Tiaan Stevenson-Brunt, 6603268

349 Grocer

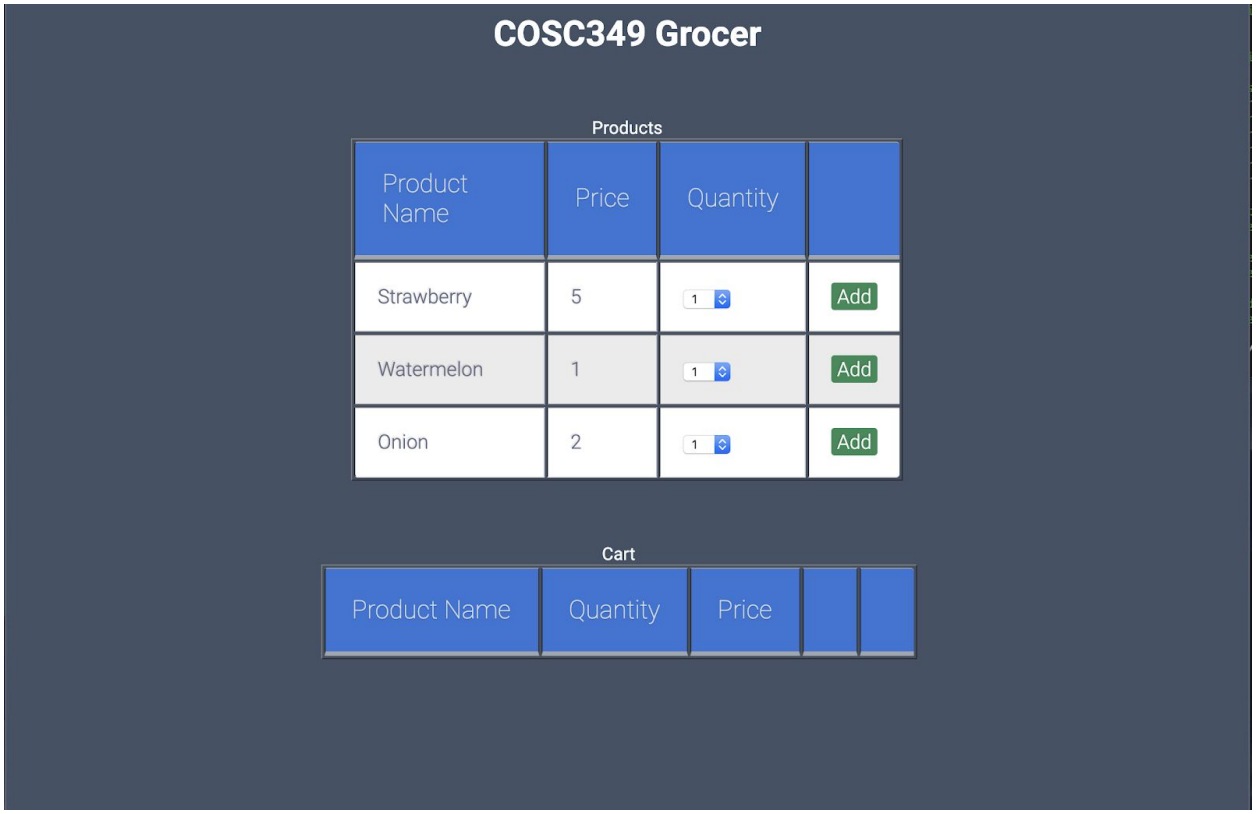
https://github.com/TiaanBrunt/COSC349-Assignment-1

My application is a simple online commerce application for a business, in this case '349 Grocer'. My application is in the very early stages of development and so does not have all the final features expected in an e-commerce application. The application is made of three components: a customer facing website, an administrative website and finally a database. These three sections are all VM's and can be run independent from each other, all though information from the database server won't be displayed if web servers are run independent. The webservers use Apache and the database uses MYSQL. In addition to this all three Virtual Machines use Ubuntu. The reason there is a separate web server for both the customer and admin is for security. You don't want customers accidentally stumbling into the admin sections of an application where there could be potentially confidential information. Please note the color scheme for the project was inspired by the tutorial I loosely followed/modified to make the login - https://codeshack.io/secure-login-system-php-mysql/.

Customer web server

The customer web server is an apache web server where a basic shopping cart system is displayed, as seen in Fig 1. All the products in the database (from the product table) are displayed here and the user has the ability to add products to the cart. Once products are added in the cart the user can remove the product from the cart or order the item. If the order button is clicked the item is automatically 'ordered' by getting added to the orders table, and removed from the cart. The customer web server interacts with the database server by displaying the products table and also inserting and removing values in the carts table.

FIG 1 - screen shot of 192.168.13/index.php
The customer website



Admin Web server

The admin web server named ‘adminwebserver’ is an apache web server where a user can act as an administrator of the business and make changes to the database. Accessed at localhost:8080 users will first be taken to a login page where they can login with the details ‘admin’ for both password and username as seen in Fig 2. If credentials are correct the orders.php page (fig 3) will be displayed where almost all the data in the database server is displayed. Users can now see additional information about the products such as quantity and also see the orders which were made on the customer web server as well as deleting orders. Currently the admin web server is missing some features such as updating product quantities and add/edit/remove products due to difficulties with php.

Fig 2. localhost:8080/index.php
The login screen for the admin web server

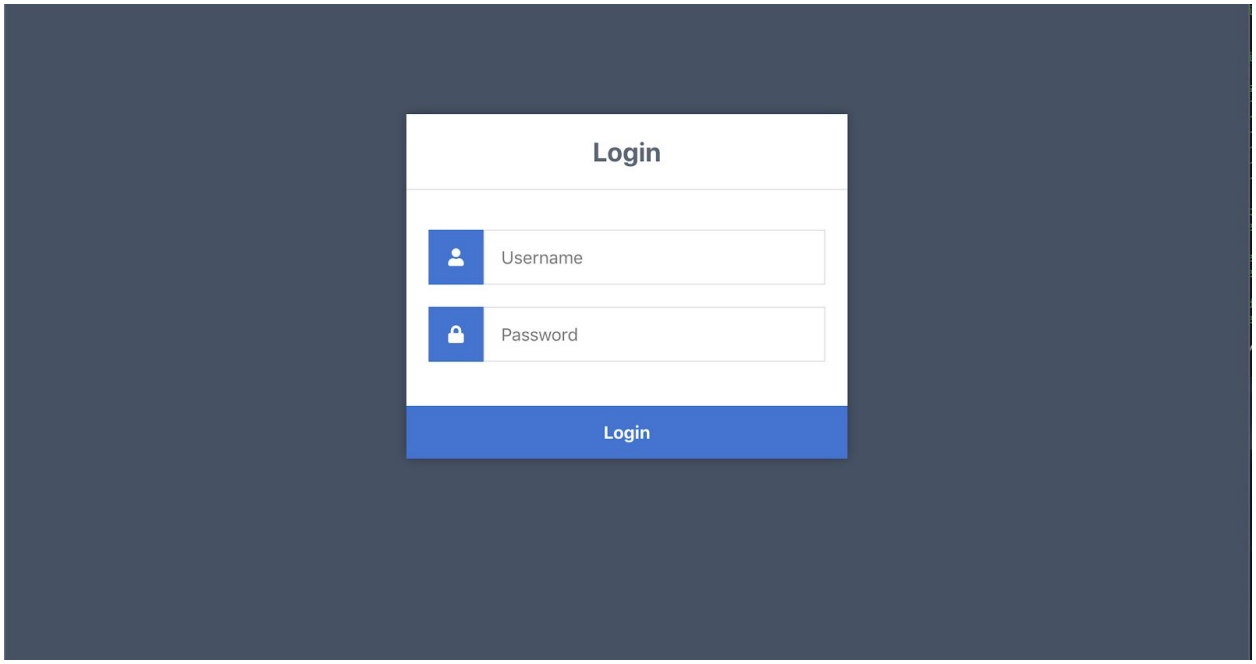
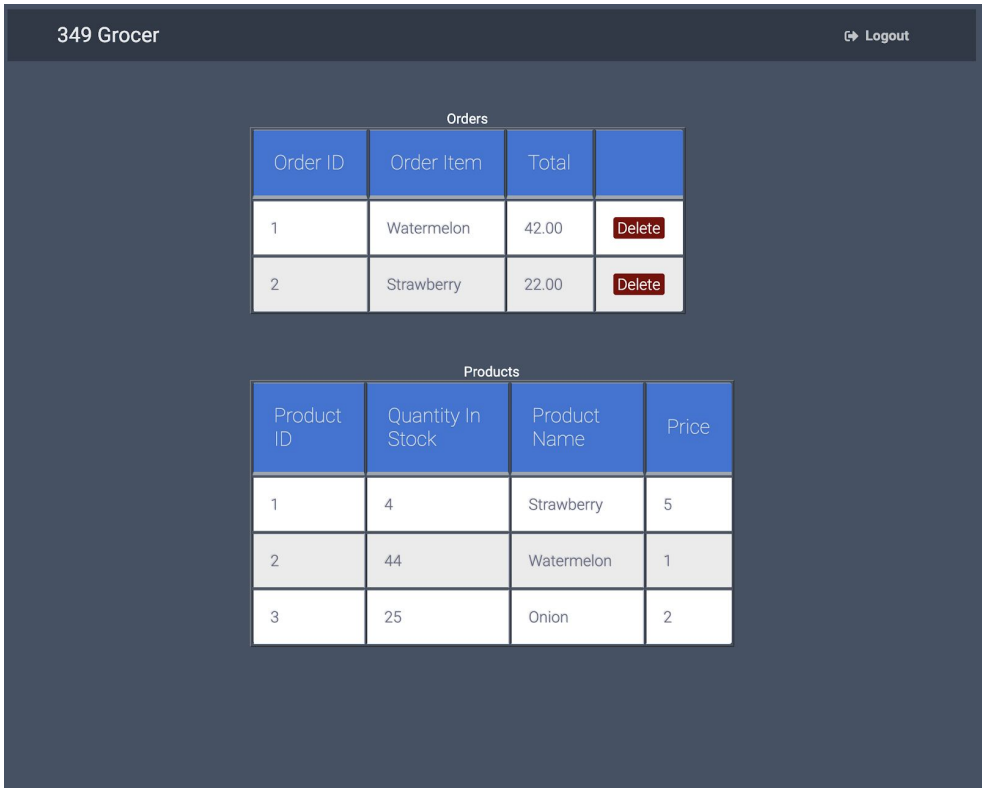


Fig 3. localhost:8080/orders.php



Database server

The database server named ‘dbserver’, is a MYSQL server containing all the data used by the other Virtual Machines. The tables stored in the database ‘fvision’ are used and updated by both the ‘adminwebserver’ and ‘customerwebserver’.

How to run

1. This application uses both Vagrant and VirtualBox so before we can run the application itself users need to make sure they have Vagrant and VirtualBox installed on their computer. The latest versions of both software can be found at

   Vagrant
   https://www.vagrantup.com/downloads.html

   Virtual Box
   https://www.virtualbox.org/wiki/Downloads

2. Next clone the repository or download the project files from github

   Clone:
   Use the command git clone https://github.com/TiaanBrunt/COSC349-Assignment-1.git

   Or Download:
   Download from the address
   https://github.com/TiaanBrunt/COSC349-Assignment-1/archive/master.zip

3. In the command line get into the directory where you saved the project from step 2.

4. Now it's time to build the application. Simply use the command ‘vargrant up’ and all three virtual machines will be built. Please note that the build time is roughly 4-5minutes depending on your computer so don’t fret if the build is taking a while.
5. Now the virtual machines will all be up and running and you can begin to use the application.

To view the Admin website:
In a browser visit the address -
localhost:8080

To view the Customer website:
In a browser visit the address -
Localhost:8081

6. Once your finished with the program use the command ‘vagrant destroy’ to destroy the running VM’s

Project Build

Build time
Build time roughly 4min 30 seconds - 4min 50seconds (based off 2 tests on lab computers)

Build Size
The first build is roughly 273Mbs for the Ubuntu Xenial 64-bit with subsequent builds being around 150-200mbs.

Problems / Extensions

Originally I was creating a Timezone converter application but decided to change to the current ecommerce application due to the fact I couldn't understand why you would need a third virtual machine for this program. A third virtual machine seemed pointless and as a big part of this assignment is based on having an application that uses 3 Virtual machines I decided to flag this application and change to my e-commerce app.

Original plan for application ->
Old App: World Timezone Converter
Create an application where users can choose a timezone from a drop down menu and the clock will display the time. Will need 3 virtual machines:

1. Web Server: This is where the actual application will be: Create the clock using a combination of javascript, html, css Get the timezone and store the data using php.
2. Database: Will have the time zones and users settings stored here.
3. Querry: Will fetch the relevant info from the database.

The main problems I've had with this project were based around learning php. As I've never used php this was a steep learning curve and unfortunately I wasn't able to implement all the features I would have liked to. The following are features that I didn't implement but would be a good extension for future developers.

1. Update quantities.
   Currently when you add a product to the cart its quantity will always be 1 no matter what you've selected. In addition to this It would be a good idea to check the quantity of the product from the product table first before adding it to the cart to ensure the user isn't ordering an item that is sold out. Lastly once an order is made it would be nice to update the quantity of the product in the products table.

2. Edit/Remove/Add products
   Currently the admin website only shows the products table. A nice extension of this application would be to give the user the option to add, edit and remove products from the table. All of these functions would use php echoing an SQL statement e.g. INSERT, ALTER or REMOVE ..

3. Improve Security
   Security could be improved across the board for the application but an easy extension to benefit the security of the application would be to hash the passwords stored in the users table. Using the php function m5 future developers could easily encrypt the stored passwords which will add another layer of protection to the application.

Once a future developer has implemented any of the above suggestions or made any other change to the project, destroy the vagrant project using 'vagrant destroy' and then 'vagrant up' to rebuild the project.