Simple Shopper

Sprint 5

CSC 4350 - Fall 2019

Group 4

Amaal Abdi, Diego Gonzalez,

Mohammad Mamun, and Kathy Nguyen

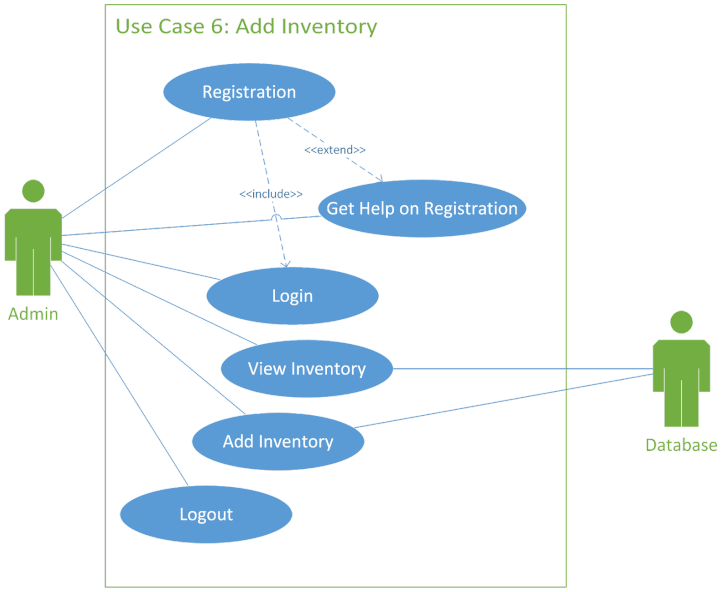
Assignee Name	Email	Task	Duration (hours)	Dependency	Due date	Evaluation
Diego Gonzalez	dgonzalez16@student.gsu.edu	Database, Screenshots, GitHub Project Board,	2 hrs	none	11/15/2019	100%
Kathy Nguyen	knguyen78@student.gsu.edu	Design Pattern Diagram, Class Diagram, Use Case Diagrams	2 hrs	none		100%
Mohammad Mamun (Coordinator)	mmamun1@student.gsu.edu	Testing	5 hrs	none	11/12/19	100%
Amaal Abdi	aabdi11@student.gsu.edu	Coding for website	30 hrs	Code necessary for testing	11/15/2019	100%

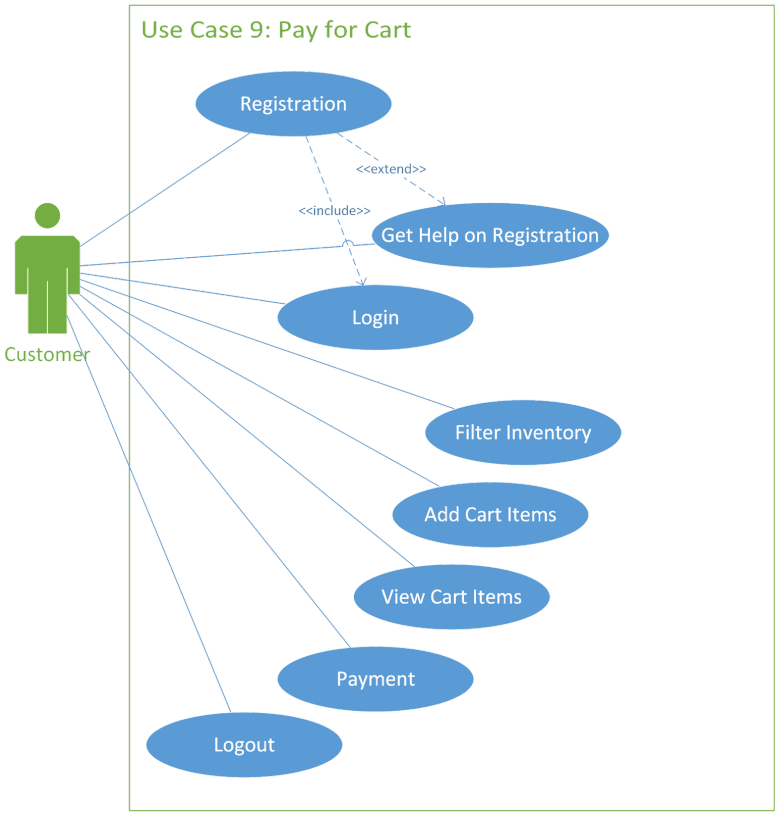
Problem Statement

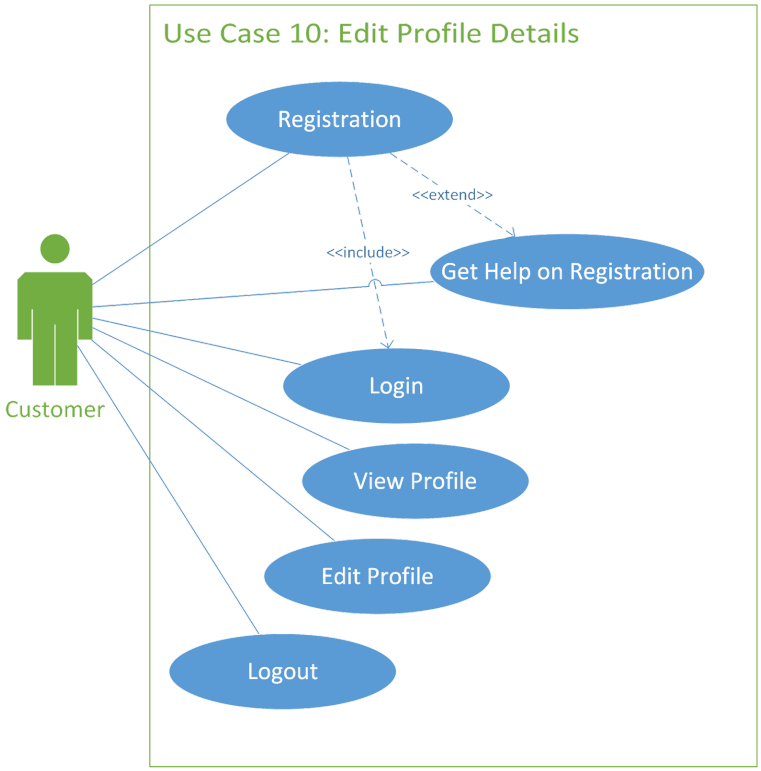
Simple shopping is a project made for people with a restrictive diet. This Product is a way to make grocery shopping simple and convenient for all users. The Product will sort items based on user needs. It will categorize foods and other merchandise into a specific group. For example: Vegans, vegetarians, etc. This Product can solve confusion while navigating through a complex shopping system when looking for specific grocery items. Our main competitors are Thrive, Walmart, Publix, etc., but they mostly sell their branded Product. Simple shopping will try to combine products from all brands and companies and will not require a membership fee. Simple shopping will be simple to produce and will not require substantial resources. This Product can be built using a simple database and will run on almost all platforms. It will be simple to develop, but the actual huge cost will be labor and the maintenance of the Product.

Requirements

Use Case Diagrams







Use Cases

Use Case #: 1
Name of the use Case: Registration
Actors: Customer, Admin, Database
Description:

- Users will be given an option to register for the website in all the web pages available in our system.
- User clicks on the link to the registration page.
- User inputs their Important information (address, credit card info (optional), name , date of birth, diet)
- Checks if the information given valid or invalid (credit card / address/email address/ username/ password).
- If the information is given not valid, display error and let them try again.
- Send the users verification message via email.

Exception Path: If the user does not want to register or their information is already in our database.
Alternative Path: There is no need to be registered to shop.
Pre-condition: Be on website.

Post-condition: The users are registered into the system and their information is saved in our database.

Use Case #: 2
Name of the Use Case: Log In
Actors: Customer, Admin, and Database
Description:

- The user will be given the login page, once accessing out site.
- The user will enter email and password.
- After the inputs are complete, the user will click login.
- If successful, the user will be brought to our inventory page.
- If unsuccessful, the user will be prompted with an error message to try again.
- The information will be validated with database.

Exception Path: If the user does not have an account the will need to Sign Up
Alternate Path: only one path
Pre-condition: Account with valid login credentials
Post-condition: getting to the inventory page.

Use Case #: 3
Name of the Use Case: Log Out
Actors: Customer, Admin
Description:

- At any point of viewing our website the user should be able to navigate to the log out button
- After ordering something the user will be prompted to either log out or continue shopping with us.

Exception Path: internet disconnection which would automatically log out user.
Alternate Path: User can either log out or keep shopping
Pre-condition: Internet, Computer, interfaces to interact with our website
Post-condition: Confirm logout/navigate back to home page

Use Case #: 4
Name of the Use Case: View Inventory
Actors: Admin, Customer
Description:

- After login, the page redirects to inventory page.
- Displays all of the inventory of the grocery website.

Exception Path: none

Alternate Path: If the user chooses to add filtering to the inventory (see use case #5), the inventory will appear differently depending on the level of filtering. If the user is a customer, each item of the inventory will have an “Add to Cart” button associated with it, which will take the user to the “View Cart” page (see user case #7). If the user is an admin, each item will have an “Edit” button next to it, and the page will include an “Add New Inventory” button at the top (see use case #6).
Pre-condition: The user should be logged in as an admin.
Post-condition: Display of inventory items, redirect to inventory editing page

Use Case #: 5
Name of the Use Case: Filter Inventory
Actor: Customer
Description:

- Customer have the option to browse shopping items in our site
- Customer clicks on the button/ link to pages.
- Clicks on tag / categories.
- Error if too many tags presents.
- Prompt to remove few tags (7 tags total)
- Inventory is sorted based on the input given by the customer.
- The link displays the page with filtered items.

Exceptional path: none
Alternate path: entering the direct address to the page.
Pre-condition: Needs to be on our site.
Post-condition: Entering filtered page.

Use Case #: 6
Name of the Use Case: Add Inventory
Actors: Admin, Database
Description:

- If the account is not admin level, show error.
- The admin will identify which inventory item needs updating when new shipment arrives.
- If the item is new and not yet in the system, admin will add inventory by entering the name, quantity, and description.
- The admin will select which item to update or add.
- The admin will enter in a new quantity for the item.
- The admin will repeat this process for all other inventory items that need updating.

Exception Path: none
Alternate Path: The admin can cancel editing at any time by returning to the inventory page.

Pre-condition: Account used to login must be admin level.
Post-condition: Confirmation of changes by viewing current inventory levels.

Use Case #: 7
Name of the Use Case: View Cart Items
Actors: Customer
Description:

- The customer clicks on a button that will take them to a list of the items they put into their cart.
- The page will show the total price of all items.
- The page will also have links to purchase items or to return to the inventory.

Exception Path: none
Alternate Path: There are two links, labeled “Continue Shopping” and “Purchase Items”. If the user clicks “Continue Shopping”, the page will redirect to the inventory page (use case 4). If the user clicks “Purchase Items”, the page will redirect to the payment page (use case 9).
Pre-condition: The user must be logged in.
Post-condition:

Use Case #: 8
Name of the Use Case: Add Cart Items
Actors: Customer
Description:

- Each item in the inventory will have an “Add to Cart” Button next to it.
- If the user clicks on one of these buttons, the page will redirect to the “View Cart” page.
- The item that was clicked will be added to the cart, and its price will be added to the total price.

Exception Path: none
Alternate Path: none
Pre-condition: The user must be logged in to add items to the cart.
Post-condition: The page redirects to the “View Cart” page. The item is added to the customer’s cart, and the price of the item is added to the total price.

Use Case #: 9
Name of the Use Case: Pay for Cart Items
Actors: Customer
Description:

- The customer clicks on a button in the “View Cart” page to purchase the items in the cart.
- This takes the user to a payment form, where user must enter card information in order to pay for the items.

Exception Path: If the card information is incorrect, the page will indicate what fields are incorrect. The user will have to enter this information again.
Alternate Path: User may enter information for another card.
Pre-condition: There must be at least one item in the cart
Post-condition: Confirmation of purchase.

Use Case #: 10
Name of the Use Case: Edit Profile
Actors: Customer
Description:

- Customer will select edit profile to change any information that is displayed on their profile.
- System will display all editable fields where customer will enter in new information.
- System will update when customer saves the new details.

Exception Path: If any fields are left blank, customer will be prompted to enter a value for blank fields.
Alternate Path: none
Pre-condition: Customer must be logged in with a valid account.
Post-condition: Confirmation of changes by viewing profile with new details.

Use Case #: 11
Name of the Use Case: View Profile
Actors: Customer
Description:

- Customer can view profile details after logging in.
- Details to be displayed may include: Name, Age, Diet, and other details that were initially entered during the registration process.

Exception Path: none
Alternate Path: Customer may choose to edit any details by selecting Edit Profile (see use case # 10)
Pre-condition: Customer must be logged in with a valid account.
Post-condition: Show profile details page.

Use Case #: 12
Name of the Use Case: Get Help on Registration
Summary: The users needs help with registration.
Actors: Customer
Description:

- The user needs help on the registration page.

- Give them example for each specific field (username, Password, date of birth, etc.)
- If the user still having problems give them contact information for customer support.

Exceptional path: none
Alternative path: not registering;
Pre-condition: The user is in registering phase.
Post-condition: The user was able to register for the site

System Requirements

Requirement #: 1 Use Case #: 1
Introduction: User registering for the website to buy items.
Rationale: User registration form for customers to sign up for an account.
Inputs: Email address, Password, Credit card information and dietary information.
Requirements Description:

- The email address must be valid and unique.
- The user name can have letters and numbers, but no space or special characters
- The username must start with a letter
- Password must be 8-20 characters and must contain upper and lower case letters and special characters.
- Credit card information must be valid.
- If the data is not valid give them example of format
- If the information is not valid display error and let the user try again.
- One the valid information is given save all the data to our database

Outputs: Registered user.
Data saved in our database

Test Case ID: 1.1
Description: Trying to sign up using valid information displayed in input boxes.

Test Inputs: Text inputs for username (first name), last name, password, email, phone number, address, city, state, and zip code
Expected Results: Successful registration resulting in displaying a link to login page.
Dependencies; none
Initialization: new email and password.
Test steps: Inputting valid email and password not in the database.
Click submit (Success)

Test Case ID:1.2
Description: Trying to sign up using invalid/linked email and password
Test Inputs: invalid email or linked email to another user.
Expected Results: error message should pop up.
Dependencies; none
Initialization:submit invalid email
Test steps: submit invalid email address.
The user is gets error and prompted to enter again.

Test Case ID:1.3
Description: Trying to sign up using invalid password.
Test Inputs: invalid password
Expected Results: error message / example password
Dependencies; none
Initialization:submit invalid password.
Test setup: entering invalid password.
Click submit Getting error message and example password is displayed

Requirement #: 2 Use Case #: 2
Introduction: Logging In
Rationale: Login process for customers to login to the website to buy food.
Inputs: Username, password, signin button
Requirements Description:

- User will enter username and password.
- If the credentials are correct, the user is logged in and now has access.
- If the credentials were incorrect, an error will show and prompt the user to try again.

Outputs: Error or successful log in

Test Case ID:2.1
Description: Testing valid username and password on data table
Test Inputs:valid username and password on data table

Expected Results: successful login, redirect to main inventory page
Dependencies; The information from the login form must match the information of a registered user in the database.
Initialization: submit valid username and password
Test steps: Enter username and password. Click the login button.

Test Case ID: 2.2
Description:Enter invalid username and password
Test Inputs:invalid username and password
Expected Results: display error message
Dependencies; none
Initialization:input invalid username and password
Test steps: Input invalid username and password.
Click the Login button and Error is displayed

Test Case ID: 2.3
Description:Enter blank username and password
Test Inputs: blank username and password
Expected Results: display error message
Dependencies; none
Initialization: enter blank username and password
Test steps: Leave username and password blank. Click the Login button.

Requirement #: 3 Use Case #: 3
Introduction: logout
Inputs: clicking logout button
Requirements Description:

- User will click the logout button to log out.
- The user should be logged out of the system.
- This will end the users access to cart, inventory, personal information, etc.

Outputs:
The user can no longer access private information from the website, unless logged back in.

Test Case ID: 3.1
Description: User successfully logs out of the site
Test Inputs: logout button
Expected Results: successful logout message and redirect of current page to home page.
Dependencies: User logged in
Initialization: click logout button

Test steps: Click logout button

Test Case ID: 3.2
Description: User is able to access and view pages that only appear for a logged in user, meaning the user did not log out successfully and the session was not destroyed.
Test Input: Logout button, url of webpages, buttons for pages that require log in, such as the profile page and inventory showing the username on the top of the page.
Expected Results: Username is still present on the inventory page, and the pages that require log in are accessible and show previous user information.
Dependencies: User must have previously been logged in
Initialization: Click login button
Test Steps: Click log out button. Enter url of inventory page. Click on profile button or any other button that requires a user to be logged in.

Requirement #: 4 Use Case #: 4
Introduction: Display inventory so that the customer or admin can view items.
Rationale: Show all food products and current quantity for admins to know when to order from supplier.
Inputs: Button click
Requirements Description:

- User must be logged in
- Items should have an “Edit” button to update quantity
- The page should display all inventory items determined selected filter options, if filter is used (see use case #5)

Outputs: List of available inventory items.

Test Case ID: 4.1
Description: Customer or admin can view and browse through entire inventory (homepage) and categories by clicking a button.
Test Inputs: Home button, View Inventory button
Expected Results: Display all items in the database along with text that welcomes the current user to the page.
Dependencies: User must be logged in
Initialization:View inventory Button
Test steps: If not logged in, log in to view inventory. On other pages of the website, click button related to inventory page.

Test Case ID: 4.2

Description: User is unable to view the inventory
Test Inputs: View inventory Button, login
Expected Results: Page is not redirected to full inventory page, entire inventory does not display.
Dependencies: Must be logged in
Initialization: Click inventory Button
Test steps: If not logged in, log in. If logged in, click on the links associated with the inventory page.

Requirement #: 5 Use Case #: 5
Introduction: Filtering inventory
Rationale: Customer should be able to view food items that are specific to their diet.
Inputs: Tags, option selection
Requirements Description:

- Every item will be labeled with tags that describe what it is safe for (vegetables for vegetarians, dairy product will be labeled with dairy tag. Etc.
- Tags cannot contradict each other(example: meat and vegetarian cannot be selected at the same time).
- Display items with those tags.
- Removing tag will re-filter the list.

Outputs: Filtered list of items.

Test Case ID: 5.1
Description: Testing filter feature with no filter
Test Inputs: no filter
Expected Results: display all items
Dependencies: none
Initialization:no filter picked
Test steps: no filter picked.

Test Case ID: 5.2
Description: Testing filter feature with two or more
Test Inputs: filter picked
Expected Results: display all items
Dependencies: no contradicting tags picked (example meat and vegetarian)
initialization:no filter picked
Test steps: pick filter 1 and filter two
Should filter item based on filter

Test Case ID: 5.3
Description:picking contradicting tags (meat and vegetarian)
Test Inputs: meat and vegetarian
Expected Results: ERROR
Dependencies:None
Initialization: input tags meat and vegetarian
Test steps:
Click tags meat and vegetarian

Requirement #: 6 Use Case #: 6
Introduction: Admin updating inventory.
Rationale: Admin needs to be able to update quantity to keep up with stock.
Inputs: New inventory values (name, quantity, description, diet)
Requirements Description:

- Accepts the new inventory value of items.
- Updates that value in the database.

Outputs: Saves new inventory values and displays on the view inventory page.

Test Case ID: 6.1
Description: Enter correct information for editing fields
Test Inputs: “Edit” button and change the stock values to (0-90000)
Expected Results: edited inventory and updated result
Dependencies: Admin account logged in
Initialization: change values
Test steps: Click “Edit” button next to an item. Edit inventory fields. Click the Save button.

Test Case ID: 6.2
Description: Enter incorrect information for the edit fields
Test Inputs: “Edit” button, change the stock values to (0-90000), enter invalid information for all other fields.
Expected Results: Page shows an error in editing. Inventory does not change in the database.
Dependencies: Access to the database.
Initialization: Click Save button
Test Steps: Click “Edit” button next to an item. Edit inventory fields with wrong information. Click the Save button.

Requirement #: 7 Use Case #: 7
Introduction: Viewing cart items for review before purchase or before continuing shopping.

Rationale: Customer needs to be able to confirm that their cart is complete before paying. Show all items that customer has previously selected.
Inputs: Button click
Requirements Description:

- Displays items in cart

Outputs: Display of current cart items.

Test Case ID: 7.1
Description: Logged in user clicks view cart button and views the items in their cart
Test Inputs: View Cart button
Expected Results: display all items added to cart along with the price of all items
Dependencies: User must be logged in as a customer
Initialization: add items to cart
Test steps: Click View Cart button

Test Case ID:7.2
Description: Clicking on the view cart button while logged out
Test Inputs: View Cart button
Expected Results: the user will see nothing in the cart
Dependencies: none
Initialization: log out if a user is logged in
Test steps: If logged in, log out of system and enter url for inventory page. Click on the View Cart button.

Requirement #: 8 Use Case #: 8
Introduction: Adding desired items to customer cart before purchase.
Rationale: Items must be added to cart for purchase.
Inputs: Button click
Requirements Description:

- Clicking on the add cart button adds the item to the database
- Displays the new item in the view cart page

Outputs: Redirect to view cart page, new item entered into customer's cart, addition of total price

Test Case ID: 8.1
Description: adding items to cart.
Test Inputs:Add to Cart button

Expected Results: Item associated with the Add to Cart button will be added to the cart, and the page redirects to view cart page. New item appears on page. Item added to order database.
Dependencies: User must be logged in as a customer
Initialization: click Add to Cart button from item inventory
Test steps: Click on the Add to Cart button on the inventory page

Test Case ID: 8.2
Description: A user that has not logged in clicks Add to Cart Button on the inventory page.
Test Inputs: Add to Cart button
Expected Result: User is unable to add items to the cart. The page does not redirect to the view cart page. The item is not added to the order database.
Dependencies: none
Initialization: Go to the inventory page
Test steps: Click on the Add to Cart button on the inventory page.

Requirement #: 9 Use Case #: 9
Introduction: Entering information to purchase cart items
Inputs: Form input, submit button
Requirements Description:

- The cart must have at least one item in it
- Card information must be correct
    - All card fields must be filled
    - Card number must match the pattern for common card types (Visa, Mastercard, Discover, etc)
    - User must enter mailing address and billing address information

Outputs: Page showing confirmation of purchase.

Test Case ID: 9.1
Description: Purchase with no items
Test Inputs: Purchase button
Expected Results: Error
Dependencies: User must be logged in as a customer
Initialization: Purchase button clicked
Test steps: click purchase button

Test Case ID: 9.2
Description: Purchase with items
Test Inputs: Purchase button

Expected Results: Jump to payment information page
Dependencies: User must be logged in as a customer
Initialization: Purchase button clicked
Test steps: click purchase button

Test Case ID: 9.3
Description: entering valid card number , security number and expiration date as well as address information..
Test Inputs: valid card information and billing address.
Expected Results: Page redirects to a page that confirms the purchase.
Dependencies: Must have items in cart.
Initialization: Valid card information and billing address.
Test steps: enter valid card information and billing address.
Click submit

Test Case ID: 9.4
Description: Invalid card information used as input.
Test Inputs: Invalid card information in the input fields
Expected Results: Error message indicating the errors on the page
Dependencies: Must have items in cart.
Initialization: Enter invalid card information
Test steps: Enter invalid card information. Click submit button.

Test Case ID: 9.5
Description: Enter blank information for each field in the purchase form
Test Inputs: Empty form fields, submit button
Expected Results: Error message indicating the errors on the page.
Dependencies: Must have items in cart
Initialization: Enter blank credit card information and address information.
Test steps: Leave all fields blank. Click submit button
Click Submit

Requirement #: 10 Use Case #: 10
Introduction: Customer can edit profile details
Rationale: Customer information may change, so they will need the ability to make changes to their accounts.
Inputs: Customer selects the option to edit profile details
Requirements Description:

- All editable fields will show on edit page.
- Information to edit includes: name, age, dietary restrictions, address, and phone number.
- Customer will enter any new details to replace old information.

Outputs: New details are saved and will display on the view profile page.

Test Case ID: 10.1
Description: Customer edits the profile information with valid user information
Test Inputs: Profile form fields with valid information, submit button.
Expected Results: The information changes in the profile page according to what the user entered. User information changes in the database.
Dependencies: Must be logged in
Initialization: User clicked View Profile link
Test steps: Click Edit button. Chang field information with valid information, as the user did in registration. Click the Save button.

Test case ID: 10.2
Description: Customer edits the profile information with invalid information.
Test inputs: Profile form fields with invalid information, submit button
Expected Result: An error message appears, informing the user that they should try again. The information submitted is not saved into the database.
Dependencies: Must be logged in
Initialization: User clicked the View Profile link
Test Steps: Click Edit button. Change field information with invalid information. Click the Save button.

Requirement #: 11 Use Case #: 11
Introduction: Customer can view profile details
Rationale: Customer should be able to see their current information to determine if any changes need to be made.
Inputs: Customer clicks link/button to view profile
Requirements Description:

- Page will show all details previously entered during registration or profile update.
- Information to display includes: name, age, dietary restrictions, address, and phone number.

Outputs: The profile page will show all the details that the customer previously entered.

Test Case ID: 11.1
Description: Logged in user clicks the View Profile page.
Test Inputs: View Profile button

Expected Results: Page redirects to profile page
Dependencies: Must be logged in.
Initialization: Click View Profile button
Test steps: Click View Profile button.

Test Case ID: 11.2
Description: User that is not logged in clicks the View Profile page.
Test Inputs: View Profile button
Expected Result: The profile page will not display user information.
Dependencies: none
Initialization: User should be on the inventory page
Test Steps: If a user is logged in, log out. Enter the url for the inventory page. Click on the View Cart button.

Requirement #: 12 Use Case #: 12
Introduction: Get help on registration
Rationale: Provide extra assistance for users wanting to create an account.
Inputs: Customer clicks link.
Requirements Description:

- Customer will be taken to a page with example registration form.
- Display customer support info.
- If the customers wants more help. They can contact customer support.

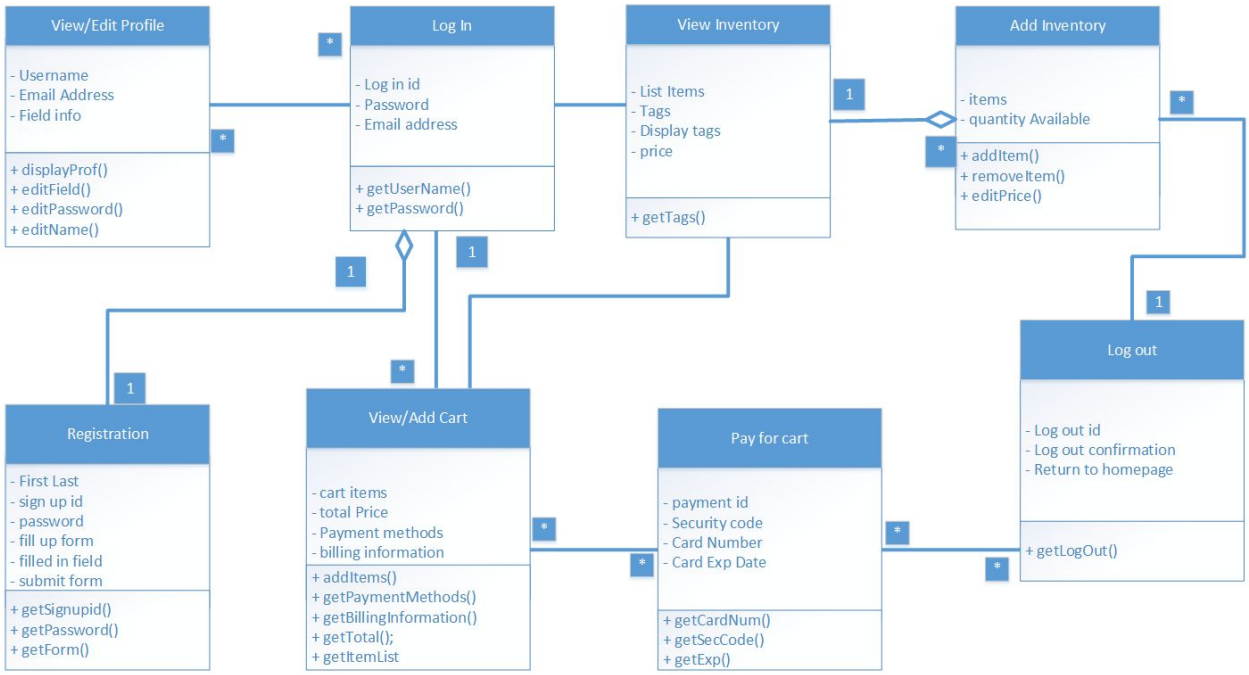
Outputs: Should help with registration.

Test Case ID: 12.1
Description: Click on help button on the registration page.
Test Inputs: Click on help button on the registration page.

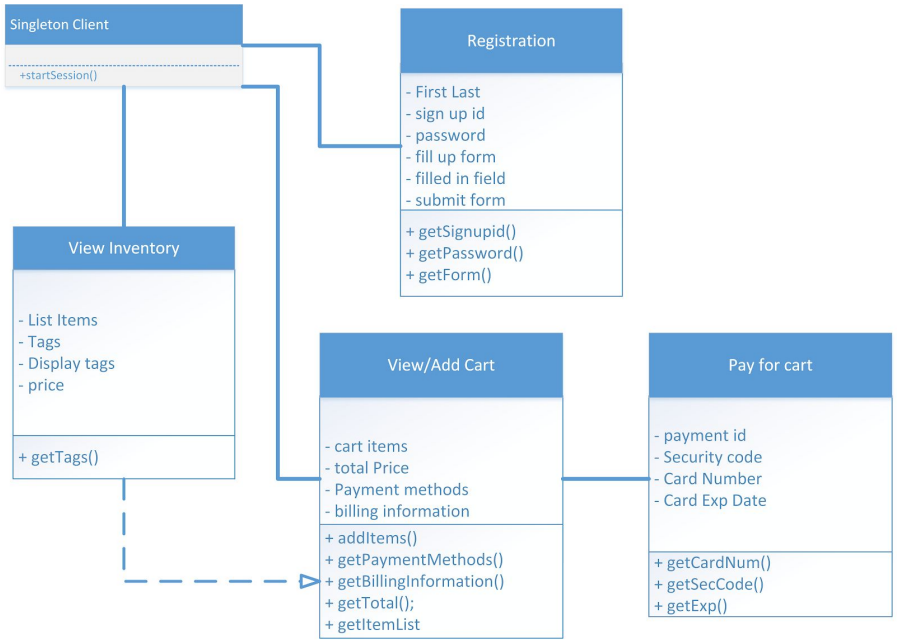
Expected Results: Display help page
Dependencies: none
Initialization: Click on help button on the registration page.
Test steps: Click on help button on the registration page.

System Modeling

Class Diagram



Singleton Model Class Diagram



Creational Design Pattern

Database Specifications

Database Management System: MySQL

Tables:

- Customers: customerId(Primary Key), customer_first_name, customer_last_name, customer_email_address, customer_password, customer_phone_number, customer_address, customer_city, customer_state, customer_zipcode, customer_dob
- Payments: paymentId(Primary Key), card_num, card_exp, card_ccv
- Food Inventory: food_id(Primary Key), food_desc, stock, price, food_image, keto_diet, paleo_diet, atkins_diet, vegitarian_diet, pescatarian_diet, vegan_diet, raw_diet, mediterranean_diet
- Customer_orders: orderId(Primary Key), customerId(Foreign Key), order_date, status

Database Implementation Code:

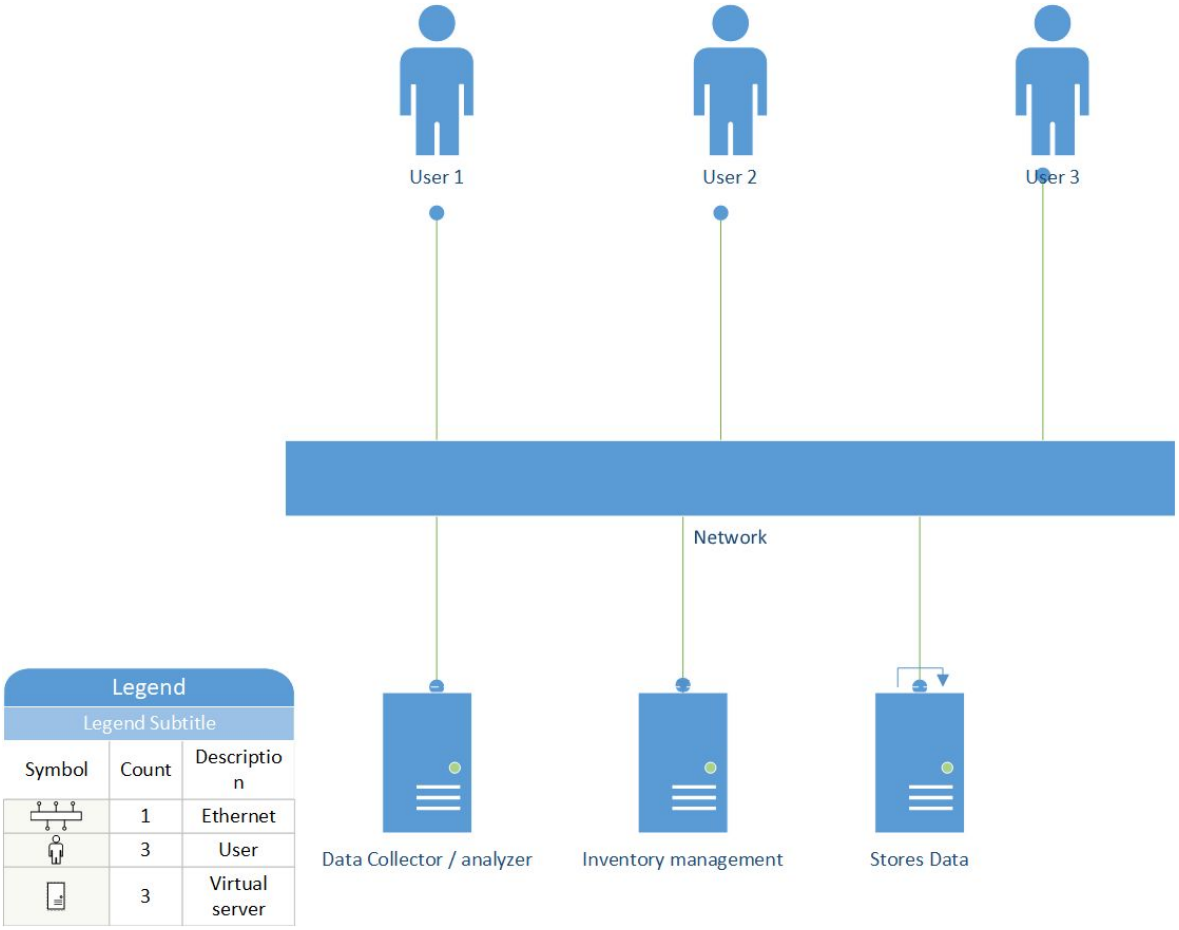
```
CREATE TABLE `simpleshopper`.`payments` (
  `paymentId` INT NOT NULL AUTO_INCREMENT,
  `card_num` VARCHAR(16) NULL,
  `card_exp` VARCHAR(4) NULL,
  `card_ccv` INT NULL,
  PRIMARY KEY (`paymentId`));

CREATE TABLE `simpleshopper`.`food_inventory` (
  `food_id` INT NOT NULL,
  ` food_desc` VARCHAR(45) NULL,
  `stock` INT NULL,
```

```
  `price` DECIMAL(5,2) NULL,
  `food_image` VARCHAR(100) NULL,
  `keto_diet` TINYINT NULL,
  `paleo_diet` TINYINT NULL,
  `atkins_diet` TINYINT NULL,
  `vegetarian_diet` TINYINT NULL,
  `pescatarian_diet` TINYINT NULL,
  `vegan_diet` TINYINT NULL,
  `raw_diet` TINYINT NULL,
  `mediterranean_diet` TINYINT NULL,
  PRIMARY KEY (`food_id`));

CREATE TABLE `simpleshopper`.`customer_orders` (
  `orderId` INT NOT NULL,
  `order_date` VARCHAR(45) NULL,
  `status` VARCHAR(45) NULL,
  PRIMARY KEY (`orderId`));

CREATE TABLE `simpleshopper`.`customers` (
  `customerId` INT AUTO_INCREMENT NOT NULL,
  `customer_first_name` VARCHAR(30) NULL,
  `customer_last_name` VARCHAR(30) NULL,
```

```
`customer_email_address` VARCHAR(50) NULL,
`customer_password` VARCHAR(20) NULL,
`customer_phone_number` VARCHAR(10) NULL,
`customer_address` VARCHAR(50) NULL,
`customer_city` VARCHAR(50) NULL,
`customer_state` VARCHAR(50) NULL,
`customer_zipcode` VARCHAR(10) NULL,
 PRIMARY KEY (`orderId`));
```

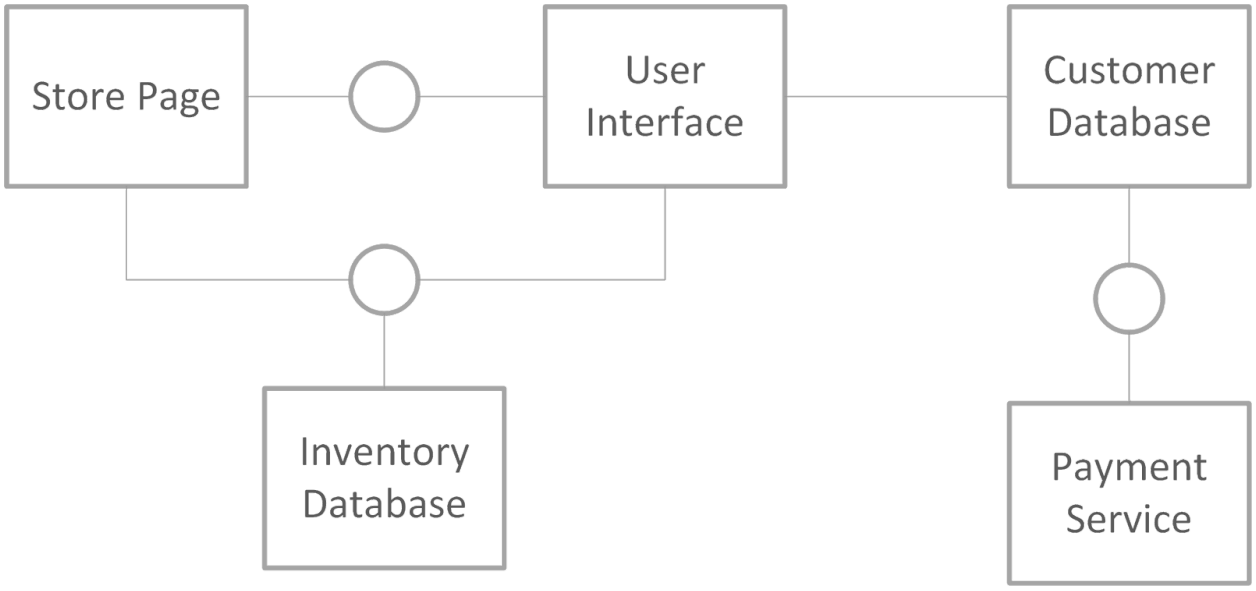
Architecture Modeling

Client-Server

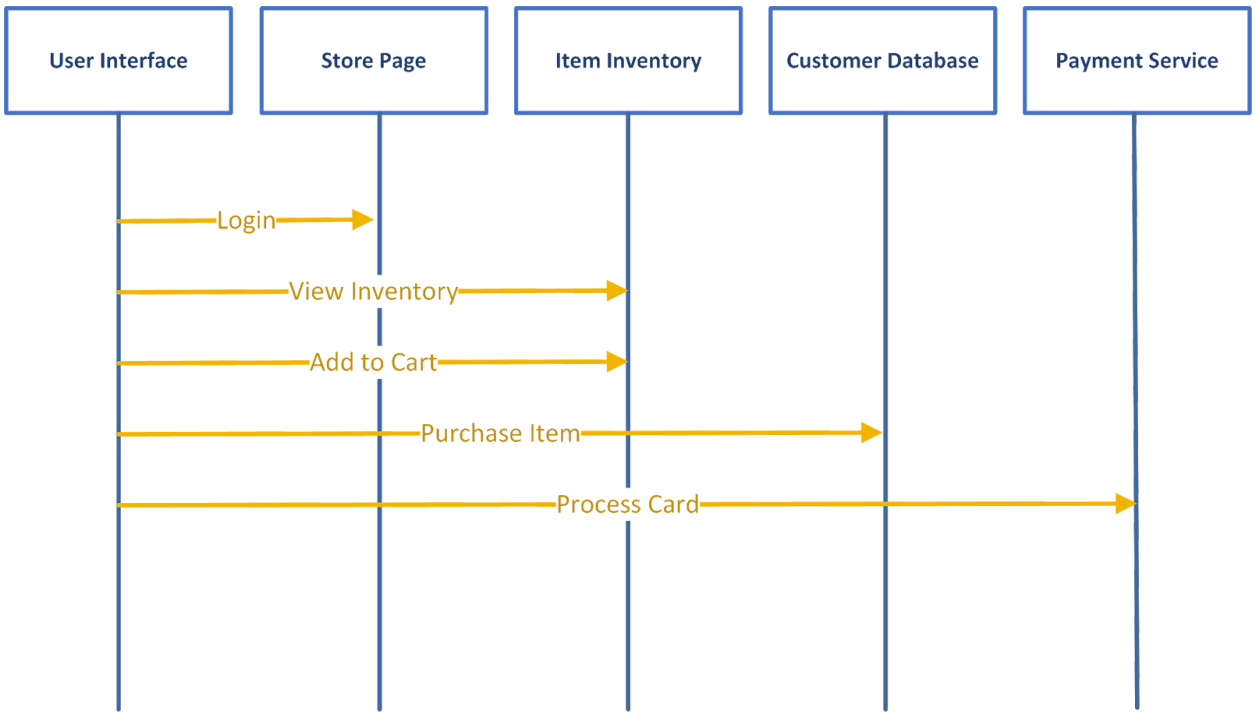


Customers will connect to the client using their own computer via web-browser and send requests to the server. Server will collect data from customers, process their requests, and store them in a database.

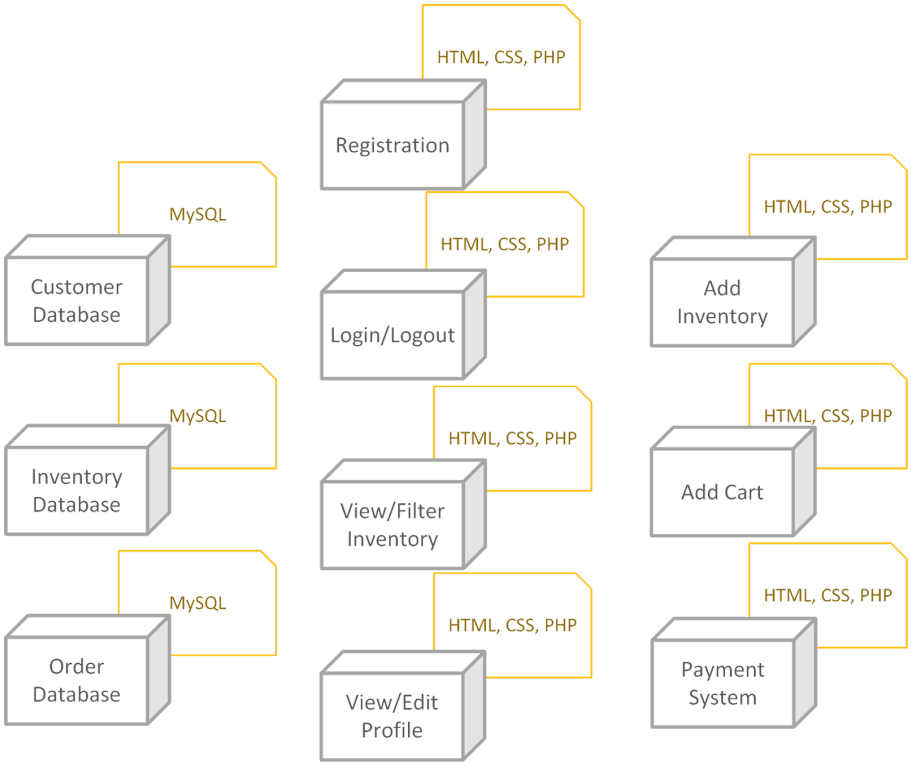
Logical View



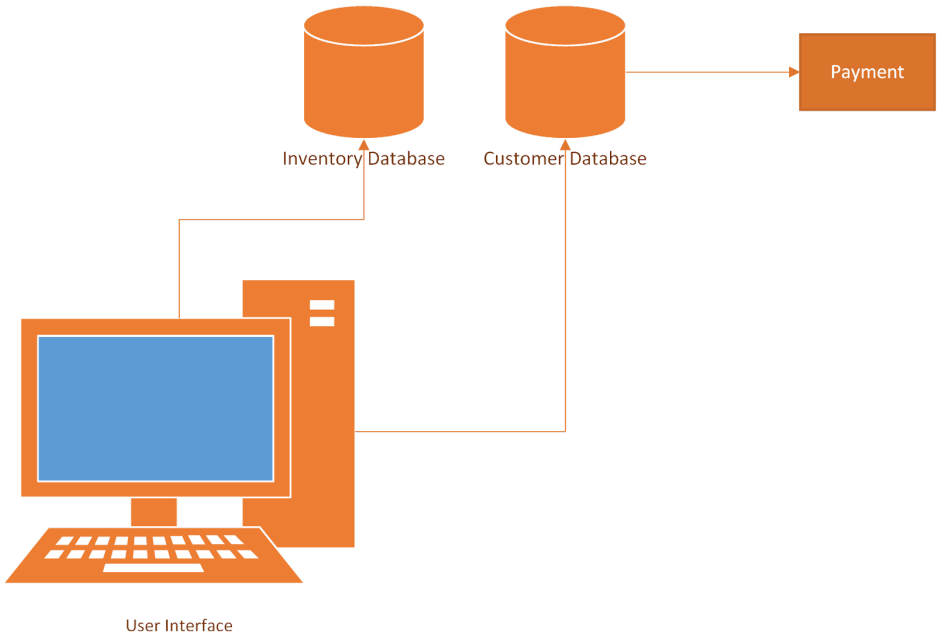
Process View



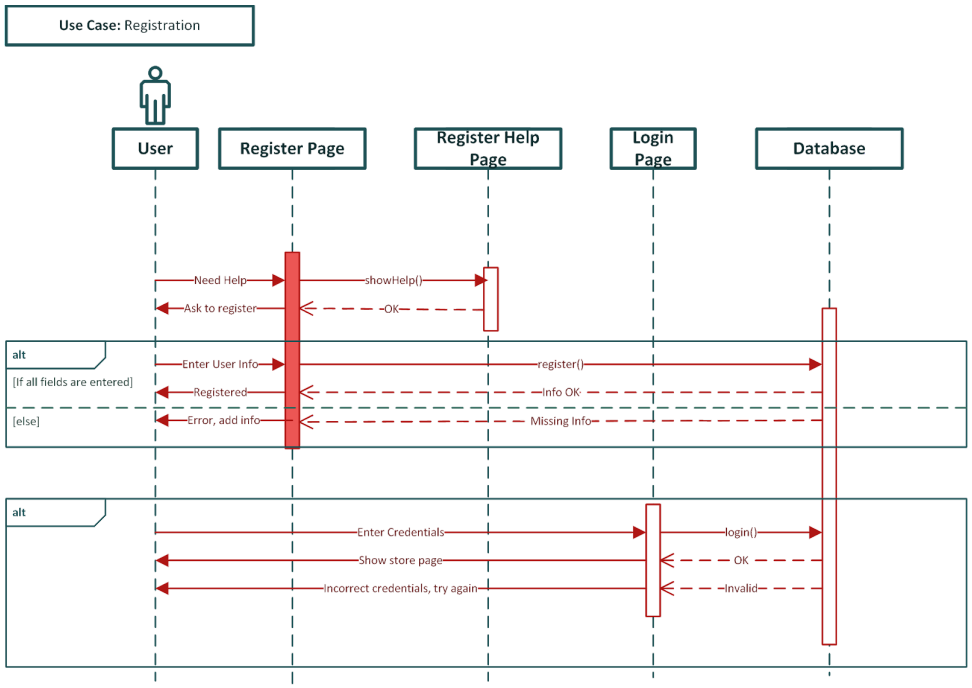
Development View



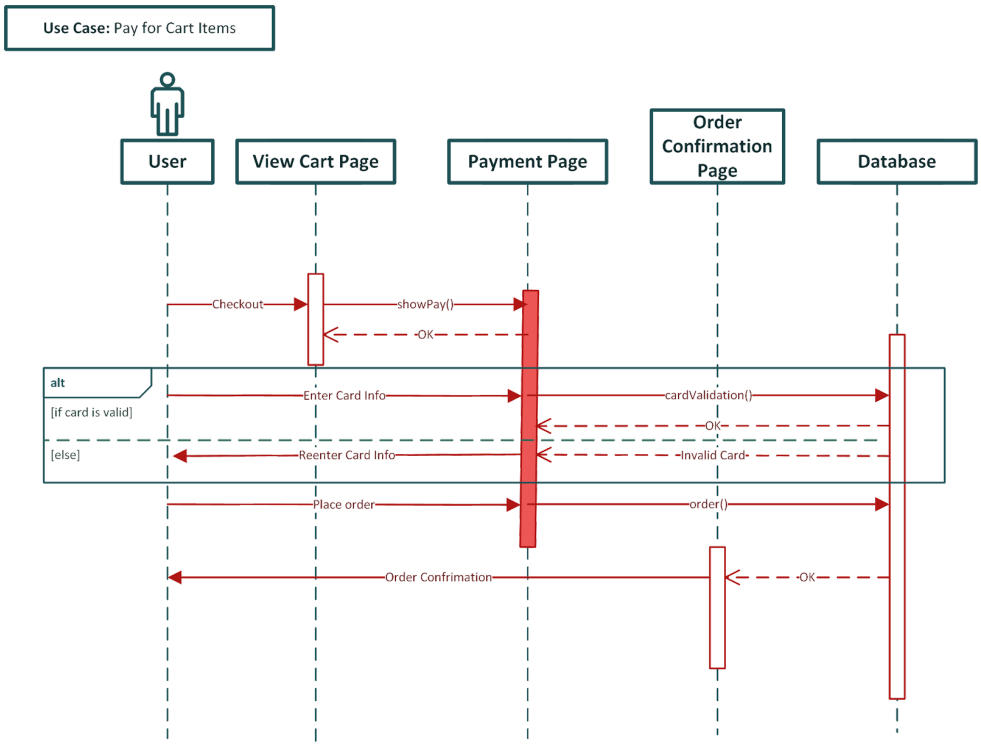
Physical View



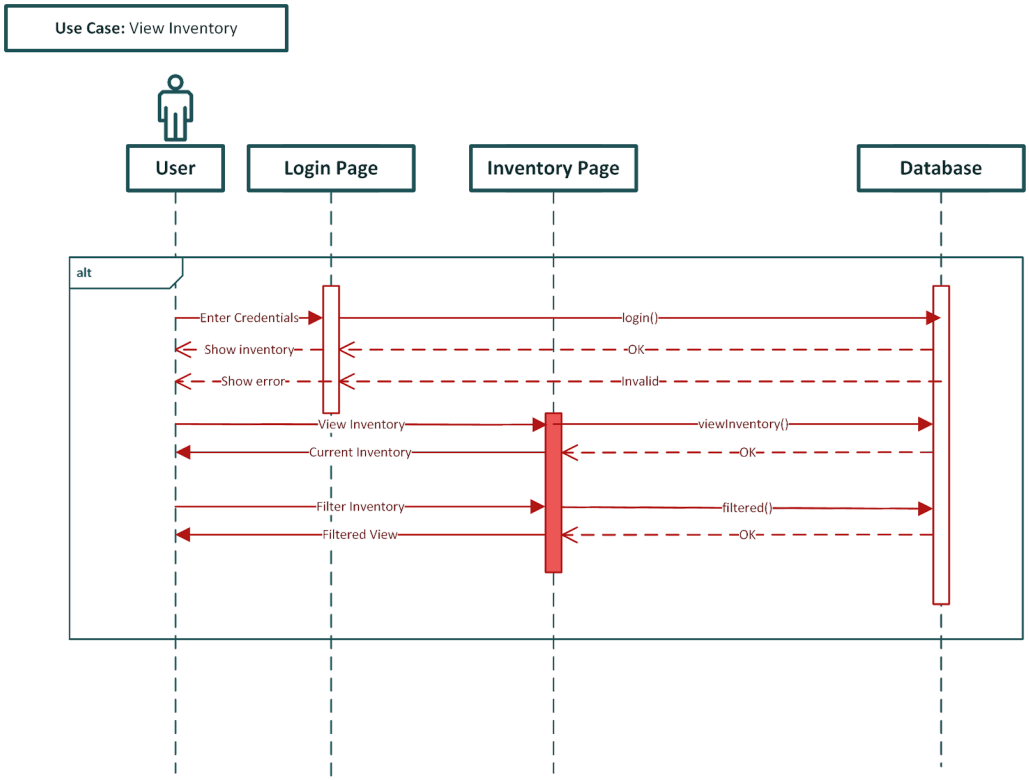
Behavioral Modeling



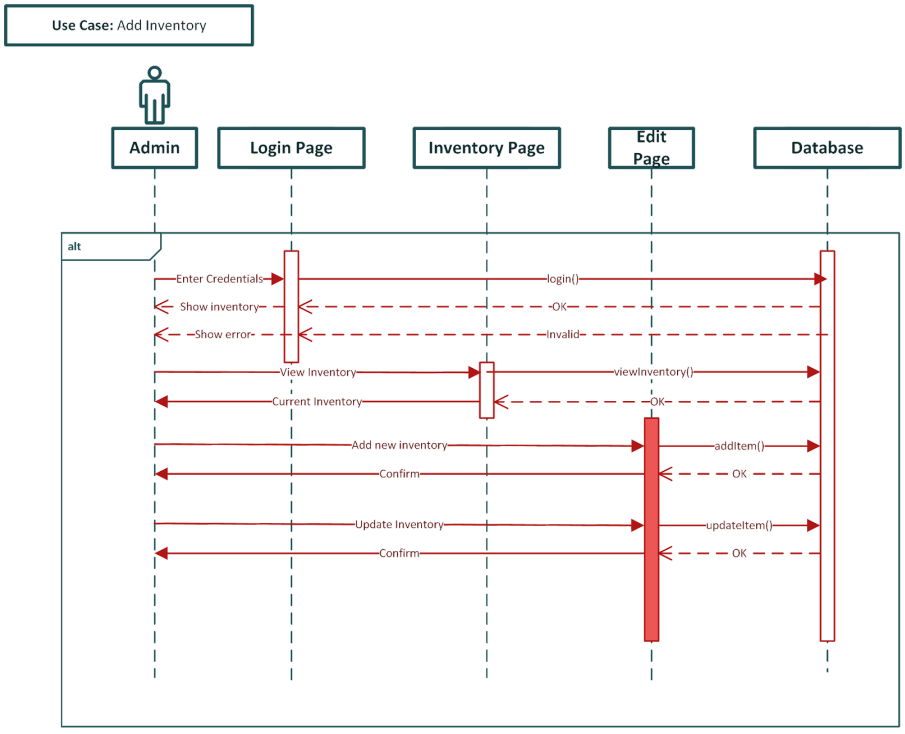
Customer will enter details to register for an account. If the customer is having trouble with registration, they can access the help page. After successful registration, customer is brought to the login page.



Customer will review cart items and checkout. Once on the payment page, customer will enter payment information. Order confirmation will show once successfully paid for items in cart.



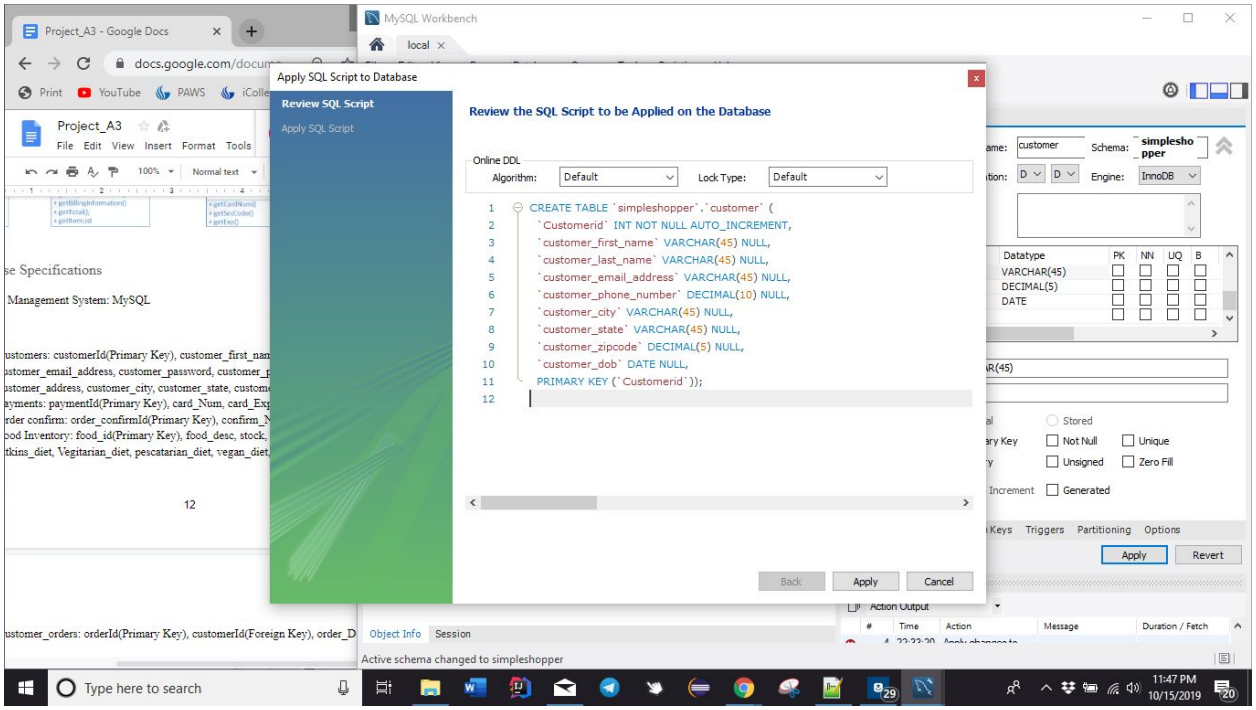
Customer can view all items after login. On inventory page, customers can filter items to select to add to cart.

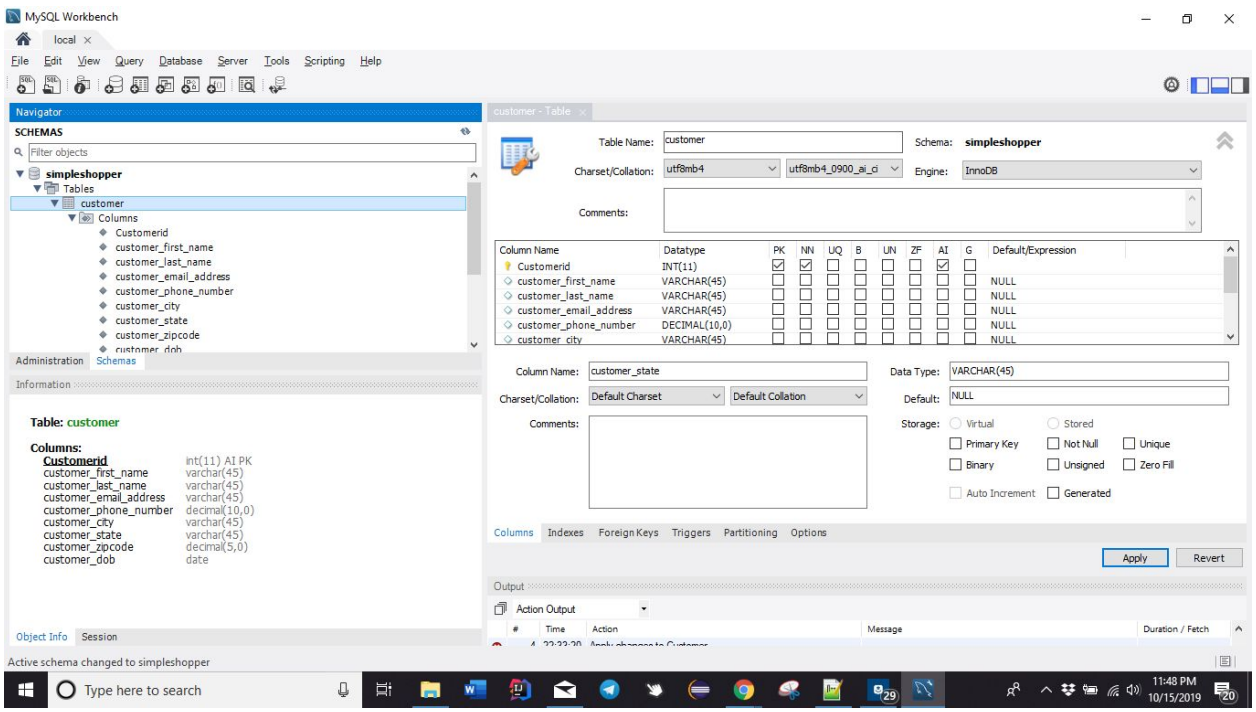


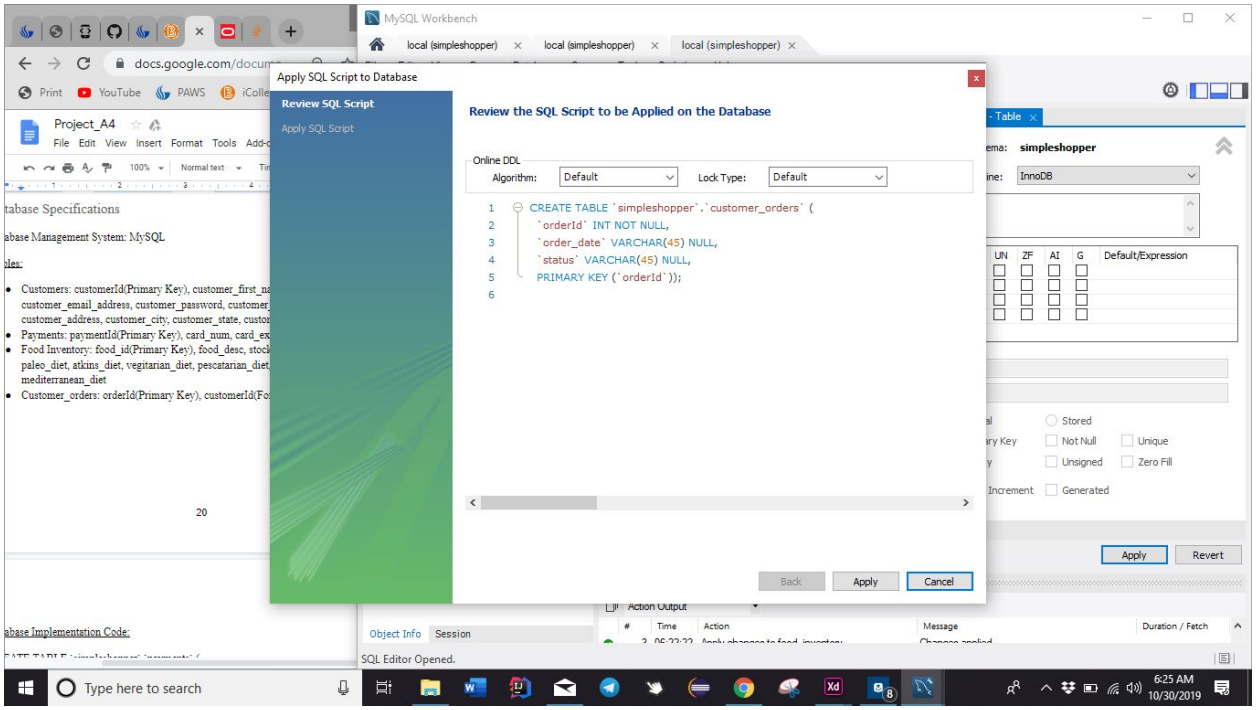
Admin account can view items after login. Admin also has option to edit inventory values of new and existing items.

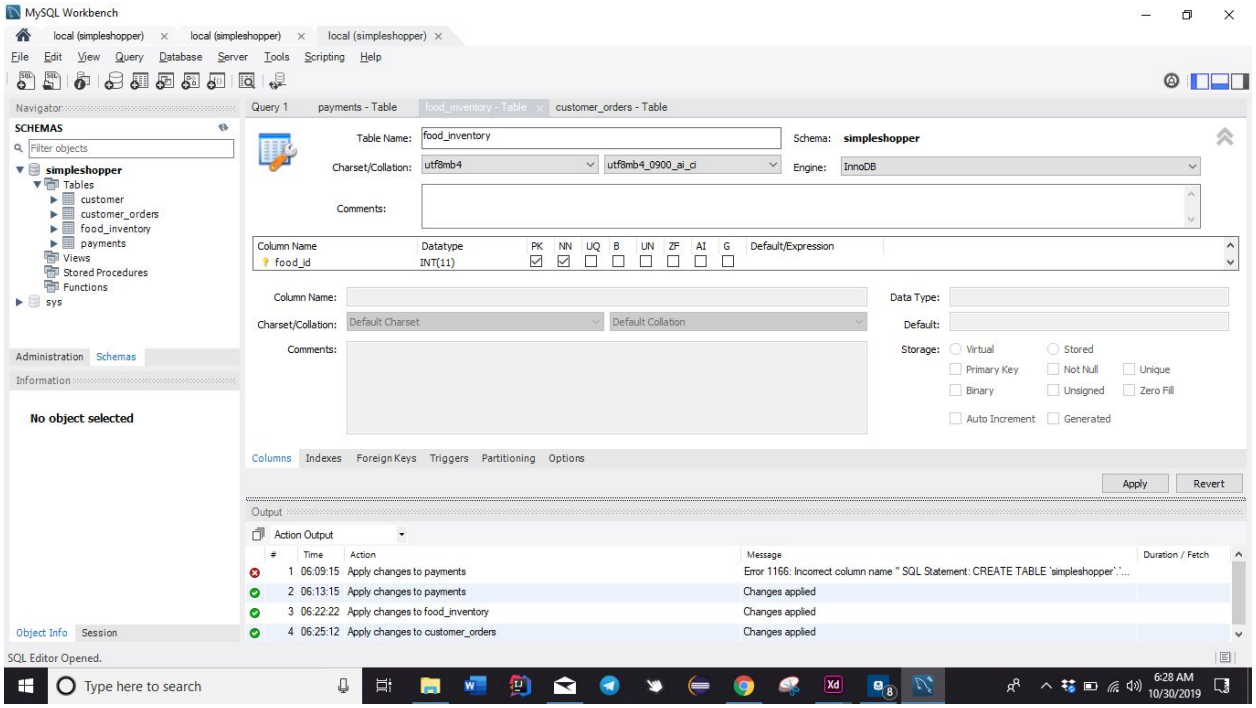
Implementation

Database Design Implementation






MySQL Workbench
Apply SQL Script to Database
Review SQL Script
Apply SQL Script
Review the SQL Script to be Applied on the Database
Online DDL
Algorithm:
Default
Lock Type:
Default
1 CREATE TABLE `simpleshopper`.`customer_orders` (
2 `orderId` INT NOT NULL,
3 `order_date` VARCHAR(45) NULL,
4 `status` VARCHAR(45) NULL,
5 PRIMARY KEY (`orderId`));
6
Back
Apply
Cancel
Project_A4
Apply
Revert
Object Info
Session
SQL Editor Opened.
Type here to search
6:25 AM
10/30/2019


MySQL Workbench
File Edit View Query Database Server Tools Scripting Help
SCHEMAS
simpleshopper
Tables
customer
customer_orders
food_inventory
payments
Views
Stored Procedures
Functions
sys
No object selected
Table Name: food_inventory
Schema: simpleshopper
Charset/Collation: utf8mb4
utf8mb4_0900_ai_ci
Engine: InnoDB
Comments:
Column Name
Datatype
PK NN UQ B UN ZF AI G
Default/Expression
food_id
INT(11)
Column Name:
Charset/Collation:
Default Charset
Default Collation
Comments:
Data Type:
Default:
Storage:
Virtual
Stored
Primary Key
Not Null
Unique
Binary
Unsigned
Zero Fill
Auto Increment
Generated
Columns
Indexes
Foreign Keys
Triggers
Partitioning
Options
Apply
Revert
Output
Action Output
1 06:09:15 Apply changes to payments
Error 1166: Incorrect column name '' SQL Statement: CREATE TABLE 'simpleshopper'.'...
2 06:13:15 Apply changes to payments
Changes applied
3 06:22:22 Apply changes to food_inventory
Changes applied
4 06:25:12 Apply changes to customer_orders
Changes applied
Object Info
Session
SQL Editor Opened.
Type here to search
6:28 AM
10/30/2019

Class Diagram Implementation

Use Case: Registration

- Code used: HTML, CSS, PHP
- To run the web code, no compilation is needed. Go to the following link to access the registration webpage.
- http://codd.cs.gsu.edu/~aabdi11/Final%20Project%20Files/php/register.php
- All fields must have a value. The page will not submit unless all fields are filled.
- Once submitted, the page will redirect to another that will say registration is successful. This is how you know the code works.

Use Case: Log In

- Code used: HTML, CSS, PHP
- How to access code: Same as registration page. Go to the link below
- http://codd.cs.gsu.edu/~aabdi11/Final%20Project%20Files/php/login.php
- Upon submission, the page will either redirect to a page that says "Name/password is incorrect" or go to a page that cannot be reached. In the second case, this means that the login was a success. This page will be replaced with the inventory page. Both cases mean the webpage is working properly and that the website can connect to the database.

Use Case: View Inventory

- Code used: HTML, CSS, PHP
- How to access: Visit the login page and enter correct user credentials to access the inventory page. Use this test user information: username=atest, password=qqq111QQQ!!!
- If you want to access the admin pages, log in with these credentials: username=admin, password=adminPASSadmin1!
- If you do not wish to log in but want to see a page that will not have all of the user features, go directly to the following link to see the inventory page without user information.
- http://codd.cs.gsu.edu/~aabdi11/Final%20Project%20Files/php/inventory.php

Use Case: Log Out

- Code used: PHP
- How to access code: The log out button will be found on the inventory page as an icon on the navigation bar. Click on the logout icon to log out of the session. This action will take you back to the login page.

Use Case: Get Help on Registration

- Code used: HTML, CSS
- How to access code: The registration help page can be found at the following link. The link can also be found on the registration page below the form.

- http://codd.cs.gsu.edu/~aabdi11/Final%20Project%20Files/php/reghelp.html

Use Case: View Profile

- Code used: HTML, CSS, PHP
- How to access code: User must be logged in. Click on the icon for viewing the profile on the navigation bar
- This page so far only shows the user information that was entered at registration. It will also show the user's order history.

Use Case: Filter Inventory

- How to access: On the inventory page, there are links for each diet that you can click on. Click on one of these links to see filtered results.

Use Case: Add Inventory

- How to access: While logged in as an admin, click on the button on the navigation bar labeled "Add Inventory"

Use Case: View Cart Items

- How to access: While logged in as a customer, click on the button on the navigation bar labeled "View Cart"

Use Case: Add Cart Items

- How to access: While logged in as a customer, look for the item in the inventory that you want to add to your cart.
- Click on the button labeled "Add to Cart" associated with the item.

Use Case: Pay for Items

- How to access: While logged in as a customer, click on the view cart button on the inventory. On the view cart page, click on the button labeled "Pay for Items"

Use Case: Edit Profile

- How to access: While logged in as a customer, click on the view profile button on the inventory page.
- On the profile page, click on the button labeled "Edit Profile"

Testing

Test ID	1.1
Purpose of the test	Purpose of this test is to see if a user can create an account using valid username and password

Test Environment	Html, CSS, PHP
Test Steps	Enter a valid username and password. Click Submit button
test Input	Valid information for registration form fields
Expected Result	Successful registration message with a link to login page.
Likely Problem/ Bugs Revealed	There is a problem with password format. Even valid password is not accepted.

Test ID	1.2
Purpose of the test	Purpose of this test is to see if a user can create an account using invalid username and valid password
Test Environment	Html, CSS, PHP
Test Steps	Enter invalid username and password Click submit.
Test Input	Invalid registration information and submit button.
Expected Result	Error
Likely Problem/ Bugs Revealed	N/A

Test ID	1.3
Purpose of the test	Purpose of this test is to see if a user can create an account using no information given

Test Environment	Html, CSS, PHP
Test Steps	Leave fields blank. Click submit
Test Input	Submit button, empty registration fields
Expected Result	error
Likely Problem/ Bugs Revealed	N/A

Test ID	2.1
Purpose of the test	Purpose of this test is to see if a user can log in using invalid username and password to log in
Test Environment	Html, CSS, PHP
Test Steps	Enter a valid username and password. Click submit
Test Input	Valid username and Password
Expected Result	Logged in and redirected to personalized inventory page
Likely Problem/ Bugs Revealed	N/A

Test ID	2.2
Purpose of the test	Purpose of this test is to see if a user can log in using invalid username and password to log in
Test Environment	Html, CSS, PHP
Test Steps	Input invalid username and password. CLick submit

test Input	Invalid username and password
Expected Result	Error
Likely Problem/ Bugs Revealed	N/A

Test ID	2.3
Purpose of the test	Purpose of this test is to see if a user can log in using empty username and invalid password to log in
Test Environment	Html, CSS, PHP
Test Steps	Leave username and password blank. Click submit
test Input	Blank login form fields
Expected Result	error
Likely Problem/ Bugs Revealed	N/A

Test ID	3.1
Purpose of the test	Purpose of this test is to see if the user can log out if they are logged in.
Test Environment	Html, CSS, PHP
Test Steps	User clicks logout button

test Input	Logout button
Expected Result	Logged out and redirected to login page
Likely Problem/ Bugs Revealed	N/A

Test ID	3.2
Purpose of the test	Purpose of this test is to see if a logged out user can view pages that require login
Test Environment	Html, CSS, PHP
Test Steps	Click the logout button. Enter the url of the inventory page. Click on the profile link
test Input	Logout button, profile button
Expected Result	User is unable to view profile page
Likely Problem/ Bugs Revealed	N/A

Test ID	4.1
Purpose of the test	The purpose of this test is to see if the user can view the entire inventory.
Test Environment	Html, CSS, PHP
Test Steps	If not logged in. log in. If logged in, click links associated with the main inventory page
test Input	Inventory button, login, home button
Expected Result	Entire inventory (homepage) is displayed

Likely Problem/ Bugs Revealed	N/A

Test ID	4.2
Purpose of the test	The purpose of this test is to see if the inventory does not display for any user, even those who are not logged in
Test Environment	Html, CSS, PHP
Test Steps	If not logged in, log in or enter the url for the inventory page(in which certain functions will not be available on the page). If logged in. click links associated with the inventory page.
test Input	Inventory button, login, home button
Expected Result	View entire inventory (homepage)
Likely Problem/ Bugs Revealed	N/A

Test ID	5.1
Purpose of the test	The purpose of this is to test out the filter feature of our grocery shopping system. This will test the how the software functions without any filter picked.
Test Environment	Html, CSS, PHP
Test Steps	Access the site, Pick no filter on inventory page
test Input	Pick no filter.
Expected Result	display all available items

Likely Problem/ Bugs Revealed	

Test ID	5.2
Purpose of the test	The purpose of this test is to examine our filter feature. This will be used to test if filter tags works while filtering food items.
Test Environment	Html, CSS, PHP
Test Steps	Access the site. Pick filter tags (1 or 2).
test Input	Pick filter “Keto”.
Expected Result	Display all items related to the keto tag.
Likely Problem/ Bugs Revealed	

Test ID	6.1
Purpose of the test	The purpose of this test to test editing inventory for admin users. This test will check if the admin can edit inventory values.
Test Environment	Html, CSS, PHP
Test Steps	Log in as admin. Click edit inventory button. Input form fields for new item
test Input	Form fields for food name, price, image. Submit button
Expected Result	New item should display on inventory page.

Likely Problem/ Bugs Revealed	

Test ID	6.2
Purpose of the test	The purpose of this test is to check if the admin can edit inventory values with invalid fields.
Test Environment	Html, CSS, PHP
Test Steps	Log in as admin, Click edit inventory button. Input invalid food information
test Input	Invalid form input, submit button
Expected Result	Error. Food does not enter into database.
Likely Problem/ Bugs Revealed	

Test ID	6.2
Purpose of the test	The purpose of this test is to check if the admin can edit inventory values with blank fields.
Test Environment	Html, CSS, PHP
Test Steps	Log in as admin, Click edit inventory button. Leave food information form fields blank
test Input	Empty form fields, submit button
Expected Result	Error message.

Likely Problem/ Bugs Revealed	

Test ID	7.1
Purpose of the test	The purpose of this test is to test the view cart and cart system of simple shopper.
Test Environment	Html, CSS, PHP
Test Steps	Log in as a customer. Click view cart button.
test Input	View cart button.
Expected Result	Display items in the cart.
Likely Problem/ Bugs Revealed	

Test ID	7.2
Purpose of the test	The purpose of this test is to test the view cart and cart system of simple shopper while the user is not logged in.
Test Environment	Html, CSS, PHP
Test Steps	Log in as a customer. Click view cart.
test Input	View cart button.
Expected Result	Redirect to login page
Likely Problem/ Bugs Revealed	Page does not redirect

Test ID	8.1
Purpose of the test	This test is designed to test adding items to cart. For simple shopper you must be logged in as customer to add items to cart.
Test Environment	Html, CSS, PHP
Test Steps	Log In as a customer. Click an Add to Cart button
test Input	Add to Cart button
Expected Result	Redirect to view cart page, new item displayed on view cart page.
Likely Problem/ Bugs Revealed	

Test ID	8.2
Purpose of the test	This test is designed to test add items to cart while not logged in. For simple shopper you must be logged in to add items to cart.
Test Environment	Html, CSS, PHP
Test Steps	Go to inventory page without logging in. Click on the add to cart button
test Input	Add to Cart button
Expected Result	Redirect to login page
Likely Problem/ Bugs Revealed	Page does not redirect

Test ID	9.1
Purpose of the test	This test is designed to test the purchase system of simple shopper. This test will determine if the user can purchase items without any items.
Test Environment	Html, CSS, PHP
Test Steps	Go to view cart page without adding anything to the cart.
test Input	Click purchase button
Expected Result	Error no items.
Likely Problem/ Bugs Revealed	

Test ID	9.2
Purpose of the test	This test is designed to test the purchase system of simple shopper. This test will determine if the user can purchase items with items in cart.
Test Environment	Html, CSS, PHP
Test Steps	Log in, add items to cart. Purchase.
test Input	Purchase Items button
Expected Result	Redirect to payment page.
Likely Problem/ Bugs Revealed	

Test ID	9.3
Purpose of the test	Payment system will be tested. This test is being used to test if a purchase goes through with valid payment information.
Test Environment	Html, CSS, PHP
Test Steps	Click purchase items. Input card information. Proceed with payment.
test Input	Confirm Purchase button, purchase form fields
Expected Result	Purchase confirmation message displayed.
Likely Problem/ Bugs Revealed	

Test ID	9.4
Purpose of the test	Payment system will be tested. This test is being used to test if a purchase goes through with invalid payment information.
Test Environment	Html, CSS, PHP
Test Steps	Purchase items. Input invalid card information. Proceed with payment.
test Input	Confirm Purchase button, invalid purchase form fields
Expected Result	Error
Likely Problem/ Bugs Revealed	

Test ID	9.5

Purpose of the test	Payment system will be tested. This test is being used to test if a purchase goes through with empty payment information.
Test Environment	Html, CSS, PHP
Test Steps	Purchase items. Do not input card information. Proceed with payment.
test Input	Purchase Items button, empty form fields
Expected Result	Error need card information
Likely Problem/ Bugs Revealed	

Test ID	10.1
Purpose of the test	This test is to see if the edit profile functionality works for simple shopper when valid information is entered.
Test Environment	Html, CSS, PHP
Test Steps	Click edit profile. Change name and password following valid format.
test Input	New username and password
Expected Result	Saved profile to database. Redirect to profile page. Profile page displays new information.
Likely Problem/ Bugs Revealed	Information does not change

Test ID	10.2

Purpose of the test	This test is to see if the edit profile functionality works with invalid password change for simple shopper.
Test Environment	Html, CSS, PHP
Test Steps	Click edit profile. Change password to an invalid password.
test Input	1111111 password
Expected Result	Error message. Information does not save to the database
Likely Problem/ Bugs Revealed	

Test ID	11.1
Purpose of the test	Testing the view profile to see if it displays correct information about the user.
Test Environment	Html, CSS, PHP
Test Steps	Click view profile button.
test Input	View profile button
Expected Result	Display profile information
Likely Problem/ Bugs Revealed	

Test ID	11.2
Purpose of the test	To see if a user can see their profile without logging in.
Test Environment	Html, CSS, PHP

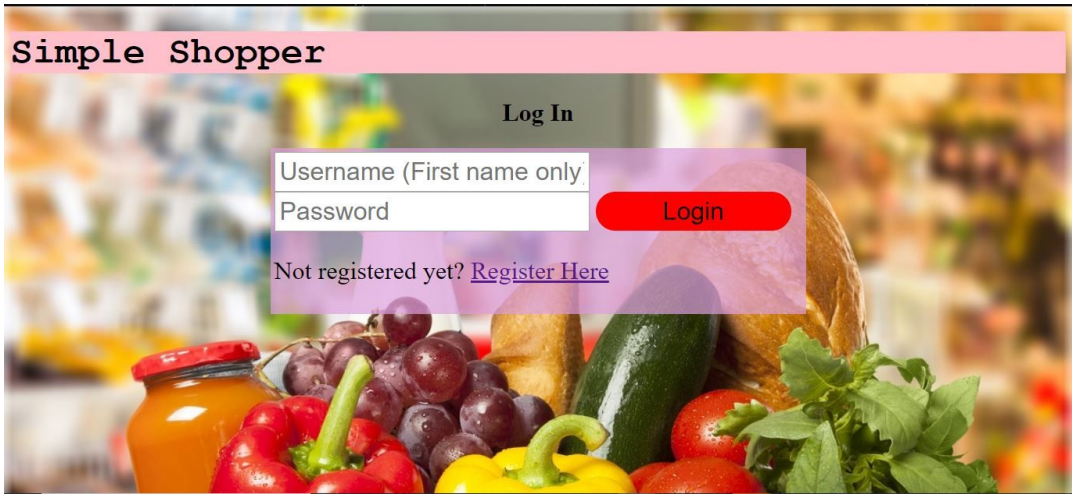
Test Steps	If logged in log out. click View profile.
test Input	View profile button.
Expected Result	Error and prompted to log in.
Likely Problem/ Bugs Revealed	

Test ID	12.1
Purpose of the test	Purpose of this test is to observe if the get help registration page displays when clicked
Test Environment	HTML, CSS , PHP
Test Steps	.click get help link on registration page
test Input	Click link
Expected Result	Display Get help page
Likely Problem/ Bugs Revealed	N/A

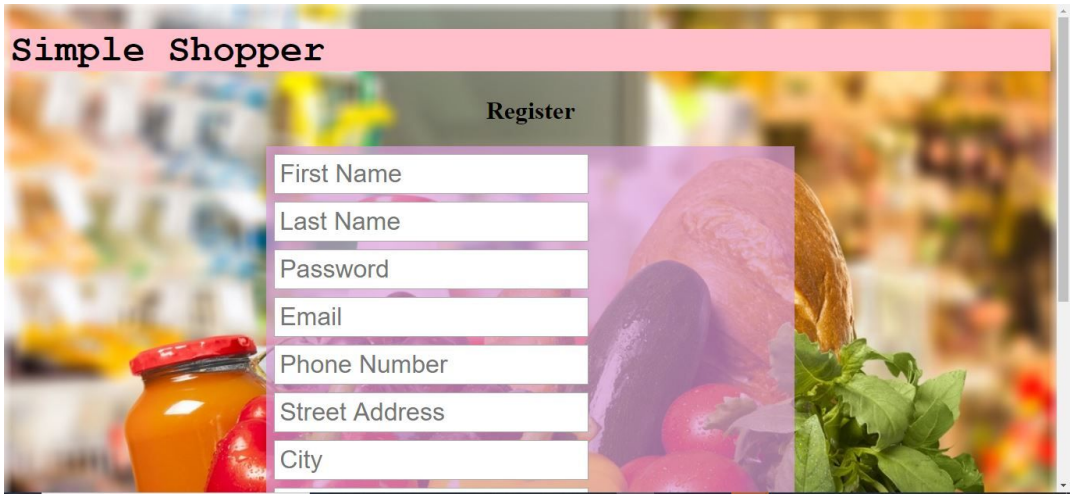
Bug	Test case ID that discovered the bug	Bug description	Action taken
Password format	1.1	Even following the rule of the format there would be an error message displayed for password.	Improved The formatting and fixed the coding error .

Implementation Screenshots

Login

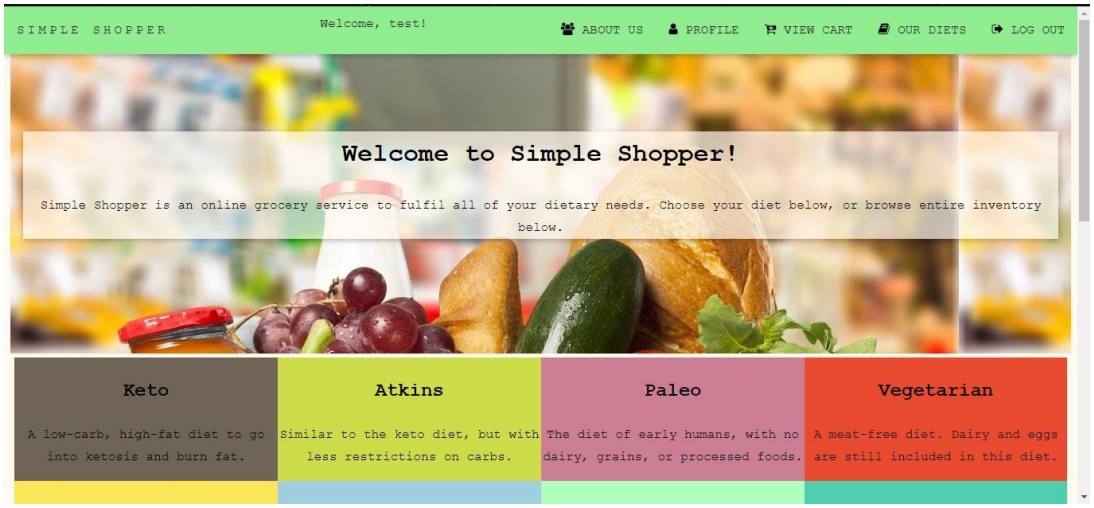


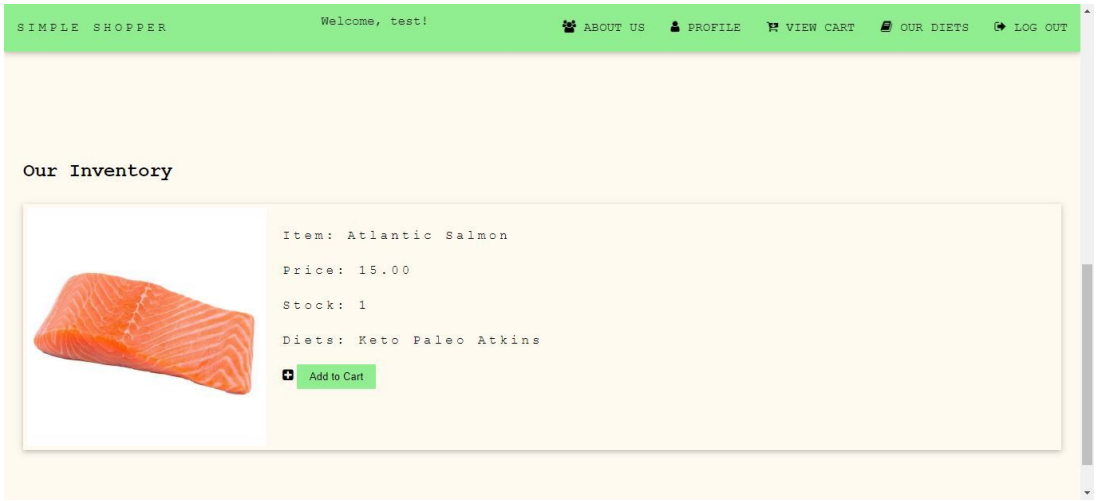
Registration



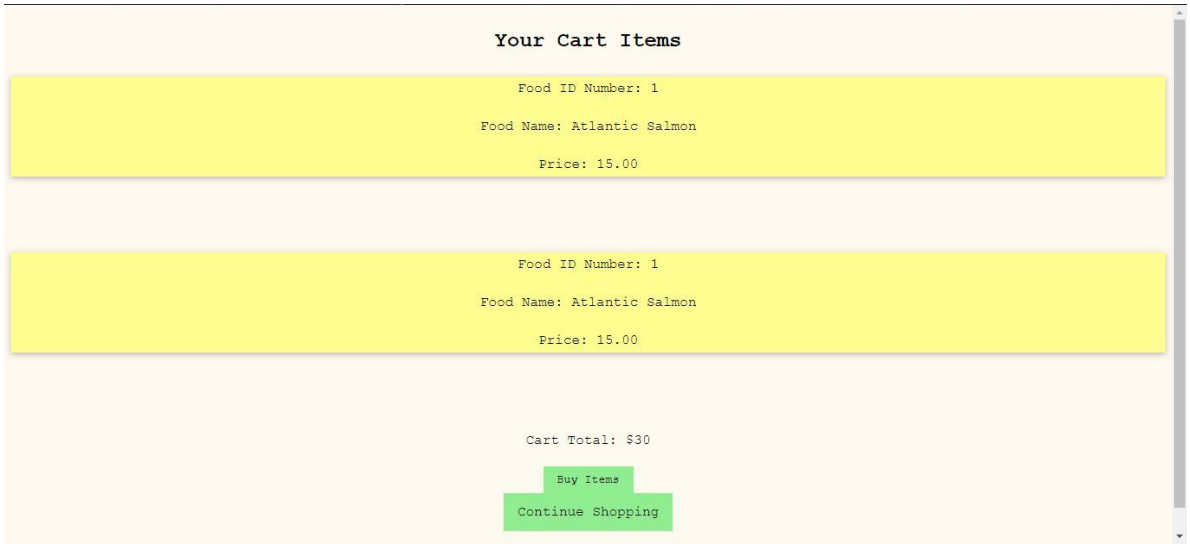
Get Help on Registration

View Inventory

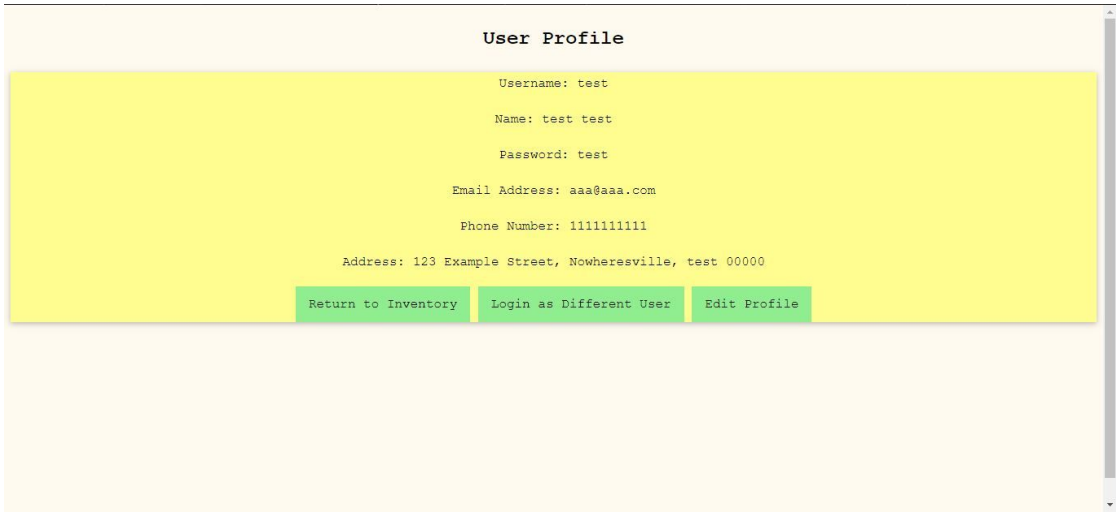




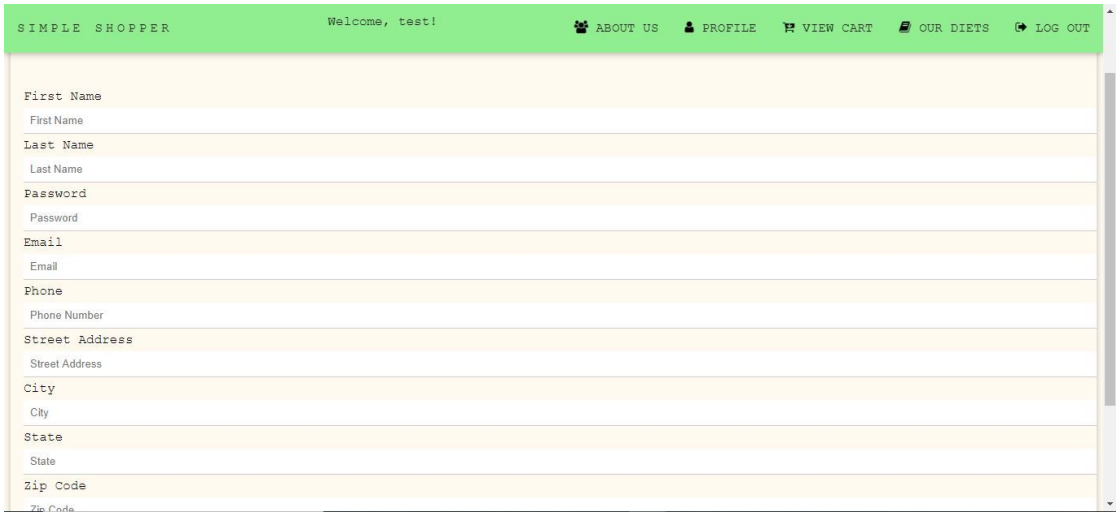
View Cart



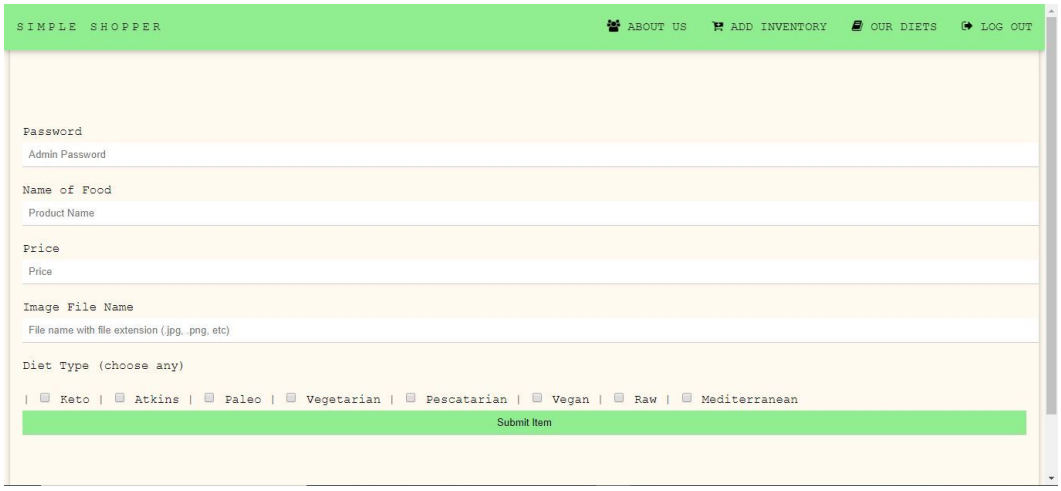
View Profile



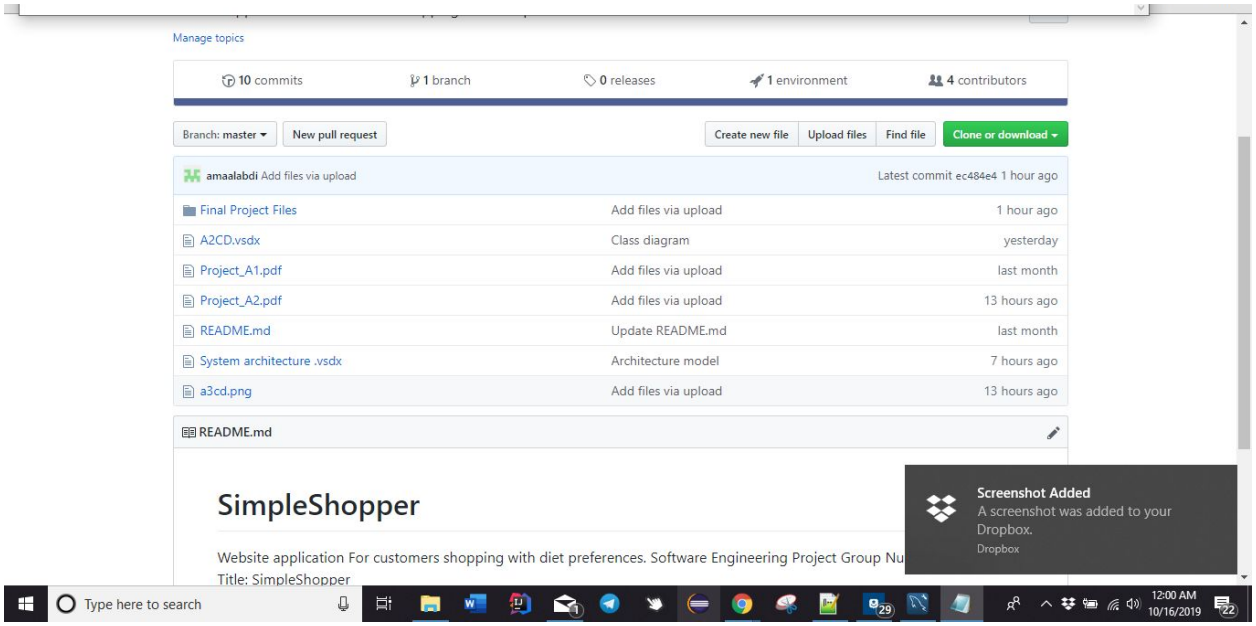
Edit Profile

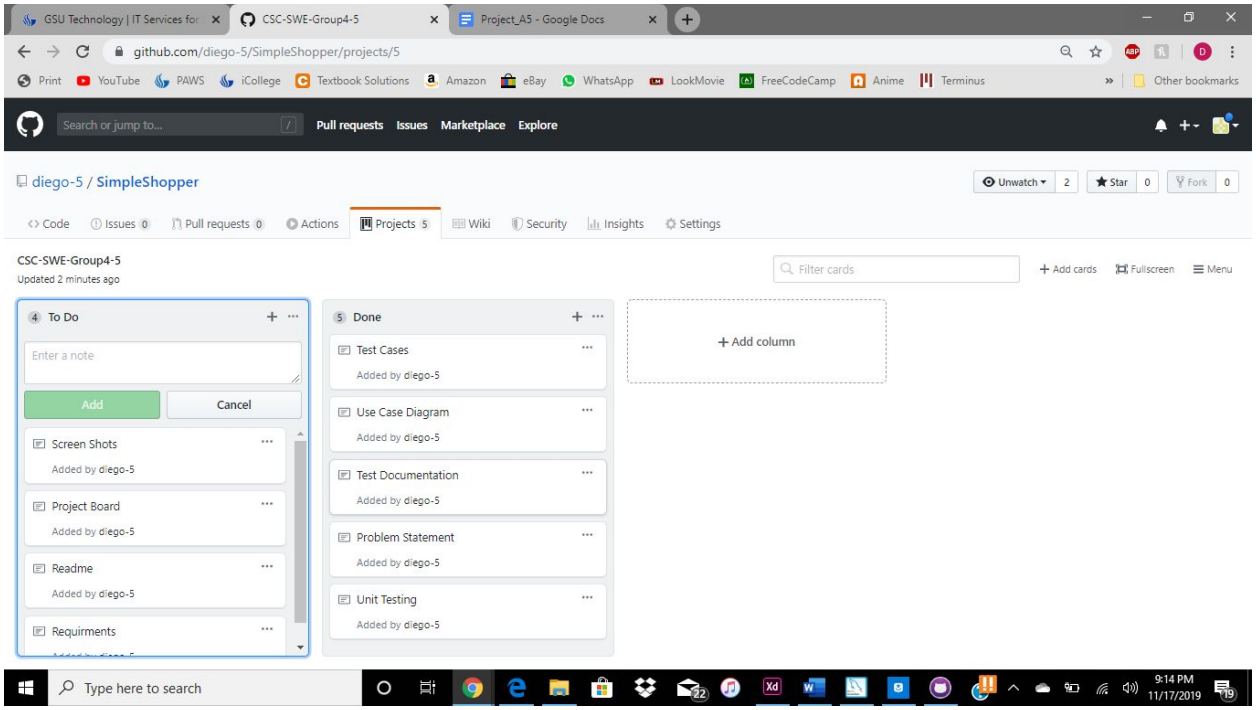


Add Inventory



Github Screenshots





Github Link:https://github.com/diego-5/SimpleShopper