SSH Delivery Services

Engineering Design Review

Author: Ghuas Ali
Date: 17 October 2024
Status: Draft

Introduction

In many shared student living spaces, students may choose to use grocery delivery services for convenience, however this may be costly. To save on delivery fees, it's common practice for housemates to order their groceries together. However, this still has some issues. Students still face challenges coordinating timings of orders, tracking costs, and managing individual orders. Prices for products change regularly, and students need an easy way to view, share, and manage their grocery orders in real time. This is an opportunity for us at SSH to leverage our partnerships with supermarkets and utilise our existing ecosystem to address this problem.

I propose to add a software extension to the SSH Console Table and SSH App that will allow students to add items to a shared grocery order that they all can view. Connectivity to the SSH Cloud will allow us to relay the relevant information to the users such as insights into product availability, current prices and offers, and individual orders. Some of the proposed features include scheduling recurring orders, inventory management, budget tracking, and auto-splitting bills.

This software would not only make the process of ordering groceries more convenient but also help students manage their costs more effectively. This ultimately adds value to our ecosystem, improving the students' experience.

Goals and non-goals

- Goal: Allow students of shared living spaces to seamlessly coordinate and manage shared grocery delivery orders. Utilising the SSH ecosystem and partnerships with supermarkets.

- Goal: Implement automatic cost splitting for shared items, helping students manage their individual expenses.

-Goal: Provide real-time updates on product availability, prices and deals to help students manage expenses.

- Non-goal: Not intended to be its own delivery service but rather a software implementation

which utilises partnerships with existing supermarkets and existing delivery service providers.
- **Non-goal:** Allow integration of multiple supermarkets into one order. Each order will be confined to one supermarket.

Design overview

The proposed platform will include 2 user interfaces: a mobile app and an SSH Console Table interface. This will allow students living together to collaboratively manage grocery orders but also individually review and budget on the app. Both interfaces will connect to the same backend and the system will offer live synchronisation (using WebSocket connections) between these interfaces, ensuring that users can view and edit their shared grocery lists seamlessly across devices.

Interfaces

Mobile app interface:

Designed for more individual and remote needs. The app interface connects the student to the same cloud backend as the SSH Console Table. Students can manage grocery lists directly from their phone. Features include adding and removing items, tracking orders, and monitoring costs and individual budget. The mobile app plays a crucial role in order finalization/acceptance.

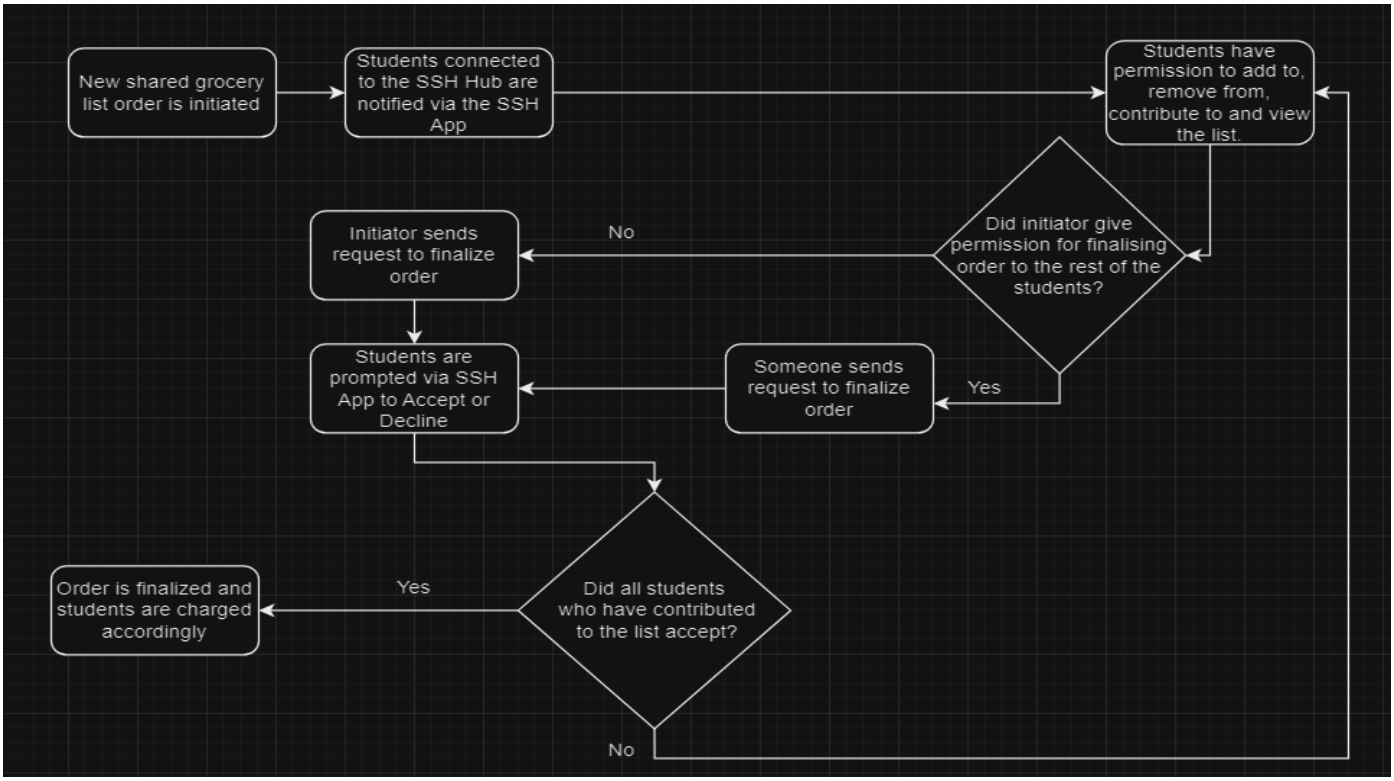
SSH Console Table interface:

Designed for group collaboration. Students can interact and manage shared lists in a group setting discussing with one another and assessing total and individual contribution costs.

Order initiation and finalisation

Students can initiate a shared grocery list from either interface. This notifies anyone connected to the same SSH Hub, informing them of the chosen supermarket, expected finalisation and delivery date, and product details.

Once the grocery list is completed, each student who added or contributed to the order will be prompted to accept the order so that it can be finalised. The relevant information is sent to the supermarket to fulfil the order, and students receive a confirmation and summary on the app.



Key points to consider:

- Initiating finalisation: The user who initiated the order has the ability to send a request to finalise the order. This user also has the ability to give this permission to other users, allowing multiple users to send the request once they have tweaked their share of the order to their liking.

- Review of Items and Costs: Each student can review the individual items they've added, and the items that they jointly contrubted to, along with the associated costs. They must either accept their items or make changes.

- Confirmation Deadline: Users have a predefined limited time to confirm. If a user fails to respond, the system will automatically confirm their participation.

Alternatives

Decentralized order finalisation

Instead of having a single user or selected users responsible for finalising the order, the system would automatically finalise the order once all users had added and confirmed their items.

Pro: Reduces potential delays as the order is automatically processed after students place an item and confirm it/ lock it in.

Pro: Removes the need to have a singular person responsible for order finalisation or selecting other users to take the responsibility.

Con: Removes the final group review which could lead to miscommunication and errors.

Con: Less flexibility for students who change their mind last minute.

Post finalisation changes

Allow students to modify the order up until the supermarket starts processing the order. Students would be able to make changes after the initial finalisation.

Pro: Adds an extra layer of flexibility for users to adjust orders if they forget an item or want to make last-minute changes.

Pro: Reduces the pressure of finalising quickly.

Con: Implementation is more complex

Con: Could lead to confusion

Multiple supermarkets integrated into one order

Allow students to place different items from different partnered supermarkets into one order.

Pro: Adds more variety and availability.

Pro: Students could save money by comparing prices of the same product in different supermarkets.

Con: Adds complexity to delivery logistics.

Con: May cost more as different supermarkets may have individual delivery fees.

Con: Deliveries may arrive at different times which could be inconvenient for students to manage.

Milestones

Milestone 1: Design the UI for the Mobile App and the SSH Console Table interface based on student needs for collaborative grocery shopping. Review the design to ensure that it meets students' expectations.

Milestone 2: Build the mobile app with core functionality such as, grocery list creation, item management, cost-splitting, budget tracking and order finalisation. Collect feedback from students to further refine this.

Milestone 3: Build the SSH Console Table interface with core functionality and a focus on group collaboration. Collect feedback from students regarding ease of collaboration and user experience to refine this.

Milestone 4: Implement the backend to support data storage, user management, and real time synchronisation between the SSH Mobile app and the SSH Console Table interface. Integrate with supermarket APIs to receive live product data. Main focus of testing will be on data synchronisation and accuracy of the APIs.

Milestone 5: Implement order finalisation workflow, enabling users to confirm their items and view cost breakdowns before submission. Include individual confirmation prompts, budget alerts, and cost-splitting options. Collect feedback from students regarding the smoothness of the experience to further refine the process.

Milestone 6 (optional): Work on implementing new features and customisability based on user feedback.

Dependencies

UI team: Will need to design the UIs for the SSH Mobile App and the SSH Console Table interface.

SSH App developing team: Will need to build on the already existing app to implement the new proposed systems and design. Focus on individual student experience.

SSH Console Table team: Will need to build on the already existing interface to implement the new proposed systems and design. Focus on group collaboration experience.

Notifications team: Will need to design and manage alert system.

Cybersecurity team: Will need to ensure secure authentication protocols are being used and data is secure.

Legal team: Will need to review legalities of the agreement between the customers to ensure that the new feature does not expose us to liabilities.

Student feedback and testing team: Will need to collect feedback, analyse consumer data, and test existing features and optimisations.

Cost

We do not expect any meaningful increase in operating costs. There will be no new hardware introduced. The real time synchronisation between the SSH App and Console table will only be active if there is an active grocery list.

Privacy and security concerns

All data handled by the platform such as user profiles, grocery lists, cost-sharing information, etc, is only accessible by authorised students within the household. Strict authentication protocols are necassary along with established security protocols (e.g. encryption of data via WebSocket) when implementing real-time synchronisation, Supermarket APIs and databases. Individual data, such as budgets, is only available to the individual authorised student.

Risks

Risks	Mitigation(s)
Data Breach or Unauthorized Access.	Use secure protocols and robust authentication processes. Regularly review access logs and flag any suspicious activity.
Platform is dependent on supermarket APIs. Downtime of the APIs can cause issues with outdated information.	Notify users if the live connection with the API is experiencing issues, that way they can make an informed decision and potentially order from another partner supermarket that isn't currently down.
Real-time synchronisation failure between app and console table which can disrupt the collaborative aspect as there may be conflicting data.	Notify users if there is an issue with the WebSocket connection so that they can make an informed decision to potentially wait to reestablish connection or to order from the console table as a group.

User mismanagement or miscommunication in finalisation	Implement a well-structured finalization process with clear, customizable notifications for all users involved in an order.

Supporting material

Intentionally empty