CODE:

**Recipe.java**

This is the class that defines the structure of a recipe.

import java.util.List;

public class Recipe {

String name;

List<String> ingredients;

String steps;

int cookingTime; // in minutes

public Recipe(String name, List<String> ingredients, String steps, int cookingTime) {

this.name = name;

this.ingredients = ingredients;

this.steps = steps;

this.cookingTime = cookingTime;

}

public void displayRecipe() {

System.out.println("\nRecipe Name: " + name);

System.out.println("Ingredients: ");

for (String ingredient : ingredients) {

System.out.println("- " + ingredient);

}

System.out.println("Cooking Time: " + cookingTime + " minutes");

System.out.println("Steps: " + steps);

}

}

**RecipeManager.java**

This is the main class that manages the recipes, allowing users to add, view, search, and delete recipes.

import java.util.List;

import java.util.ArrayList;

import java.util.Arrays;

import java.util.Scanner;

public class RecipeManager {

static List<Recipe> recipes = new ArrayList<>();

static Scanner sc = new Scanner(System.in);

public static void main(String[] args) {

int choice;

// In OnlineGDB, we can't use file I/O, so commenting out load/save operations

// loadRecipes();

do {

System.out.println("\n1. Add Recipe");

System.out.println("2. View All Recipes");

System.out.println("3. Search Recipe by Name");

System.out.println("4. Search Recipe by Ingredient");

System.out.println("5. Delete Recipe");

System.out.println("6. Exit");

System.out.print("Choose an option: ");

choice = sc.nextInt();

sc.nextLine(); // clear buffer

switch (choice) {

case 1 -> addRecipe();

case 2 -> viewAllRecipes();

case 3 -> searchRecipeByName();

case 4 -> searchRecipeByIngredient();

case 5 -> deleteRecipe();

case 6 -> {

System.out.println("Exiting...");

// saveRecipes(); // Can't save to a file in OnlineGDB

}

default -> System.out.println("Invalid choice.");

}

} while (choice != 6);

}

static void addRecipe() {

System.out.print("Enter recipe name: ");

String name = sc.nextLine();

System.out.print("Enter ingredients (comma separated): ");

String ingredientsInput = sc.nextLine();

List<String> ingredients = Arrays.asList(ingredientsInput.split(","));

System.out.print("Enter cooking time (in minutes): ");

int cookingTime = sc.nextInt();

sc.nextLine(); // clear buffer

System.out.print("Enter steps: ");

String steps = sc.nextLine();

Recipe recipe = new Recipe(name, ingredients, steps, cookingTime);

recipes.add(recipe);

System.out.println("Recipe added.");

}

static void viewAllRecipes() {

if (recipes.isEmpty()) {

System.out.println("No recipes found.");

return;

}

for (Recipe recipe : recipes) {

recipe.displayRecipe();

}

}

static void searchRecipeByName() {

System.out.print("Enter recipe name to search: ");

String name = sc.nextLine();

boolean found = false;

for (Recipe recipe : recipes) {

if (recipe.name.equalsIgnoreCase(name)) {

recipe.displayRecipe();

found = true;

}

}

if (!found) {

System.out.println("Recipe not found.");

}

}

static void searchRecipeByIngredient() {

System.out.print("Enter ingredient to search: ");

String ingredient = sc.nextLine();

boolean found = false;

for (Recipe recipe : recipes) {

if (recipe.ingredients.contains(ingredient.toLowerCase())) {

recipe.displayRecipe();

found = true;

}

}

if (!found) {

System.out.println("No recipes found with that ingredient.");

}

}

static void deleteRecipe() {

System.out.print("Enter the name of the recipe to delete: ");

String name = sc.nextLine();

boolean found = false;

recipes.removeIf(recipe -> {

if (recipe.name.equalsIgnoreCase(name)) {

System.out.println("Recipe deleted.");

return true;

}

return false;

});

if (!found) {

System.out.println("Recipe not found.");

}

}

// In-memory only (no file I/O for OnlineGDB)

static void saveRecipes() {

System.out.println("Recipes won't be saved in this environment.");

}

static void loadRecipes() {

System.out.println("No previous recipes to load.");

}

}