COS 731

Requirements Documentation v2

Bilal Muhammad

Table of Contents

Introduction

The purpose of this document is to provide a detailed overview of the functional and architectural requirements with regards to the Bellisimo project. It will highlight architectures used as well as non-functional requirements as user experience is an important aspect of an online store.

Project Objectives

The primary objective of the project is to create a web based application whereby users can browse through listed items. A user should be able to see the image of the product as well as the price.

Users should be able to select items they wish to purchase and have it added to their cart. These items can be removed, if the user wishes not to purchase at a later stage.

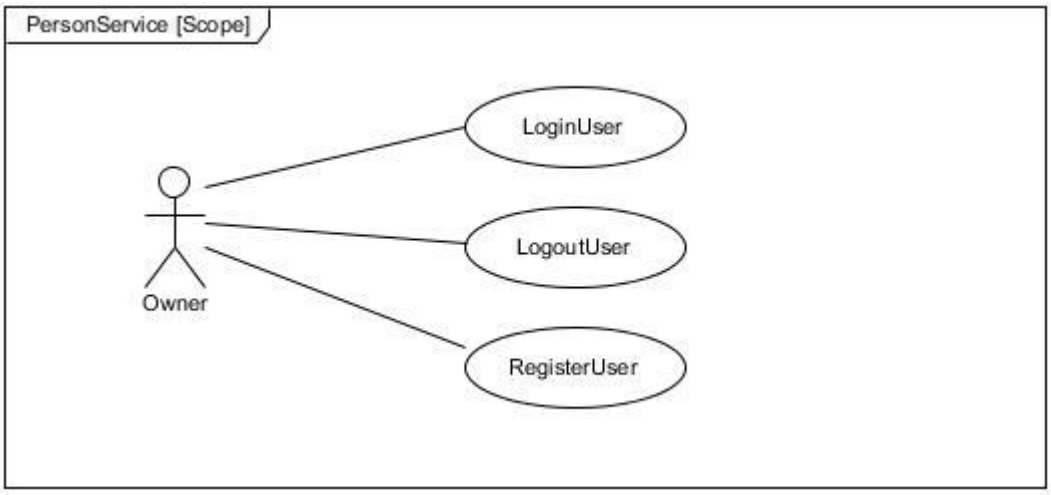
The payment system is mocked for the purpose of this project and will be handled by an external payment system which has a success rate of 95%

There should also be an administrator login which will serve the functionality to add new items as well apply specials to items.

Functional Requirements

User Management

The PersonService module will be responsible for maintaining user information. The module will also have the responsibility to add new users as well as delete them.



Use Cases

- LoginUser – Allows a user to log into the system

Preconditions

- The user has already been registered into the system
- The user has provided the correct login credentials

Post Conditions

- The user is logged into the system
- Determine whether admin or normal user
- Give access to services available on the site

- LogoutUser – Allows a user to log into the system

  Preconditions

  - The user has already logged into the system

  Post Conditions

  - The user is logged out
  - The admin can no longer change products and users cannot purchase them

- RegisterUser – Allows a user to log into the system

  Preconditions

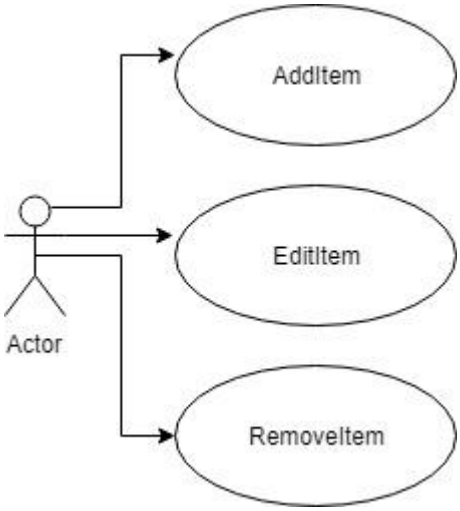
  - The user does not already exist on the system

  Post Conditions

  - The user is added to the system and may start shopping

Catalogue Management

The CatalogueService module will be responsible for maintaining Catalogues. The module will be responsible for adding and editing catalogues/items.



Use Cases

- AddItem – Allows an administrator to add an item

  Preconditions

  - The administrator is successfully logged in

  Post Conditions

  - The item is added to the catalogue for users to view

- EditItem – Allows an administrator to edit existing items

  Preconditions

  - The item needs to exist as a listed item

  Post Conditions

  - Item is updated with specials applied by the Admin user

- RemoveItem – Allows an administrator to delete an item from the Db

  Preconditions

  - The item needs to be existent

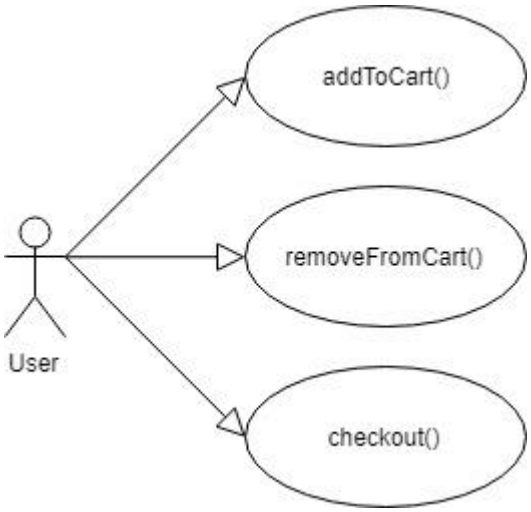
  Post Conditions

  - The item will be removed from the list of items available for shopping to the user.

Sales Management

The sales management is a combination of both users and a catalogue. Each user will have associated with it a bank balance and a history record of all items previously bought from the store. A user will be able to do 3 main functions;

- Add a new item to card
- Remove item/s from cart
- Check out when ready for payment



Use Cases

- addToCart – Allow a user to add an item to their cart

  Preconditions

  - A registered user must be logged in

  Post Conditions

  - The item should appear in the user's cart

- removeFromCart – Enables the user to delete an item from cart

  Preconditions

  - The item needs to be in the cart

  Post Conditions

  - The item should no longer be in the user's cart

- checkout – Allow a valid user to pay for the items currently contained in the cart

  Preconditions

- The item needs to be existent

Post Conditions

- Payment is either successful or fails, and users balance is then updated.

Architectural Requirements

The technologies and infrastructure have been assigned by the client and are as follows:

Front-End

1. Html5
2. Angular2
3. NodeJS Server

Back-End

1. Spring Boot
2. PostGreSQL
3. Apache Maven

The web application will consist of these two aspects mentioned above which will communicate via HTTP using the REST Framework. Hibernate is a technology that will be used to tie both ends together and will allow objects to be passed between the 2 systems.

The back-end will be hosted on Spring Boot, while the front-end will be hosted on NodeJs.

I will be making use of the monolithic implementation which consists of one backend spring boot application.

Global Quality Requirements

Performance

The system has to be available when needed, the user should not have difficulty purchasing goods as the amount of concurrent users increase.

Security

The system has to have a secure login, else people will mark down items and essentially start purchasing them for free.

At the same time general users should also be authenticated and their passwords hashed as if compromised people will be able to purchase goods on their accounts.

Reliability

The system should not glitch especially when payments are made, this will have undesirable outcomes like people losing trust in the website.

If payments are successful then delivery should be scheduled and a unique tracking number should be given so that it can be later tracked in case a technology error has occurred.

Flexibility

The system should be developed for all types of browsers as well as for smaller screens, i.e. Mobile Devices.

Maintainability

The site has to be maintainable in a way that the admin can make most of the changes.

Usability

The navigation of the site should be kept simple and should look reliable. If the impression is not good, users will not trust the site to purchase their goods.