**Restaurant Management System**

*In partial fulfillment*

*for the requirements of:*

CSCC 20: Object-Oriented Programming

Submitted by:

Julius Czar Makiling

Sidnel B. Gramata

**Table of Contents**

I. Introduction

II. Requirements Analysis

III. Use-Case Analysis

IV. Class Diagram

V. Sequence Diagram

VI. API

**I. Introduction**

Sovereign Restaurant Management System is an inventory focused system for kitchen related management. The system is divided into different areas; the Recipes, Orders, Ingredients, and Requests.

The chef creates/removes different recipes. Each recipe has names, ingredients, unique recipe number, and a main ingredient. Each ingredient has a unique ingredient number, name, recipe number, and date acquired.

The customers have orders. Each order stores a unique order number, recipe number, date ordered, and the number of servings that they ordered.

The suppliers can request for new ingredients in case a new recipe is made, or if an ingredient is low on stock. Each request form contains a unique request number, ingredient name to be requested, amount of ingredients, recipe number that it will be used in, and date acquired.

Data related to the inventory system of the restaurant is managed by the head chef and the assistant. An external factor is also present through the supplier which provides the ingredient to the restaurant whenever it is needed. The assistant is responsible for the service side of the restaurant like taking orders or utilization of the ingredients. While the head chef takes care of the management side, including creating and removing of recipes and requesting ingredients from the supplier. Each actor has their own corresponding responsibility, though the head chef can access the systems inventory, the assistant on the other hand can only access the ingredient aspect. As well as the supplier not having access to the restaurants system but can affect the inventory through acquisition of ingredients.

**II. Requirements Analysis**

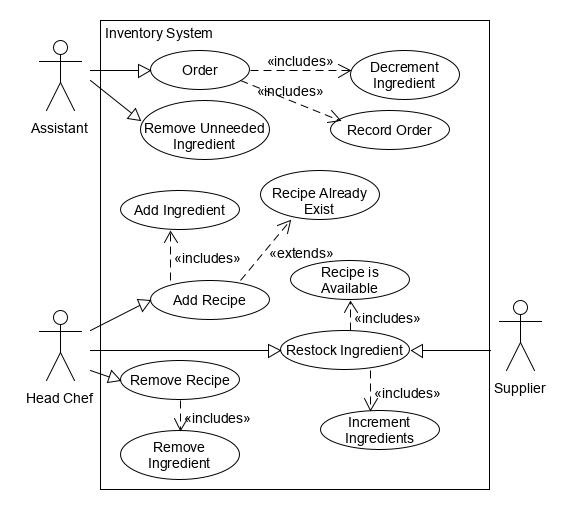
1. Display an updated menu that shows what recipes are available
2. Display all recipes in the system (including those recipes that lack ingredients).
3. Show which recipes are available depending on the remaining stocked ingredients
4. Keep track of stocked ingredients left in the system
5. Update system ingredients whenever restocked or when orders are made
6. Add/remove recipes
7. Record recent orders

* Orders
  + Orders made/being made
  + Record orders made
* Recipes
  + Create new recipe
  + Remove recipe
* Ingredients
  + Add new ingredients
  + Restock ingredients
  + Remove ingredients
* Menu
  + Show recipes available
  + Show all recipes in system

Non-Functional Requirements:

* Accessibility
* Portability
* Data Integrity
* Efficiency

**III. Use-Case Analysis**



**Use-Case 1**

Title: Add Recipe

Actor: Head Chef

Success Scenario: 1. Chef names recipe

2. Chef enters ingredients needed, and proportions

3. If: recipe needs ingredients that aren’t in the system, system must add the new ingredients

Pre-Condition: User logs in as Head Chef.

Post-Condition: A new recipe is added into the system.

Extension:

* + - If recipe already exists, an error message is displayed.

**Use-Case 2**

Title: Remove Recipe

Actor: Head Chef

Success Scenario: 1. System shows current list of recipes.

2. Chef chooses which recipe should be removed.

Pre-Condition: User logs in as Head Chef

Post-Condition: Recipe within system is removed completely.

Extension:

* If chef changes his mind, system goes back.

**Use-Case 3**

Title: Restock Ingredients

Actor: Supplier

Success Scenario: 1. System shows current list of ingredients and the amount left.

2. Supplier chooses which ingredient to be restocked.

3. Supplier indicates amount to be restocked.

4. System indicates if supplier wants to restock another ingredient.

Pre-Condition: The ingredient must be present in the system.

Post-Condition: The transactions is recorded in the system, and ingredients are incremented.

Extension:

**Use-Case 4** //Main scenario

Title: Order

Actor: Assistant

Success Scenario: 1. The system shows the available recipes

2. The Assistant chooses the recipe

3. System asks for number of servings to be made.

4. The system decrements the required ingredients according to order and serving count.

5. The order information is recorded by the system.

Pre-Condition: The ingredients for the recipe is available.

Post-Condition: The number of ingredients is decreased

The order is recorded in the system.

Extension:

* If indicated serving to be made exceeds the available count, error message is displayed.

**Use-Case 5** //Main scenario

Title: Inventory Report

Actor: Assistant

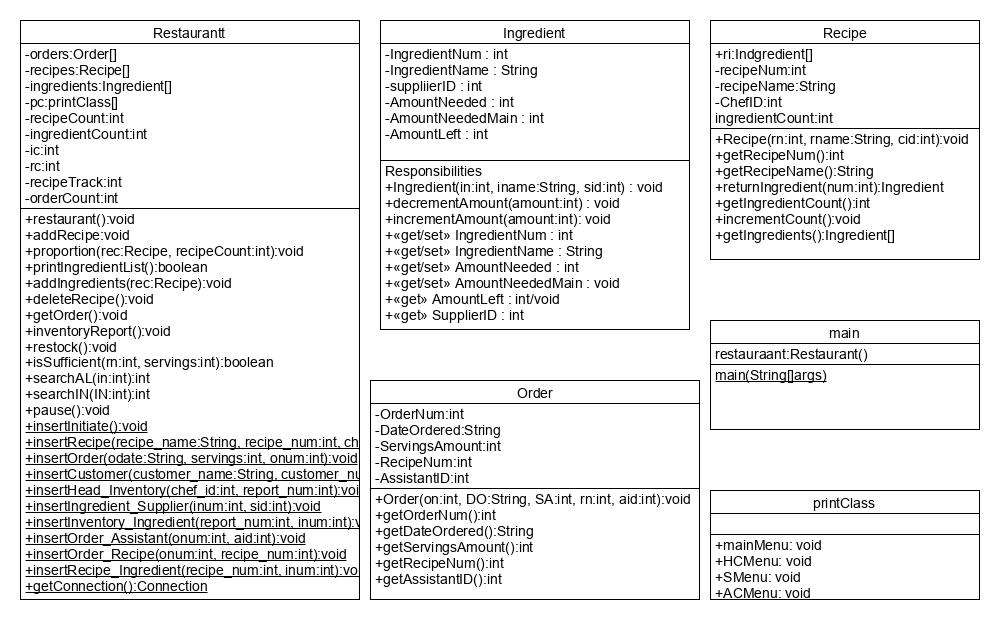
Success Scenario: 1. The system shows the available recipes

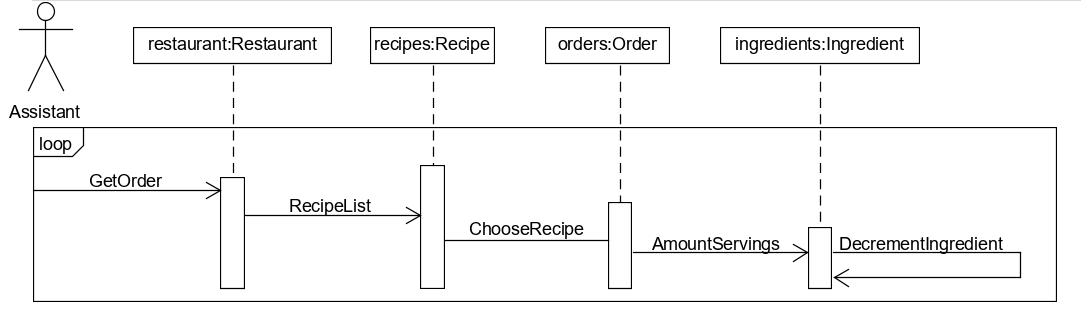
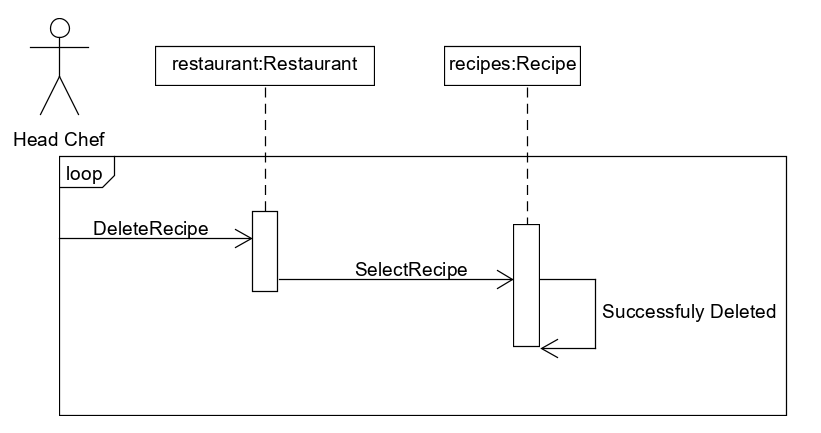
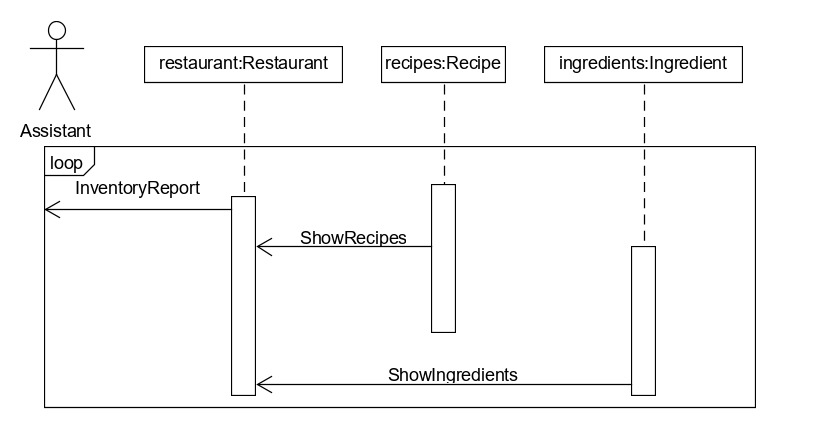
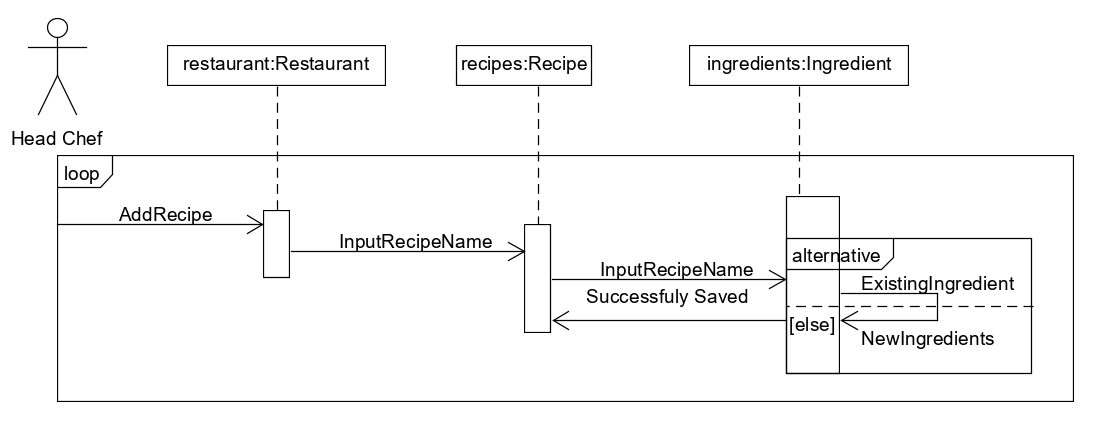
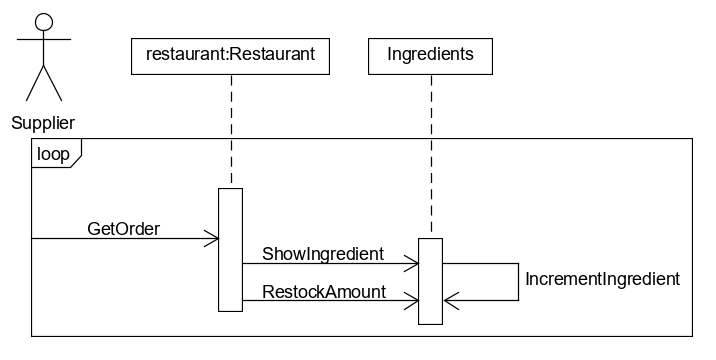
2. The system shows the present ingredients

Pre-Condition:

Post-Condition:

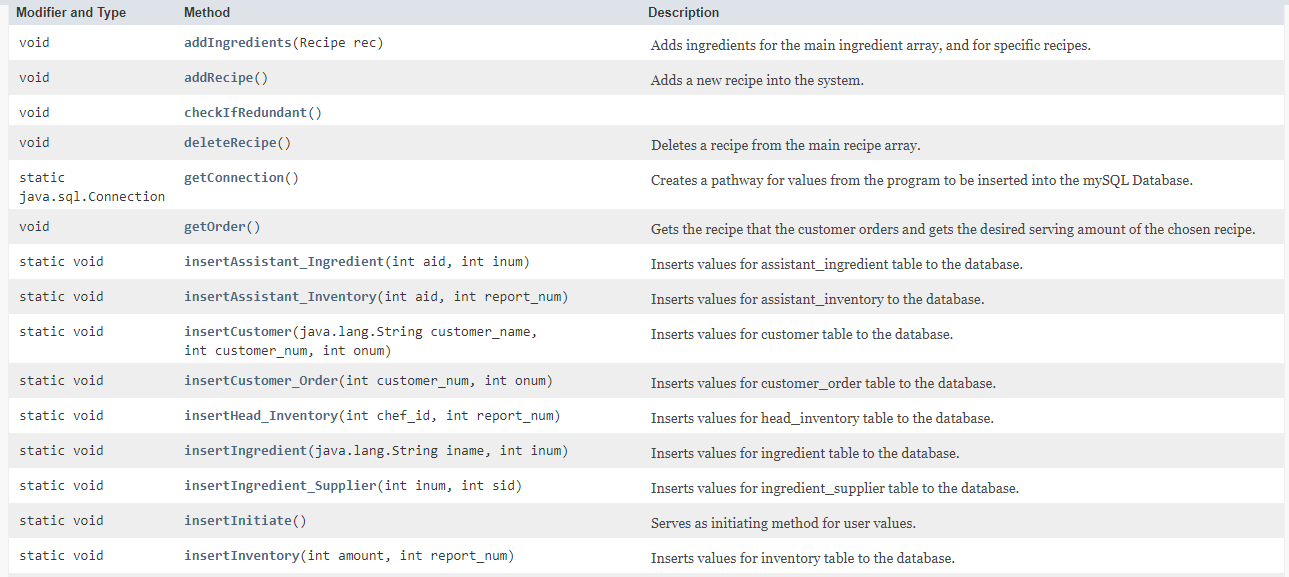
**IV. Class Diagrams**

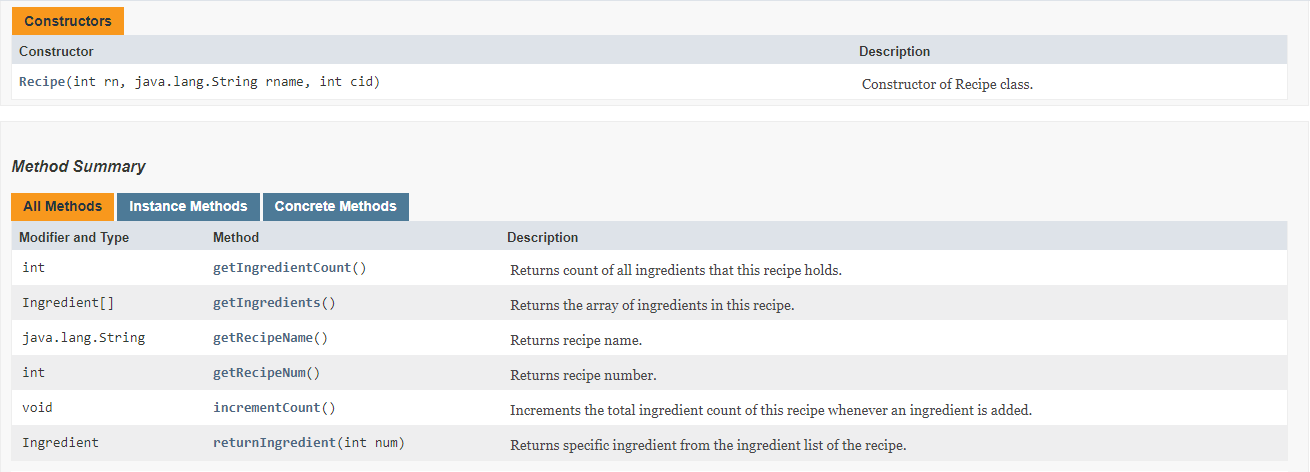
****

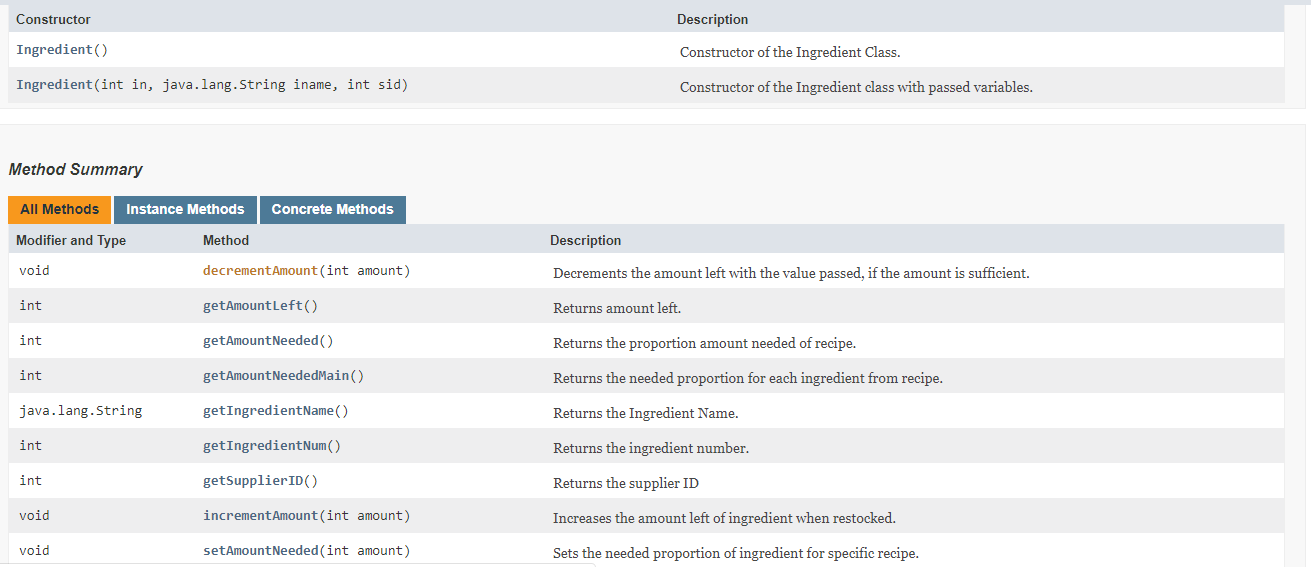
**V. Sequence Diagrams**

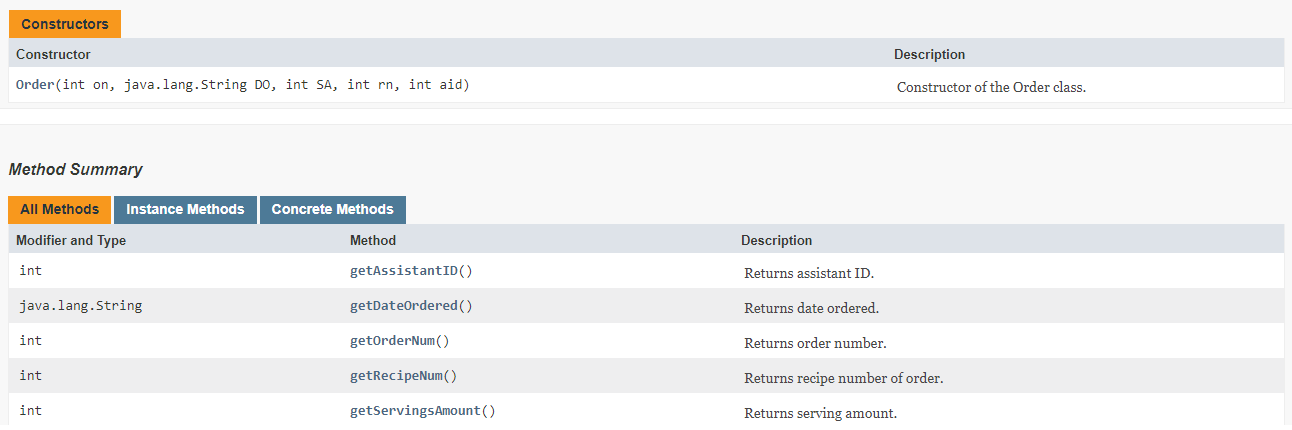
**VI. API**

Restaurant API

****

****Recipe API

Ingredient API

Order API****