**210705 — Rajesh U — Java Batch — TRNG-00001122**

**Restaurant Management System / contact-less dining**

## Abstract:

Restaurant Management System is a system to automate the day-to-day tasks in a restaurant. The main scope of this system is to improve the performance of the restaurant by eradicating the paperwork.

## Users/ Actors:

1. Customers - interact with the system to place / modify order, get receipt and give feedback.
2. Kitchen Manager / Head Chef - interacts with the system to approves the order, marks as ‘Prepared’ when dish is prepared, and marks “Served” when the dish is served to the customer.
3. Supervisor - interacts with the system to check the table status (Empty or filled). He receives notification when the customer asks for help through the system and if the customer requests the paper bill.
4. Admin - interact with the system to manage the chefs and menu items.

## Customer Dashboard:

In the customer dashboard, the system allows the customer to place orders, edit/cancel order, request help, request bill.

**1.Place order:**

In the place order menu, the system shows the list of dishes available. Each dish will have an image, time, and its price.

Depending on the time, the system will show the corresponding dishes automatically. (For eg: Breakfast dishes must be available from 8AM to 11AM, Lunch dishes must be available from 11AM to 3:30PM)

**2. Edit / Cancel Order:**

After the order is placed, the system allows the customer can edit or cancel the placed order. This option will be available only after the order is placed.

**3. Timer:**

After the order is placed, the system automatically calculates the time for the dishes to be served.

**3.1 Timer Rules:**

**The timer of an order** is the sum of time taken for each order to cook.

If a customer orders the same dish more than once, the time will be calculated as n+2 minutes for each item (Where n is the time taken for each item).

**4. Help:**

The system allows the customer to request help when he/she has need water or any queries regarding the menu/dishes.

**5.Request Bill:**

The system allows the customer to request the bill any time after the order is placed. The system displays the consolidated receipt/bill. The customer can pay through the system or request help to get printed bills or pay through cash. The bill will be sent as a link to mail or text message.

6**.Feedback:**

The system allows the customer to rate his/her experience in the restaurant through a form (Multiple choice questions). The feature will be automatically displayed once when the service is completed / closed.

The feedback is optional. So, the customer has options to skip.

## Kitchen Manager / Head Chef Dashboard:

In the kitchen Manager dashboard,

* The system notifies the kitchen manager once the order is placed by a customer.
* The system provides all the current order details (Order number, table number, dishes to be served etc.).
* The system allows the kitchen manager to mark the order has complete when the dishes are served to the customers.
* The system allows the kitchen manager to cancel a dish/order, when a customer request (Kitchen manager can reject the request if the dish has been prepared).
* The system allows the kitchen manager to mark an item/dish from the menu as “Unavailable,” if the dish is unavailable or cannot be served.

## Supervisor Dashboard:

In the hotel supervisor dashboard,

* The system allows the supervisor to view all the order details (Order number, table number, dishes to be served etc).
* The system will notify you when an order is completed.
* The system allows the supervisor to view the unoccupied tables
* The system will notify the supervisor if the customer requests help from the application.
* The system allows the supervisor to generate a bill / receipt for a table / customer and mark as “Paid” if the customer mode of transaction was cash.

## Admin Dashboard:

In the Admin Dashboard, admin has complete control all the stakeholders (Customers, Hall & kitchen Manager).

* The system allows the admin to create user credentials for hall and kitchen manager.
* The system allows the admin to add, modify and delete dishes in the inventory based on the timing (Breakfast, afternoon or dinner)
* The system allows the admin to view all the order details of a customer.

## Application Journey:

* Customer has to enter his details (Name, phone number and mail id) in order to use the application.
* Once the credentials are entered, the system allows the customer to scroll through the menu and select the dishes.
* Once the order is confirmed by the customer, the kitchen manager/hotel manager will be notified about the order details.
* The kitchen manager allocates the dishes to his chefs based on the order. The kitchen Manager serves the order based on the order queue.
* The kitchen manager will mark the order as “Complete”, when the the order has been served to the customer.
* The hall manager will also be notified once the order has been served to the customer.
* Customers can close the order by requesting bill. The bill/ receipt will be sent to your mail id and mobile. The payment can be made through the mobile or through the application using UPI, net banking or cash (by requesting help from the hall manager)
* The hall manager will be notified when the order is completed / closed. If the customer pays through cash, the hall manager has to manually mark the order as complete.

**Technical Design/Architecture needs to be derived for the following - Before the start of the Development**

* Database Schema/Table Design
* Presentation Layer - meaning UI Screens - for all the requirements stated above - none should be modified in the backend
* Business/service Layer - based on the requirements create the functionality and have them in the middle layer
* Database Layer - to access the DB data

**Validations**

* All front-end validations need to be done
* All business validations need to be properly reflected to the users with proper messages to change the data if needed
* All database validations need to be properly reflected to the users with proper messages to change the data if needed

**Logs**

* To be maintained like which user using which functionality
* App logs for each layer need to be differentiated

**Exclusions or Implicit Assumptions**

* Payment gateway or payments are assumed that all goes well without invoking

**Implicit - Testing**

For all the development implicit usage of Unit testing code should be done.

**Devops - Tools should be used**

Github, Maven, Jenkins must be used to build and deploy

**Notifications (optional)**

Sent through Emails and SMS

**Output Expected**

* Technology Stack going to be used
* Architecture
* Architecture Flow
* DB Design
* Wireframe
* Technical requirements – LLD – Low Level Design Document
* Test Cases
* Code
* RTM – Requirements Traceability Matrix