Munchies

An app to simplify the food ordering process in small and medium size companies

***S**oftware **R**equirement **S**pecification*

Application Summary

Food is an important and integral part of everyday life at any company. A lot of time and effort is spent when:

a) employees decide on the restaurant
b) make the order to the chosen restaurant

This application tries to alleviate these problems by offering employees a centralized repository of restaurants along with contact information and menus. Furthermore it allows employees to create group orders that can easily be relaid to restaurants. It does this by having two distinct parts:

1. **Administration portal**
   Used for: adding restaurants, global configuration
2. **Employee portal**
   Used for: viewing restaurant information, creating a group order, submitting individual orders to the group order

Roles

There is a total of 2 roles:

1. **Administrator**
   Authenticated by an email and password. Only this role can access the administration portal.
2. **Employee (anonymous)**
   This role does not have any authentication requirements. All unauthenticated requests are considered to belong to this role. This role can access the employee portal.

Functional Requirements

Administration portal

Following is a list of pages that should be implemented as part of this module.

Page Name	***Description***
Log-in	Simple login form with email and password fields.
Add restaurant	A form which allows administrators to add a new restaurant to the system. Input fields: restaurant name, address, phone number, menu URL, delivery information (delivery time, additional charges, etc.).
View all restaurants	A table with all restaurants in the system. Following columns should be present: restaurant name, restaurant short name (generated by replacing spaces with underscores), address, phone number, menu link.
View restaurant details	Page which allows the administrator to view all restaurant details. This page has all fields associated with the restaurant. Furthermore it allows the administrator to delete and edit the restaurant. In case the delete action has been selected ask the user has to confirm his choice through a confirmation dialog.
Edit restaurant	A form which allows the administrator to edit restaurant information. Structure of the edit form should be the same as in the *Add restaurant* page.

Employee portal

Following is a list of pages that should be implemented as part of this module.

Page Name	***Description***
View all restaurants	A table with all restaurants in the system. Following columns should be present: restaurant name, restaurant short name (generated by replacing spaces with underscores), address, phone number, menu URL. Furthermore, the last column should contain a link to the *New group order* page for this restaurant.
View restaurant details	Page which allows the employee to view all restaurant details. This page has all fields associated with the restaurant.
New group order	This page allows the employee to start a new group order. Employees must enter their name and the order timeout. Clicking the create button should create the order, and redirect to the *Group order* page.
Group order	This page is displayed when a new group order has been created for a specific restaurant. It consists of three parts: 1. Order information: order URL, order timeout (always 10 mins since the order start), order creator, restaurant name, restaurant phone no., menu URL 2. Add selection form which is used by other employees to add their selection to the order. This form has fields for: employee name, item name, and price. 3. Table of all selections for the current order. This table contains employee name, item name, and price and is automatically refreshed every 2 seconds. A total of all orders is displayed at the end of the table.

Non-functional Requirements

Following is a list of non-functional requirements that should be satisfied:

- Use Spring via Spring Boot
- Use MySQL as the RDMBS
- Use migrations (Flyway and Liquibase play nicely with Spring)
- Use Hibernate as an ORM
- Use thymeleaf for server-side templating