Online Restaurant System
Software Requirements Specification
For Web Application

Version <1.0>

Revision History

Date	Version	Description	Author
16/OCT/20	1.0	First phrase software requirements specification	Abir Deb, Samuel Fils, Andrey Goryuk, Michal Moryosef, Syed Sadman

Table of Contents

Software Requirements Specification

1. Introduction

1.1 Purpose

This software requirement specification will provide a detailed description of the Online Restaurant System (ORS). In this document, the user case models, system requirements and specifications are detailed.

1.2 Scope

The Food Ordering System (FOS) will be a web based application where users can order a variety of food items.

This ORS, includes two use-case diagrams: the first use-case diagram describes the customer functionalities (order, deposit money etc.) The second use-case describes the employees class: chef and manager. Each employee has its own separate specifications according to the task description provided (see *References*). In this document includes the software requirements such as frameworks which will be used, as well as relevant dependencies used.

1.3 Definitions, Acronyms, and Abbreviations

Terms	Definitions
ORS	Online Restaurant System
FOS	Food Ordering System
ReactJS	React is a front end JavaScript library for building user interfaces maintained by Facebook.
Java	Java is a class-based, object-oriented programming language.
Spring Boot	Spring Boot is an open source Java-based framework used to create a micro service.
SQL	Database querying language/management system
NodeJS	Javascript runtime environment that runs outside the web browser

1.4 References

Software Requirements Specification, www-cs.ccny.cuny.edu/~csjie/322/spec_sample.pdf.

1.5 Overview

The "Overall Description" section of the document illustrates an overview of the system and its relation to users. This section is intended to be a top down view of the system as a whole through use-case diagrams which link different types of users to features in the ORS.

The "Specific Requirements" section details how the different software systems beneath the GUI layer in the ORS interact with each other. This section is aimed at developers to understand the flow of data and various connections between the frontend and the backend of the ORS.

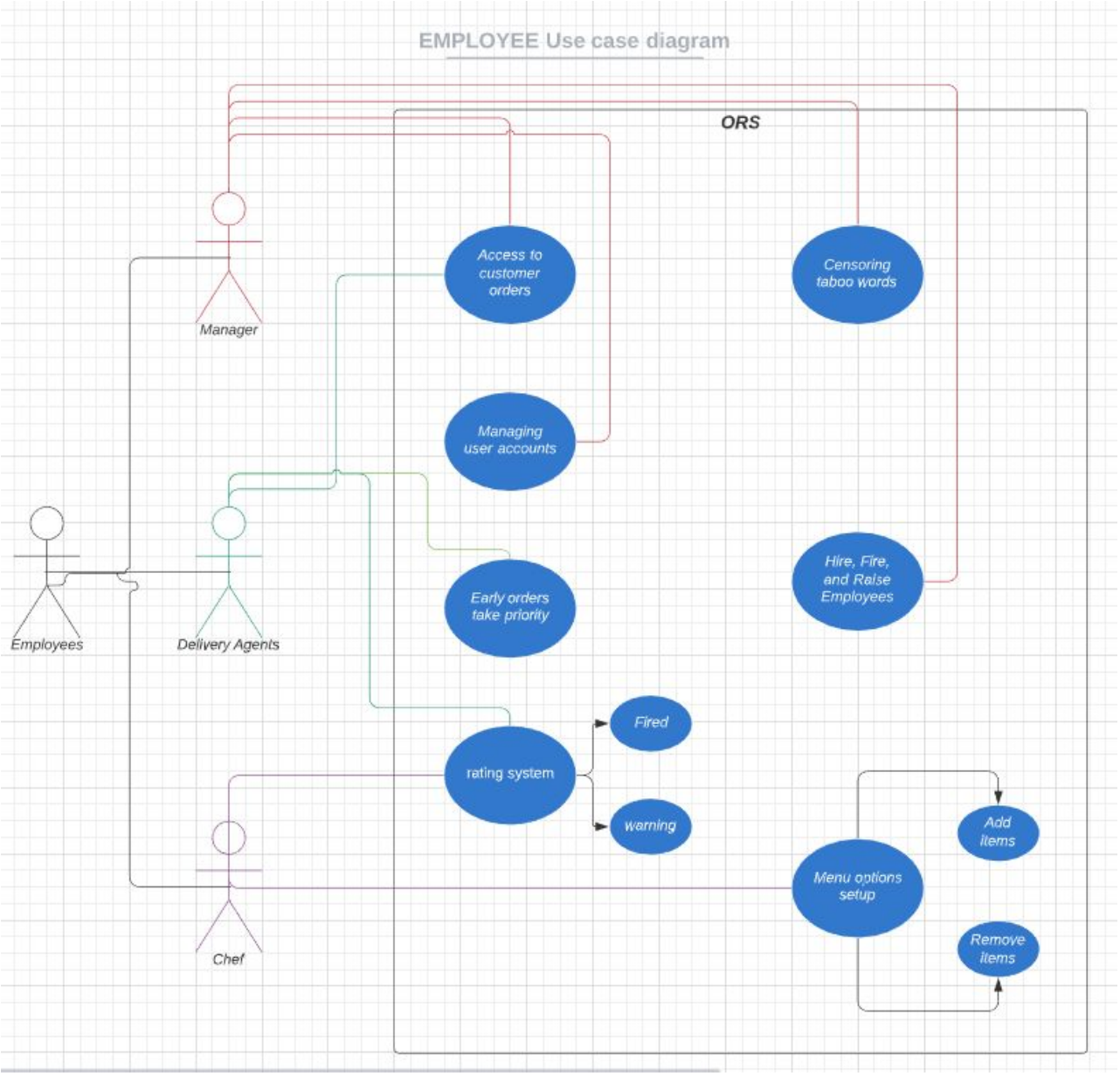
2. Overall Description

The ORS features an online food ordering service that is accessible by different types of users. It features a GUI web based application that provides participants with an interface to create and manage order while incorporating a delivery system that drives competition between food delivery agents. The delivery agents can pick the orders they want to deliver from the queue, that is set up by the recency of the order. It also features a dashboard for chefs to be able to present their menu items, providing various food options for customers, setting the prices. These menu items can also be rated from 1 to 5 stars. Managers have various jobs such as approving every account manually including delivery, chef and customer accounts, manually reviewing banned user's unban requests. Managers can also choose to fire each of his employees or give them a warning based on reviews by the customers. Customers with 50 orders or 500 dollars spent also can acquire the VIP status that leads to their rating counting as twice the normal rating and having access to 10% discount on the menu items.

2.1 Use-Case Model Survey

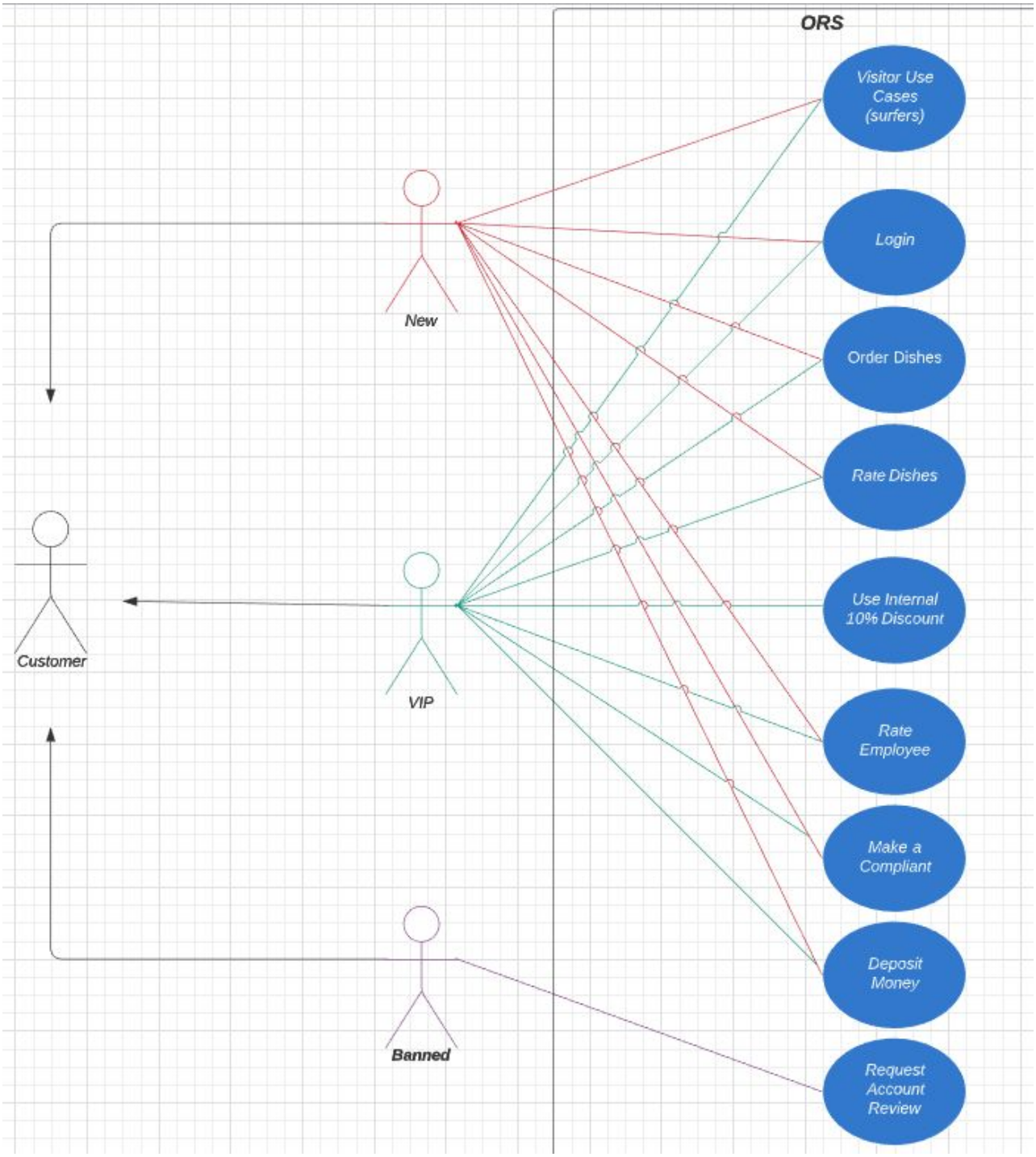
There are four different types of users: Customers, Delivery Agents, Chefs and the Manager/Admin. The use case model for the employees is displayed below. This includes the Manager, Delivery Agent and Chefs.

https://lucid.app/invitations/accept/f050bb87-9f77-4807-9ee0-611916b89b10 (For better view)



The customers have three roles such as Surfers, registered users and VIPs.

https://lucid.app/lucidchart/f3500199-962f-44a2-9ab0-aad621bad561/edit?shared=true&page=.Q4MUjXso07N#?folder_id=home&browser=icon



2.2 Assumptions and Dependencies

Assumptions

1. Customers pay in USD
2. A bank account is required to make transactions. Other forms of payment are currently not supported
3. Users are required to have an internet connection to view and make orders

Dependencies

4. ORS hosted on Tomcat servlet
5. Spring Maven dependencies
6. H2 Database
7. Spring MVC

3 Specific Requirements

This section contains a detailed overview of user permissions with case diagrams.

3.1 Use-Case Reports

[Unregistered (Surfer) Customer Use Case]
Use-Case: Browse menu items
Description: Surfer will be able to access the menu page on the website and look through available dishes.

[Registered Customer Use Case]

Use-Case: Login
Description: Customer must provide a username and password for authentication.

Use-Case: Order Dishes
Description: Customers can order dishes from the menu.

Use-Case: Rate Dishes
Description: Customers can rate dishes that they ordered before through their order history page.

Use-Case: View Order History
Description: Customers will have a profile page where they can see a list of their previous orders.

[VIP Customer Use Case]

Use-Case: Use 10% Internal Discount
Description: VIP customers will be access to a 10% Discount on all orders.

Use-Case: Rate Dishes
Description: Customers can rate employees that are associated with their orders.

Use-Case: Make a Complaint
Description: Customers can make a complaint to the manager.

Use-Case: Deposit Money
Description: Customers can deposit money to account balance

[Banned Customer Use Case]

Use-Case: Request Account Review
Description: Banned customers can contact the manager for account review. They are restricted from accessing the above use cases.

[Chef Use Case]

Use-case: Add Menu Item
Description: Chef is able to add an item to the menu. Chef must include a name, brief description, and price. **Optional: Chef can include a picture of the item*

Use-case: Remove Menu Item
Description: Chef can remove items from the menu.

[Manager Use Case]

Use-case: Access Customer Orders
Description: Manager has a tab to see a list of all orders placed

Use-case: Managing User accounts
Description: Managers can remove banned users from the system or change them back to registered customers

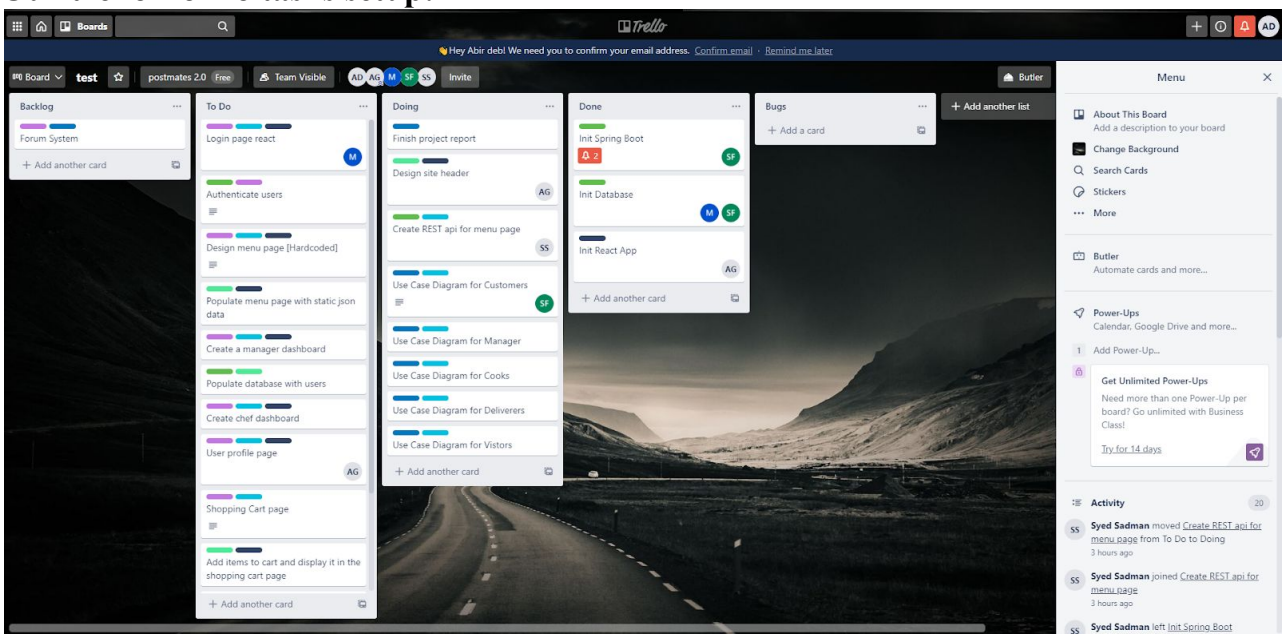
Use-case: Managing Employee Accounts
Description: Managers can remove/fire employee accounts (Chef or Deliverer)

3.2 Supplementary Requirements

Frontend Technologies: ReactJS, HTML, CSS
Backend Technologies: Spring Boot, NodeJS,
Databases: SQL, H2 Database Engine
Hosting: Heroku, GitHub pages

4. Supporting Information

Our trello To-Do tasks setup:



Wireframe of the main menu page (under progress):