UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

Requirements Specification for the Bellisimo System

by
Martha Mohlala *u10353403*

Github page.
`https://github.com/Potlake/Bellisimo`
September 7, 2017

Contents

1 Vision and scope

1.1 Project background

Online shopping has grown in popularity over the years. Most people shop online to save time and avoid long queues. There are people who browse through the internet to find where are specials and sales in several stores, and compare the prices amongst the stores. The platform to achieve what mentioned has not readily available to the stores around Hatfield. This becomes a problem when there are specials available and people do not have knowledge of it, and they end up buying items expensively whereas they could have save more on the specials.

1.2 Project Vision

The Bellisimo is a web application that aims at providing an online platform for customers to browse clothing as well as food catalogues provided by the businesses located in Hatfield. The catalogues include the items and their prices.The systems will provide the latest information regarding the items,sales and specials.

1.3 Architecture Design of Bellisimo

The system consists of two subystems that communicate via HTTP using REST framework, and it follows the monolithic architecture where a single unit is deployed.

1.3.1 Architectural Patterns

The systems follows the Model-View-Controller (MVC) pattern, where the view is the Bellisimo front-end, model is the data model with PostgresSQL database and Controller is Angular2 that communicates with the back-end.

1.3.2 Quality Requirements

Quality requirements include the following:

(a) Performance

(b) Availability

(c) Maintainability

(d) Scability

(e) Reliability

(f) Accessibility

1.3.3 Architectural tactics

Load balancing and connection/thread pooling address performance.
Fault detection, recovery and prevention address the availability.

2 User Module

2.1 Scope

Bellisimo's main function will be to provide an online platform for customers to browse clothing as well as food catalogues provided by the business located in Hatfield.

Guest: No information is stored for the guest user. When accessing the service, the user assumes the guest role without being logged in. The guest user may use public services and may register and login.

User: When a user is registered, the fields shown in the domain model are stored for the user, using a unique automatically assigned ID. The user provides all other fields. The password should be stored in encrypted format. The user may change the value of any of the fields of his/her own record. The difference between a user and an admin is the value of the isAdmin field. Only Admin users may add,remove and update items.

User use case diagram

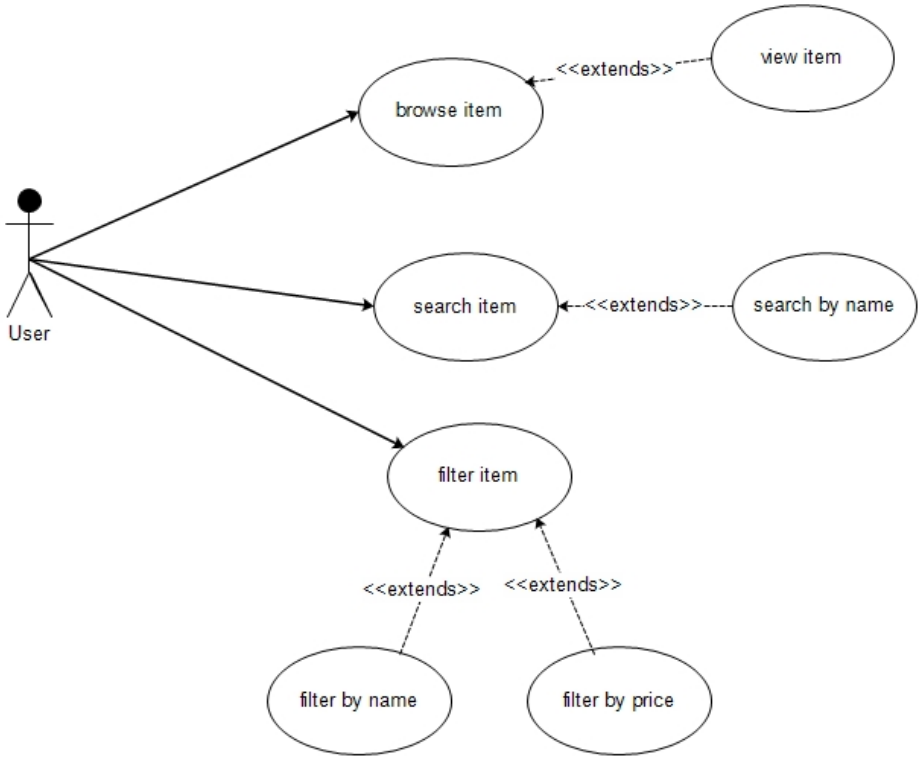

Figure 1: User use case

User Class Diagram

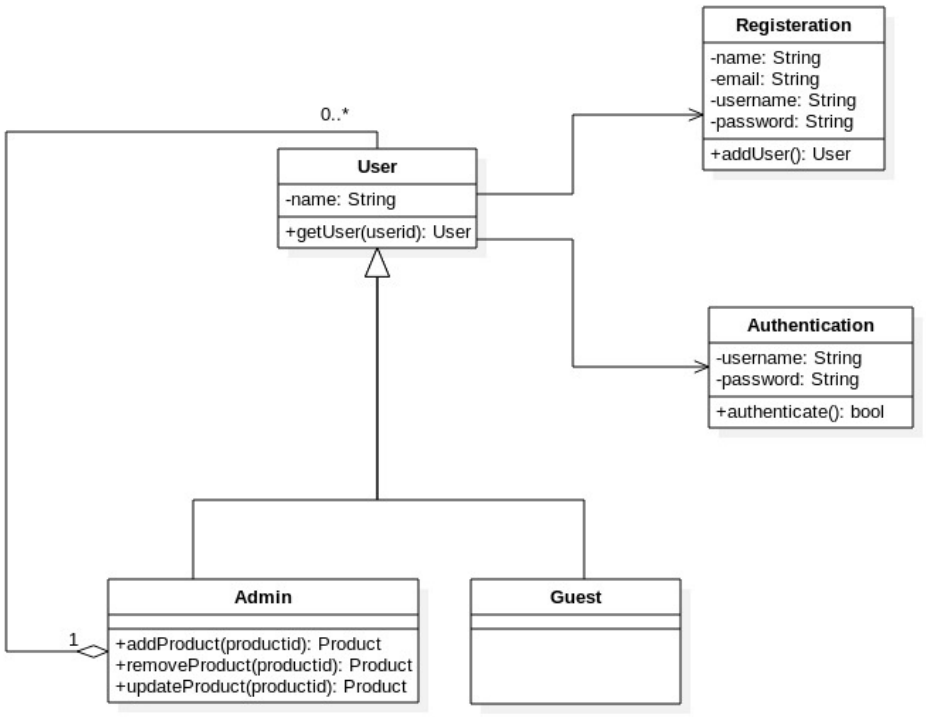

Figure 2: User Class Diagram

The user can wish to register to receive emails about available special. Figure 3 depicts the process of registration.

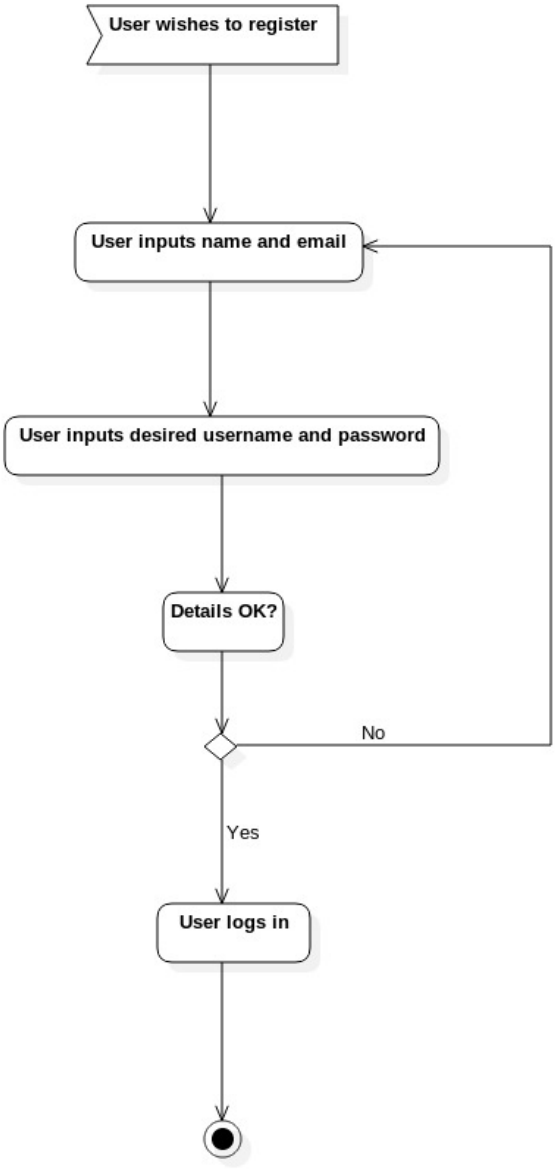

Figure 3: Registration Process

3 Admin Module

3.1 Scope

The admin's main function is to add, delete and update the products. That extends to adding specials. Specials can be added as singular or as a group.

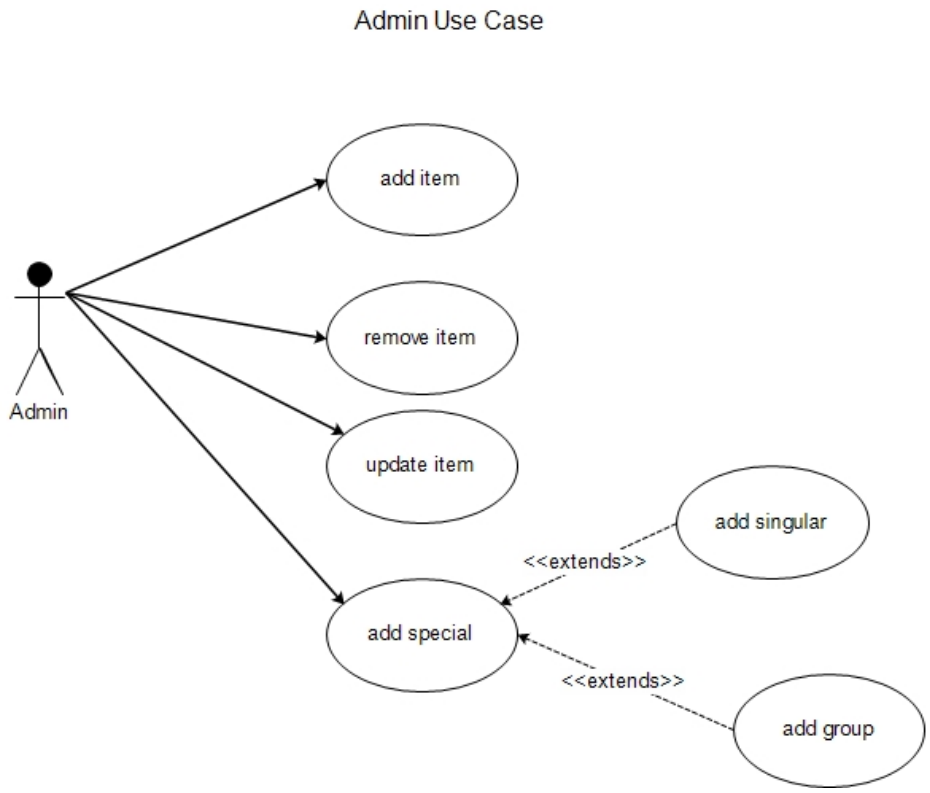

Figure 4: Admin Use case

The below figure shows the activities perform by the admin.

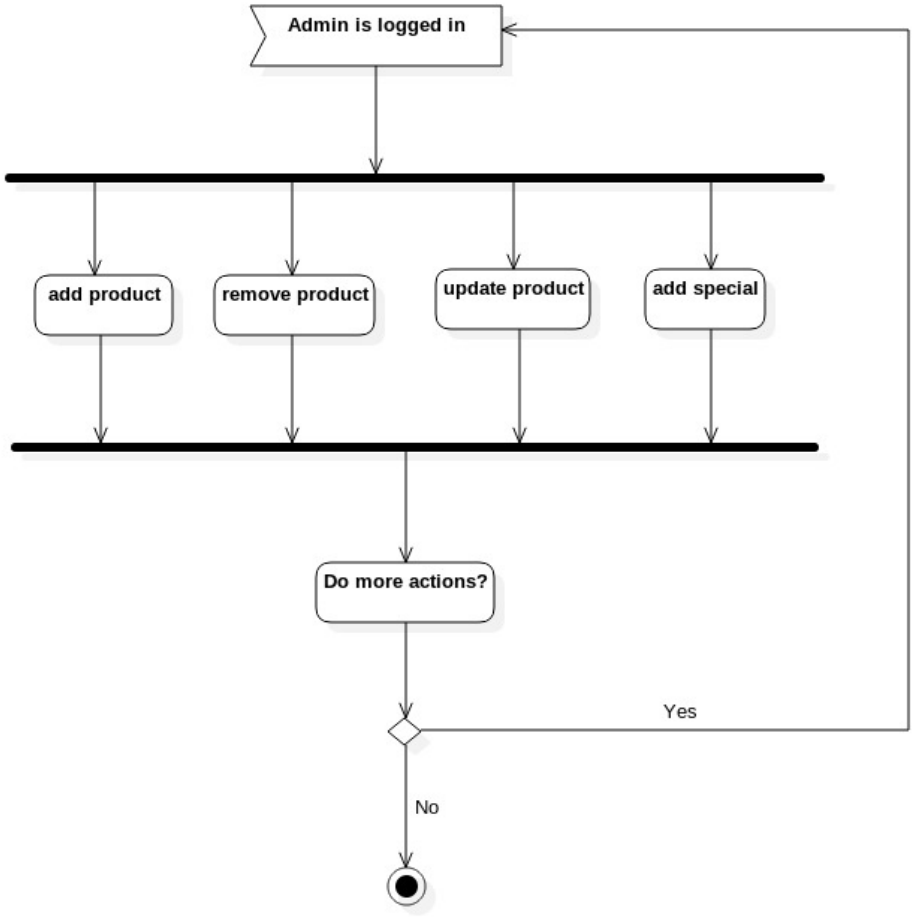

Figure 5: Admin activities

4 Catalogue Module

4.1 Scope

Catelogues include food items and clothing items from different stores.

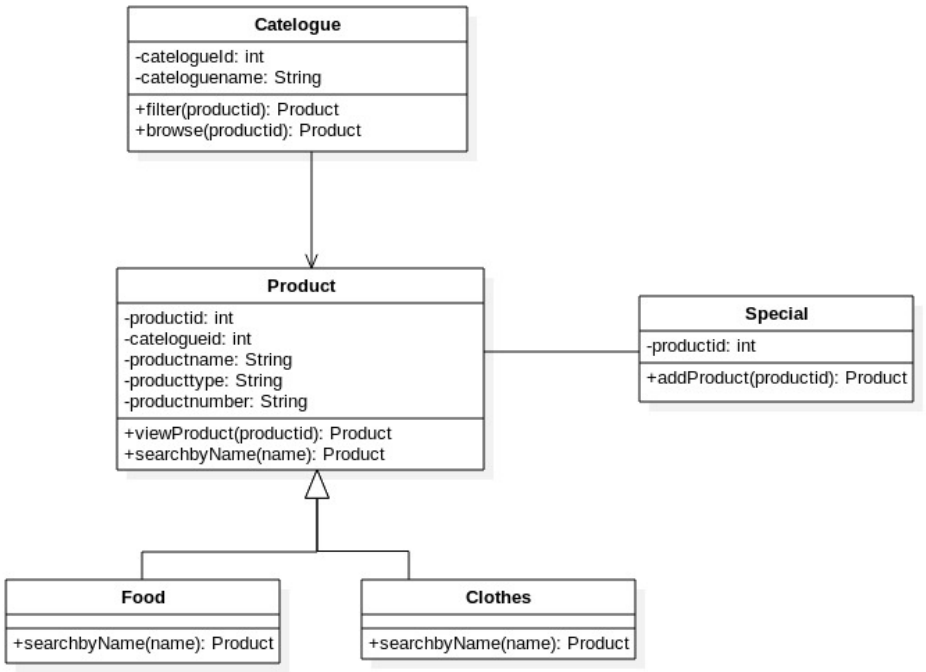

Figure 6: Catelogue Class diagram

5 Technologies

Html and Bootstrap: it is used to design the front-end interface.
Angular2: for testing web application.
Nodejs:allows Angular2 and Javascript to run on the server.
Spring boot: implements the back-end of the system to connect to the data model.
PostgresSQL: it used to design and implement database.
Apache Maven: manages the system's build.