***App-etizers at SC***

**Program Requirements Specifications**

**Version 1.0**

**Prepared by:**

**Yuming Fei, Renhe Song, Xizhe Ma, Yuting Gao, Peng Gan, Marika Perlmutter**

**CSCI 201: Principles of Software Development**

**Table of Contents**

**Table of Contents**

**Revision History**

1. **Introduction**

1.1 Purpose

1.2 Document Conventions

1.3 Intended Audience and Reading Suggestions

1.4 Project Scope

1.5 References

1. **Overall Description**

2.1 Product Perspective

2.2 Product Features

2.3 User Classes and Characteristics

2.4 Operating Environment

2.5 Design and Implementation Constraints

2.6 User Documentation

2.7 Assumptions and Dependencies

1. **System Features**

3.1 Sign-in and Sign-up

3.2 Search Bar/Query

3.3 Result Pages

1. **External Interface Requirements**

4.1 User Interfaces

4.3 Software Interfaces

4.4 Communication Interfaces

1. **Other Nonfunctional Requirements**

5.1 Performance Requirements

5.2 Safety Requirements

5.3 Security Requirements

5.4 Software Quality Attributes

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| Yuming Fei | Oct. 21 | Adding product features--- Sort reviews based on user’s preference | 1.0 |

**Introduction**

**1.1 Purpose**

This document specifies requirements for building the web application *App-etizers at SC* version 1.0. The document specifies how the developers should review and edit the project, how the target audience (CSCI 201 TAs, CPs, and instructor) should read it, how to build the project features, which languages, libraries, databases, and APIs the project will use, what type of operating environment the project requires, and how the project will function under different external circumstances.

**1.2 Document Conventions (For Developers)**

**Please refer to the project template file in the CSCI 201 Final Project folder if you have questions about writing a specific section.** When editing this document, please use the Arial font in size 11 (which is the Google Docs default). **Please bold sections of information that you think are important, instructional, or otherwise helpful to other developers or to users.** Every time you make changes to this document, please fill in the revision history table above and contact the development team by text or email to let the other project members know. Moreover, when doing research, please save links to websites, libraries, and databases so that they can be put into the reference section for future usage. If developers refer to new libraries, be sure to add them into the 1.5 Reference section so that they are visible and available to all members.

**1.3 Intended Audience and Reading Suggestions**

The document is specifically written for developers in the project group so that it is convenient for them to communicate about the changes and progress of this program. **For developers writing these specifications for the first time, please read section 1.2 before making changes.** This document is also written for testers and project managers (CSCI 201 TAs, CPs, and the professor). For reading, please refer to the table of contents if you have difficulty in finding specific sections. All the titles and subheadings are bolded, and some sentences are bolded because of their importance. Be sure to consult them if you have any questions.

**1.4 Project Scope**

*App-etizers at SC* is a web app that aggregates data from menus from restaurants on and near the USC campus. It aims to help students view food choices in a more convenient manner so that they can save time deciding on meal choices for nearby restaurants. They are not necessarily restricted to campus dining halls and restaurants, but they can discover more food choices outside the campus (for example, Blaze, Subway, and Wingstop).

**1.5 References**

Google map API for a website built-in map to display possible restaurants choices.

**[Additional references needed (restaurant websites, USC dining hall map list, others)]**

We will also refer to restaurants that are included in Grubhub, in addition to specific restaurant websites like <https://www.wingstop.com/> and <https://www.subway.com/en-us>, in order to build our menus and choices for different users’ specifications.

**Overall Description**

**2.1 Product Perspective**

The purpose of this web app is to make students’ on-campus dining decisions more streamlined and convenient. Students, especially those with limited USC meal plans, often find it difficult to decide whether the daily menu at a dining hall or restaurant would be worth the trip. With the *App-etizers at SC* website, USC students can easily view and compare what daily options the dining halls have to offer. In addition, they can also consult menus from various restaurants around the campus in order to provide a greater variety of choices for meals.

**2.2 Product Features**

*App-etizers at SC* allows users to type in specific ingredients, flavors, and types of food, and the website will display relevant meal options. The site will offer a chart for users to reference when searching for the food they want. For example, if the user types “beef,” the website will show steakhouses near the user or other restaurants whose menus offer beef products. Typing multiple keywords is also possible. The website has a built-in map to display locations of restaurants, in order for users to make informed decisions about which eating establishments are most convenient. Guest users can only view the menus, the map, the price of meals, and the rating of each restaurant. However, if they choose to log in to an account, they can see the rating of each dish from the selected restaurants. In addition, they will be able to write reviews and give ratings to the restaurants and dishes, which will be visible to other users.

Reviews are sorted based on users preference. If the user chooses the most recent one, then they will get the latest review. The default filter displays most-liked reviews

**2.3 User Classes and Characteristics**

Most users will be newer USC students who are relatively unfamiliar with the surrounding area or whose meal plans limit their access to dining halls. *App-etizers at SC* serves to introduce students to the various dining options near them. Visitors to USC are also potential users of this website, but their usage frequency is estimated to be less than that of USC students. In addition, school staff and faculty members can use the program for personal eating choices or to assist in making school policy decisions about USC students’ dining environment.

**2.4 Operating Environment**

The project is web-based. As long as users have Internet connection, they can view the website. The current intended server for the website is Tomcat, which may need to be installed in order to run the program for testing purposes (for both developers and users). In the future, with sufficient funding, the project can be deployed on a better and more accessible server, and users can go to the website directly without any installation.

**2.5 Design and Implementation Restraints**

The app faces various limitations in terms of scope and design, including but not limited to the following:

* Certain restaurants’ failure to accurately list menu information on their websites
* Differences in user OSes (MacOS, Microsoft Windows, etc.)
* Restriction to web browsers only
* Data received from existing restaurants’ sites
* Usage of Java language with associated dynamic web project technologies (AJAX, JSP, etc.)
* Usage of SQL for database access and manipulation
* Parallel retrieval and comparison of restaurant data
* Real-time updates of user reviews
* Communication between app database and user web page
* Security requirements surrounding user login details (e.g., ensuring protection and uniqueness of login info)
* Relational design components of restaurant map, keyword search results, and user review list
* Responsibility of programming team to periodically maintain and update restaurant database for item accuracy

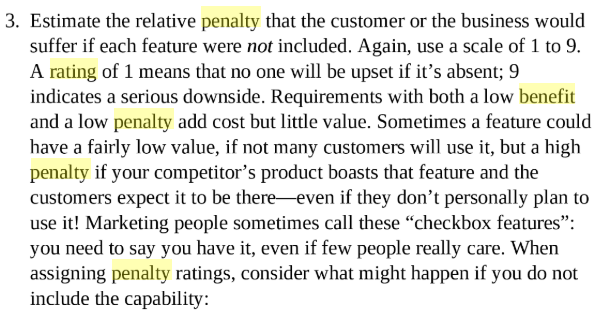
**2.6 User Documentation**

Upon delivery, the software team will include documentation to assist the product recipients with understanding and implementing the provided program. This documentation will include the project files themselves (with associated in-line comments for clarification), the restaurant database used for data sorting and analysis, relevant online official documentation sources for the languages and technologies used, initial requirements pages provided by the user (CSCI 201), and this current specifications document. Files will be included in a zipped PDF format unless otherwise specified.

**2.7 Assumptions and Dependencies**

Several assumed factors will influence the design of this project. Database accuracy is dependent upon the correctness of existing campus restaurant websites, as well as the dining locations map provided by USC (<https://hospitality.usc.edu/dining-map/>), which will form the basis for the interactive restaurant map on the program web page. The service on which the site is deployed and hosted will also vary according to future CSCI 201 assignment specifications. Finally, on a macro scale, it is reasonably assumed that users will possess software and hardware compatible with running the *App-etizers at SC* web application, though factors including internet access, web browser, server, and operating system type, and dynamic web page display will affect the final result.

**System Features**

****

Source: <https://books.google.com/books?id=nbpCAwAAQBAJ&pg=PT635&lpg=PT635&dq=benefit+rating+penalty+rating&source=bl&ots=9oQUFW7xTj&sig=ACfU3U0Xn1wVj1j8GDojyP24-5pF8m78vQ&hl=en&sa=X&ved=2ahUKEwi12YPy7avlAhVDXK0KHc9wBUcQ6AEwCnoECAkQAQ#v=snippet&q=benefit%20rating%20penalty%20rating%20cost&f=false>

**3.1 Sign-up and Sign-in**

3.1.1 Description and Priority

**<HIGH>** *App-etizers at SC* allows users to sign up to view specific dishes’ ratings and to comment and rate on dishes. Since the benefit to signing up is the ability to view and give ratings on food, the benefit rating is 7. The penalty rating is 4, since even if the user does not sign up, they can still view the menus, the map, the price, and the rating of a restaurant. The cost rating is 1, since it takes one minute to sign up.

3.1.2 Stimulus/Response Sequences

*User actions (1):* Click sign up.

*System responses (1):* Direct to sign-up page.

*User actions (2):* Input username, email, password, and repeat password, check the conditions and terms.

*System responses (2):* Check username availability and email format/availability. Check password strength and check if the passwords are the same. Check if the terms and condition box is checked.

*User actions (3):* Click to verify sign up.

*System responses (3):* Create user account.

3.1.3 Functional Requirements

Error conditions and invalid input: *Username exists*, in which case prompt user to re-pick a username. *Email invalid,* in which case prompt user to type correct email address. *Email exists* in which case prompt user to re-pick an email. *Password too weak* in which case prompt user to choose a stronger password. *Passwords different* in which case prompt user to check if they are the same.

**Database:** The app needs to store existed user information.

*Valid Username:* User signs up with a valid username (does not already exist).

*Valid Email:* User signs up with a valid email (does not already exist and is formatted correctly).

*Valid Password:* User signs up with a valid password (strong enough and is the same as the retyped password).

*Terms and Conditions:* User checks the checkbox “terms and conditions”.

*Privacy Consent:* User checks the checkbox “privacy policy”.

**3.2 Search Bar/Query**

3.2.1 Description and Priority

**<HIGH>**Search bar is intended for users to input any information (i.e dishes, ingredients) in order to search for restaurants.

|  |  |  |  |
| --- | --- | --- | --- |
| Benefit | Penalty | Cost | Risk |
| 9 | 9 | 3 | 1 |

3.2.2 Stimulus/Response Sequences

1. User inputs a string of query word(s)
   1. No response
2. User selects a searching criteria, (e.g., dishes, restaurant names, ingredients, etc.)
   1. Highlight the chosen criteria
3. User clicks on the “Search” button or presses “enter” on the keyboard
   1. System displays loading icon and will redirect to result pages

3.2.3 Functional Requirements

REQ-1: Provide a list of criteria: dishes, ingredients, restaurant names. Each of them should have an associated pop-up icon for visibility.

REQ-2: System must respond to input queries from users according to searching criteria, and should redirect from search pages to result pages. If the system is not able to produce results, there should be an error message informing users. Also, an empty query will not be submitted to the server, to save performance. Upon encountering other connection errors, the system should not break down; rather, the system will inform users of the failure and prompt for possible solutions. For example, asking users to switch to another internet provider or sending an email to the server host.

Optional O-REQ1: The search bar is able to provide a drop-down list of common user queries (i.e., autocompletion) in an effort to increase ease of use.

**3.3 Result Pages**

3.3.1 Description and Priority

**<HIGH>** Results pages give a list of search results according to user inputs on [3.2]. Each entry is a **restaurant** followed by brief text information. On the header section, there is a search bar for quick access if the result is not satisfying.

|  |  |  |  |
| --- | --- | --- | --- |
| Benefit | Penalty | Cost | Risk |
| 9 | 9 | 9 | 9 |

3.2.2 Stimulus/Response Sequences

1. Users scrolling through pages
   1. System should display the rest of the pages.
2. Users move the cursor across entries
   1. Entries will be highlighted
3. Users click on the highlighted entries
   1. Redirect to detail pages.
4. Users decide to search again
   1. Response should be identical to **Function [3.2]**
5. Users choose a sorting filter
   1. System will dynamically refresh the entries according to the filter they have chosen.
6. Users click on an entry’s address
   1. System will redirect to Google Maps.

3.2.3 Functional Requirements

**Non-login users:**

REQ-1: Each entry should include a picture for the restaurant. The picture is customizable for the login users to vote on, and the highest vote will be displayed in the entry. A general text logo should be applied if no picture is provided.

REQ-2: Each entry should include a brief text introduction.The introduction could be either from official users’ most recent review comments or from the restaurants’ official description.

REQ-3: Each entry should include multiple tags (each tag limited to 3 words) to give restaurants identity and specialization.

REQ-4: Each entry should include a clickable address. The system will redirect users to Google Maps, and a pin will be shown on the map for accurate localization.

REQ-5: Entries will not be sorted for unofficial (guest) users.

REQ-6: Entries should display a color code to indicate the restaurant’s availability status (Red: “closed” or “closed forever,” Orange: “closed soon” or “open soon,” Green: “open currently”)

**Login users:**

REQ-6: Entries will be sorted by the rating of restaurant by default. If two restaurants have the same rating, then the one with more review comments will be in higher priority in the sorting.

REQ-7: Users have the ability to view rating stars in result pages.

REQ-8: Users have the ability to choose additional special filters. For example, they can sort by dishes’ prices from low to high or from high to low.

**External Interface Requirements**

4.1 User Interfaces

The user will be directed to a website that has clear instructions on it. There will be a search bar where users can type in their desired dish. Once a choice is made, the user will then be directed to a list of restaurants offering those dishes, in addition to their locations on the map. Next to the dishes and restaurants, there will be prices and ratings. If the user chooses to click on one of the dishes, they will be able to see relevant food photos (if they exist) and the reviews that other users wrote.

On the top right corner, there will be a login button for people to create or log in to their account to rate and write reviews for restaurant.

4.2 Software Interfaces

The web app will only contain restaurants on or around campus; therefore, a fixed database will be built inside the program. The map within the web app that shows the locations will use the Google Maps API. A customer database will also be completed to record the information each user typed in while signing up. The user’s username and password will also be saved. Since the MySQL database is more secure, we will use MySQL database to add the login feature on the website.

4.3 Communication Interfaces

Our app requires modern browsers using HTTP for data transferring. This includes, but is not limited to, the following browsers: Firefox, Chrome, Safari, Edge, Opera, Vivaldi, IE 9.0 or later.

To ensure users’ data security, the servlet will handle sensitive data for validation instead of using Javascript. Also, the servlet is powered by Google Cloud SQL for secure connection between servlet and database. This further prevents the possibility of malevolent users breaching through our system. Note that our system does not allow SQL syntax input on the search bar. This should be done by syntax checking on the servlet.

All pages are asynchronous by default. The only exception is on the detail pages where

users are able to view real-time update on the comment. This is done by multi-threading

on the server side to ensure the no-lagging experience.

**Other Nonfunctional Requirements**

5.1 Performance Requirements

The system will be available 24/7. The user interface should not “hang” during operations that require significant amounts of time (> 1s) to complete. The response will be fast enough to operate within a reasonable user’s attention span.

5.2 Safety Requirements

If *App-etizers at SC* removes any files from the user’s system (reviews, ratings, etc.), a prompt must be displayed asking the user for confirmation.

5.3 Security Requirements

Usernames and passwords for access to the *App-etizers at SC* website will be stored securely. Only website developers can see the profile records of the official (logged in) users.

5.4 Software Quality Attributes

*Availability:* The system will be available at all times. Specific periods of availability or unavailability will be reported by the web developers.

*Usability:* System usability will be measured via online user surveys every 3 months.