**Software Requirements and Design Document**

**For**

**Group <31>**

Version 1.0

**Authors**:

Jayna Spikes

Donald Walton

Brunas Joseph

Mihye Lim

Kesnel M

# Overview (5 points)

CraveRank is a web-based application designed to help users find top-rated restaurants based on their location and food preferences. By entering a location and selecting food categories, users receive a ranked list of the best five nearby restaurants by default. The system allows users to filter results based on different preferences, such as price range, cuisine type, and customer ratings. Additionally, users can create an account to save personalized settings and queries for a more tailored experience.

# Functional Requirements (10 points)

**User Registration and Authentication** (High Priority)

* The system shall allow users to register an account using an email and password.
* The system shall support secure authentication via login credentials.

**Search Functionality** (High Priority)

* The system shall allow users to enter a location (city, zip code or full address) and select food categories to retrieve restaurant recommendations by default.
* The system shall default to returning the top five ranked restaurants based on user input for the location.

**Filtering and Sorting Options** (High Priority)

* The system shall allow users to filter restaurant search results by price range, cuisine type, rating, and distance.
* Users shall be able to sort results based on relevance and rating using the star system.

**Personalized User Experience** (Medium Priority)

* Registered users shall be able to save their past searches and preferences.
* Users shall be able to create a list of favorite restaurants so it can be saved on their personalized accounts.

**Restaurant Details and Reviews** (High Priority)

* The system shall display restaurant details, including name, address, phone number, website, rating, and available reviews from yelp’s API.

**API Integration** (High Priority)

* The system shall fetch restaurant data from external APIs like yelp.
* The system shall update restaurant information dynamically based on API responses.

**Responsive UI** (Medium Priority)

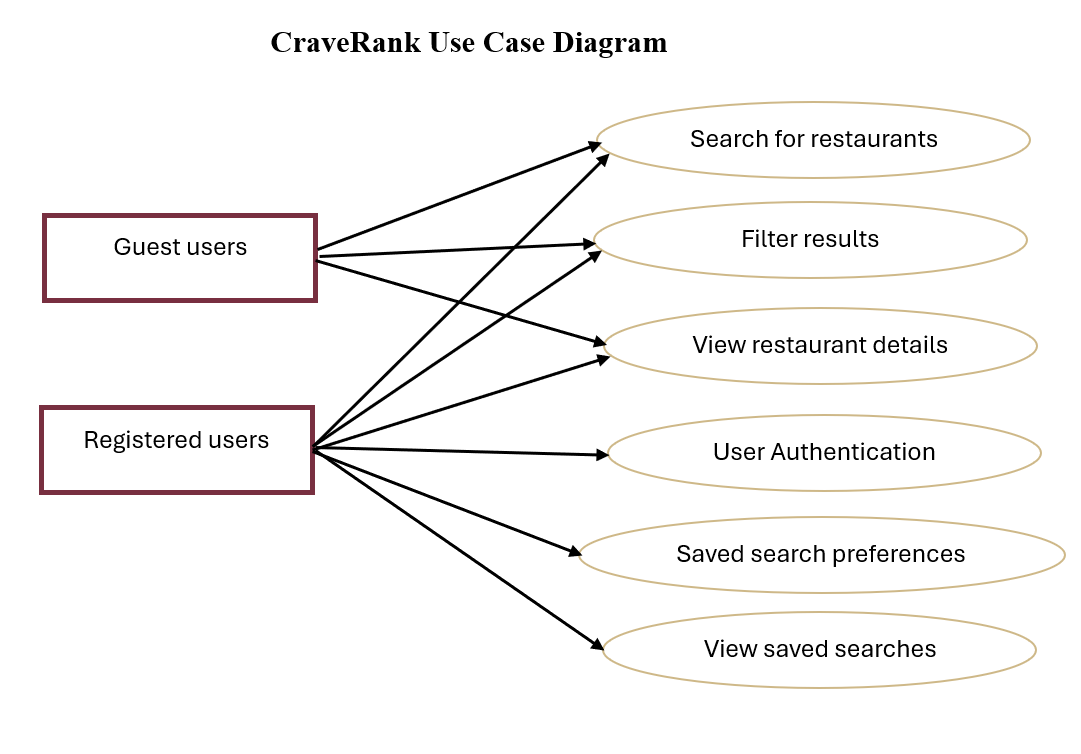
* The system shall be web-based friendly and support desktop, tablet, and smartphone displays on multiple search engines.

# Non-functional Requirements (10 points)

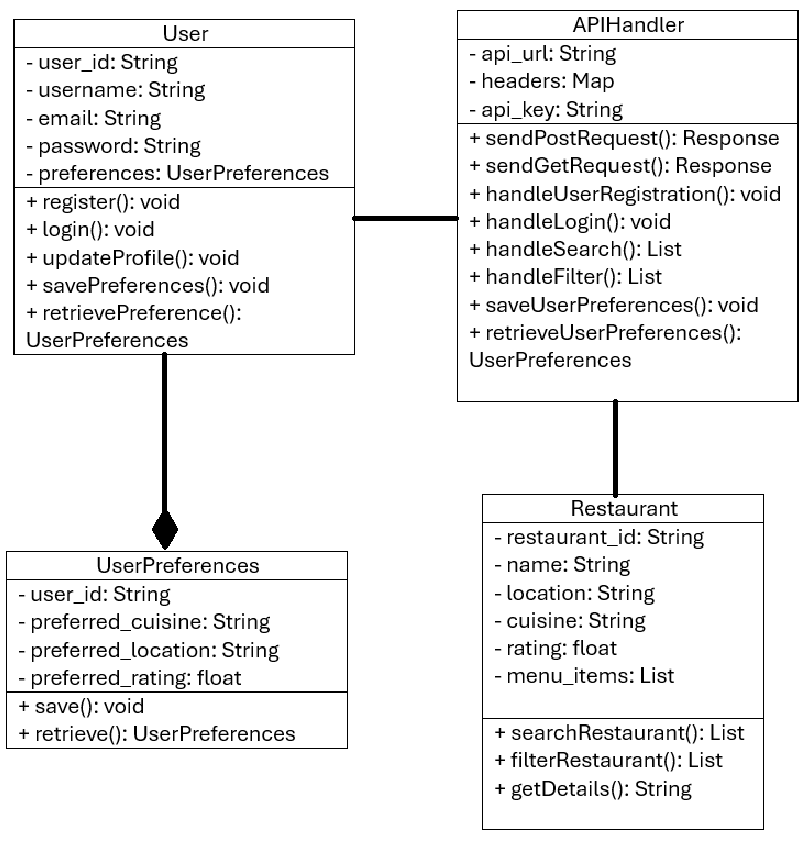
1. **Performance**
   * The system shall return restaurant recommendations within 3 seconds of a query submission showing reliability and reactiveness.
2. **Scalability**
   * The system shall handle many users simultaneously without people degrading the performance of our website.
3. **Security**
   * User passwords shall be securely stored using hashing and encryption techniques and it should be stored into our database for clarification when the user connects to our website.
   * The command will perform a check to see if the user is in our database. If not, they will be prompted to register.
4. **Reliability and Availability**
   * The system should have a high uptime demonstrating reliability and availability for users and administrators.
5. **Usability**
   * The interface should be simple and easy to navigate for user quick needs.
   * The system shall support major web browsers such as Chrome, Firefox, Safari, Edge.
6. **Maintainability**
   * The system should be modular, meaning that code should be grouped into different sections so that it can help detect issues and fix bugs immediately.

# Use Case Diagram (10 points)

A Use Case Diagram will be included to illustrate interactions between users and the system, covering key functionalities like user authentication, restaurant searches, filtering, and saving preferences



# Class Diagram (15 points)



Three sequence diagrams will depict interactions for: user registration and login, searching for restaurants and filtering results, and finally, saving and retrieving the user preferences.

# Operating Environment (5 points)

The system shall run on cloud-based servers with at least 2 CPU and 4GB RAM. Allow all admins different systems to access it. The backend will run on a Linux system. The software that will be used is python (Flask), HTML, CSS, Javascript, MySQL, Yelp’s API

# Assumptions and Dependencies (5 points)

* The system assumes that Yelp’s API will remain available and functional.
* It is assumed that users will have stable internet connections to access the platform.
* The project depends on cloud hosting services for deployment.