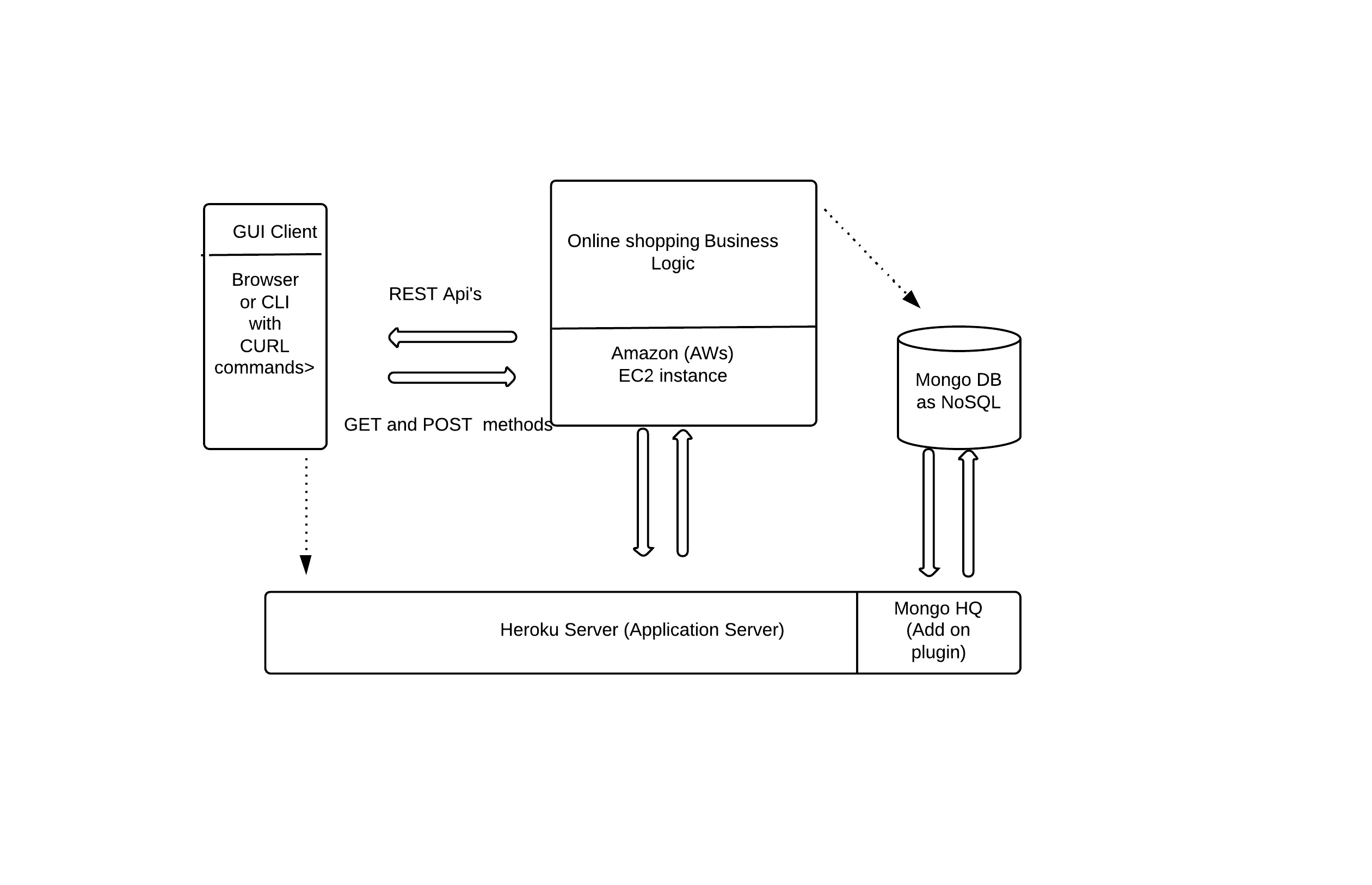
## Design

### Architecture

Our current design of online shopping website is based on pure MVC architecture. GUI being the controller, database being the view and model being our business logic. Below figure describes the high level architecture of the system combining the main components



<Fig: High level design of online shopping website

The above figure depicts four main components helped completing our design.

1. AWS Server : Key infrastructure in our design is from Amazon web service. Amazons infrastructure as a service provided us with a placeholder for us to keep our business logic in cloud and making it accessible from anywhere. Amazon EC2 instance was used to host the system logic.
2. MongoDB : As part of No-SQL database, Mongo DB was chosen to be our datastore . Since it is an e-Commerece website, the user or product data is usually semistructred and in form documents. It is not scalable to have the data stored on SQL databases. MongoDB is best suited on this context and for the known reasons that it supports document type store. Also, we have lot many api’s and drivers built on top of MongoDB which can be used to buid across platforms and languages.
3. Heroku Server : There are various reasons to choose heroku as our web application hosting server. Heroku provides a cloud platform to build and host web applications. It provides tools and technologies in the form of add-ons, services, app etc, which makes the developers job easier in developing, testing and maintaining the tasks
4. Client : The client in our design is a browser and command line interface to test the server side logic.

Workflow

* Business Logic is placed on AWS server. All source code and other files are located on amazon EC2 instance.
* Once code is checked onto GitHub, it is then pushed onto Heroku server for the application hosting onto cloud.
* Client sends request to server in the form REST api’s . Server verifies and contacts database for sufficient information. Server then encapsulates the response and sends back to client. This process is ongoing , until the server is up and running.

Components

The framework contains basically 2 python scripts and rest being the HTML,css,js etc

|-shopping\_cart.py

|-topicDAO.py

|-static

|assests

|css …etc

|-views

|-index.tpl

|-login.tpl …etc

* Shopping cart.py contains all logic related REST api’s
* topicDAO.py is our database object methods. This contains all api’s related to database connectivity
* views- index.tpl, login.tpl are all the templates/html pages which is the front end for our application
* static folder contains all static pages,images,links,css which consumed by the front end applications

### Implementation

Environment: Ubuntu 13\_0\_4, Windows8, AWS EC2 instance

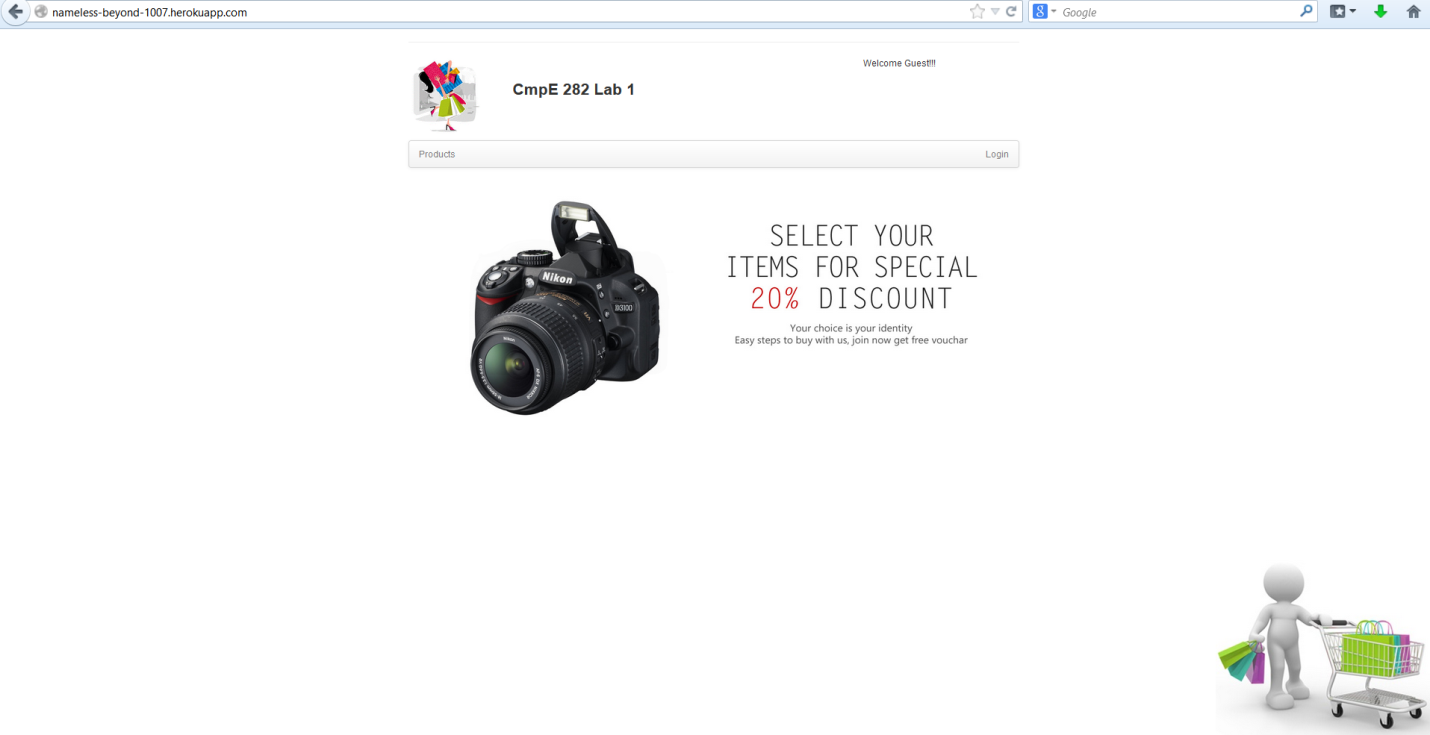
Tools /Languages: Heroku, GitHub, Python, MongoDB, Bottle Micro Framework, REST

Approaches:

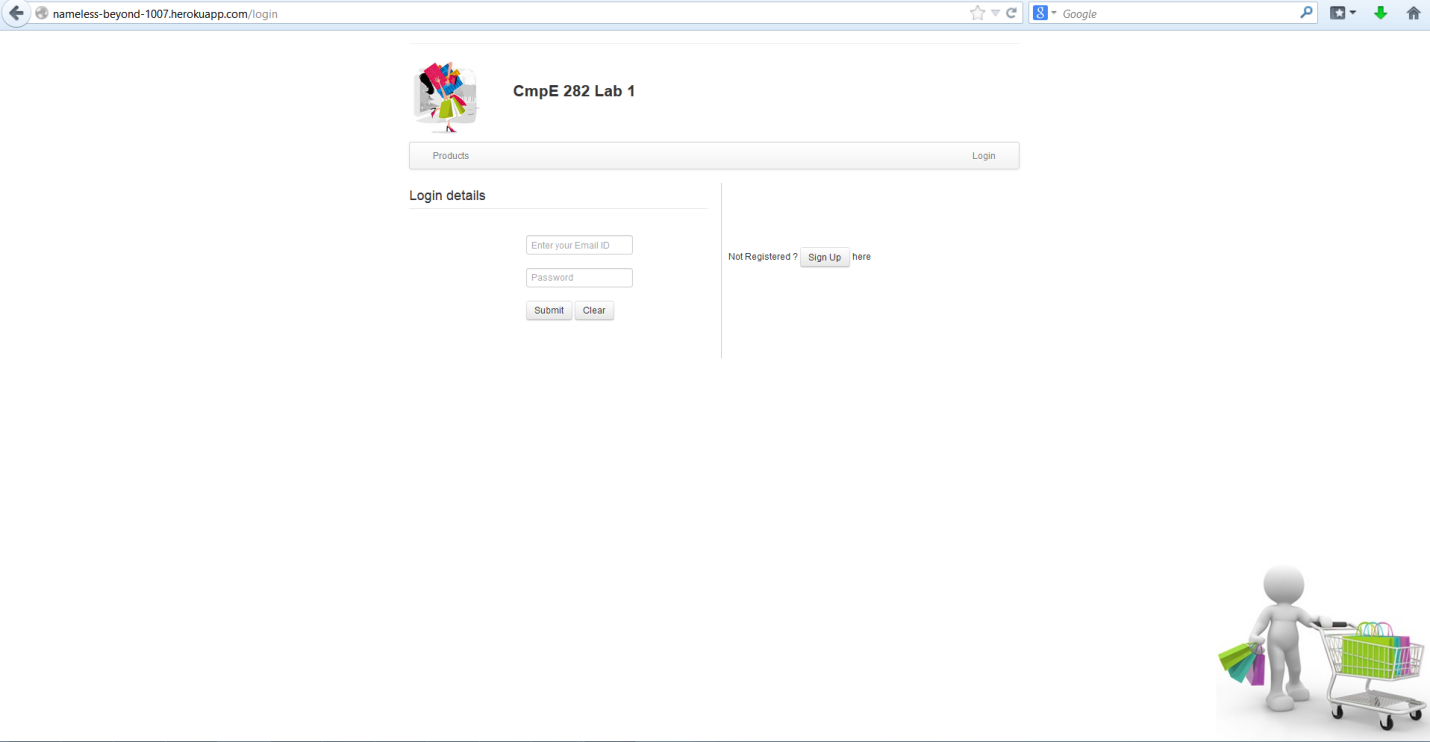
1. Micro framework ‘Bottle’ was used to build the application. The entire application is purely built on Python. ‘Bottle’ is built on top of python. This project uses bottle as it key weapon to develop REST api’s
2. Python along with bottle takes much lesser time to develop and deliver the features. Pythons is far easiest to learn and adapt, making it more acceptable among coders.
3. Single database was used to remove the overhead of maintaining multiple databases. User information is maintained in MongoDB, which is comparatively less complex than storing them in multiple databases with having costly operation such as search and join queries.

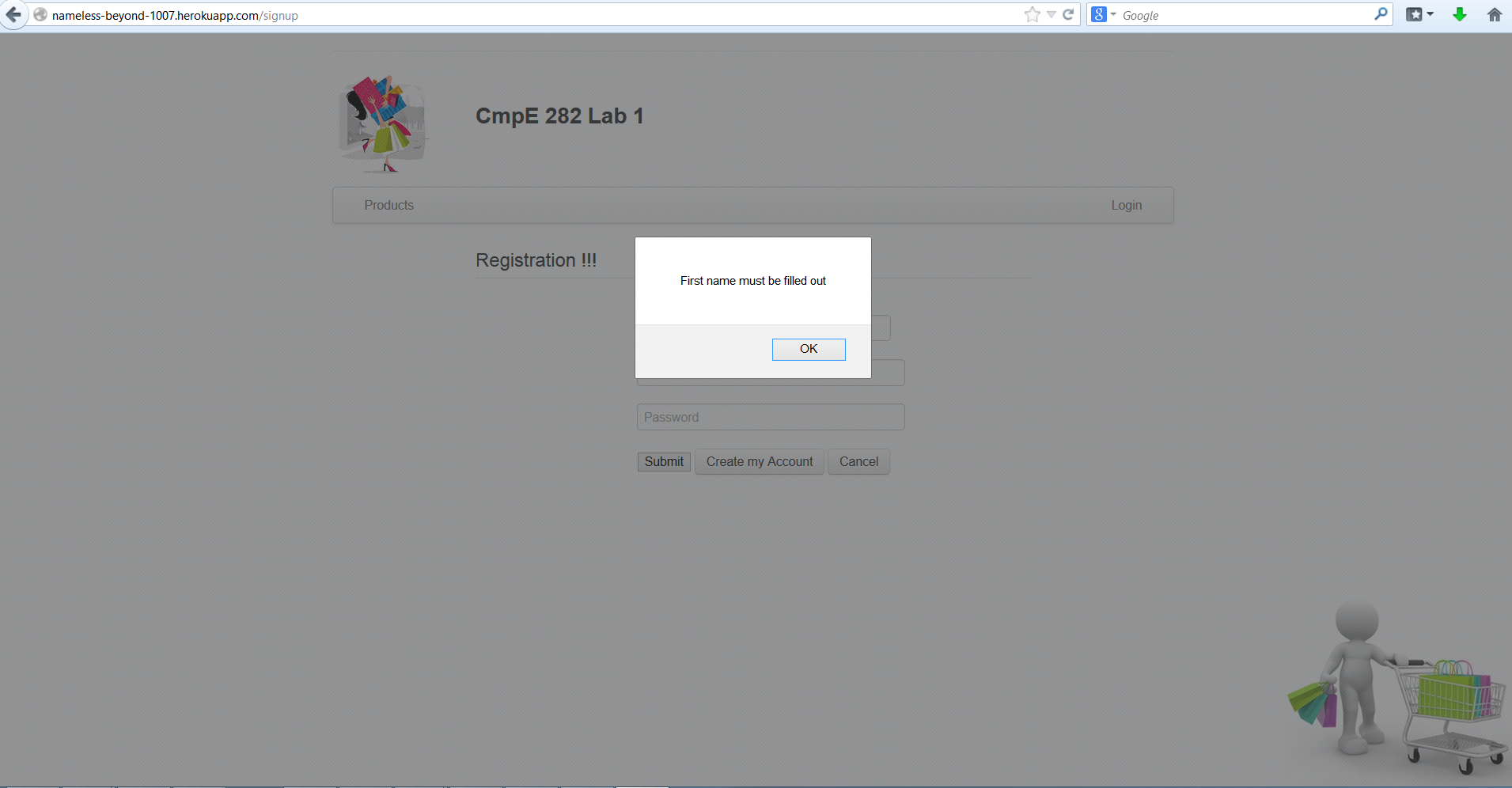
### Screen Captures:

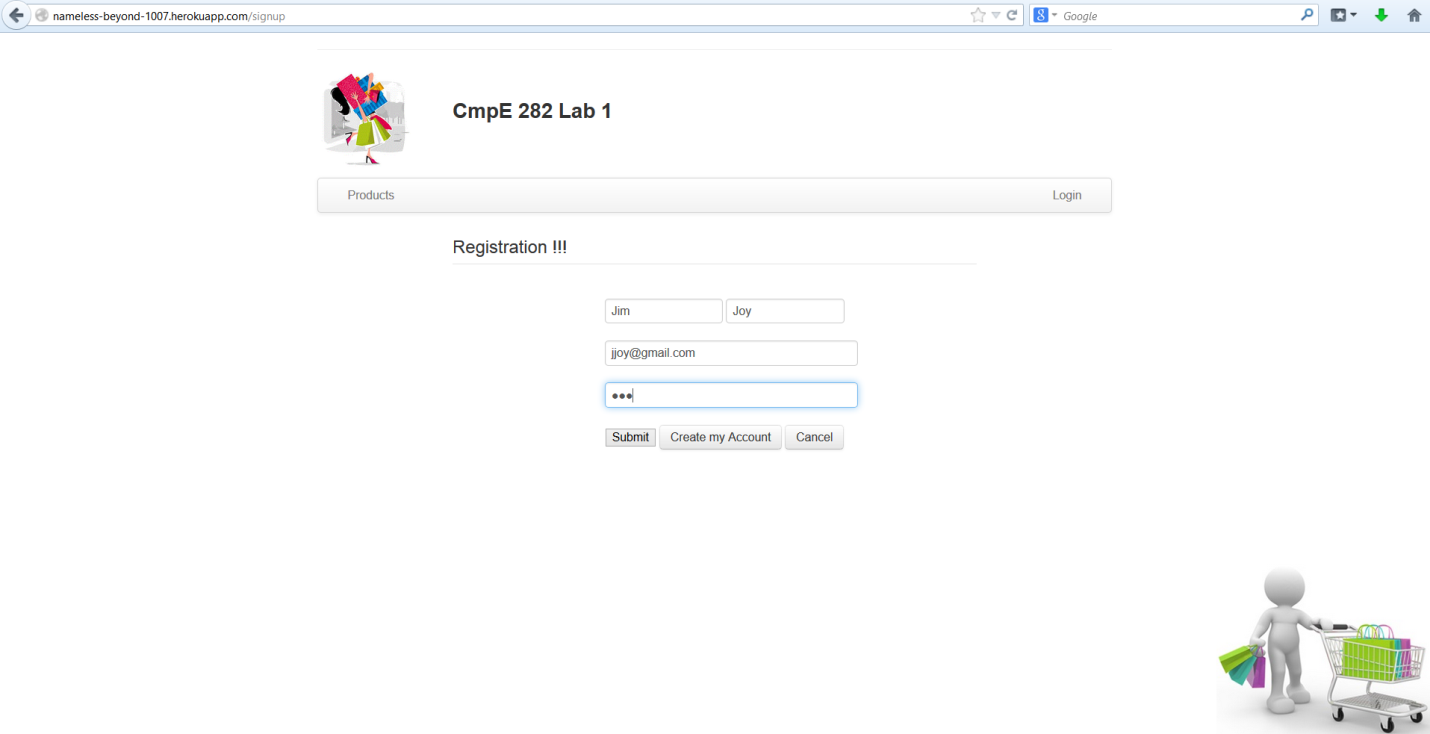
1. Home Page , <http://nameless-beyond-1007.herokuapp.com/>



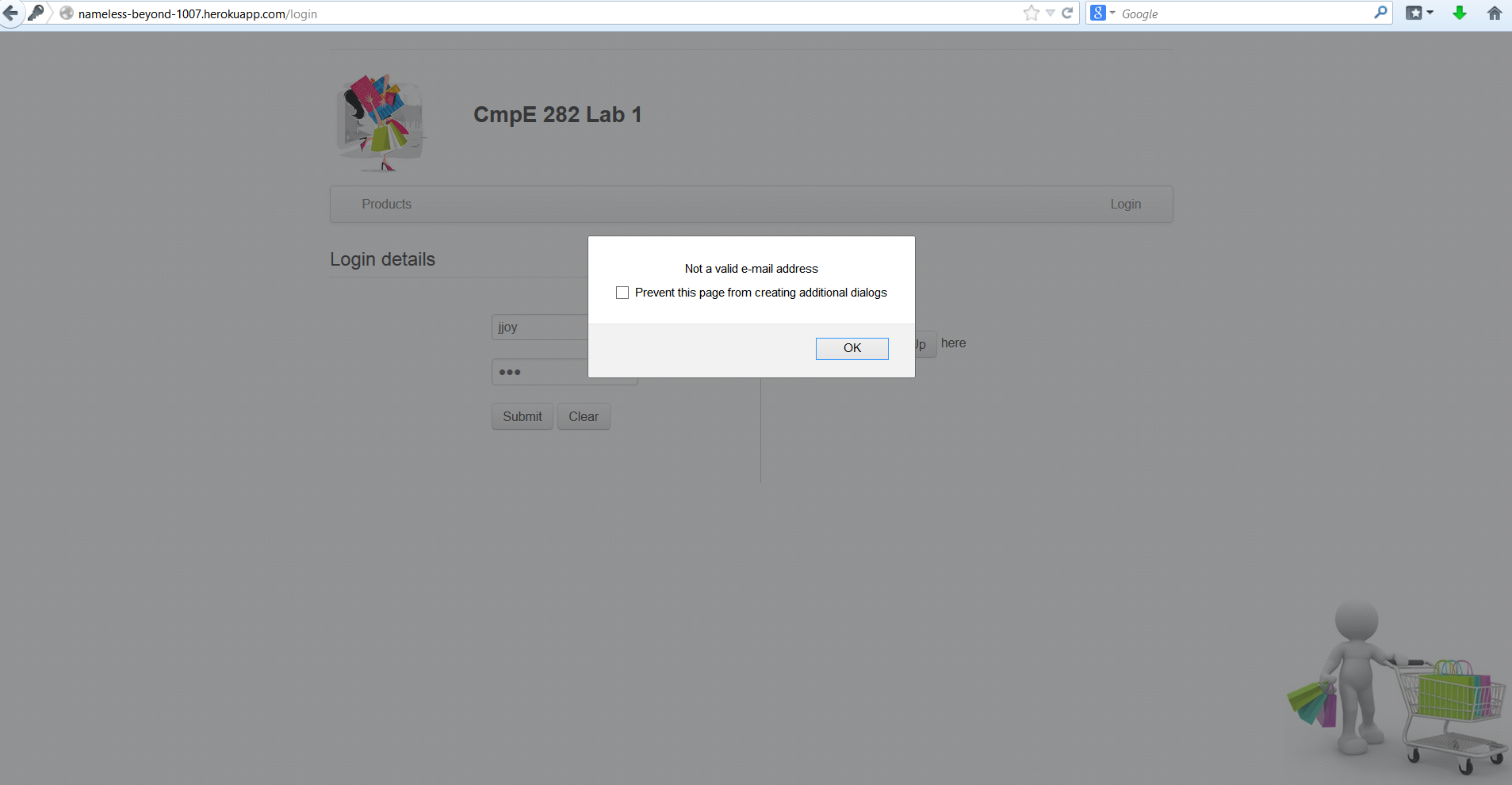
1. Login and signup links. Click on Login link for this screen



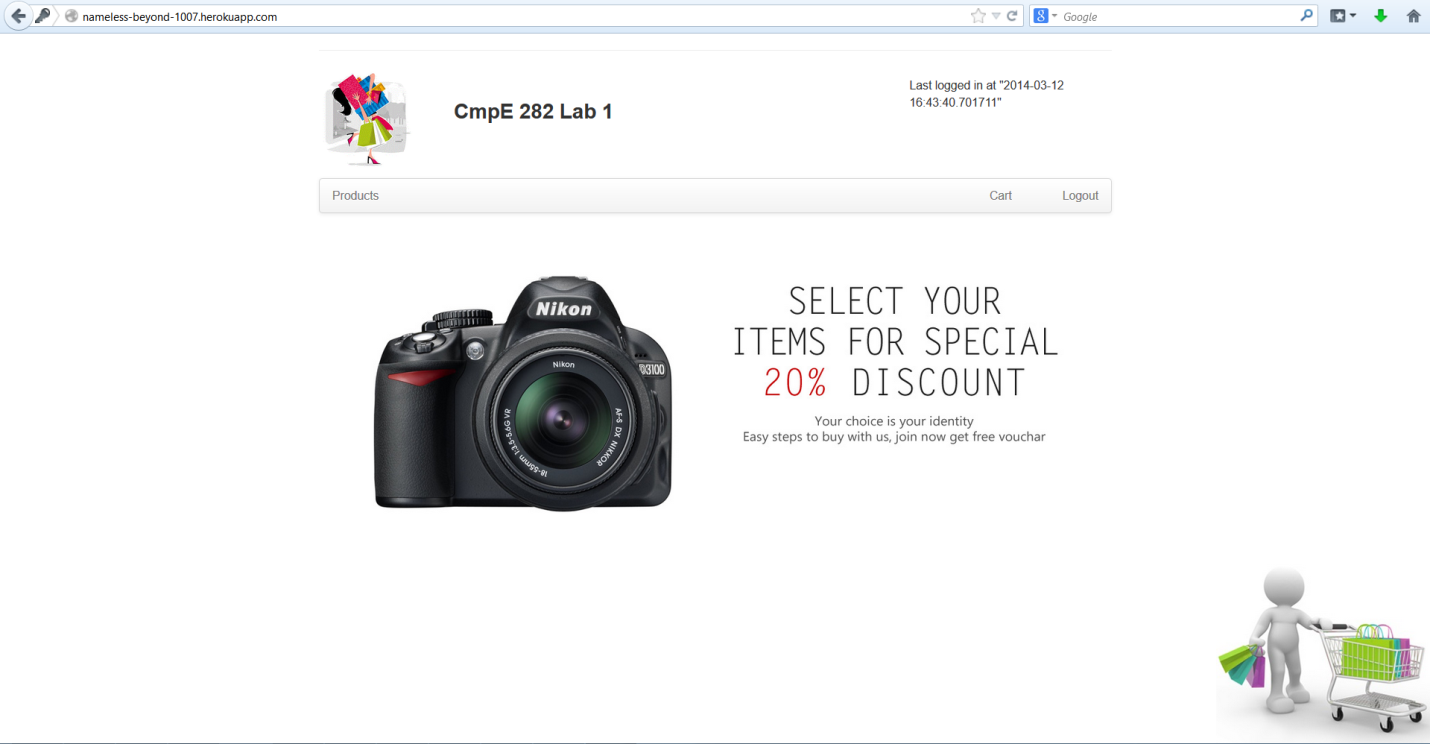
1. Sign Up page. All fields have been validated . Warning prompt for not filling Firstname
2. Sign Up success, now login with same credentials.



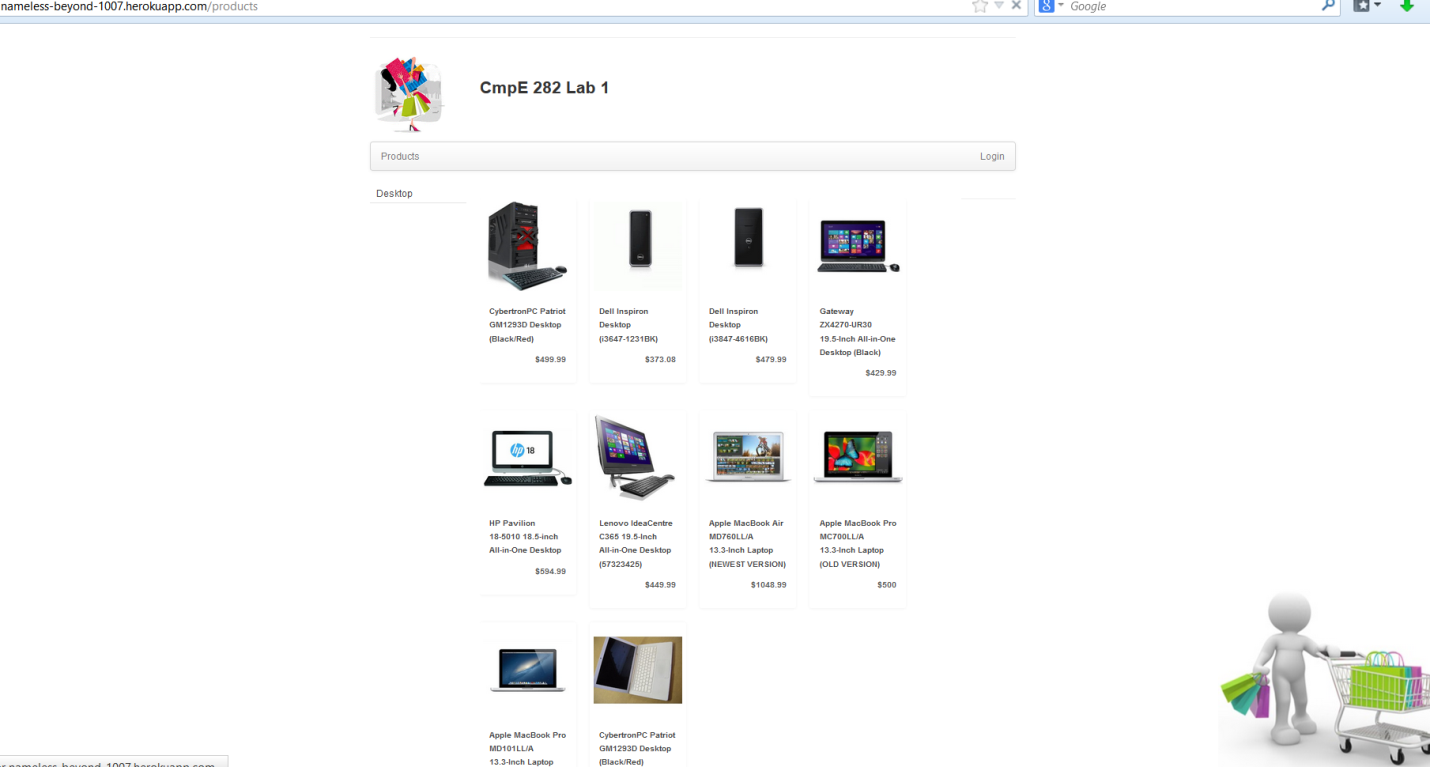
1. Login validation. Warning on wrong email



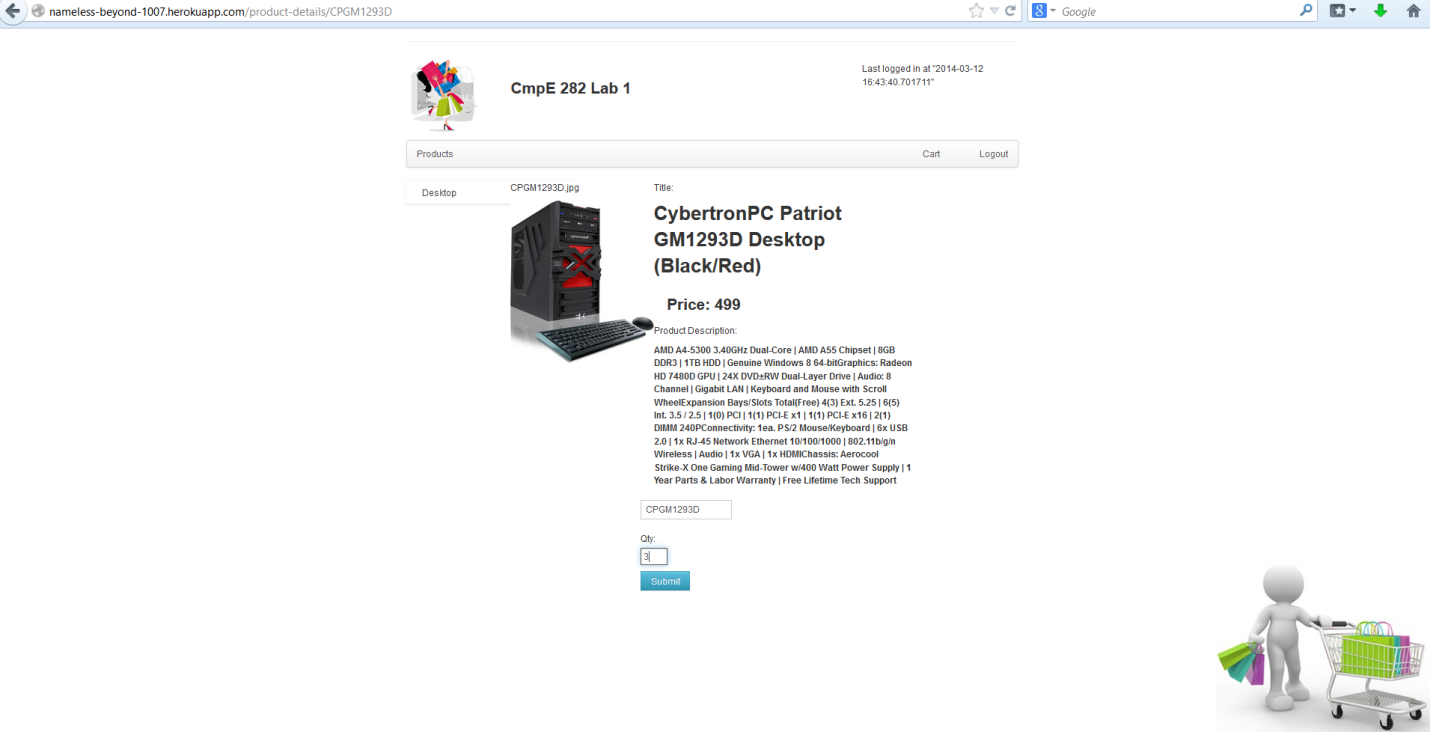
1. After successful login. This is the home page showing last logged in time on top right



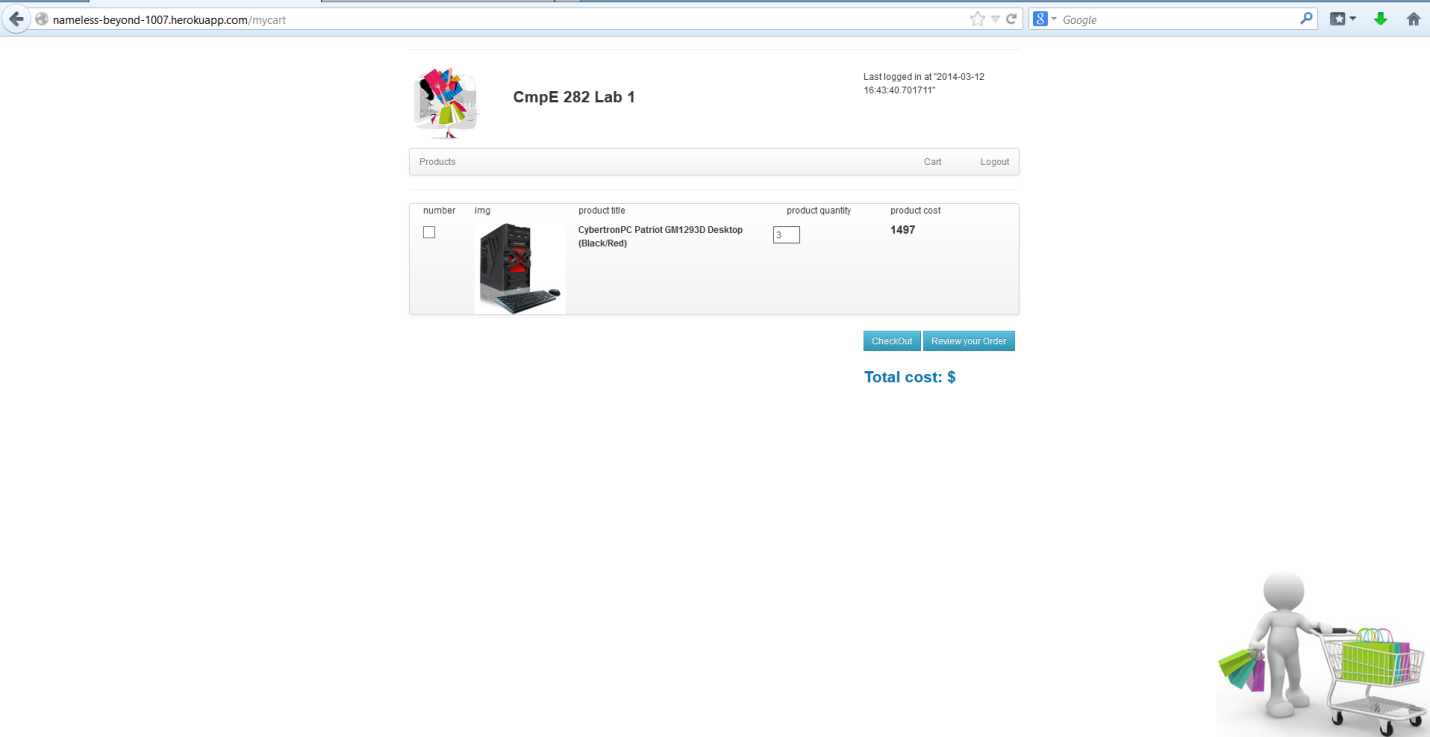
1. Products page shows various products



1. Product Details with quantity to select and to add to cart



1. Cart showing after adding 3 quantities of same product with total cost being displayed



1. Credit cart details after checkout from cart with each field validation. Warning message shown for not entering the name

