Project Summary

The **Smart Cooking Assistant** is an AI-powered application designed to guide users through meal preparation while personalizing their cooking experience. The assistant adapts to each user's preferences, skill level, and available kitchen tools, using an intelligent profiling system that evolves based on interactions. This project aims to create a tool that not only provides recipes but also enhances the user's culinary skills by offering tailored suggestions, step-by-step guidance, and dynamic optimization of recipe plans. Whether the user is a novice cook, a parent managing dietary restrictions, or a caregiver navigating complex nutrition needs, the assistant learns and grows with each interaction to deliver an increasingly personalized experience.

Key to the assistant's functionality is the creation of detailed user profiles. These profiles store information about food preferences, available ingredients, and kitchen tools, and are continuously updated through feedback during and after the cooking process. By leveraging the OpenAI ChatGPT API, the system uses machine learning models to analyze user inputs (both text and voice), extracting insights that further refine the profile. This allows the assistant to suggest recipes and tools that suit the user's specific needs. The assistant can also connect with external apps, such as grocery trackers, to update profiles based on ingredient availability and provide real-time recipe suggestions based on what's in the pantry.

The initial prototype will feature a desktop-based graphical user interface (GUI) with text and speech input capabilities. Users will interact with the assistant conversationally, and while voice recognition is a long-term goal, it may be omitted in early versions. The assistant will display recipe instructions, ingredient lists, and other relevant data, all while maintaining a smooth, guided user experience. The prototype will rely on Python dicts stored as JSON files for profile management, although further research into scalable data storage (e.g., databases) is planned as the project progresses.

In addition to the OpenAI API for natural language processing, future versions may incorporate APIs for voice recognition and ingredient tracking. As the assistant grows, the focus will be on refining how user feedback is handled and how the assistant dynamically updates profiles in real time, ensuring that each cooking experience is optimized for the individual.

This project explores the boundaries of AI in daily life, particularly in the kitchen, where personalization can have a significant impact on user satisfaction and cooking efficiency. The ultimate goal is to create a system that not only improves the cooking process but also enhances the user's confidence and knowledge over time.

GitHub Repository: Cooking Assistant

Task Vignettes

1) User Recipe Selection and Cooking Assistance

Vignette:

John is a novice cook looking to prepare dinner for his family. He opens the Smart Cooking Assistant on his laptop and is greeted by a friendly prompt asking how he would like to proceed. John asks for recipe suggestions, and the assistant replies with a few personalized options based on his profile, which includes his family's dietary preferences (vegetarian) and the ingredients available in his pantry. John selects a "Vegetable Stir-Fry" recipe from the suggestions.

The assistant verifies that John has all the required ingredients and tools, cross-referencing the grocery list and tools stored in his profile. After confirming he has everything, the assistant walks John through the steps, pausing after each one to ensure he's ready to continue. The assistant also provides additional context when John asks for clarification on how to properly dice vegetables, utilizing video or image instructions. Throughout the process, the assistant notes John's feedback on each step, improving the profile for future recipe recommendations.

Technical Details:

- Input:
  - The user requests a recipe suggestion via voice or text. Inputs are passed to the OpenAI ChatGPT API to generate responses.
- Profile Access:
  - The system retrieves user profile data (dietary preferences, ingredient availability, and cooking skills) to personalize recipe suggestions.
- External Data:
  - The system checks the user's ingredient inventory by accessing linked grocery tracking apps or stored profile data.
- User Interface:
  - The GUI displays recipe options with the ability to select and confirm ingredients. After selection, a step-by-step guide is displayed. The assistant also provides an option to replay or elaborate on any specific step.
- Voice Interaction:
  - Basic voice interaction for receiving user commands and asking clarifying questions (optional for early prototypes).
- Data Flow:
  - User interaction data is collected and stored in the profile, refining future interactions and recipe suggestions.
- Feedback Loop:
  - User feedback after completing steps is analyzed by the LLM to further optimize future experiences.

2) Profile Creation and Personalization

Vignette:

Sarah is a new user of the Smart Cooking Assistant and needs to set up her profile. Upon launching the assistant, she is asked if she would like the assistant to remember her preferences. Sarah confirms, and the assistant asks a few key questions to personalize her profile: "Do you have any dietary restrictions?", "Which cuisine do you prefer?", and "Which cooking tools do you use frequently?"

Sarah inputs that she is vegan, loves Mediterranean cuisine, and has a slow cooker, blender, and basic kitchen tools. The assistant uses Sarah's answers to create her initial profile, which will be refined over time based on her cooking behavior and additional feedback. Sarah can also integrate her grocery app with the assistant, allowing it to track her pantry items and recommend recipes accordingly. Once the profile is created, Sarah can immediately start receiving personalized recipe suggestions.

Technical Details:

- Input:
  - Initial user responses to questions about dietary restrictions, cuisine preferences, and kitchen tools.
- Profile Creation:
  - A user profile is created, with key preferences stored in a JSON file (or later in a database). Each user has a unique profile that evolves based on interactions.
- External Integration:
  - The assistant connects with third-party apps (e.g., grocery trackers) to sync data on ingredient availability.
- Data Flow:
  - Data is stored locally or on the cloud depending on user preference. Profile updates happen automatically when Sarah provides feedback or makes changes to her preferences.
- User Interface:
  - The profile setup is guided via a clean, user-friendly GUI with text and optional voice interaction.

3) Ingredient Substitution and Alternative Suggestions

Vignette:

Emma is preparing a recipe for lasagna when the Smart Cooking Assistant informs her that she is missing one key ingredient: ricotta cheese. The assistant suggests several substitutions based on the ingredients Emma has in her pantry, such as cottage cheese or a tofu-based alternative. It also offers to adjust the recipe slightly to accommodate the substitution. Emma selects the cottage cheese option, and the assistant seamlessly updates the instructions.

Technical Details:

- Input:
  - Missing ingredient triggers the assistant to suggest alternatives.
- Data Handling:
  - The assistant accesses the user's pantry data, retrieves ingredient substitutions from a pre-defined database and suggests alternatives.
- User Interaction:
  - Emma selects an alternative via the smart display or verbally.
- Output:
  - The recipe is dynamically updated with the new ingredient.

4) Cooking Progress Reminders and Timers

Vignette:

David is making a slow-cooked stew using the assistant, and after prepping the ingredients, he starts the cooking process. The assistant reminds him to check on the stew every 30 minutes and provides updates on the remaining cooking time. When it's time to add the vegetables, the assistant alerts him via voice and updates the progress on the screen.

Technical Details:

- Input:
  - Cooking start triggers reminders.
- Data Handling:
  - Timer data is managed locally or on the cloud, with notifications linked to each cooking step.
- User Interaction:
  - Verbal or text feedback acknowledges the alert and moves forward with the cooking process.
- Output:
  - Verbal reminders, and visual updates on the screen.

5) Post-Meal Feedback and Profile Update

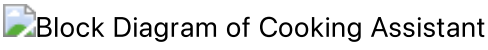
Vignette:

After completing the meal, Lisa provides feedback to the assistant on how it went. She rates the recipe and answers a few follow-up questions about the cooking process—whether she found it easy or if there were confusing steps. The assistant uses this feedback to further refine Lisa's profile, noting her preference for faster, easier recipes and reducing the complexity of future recipe suggestions.

Technical Details:

- Input:
  - User feedback through verbal or text input, rating system.
- Data Handling:
  - The assistant processes feedback and updates the user profile accordingly.
- User Interaction:
  - Post-cooking feedback, and recipe rating on the display or through voice.
- Output:
  - An updated profile that adjusts future recommendations.

Assistant Data Flow Block Diagram



Block Descriptions

1) **User Input Block**

- Description:
  - This block represents all user interactions, including voice commands, text input, and feedback after cooking.
- Data Flow:
  - Input data includes strings (text or voice input), which are sent to the Natural Language Processing (NLP) and Profile Management blocks.
- User Interaction:
  - Input (verbal/text input for recipe suggestions, ingredient checks, and post-cooking feedback).
- Data Flows to:
  - NLP Block, Profile Management Block

2) Input Natural Language Processing (NLP) Block

- Description:
  - Processes the input from the user, converts voice/text commands into structured strings.
- Data Flow:
  - Receives user input as strings and outputs a structured string of recognized commands or queries.
- User Interaction:
  - None directly, purely backend processing.
- Data Flows to:
  - Recipe Selection Block, Feedback Analysis Block

3) Feedback Analysis Block

- Description:
  - Analyzes user feedback to update the user’s profile and improve the personalization of the assistance
- Data Flow:
  - Inputs include user feedback from strings/dictionaries and updates to the profile. Outputs updated preferences to the Profile Management Block.
- User Interaction:
  - None directly; feedback is stored and analyzed behind the scenes.
- Data Flows to:
  - Profile Management Block

4) Profile Management Block

- Description:
  - Manages user-specific preferences, dietary restrictions, past cooking behaviors, and available tools.
- Data Flow:
  - Uses dictionaries or JSON files to store user profiles. Receives user feedback from the User Input Block with context from the Feedback Analysis Block and updates the profile.
- User Interaction:
  - Output (prompts the user to confirm profile creation, updates, or modifications).

- Data Flows to:
    - Recipe Selection Block, Ingredient Checker Block, Feedback Analysis Block, Profile Database, and Output NLP

5) Profile Database

- Description:
    - Stores profile dictionary variables as JSON files that can be accessed by the profile manager
- Data Flow:
    - Triggered by the profile manager, json files flow to a storage location
- User Interaction:
    - None, interaction is all in the backend
- Data Flows to:
    - Profile Management Block

6) LLM Recipe Knowledge

- Description:
    - We're going to utilize the ChatGPT model's base understanding of cooking to determine the majority of the recipe information and guidance steps.
- Data Flow:
    - Supplies the Recipe Selection Block with contextual cooking information to fill in the gaps not specified by the user profile.
- User Interaction:
    - None, all interaction is in the backend
- Data Flows to:
    - Recipe Selection Block

7) Recipe Selection Block

- Description:
    - Compiles the user input string, with the profile into a structure string that is then fed to an LLM that determines an optimal recipe for the user
- Data Flow:
    - Pulls from LLM Recipe Knowledge Block and cross-references user profile data to output an optimized recipe object (dictionary).
    - If needed, the block sends a list of new ingredients or tools that need to be validated.
- User Interaction:
    - Output (displays recipe options to the user, and asks for confirmation to proceed).
- Data Flows to:
    - Output NLP, Cooking Guidance Block

8) Cooking Guidance Block

- Description:
    - Provides step-by-step cooking instructions and manages progress through the recipe.
- Data Flow:
    - Inputs the selected recipe object (dictionary) and presents each step as a string to the user.

    - Outputs visualizations as required to aid the user
    - Controls the timers and reminders
        - These timers will be internally monitored and notifications provided when needed
    - Controls the trigger for post-cooking feedback acquisition and analysis
- User Interaction:
    - Output (guides user through cooking steps via voice/text and visual feedback).
- Data Flows to:
    - Timer/Reminder Block, Post Cooking Feedback Block, Output NLP, UI Manager Block

9) Timer/Reminder Block

- Description:
    - Manages cooking time, setting reminders for when to check on food, add ingredients, or perform specific tasks.
- Data Flow:
    - Receives time-sensitive data (e.g., "set timer for 20 minutes") and triggers alerts for the user.
    - Output data in integer (time) format.
- User Interaction:
    - Output (provides verbal/text reminders about time-related tasks during cooking).
- Data Flows to:
    - UI Manager

10) Post-Cooking Feedback Block

- Description:
    - Collects user feedback on the recipe and cooking experience to further refine the profile and future interactions.
- Data Flow:
    - Receives alert that recipe has been completed, analyzes the conversation for key issues or hang-ups, and provides 3 personalized follow-up questions to the output blocks.
- User Interaction:
    - Output (provides questions related to the recipe execution chat log and the user’s profile)
- Data Flows to:
    - Output NLP

11) Output NLP

- Description:
    - Collects information from various parts of the program logic and outputs a string that can be read to the user.
- Data Flow:
    - Receives communication requests from various other blocks and needs to prioritize and communicate these requests to the user in a way that is logical and doesn’t confuse the user.
- User Interaction:
    - Output (provides the logic and organization of what the assistant says to the user. Since we want the UI to be conversational this is important and requires its own block)
- Data Flows to:
    - UI Manager

12) UI Manager

- Description:
  - Collects information from the Output NLP Block, Cooking Guidance Block, and Timer Block and either speaks, prints, or visualizes the information according to its type.
- Data Flow:
  - Receives strings from the Output NLP Block, structured recipe data from the Cooking Guidance Block, and times from the Timer/Reminder Block.
  - Also, provides the means for reading the next user input as either a typed string or voice input.
- User Interaction:
  - Output (provides primary UI/face of the assistant)
- Data Flows to:
  - User Input

Current Self Assessment

What is the biggest or most unexpected change I had to mkae from the sketch?

- I wasn't expecting how the data flow would break down and what the major logical tasks the assistant would need to do.
- I was pleasently surprised that the tasks lend themselves to modularization fairly well which will help with the subfunction development and troubleshooting.

How condifent do I feel that I can implement the spec as it's written right now?

- Not confident at all but I wrote the spec to include things that I don't currently know how to do or may not have enough time to complete.
- My goal is to scope the concept and then determine how much of an MVP I can develop in a semester.
- I know that things like the voice recognition are long shots to be implemented in the inital prototypes but I feel its important to note that in the spec since the intial development might be tailored towards that goal while not actually being able incoorporate it.

What is the biggest potential problem that you NEED to solve?

- I think a temporary UI that allows for conversation like communication without the voice recognition is crucial to testing the inital prototypes but I'm not sure how "glossed up" it needs to be.
- If something more in depth is required it could add more work that will slow down other progress.

What parts am I least familiar with and could use some help?

- Scoping GUI requirements
- Scoping Databasing and Data Security requirements
  - I'm not sure how much of this is needed for a prototype, I'm assuming not much but could be a case of I don't know what I don't know.