UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

Bellisimo
Requirements Specification

10 July 2017

Contents

1 Project Background

Bellisimo is a company aimed at providing an online platform for customers to browse clothing as well as food catalogues provided by the business located in Hatfield. Information about specials and promotions will be published on the online platform.

2 Vision and Scope

The core of the system will be catalogues of items and their prices. Since Bellisimo is involved in clothing and food, the catalogues will have to ensure that these lines are well maintained. Sales and specials in each line will have to be accounted for and managed. The scope of the system is to ensure that the latest information is being provided about items and their prices.

3 Design Specifications

The system should be designed in such a way that there are two different interfaces that depict the clothing catalogue and the food catalogue. The following design specifications are what the system should adhere to:

- There should be at least two interfaces. An admin interface to maintain the catalogues and an anonymous user interface that will allow people to interactively view the catalogue.
- The prices of each item in the database should be displayed along with the image of the item.
- The administrator should be able to add, remove and update any items in the catalogue list.
- The administrator should be able to add specials to the module. Specials can be applied to singular items or groups of items depending on a special group configuration.
- Users should be able to browse, search and filter the catalogues.

4 Design Technologies

The following technologies will be used to implement the system:

- Html 5 (Html and Bootstrap CSS)
- Angular2
- NodeJS
- Spring Boot
- Spring Cloud (Microservices)
- Netflix OSS (Microservices)
- PostgresSQL
- Apache Maven
- Git (Github)

5 Architectural Technologies

The web application will consist of two subsystems that communicate via HTTP using REST Framework. The Java/Spring Boot application will be known as the "backend" application. The HTML5/Angular2 application will be known as the "frontend" application. The backend application is expected to communicate with the database and use Hibernate which can be imported into Maven, a dependency management tool whereas the frontend application will be hosted in the browser and NodeJS is expected to manage packages required for the application to run successfully.

There are 2 choices on how to implement the backend application. The first choice is to use a monolithic architecture and the second one is to use a microservices architecture.

5.1 Monolithic Implementation

The monolithic implementation will consist of one backend **spring boot** application that will be deployed as a single unit. Apache Maven will be used as the dependency management tool.

5.2 Microservices Implementation

The microservices implementation will consist on n spring boot applications where n = no. of modules or components in the system. These applications will be integrated by using spring cloud and Netflix OSS Zuul Integration. The integration server will be regarded as one component of the n components to be implemented.

5.3 Front-end Application

The frontend application will be an Angular2 application that will communicate with the backend application via the HttpModule frovided by angular2. There are a number of generators that one can use to create an angular2 skeleton application. Links listed in the *Further Reading* section.

6 Further Reading

Below is a list of links that you can read up on to familiarise yourself with the relevant technologies. Take special note of the links related to the monolithic and microservices implementations.

- **Spring Boot** - `https://spring.io/guides/gs/spring-boot/`
- **Spring Cloud** - `https://piotrminkowski.wordpress.com/2017/02/05/part-1-creating-microservice-using-s`
- `http://www.baeldung.com/spring-rest-with-zuul-proxy`
- **Angular2** - `http://www.angular2.com/`
- `https://angular.io/tutorial`
- **Maven** - `https://maven.apache.org/guides/getting-started/maven-in-five-minutes.html`

  Additions to the user interface may be added if desired but these additions must serve a purpose.