Student Smart Homes – Recipe Suggestion Solutions (RSS)

Engineering Design Review

Author: Luke Thornley
Date: 31st October 2024

Introduction

Many university students face the challenge of managing busy schedules, balancing academics, extracurricular activities, and social lives. Given their extensive responsibilities, students often neglect household duties such as cleaning and laundry. Cooking, in particular, is often seen as time-consuming, with it requiring shopping, preparation and clean up just to have a simple healthy meal. This can lead to students to relying on other alternatives such as less healthy ready-made meals, paying more to eat out or skipping meals all together. This lack of time for meal preparation not only impacts diet quality but also contributes to food and monetary waste. Students often leave purchased groceries unused, leading them to expire.

To address these challenges, at Student Smart Homes (SSH) we propose the *Recipe Suggestion Solution (RSS)* with the aim to help students make the most of their ingredients. The RSS feature takes the live contents of a fridge and provides the student with a list of potential recipes they can make, utilising the ingredients they already own. These recipes will only make use of ingredients that each individual users put in the fridge as to prevent the use of others food. In the event a student is missing part of the ingredients to a recipe, the RSS will automatically provide a shopping list of the exact items and quantity the student needs to buy. This is an extension of the SSH Camera and SSH Cloud, so is available to any student who own these products. This feature will simplify the cooking process for students, allowing them to reduce time and energy spent on keeping track of their food items and trying to come up with what they can cook.

Goals and non-goals

- **Goal:** Provide users with recipes categorised as Complete (all ingredients available) or Semi-Complete (some ingredients missing), based on fridge contents and provide students with a list of shopping items based on missing ingredients.
- **Goal:** Save meal preparation decision time for students. Time reduced should at least be 20% on average.
- **Non-Goal:** Calculate the monetary cost of ingredients or offer budgeting assistance for planning meals
- **Non-Goal:** Customisation of recipes for dietary requirements (e.g. vegetarian, gluten intolerant)

Design Overview

Leveraging the SSH Camera, SSH Hub and SSH Cloud, the RSS will detect and log fridge items in real time and will categorise and suggest recipes based on available ingredients, generating a shopping list for recipes that are nearly complete.

The RSS's design relies on structured data integration and SQL-based querying to manipulate three key tables: the fridge_contents database, dish_recipe database and shopping_list.

Table	Fields
fridge_contents	• **food_item**: The name of the ingredient (e.g. "milk") • **item_quantity**: The count of each ingredient (e.g. 4 eggs) • **item_weight:** Specific weight or volume for items where applicable (e.g. 500g of chicken) • **expiry_date:** The expiration date of each item • **item_owner:** The student who added the item to the fridge
dish_recipe	• **recipe_name**: The name of the recipe (e.g., "Thai Green Curry"). • **preparation_steps**: Step-by-step instructions for completing the recipe. • **required_ingredients**: list of required ingredients for the recipe • **quantity_required**: The exact amounts needed for each ingredient in the recipe. • **weight_required**: The exact weight needed for each ingredient in the recipe.
shopping_list	• **incomplete_recipe:** The name of the incomplete recipe that requires a list **food_item:** The ingredient to buy • **absolute_quantity:** quantity of respective ingredient to buy • **absolute_weight:** weight of respective ingredient

Fridge Contents Table
The SSH Camera uses its built-in AI, optical recognition (OCR) and facial recognition to identify food items, labels and students. This data is transmitted as a CSV file via the SSH Hub, listing items along with quantities, weights and owners to match the table. Each time an item is added or removed, a new or updated entry is recorded in the **fridge_contents** table, ensuring that the database remains accurate and up to date. Entries in **fridge_contents** are grouped by **item_owner** and ordered from earliest to latest expiry date.

Existing Recipe Database Integration
The **dish_recipe** table is populated using recipe data from an external source. The data is retrieved via an API, which allows the SSH system to request and process data into a structured format. Recipe names, ingredients lists and required quantity and weights, are all formatted to fit the SSH Cloud Schema for simple comparison with fridge contents. Once populated, the recipe table remains unchanged, ensuring consistency and to allow for efficient querying.

Recipe Categorisation
To determine which recipes are available to users based on their current fridge contents, the RSS

queries the **dish_recipe** table, and parses through the list of ingredients (**required_ingredients)**. For each ingredient in a given recipe, the system confirms that the available quantity and weight meet or exceed the amounts specified in the **quantity_required** field. This is done by performing simple arithmetic using fields from both tables:

$$(item_{\_quantity} - quantity_{\_required}) \ \& \ (item_{\_weight} - weight_{\_required})$$

If either of these calculations are negative, the ingredient will be labelled insufficient, and their absolute values will proceed to be logged in the **shopping_list** table under the **absolute_quantity** and **absolute_weight** fields.

Using this, the recipe database will then be categorised into 3 groups, dependant on the number of insufficient items in each recipe respectively.

-Complete: Recipes where all ingredients are labelled as sufficient. These recipes are immediately available for users to prepare.

-Semi-Complete: Recipes with at least 70% of items labelled as sufficient. For these recipes, any missing or insufficient ingredients are logged in the **shopping_list** table.

-Incomplete: These recipes do not meet the 70% threshold are not displayed to users.

The thresholds are calculated by taking the number of sufficient ingredients in each recipe and dividing that value by the total number of ingredients.

Shopping List Generation
For recipes grouped as 'Semi-Complete' the RSS will automatically generate a shopping list, using the previously logged insufficient ingredients, and their respective absolute quantities and weights.

User Interface
The SSH Cloud User Interface will have 2 main views, one displaying the current contents of the fridge, where students can only view their own items, and another listing 'Complete' and 'Semi-Complete' recipes. Once a recipe is selected, another page is opened displaying step-by-step cooking instructions for students to follow, as well as a shopping list outlying the items and specific quantity or weight they needed to complete the recipe.

Alternatives

User Manual Input:

Instead of the SSH Camera recording and sending fridge content data to the SSH Cloud, we could implement it so that the student inputs all the information manually.

- *Pro:* Without the use of cameras, it would remove the need for implementation of a connection between the SSH Camera and SSH Hub.
- *Pro:* Less mistakes due to unforeseen errors such as camera obstruction and system malfunction.

- *Con:* Would be time consuming for the user having to write every key piece of information
- *Con:* There can still be mistakes due to human error
- *Con:* Would defeat the purpose of everything being automated and streamlined to make stude

Frequently Updating Recipe Database:

Instead of using a preexisting database that doesn't change, an alternative approach could be regularly updating the database with recipes from external sources such as websites with methods like web scraping.

- *Pro:* Users have access to new recipes, giving them more variety to choose from.
- *Con:* Frequent updates to the database can introduce complications as newly added recipes would require categorisation and implementation into the existing database.
- *Con:* Using other external sources could lead to inconsistencies in data quality, potentially disrupting recipe suggestions.

Alternative threshold for Semi-Complete Recipes

We could decide on another threshold to classify a recipe as Semi-Complete. This could be lowered for students looking for a larger option of recipes or could be increased for those who want a minimal shopping list. Instead, we assume that this will be a customisable option for users in the future.

Milestones

Milestone 1: Using the AI ingredient recognition feature of the SSH Camera, implement the back-end logic for automatic updates to the **fridge_contents** table through the SSH Hub. This allows us verify the accuracy and speed of data being populated and updated. If we find that the data is too inaccurate, slow or populating the table incorrectly, we can stop here.

Milestone 2: Implement API integration to populate the **dish_recipe** table with external recipe data. We can use this with the **fridge_contents** table and perform calculations to test functionality using real data.

Milestone 3: Implement the back-end logic to categorise recipes as Complete, Semi-Complete or Incomplete, and generate the shopping list. We can perform targeted tests to confirm the functionality of the recipe categorisation system.

Milestone 4: Extend SSH Cloud to support detailed recipe views, personalised shopping lists and all remaining RSS features.

Milestone 5: Collaborate with the design team to create a user-friendly layout for the page. We can conduct student demos on the final design to gather feedback on interface functionality and usability.

Milestone 6: Optimise the system for deployment. We will also conduct security and privacy reviews to verify data protection.

Dependencies

-*Research Team:* We will need structured recipe data for the **dish_recipe** table

-*SSH Products Team:* Will need to ensure accurate recognition from the Camera, as well as implement ability to send information to the SSH Cloud regularly

-*Database Team:* We will need tables to track food and recipes, as well as queries written to categorise items

-*Legal Team:* We will review legal agreements related to filming and storing household data to limit legal liability

-UI Team: Design a user-friendly page for students.

Cost

We anticipate some concurrent costs for up-keeping RSS. Although there will be minimal additional cloud storage costs for the **fridge_contents** and **dish_recipe** databases, additional processing costs will arise from the frequent updates of the database, SQL querying, and real-time data syncing. If a third-party recipe API is integrated there may also be a charge.

Privacy and security concerns

We must be careful when storing sensitive data such as recordings of people's faces. Recordings from the SSH Camera should never be stored in the cloud and instead should only be temporarily stored locally within the camera. This local video data should also be set to delete, once the SSH Camera has updated the database. As the SSH Camera provides a live feed inside students homes, it is also crucial to secure this feed against hacking attempts. Additionally, personal data such as student identities used in facial recognition needs to be stored securely to protect privacy. Ingredient and recipe data, while less sensitive individually, can reveal students' eating habits when aggregated over time. To safeguard this information, ingredient data should be automatically deleted once items are used or have expired, further limiting potential privacy risks.

Risks

Risks	**Impact**	**Mitigations**
Camera is obstructed/Food labels cannot be read	Food Items are not or inaccurately identified.	Option to enter information manually
Someone using another person's food item	Could lead to inaccurate tracking and recipe suggestions to wrong people	Register food to only first person to place each item
Security Breach	Information stored on the SSH devices are compromised leaving users vulnerable	Use encryption and limit access to sensitive data. Perform regular checks for malware and signs of intrusion.

Server interruption	Fridge content doesn’t update, leading to wasted food or inaccurate recipes	Complete regular system checks and optimise data flow between SSH Products

Supporting Material

- University student practices and perceptions on eating behaviours whilst living away from home **International Journal of Educational Research., 2023 Volume 117**