Chef’s Adventure

System Specification

CSC 3150

Andy Cameron

06/06/2024

Tsion Ballo

[ballot@spu.edu](mailto:ballot@spu.edu)

A screenshot of a phone

Description automatically generated

# **Table of Contents**

Table of Contents

[Table of Contents 2](#_Toc168603565)

[1. Executive Summary 3](#_Toc168603566)

[2. Introduction 3](#_Toc168603567)

[2.1. Problem Statement / Project Vision 3](#_Toc168603568)

[2.2. System Capabilities 3](#_Toc168603569)

[2.3. Non-functional Requirements and Design Constraints 3](#_Toc168603570)

[2.4. System Evolution 4](#_Toc168603571)

[2.5. Document Outline 4](#_Toc168603572)

[3. Structural Model 4](#_Toc168603573)

[3.1. Model Introduction 4](#_Toc168603574)

[3.2. Class Diagrams 5](#_Toc168603575)

[3.3. Metadata 5](#_Toc168603576)

[4. Architecture Design 12](#_Toc168603577)

[4.1. Architecture Overview 12](#_Toc168603578)

[4.2. Infrastructure Model 12](#_Toc168603579)

[**4.2.1.** **Deployment Diagram 1 – Architecture Overview** 12](#_Toc168603580)

[**4.2.2.** **Deployment Diagram 2 – Nodes and Artifacts** 13](#_Toc168603581)

[4.3. Hardware and Software Requirements 15](#_Toc168603582)

[**4.3.1.** **Hardware Components** 15](#_Toc168603583)

[**4.3.2.** **Required Software Components** 15](#_Toc168603584)

[4.4. Security Plan 15](#_Toc168603585)

[**4.4.1.** **Security Overview** 15](#_Toc168603586)

[**4.4.2.** **Security Plan** 15](#_Toc168603587)

[5. User-Interface 17](#_Toc168603588)

[5.1. User-Interface Requirements and Constraints 17](#_Toc168603589)

[5.2. Window/Screen Navigation Diagram 17](#_Toc168603590)

[5.3. UI Wireframes 18](#_Toc168603591)

[5.4. Reports: "Formal Output" Design 20](#_Toc168603592)

[6. Appendices 20](#_Toc168603593)

[6.1. Glossary 20](#_Toc168603594)

[6.2. References / Bibliography 21](#_Toc168603595)

[6.3. Supporting documentation 21](#_Toc168603596)

# **Executive Summary**

This document is the System Specification for the app called Chef’s Adventure. The client is someone who enjoys cooking and trying out new recipes. This project aims to save recipes people want to try and the ingredients that go with them, as well as the steps on how to make the meal, making it easier to try cooking new meals. This document covers the Introduction, Structural Model, Architecture Design, User Interface, and Appendices.

# **Introduction**

## **Problem Statement / Project Vision**

This app aims to encourage people to try new recipes. The idea will help users organize recipes they would like to try as well as the ingredients needed for the recipe. The app will be able to let users categorize their recipes as ‘Breakfast’, ‘Lunch’, ‘Dinner’, ‘Desserts’, and ‘Snacks’. Since this will be a single-user app, I will also include the ability for users to edit current entries. They will be able to edit descriptions, ingredient lists, instruction steps, and category tags. This app will be made to be used on a phone. Thee stakeholders will be the project team and the users. The project team will have spent much time building this app, and they will want it to function the way they have planned and, if possible, be successful. The users will want the app to benefit them in the way they were promised. They want it to be a record holder for all of their recipes.

## **System Capabilities**

There will be an Add Recipe function ID number RE111. The user will find a recipe they would like to add to the list and add it in.

There will be an Add Ingredients function ID number IG222. The user will add ingredients that match the recipe that was added.

There will be an Add Steps function ID number ST333. The user adds steps for cooking the recipes added and will later use them to cook.

There will be a Search Recipes function ID number SR444. The user will search and find a specific recipe from the recipes they have collected and saved.

There will be a Check function ID number CL555. When finished using a recipe or steps, users will be able to click check. For more detail, look to the requirements Model Section in the System Proposal.

## **Non-functional Requirements and Design Constraints**

We don’t have a big budget, so it is hard to include advanced features. The main non-functional requirement for this app is usability; it should be easy and accessible to users of all skill levels. Since the developer doesn’t have a lot of experience, the main concern from the Feasibility Assessment is organization feasibility. For additional details, please look at Sections 1 ,3, and 4 in the System Proposal.

## **System Evolution**

* + 1. **Version 2 Changes**

In version 2, we will deliver advanced tools like a voice recording function. We will also be adding a clear function for all of the checklists.

* + 1. **Version 3 and beyond Changes**

In Version 3, we would like to add visual demonstrations for the recipes, like having a step-by-step video to prepare the meal. As well as a social media feature where the users can share their meals and recipes.

## **Document Outline**

The rest of this system specification will include an analysis of the app's structural model, architectural design, as well as the user interface.

# **Structural Model**

## **Model Introduction**

These sections contain the class diagram and the Metadata. The class diagram contains the class Recipe with its attributes and methods, the class Ingredients with its attributes and methods, and the class Steps with its attributes and methods. The class diagram also shows the association between the classes. The metadata provides extra information about the attributes and methods.

## **Class Diagrams**

A diagram of a recipe

Description automatically generated

## **Metadata**

A close-up of a recipe

Description automatically generated

Description: The ingredient class recpresents an ingredient used in a recipe. It has information such as the ingredients name, quantity and measurements.

Visibility: public

Is abstract: no

Additonal Information

**Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Description | Read Only? | Multiplicity |
| IngredientsID | Unique identifier for Ingredients | Yes | 1 |
| Name | Name of the ingredient | No | 1 |
| Quantity | Number of Ingredients needed | No | 1 |
| Measurements | Measurement of the ingredients needed | No | 1 |
| Checked | Whether or not the ingredient is checked off. | No | 1 |

**Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Description | Is Query? | Is Polymorphic? |
| AddIngredient | Adds an ingredient to a recipe | No | No |
| ClickCheck | Marks the step as checked | No | No |

**Processing Outlines**

AddIngredient()

Set Id to a unique Identifier

Set the name

Set quantity

Set checked to FALSE

Save the new ingredient in recipe

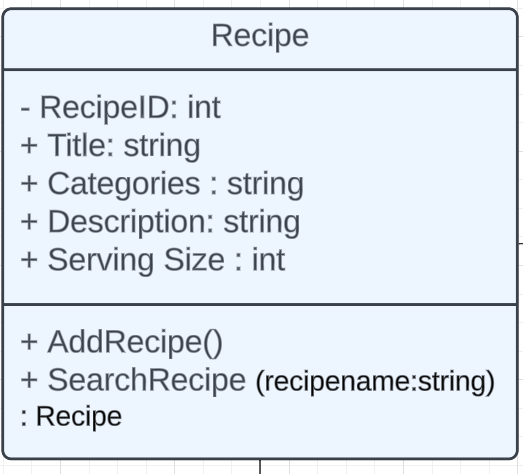
ClickCheck()

IF Checked IS FALSE

SET check to TRUE

Else

SET check to FALSE

****

Description: The Recipe class holds the recipe information.

Visibility: public

Is abstract: no

**Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Description | Read Only? | Multiplicity |
| RecipeID | Unique identifier for Recipe | Yes | 1 |
| Title | Title of the Recipe | No | 1 |
| Categories | The Category of the Recipe | No | 1 |
| Description | Description of the Recipe | No | 1 |
| Serving Size | The amount of servings is needed | No | 1 |

**Methods**

| Name | Description | Is Query? | Is Polymorphic? |
| --- | --- | --- | --- |
| AddRecipe | Adds a new Recipe to the recipe list | No | No |
| SearchRecipe | Searches for a recipe | Yes | No |

**Processing Outlines**

AddRecipe()

Set Id to a unique Identifier

Set the Title

Set the Description

Save the new recipe in the recipe list

SearchRecipe(recipename:string) : Recipe

For recipe in recipe list

IF recipe.name EQUALS recipename

RETURN recipe

RETURN 0

**A screenshot of a computer program

Description automatically generated**

Description: The Steps class holds the steps of the recipe.

Visibility: public

Is abstract: no

Additional Information:

**Attributes**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Description | Read Only? | Multiplicity |
| StepsID | Unique identifier for Steps | Yes | 1 |
| Description | Description of what needs to be done | No | 1 |
| Checked | Whether or not the step is checked off. | No | 1 |

**Methods**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Description | Is Query? | Is Polymorphic? |
| AddSteps | Adds a step to a recipe | No | No |
| ClickCheck | Marks the step as checked | No | No |

**Processing Outlines**

AddSteps()

Set Id to a unique Identifier

Set Description

Set checked to FALSE

Save the new step in recipe with RecipeID

ClickCheck()

IF Checked IS FALSE

SET check to TRUE

Else

SET check to FALSE

# **Architecture Design**

## **Architecture Overview**

This will be a multi-tier client-server system. The design will show you several components. User’s Mobile Device where the app is installed and executed. Web Server, which will handle the HTTP requests from the mobile device and distribute it to the application servers. Application Server, which handles API requests, serves as the core of the functionality of the app. Database Server, which stores all the recipe data managed by a DBMS

## **Infrastructure Model**

* + 1. **Deployment Diagram 1 – Architecture Overview**

A computer server diagram with text

Description automatically generated with medium confidence

* + 1. **Deployment Diagram 2 – Nodes and Artifacts**

A diagram of a computer

Description automatically generated

## **Hardware and Software Requirements**

* + 1. **Hardware Components**

Some hardware that will be needed are processors, SSD, and memory. If the currently owned HW meets the requirements and specifications, then it can be reused. However, if the currently owned HW doesn’t run as needed, there might be a need to purchase a new one. They don’t need any other product.

* + 1. **Required Software Components**

The device will need to be a smartphone, and the Software needed is the operating system, which can be an Android or an IOS, and a reliable internet connection. Also, another component will be a firewall.

## **Security Plan**

* + 1. **Security Overview**

This security plan addresses some potential threats like data breaches, disruptions, and unauthorized access.

* + 1. **Security Plan**

|  |  |  |
| --- | --- | --- |
| Threats  Components | Disruption, Destruction, Disaster | Unauthorized Access |
| User’s Device : Mobile |  | 1 |
| Internet | 5 | 5 |
| Web Server | 2 | 2 |
| Application Server | 3 | 3 |
| Database Server | 4 | 4 |

Controls

1. Secure app Permissions
2. Web Application Firewall and updates
3. Secure API
4. Data encryption, regular backups, and Database Monitoring
5. Network firewalls

# **User-Interface**

## **User-Interface Requirements and Constraints**

In this section, you will be able to see the Window Navigation as well as the wireframe. You will get a glimpse into what I am hoping the app is going to look like and how it is supposed to work.

## **Window/Screen Navigation Diagram**

A white board with blue writing

Description automatically generated

## **UI Wireframes**

**A menu of a meal

Description automatically generatedA screenshot of a recipe menu

Description automatically generatedA screenshot of a recipe form

Description automatically generatedA screenshot of a recipe menu

Description automatically generatedA screen shot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated**

## **Reports: "Formal Output" Design**

# **Appendices**

## **Glossary**

* Hardware: the external and internal devices and equipment that enable you to perform major functions such as input, output, storage, communication, processing, and more.
* Software: a set of instructions, data or programs used to operate computers and execute specific tasks.
* Firewall: a network security device that monitors incoming and outgoing network traffic and permits or blocks data packets based on a set of security rules.
* API: stands for Application Programming Interface.
* Attributes: properties or characteristics that can be assigned to elements like variables, objects, or classes in programming.
* Methods: a behavior of an object parametrized by a user
* Metadata: Metadata is "data about data", and in this case, data about the classes, their attributes/data elements, and operations/methods.

## **References / Bibliography**

Cameron, A. (n.d.). Panopto. <https://spu.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=4f4ab69e-60bd-4264-adc5-b1630175a9f5>

## **Supporting documentation**