**SSH – Recipe Suggestion Feature**

Engineering Design Review

**Author**: Rohit Mamtora

**Date**: 31st October 2024

**Status:** Final Version

**Introduction**

Maintaining healthy eating habits is a challenge that many university students face. Students often find it difficult to balance their university work, social life, and day-to-day activities including chores, cooking or shopping. In addition, some students may hold part-time jobs, adding to the challenges of time management and motivation for essential daily tasks. Our SSH camera currently provides us with key metrics relating to a student house’s fridge contents; but it currently offers limited direct benefits to the students.

We propose developing a recipe suggestion feature to streamline the students’ cooking experience within their home. This feature uses our existing fridge metrics to suggest meal recipes to save students’ valuable time. Specifically, it tailors meal ideas based on dietary restrictions, as well as general preferences that can be inputted through filters on the SSH cloud, app, and console table.

This function is powered by two primary databases: the fridge contents database and the recipe database. Using data from the current ingredients and user-selected filters, the recipe database is queried to generate a ranked list of recipes, organised from most to least suitable for the user. Additional information is also displayed if the user is missing one or two ingredients. This feature streamlines meal planning by offering students easy, personalised cooking ideas to access from our existing SSH services and devices.

**Goals and non-goals**

* **Goal:** Utilise the real time ingredient inventory (assumed to be an existing feature of the SSH camera) in line with the recipe database, suggesting meals based on fridge contents. Display and rank all recipes needing 0 to 2 missing ingredients.
* **Goal:** Introduce personal preferences and dietary restrictions to limit what recipes are displayed to each user in the household using optional filters that can be applied to the recipe database. Use provided information as well as amount of ingredients missing to rank the suggested recipes. Filters should be 85% accurate before feature release.
* **Goal:** Develop a user-friendly interface accessible via the SSH cloud, app or console table, incorporating images and clear descriptions of the displayed recipe.
* **Non-Goal:** Integrate dietary restrictions in the user’s profile metrics to streamline suggestion ranking.
* **Non-Goal:** Generate an AI model that automates recipe suggestion based on nutritional information, time of the day and meal history, without the need for filters. This would need to be at least 60% accurate before implementation.
* **Non-Goal:** Extend the capabilities of the SSH camera to work in line with other ingredient storage spaces and allow for real-time cooking assistance. Allowing for over 3 SSH camera’s to be used simultaneously.

**Design Overview**

**Front-End**

Our SSH devices already have functional user interfaces. With the addition of a button/tab (or other UI implementation) we can direct a user to the Recipe suggestion function’s landing page. This page will initially display a checklist of filters that can be applied, followed by a “Begin generating recipes” button. The checklist will be organised into specific sections with the following headers: Type of Meal, Dietary Restrictions, Type of Protein, Cooking Time. The contents under each header will include the following:

|  |  |  |  |
| --- | --- | --- | --- |
| **Type of Meal** | **Dietary Restrictions** | **Type of Protein** | **Cooking Time** |
| Any  Breakfast  Lunch  Dinner  Snack  Dessert  Smoothie | Vegetarian  Vegan  Dairy-Free  Gluten-Free  Nut-Free  Paleo  None | Any  Chicken  Eggs  Beef  Pork  Lamb  Fish & Seafood  Plant-Based Meats  None | Any  0-10 Minutes  10-20 Minutes  20-30 Minutes  30-45 Minutes  45+ Minutes |

A ranked list of suitable recipes will display, each featuring: an image, name, cooking time and ingredients missing if any. Once the user has selected a recipe to use, another page will load, displaying all relevant information for the given recipe ID, including a list of ingredients to use and step-by-step instructions.

**Back-End**

The primary function will be to search the recipe database against the list of ingredients captured by the SSH camera. We will use SQL querying in order to achieve this, however there are some prerequisites that will need to be ironed out. To ensure that ingredient names and units are consistent, we will include an ingredients table and a unit table. Ingredients will be generalised i.e. ‘milk’ will cover all types of milk (e.g. soya, almond, whole).

The SSH can already track the ingredients stored in the fridge and who owns them. With the addition of our Ingredients and Units table, we can assume the following layout for our fridge database:

|  |  |  |
| --- | --- | --- |
| **Table name** | **Fields** | **Relevance** |
| Fridge\_contents | item\_id  user\_id  ingredient\_id  quantity  unit\_id | We will use the ingredient\_id and quantity and check with our recipe database. |
| Units | unit\_id  name | Ensures that units are consistent. |
| Ingredients | ingredient\_id  name  unit\_id | Ensures that ingredient names are consistent. |

To achieve consistency in unit measurements and ingredient names, our recipe database will be structured as follows:

|  |  |  |
| --- | --- | --- |
| **Table Name** | **Fields** | **Explanation** |
| Recipes | recipe\_id  dish\_name  meal\_type  protein  cooking\_time  img\_url | Contains the main information that applies to a specific recipe. Contains almost all of the information that will be displayed when suggesting a recipe. |
| Recipe\_ingredients | recipe\_id  ingredient\_id  quantity  unit\_id | Outlines the ingredients and what quantity is needed for the given recipe\_id. Also contains the unit of each measurement. |
| Instructions | instruction\_id  recipe\_id  step\_number  instruction\_description | Stores the step-by-step instructions that will be displayed once the user chooses a recipe. |
| Dietary\_requirement | dietary\_id  name | Gives an id to all dietary requirements |
| Dietary\_suitability | recipe\_id  dietary\_id | References recipe\_id and what dietary\_id it is suitable for. |

We will utilise cross database access to join our Recipe\_ingredients table with our Fridge\_contents table to check each required ingredient against the ingredients owned by the current user. Missing ingredients will be counted to a maximum of 2 and recipes will be ordered based on missing ingredients. A filtering function will be applied to our query based on the user’s input. The ‘dietary restrictions’ filter takes precedence over our ‘Type of protein’, for example if a user has disclosed that they are Vegan but have selected Chicken in the ‘Type of Protein’ filter, the feature will only display Vegan dishes. Our final query should return the recipe\_id of all recipes that match both the current user’s ingredients and the filters applied.

All recipe\_id’s selected by our query will be organised and displayed for our user to select their chosen recipe. The action of clicking a recipe will return its recipe\_id which will be used to select all its relevant information and displayed as described above.

**Alternatives**

**Use of embedding and TF-IDF when finding suitable recipes**

Instead of solely relying on queries to find suitable recipes, we can use TF-IDF to group recipes based on ingredients within the fridge.TF-IDF measures the importance of a word to a collection of documents and while it will provide an accurate representation of suitable recipes, it may not be the best fit for our goals. This approach creates issues with quantity measures, splitting our process into two steps. An SQL query will still be needed in order to make sure that available quantity is equal to or more than the required quantity. TF-IDF would be useful if our ultimate goal was to rank our recipes based on suitability, however with the introduction of filters, less pressure is put on how appropriate a recipe is and more pressure is put onto whether or not a recipe is valid based on a query. Finally, further steps in lemmatisation and text normalisation would be crucial in preventing inconsistencies in the model.

**Alternative ranking metric**

We are currently ranking recipes based on the missing ingredients however there are numerous ways in which they can be ranked. One example is to introduce a preference metric where the user can input their general food preferences. Extra data would need to be stored in our recipe database and that data would passed into an algorithm, displaying recipes based on user preference. While this may be something to visit in future versions, it currently exceeds our scope. We could also introduce a difficulty metric rank recipes based on the skill required along with its cooking time. Again, this exceeds the scope of our current design. Compared to the alternative ranking systems, our current ranking system is broad, however the main focus is suggesting recipes based on the contents of the user’s fridge.

**Addition of specific ingredient search**

Our existing filters are ‘Type of Meal’, ‘Dietary Restrictions’, ‘Type of Protein’ and ‘Cooking Time’. Additionally, we can introduce a search bar for quick ingredient inclusion. This would allow the user to enter specific ingredients that they may not already own and would improve but complexify our ranking system. As our program already compares ingredients available against required ingredients, we have decided that this feature conflicts with our current model and would yield no further customer satisfaction.

**Milestones**

**Milestone 1:** Standardise ingredient entries to the SSH fridge database with the addition of the Ingredients and Units tables. This schema normalisation will streamline cross-database referencing, ensuring consistency in ingredient names and measurement units to avoid discrepancies across the SSH ecosystem.

**Milestone 2:** Define the user interface wireframes for the recipe suggestion landing page across the SSH cloud, app and console table. These wireframes will establish a structured blueprint for presenting filter options and displaying query results.

**Milestone 3:** Set up our recipes database and populate with mock data. Ensure secure and consistent connections for referencing external data sources, preparing for inter-database queries.

**Milestone 4a:** Populate the fridge database with mock data and begin structuring our query requests. Use mock data to test and debug any discrepancies, ensuring the integrity base query functionality.

**Milestone 4b:** Apply simulated filter sets to our base query structure, assessing compatibility of each filter option with the current query model. Identify and resolve any issues with the current logic systems in preparation for real filter integration.

**Milestone 5:** Implement the finer details of our landing page, including our filters checklist and recipe generation button. Display query results with recipe name, image, cooking time and missing ingredients.

**Milestone 6a:** Refine the back-end querying system to synchronise with incoming front-end data. Establish precedence of ‘dietary restrictions’ over ‘type of protein’ and apply the recipe ranking metric described in the design overview. With our queries now established and robust, focus will shift towards enhancing efficiency.

**Milestone 6b:** (optional) Optimise performance by indexing high-usage columns and implementing caching mechanisms to reduce latency and improve recipe retrieval speeds.

**Milestone 7:** Build the detailed recipe page to display all information relevant to the recipe chosen by the user. Ensure that information is displayed in a coherent manner and further optimise with an accessibility review.

**Milestone 8:** Deploy first version of the recipe suggestion feature in the next update for the relevant SSH devices.

**Dependencies**

***UI/UX Design team*** – Structure the blueprint of our landing page as well as detailed recipe view.

***LII team*** – Conduct accessibility review based on the Design Team’s landing page prototype and provide back-end engineers with suitable test cases to examine our queries against.

***Database team*** – Introduce our 2 new tables to the existing fridge database to ensure ingredient standardisation. Furthermore, set up the recipe database and ensure that cross-database access is granted for the fridge database.

***Legal team*** – Ensure recipe details are of fair use and review our privacy agreements with customers to prevent possible liabilities.

**Cost**

With the enhancement of our current fridge contents database and additional recipe database, we can expect an increase in storage and bandwidth needs. This could increase our hosting costs. While the recipe database may rely on third-party APIs, building it in-house with the Database team may reduce costs. Due to the nature of this feature, we will need real-time updates of our fridge contents, however, as the SSH camera already has the capabilities to keep track of this, we can predict that this will not affect our costs. Finally, unless all recipes are of fair use, we may need to anticipate licensing fees. In summary, additional costs are expected for the development and maintenance of this new feature.

**Privacy and security concerns**

* Our users are already familiar with the SSH camera, and since we do not require the camera to perform any functions beyond its current uses, we can reasonably assume that implementing this feature will not introduce any new privacy or security concerns regarding the camera's data captures. If changes need to be made in how we handle data from the SSH camera, new permission request will be released in line with the Data Protection Act and GDPR.
* Dietary preferences are classified as Personally Identifiable Information (PII) so encryption and secure handling will be crucial should we decide to incorporate this in our user’s stored metrics.

Users will be asked for permission before we store dietary information.

* Should we choose to use any third-party APIs in the development of our recipe database, we will need to ensure that they comply with our current security standards.

**Risks**

|  |  |
| --- | --- |
| **Risks** | **Mitigations** |
| Data inconsistencies in ingredient and unit storage | Maintain a strict schema normalisation to avoid issues with ingredients being mislabelled. |
| Insufficient recipe suggestions | Ensure that our recipe database is populated with a wide variety of recipes to ensure accessibility for all users. |
| Performance issues as databases develop | Optimise queries with indexing on high-use fields and cache common filter combinations, improving response time. |
| Third-party API dependency | Maintain local backups of critical data and regularly review and assess quality and uptime of third-party APIs to avoid data breaches. |
| Misuse of SSH camera and potential data breach. | Security measures are already in place to ensure that the SSH camera and the data it collects aren’t abused. |
| Legal risk from copyrighted recipe usage | Only use open-source libraries to build our recipe database or enquire our legal team into licensing recipes. Ensure all images used are of fair-use. |

**Supporting materials**

* Almoraie, N.M., Alothmani, N.M., Alomari, W.D. and Al-amoudi, A.H. (2024). Addressing nutritional issues and eating behaviours among university students: a narrative review. *Nutrition Research Reviews*, [online] pp.1–16. doi: https://doi.org/10.1017/S0954422424000088.
* Bbc.co.uk. (2019). *Recipes - BBC Food*. [online] Available at: https://www.bbc.co.uk/food/recipes.
* Copyrightlaws.com: Copyright courses and education in plain English. (2020). *Copyright Protection in Recipes*. [online] Available at: <https://www.copyrightlaws.com/copyright-protection-recipes/>.
* Garcia, R. (2020). *What Personal Data is Sensitive? - Complete IT*. [online] Complete IT. Available at: https://www.complete-it.co.uk/what-personal-data-is-sensitive/ [Accessed 31 Oct. 2024].
* ico.org.uk. (2023). *Personal information - what is it?* [online] Available at: https://ico.org.uk/for-organisations/uk-gdpr-guidance-and-resources/personal-information-what-is-it/.
* Leitch, J. (2021). *Building a Recipe Recommendation System*. [online] Medium. Available at: <https://towardsdatascience.com/building-a-recipe-recommendation-system-297c229dda7b>.
* Reddit.com. (2022). *Reddit – Recipe app database design*. [online] Available at: https://www.reddit.com/r/SQL/comments/xy74h1/recipe\_app\_database\_design/ [Accessed 29 Oct. 2024].
* The Real Food Dietitians. (2016). *Recipe Filter*. [online] Available at: https://therealfooddietitians.com/recipe-filter/ [Accessed 28 Oct. 2024].