PROJECT FINAL REPORT

“Binary Bazaar”

Online Marketplace

INFO 2413 S10

System Development Project

April 6, 2017

Submitted By:

William Corcoran

Lucas Magnien

Nirosh Ratnam

Yu-Tang Yu

Submitted To:

Mehdi Talwerdi

Office G2060

Kwantlen Polytechnic University

Table of Contents

[**Project Definition and Description** 3](#_Toc479244515)

[**Research and Study, Links** 3](#_Toc479244516)

[**Plan for Implementation** 4](#_Toc479244517)

[Home 5](#_Toc479244518)

[New Item 5](#_Toc479244519)

[Show Item 6](#_Toc479244520)

[Add Comment 6](#_Toc479244521)

[Sign up 7](#_Toc479244522)

[Login 7](#_Toc479244523)

[Cart 8](#_Toc479244524)

[CheckOut 8](#_Toc479244525)

[**Gantt Chart** 9](#_Toc479244526)

[**ER and Database Schematic**1 10](#_Toc479244527)

[**Actual Implementation Architecture** 11](#_Toc479244529)

[**How To Use the Application** 11](#_Toc479244530)

[**Obstacles** 12](#_Toc479244531)

[**Risk Assessments** 12](#_Toc479244532)

[**Conclusion** 13](#_Toc479244533)

**Project Definition and Description**

For our project, our intention was to design and build an online store application (cloud SAAS) providing online POS for customers of client businesses, including as stretch goals, collation of shopping information for business statistical analysis, and the creation of financial statements based on collated information. The long-range focus of the project would be to provide exposure and feedback for business startups with new business identities and products as quickly and easily as possible, within a community setting.

An important aspect of this concept is ease-of-use, facilitating the rapid development of new identities and the introduction of new products to the marketplace community. This would provide new businesses with the fastest and easiest “proof-of-concept” possible. The final application includes a web-based service with attached database to facilitate the marketplace. Comprehensive client and customer portals are developed to provide access to services, eventually including access to social aspects and analytics.

**Research and Study, Links**

Many online materials were referenced and utilized for the proper implementation of the technologies used during the development of this project. The following is a list of some of the most notable:

**W3Schools**. A popular web site (run by a Norwegian group) providing complete references to web technologies, including HTML, CSS, JavaScript, Bootstrap, and jQuery, among others. <https://www.w3schools.com>

**jQuery Learning Center**. Comprehensive guide to using this JavaScript library for client-side scripting, for UI effects. <http://learn.jquery.com>

**GitHub Desktop User Guides**. GitHub was used as our main code and documentation repository. Access was possible via web services, and also through desktop client apps, such as GitHub Desktop. <https://help.github.com/desktop/guides/>

**GitKraken Support**. Another GitHub client app. <https://support.gitkraken.com>

**Slack**. Slack is a cloud-based team communication tool that allows real time chat over multiple channels. This tool was used during remote group developmental sessions, and general updates and situational awareness. <https://slack.com>

**Sublime Text 3 Documentation**. Sublime was used as our primary source code editor. It is designed to natively support many programming languages, such as HTML, CSS, JavaScript, and SQL. <http://www.sublimetext.com/docs/3/>

**Kong Documentation**. Kong was used as an API layer residing between web servers and database implementation. <https://getkong.org/docs/>

**MongoDB Documentation**. MongoDB is a document based database used in the backend of Binary Bazaar. <https://docs.mongodb.com/?_ga=1.143132085.1064778195.1491406684>

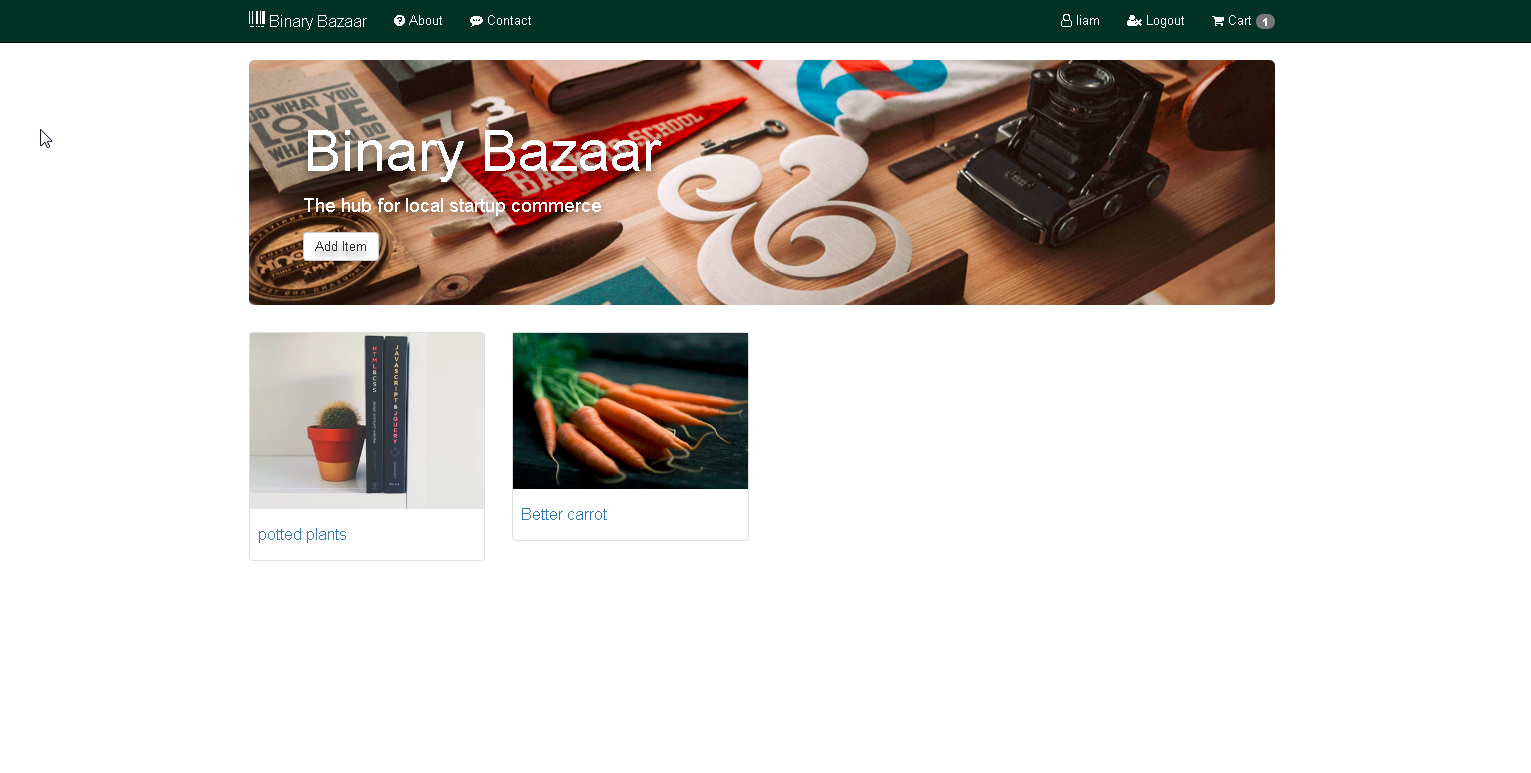
**Plan for Implementation**

**Architecture**. Several application tiers were required for the project:

At the front end, the customer and client web interfaces are made available by servers using Nginx web server, where HTML, CSS and JS web application archives are stored. The tables and data are implemented using the open source MongoDB on separate servers. In between the DB and web servers are machines running Kong, an open source API gateway and manager geared for RESTful Nginx web services. All tiers are to be implemented on Amazon Web Services cloud (PaaS). Please refer to the following diagram for a graphical representation of this design framework:

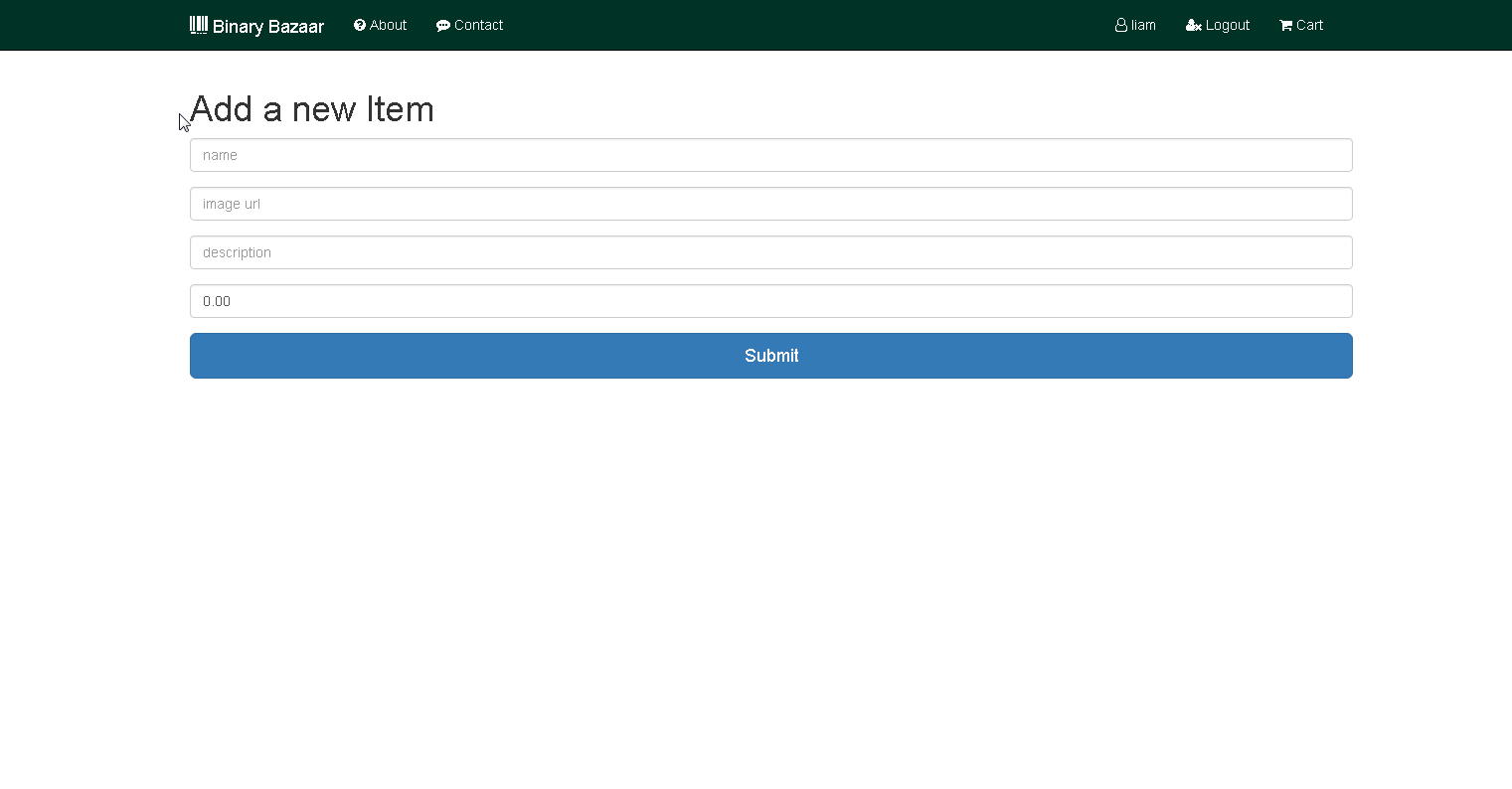
**User Interfaces**

## Home



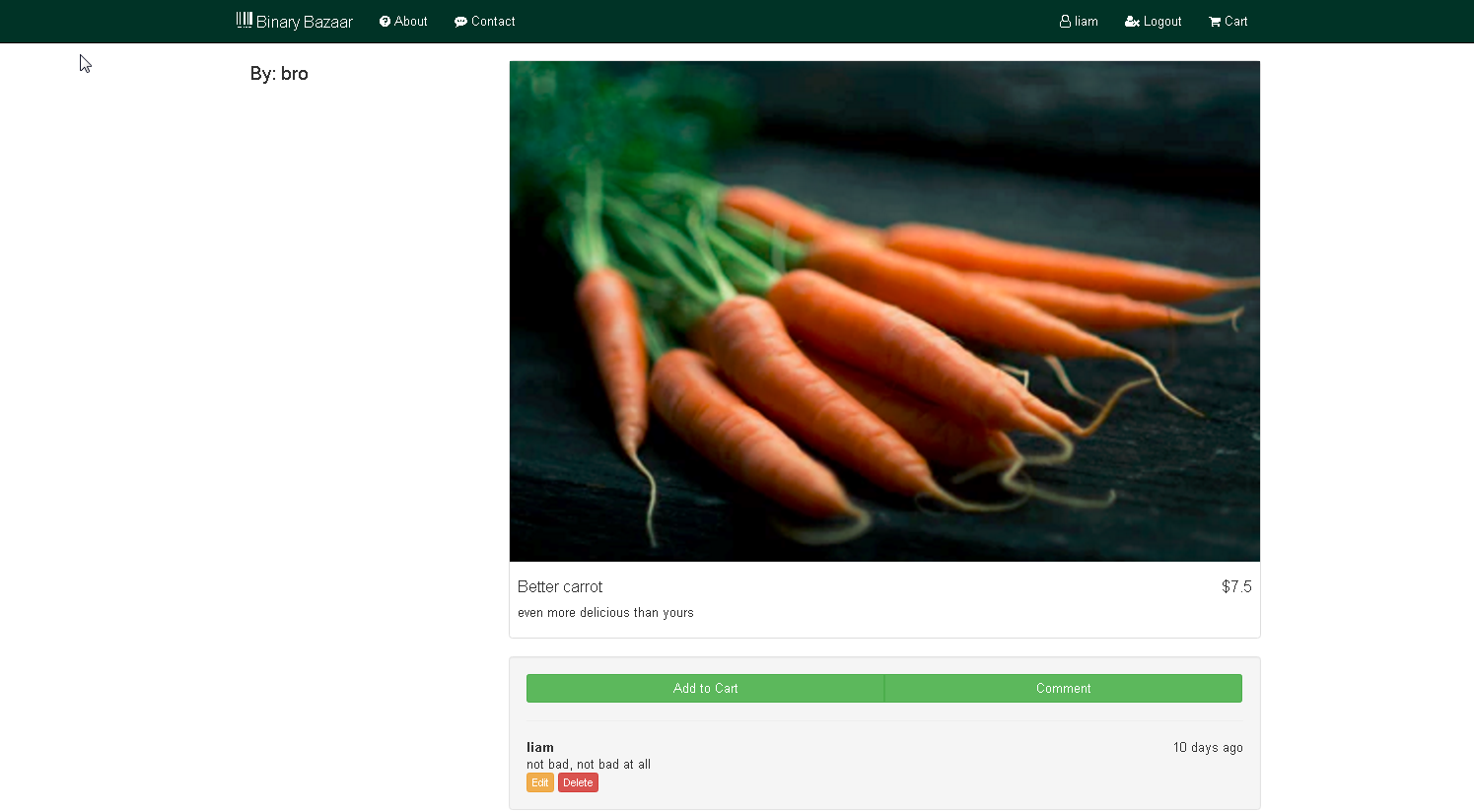
Landing page of the site where users browse items

## New Item



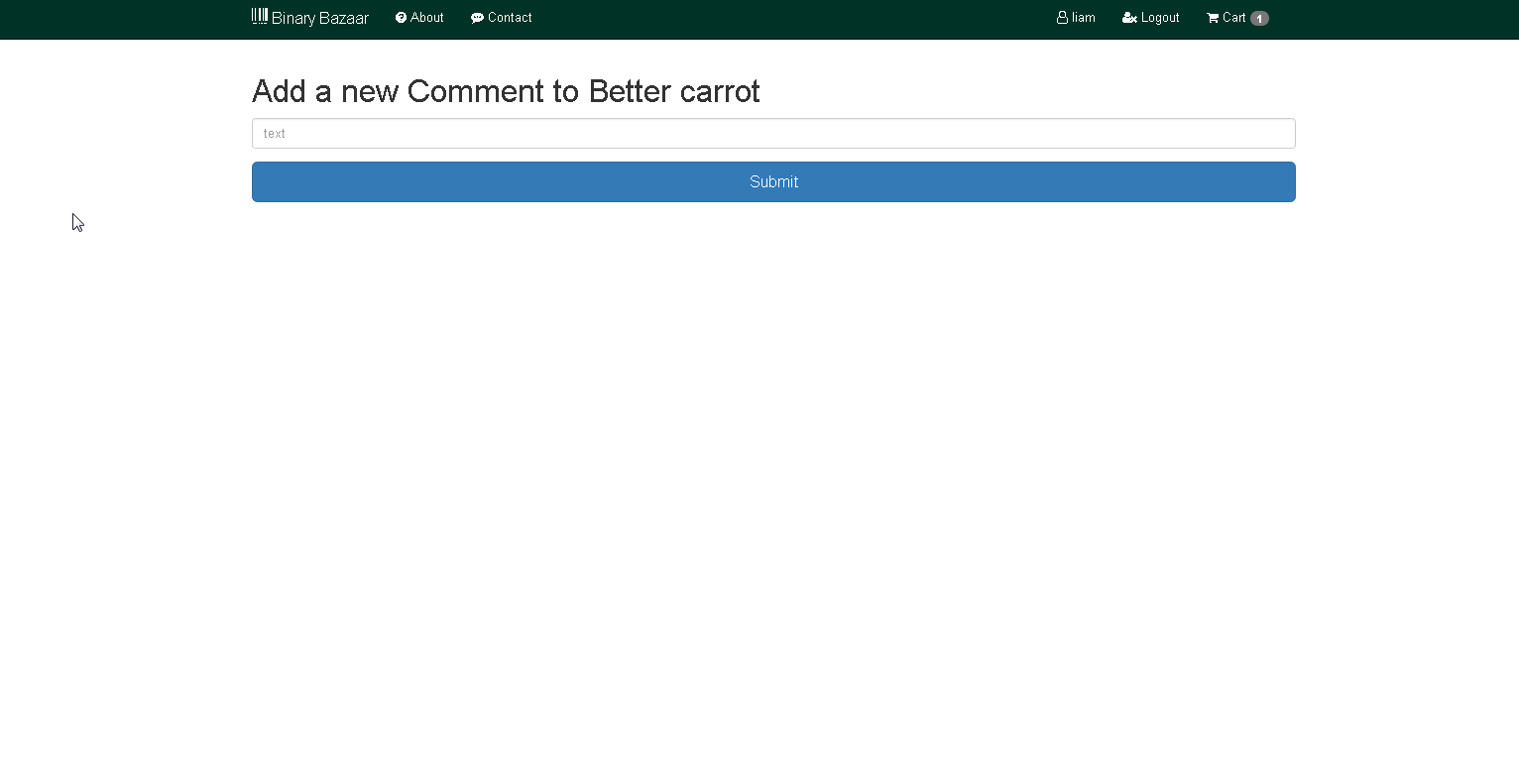
Logged in users add new Items to the store here

## Show Item



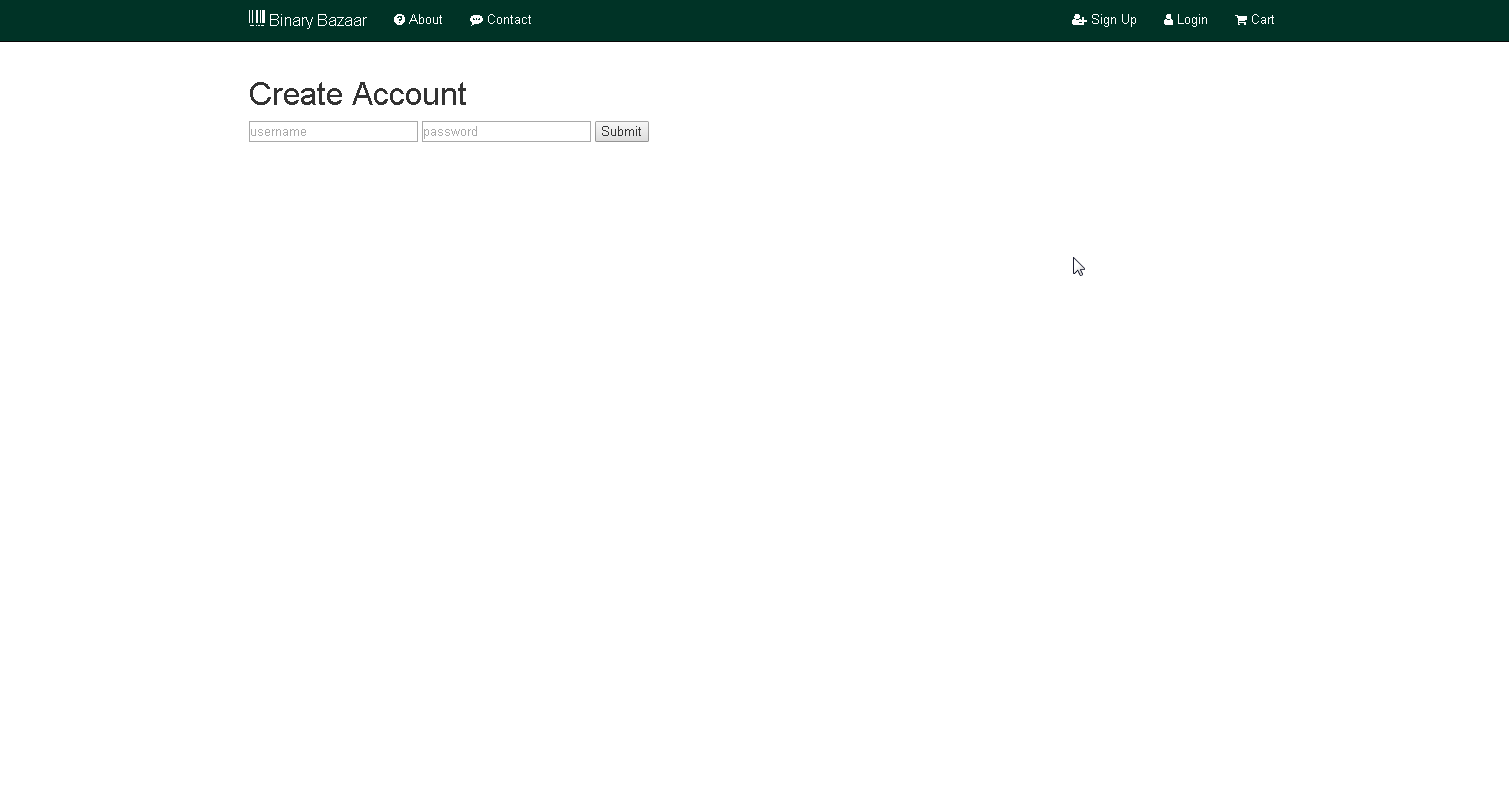
Users can view item details as well as add it to the cart as well as view comments or add/edit/delete their own comments

## Add Comment



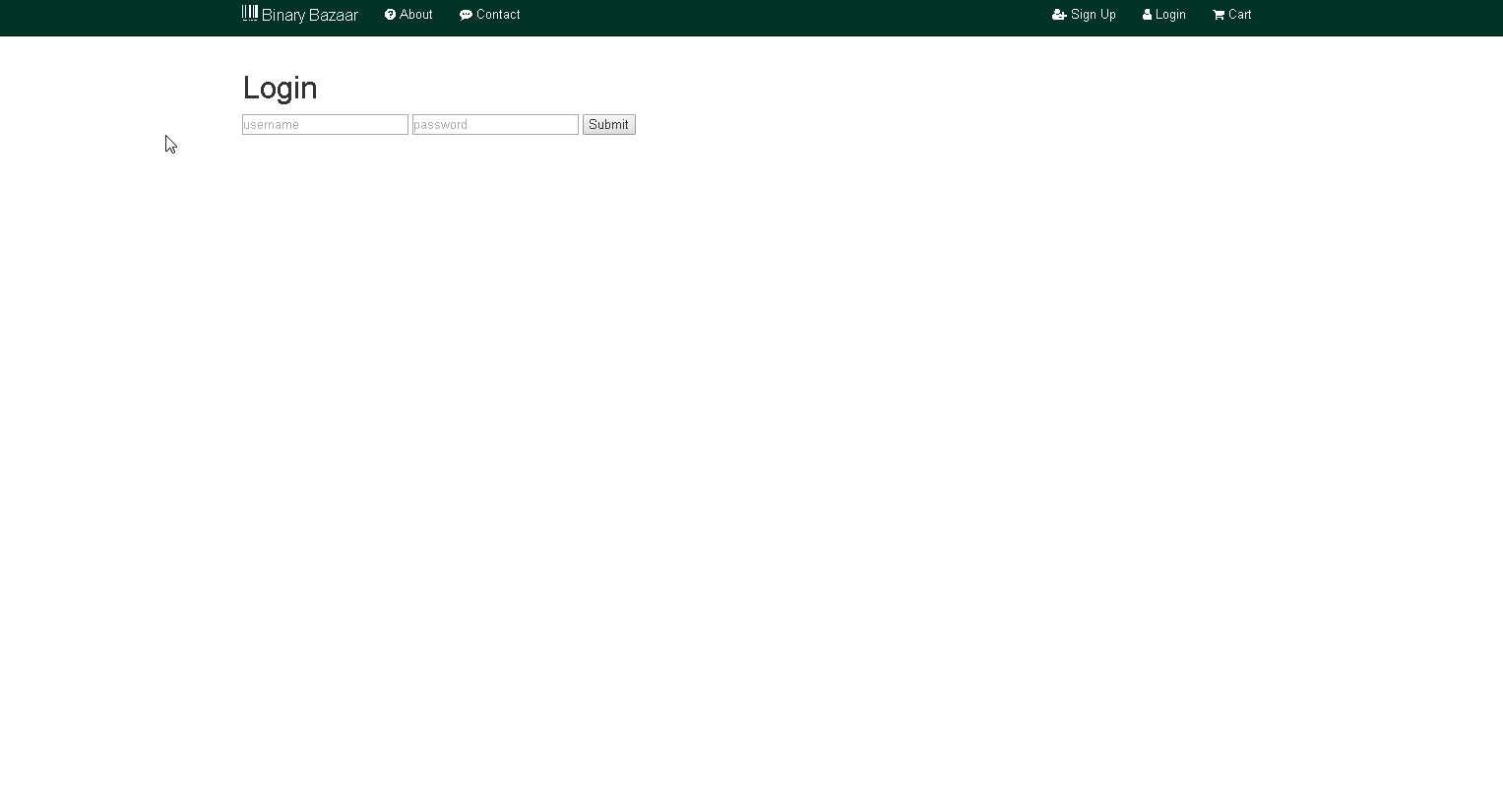
Allows user to add a comment if they’re signed in

## Sign up



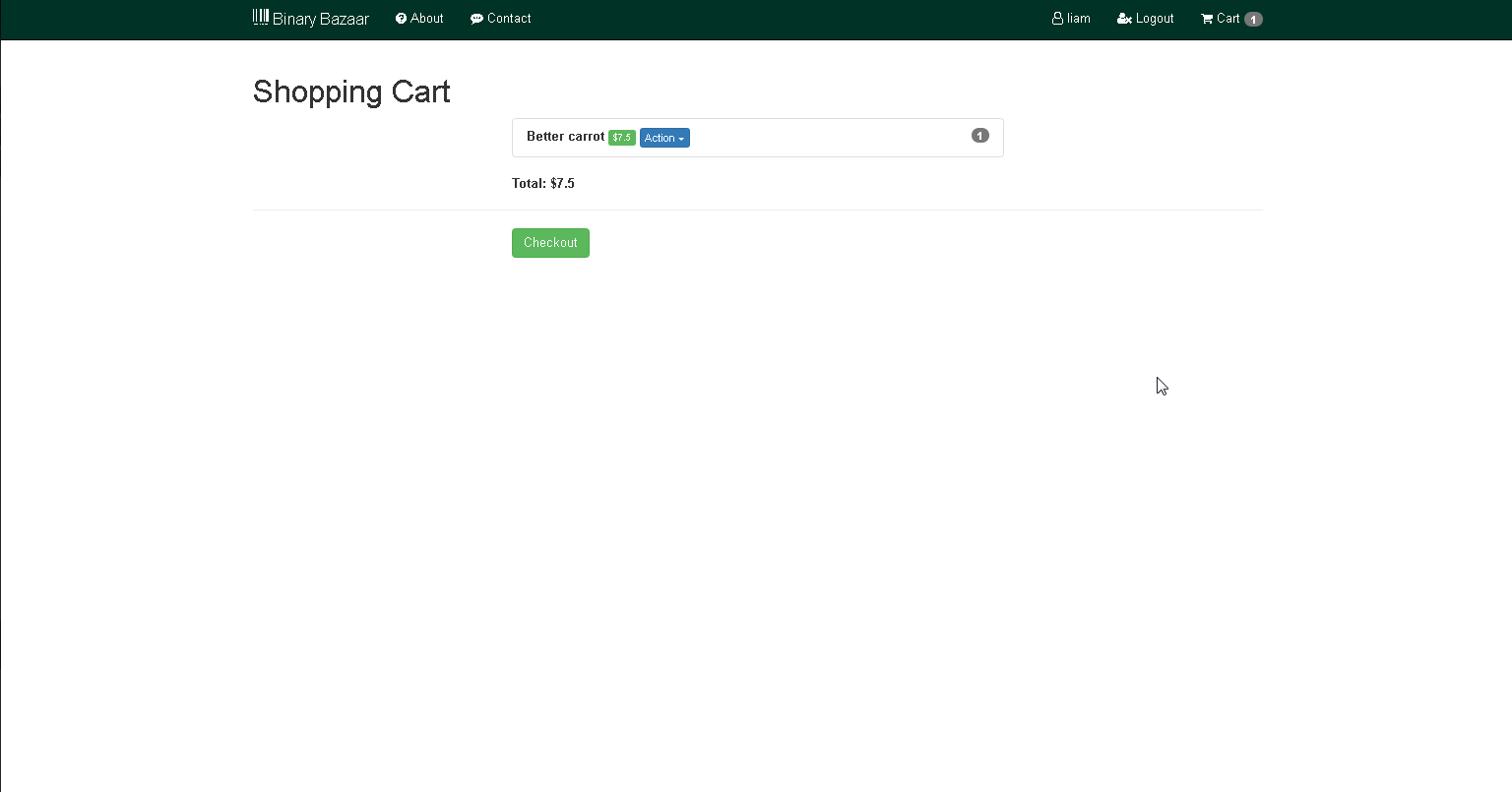
User sign up page allows them to create an account

## Login



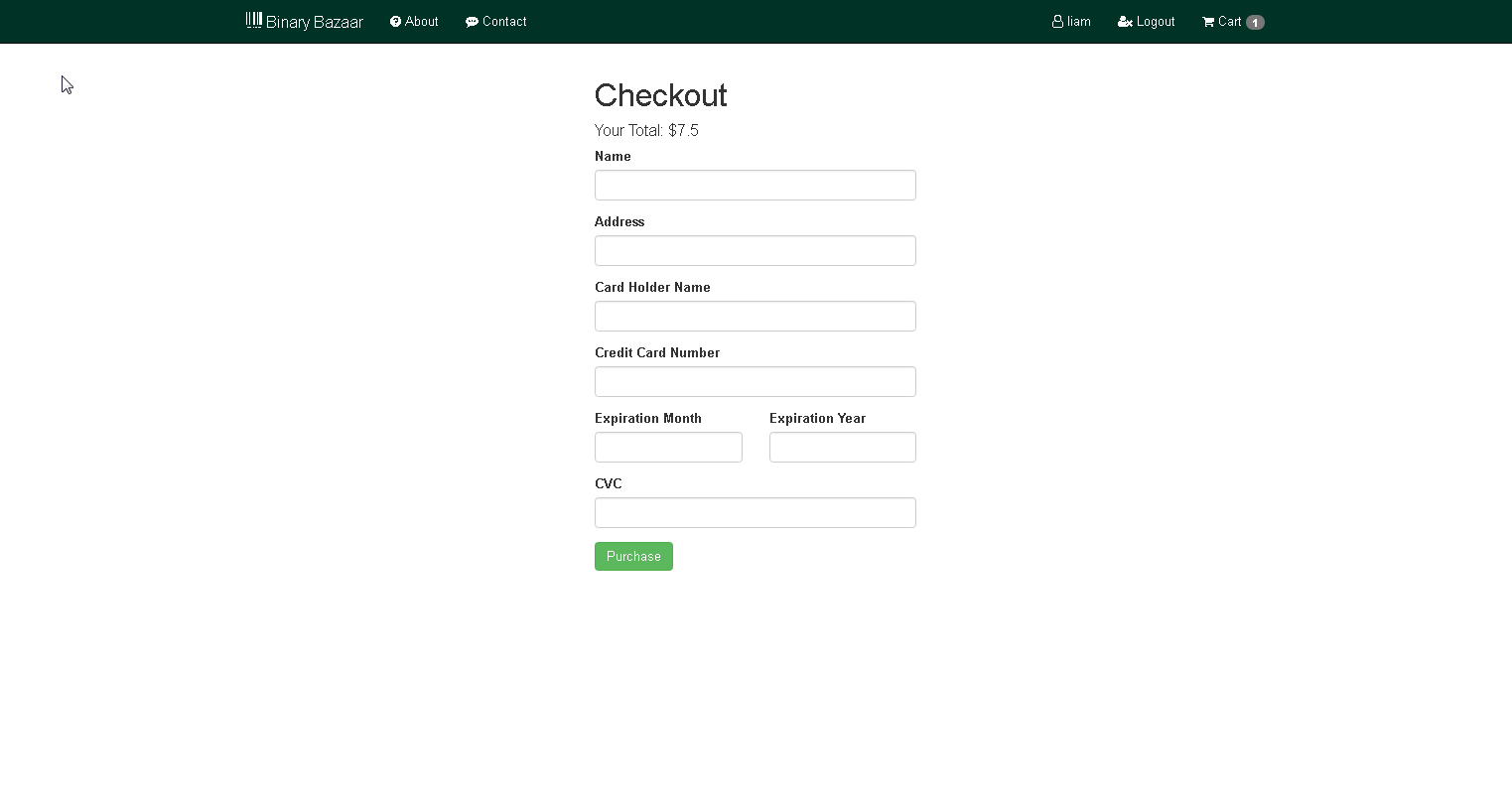
User login page allows users to login to their account

## Cart



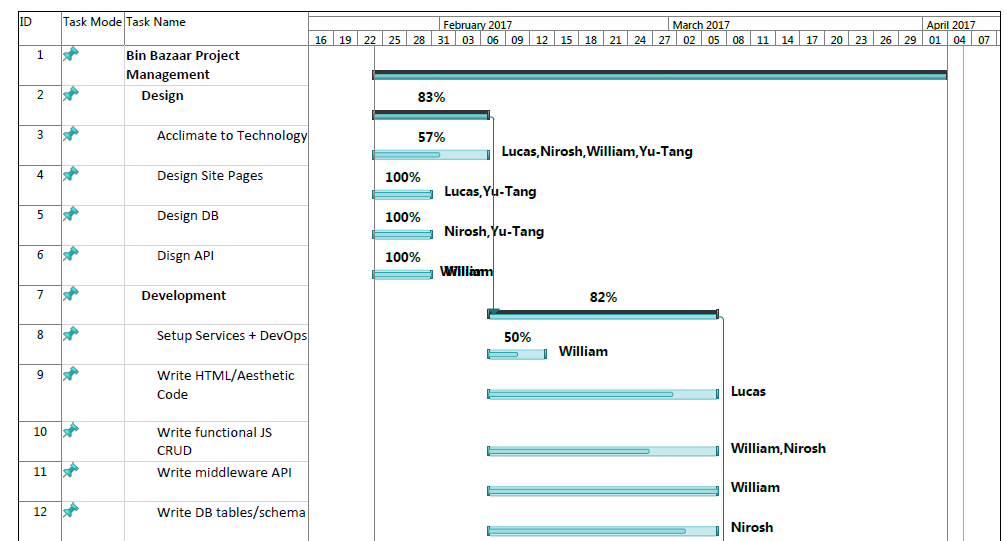
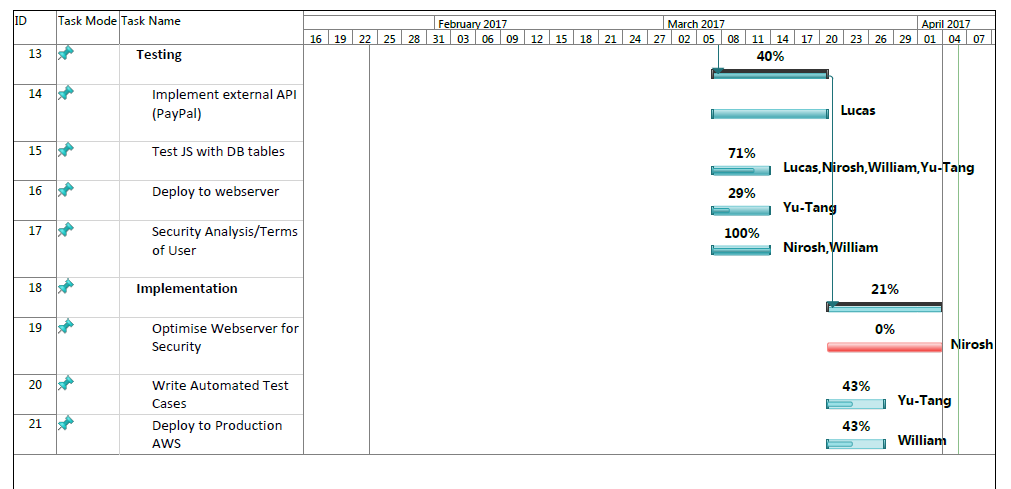
Displays items user has added to cart as well as links to checkout

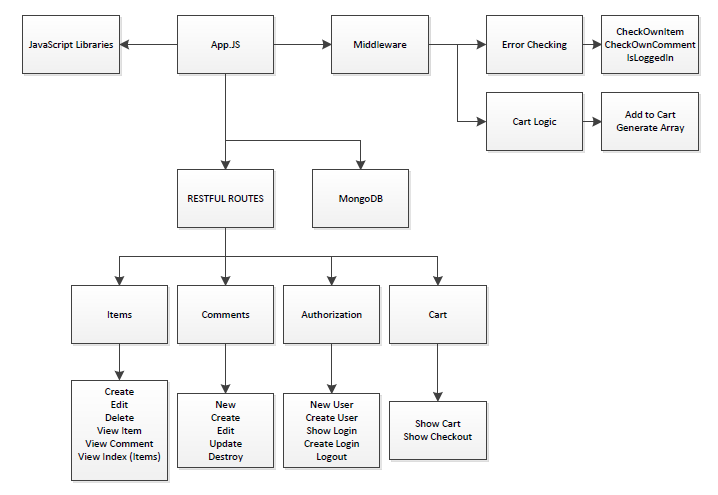
## CheckOut

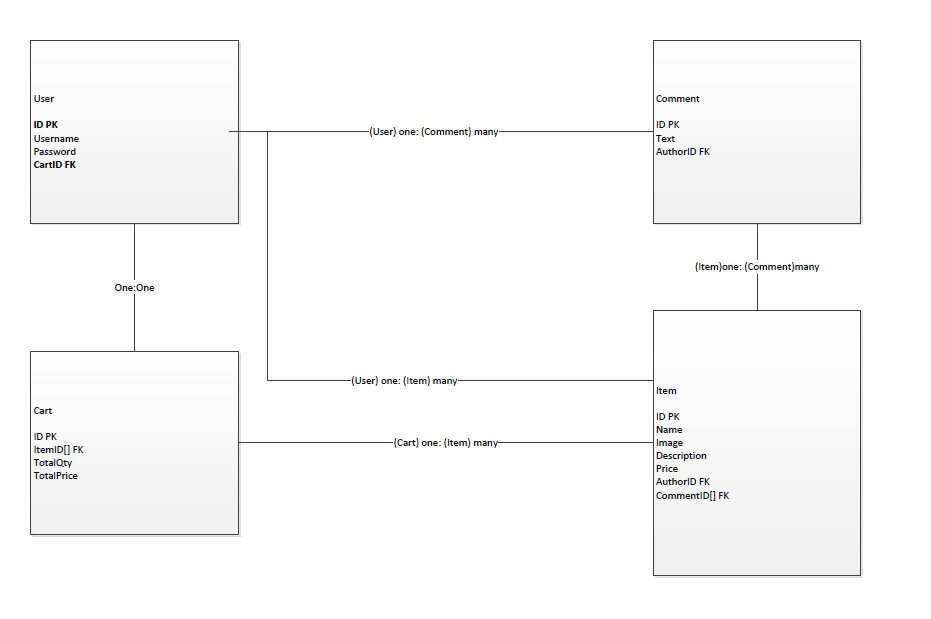


Allows users to fill payment information and trigger purchase function

**Gantt Chart**

**ER and Database Schematic**



**Actual Implementation Architecture**

Node.JS using Express framework to process routing and setup the infrastructure from a main app.js file. Using Mongo DB for database because of the ability to continuously modify tables after initial creation with ease, not to mention the storage method of JSON objects meshes particularly nicely with our JavaScript-based logic.

Restful Route setup to trigger functions based on url requests: this loads pages and passes data to the pages to be output via imbedded JavaScript through the use of .ejs files.

Cart is stored in session data, which is configured to be stored in the users browser and expire after three hours.

We also implemented error checking on an app wide scale so the functions can be called in separate files without being declared in the header.

We used bootstrap for UI and customized it with our own CSS.

**How To Use the Application**

Steps:

1. Browse by clicking on items as you see fit
2. Click item
3. Add to cart
4. Click cart in upper right
5. Fill in form and click checkout
6. To comment or create an item click signup or login
7. Fill out username and password with valid information
8. Click on item
9. Click add comment
10. Type comment and click submit
11. Next to your comment click edit or delete to edit or delete comment
12. To add Items to the store click signup or login
13. Fill out username and password with valid information
14. Click add item
15. Fill out the form and submit

**Obstacles**

* Implementing cross user payment routing with stripe
  + It is a complex task that requires the site to already be published
* Deploying application to AWS
  + AWS node template didn’t properly run the app through their deployment script which would require us to connect
* Connecting to AWS console from Windows
  + The openssh client for windows didn’t connect to aws due to the permission level of the private key, which aws being linux based couldn’t be modified on our local using chmod because Windows doesn’t have that command
  + We ended up connecting via mac but by that time the issue had cost too much time
* Time: we needed to cushion our tasks much more to allow for technology issues such as the above mentioned
* Education: the technologies should have been solidly laid out in the beginning to give all the members more time to acclimatize

**Risk Assessments**

During the initial planning phase of the project, and throughout its early development, several potential pitfalls were considered, and possible approaches that could mitigate them. The following is a description of these considerations, in addition to a general outline of our design philosophies:

First, the skill levels of the different group members, with regards to the varied tasks that were required (such as web development, database and API design), was quite diverse, and considerable personal study and research was required to approach these tasks. The developmental approach we decided was best suited to these circumstances was a modular approach, in which each phase of early development was attempted by all group members. This enabled a more complete understanding of each facet of the application for every group member, and still allowed for successive task load balancing between group members for later iterations.

With regards to technology, the creation of features was scheduled more on an “as-needed” basis, meaning that the most important work was completed first, and further refinement was planned for future iterations. In theory, this would allow us to find out what technologies work early enough to still have time to change strategy or technology as development progressed.

**Conclusion**

Throughout the academic careers of our group’s members, numerous development technologies have been studied and implemented, but mostly in isolated case studies, and rarely in combination with one another. Understanding how these technologies can be integrated to work together as the various components of a whole application could only be achieved through experience.

Additionally, learning to cope with the numerous potential pitfalls of software development, ranging from human resources and communications issues to aspects of the utilized technologies and beyond, is a purely empirical undertaking. For these reasons, the development of Binary Bazaar, from conception to the form it finally took, has been extremely educational and rewarding. We feel the final deliverable project has real potential utility, and could be further developed in a real world environment, to suit the needs of actual client and customer users.