Shaun Liew 2553068
Josh Casey 3023930

COSC349 Assignment 2: your software in the cloud.

By Josh Casey & Shaun Liew

How the application was deployed:

This project builds on the application that we had previously created using three VMs to simulate a basic e-commerce platform. The application was deployed through 2 instances of AWS EC2 and a database server through Amazon RDS.

We first created our database server on RDS, a MySQL database with public access. To run our database on RDS, we removed our original database server VM that was created and used from our first assignment. As we provisioned our original database script in MySQL, RDS allowed for easy transition and deployment of our database server to the cloud.

After creating the RDS database server, we deployed our two webservers to the cloud. This was achieved using Amazon's EC2, which allows for deployment and development of virtual servers hosted through the AWS cloud computing platform. We deployed our client and admin webservers as t2.micro instances of EC2. T2 instances are a general-purpose instance types that come at a low-cost (one of the cheaper instances), which is ideal for dealing with our non-convoluted application that has limited interactive capabilities and latency requirements.

For our EC2 instances to work, we first had to establish certain parameters:

Security Groups:

There are 3 defined security groups for our servers:

- A security group for http traffic - port 80, 443.
- A security group for ssh traffic – port 22
- A security group for MySQL access – port 3306

Subnet ID:

The EC2 instances availability zones are set under us-east-1a. From using the command **aws ec2 describe-subnets –region us-east-1,** we were able to obtain the us-east-1a Subnet ID **"subnet-b2442fff"**, which is stated in our Vagrantfile.

AMI:

AMI is used to determine what disk image that VM should boot from and what the EC2 instance should use. The AMI was obtained through using Ubuntu's 'Amazon EC2 AMI Locator' where **ami-0f40c8f97004632f9** was the ID for us-east-1.

Shaun Liew 2553068
Josh Casey 3023930

After establishing these parameters and fixing some permission issues, we were able to launch vagrant (**vagrant up**) and access the webpages hosted by the cloud.

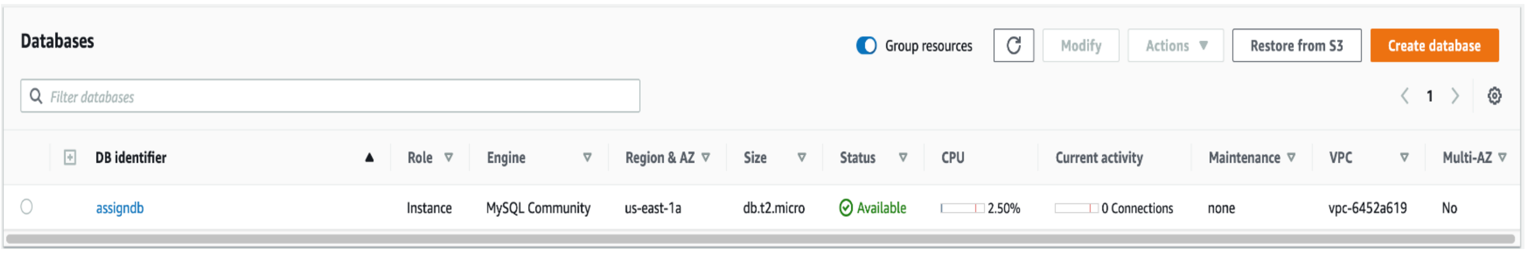
How the to reach our application in the cloud, and what a user can do to easily interact with it:

Our application is no different to the one created in our first assignment in terms of functionality. It is still a simple store that allows users to simulate ordering products on the client server, then being able to view the orders on the admin server.

A user can reach these webservers by inputting these links into their browser:
Client Server: http://ec2-34-207-78-74.compute-1.amazonaws.com/
Admin Server: http://ec2-54-235-61-208.compute-1.amazonaws.com/

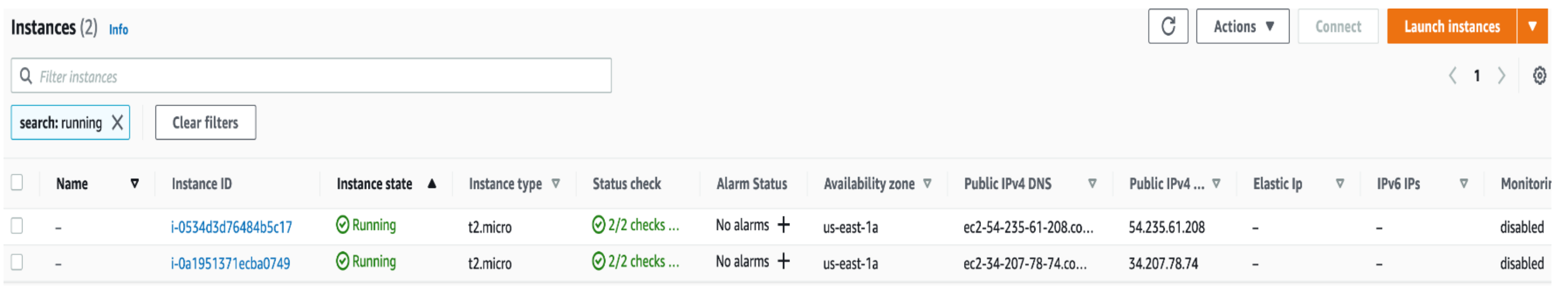
A user can then interact with the client server by selecting the products listed and entering customer details to place an order. A user can then access the client webserver to view the orders placed, and update the shipping status.

Database:



DB identifier	Role	Engine	Region & AZ	Size	Status	CPU	Current activity	Maintenance	VPC	Multi-AZ
assigndb	Instance	MySQL Community	us-east-1a	db.t2.micro	Available	2.50%	0 Connections	none	vpc-6452a619	No

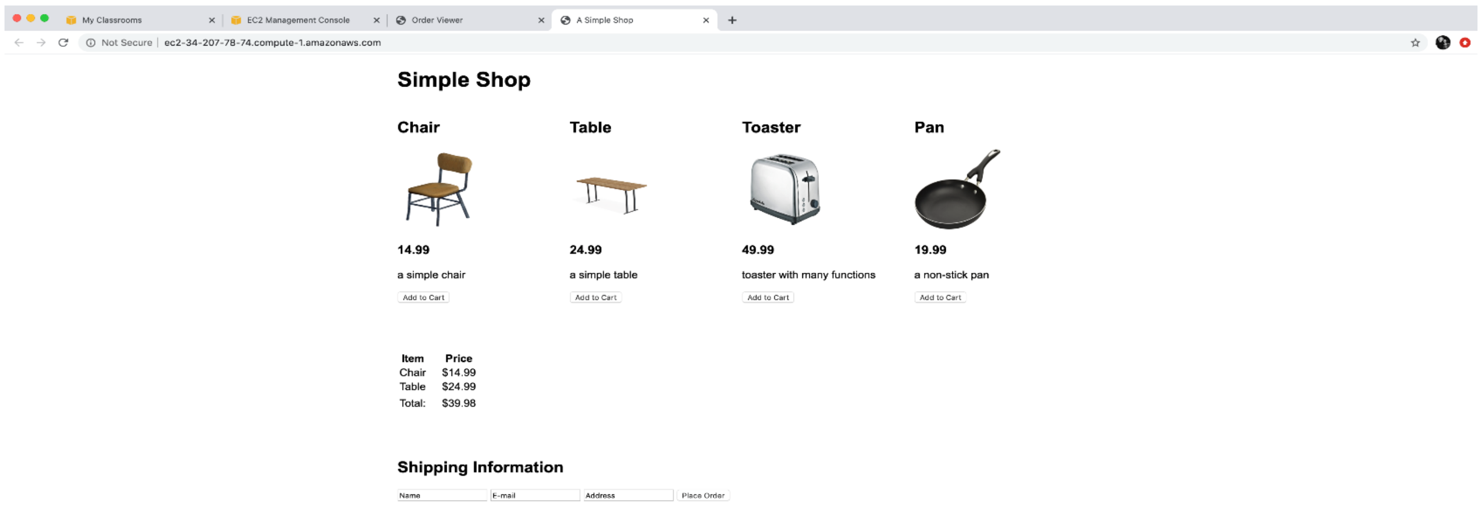
EC2 instances:



Name	Instance ID	Instance state	Instance type	Status check	Alarm Status	Availability zone	Public IPv4 DNS	Public IPv4 ...	Elastic Ip	IPv6 IPs	Monitorir
–	i-0534d3d76484b5c17	Running	t2.micro	2/2 checks ...	No alarms +	us-east-1a	ec2-54-235-61-208.co...	54.235.61.208	–	–	disabled
–	i-0a1951371ecba0749	Running	t2.micro	2/2 checks ...	No alarms +	us-east-1a	ec2-34-207-78-74.co...	34.207.78.74	–	–	disabled

Shaun Liew 2553068
Josh Casey 3023930

Client webserver running:



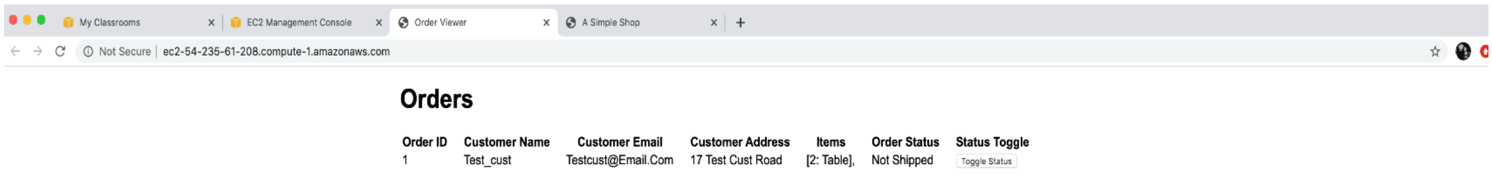
Simple Shop

Chair	Table	Toaster	Pan
14.99	24.99	49.99	19.99
a simple chair	a simple table	toaster with many functions	a non-stick pan
Add to Cart	Add to Cart	Add to Cart	Add to Cart

Item	Price
Chair	$14.99
Table	$24.99
Total:	$39.98

Shipping Information

Admin webserver running:



Orders

Order ID	Customer Name	Customer Email	Customer Address	Items	Order Status	Status Toggle
1	Test_cust	Testcust@Email.Com	17 Test Cust Road	[2: Table],	Not Shipped	Toggle Status

Use of cloud service in the application:

Another cloud service implemented in this assignment (aside from EC2) was a Relational Database Server on AWS. Through the implementation of our database system on the cloud, it allowed us to remove one of our virtual machines running on Vagrant to limit traffic and load up time of the application. We implemented RDS in our project to take advantage of the utilities given to us, as RDS allows for an easy set-up, operation, and scalability. To form our database in AWS, we had to first create an instance of an RDS that was under a MySQL engine. After that, we created a security group to allow interaction between virtual servers and adapted the code for the webservers to allow for communication with the database server.

How the project was incrementally developed and debugged:

This assignment has been developed through initial setup stages as well as debugging. For the initial stages of setting up deployment of our VMs to the cloud, we referred to the lab material provided. However, during the process of completing our deployment, there were several issues incurred due to security permissions restricting user access. For instance, when attempting to access our webpages, we would encounter an error message stating: 'Permission Denied 403'. To overcome these issues, we researched the ports that required to be allowed in order to surpass inbound traffic. These changes are illustrated through the GitHub commits made in our project repository.