THE DESIGN AND IMPLEMENTATION OF ONLINE STORE

By

Maheshwar Reddy Chappidi

mach17@student.bth.se

960811-4956

For the ET1446 Software Development Course in

Telecommunication Systems

Blekinge Institute of Technology

Sweden, 2018.

ABSTRACT

In today's fast-changing business environment, it's extremely important to be able to respond to client needs in the most effective and timely manner. If your customers wish to see your business online and have instant access to your products or services.

Online Shopping is a lifestyle e-commerce web application, which retails various fashion and lifestyle products (Currently Men's Wear). This project allows viewing various products available enables registered users to purchase desired products instantly. This project provides an easy access to Administrators to view orders placed.

In order to develop an e-commerce website, a number of Technologies must be studied and understood. These include, server and client-side scripting techniques, implementation technologies such as programming language (such as HTML, CSS, JavaScript) and relational databases. This is a project with the objective to develop a basic website where a consumer is provided with a shopping cart application and also to know about the technologies used to develop such an application.

This document will discuss each of the underlying technologies to create and implement an ecommerce website.

Table of contents:

1. Introduction

1.1 online store

E-commerce is fast gaining ground as an accepted and used business paradigm [1]. More and more business houses are implementing web sites providing functionality for performing commercial transactions over the web. It is reasonable to say that the process of shopping on the web is becoming commonplace.

The objective of this project is to develop a general-purpose e-commerce store where product like clothes can be bought from the comfort of home through the Internet. However, for implementation purposes, this paper will deal with an online shopping for clothes.

An online store is a virtual store on the Internet where customers can browse the catalog and select products of interest. Where users can self-register to the store with a name and a password. Apart from creating accounts, users can purchase items or delete their account. The admin account is used to modify the type of items available in the store and their quantity. In addition, the admin user can remove regular user accounts. After users log in to the store, they can see the list of available items. They are free to place in a shopping cart any type and number of items from the list, as long as it does not exceed the available quantity and when the quantity is exceeded, an error message will be displayed. Once the users are satisfied with the contents of the cart, they will be given the opportunity to either empty the cart or purchase the items.

The online store will be implemented using two Virtual Machines (VMs) in Linux environment. One VM is designated as front end (we call it VM_WWW) and the other as the back end (we call it VM_DB). VM_WWW and VM_DB will be able to communicate over a VirtualBox host-only network [2].

The front end (VM_WWW) will consist of a web server or web service that implements a complete RESTful API towards the client. The API will that make use of all four main HTTP methods GET, PUT, POST, DELETE according to the REST paradigm.

The back end (VM_DB) will consist of a MySQL database. All details related to the store contents, users and shopping carts will be stored in the database. Any action that the client performs towards the front end (list store contents, update cart etc.) will generate an operation towards the database.

2. Software Requirements

Requirement Numbers	Requirements	Description
R1	A complete RESTful API towards the client.	REST APIs built on HTTP, the uniform interface includes using standard HTTP verbs to perform operations on resources. The common operations are GET, POST, PUT and DELETE. The Restful API functions are implemented using PHP. Whenever the client performs the action towards the web application Rest API functions GET, POST, PUT and Delete are used to post, retrieve, update and delete the information. The PHP server sends the response when the client performs some action.
R2	Implementation of VMs	The online store will be implemented using two Virtual Machines in Linux environment. One VM is designated as front

		end (we call it VM_WWW) and the other as the back end (we call it VM_DB). VM_WWW and VM_DB will be able to communicate over a VirtualBox host-only network with the network prefix 192.168.200/24. The VM_WWW will be configured to include a NAT interface. The host (where the VMs are running) will act as client and connect to VM_WWW over the NAT interfaces. This will require to configure VirtualBox to do port forwarding for VM_WWW
R3	Backend (database connection)	The back end (VM_DB) will consist of a MySQL database system. All details related to the store contents, users and shopping carts will be stored in the database. Any action that the client performs towards the front end (list store contents, update cart etc.) will generate an operation towards the database.

R4	Admin account page	The admin account page is used to modify the type of items available in the store and their quantity. In addition, the admin user can remove the regular user accounts from this admin account page.
R5	Signup page	The online store is provided with signup page. The user can create an account with username and password credentials. The error message will be displayed when the credentials does not match correctly.
R6	login page	The online store main page is provided with login page where the user can login with the credentials which was registered. The error message is displayed when the user enters wrong credentials.
R7	Product page	The product page should fetch the data from the database and list the details of the available items in the online store with the item properties (Type

		(shirts, pants), Brand, Color, Size, Price).
R8	Select type of items.	The select option is provided for the user to shop either shirt or pant.
R9	Add products to cart	Here, the users are free to place in a shopping cart any type and number of items from the list.
R10	List in cart page	We can see the list of number of items stored in a cart to purchase.
R11	Deleting products in a cart	User can remove the items in a cart. When user remove the items in a cart then the item is added back to the list items in home page.
R12	Products amount to buy	The total price of all the product list in the cart is displayed in its page.
R13	Purchase products	The user can purchase the number of products added in a cart.

R14	Quantity Limit	When user try to add the zero-quantity item then the error message will be displayed.
R15	Delete account	The user can delete his own account.
R16	Admin page for removing users account.	The admin user can remove the user accounts from this admin account page.
R17	Admin page for editing products	Here the admin can edit the products(add/delete) quantity in a store.
R18	Users database information	All user account details information will be stored in an accounts database table.
R19	Product items database information	All details related to the store products information will be stored in a items database table.
R20	Cart database information	The online store database will consist of an information regarding the products added to the cart in a cart database table.

3. DESIGN

3.1 Model:

Fig 1 shows the complete model of an online store which was designed with two Virtual Machines in Linux environment. One VM is designated as front end and the other as the back end. VM_WWW and VM_DB will be able to communicate over a VirtualBox host-only network with the network prefix 192.168.200/24. The VM_WWW will be configured to include a NAT interface. The host will act as client and connect to VM_WWW over the NAT interfaces. This will require to configure VirtualBox to do port forwarding for VM_WWW.

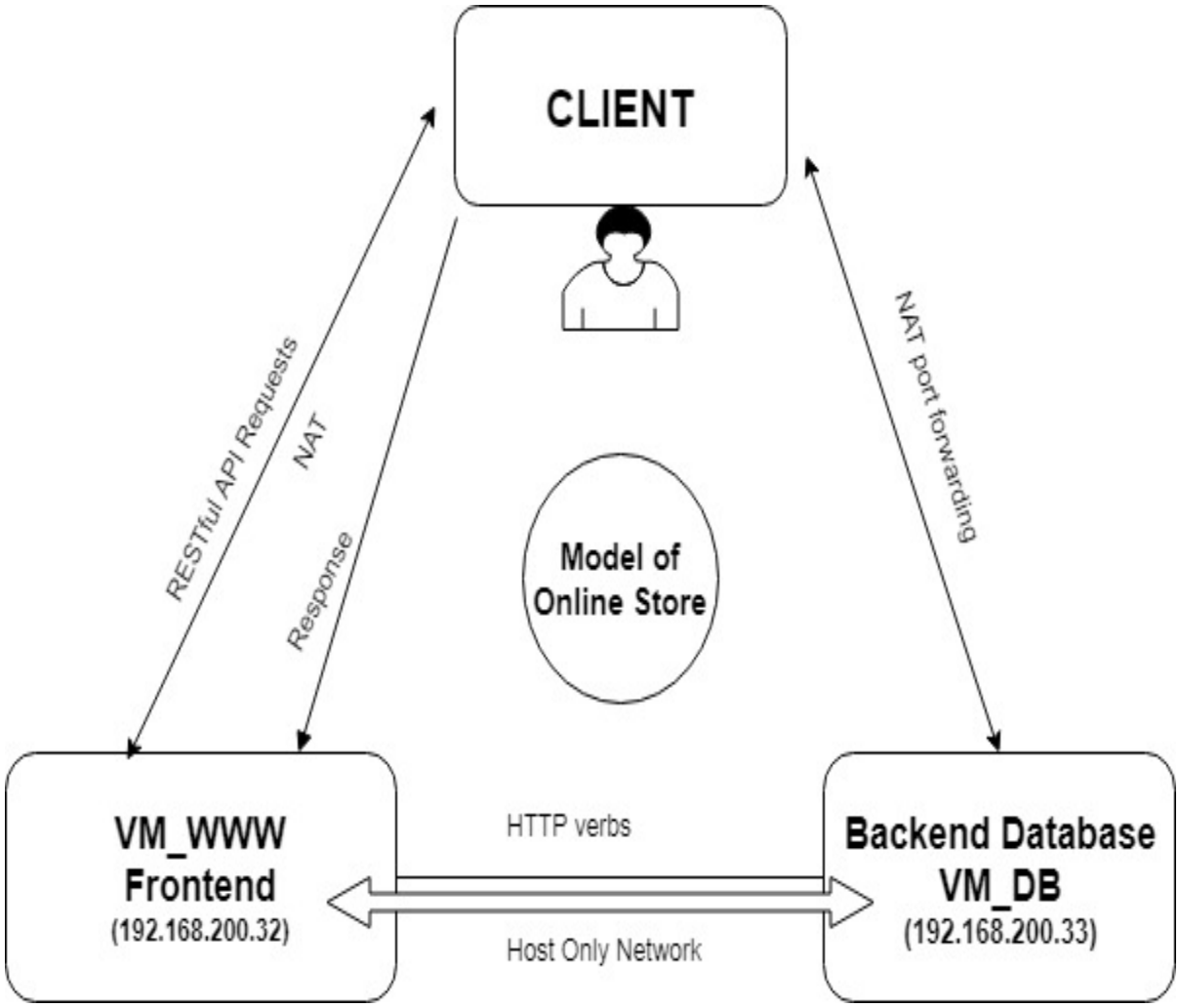

Fig 1: The complete model of an online store.

3.2 Function Model of Online Store Web Application.

Fig 2 shows the complete functional model of online web application.

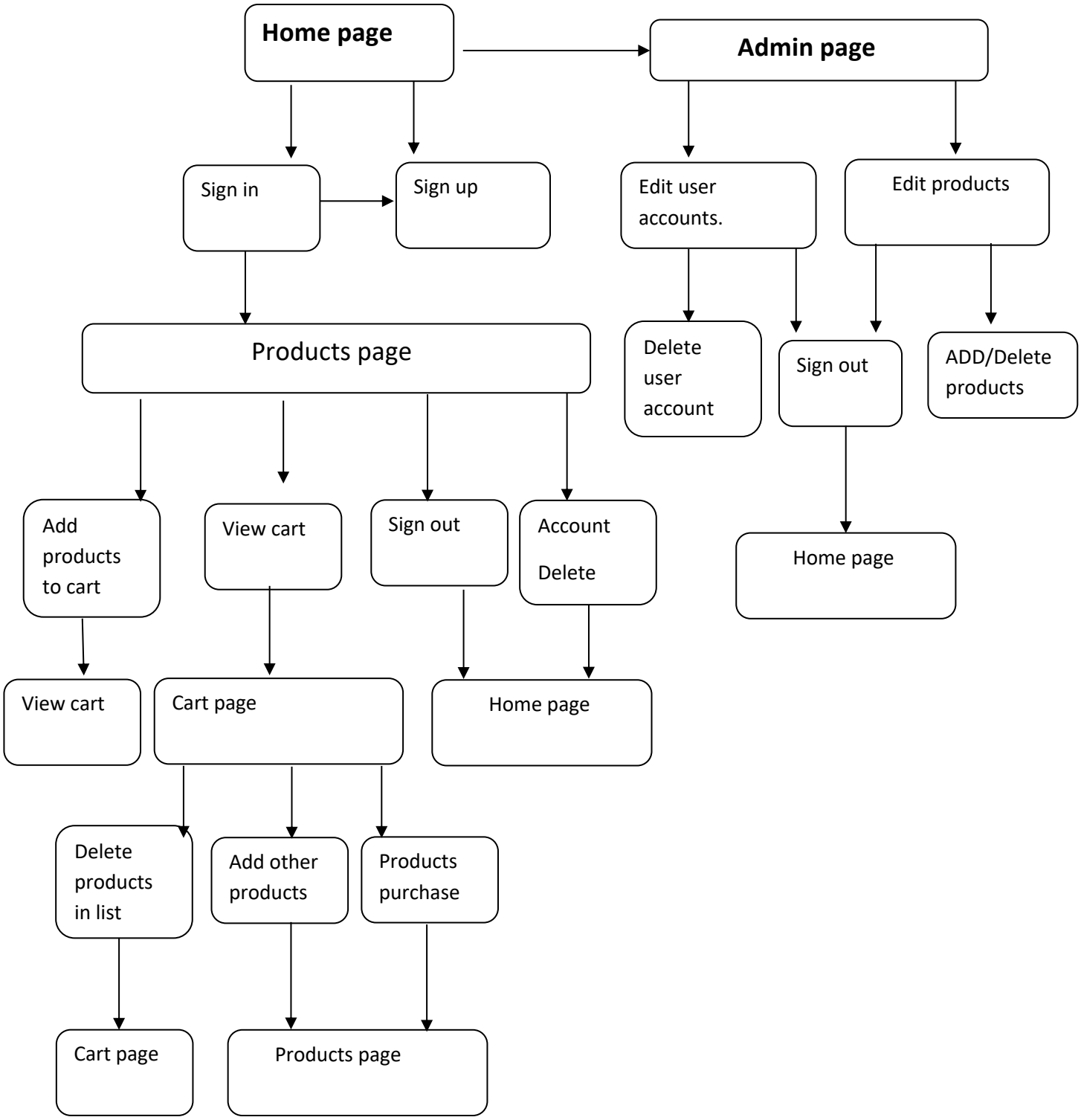
Home page
Admin page
Sign in
Sign up
Edit user accounts.
Edit products
Products page
Delete user account
Sign out
ADD/Delete products
Add products to cart
View cart
Sign out
Account
Delete
Home page
View cart
Cart page
Home page
Delete products in list
Add other products
Products purchase
Cart page
Products page

Fig: 2

3.3 Database schema:

In this section, the tables show the basic structure of the tables composing the database for the project.

3.3.1 Accounts database table:

The online store database consists of information regarding user accounts in user's database table as shown in table 1:

Field	Type	Null	Key	Default	Extra
ID	int(11)	No	PRI	NULL	auto_increment
EMAIL	varchar(255)	No			
PASSWORD	varchar(255)	No			

Table: 1

3.3.2 Cart database table:

The online store database consists of information regarding items added to the cart in a cart database table as shown in table 2:

Field	Type	Null	Key	Default	Extra
ID	int(11)	N0	PRI	NULL	auto_increment
type	varchar(30)	N0			

brand	varchar(30)	N0			
colour	varchar(30)	N0			
size	varchar(30)	N0			
price	int(30)	N0		NULL	

Table: 2

3.3.3 Items database table:

The Online store database consists of information regarding products in Items database table as shown in table 3:

Field	**Type**	**Null**	**Key**	**Default**	**Extra**
ID	int(11)	N0	PRI	NULL	auto_increment
type	varchar(30)	N0			
brand	varchar(30)	N0			
colour	varchar(30)	N0			
size	varchar(30)	N0			
price	int(30)	N0		NULL	
quantity	int(30)	N0		NULL	

Table: 3

4. IMPLEMENTATION

4.1 FRONTEND

The front end (VM_WWW) will consist of a web server that implements a complete RESTful API towards the client. The API will that make use of all four main HTTP methods GET, PUT, POST, DELETE according to the REST paradigm. These HTTP methods can be shown by using a Wireshark. The frontend web server application is implemented by using HTML, PHP, CSS is used for the display of the GUI.

At frontend, the web page is implemented to provide the users to self-register to the store with a name and a password. Apart from creating accounts, users can purchase items or delete their account. The admin account is used to modify the type of items available in the store and their quantity. In addition, the admin user can remove regular user accounts. After users log in to the store, they can see the list of available items. They are free to place in a shopping cart any type and number of items from the list, as long as it does not exceed the available quantity and when the quantity is exceeded, an error message will be displayed. Once the users are satisfied with the contents of the cart, they will be given the opportunity to either empty the cart or purchase the items.

4.2 BACKEND DATABASE

The backend database (VM_DB) will consist of a MySQL database system. A remote MySQL connection is established between the frontend and backend database. All details related to the store contents, users and shopping carts will be stored in this database.

Any action that the client performs towards the front end (list store contents, update cart etc.) will generate an operation towards the database. If the user modifies the contents of the cart, we can update the state of the cart in the DB and the item quantities.

4.3 COMMUNICATION BETWEEN CLIENT AND SERVER

In the REST architecture, clients send requests to retrieve or modify resources, and servers send responses to these requests. Let's take a look at the standard ways to make requests and send responses.

4.3.1 MAKING REQUESTS

REST requires that a client make a request to the server in order to retrieve or modify data on the server [3]. A request generally consists of:

- an HTTP verb, which defines what kind of operation to perform
- a header, which allows the client to pass along information about the request
- a path to a resource
- an optional message body containing data

4.3.2 HTTP VERBS

There are 4 basic HTTP verbs we use in requests to interact with resources in a REST system:

- GET — retrieve a specific resource (by id) or a collection of resources
- POST — create a new resource
- PUT — update a specific resource (by id)
- DELETE — remove a specific resource by id

4.4 Communication settings for VM_WWW and VM_DB

The online store will be implemented using 2 VirtualBox Ubuntu Linux VMs (1GB RAM per VM). One VM is designated as front end (VM_WWW) and the other as the back end (VM_DB). VM_WWW and VM_DB will be able to communicate over a VirtualBox host-only network. The VM_WWW and VM_DB will be configured in the network adapter along with the NAT interface connection with the host. The IP addresses of VM_WWW and VM_DB are 192.168.200.7 and 192.168.200.8 respectively. The host (where the VMs are running) will act as client and connect to VM_WWW over the NAT interfaces. The VirtualBox will be configured to do port forwarding for VM_WWW.

4.5 Database connection between VM_DB and VM_WWW

The following commands are used for the database connection between VM_WWW and VM_DB.

- CREATE USER 'vm_www'@'192.168.200.7' IDENTIFIED BY 'password'. By running this command at VM_DB, a user with this IP address of the client is added to the table in MySQL database.
- GRANT ALL PRIVILEGES ON *.* TO 'VM_WWW'@'192.168.200.3' with GRANT OPTIONS. It will grant all privileges to create user which wants to access this database remotely is performed.
- FLUSH PRIVILEGES.

- FLUSH HOSTS.
- To check either the user is added or not run the following command in MySQL database: SELECT HOST, USER from mysql.user.

- Add the following line in the MySQL configuration file in /etc/mysql/mysql.confd/mysqld.cnf

  - Bind-adress = 0.0.0.0. (listens universal clients).
  - Port = 3306 (default port for MySQL).
  - Restart the mysql service by running the following command.
  - Sudo mysql service restart.

4.6 Implementation of Rest API functions

REST APIs built on HTTP, the uniform interface includes using standard HTTP verbs to perform operations on resources. The common operations are GET, POST, PUT and DELETE. The Restful API functions are implemented using Slim framework. Whenever the client performs the action towards the web application Rest API functions GET, POST, PUT and Delete are used to post, retrieve, update and delete the information. The PHP server sends the response when the client performs some action.

4.6.1. HTTP GET Operation:

The GET operation will requests to retrieve resource information when the user sign in to the store – and not to modify it in any way. As GET requests do not change the state of the resource, these are

said to be safe methods. Additionally, GET APIs should be idempotent, which means that making multiple identical requests must produce the same result every time until another API (POST or PUT) has changed the state of the resource on the server.

Example GET requests URL:

http://127.0.0.1:8080/cartitems.php?user=1

4.6.2 HTTP POST Operation:

The POST APIs is to create new subordinate resources, e.g. a file is subordinate to a directory containing it or a row is subordinate to a database table. Talking strictly in terms of REST, POST methods are used to create a new resource into the collection of resources.

Example GET requests URL:

http://127.0.0.1:8080/shirt.php

4.6.3 HTTP PUT Operation:

The PUT APIs primarily to update existing resource. When the client wants to edit the information in the server then the PUT operation is performed and the information is updated in the server and the response is sent back to the client.
Example GET requests URL:

http://127.0.0.1:8080/totalitems.php

4.6.4 HTTP DELETE Operation:

When the client wants to delete information in the server then the HTTP DELETE operation is performed, then the information is deleted in the server and the response is sent back to the client.
Example GET requests URL:

http://127.0.0.1:8080/delcart.php?id=2&user=2

4.6.5 To test the REST API methods:

The Wireshark is used to verify the GET, PUT, POST and DELETE methods between the client and the frontend server.

5. Curl commands

- To get a specific user from a database using GET we use the following command.
  curl -i -X GET http://localhost/api/user/1

- For creating a new user using POST we use the following command.
  curl -i -X POST -d "EMAIL=sample1&PASSWORD=sample" http://localhost/api/user

- To update a user using PUT we use the following command.
  curl -i -X PUT -d " EMAIL=sample1&PASSWORD=sample" http://localhost/api/user

- To delete a user using DELETE we use the following command.
  curl -i -X DELETE http://localhost/api/user/1

6. REST URLs for different functions:

- http://127.0.0.1:8080/home.html - To open the frontend web page of Online Clothes Store. Here the admin can sign in and the user have options to sign in/signup.

- http://127.0.0.1:8080/signup.html - This URL will direct in to the user sign up page to enter the credentials for account registration. If the user tries to register with empty fields then the error message will be displayed. After registration the user will move to login page.

- http://127.0.0.1:8080/signup.php - When the user registered properly the message will be displayed that user registered successfully then user have options to sign in or can go to online clothe store home page. If the user doesn't register properly then the error message will be displayed.

- http://127.0.0.1:8080/login.html - Here the user can login to the online clothe store by providing credentials or the user have an option signup or move to home page.

- http://127.0.0.1:8080/login.php - When the user sign in successfully, then he welcomed to the online clothing store and he can select to view the products in a store.

- http://127.0.0.1:8080/items.php?id=3 - The above URL is used to retrieve the information of the product items and can add the items into a cart or can delete the account or he can sign out from

the store. The id in URL specifies who logged into store. Whenever a user is registered, the user gets a unique id and the id is auto incremented for newly created user again and again in the ACCOUNTS table.

- http://127.0.0.1:8080/addcart.php?id=2&user=3 – After adding the items in a cart, the item is added successfully into a cart message will be displayed. Here, the user has two options either he can select to view carts or add more items to a cart. In the above URL, that id=2 represents that item 2 is added to the cart by user number 3. Whenever the new item is added by the admin person, then the product id is auto incremented in the PRODUCTS table and whenever the user is registered the user id will be auto incremented.

- http://127.0.0.1:8080/cartitems.php?user=3 – Here the user can see the list of items added into a cart and total price of items to purchase or he can delete the items in a cart. When the user sign-out without purchasing the products then the quantity of items in cart will be added into a store list item.

- http://127.0.0.1:8080/delcart.php?id=3&user=3 – The above URL is used to delete the specific item from the cart and after removing the item from the cart then the item is removed successfully from the cart message will be displayed. Then the user has an option to view cart or add more items to a cart. In the above URL, that id=3 represents that item id number 3 is deleted from the cart by user id number 3.

- http://127.0.0.1:8080/admin.html - This URL will direct to the admin page to login by providing the credentials. The username and password to login as admin is “maheshwar, 0000”.

- http://127.0.0.1:8080/admin.php - Here the admin has an option to edit items in the online clothing store or he can delete the users accounts who are registered in online clothing store.

- http://127.0.0.1:8080/accounts.php - In this page the user accounts will be showed in a table. Here the admin can delete the user accounts or have an option to edit items in a store or he can sign out from the page. When the admin deletes the user then the confirmation message will be displayed either to delete or cancel.

- http://127.0.0.1:8080/totalitems.php - This page shows the total items available in an online clothing store in a table. The admin can edit the item or delete the items from the table list. The admin has an option to add new items to the store or he can sign out from the page.

- http://127.0.0.1:8080/shop.html - Here the admin has an option to add which type(shirt/pant) of items to add into a store.

- http://127.0.0.1:8080/shirt.html - The admin can edit and add shirt item to a store by providing the quantity, colour, size, price, brand to the shirt.

- http://127.0.0.1:8080/shirt.php - After adding the items to a store then the “new item is added successfully to the store” message will be displayed in this page and the admin have an option to view the all items in a table. When the admin tries to add with empty fields then the missing field type error message will be displayed.

- http://127.0.0.1:8080/pant.html - The admin can edit and add pant item to a store by providing the quantity, colour, size, price, brand to the shirt.

- http://127.0.0.1:8080/pant.php - After adding the items to a store then the “new item is added successfully to the store” message will be displayed in this page and the admin have an option to view the all items in a table. When the admin tries to add with empty fields then the missing field type error message will be displayed.

- http://127.0.0.1:8080/itemedit.php?id=2 – Here the admin can edit the item like he can change the quantity, price, colour, size or brand. After editing the item, he has an option to update the items or he have an option to view the list items in a store.

- http://127.0.0.1:8080/db.php - By running this URL, the database name “mach” and tables (CART, ACCOUNTS, ITEMS) will be created in database.

7. References:

[1] G. Lan-juan, L. Quan, and J. Xue-mei, “The Design and Implementation of the Online Shopping System for Digital Arts,” in *2010 Ninth International Symposium on Distributed Computing and Applications to Business, Engineering and Science*, 2010, pp. 414–416.

[2] “Chapter 6. Virtual networking.” [Online]. Available: https://www.virtualbox.org/manual/ch06.html#network_hostonly. [Accessed: 10-Nov-2018].

[3] “Introduction to REST and .net Web API – Martin Kearn.” [Online]. Available: https://blogs.msdn.microsoft.com/martinkearn/2015/01/05/introduction-to-rest-and-net-web-api/. [Accessed: 10-Nov-2018].