

# Schoolify (Ecommerce website)

Name

Sultan Arafat - 30057064

# Table of Contents

<b>Table of Contents</b>	<b>2</b>
<b>Abstract</b>	<b>5</b>
<b>Introduction</b>	<b>5</b>
<b>Describing our system</b>	<b>5</b>
<b>Section: Project design</b>	<b>6</b>
Different users of our system	6
Buyer users	6
Seller users	6
Both seller and buyer users	7
Guest users	7
Administrator users	7
Extended-Entity relationship diagram	8
Changes made on our ER diagram since the presentation (Project demo)	9
Changed the name of the entity ITEM to PRODUCT	9
Removed the relationship type COMMUNICATES	9
Removed the GUEST entity's IP address	9
Removed GUEST's connection with the USER entity	9
Changed ternary relationship type "chosen products" to be binary relationship type	9
Added binary relationship type "Purchases trough"	9
Changed entity type name TRANSACTION to PAYMENT	10
Moved the attribute Shipping address from BUYER entity to relationship type "view order details"	10
Removed the attribute bank account number from SELLER entity	10
Added the relationship type "posts_reviews"	10

<b>Section: Implementation</b>	<b>11</b>
Relational Model	11
Unusual decisions made during the conversion from ER diagram to Relational model	12
GUEST relation will not be part of our Relational model	12
PRODUCT relation will not include foreign key email address of ADMINISTRATOR relation	12
MEMBER relation will not include the foreign key email address of ADMINISTRATOR relation	12
Selected Database Management System	13
For admin table:	13
Statement 1	13
Statement 2	13
Statement 3	13
For cart table:	14
Statement 1	14
Statement 2	14
Statement 3	14
For member table:	14
Statement 1	14
Statement 2	14
Statement 3	15
Statement 4	15
Statement 5	15
For orders table:	15
Statement 1	15
Statement 2	15
Statement 3	16
Statement 4	16
Statement 5	16
For order_details table:	16
Statement 1	16

Statement 2	17
For payment_information table:	17
Statement 1	17
For product table:	17
Statement 1	17
Statement 2	17
Statement 3	18
Statement 4	18
Statement 5	18
Statement 6	18
Statement 7	18
Statement 8	19
For product_review table:	19
Statement 1	19
Statement 2	19
<b>API Documentation</b>	<b>20</b>
<b>User Manual</b>	<b>21</b>
<b>Instructions on how to get the code (ie:website) running</b>	<b>34</b>
<b>References</b>	<b>34</b>

# Abstract

Everyday general people are faced with situations where they find themselves always with a busy schedule, one day after another. With this being the case, it can be really difficult to dedicate any more spare time searching for and buying needed supplies that not only are available, but also budget friendly. This report will attempt to address this problem by providing a system, specifically for people that are general learners, that is more friendly in terms of sparing people's time searching for needed learning supplies while at the same time also providing a budget friendly option for shopping for these supplies. Some of the problems that have been investigated is figuring out ways to minimize the time people would spend searching for products. Our system, which is an e-commerce buying and selling website, was attempted to be designed in a way so that the person (ie: user) can immediately know where to click and navigate to the category of products that they are looking for. Another problem that has been investigated is finding ways to provide budget friendly options. Our system not only offers products that can be purchased in the general stores, but also products owned by previous users. These products have proven to offer a more friendly option for people that fall a bit short on budget. Our system (ie: ecommerce website) is designed so that it assumes that the person that will be using it is either some form of a buyer or seller. Some of the findings we found with this design is that not much learning needs to be done from users end when using our website. If the user is a seller, he posts the product on the website; If the user is a buyer, they just search for the product.

# Introduction

Task that our database was designed to address is to conveniently (ie: smoothly) manage information flow between the administrators, current members of our website and products being sold by seller members. With the potential to have many members stored on the website, along with products the seller members attempt to sell, the administrator will be spending more and more time managing both the products and the members. Most ecommerce websites start small, with not many people knowing about the websites. Overtime, more and more people know about the ecommerce website and will sign up on it. So storing data efficiently was the task our database was designed to address.

# Describing our system

Our system, which is a website, is an human-interaction type system that allows for an efficient way of buying and selling products among members (ie: people who have accounts on our website). These members (ie: account users) of our system may either be classified as buyers or sellers. Buyer members are members who have the intention to buy products offered by sellers. A member

that provides only their billing address during the registration process falls in the class of buyer members. Seller members on the other hand are members who have the intention to sell products. A member that provides a bank account number only falls in the class of seller members. If in any case, both are provided, the member can then be classified as both a buyer and a seller. Now, these members and the products are managed by our administrator users. When a product is added by a seller member, the administrator has to approve that product (and/or make any other necessary adjustments) based on the information the seller member has provided. Adjustments an administrator may make can be anything from changing the category the product falls under to changing the actual products name if necessary. This is the general structure of our system: an ongoing process of managing product and member flow as products are being sold and offered.

## Section: Project design

### Different users of our system

We have four main different types of users. These users are Buyer users, Seller users, Guest users and Administrator users.

#### Buyer users

Buyer users are users that are in the intention to buy a product on our website. They know for sure that they will buy something on the website because these users have become a member of our ecommerce website system. For these types of users, they have the option to view any of the products they have previously purchased on our website (ie: view previous orders), purchase any products or to contact the seller about any items that they are in demand along with leaving reviews of the products purchased. Furthermore, since buyers are users that are members of our system, they may make changes accordingly to their user account (ie: Edit their account information).

#### Seller users

Seller users are users that are in the intention to sell products(s) on our website. They are certain that they will want to sell products because in order to become a seller on our website, you have to become a member on our website. For these types of users, they have the option to sell any product they would like in any type of category, edit/remove any previously added products from their list of products, and to update the buyer on when the shipping date of the order is. Also, seller users may communicate with the buyer

via email according to the messages received by the buyer. Furthermore, since sellers are members of the website, they may also make any edits on their personal information just like the buyers can (ie: Edit their account information).

## Both seller and buyer users

Being both seller and buyer user assumes that you want to both buy and sell information on the website. With this being the case, this type of user then has all the features available that have been specified above for buyers and sellers.

## Guest users

Guest users have the intention only to browse the set of products offered on our system. This means that these users may only view the products that are displayed on our website and navigate from one category of products to the other, all the other features are not available to the user. Furthermore, it is assumed that this type of user does not have the intention to buy any of the products offered. This also means that these types of users are not members of our system. Guest users may become members of our system by creating an account and becoming Buyer and/or Seller members.

## Administrator users

Administrator users are users that manage information flow between buyers, sellers and products. These types of users may only have the intention to:

- Make changes to any of the categories of products: That is, to add/remove/edit any categories. Or, make any other edits to the product.
- Make changes to the product: If the seller no longer wants to sell a product, or has sold it, administrator users would remove it.
- Make changes to the members of the system: If a member no longer wants to be part of the system, they may be removed by contacting the administrator.
- Make changes in the process of how a transaction is performed: Whether a change needs to be made due to an update on the way the transition process is done, the administrator will change it accordingly.

A user that has the intention to become a member of the website can become an administrator type of user, but requires special permission from other administrators. These types of users (Administrator users) do not have the intention to buy or sell products on the website, and thus they cannot sell or buy products.

## Extended-Entity relationship diagram

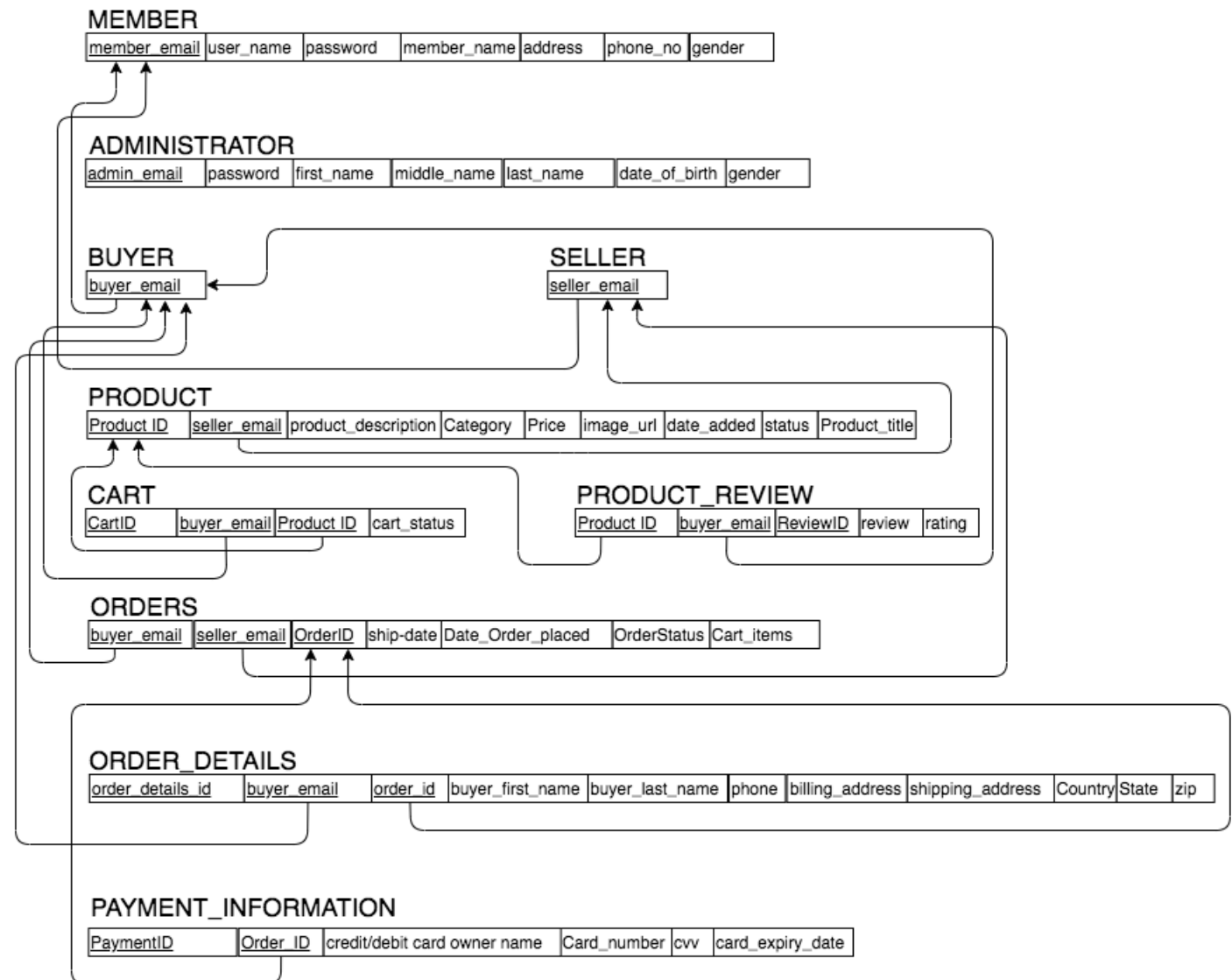
Following is our updated extended entity relationship (EER) diagram



## Section: Implementation

### Relational Model

For our relational model, we will make use of the algorithm outlined in the textbook Fundamentals of Database systems to convert our current ER diagram to Relational schema. Our latest relational model is the one on right:



## Unusual decisions made during the conversion from ER diagram to Relational model

### GUEST relation will not be part of our Relational model

In our Entity-relationship diagram design, GUEST does not have any attributes associated with it. This means that GUEST relation would then be an empty relation. For this reason, we don't include GUEST relation in our relational model.

### PRODUCT relation will not include foreign key email address of ADMINISTRATOR relation

In our entity-relationship diagram, the PRODUCT entity shares a 1 to many relationship type "Approved\_products" with ADMINISTRATOR entity. Based on the conversion algorithm, this would mean that we would have to include the foreign key email address in the PRODUCT relation. But, for our system requirements, we have found that this is unnecessary to do. Reason for this is because the admin's email does not have any relevance with the product itself. Administrator does need to approve a product that is published on the website, but it is not necessary to store the email of the administrator with the product added.

### MEMBER relation will not include the foreign key email address of ADMINISTRATOR relation

In our entity-relationship diagram, the MEMBER entity shares 1 to many relationship type "controls" with the ADMINISTRATOR entity. Based on the conversion algorithm, this would mean that we would have to include the foreign key email address in the MEMBER relation. But, for the feature requirements of our system, this implication does not quite make sense because, in our database, MEMBER relation is used solely for the purpose to store personal information and credentials of that member user. Administrators do have the ability to change any personal information about the members, but placing the administrators email to the member table for that reason does not quite make sense.

## Selected Database Management System

For our project, we have used MySQL as a way to manage our database. All the queries were then executed using mysqli functions built in the Personal Home Page (PHP) programming language. Some of our endpoints are executed by first preparing the statement, then binding a parameter and then executing the statement.

What follows are the SQL statements for each of our transactions. We will group our SQL statements by the tables we have created in our database:

For admin table:

Statement 1

```
select *  
from admin  
where email='{ $email}' and password='{ $password}'
```

The query above is used for authenticating the admin users. If the query above returns a single user, the admin is given permission to log in to our website.

Statement 2

```
insert into admin values  
('{ $email}', '{ $password}', '{ $firstname}', '{ $middlename}', '{ $lastname}', '{ $DOB}', '{ $gender}')
```

For the query above, whenever an administrator user creates an account on our website, the query above is used to add this new administrator user.

Statement 3

```
UPDATE admin  
Set mname='{ $middlename}', lname='{ $lastname}', fname='{ $firstname}', password='{ $password}',  
dob='{ $DOB}'  
where email='{ $admin_email}'
```

Query above is used whenever the administrator user makes changes to their account information.

For cart table:

Statement 1

```
SELECT cart.id,product.product_title,product.price,product.category from product inner join  
cart on cart.product_id=product.id inner join member on member.email = cart.buyer  
where cart.buyer = '{$user_email}' and cart.status=1
```

Query above is used whenever we display product information of a product that is added into the buyer members cart. This way, all products of the buyer can be displayed.

Statement 2

```
INSERT into cart(product_id,buyer) VALUES ($id,'$email')
```

This Query above is used whenever a buyer member adds an available product to cart. More, specifically, whenever we need to add an item to the cart, the query above is used.

Statement 3

```
DELETE FROM cart WHERE id={$id}
```

Query above is called wherever a buyer member makes a request to delete an item from the cart (i.e.: No longer wants it).

For member table:

Statement 1

```
SELECT * from member
```

Whenever the administrator user is logged in to our website, one of their responsibilities when on our website is for them to know and see who all the members are in the website. The SQL statement above accomplishes this by displaying all the members on the administrator user's main page of the website.

Statement 2

```
SELECT * from member where email='{$email}' or user_name='{$user_name}'
```

Whenever a member attempts to register to our system, we need to ensure that there is no duplicate member with the same email and/or username. If for any reason, the query above returns a member, then the member will not be created under this email and username, and as a result, will be asked to use another email and/or password.

### Statement 3

```
insert into member  
values('${email}','${password}','${name}','${user_name}','${address}','${phone}','${gender}')
```

Once the user provides appropriate information when registering, they then may be added as a member to our database. This is what the above query does. It simply adds a member based on the information they have provided in their registration process.

### Statement 4

```
UPDATE member set  
member_name='${name}',phone_no='${phone}',password='${password}',address='${address}' where  
email='${user_email}'
```

The query above is used whenever a member of our website decides to make edit on their personal account information. Whether it be that the seller wants to update their own Address or buyer to update their billing information.

### Statement 5

```
DELETE FROM member WHERE email='${email}'
```

Administrator user has the ability to remove any member of our website. Whether it be that the member no longer wants to be part of our buying and selling website or that they have done something that they shouldn't have admin, admin users may remove them anytime. This is when the query above is used.

### For orders table:

#### Statement 1

```
SELECT *  
from orders  
where buyer='${email}' order by order_place_date desc
```

Whenever the buyer member decides to look at their previous orders that they have placed, the above query is used to display all the orders of that buyer.

#### Statement 2

```
SELECT *  
from orders
```

```
where seller='{ $email}' order by order_place_date desc
```

Whenever an order has been placed by a buyer member, the seller of the product receives an order as well. This is then to be used later for sellers reference.

#### Statement 3

```
UPDATE orders set ship_date='{ $date}',order_status='shipped' WHERE order_id='{ $id}'
```

When the Seller member has an order as a reference from a buyer who purchased his/her product, it is the responsibility of the seller to provide the date when the order is to be shipped and the current status of that order. This is when the query above is then used.

#### Statement 4

```
INSERT into orders (order_id,order_status,card_items,buyer,seller) VALUES ('$order_id','In Progress','$card_items','$user_email','{ $seller}')
```

Once the translation of the buyer has taken place, we immediately store the orders information. The query above is used to make this happen.

#### Statement 5

```
DELETE from orders where order_id='{ $order_id}'
```

Whenever the details of the order were not successfully stored in our database, we need to ensure that the order is then immediately removed from our database. This is to ensure that our database has order details for each order placed by buyer members.

For order\_details table:

#### Statement 1

```
SELECT *  
from order_details  
where order_id='{ $order_id}'
```

Whenever a single order previously placed by a buyer member is being viewed by the buyer, we make use of the above SQL statement to display that single order in more detail.

### Statement 2

```
INSERT into order_details (fname,lname,phone,email,b_address,s_address,country,state,zip,
order_id)
values('${firstname}','${lastname}','${phone}','${email}','${baddress}','${saddress}','${country}','${state}','${zip}','${order_id}')
```

As soon as an order has been placed by a seller, we store the information that they enter. The sql statement above is used to accomplish this task.

For payment\_information table:

## Statement 1

```
INSERT into payment_information(card_owner,card_number,card_holder,expiry_date,order_id) values
('${cardname}','${cardnumber}','${cvv}','${edate}','${order id}')
```

Whenever a user places an order successfully, we want to ensure that they fill out all the payment information appropriately. Successfully storing payment information into our database lets us know that valid payment information has been entered.

For product table:

### Statement 1

```
SELECT * from product where status=$stat order by date added asc;
```

Whenever an admin user manages products posted by seller members on the website, they are interested in knowing which products are currently visible to buyer users and which products are not. The SQL statement above is then used to accomplish this task by displaying products based on whether they are active or pending.

## Statement 2

```
SELECT *
from product
where seller=$ userEmail order by date added desc";
```

Seller members need to have the ability to view what products they are currently selling. The SQL statement above does this task.

### Statement 3

```
SELECT *  
from product  
where id=$id
```

When a buyer wants to view a single product in detail, that is, a product description, reviews, etc., we then use the above SQL statement to accomplish this.

### Statement 4

```
INSERT into product(product_title,product_description,price,image_url,seller,status,category)  
values('${title}','${description}',${price},'${path}','${user_email}','pending','${category}')
```

Whenever a seller member decides to sell a product, they are required to provide information about the product that they are trying to sell. The SQL statement above is then used to store the information provided.

### Statement 5

```
UPDATE product set product_title='${title}',  
product_description='${description}',price=${price},image_url='${path}',seller='${user_email}',  
category='${category}'  
where id =${id}
```

Seller members may sometimes want to make changes on the product they are selling. Whether it be that they want to change the price of the product, change the category the product falls under or anything else, the SQL statement above is then used to accomplish this.

### Statement 6

```
UPDATE product set category='${category}' WHERE id=${id}
```

In some cases, when a seller member decides to have a selling products category be changed, he may directly ask the administrator user to make the change for them. In such cases, the SQL statement above is then used in such cases.

### Statement 7

```
UPDATE product set status='${status}' WHERE id=${id}
```

Whenever a new product is added to be sold by a seller on our website, the product does not immediately appear visible to our buyer members. Instead, the product has to first be approved by the administrator user in order to be visible to buyer members. Once the



administrator user approves the product (ie:where administrator changes the status of product to active), the above SQL statement is executed to make the change.

#### Statement 8

```
DELETE FROM product WHERE id={$id}
```

Whether it be that the product is sold or that the seller user decides not to sell the product, the product added by the seller member is eventually removed from our website. This is usually manually done by the seller members. Once the action is taken to remove the product, the SQL statement above is executed.

For product\_review table:

#### Statement 1

```
SELECT product_review.rating,product_review.review,member.member_name  
from product_review inner join member on  
product_review.buyer = member.email  
where product_review.product_id={$id};
```

Whenever a certain product is viewed in detail by our buyer members, we also display the reviews of the product from previous owners of the product. This way, our buyer members get a better idea on how and what the currently viewed product is like. We make use of the SQL statement above to display all the reviews associated with a single product.

#### Statement 2

```
INSERT into product_review(review,rating,product_id,buyer)  
Values ('{$review}',{$rating},{id},'{$email}')
```

Once the buyer member has placed an order, they may optionally decide to provide comments and ratings of how they are liking the product so far that they have purchased. Once the rating is provided, the review is stored into our database using the SQL statement above.

# API Documentation

In total, we have 27 endpoints gathered for our website Schoolify. To see our API documentation of all these endpoints and the details of each endpoint, please use the link below.

Link to our API Documentation:

<https://documenter.getpostman.com/view/15334435/TzJrDKXF>

## Instructions on how to get the code (ie:website) running

- Install Xampp in your desired location. Open the folder where Xampp is installed and inside the htdocs folder store our project folder(Schoolify).
- Open the xampp control panel and start both Apache and MySQL servers
- Run localhost/phpmyadmin/ in a browser
- Open phpmyadmin and create a database named 'ecommerce' and finally import the ecommerce.sql file provided by us inside the schoolify folder. Press go and initialize the database.

\*\*see user manual below to continue.....

# User Manual

## Here is a stepwise guide to show how the website runs and works

1. First sign up as an admin. We didn't integrate admin sign up to the website as anyone can sign up as an admin and be a root user of the website.

run-

**localhost/Schoolify/admin/register.php**

Fill all the fields and sign up as an admin.

2. Sign up as a buyer and a seller.

**localhost/Schoolify/register.php**

3. Login as a seller by pressing the login button on top right

4. Upload a product by filling up all the required fields.

5. sign in as admin to activate the product posted by the seller. If the product is not activated first then it doesn't show up on the website yet.

\*\*\*please type the following line as admin login is not integrated into the website

**localhost/Schoolify/admin/login.php**

6. After logging in as an admin, go to pending products and approve the product by the seller. After activation this product will show up on the homepage of the website and it is ready for purchase by the buyers.

7. Now login with a new buyer account. (\*\*\*Please do not use the same seller account to purchase the product as we did not allow sellers to purchase his/her own product.)

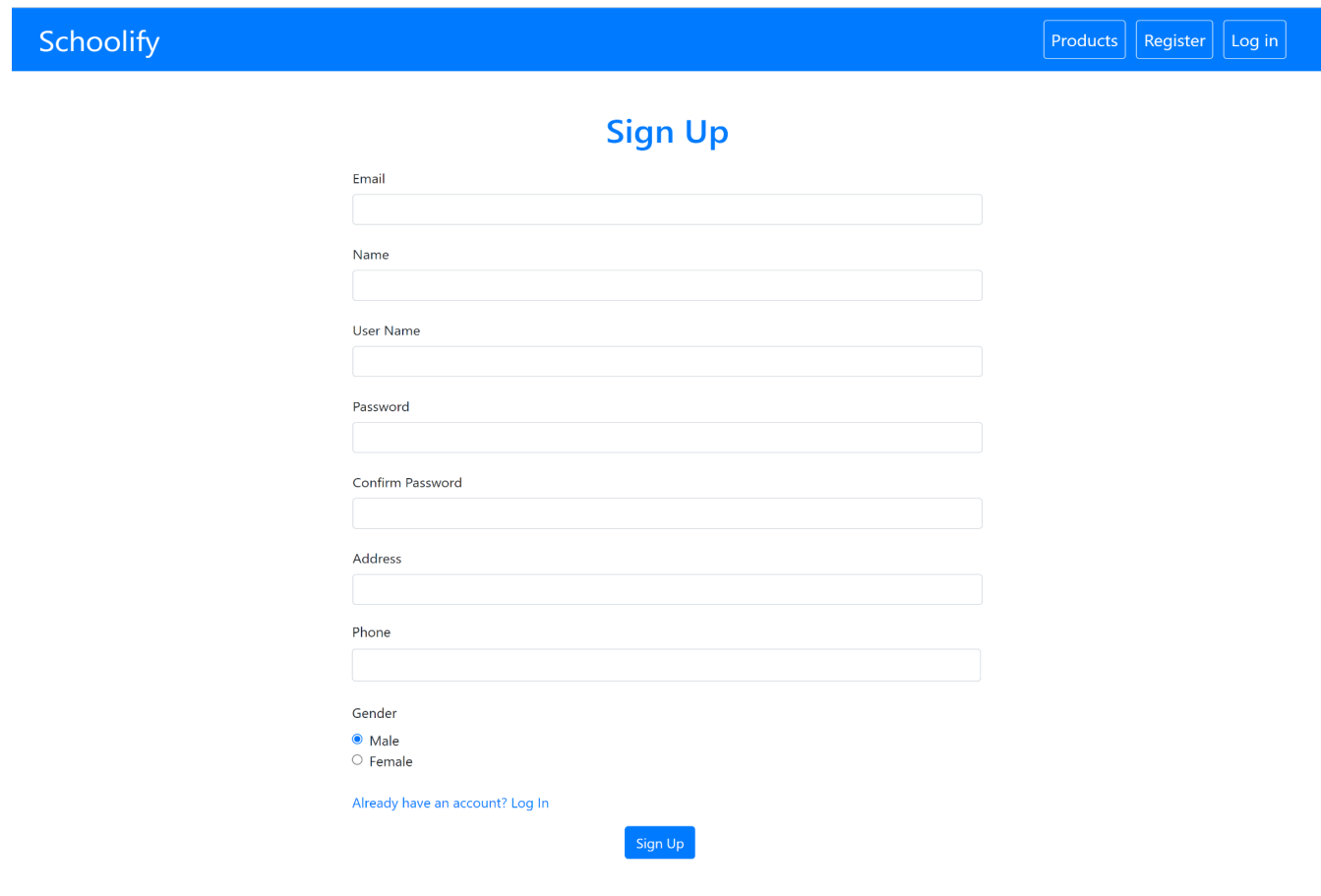
8. After logging in, please press 'switch to buying' as the profile as a seller shows up first. Now choose the product you want to purchase

9. Add the product to the cart, press 'my cart' on top to proceed to checkout and fill all the fields. Now wait for the seller to ship this product.

10. log in as a seller (follow step 3). Press 'view orders' to check for the order placed and press "ship" to ship the item.

# Member (Buyer/Seller) Registration Page:

Run localhost/Schoolify/register.php in a browser to run this page. This is the registration form for both the buyer and the seller. All the fields should be filled for a successful registration and if not, the user will be prompted to complete all the fields. If successful, the user is registered in the database's member table and he/she can continue to use the website. Registration is unsuccessful when a user tries to log in with a pre-existing user's email. In such a case a message saying "user already exists" shows up and no record is registered in the database.



The screenshot shows a web browser window displaying the 'Schoolify' registration page. The page has a blue header bar with the 'Schoolify' logo on the left and three buttons ('Products', 'Register', 'Log in') on the right. The main content area is white and features a 'Sign Up' heading in blue. Below the heading is a registration form with the following fields: Email, Name, User Name, Password, Confirm Password, Address, Phone, and Gender. The Gender field has two radio buttons, 'Male' (which is selected) and 'Female'. At the bottom of the form is a blue 'Sign Up' button. A link 'Already have an account? Log In' is located below the form fields. The browser's scrollbar is visible on the right side of the page.

Schoolify

Products Register Log in

## Sign Up

Email

Name

User Name

Password

Confirm Password

Address

Phone

Gender

☒ Male

☐ Female

[Already have an account? Log In](#)

Sign Up

## Member successful signup:

(i)

Schoolify

ProductsRegisterLog in

### Sign Up

Email

user@gmail.com

Name

user

User Name

user100

Password

\*\*\*

Confirm Password

\*\*\*|

Address

University of Calgary, Canada

Phone

8889991111

Gender

☒ Male

☐ Female

Already have an account? Log In

Sign Up

## ii) Registered successfully

Schoolify

ProductsRegisterLog in

Sign Up

Registered Successfully

Email

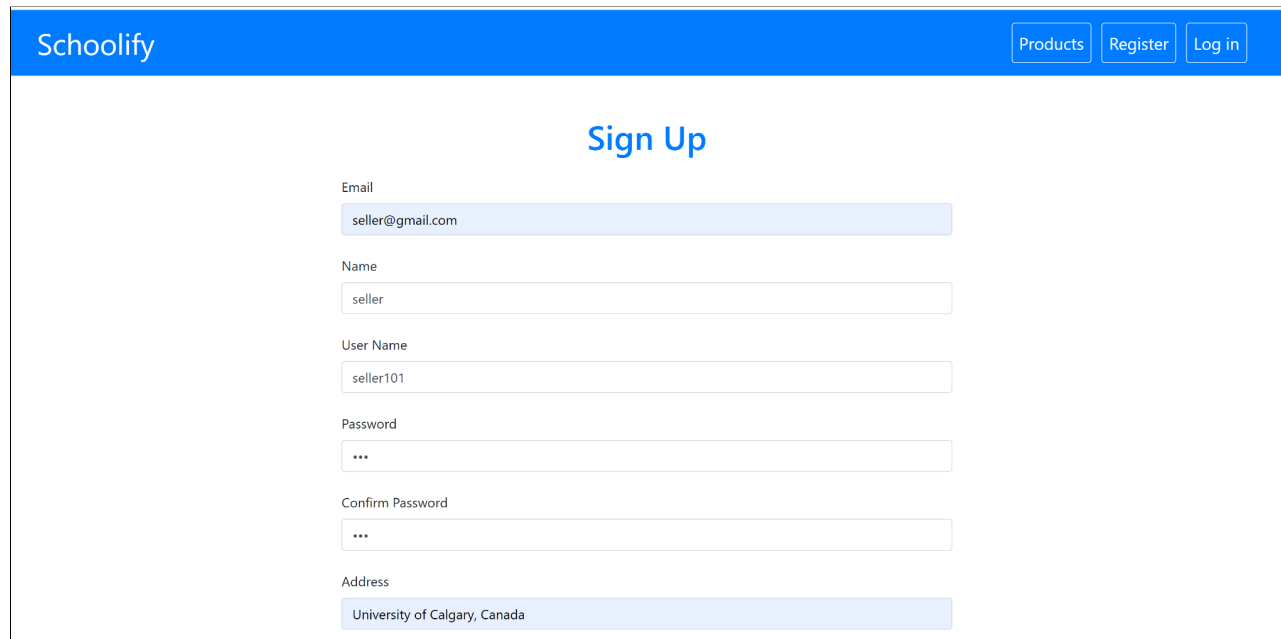
Name

User Name

Password

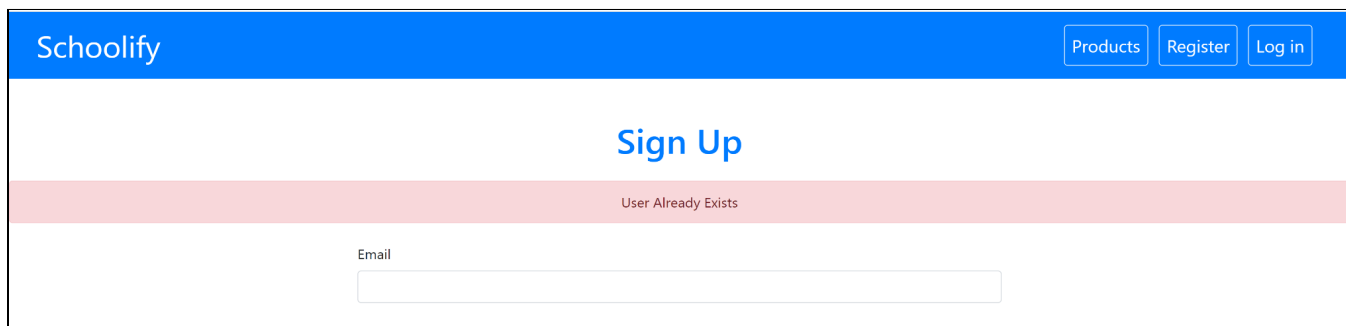
Confirm Password

**i) Existing user/Failed Sign up :**



The screenshot shows the 'Schoolify' Sign Up page. The header is blue with the 'Schoolify' logo on the left and 'Products', 'Register', and 'Log in' buttons on the right. The main content area is white and titled 'Sign Up' in blue. Below the title, there are six input fields, each with a label to its left: 'Email' (filled with 'seller@gmail.com'), 'Name' (filled with 'seller'), 'User Name' (filled with 'seller101'), 'Password' (filled with '\*\*\*'), 'Confirm Password' (filled with '\*\*\*'), and 'Address' (filled with 'University of Calgary, Canada').

**An message is displayed saying “user already exists”**

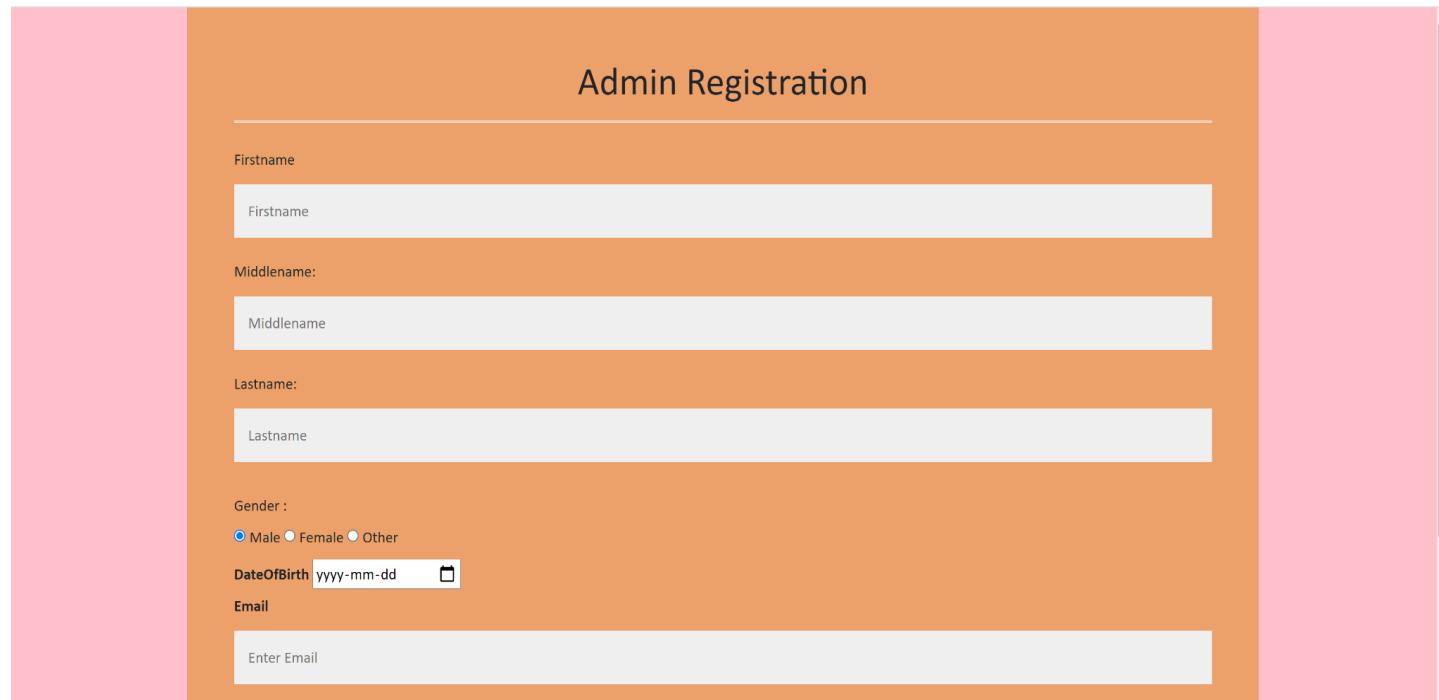


This screenshot shows the 'Schoolify' Sign Up page after a failed registration attempt. The header and 'Sign Up' title are the same as in the previous image. However, a light red horizontal banner spans the width of the form area, containing the text 'User Already Exists' in the center. Below this banner, the 'Email' input field is visible and is currently empty, while the other fields are not shown.

# Admin sign up/sign in

Run the page by typing localhost/Schoolify/admin/register.php as we did not integrate the page inside our website due to privacy issues. Admin should fill up all the fields for a successful registration and no 2 admins with the same email can sign up or else the registration will be unsuccessful.

## a)Admin Registration Page:

A screenshot of a web browser displaying the 'Admin Registration' page. The page has a light orange background with a pink sidebar on the left and right. The title 'Admin Registration' is centered at the top. Below the title, there are several input fields: 'Firstname', 'Middlename:', 'Lastname:', 'Gender :', 'DateOfBirth', and 'Email'. The 'Gender' field has three radio buttons: 'Male' (selected), 'Female', and 'Other'. The 'DateOfBirth' field has a placeholder 'yyyy-mm-dd' and a calendar icon. The 'Email' field has a placeholder 'Enter Email'.

Admin Registration

Firstname

Firstname

Middlename:

Middlename

Lastname:

Lastname

Gender :

☒ Male ☐ Female ☐ Other

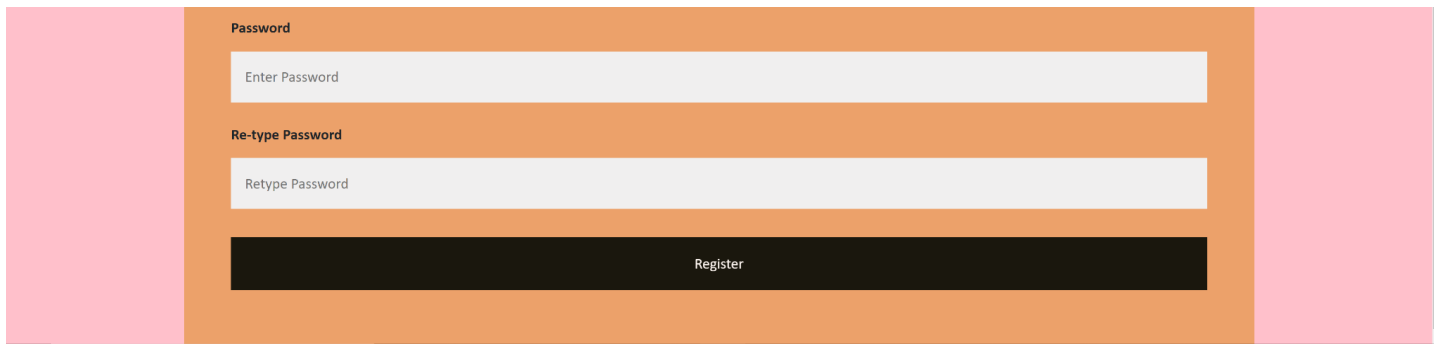
DateOfBirth yyyy-mm-dd

Email

Enter Email

---page continuation---





A registration form UI mockup. It features a central orange container with a light pink sidebar on the left and a light pink right margin. The form has two text input fields: the first is labeled "Password" and the second is labeled "Re-type Password". Below these fields is a black button with the text "Register".

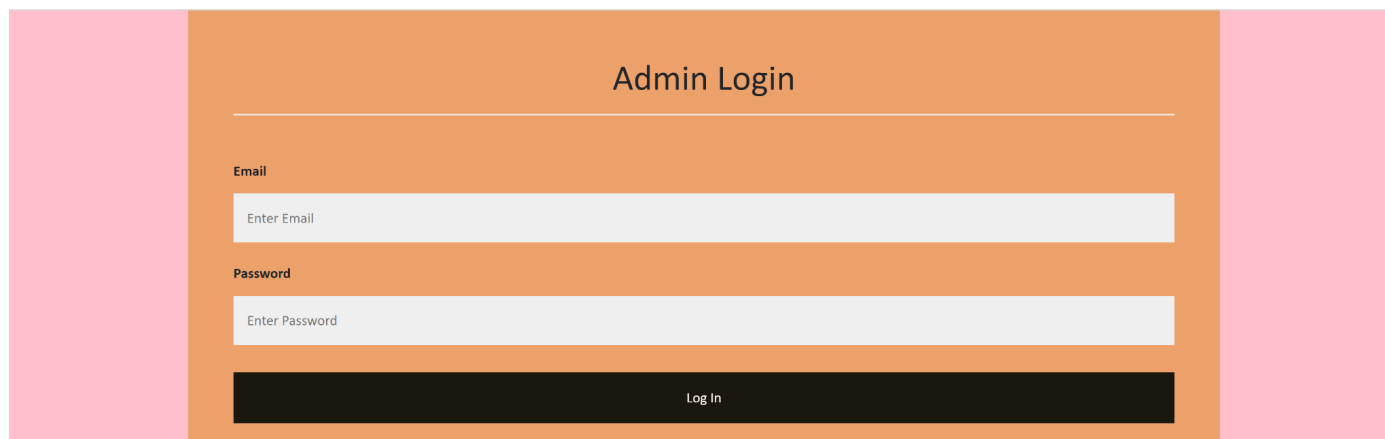
**b)Admin Successful registration:**



An "Admin Registration" UI mockup. The orange container has the title "Admin Registration" at the top. Below the title is a green success message box that says "Registered Successfully". Underneath this is a horizontal line, followed by a "Firstname" label and a text input field. Below that is a "Middlename:" label and a text input field.

**c)Admin login-**

After successful signup admin can login with the same credentials used to fill the registration form. If the email and password don't match our database then admin will not be able to login.



An "Admin Login" UI mockup. The orange container has the title "Admin Login" at the top. Below the title is a horizontal line, followed by an "Email" label and a text input field. Below that is a "Password" label and a text input field. At the bottom is a black button with the text "Log In".

## Member (Buyer/Seller) LOGIN PAGE:

After a successful registration, users can sign in with the same credentials used to fill the registration form. If the credentials don't match with the database, the user will not be allowed to login. If the user doesn't want to sign in then he/she can just browse the website as a guest by pressing the “products” button on top.

Schoolify

ProductsRegisterLog in

Login

Email

example@gmail.com

Password

Password

[Don't have an account? Sign Up](#)

Login

## Failed Login Page:

If the wrong email or password is entered, the page will prompt the wrong credential warning. As the database either does not have information in the member table or email or password is entered incorrectly.

### Login

Wrong Credentials

Email

example@gmail.com

Password

Password

[Don't have an account? Sign Up](#)

Login

## Seller My product Page:

After signing as a seller he is redirected to his homepage where all the products posted by him are displayed. From the seller's dashboard, they can view orders posted by the buyers, upload a product, edit his own profile and also switch to buying if he wants to buy anything from the website.

Seller Account

My Products

View Orders

Upload Product

Edit Profile

Switch To Buying

Log Out

### My Products

No Products

## Edit profile-

Seller can edit his profile if he wants to update his profile by filling up all the fields. If ``edit” button is pressed the changes will be reflected accordingly in his profile and also the database.

Seller Account

My Products

View Orders

Upload Product

Edit Profile

Switch To Buying

Log Out

Edit Profile

Name

buyer

Password

Confirm Password

Address

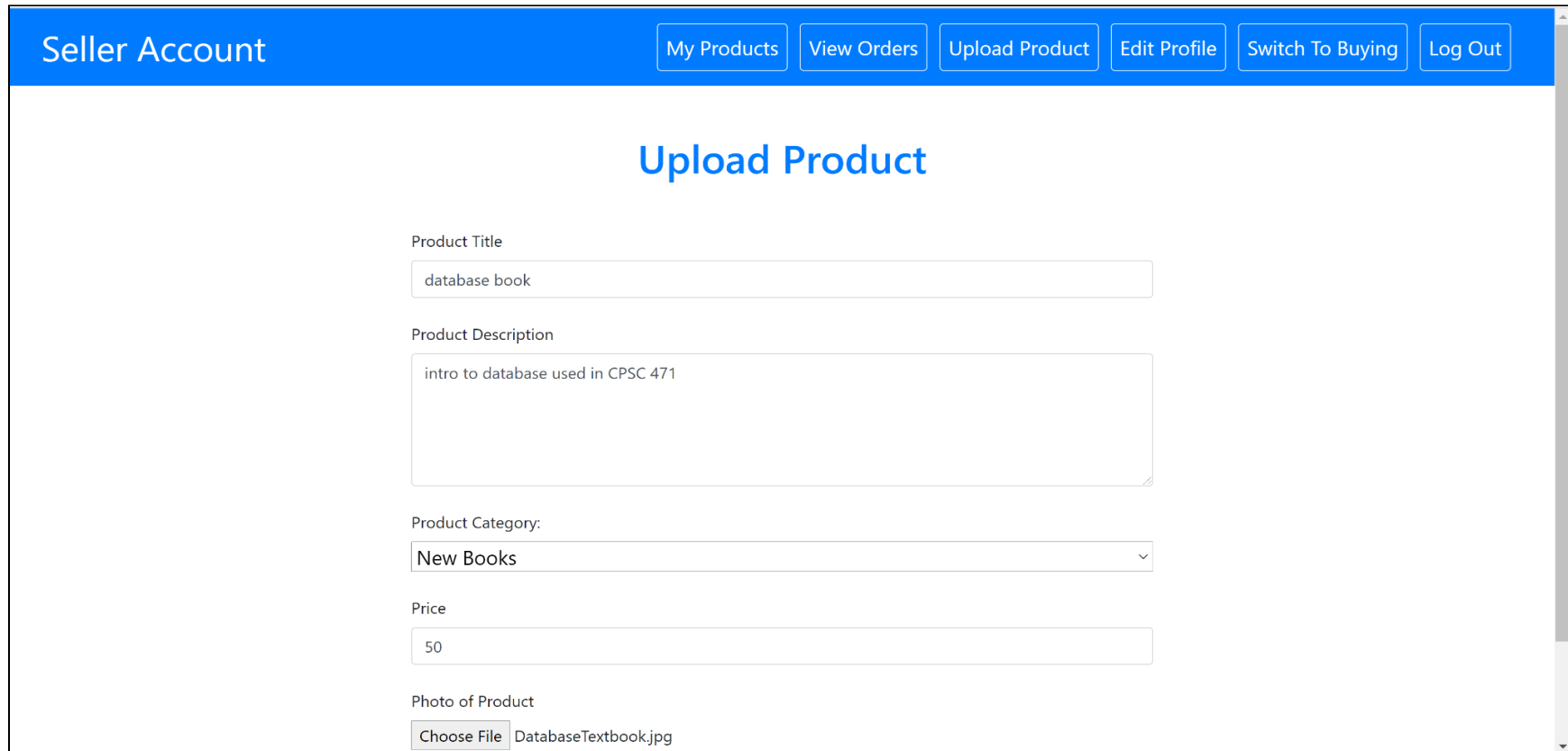
6327

Phone

5879663292

Edit

## Seller Upload product page:



The screenshot shows a web interface for a seller's account. At the top is a blue navigation bar with the text "Seller Account" on the left and five buttons: "My Products", "View Orders", "Upload Product", "Edit Profile", and "Switch To Buying", followed by a "Log Out" button. The main content area has a heading "Upload Product" in blue. Below this heading are five form fields: "Product Title" with the text "database book", "Product Description" with the text "intro to database used in CPSC 471", "Product Category:" with a dropdown menu showing "New Books", "Price" with the text "50", and "Photo of Product" with a "Choose File" button and the filename "DatabaseTextbook.jpg".

**Seller Account**   [My Products](#)   [View Orders](#)   [Upload Product](#)   [Edit Profile](#)   [Switch To Buying](#)   [Log Out](#)

### Upload Product

Product Title  
database book

Product Description  
intro to database used in CPSC 471

Product Category:  
New Books

Price  
50

Photo of Product  
[Choose File](#) DatabaseTextbook.jpg

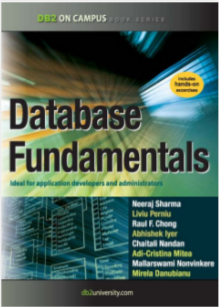
Seller members can use this page to upload a product by filling up the details. All the fields should be filled or else the product upload will not be successful. Seller members can also choose a category and upload an image with the product.

## Product added successfully

**Seller Account**

My ProductsView OrdersUpload ProductEdit ProfileSwitch To BuyingLog Out

### My Products



database book
Category: New Books
Price: 50\$
Status: pending
Date Added: Apr 18,2021
<a href="#">Edit</a> <a href="#">Delete</a>

After the product is added successfully, the seller should wait for the admin to approve the product. After this approval, only then will the product show up on the website's homepage and will be ready for purchase. Seller can Edit/Delete the product.

## Edit product

Sellers can edit the details about the product and the changes will be displayed accordingly.  
Seller can also edit his profile too.

Seller Account

My Products

View Orders

Upload Product

Edit Profile

Switch To Buying

Log Out

### Edit Product

Product Title

database book

Product Description

intro to database used in 471

Product Category:

Old Books

Price

50

Photo of Product

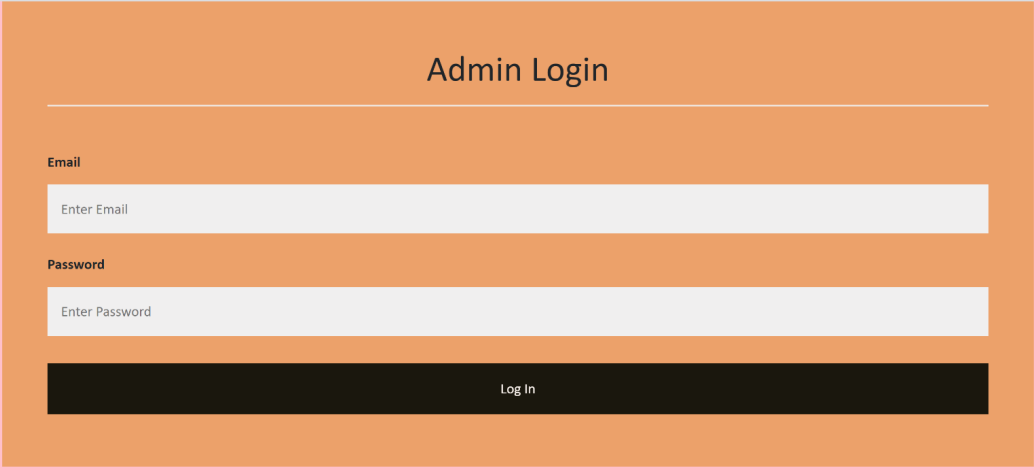
Choose File

No file chosen

Edit

# ADMIN LOGINS

Now admin needs to sign in to approve the products posted by the seller. Admin login works the same way as the member login but instead of checking the member table in the database, admin login only checks in the admin table.

The image shows a web form titled "Admin Login" centered within an orange rectangular container. The container is set against a light pink background. The form itself has a white background and includes a horizontal line at the top. Below the line, the word "Email" is displayed in a small, dark font. Underneath this label is a wide, light gray rectangular input field containing the placeholder text "Enter Email". Below the email field, the word "Password" is displayed in a small, dark font. Underneath this label is another wide, light gray rectangular input field containing the placeholder text "Enter Password". At the bottom of the form is a solid black rectangular button with the text "Log In" centered in white.



## Activate product by seller.

Admin views all the pending products waiting for activation in products list. Admin may approve or disapprove a product. If they approve the product, the product will be on the website. Otherwise, it will not. Admin can also remove the product altogether.

Schoolify

[View Members](#) [View Orders](#) [View Products](#) [Pending Products](#) [Edit Profile](#) [Log Out](#)

Products List

Count	Product Title	Product Category	Details	Activate	Remove
1	database book	New Books	<a href="#">View</a>	<div><div>---</div><div>---</div><div>Approve</div><div>Disapprove</div></div>	<a href="#">Remove</a>

## Admin view/change category/remove a product

Admin can view the details about a product posted by the seller. He can sort the category or change it too. He can also remove the product from the website altogether.

Schoolify

[View Members](#) [View Orders](#) [View Products](#) [Pending Products](#) [Edit Profile](#) [Log Out](#)

Products List

Count	Product Title	Status	Details	Change Category	Remove
1	database book	active	<a href="#">View</a>	<a href="#">Change</a>	<a href="#">Remove</a>

Edit Category

Old Books

Old Books

New Books

Lab Supplies

Gifts

Electronics

## View/remove members

Admin can see all the registered members here. He can view the profiles of all the members and also remove any users from the website.

Schoolify						<a href="#">View Members</a>	<a href="#">View Orders</a>	<a href="#">View Products</a>	<a href="#">Pending Products</a>	<a href="#">Edit Profile</a>	<a href="#">Log Out</a>
Members											
Count	Name	Phone	Address	Remove	Details						
1	buyer	5879663292	6327 Dalmarnock Crescent ,Nw	<a href="#">Remove</a>	<a href="#">View</a>						
2	seller	5879663292	6327 Dalmarnock Crescent ,Nw	<a href="#">Remove</a>	<a href="#">View</a>						

## Orders list-

Admin can see the details about all the orders placed here and the previous orders.

Schoolify					
View Members View Orders View Products Pending Products Edit Profile Log Out					
Orders List					
Count	Order ID	Order Date	Order Status	Order Ship Date	Details
1	607c3f8c81707	April 04, 2021	shipped	2021-04-18	<a href="#">View</a>

## After activation by admin, all activated products show up in the homepage-

After the admin approves sellers' posted product, the product is now visible in the homepage of the website. All the user can see the product and product details. Also, to see further details of the product, such as reviews/feedback of the product, the user can click on the blue text under the image. Users can also use the search option to find a specific book, they can also select a category to display all the products related to particular category (New Book, Old Book, Lab Supplies, Electronics, Gifts).

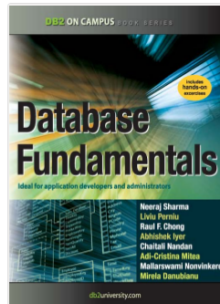
## All Products

Show All Products

Select Category ▾

Search Products

Search



database book

Category: New Books

Price: 50\$

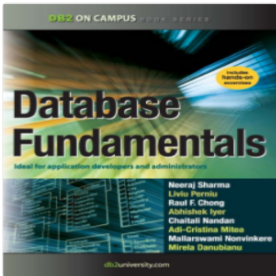
Date Added: Apr 18,2021

## View product

In the product display page, the buyer can see the product reviews and sellers description of the product. The buyer can press on the “add to my cart” button which will add the currently viewed product to the buyer's cart. Or the buyer can click on the “homepage” which will take Buyer to the previous page.

**Buyer Account**[Buy Products](#)[View Orders](#)[My Cart](#)[Switch To Selling](#)[Log Out](#)

database book



BOOK ON CAMPUS

**Database Fundamentals**

ideal for application developers and administrators

Neeraj Sharma  
Lyle Priddy  
Raul F. Chong  
Akhilesh Tyagi  
Charlton Rowden  
Adi-Cristina Mitrea  
Muthuswami Narayanaswami  
Mirela Dumitrescu

edu.university.com

Price: \$50.00

Category: Used Items

Sellers Description: intro to database used in 471

Add to my cart

Homepage

User product reviews

User: seller

Rating: 4

Review: Excellent book to read for database knowledge

## My Cart Page-

After clicking the “Add to my cart”, the product can get added to the cart and the user can see the added product in the My Cart page. Here, the user can see the added product details, and get the total price of the product. Users can remove products from the cart by clicking the red cross. User can also click continue shopping which will take the user back to the homepage or user can click the “Proceed to Checkout” button to continue and checkout.

Buyer Account

Buy Products

View Orders


My Cart

Switch To Selling

Log Out

Online Buying and selling Items System

My Cart:

Item name	Category	Price	Remove
database book	New Books	\$50	

Total number of items:

1

Shipping price (5% of original price):

\$2.50

Total price:

\$50.00

Continue Shopping

Proceed to Checkout

## Fill checkout form:

After clicking the “Proceed to Checkout”, the user is brought to the checkout page. Here, the user is asked to fill the checkout information. After filling up the form, the user clicks the checkout button.

Buyer Account

Buy Products

View Orders

My Cart

Switch To Selling

Log Out

Checkout Information

Firstname:

Sultan

Lastname:

Arafat

Phone :

5870653292

Email Address:

sultanarafat10@gmail.com

Billing Address :

6327 Dalmarnock Crescent ,Nw

Shipping Address :

calgary

Country

Canada

State

Alberta

Zip:

T3A 1H3

Payment Information:

Card Owner Name:

sultan

Card Number:

12344

CVV:

234

Expiry :

2024-04-14

234

Checkout



## Order receipt/summary:

After clicking the checkout button, the order has been placed and the order info and order receipt is displayed to the user.

Buyer Account	<a href="#">Buy Products</a> <a href="#">View Orders</a> <a href="#">My Cart</a> <a href="#">Switch To Selling</a> <a href="#">Log Out</a>
<div>Order Info:</div> <div>Email address: sultanarafat10@gmail.com</div> <div>Billing address: 6327 Dalmarnock Crescent ,Nw</div> <div>Postal Code: T3A 1H3</div> <div>Phone: 5879663292</div> <div>Grand Total: 52.50\$</div> <div>Order Summary:</div> <div>Order #: 607cb5e0376e7</div> <div>Subtotal of Items: 52.50\$</div> <div>Total before tax: 52.50\$</div> <div>Tax: 0.00\$</div> <div>Grand Total: 52.50\$</div>	

## Seller logs in and ship product

After an order has been placed by the buyer, the seller associated with the sold product needs to log in and ship the order. The seller Logs in and Goes to My orders, here the seller can see all the products that have been ordered by the user. The seller can click on the ship or view button.

Seller Account						My Products	View Orders	Upload Product	Edit Profile	Switch To Buying	Log Out
My Orders											
Count	Order ID	Order Date	Order Status	Order Ship Date	Ship Order	Details					
1	607cb5e0376e7	April 04, 2021	In Progress	Not Shipped Yet	Ship	View					
2	607c3f8c81707	April 04, 2021	shipped	2021-04-18	shipped	View					

# Seller Views the order:

Seller clicks on the view button and it displays the buyer information and the product details



database book

Category: New Books

Price: 50.00\$

## Shipping details

First Name:	Sultan
Last Name:	Arafat
Email:	sultanarafat10@gmail.com
Phone:	5879663292
Billing Address:	6327 Dalmarnock Crescent ,Nw
Shipping Address:	canada
Country:	Canada
State:	Alberta
Zip:	T3A 1H3

# Order is shipped

Here the seller clicked the ship button and now it is shipped and on the way for delivery to the buyer's provided address.

Seller Account		My Products	View Orders	Upload Product	Edit Profile	Switch To Buying	Log Out
My Orders							
Count	Order ID	Order Date	Order Status	Order Ship Date	Ship Order	Details	
1	607cb5e0376e7	April 04, 2021	shipped	2021-04-19	shipped	<button>View</button>	
2	607c3f8c81707	April 04, 2021	shipped	2021-04-18	shipped	<button>View</button>	

# Add product Ratings:

Buyer can log back in to view orders page and see all the ordered products and give feedback/rating to the product

(i)

Buyer Account

Buy Products

View Orders

My Cart

Switch To Selling

Log Out

My Orders

Count	Order ID	Order Date	Order Status	Order Ship Date	Details
1	607cb5e0376e7	April 04, 2021	shipped	2021-04-19	<div>View</div>
2	607c3f8c81707	April 04, 2021	shipped	2021-04-18	<div>View</div>

(ii)

Buyer Account

Buy Products

View Orders

My Cart

Switch To Selling

Log Out

Products In the order

Database Fundamentals

database book

Category: New Books

Price: 50.00\$

Rate

Rating and Review

Rating 1

Review

save

# References

Here is a list of references that were used to complete our final report.

- KIJJI website: <https://www.kijiji.ca/>
  - Our main idea for our project came from the KIJJI website itself
- Textbook Fundamentals of Database systems (7th Edition)
  - This textbook helped us build our new Entity-Relationship diagram (ER diagram) and eventually follow the algorithm to convert the ER diagram into a relational model.