# Card 1:

A user should be able to register for an account on Shoe++

**Conversation:**

A user should be able to access a form to create an account. The form should ask for the user’s email address, password, first name, last name and current residing country

**Conformity:**

* Register a user and if the user is properly registered, the registration is successful.
* Login successful if registered.

**Implementation:**

HTML forms (text fields)

Javascript error checking (REGEX)

Hash the password

INSERT INTO users(...)

# Card 2:

A user can login to his / her account on Shoe++

**Conversation:**

A user must enter his email address and password to login

**Conformity:**

* Attempt login with good credentials and get logged in
* Attempt login with bad credentials and get rejected

**Implementation:**

Store user input in variables and query database using prepared statement. SQL Command:

SELECT \* FROM USERS WHERE EMAIL=?  
password\_verify(...)

session variable (store user information if login is successful)

# Card 3:

A user can logout of his / her account on Shoe++

**Conversation:**

A user should click a link in the navigation bar to logout

**Conformity:**

* Click link and try to access a page a user should access only when logged in and get redirected to the main page

**Implementation:**

An html link or button that logs out the user

<a id=”logoutButton”>Logout</a>

Session variable (set user info to null)

# Card 4:

A user should be able to access the registration page from the login page (besides the navbar).

**Conversation:**

On the login page, there should be a link to a form to create a new account for those who do not have one.

**Conformity:**

* I first click on the ‘Don’t have an account?’ link.
* I then register as a new user and if I am properly registered, the registration is successful.
* The login is only successful if registered.

**Implementation:**

HTML form with input fields

# Card 5:

An existing user should be able to change his password

**Conversation:**

In the user’s account setting, a user should be able to access a form to change the his / her password.

**Conformity:**

* I first click on the link and then enter the current and valid password for confirmation.
* Then, I go to the login page and successfully login with my new password.

**Implementation:**

HTML form

password\_hash()

SELECT password\_hash FROM users

UPDATE users SET password\_hash = ? WHERE email = ?

# Card 6:

A user, as a buyer, can add items to his shopping cart

**Conversation:**

When a user is interested in an item, he can added that same item to a virtual shopping cart

**Conformity:**

* I click on the ‘Add to Cart’ button. Then, I click the ‘View Cart’ button in order to ensure that the item is there.

**Implementation:**

INSERT INTO order\_details VALUES(...)

if (the order has not been initialized)

{

INSERT INTO order VALUES(...)

}

HTML button

# Card 7:

If a user is not logged in and clicks the button to add an item to the cart, he will be redirected to the login page.

**Conversation:**

The system should detect if the user is logged in or not. If the user is not logged in and attemps to add an item to the cart, he is redirected to the login page.

**Conformity:**

* I am not logged in as a user on Shoe++
* I go to an item’s page
* I clicked the “ADD TO CART” button and I am redirected to the login page

**Implementation:**

if( isset($\_SESSION[“userID”]) )

{

// add to cart  
}

else

{

// redirect to login page

}

# Card 8:

A user, as a buyer, can remove items from his shopping cart

**Conversation:**

If the user is no longer interested in an item, he should be able to remove the item from the view shopping cart page

**Conformity:**

* I click on the ‘Remove from Cart’ button and I notice that the item in question is no longer displayed.

**Implementation:**

DELETE FROM order\_details WHERE item\_id = ?;

HTML button

# Card 9:

A user, as a buyer, should be able to clear his shopping cart with a button

**Conversation:**

If the user wants to remove all items from his shopping cart, there should be a button to remove all items from the cart. That way the user won’t have to remove the items one by one.

**Conformity:**

* I click on the ‘Clear Cart’ button and I notice that the list of items is now empty.

**Implementation:**

DELETE FROM order\_details WHERE order\_id = (SELECT id FROM orders WHERE user\_id = ?);

HTML button

# Card 10:

A user, as a buyer, should be able to proceed to the payment of all items in the shopping cart

**Conversation:**

When the user is done browsing Shoe++ for products and is ready to pay, there should be a button on the shopping cart page to purchase the items selected.

**Conformity:**

* I click on the ‘Proceed to Checkout’ button and then I fill in all of the information needed in order to purchase my item.
* Once the process is complete, I click on ‘View Cart’ and notice that the list of items is now empty.

**Implementation:**

UPDATE orders SET status = ‘purchased’, confirmation = ? WHERE user\_id = ?;

HTML form

# Card 11:

The system should be able to calculate the total price of all items in the cart.

**Conversation:**

When the user is on the cart page, he should see the total price of all items in the cart.

**Conformity:**

* I am taken to the cart page and I see the total price of the items in my cart

**Implementation**

SQL select price from item where id = ?

HTML display in <p>Total Price: ??.??</p>

# Card 12:

The system should display a message if there are no items in the cart.

**Conversation:**

When the user is on the cart page and he has no items in the cart, the system should display the message : “You have no items in your cart.”

**Conformity:**

* I have no items in the cart and I visit the cart page. I should see the message “You have no items in your cart.”

**Implementation**

if(isset($data[“items”]))

{

// html code if we have items in the cart

}

else

{

// html code if we don’t have items in the cart

}

# Card 13:

A user should be able to view all items that were purchased under his account

**Conversation:**

A user should be able to access a page where he can view all previously bought items to keep track of all purchases.

**Conformity:**

* I click on the ‘Purchase History’ link and notice that all of the items that I’ve purchased are listed.

**Implementation:**

HTML form

SELECT \* FROM orders WHERE user\_id = ? AND status = ‘Purchased’;

# Card 14:

A user should be able to recommend an item to someone else by email

**Conversation:**

There should be link on any item’s page to recommend that same item to a friend. The link will send an email through PHP.

**Conformity:**

* I click on the ‘Recommend’ link and then type the email address of my friend in the form. I then click on the ‘Send’ button to send the message.
* I then login to an email service as my friend to ensure that the recommendation was successfully sent.

**Implementation:**

PHP mail function

# Card 15:

A user, as a buyer, can review the item purchased.

**Conversation:**

After purchasing an item, a user should be able to let a review on the item’s page to help other user with their shopping.

**Conformity:**

* Post a review from a user account and check if that same review can be viewed under another’s account.

**Implementation:**

SELECT \* FROM orders, order\_details WHERE user\_id = ? AND item\_id = ? AND status LIKE ‘Purchased’;

INSERT INTO reviews(...);

HTML form

# Card 16:

A user should be able to view all reviews for an item

**Conversation:**

On an item’s page, the user should be able view all reviews for that item. This can help him make a decision for whether or not the item is worth the purchase.

**Conformity:**

* After repeated the instructions on Card #13, I login as another user to ensure that the new review is visible to everyone.

**Implementation:**

SELECT \* FROM reviews WHERE item\_id = ?;

# Card 17:

A user should be able to delete his review

**Conversation:**

A user can have the choice of deleting his review

**Conformity:**

* I click on the ‘Delete Review’ button and then I notice that the review in question is no longer displayed alongside the other reviews.

**Implementation:**

DELETE FROM reviews WHERE item\_id = ? AND writer\_id = ?;

# Card 18:

The system should be able to calculate and convert the price of one currency (ex. British Pound) to another (ex. Canadian Dollar).

**Conversation:**

This function should take place automatically and any prices displayed shall be displayed in the resultant form, based on the user’s location.

**Conformity:**

* I click on an item that is being sold from Great Britain and I notice that the price displayed is in Canadian Dollars instead of British Pounds.

**Implementation:**

Converted price = Original price \* Exchange rate

# Card 19:

The system should be able to display the positivity percentage next to an item’s ‘Reviews’ section.

**Conversation:**

This function should take place in the case when an item’s page is shown.

**Conformity:**

* I examine the item’s ‘Reviews’ section and I notice the positivity percentage.
* I validate the number through inspection of the database and comparison.

**Implementation:**

Positivity percentage = Number of positive reviews / Number of positive and negative reviews \* 100

SELECT COUNT(posorneg) FROM reviews WHERE posorneg = 1;

SELECT COUNT(posorneg) FROM reviews;

# Card 20:

A user should be able to change his username (not the email address, but the flashy nickname).

**Conversation:**

This function should take place when the user clicks on a button or link that reads “Update Profile”.

**Conformity:**

* I first click on the link and then enter the valid password for confirmation and also my new username.
* Then, I go to the login page and successfully login with my new username.

**Implementation:**

HTML form

if (password is correct)

{

UPDATE users SET uname = ? WHERE id = ?

}

# Card 21:

A user should be able to sell an item.

**Conversation:**

This function should take place once all fields have been verified with positive results after the user has clicked on the “Submit” button after having filled out the form.

**Conformity:**

* I click on the ‘Sell’ button and fill in the form.
* Once the form has been filled, I click on the ‘Submit’ button to complete the process.
* I then search for the item to ensure that it is listed.

**Implementation:**

HTML form

INSERT INTO items VALUES (id, seller\_id, name, description, price);

# Card 22:

A user should be able to take down an item that they have put up for sale.

**Conversation:**

This function should take place once the user has clicked on the “Withdraw Sale” button.

**Conformity:**

* I click on the ‘Withdraw Sale’ button and then I refresh the page and notice that the form is empty.

**Implementation:**

HTML form

if (item is unsold)

{

DELETE FROM items WHERE (id = ?) AND (seller\_ID = ?);

}

# Card 23:

A user should be able to view his cart as he is searching for items to purchase.

**Conversation:**

This function should take place once the user has clicked on the “View Cart” link.

**Conformity:**

* I click on the ‘View Cart’ link and notice that all of the items that are in my cart and that have not yet been purchased are listed.

**Implementation:**

HTML form

SELECT \* FROM orders o, order\_details d WHERE user\_id = ? AND o.id = d.order\_id;

# Card 24:

A user should be able to view his user information.

**Conversation:**

This function should take place once the user has clicked on his or her own username.

**Conformity:**

* I click on my own username and I notice that all of my information is displayed along with various links to edit where they apply.

**Implementation:**

HTML form

SELECT \* FROM users WHERE id = ?;

# Card 25:

A user should be able to change their country or change their email address (in terms of notifications, receipts, etc.).

**Conversation:**

This function should take place once the user has clicked on the “Update Profile” link.

**Conformity:**

* I first click on the link and then enter the valid password for confirmation and also my new country/new email address.

**Implementation:**

HTML form

if (password is correct)

{

UPDATE users SET country\_id /\*or SET email\*/ = ? WHERE id = ?

}

# Card 26:

A user should be able to search for an item by its name.

**Conversation:**

Once the user has pressed “Enter” after typing their search string in the search bar at the top of the page, this function should take place.

**Conformity:**

* I click on the search bar and then type in a string that would match one or more of the items.
* I then press ‘Enter’ to ensure that the item(s) is/are listed.

**Implementation:**

HTML form

SELECT \* FROM items WHERE name LIKE ‘%?%’;

# Card 27:

The user should be able to modify their review(s).

**Conversation:**

If a user has a different point of view about the product that they bought soon after writing the initial review, they should be able to update the review and rating by clicking on the “Edit” link.

**Conformity:**

* I click on the link, modify the review, and click on the ‘Save’ button.
* I then login as a different user to ensure that the review has different text now.

**Implementation:**

SELECT \* FROM orders, order\_details WHERE user\_id = ? AND item\_id = ? AND status LIKE ‘Purchased’;

UPDATE reviews SET rating = ?, content = ‘?’, posorneg = ? WHERE writer\_id = ? AND item\_id = ?;

HTML form

# Card 28:

A user should be able to navigate to a “About Us” Page where he can discover more about the Shoe++ business.

**Conversation:**

There should be a navigation link to the page in the nav bar. The page itself should describe the business.

**Conformity:**

* In the navigation bar, I should see a “About Us” link that takes me to a page that describes the business.
* Once on the page, I should expect to see the company’s vision and description.

**Implementation:**

HTML navbar

HTML page for about us

# Card 29:

A user should be able to change their first name or change their last name.

**Conversation:**

This function should take place once the user has clicked on the “Update Profile” link.

**Conformity:**

* I first click on the link and then enter the valid password for confirmation and also my new first name/new last name.

**Implementation:**

HTML form

if (password is correct)

{

UPDATE users SET fname /\*or SET lname \*/ = ? WHERE id = ?

}

# Card 30:

A user should be able to select what shipping plan they would like to have as they are about to purchase an item (Standard Shipping [+5%], Premium Shipping [+15%], and Yotta Shipping [+25%] are the options).

**Conversation:**

Presented as three audio buttons, this function should take place once one of those three has been clicked upon (Standard Shipping by default).

**Conformity:**

* I click on the ‘Premium Shipping’ audio button and I notice that the total changes.
* I click on the ‘Yotta Shipping’ audio button and I notice that the total changes again.
* I click on the ‘Standard Shipping’ audio button and I notice that the total changes back to its default form.

**Implementation:**

HTML form

Total price with shipping = Total price + Total price \* 0.[Shipping plan percentage increase]

UPDATE orders SET shipping = ? WHERE id = ?;