

# Restaurant Menu Bot Documentation

Umut ŞENER

## Contents

<b>1</b>	<b>Project Overview</b>	<b>2</b>
<b>2</b>	<b>File Structure</b>	<b>2</b>
<b>3</b>	<b>Class Architecture</b>	<b>2</b>
3.1	MenuItem and Derived Classes . . . . .	2
3.2	Menu Class . . . . .	3
3.3	User Class . . . . .	3
3.4	AI Recommender . . . . .	3
<b>4</b>	<b>JSON Data Handling</b>	<b>3</b>
4.1	Master Menu . . . . .	3
4.2	User Data . . . . .	3
<b>5</b>	<b>Main Program Flow</b>	<b>4</b>
<b>6</b>	<b>Input Validation</b>	<b>4</b>
<b>7</b>	<b>Taste Matching Algorithm</b>	<b>4</b>
<b>8</b>	<b>Conclusion</b>	<b>4</b>

# 1 Project Overview

The **Restaurant Menu Bot** is a C++ console application that allows users to create a custom menu, interactively add or remove items, and receive AI-based recommendations based on taste preferences. It stores user information and menus persistently using JSON files.

Key features include:

- Loading a master menu from `menu.json`.
- Adding items from master menu or creating custom items.
- AI taste-based recommendation using a simple linear model.
- Saving and loading multiple users' data in `user_data.json`.
- Interactive console interface with input validation.

## 2 File Structure

- `main.cpp` – Main program logic, user interaction loop.
- `Menu.hpp` / `Menu.cpp` – Menu and item class definitions and JSON serialization.
- `Recommender.hpp` – AI linear regression recommender.
- `menu.json` – Master menu data with items, prices, and taste attributes.
- `user_data.json` – Stores multiple users' menu and preferences.
- `weights.json` – Hold the AI's data inside.

## 3 Class Architecture

### 3.1 MenuItem and Derived Classes

**MenuItem** is the abstract base class representing a generic menu item. It contains:

- Name, price, and a 5-element taste array (`sweet`, `sour`, `bitter`, `salty`, `savory`).
- Getters and setters for all fields.

Derived classes include:

- **Starter** – Includes `isHot` attribute.
- **Salad** – Includes `hasTopping` attribute.
- **MainCourse** – Includes `isVegetarian` attribute.
- **Drink** – Includes `isCarbonated` and `hasAlcohol`.
- **Appetizer** – Includes `serveTime` (before/after main course).
- **Dessert** – Includes `extraChocolate` boolean.

## 3.2 Menu Class

The `Menu` class manages a collection of `MenuItem` pointers:

- Add/remove items.
- Show full menu or filtered by type.
- Compute total cost.
- Save/load to/from JSON files.

## 3.3 User Class

Represents a user with:

- First name, last name, and gender.
- An associated `Menu` object.
- Methods to serialize/deserialize JSON.

## 3.4 AI Recommender

The `Recommender` class predicts user satisfaction based on a linear regression model:

- Input: 5-element taste vector.
- Output: Predicted satisfaction (0–10).
- Supports updating weights based on user feedback.

# 4 JSON Data Handling

## 4.1 Master Menu

`menu.json` contains all available items, categorized:

- Starters, Salads, Main Courses, Drinks, Appetizers, Desserts.
- Each item has name, price, and taste attributes.

## 4.2 User Data

`user_data.json` contains user's last entry:

- Name Surname
- Taste Profile
- Last Menu

## 5 Main Program Flow

1. Load master menu from `menu.json`.
2. Prompt user for first and last name.
3. Load previous user data if exists, otherwise ask for gender.
4. Enter main menu loop:
  - Browse master menu and add items.
  - Add custom items with price and taste attributes.
  - Remove items from the menu.
  - AI-based recommendations (full menu or by type).
  - Predict satisfaction for a specific item. Also train AI.
  - Save user data to `user_data.json`.
  - Exit program.

## 6 Input Validation

- Integer input within bounds (`getInt`).
- Yes/No prompt (`getYesNo`).
- Safe string input (`safeGetline`).

## 7 Taste Matching Algorithm

AI recommendation uses weighted Euclidean distance:

$$D = \sqrt{\sum_{i=1}^5 w_i (t_i^{user} - t_i^{item})^2}$$

where  $w_i$  are taste weights and  $t_i$  are taste scores. The smallest distance indicates the best match.

## 8 Conclusion

The Restaurant Menu Bot demonstrates:

- Object-oriented design with inheritance for menu items.
- Persistent storage using JSON.
- Simple AI for personalized menu suggestions.
- User-friendly console interface with input validation.