B$ .

- A table for basic details of customer.

*/\* Here: customer\_id  $\rightarrow$  R is the only relevant dependency and hence it is in BCNF \*/*

```
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
);
```

- A table for basic details of seller.

```
/* Here: seller_id -> R is the only relevant dependency
and hence it is in BCNF */
/* Rating will be updated with the help of triggers. */
create table seller (
    seller_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,
    rating float
);
```

- A table for basic details of shipper.

```
/* Here: shipper_id -> R is the only relevant dependency
and hence it is in BCNF */
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
);
```

- Having described these basic tables, we can now describe table for products. Note that each product can be sold by different sellers in different price and quantity, thus, primary key is formed by both product\_id and seller\_id.

```
/* Here: (product_id, seller_id) -> R is the only
relevant dependency and hence it is in BCNF */
/* Rating will be updated with the help of triggers. */
create table product (
    product_id varchar (20) not null,
```



```

product_name varchar (20) not null,
product_image varchar(300),
seller_id varchar (20) not null,
price float not NULL,
total_stock int,
pickup_address varchar (60) not null,
description varchar (60),
rating float,
foreign key (seller_id) references seller (seller_id) on
delete cascade,
primary key (product_id, seller_id)
);

```

- When user makes a payment, we want to store payment details for which we have the following table.

```

/* Here: payment_id -> R is the only relevant
dependency and hence it is in BCNF */
create table payment (
    payment_id int AUTO_INCREMENT primary key not null,
    credit_card_number VARCHAR (20) not null,
    date_ timestamp,
    billing_address varchar(60) not null
);

```

- User will have a front end feature to add items in cart. When the user is ready to buy, it will generate an *order\_id* for all those products which they chose. Note that *order\_id* will be generated *only* when user successfully does the payment.

```

/* Here: order_id -> R is the only relevant dependency
and hence it is in BCNF */
/* Initially payment id can be null and then later once
the customer does the payment, trigger will add the
payment id */
create table order_ (
    order_id int AUTO_INCREMENT primary key not null,
    customer_id VARCHAR (20),
    shipping_address varchar(60) not null,
    payment_id int,
    foreign key (customer_id) references customer
(customer_id) on delete set null,

```

```

    foreign key (payment_id) references payment (payment_id)
    on delete set null
);

```

- After generating the *order\_id* (by successful completion of payment), we have to put the details of the bought items along with their *order\_id*.

```

/* Here: (product_id, order_id, seller_id) -> R is the
only relevant dependency and hence it is in BCNF */
create table product_order (
    product_id varchar(20) not null,
    order_id int not null,
    seller_id varchar (20),
    product_rating int check (product_rating in (NULL, 1, 2,
3, 4, 5)),
    seller_rating int check (seller_rating in (NULL, 1, 2,
3, 4, 5)),
    ship_index int,
    product_review varchar (60),
    seller_review varchar (60),
    quantity int,
    selling_price float,
    primary key (product_id, order_id, seller_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 (seller_id) references seller (seller_id) on
delete cascade,
    foreign key (ship_index) references track (index_) on
delete set null
);

```

- Note that we used a foreign key in the above table which we haven't defined yet, which is *ship\_index*. It is basically a unique identifier for each ordered product serving as an index of track table which we will use to track our items.

```

/* Here: index_ -> R is the only relevant dependency and
hence it is in BCNF */
create table track (
    index_ INT AUTO_INCREMENT primary key not null,

```

```

    shipper_id varchar (20),
    tracking_id varchar (20),
    date_ DATE,
    foreign key (shipper_id) references shipper (shipper_id)
    on delete set null
);

```

- A relation for *cart* .

```

/* Here: (customer_id, product_id, seller_id) -> R is
the only relevant dependency and hence it is in BCNF */
create table cart (
    customer_id varchar(20),
    product_id varchar(20),
    seller_id varchar(20),
    quantity int,
    primary key (customer_id,product_id,seller_id),
    foreign key (customer_id) references
customer(customer_id) on delete cascade,
    foreign key (product_id,seller_id) references
product(product_id,seller_id) on delete cascade
);

```

- We also have an auxiliary table for keeping track of users with their old passwords as mysql.user encrypts the passwords and there is no way to get it back also this table is required for validation when the user tries to log in the system.

```

-/* Here: username -> R is the only relevant dependency
and hence it is in BCNF */
create table Users (
    username VARCHAR (20) primary key not null,
    passcode VARCHAR (20) not null,
    roles VARCHAR (20) not null
);

```

## 6 Roles, Views and Triggers

### 6.1 Roles

As mentioned before, we have three roles, viz., Customer, Seller and Shipper. And of course above them all we have database administrator, "root", viz. "dbadmin". Each role will be able to access views, functions, procedures written for it and granted to it.

### 6.2 Views

Almost all what we wanted to achieve was possible with the help of procedures, views, thus have little to offer but still we are listing here all the views that we wrote irrespective of whether they are actually part of our GUI.

#### 6.2.1 Customer Views

1. View that will allow customer to view its profile, "customerProfile".
2. View that will allow customer to see the total cost of his/her various orders, i.e. total money spent on the platform till date, "orderPrice".
3. View that will allow customer to see and add products to his cart, "showCart".
4. View that will tell the customer details corresponding to his/her all order\_id mentioning complete order details (order\_id, shipping\_address, date\_, total\_price) except the products in that order, "previousOrders".
5. View that will allow customer to just see his various order\_id, "listOrders".
6. View that will give us shipment index (ship\_index), order ID, product ID from product\_order relation corresponding to our orders, "trackID".
7. View that will augment the above view with tracking ID as well, "packageStatus".

### 6.2.2 Seller Views

1. View that will allow seller to view its details, "sellerProfile".
2. View that will allow seller to see his/her various products, "sellerProducts".
3. View that will allow seller to see various orders which they have sold (seller\_id, product\_id, quantity, selling\_price, date\_), "sellerOrders".

### 6.2.3 Shipper Views

1. View that will allow shipper to view its details, "shipperProfile".
2. View that will allow shipping details (pickup\_address, shipping\_address, tracking\_id), "shipperTrack".

And their details is best understood with the help of the following code:

## 6.3 Triggers

1. When a product is sold, we want to mention its selling\_price as later the seller can update the price, "setPrice". (sets NEW.selling\_price entry in product\_order).
2. When a customer passes a rating for product we have to update it in our product table, "updateRatingProduct" (update is done by averaging over all ratings).
3. When a customer passes a rating for seller we have to update it in our seller table, "updateRatingSeller" (update is naturally done in "seller" relation by averaging over all ratings passed to our seller).
4. When a product is sold, we need to add an entry to our track table for the same and also we have to reduct the total\_stock of that product offered by that seller. Both of these are done together by our trigger, "stockCheckandaddTrack" (naturally it will have shipper\_id, tracking\_id field NULL as of now and this info will be used to determine whether the product has been shipped or not)
5. To check whether the products added in cart are valid or not. In case it is valid we proceed else we raise an exception. Trigger name "validCartinsert".

6. Same as above but for update operation on cart, "validCartupdate".

## 7 Functions And Procedures

Below you can see details description and implementation of various Procedures and Functions, due to the nature of operations mentioned in our requirements we have preferred procedures in most cases over functions, thus if we have used function, we explicitly mention it.

### 7.1 Customer Procedures

1. Procedure to see purchases between some duration, "seePurchases-ByDate(IN startTime TIMESTAMP, IN endTime TIMESTAMP)" -> Will return complete details from order, payment, product and product\_order table.
2. Procedure to see items in cart, "getProductsFromCart ()" -> which selects everything from view "showCart".
3. Procedure for customer to checkout items present in cart, "purchaseEverythingInCart(IN oid varchar(20))" -> which takes in order id and then for each product present in cart, will add the corresponding entry to product\_order table.
4. Procedure for customer to remove product from cart, "removeProduct-Cart(IN pid varchar(20), IN sid varchar(20))" -> which takes in that products product\_id and seller\_id and thus delete such product from cart (deletion is performed using view showCart).
5. Procedure for customer to update a product in cart (i.e. change quantity of the chosen product), "CREATE PROCEDURE updateProduct-Cart(IN pid varchar(20), IN sid varchar(20), IN N INT)".
6. Procedure for customer to make an order, "makeorder(IN cnum varchar(20), IN badd varchar(20), IN cid varchar(20), IN sadd varchar(20))" which takes a credit card number, "cnum", billing address, "badd", customer ID, "cid" and a shipping address, "sadd" and first adds an entry into payment table by selecting date using "NOW()" function and then since payment table had payment\_id which was set to automatic increment so we after fetching it insert the corresponding entries into order\_table and then again by fetching order\_id as it was auto increment, we call already discussed procedure, "purchaseEverythingInCart".

7. Procedure to add product to cart, "addProductToCart(IN cid varchar(20), IN pid varchar(20), IN sid varchar(20), IN q int)" which will decide whether to update the quantity if the product is already there in cart, or to add this new record into our cart.
8. Procedure to see latest N purchases, "seeLatestNPurchases(IN N INT)" which takes in N and then gives rows corresponding to payment, order\_, product\_order, product, table and track (by taking join) ordered decreasingly by payment date.
9. Procedure to see Purchases between dates, "seePurchasesByDate(IN startTime TIMESTAMP, IN endTime TIMESTAMP)" -> works same as before just that it gives entries between the given time.
10. Procedure to see products within price range, "queryProductsTim(IN productName varchar(20), IN lowRange FLOAT, IN highRange FLOAT)" -> which after receiving suitable parameters, return entries from product sorted by price between the given range.
11. Procedure to see reviews of a given product, "ProductReviews(IN pid varchar(20), IN sid varchar(20))" -> will fetch ratings and reviews of the given product from product\_order, etc.
12. Procedure to add review for a product, "addReviewProduct(IN pid varchar(20), IN oid varchar(20), IN sid varchar(20), IN rev varchar(60))" -> as only those who have purchased successfully the product can add reviews, we need order ID for verification.
13. Procedure to add review for a seller, "addReviewSeller(IN pid varchar(20), IN oid varchar(20), IN sid varchar(20), IN rev varchar(60))" -> again we need order ID for verification.
14. Procedure to add rating for product, "addRatingProduct(IN pid varchar(20), IN oid varchar(20), IN sid varchar(20), IN rating INT)".
15. Procedure to add rating for seller, "addRatingSeller(IN pid varchar(20), IN oid varchar(20), IN sid varchar(20), IN rating INT)".
16. Procedure to see products sorted by rating, "queryProductsRat(IN productName varchar(20))".
17. Procedure to update customer info, custUpdateInfo(IN customer\_id varchar(20), IN password VARCHAR(20), IN name varchar(20), IN address VARCHAR(60), IN phone\_number DECIMAL(10) UNSIGNED, IN emailId VARCHAR(20)).



## 7.2 Seller Procedures

1. **Seller Function:** Function to return the total earning of a seller between supplied dates, "sellerStatsBetweenDate(startTime TIMESTAMP, endTime TIMESTAMP)".
2. Procedure for seller to see sold but not shipped products, soldButNotShipped(IN seller\_id varchar(20)) -> fetches entries from product\_order table which have corresponding shipper\_id NULL.
3. Procedure for seller to ship a sold product, "shipSoldProduct(IN gproduct\_id varchar(20), IN gorder\_id varchar(20), IN gseller\_id varchar(20), IN gshipper\_id varchar(20), IN gtracking\_id varchar(20), IN gdate DATE)" -> will ship the product (update shipper\_id and tracking\_id in track table) by taking relevant input.
4. Procedure for seller to see his rating, "getRating(IN seller\_id varchar(20))".
5. Procedure for seller to check whether there is already a product with given seller\_id and product\_id (this is useful when seller is adding a new product), "sellerCheckExistProd(IN product\_id varchar(20), IN seller\_id varchar(20))" -> will return a row if there is a corresponding product.
6. Procedure for seller to add new product, "addProduct(IN product\_id varchar(20), IN seller\_id varchar(20), IN product\_name varchar(20), IN product\_image varchar(300), IN price float, IN total\_stock int, IN pickup\_address varchar(60), IN description varchar(60))" -> adds an entry in product table by taking suitable input details.
7. Procedure for seller to update his specific product details, "updateProductInfo(IN product\_id varchar(20), IN seller\_id varchar(20), IN product\_name varchar(20), IN product\_image varchar(300), IN price float, IN total\_stock int, IN pickup\_address varchar(60), IN description varchar(60))" -> Note that it is possible to call this procedure by giving empty string for entries whose update is not deemed fit.
8. Procedure to update seller's info, "sellerUpdateInfo(IN seller\_id varchar(20), IN passwordd VARCHAR(20), IN named varchar(20), IN addressd VARCHAR(60), IN phone\_number DECIMAL(10) UNSIGNED, IN email\_id VARCHAR(20))" -> Note that it is possible to call this procedure by giving empty string for entries whose update is not deemed fit.

9. Procedure for seller to see their past sold products within a specific time duration, "seeSellingsBetweenDuration(IN startTime TIMESTAMP, IN endTime TIMESTAMP)" -> fetches the relevant entries from product\_order table.
10. Procedure to see latest N sellings, "seeLatestNSellings(IN N INT)" -> fetches the relevant entries from product\_order table.
11. Procedure to see seller's already listed similar products with increasing price, "selQuerySimProducts(IN productName varchar(20))".
12. Procedure to see seller's similar products sorted by rating, "selQueryProductsRat(IN productName varchar(20))".

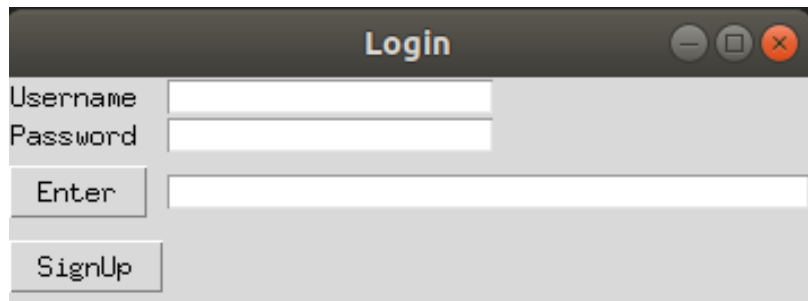
### 7.3 Shipper Procedures

1. Procedure to update shipper's info, "shipperUpdateInfo(IN shipper\_id varchar(20), IN passwordd VARCHAR(20), IN named varchar(20), IN addressd VARCHAR(60), IN phone\_number DECIMAL(10) UNSIGNED, IN email\_id VARCHAR(20))" -> works in the same way as other user's update info.
2. Procedure for shipper to see their past shipments within a specific time duration, "seeShipmentsBetweenDuration(IN startTime DATE, IN endTime DATE)" -> fetches targeted entries from track table.
3. Procedure to see latest N Shipments, "seeLatestNShipments(IN N INT)" -> fetched targeted entries from track table.

## 8 Tour Of GUI And Use Cases

### 8.1 Login Window

On start, user will be popped with this window where they can enter their detail, and if the credentials are incorrect message will be shown depicting the same. User need not specify their role, it will be automatically determined from our User Table. Also if the user wants to create an account, they can select "SignUp" option.

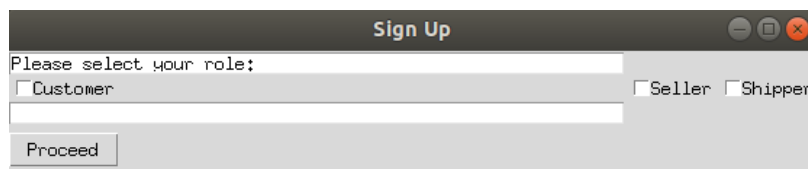


A screenshot of a 'Login' window. It features a title bar with the text 'Login' and standard window control buttons (minimize, maximize, close). The main area contains three input fields: 'Username', 'Password', and a third unlabeled field. Below these fields are two buttons: 'Enter' and 'SignUp'.

Figure 4: Login Window

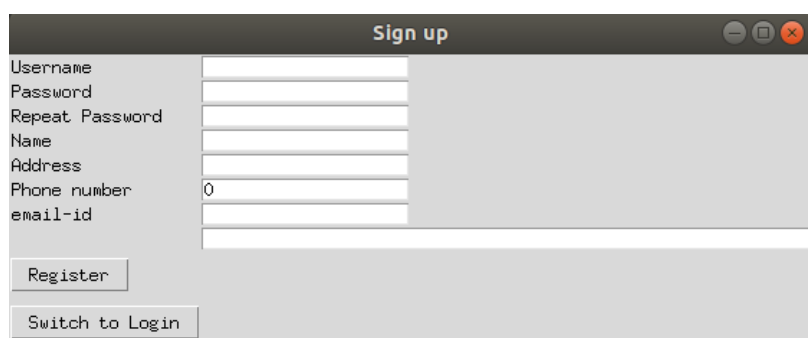
## 8.2 Sign Up

User can first select their role and then fill in their details, it implements various logic such as to check whether user has checked exactly one tick box, passwords match, whether the required fields are left empty, such a user already exists, etc.



A screenshot of a 'Sign Up' window. The title bar says 'Sign Up'. The main text says 'Please select your role:'. Below this are three radio buttons: 'Customer', 'Seller', and 'Shipper'. There is a 'Proceed' button at the bottom.

Figure 5: Sign Up: Select Role



A screenshot of a 'Sign up' window. The title bar says 'Sign up'. The main area contains several input fields: 'Username', 'Password', 'Repeat Password', 'Name', 'Address', 'Phone number' (with a '0' in the field), and 'email-id'. Below these fields are two buttons: 'Register' and 'Switch to Login'.

Figure 6: Sign Up: Enter Details

## 8.3 Customer Window

Now suppose a user of role customer logs in, they would be greeted with following windows which are shown in figure.

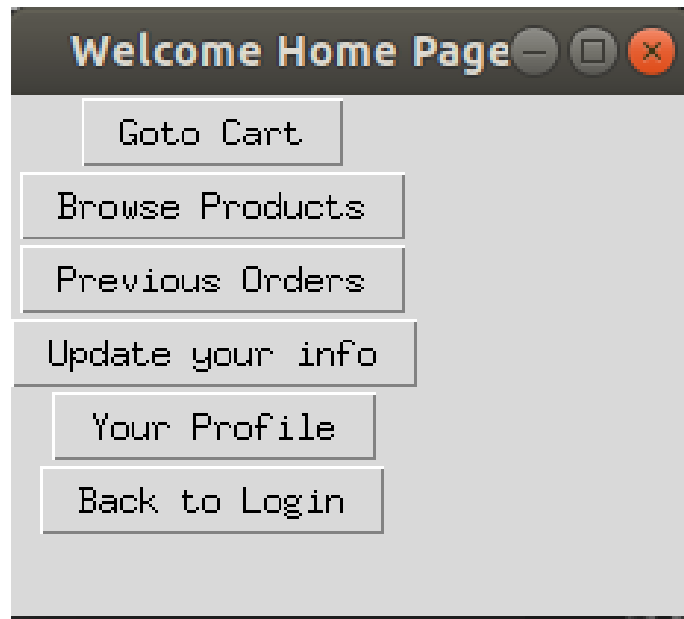


Figure 7: Customer Window

### 8.3.1 Previous Products

Here user can chose how to see their previous products, either by date (again various logics are implemented to check whether the entered date is correct, etc.) or by giving "N" for latest "N" purchases. Users can now select a particular purchased product and give ratings, reviews, etc. They can also see shipment status.

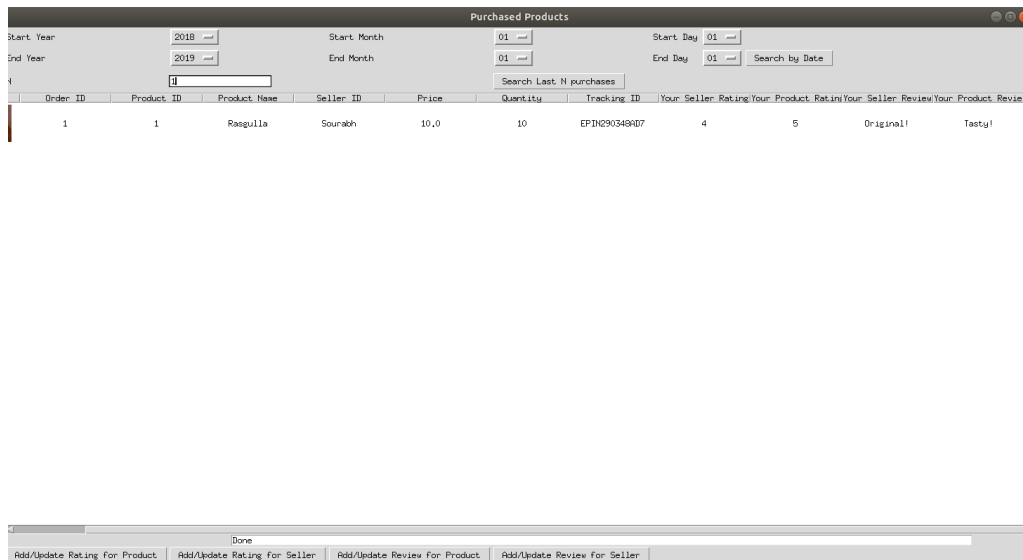


Figure 8: Previous Products

### 8.3.2 Browse Products

User can browse either by searching by name which will give results in decreasing order of rating or by searching within their budget. For the selected item, they can choose how much amount to add in cart and also see previous reviews of the product.

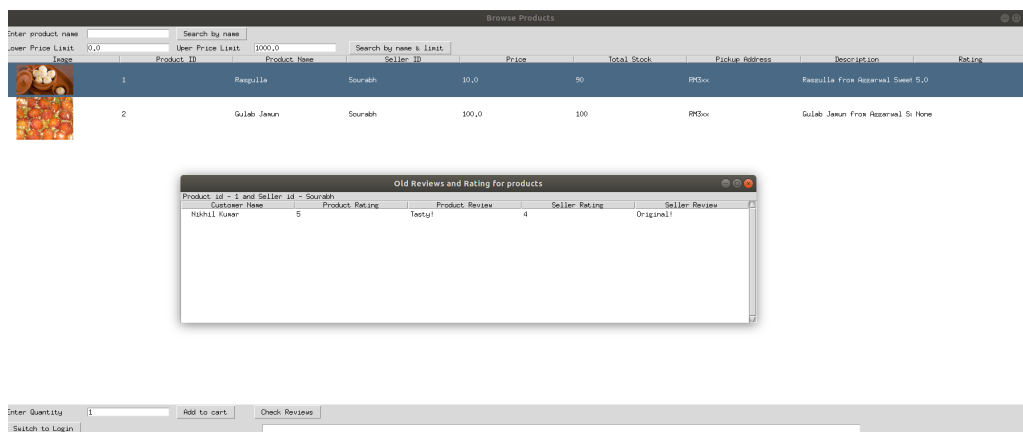


Figure 9: Browse Products

### 8.3.3 Cart

Here user will be able to see items added in cart where they can update the quantity or remove the item from cart and obviously checkout from cart option is there whereby user will be asked to enter details payment details.

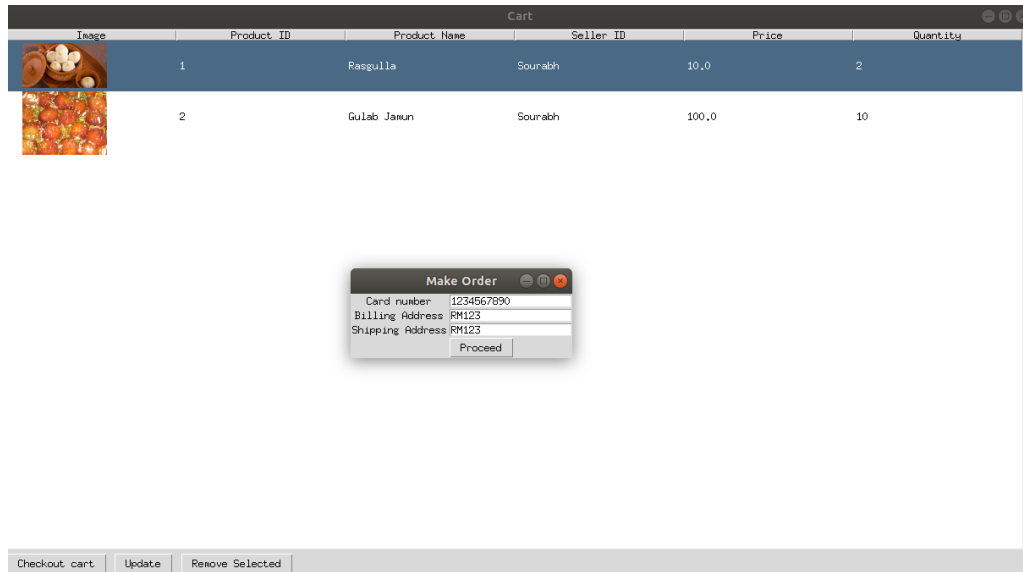


Figure 10: Cart

### 8.3.4 Update Your Info

Facility for User to update their profile.

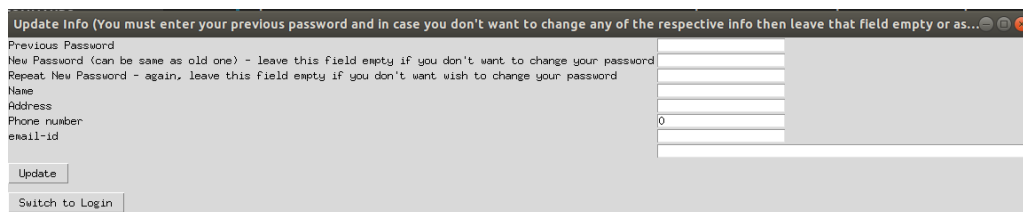
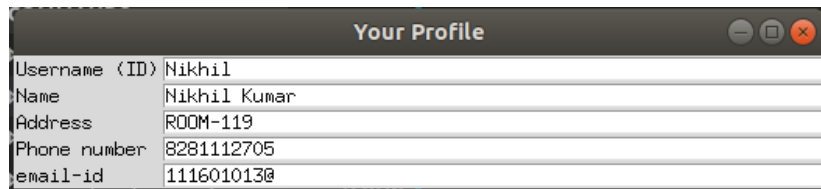


Figure 11: Update Info: Customer

### 8.3.5 View Your Profile

Facility for User to view their profile.

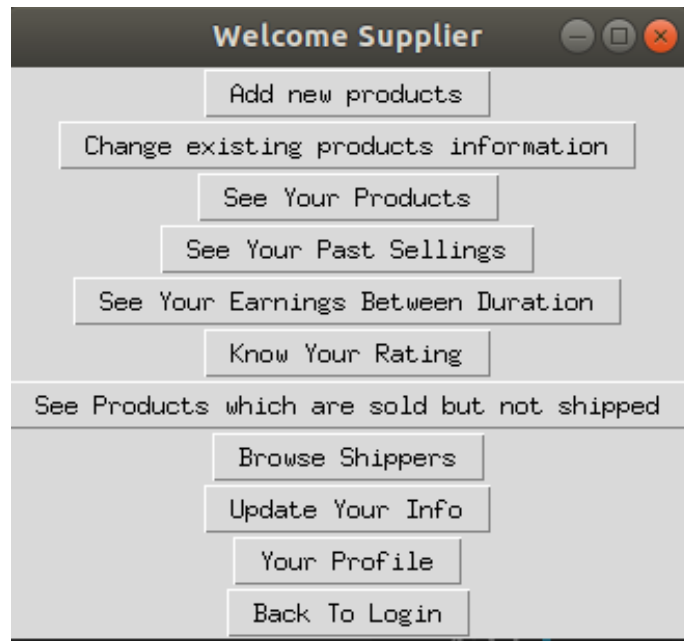


Your Profile	
Username (ID)	Nikhil
Name	Nikhil Kumar
Address	ROOM-119
Phone number	8281112705
email-id	111601013@

Figure 12: View Profile: Customer

## 8.4 Seller Window

Now suppose a user of role seller logs in, they would be greeted with following windows which are shown in figure.



Welcome Supplier
Add new products
Change existing products information
See Your Products
See Your Past Sellings
See Your Earnings Between Duration
Know Your Rating
See Products which are sold but not shipped
Browse Shippers
Update Your Info
Your Profile
Back To Login

Figure 13: Seller Window

### 8.4.1 Add New Products

Facility for seller to add a new product.

**Add Product (You must enter each field)**

Product ID

Product Name

Product Image URL

Price (Min Rs 1)

Total Stock (Min 1)

Pickup Address

Description

Figure 14: Add New Product

### 8.4.2 Update Existing Product

Facility for seller to update change existing product information.

**Update Info (In case you don't want to change any of the respective info then leave that field empty (or as it is) but clearly you must enter product\_id)**

Product ID

Product Name

Product Image URL

Price (Min Rs 1)

Total Stock (Min 1)

Pickup Address

Description

Figure 15: Update Existing Product

### 8.4.3 See Your Products

Seller can browse their existing products sorted either by price or rating.



Product Name	Product ID	Product Name	Seller ID	Price	Total Stock	Pickup Address	Description	Rating
	1	Rasgulla	Sourabh	10.0	88	RHBox	Rasgulla from Apparna's Sweet 5.0	
	2	Gulab Jamun	Sourabh	100.0	90	RHBox	Gulab Jamun from Apparna's Sweet 5.0	

Figure 16: See Your Products

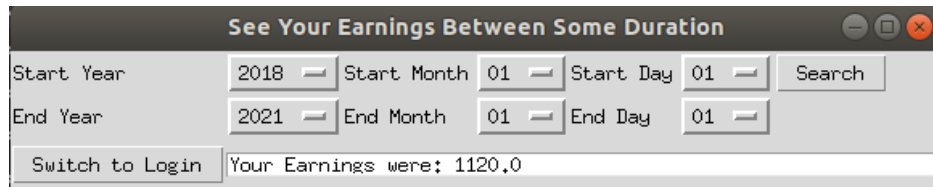
#### 8.4.4 See Past Sellings

Facility for seller to see their past sellings either between some date or latest "N" sellings.

Product ID	Order ID	Seller ID	Product Rating	Seller Rating	Ship Index	Product Review	Seller Review	Quantity
2	2	Sourabh	None	None	3	None	None	10
1	2	Sourabh	None	None	2	None	None	2
1	1	Sourabh	5	4	1	Tasty!	Original!	10

Figure 17: Past Sellings

#### 8.4.5 See Earnings Between Duration

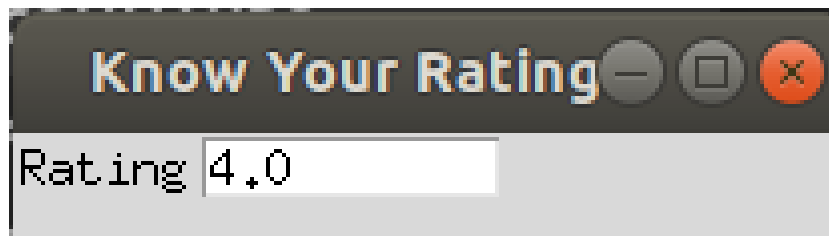


A screenshot of a web application window titled "See Your Earnings Between Some Duration". The window contains a form with date selection fields for Start Year (2018), Start Month (01), Start Day (01), End Year (2021), End Month (01), and End Day (01). A "Search" button is located to the right of the Start Day field. Below the date fields, there is a "Switch to Login" button and a text field displaying "Your Earnings were: 1120.0".

Figure 18: See Earnings Between Duration

#### 8.4.6 See Your Rating

Facility for seller to view their rating.

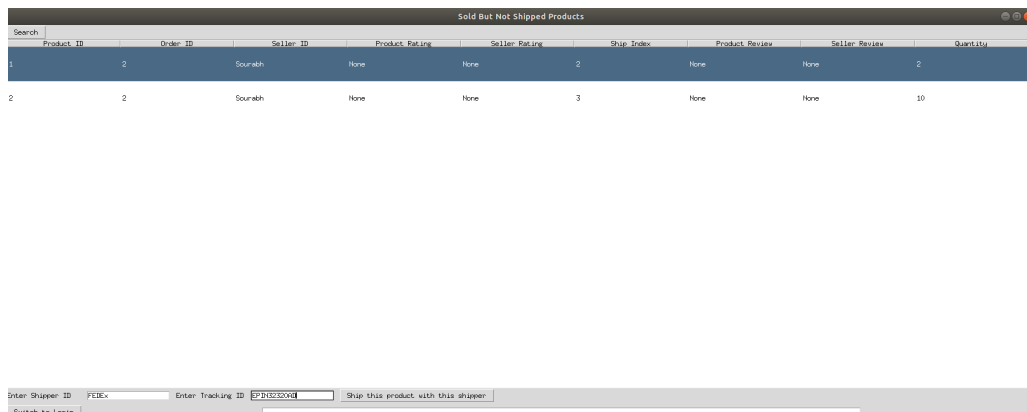


A screenshot of a web application window titled "Know Your Rating". The window displays a large text input field containing the text "Rating 4.0".

Figure 19: Know Your Rating

#### 8.4.7 See Products Which Are Sold But Not Shipped

Facility for seller to ship a sold but non shipped product.



A screenshot of a web application window titled "Sold But Not Shipped Products". The window displays a table with the following columns: Product ID, Order ID, Seller ID, Product Rating, Seller Rating, Ship Index, Product Review, Seller Review, and Quantity. The table contains two rows of data. Below the table, there is a form with fields for "Enter Shipper ID" (containing "FEDEX") and "Enter Tracking ID" (containing "57ND230V4"), followed by a "Ship this product with this shipper" button. At the bottom left, there is a "Switch to Login" button.

Product ID	Order ID	Seller ID	Product Rating	Seller Rating	Ship Index	Product Review	Seller Review	Quantity
1	2	Sourabh	None	None	2	None	None	2
2	2	Sourabh	None	None	3	None	None	10

Figure 20: Products Sold But Not Shipped

### 8.4.8 Browse Shippers

Facility for seller to see various shippers.

See information about various shippers

Shipper name:

Shipper ID	Name	Head Quarters	Phone Number	Email ID
FEDEX	FEDEX	Delhi	1800123343	1116010209

Figure 21: Browse Shippers

### 8.4.9 Update Your Info

Facility for User to update their profile.

Update Info (You must enter your previous password and in case you don't want to change any of the respective info then leave that field empty or as...)

Previous Password:

New Password (can be same as old one) - leave this field empty if you don't want to change your password:

Repeat New Password - again, leave this field empty if you don't want wish to change your password:

Name:

Address:

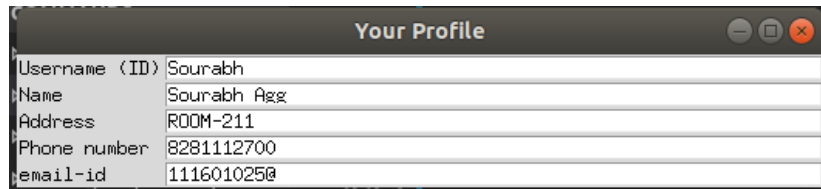
Phone number:

email-id:

Figure 22: Update Info: Seller

#### 8.4.10 View Your Profile

Facility for User to view their profile.



A screenshot of a web application window titled "Your Profile". The window contains a table with the following data:

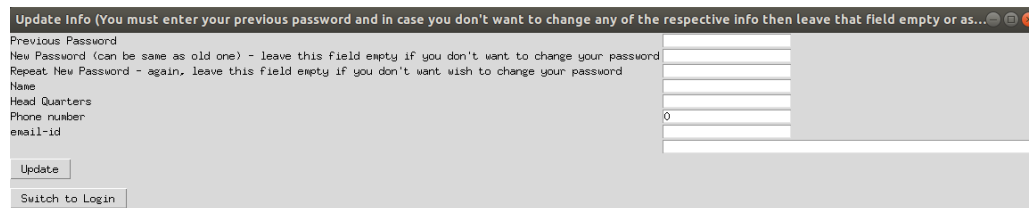
Username (ID)	Sourabh
Name	Sourabh Agg
Address	ROOM-211
Phone number	8281112700
email-id	111601025@

Figure 23: View Profile: Seller

### 8.5 Shipper Window

#### 8.5.1 Update Your Info

Facility for User to update their profile.



A screenshot of a web application window titled "Update Info (You must enter your previous password and in case you don't want to change any of the respective info then leave that field empty or as...)". The window contains a form with the following fields:

- Previous Password
- New Password (can be same as old one) - leave this field empty if you don't want to change your password
- Repeat New Password - again, leave this field empty if you don't want wish to change your password
- Name
- Head Quarters
- Phone number
- email-id

Below the fields are two buttons: "Update" and "Switch to Login".

Figure 24: Update Info: Shipper

#### 8.5.2 See Past Shipments

Facility for shipper to see past shipments either by date or by setting latest "N" shipments.

The screenshot shows a window titled "See Past Shipments". It contains search filters for Start Year (2018), Start Month (01), Start Day (01), End Year (2020), End Month (01), and End Day (01). There is a "Search by Date" button. Below these is a field "N:" with the value "1" and a "Search by N" button. The main area is a table with the following data:

Index	Shipper ID	Tracking ID	Date
1	FEDEX	EPIN290348AD7	2019-03-27

At the bottom, there is a "Switch to Login" button and a "Done!" button.

Figure 25: Update Info: Shipper

There is also a facility to see shipments in detail to see from where it was taken and where is to be delivered.

The screenshot shows a window titled "See Your Earnings Between Some Duration". It contains search filters for Start Year (2018), Start Month (01), Start Day (01), End Year (2021), End Month (01), and End Day (01). There is a "Search" button. Below these is a "Switch to Login" button and a text field showing "Your Earnings were: 1120.0".

Figure 26: Update Info: Shipper

### 8.5.3 View Your Profile

Facility for User to view their profile.

Your Profile	
Username (ID)	FEDEx
Name	FEDEx
Address	Delhi
Phone number	1800123343
email-id	111601020@

Figure 27: View Profile: Shipper

## 9 Further Reading And Useful Links

**Schema:** Complete implementation of schema, views, procedures, triggers, etc. can be found here.

**GUI:** GUI source code can be found here.

**Complete Project:** All our work, including above can be found here.