SSH – Recipe Suggestion Function

Engineering Design Review

Author: Grace Daniels
Date: 19 October 2024
Status: Draft

Introduction

In student households, managing food supplies and planning meals can be a daily challenge. Most university students are experiencing full independence for the first time, and, amongst their academic studies and other everyday chores, deciding what to eat for dinner is the last thing they want to consider.

To address this problem, we aim to make use of the SSH Camera and its ability to check the contents of a student home refrigerator. Currently, this allows for functionalities such as automatically billing those who take food that isn't theirs, which provides more trust between household members. However, we believe that the SSH Camera's AI features that detect what each household member has added to the fridge could be used to target a student's needs more specifically.

We propose to enhance the SSH Cloud with a *recipe suggestion function* that uses the data collected by the SSH Camera. This feature will automatically propose recipes based on the available and ingredients. It will be available via the SSH App and aims to reduce the time and effort students spend planning meals. Additionally, for recipes that require extra ingredients, the system will provide purchase suggestions to make it easier for students to plan what to buy. By implementing this extension, we hope to streamline student meal planning, aligning with our goal of offering smart, time-saving solutions that boost convenience in student homes. Furthermore, this new feature could also lead to reduced food waste, better budgeting, and more nutritious and balanced meals, all of which are important to the student users we serve.

Goals and non-goals

- **Goal:** Integrate a recipe database via the SSH Cloud and use ingredient data from the SSH Camera to generate recipe suggestions based on the available fridge contents, ensuring that recipe suggestions are updated in real time.
- **Goal:** Aggregate recipe results and display them on a dedicated recipe suggestions page within the SSH app. Set up a prioritisation method such that recipes that can be fully made with available ingredients are shown first, before partially achievable recipes and the choice to buy these missing items.
- **Goal:** Achieve a usage rate of at least 30% among students who access the SSH system and own an SSH Camera within the first year of release.
- **Non-goal:** Suggest recipes based on other factors, such as health goals and serving numbers, to encourage group cooking or meal prepping.
- **Non-goal:** Send mobile notifications to household members about food nearing their use-by date.

Design overview

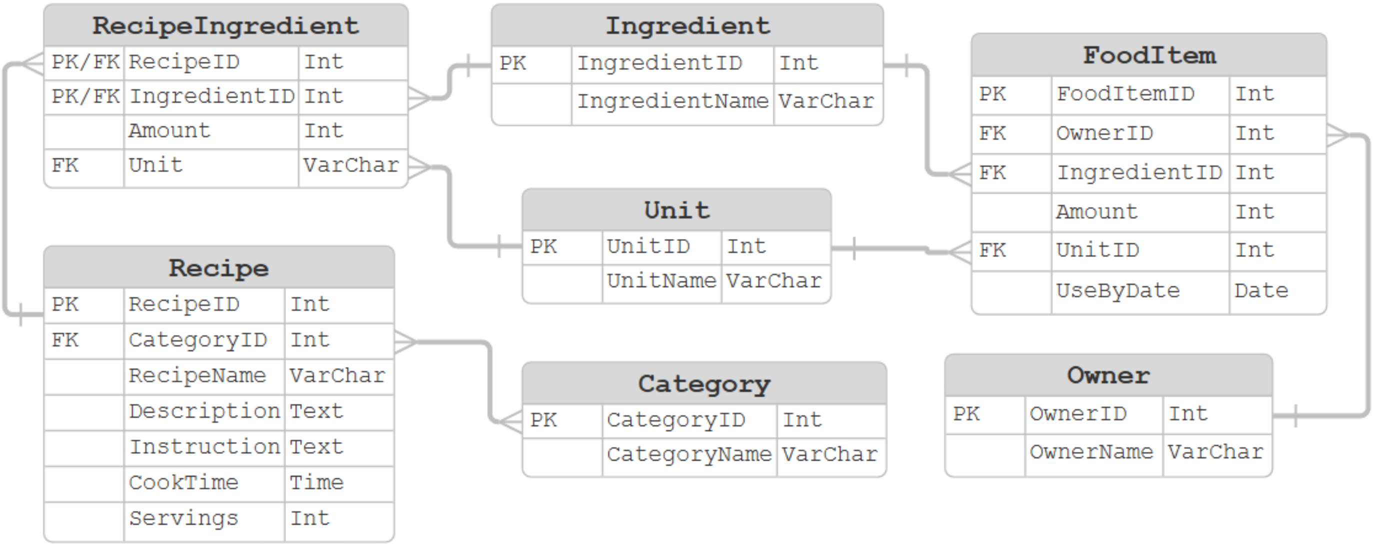
The new recipe suggestion function for the SSH Cloud will use data from the SSH Camera to analyse the contents of the refrigerator and match them with suitable recipes from a pre-existing recipe database. The key components of the system will include:

- **Ingredient Detection:** The SSH Camera will continue to check and log the items in the fridge and their quantities in real time. The data will be sent to the SSH Cloud to be pre-processed into a list of ingredients, and then stored.
- **Recipe Matching Algorithm:** A recipe database will be integrated into the system, with each recipe categorised by required ingredients, preparation time and dietary preferences (e.g., vegan, vegetarian, gluten-free). The backend algorithm will query this database to match the available ingredients with suitable recipes. For recipes that require extra ingredients, the system will find and suggest missing items and give the user the option to buy them.
- **User Interface:** The user will access the recipe suggestion function via a dedicated tab within the SSH App. The interface will consist of two main parts, the results page and a page to display the full recipe. The results page will display the list of recipes, sorted as described below (Ranking and Sorting), with options for filtering by dietary preferences or preparation time. Each recipe result will display basic details about the dish: the name, the preparation time, an icon to communicate its dietary category and an image of the completed dish. Additionally, each result will specify whether the recipe can be cooked with only the ingredients currently in the fridge, or whether more need to be bought to complete the meal. When a recipe is selected, the user will enter its dedicated page and will be presented with this information again in more detail, followed by the ingredients list and their quantities, followed by the full cooking instructions. There will also be an option to adjust the number of servings, and the system will scale quantities appropriately.
- **Ranking and Sorting:** Recipes will be ranked based on the number of ingredients that the user has available, such that fully achievable recipes will be prioritised and displayed first. Recipes that are partially matched will appear next, ranked in ascending order based on the number of missing ingredients. This system will help users to find the meals they can prepare with minimal shopping.

The SSH Camera gathers and caches ingredient data whenever it detects the fridge door opening. The recipe suggestion system will run a backend job that retrieves the list of available ingredients when the user accesses the recipe suggestion tab, and then query the recipe database. This approach will ensure recipe suggestions are responsive and based on the latest data. If real-time data is unavailable due to network issues or camera malfunction, the system will automatically revert to the most recent cached data for recipe suggestions.

Existing data

The following data, relevant to the new recipe suggestion function, is currently recorded and accessible via the SSH Cloud. The entity-relationship diagram below gives an overview of the primary data entities from pre-established databases and their relationships:



RecipeIngredient		
PK/FK	RecipeID	Int
PK/FK	IngredientID	Int
	Amount	Int
FK	Unit	VarChar

Ingredient		
PK	IngredientID	Int
	IngredientName	VarChar

FoodItem		
PK	FoodItemID	Int
FK	OwnerID	Int
FK	IngredientID	Int
	Amount	Int
FK	UnitID	Int
	UseByDate	Date

Unit		
PK	UnitID	Int
	UnitName	VarChar

Recipe		
PK	RecipeID	Int
FK	CategoryID	Int
	RecipeName	VarChar
	Description	Text
	Instruction	Text
	CookTime	Time
	Servings	Int

Category		
PK	CategoryID	Int
	CategoryName	VarChar

Owner		
PK	OwnerID	Int
	OwnerName	VarChar

- **Ingredient Inventory:** The SSH Camera already captures the fridge's contents, including each food item's name, owner, and quantity, so that it can perform functionalities such as billing household members who have taken food that does not belong to them. This information is stored in the `FoodItem` table, which references `Ingredient`, `Unit` and `Owner` tables for further details about each item.
- **Recipe Database**: A pre-existing recipe database includes details on recipe ingredients, instructions, preparation time, servings, and dietary categories. The `Recipe` table holds the metadata for each recipe and references the `RecipeIngredients` table, which connects each recipe with its required ingredients, quantity and unit.

Existing & new utility functions

For the backend job that will retrieve the list of available ingredients, we will need to request this information from the SSH Camera by following the pre-defined methods through the SSH Cloud, and this event will be triggered whenever the recipe suggestion function is opened by the user.

As described above, we already have existing representations of all the database entities. However, we will need to develop new utility functions to query the data from across multiple tables so that we can perform ingredient matching and recipe suggestion. To do this, we will create a view of the databases which joins the fridge contents and recipes on their ingredients, and then retrieve this list of recipes to be presented to the user.

Finally, we will need to design and implement the user interface that can display the records retrieved by our query as recipe suggestion results in the SSH App.

Alternatives

Alternative 1: Scheduled Ingredient Updates

Instead of retrieving ingredient data when a user accesses the recipe suggestion page, and, therefore, on-demand recipe matching, this alternative would use a scheduled backend job to update the available ingredients regularly.

Pros: Reduced load on the SSH Camera and SSH Cloud as it will perform data retrieval operations less often. Schedules updates also provide more consistent performance and faster load times for the user.

Cons: Potential to have inaccuracy between the actual ingredients available and the recipes suggested, giving a less personalized/responsive experience. For example, if a user places something in the fridge and then checks the app for suggestions, there could be lag time between the updates.

Alternative 2: External Recipe API

Instead of using a preloaded and local recipe database, this choice would take recipes from an external API that would give broader recipe options and variety.

Pros: Allows for recipes to be added, updated or improved over time without the need to manage this internally, and reduces the development time to maintain the recipe database.

Cons: Relying on an external API could give slower response times, and dependency on third-party services can be unreliable and lower control over users' data. Additionally, the formatting of recipes from the API could be more difficult to integrate with existing data from the SSH Camera due to inconsistencies.

Milestones

Milestone 1: Implement backend logic to aggregate and cache ingredient data recorded by the SSH Camera from the SSH Cloud and convert it into a usable ingredient list. This will allow us to evaluate the accuracy and performance of data aggregation. To verify that the system captures the required data accurately, we can compare the results to the fridge's contents.

Milestone 2: Implement the recipe matching algorithm that queries the database to find recipes that match available ingredients, as well as returning the results in the correct priority order. To test this, we should use the data gathered in Milestone 1 to ensure that the recipe results are both up to date, feasible and in the correct order.

Milestone 3: Implement the new recipe suggestion page to assess the how the algorithm in Milestone 2 performs on the SSH App using the data retrieved in Milestone 1 in an on-demand setting. This can be presented to current users with a temporary UI for get customer feedback. Depending on how the feature is received, work can be stopped, or changes can be made.

Milestone 4: Allow the design team to create a suitable UI design for the new recipe suggestions page. This can go through accessibility review and again be demonstrated on current users for feedback.

Milestone 5: Implement the purchase suggestion feature for missing ingredients. This should be integrated with any other purchasing systems in SSH products and can be released to current users via the newly developed user interface for testing.

Milestone 6 (Optional): Allow the notifications team to write and implement notifications for food items that are about to go out of date using data gathered by the SSH Camera.

Milestone 6 (Optional): Implement features into the user interface that allows the customer to filter their recipe results (e.g., by dietary category) to give a more customisable experience.

Dependencies

- *UI/Design Team:* Needed to design the user interface for the recipe suggestion results page and dedicated recipe page. Change existing SSH App views to link to the new feature.
- *Backend Development Team:* Will need to implement the backend logic to aggregate all the relevant data from the SSH Cloud and develop the recipe matching algorithm and help with the integration of the ingredient purchase suggestions feature.
- *Database Team:* Required to manage the integration of the pre-existing recipe database with the SSH Cloud, and write the queries needed to efficiently retrieve recipe results. Finally, they will also need to maintain the database views to ensure smooth performance.
- *Testing and Quality Assurance Team:* Needed to validate functionality, usability and accuracy across the milestones defined above, and ensure that the feature meets user expectations.

Cost

We predict a slight increase in operating costs as a result of the new recipe suggestion function proposed here. Maintaining the recipe database will require additional attention, and on-demand recipe matching could create large volumes of traffic if many users try to access the function at once, which contributes to performance costs. If it is found that this performs poorly, we could consider running the ingredient data retrieval using a scheduled back-end job instead.

Privacy and security concerns

All ingredient data used during the recipe suggestion is already recorded and stored by the SSH Camera, with the only new piece of collected information about users being their dietary preferences/category. Despite this, we must still be careful not to introduce new security vulnerabilities, especially those that might allow unauthorised access to the SSH Camera's view or data stored in the SSH Cloud, to protect our customers' privacy. To do this, we will follow all the appropriate and necessary pathways to access the camera data from the Cloud and make data handling practices transparent to users.

Risks

Risk	**Mitigation(s)**
Students will not want to make use of the recipe suggestion function	The new feature could be advertised on the SSH App and Cloud start pages. We could provide eye-catching graphics and videos explaining the new functionality and its benefits to encourage existing student users to test it out.
Viewing the recipe suggestion page and triggering the recipe matching algorithm will cause excessive load on the system and poor performance	We could implement the alternative (mentioned above) where the ingredient data is fetched from the SSH Cloud at regularly scheduled times.

Ingredient detection errors could lead to inaccurate recipe suggestion results	We could implement regular testing of the SSH Camera to ensure ingredients are found correctly. We could also implement a feature to allow users to manually adjust/correct items to help improve the system’s learning over time.
Students might find recipe suggestions to be outdated or repetitive after a while	Releasing regular updates to the recipe database with new content and incorporating seasonal recipes will help to maintain user interest. Also, we could allow user feedback on recipes to find out trends.

Supporting Materials