Who Did What:

Yumiko and Safeeyah will continue to focus on the database and backend development. Ameera and Beleny will concentrate on the front-end development. Adissem will ensure code consistency and cleanliness, assist with research, and make sure that the frontend and backend code integrate smoothly. As always, everyone is available to support one another as needed throughout the project.

Updated Implementation Plan:

The implementation plan, for the most part, remains the same. Safeeyah finished up doing the Customization section this week and began working on the Chores section. Yumiko finished the Account section herself, and Ameera did the UI for Account herself. We are ahead of schedule. We are also including Thanksgiving break to finish up the project and polish, and have Week 10 set up for Presentation Practice. We decided to remove the expense history from our ER diagram as we realize they are very similar and can be combined. Additionally we changed our Highest Priority Non MVP from Messages to Expenses as we realized that Expenses would be more important to have.

Week 7: (Midterm week for everyone - Less work)

Safeeyah & Yumiko: ER Diagram

Yumiko: Finish setting up the database

Yumiko: Account (1.1 – 1.1.3)

Safeeyah: Group/Household Management (5.1 – 5.2.1)

Ameera: Work on the general front-end and add UI to functionalities as they are coded.

Adissem: Review code for consistency and cleanliness, assist with research, and support the backend team as needed with functionality

All: Include Error Recovery (7.1-7.2) as you code.

Week 8:

Yumiko: Account (1.2 – 1.2.3)

Safeeyah: Customization (6.1-6.1.2)

Yumiko: Begin Compatibility IF TIME ALLOWS (4.1 – 4.2.1)

Yumiko: Expenses (Highest Priority: Non-MVP)

Ameera and Beleny: Work on the general front-end and add UI to functionalities as they are coded.

Adissem: Review code for consistency and cleanliness, assist with research, and support the backend team as needed with functionality

Adissem & Beleny: Chore/Expense Activity Diagram, Household Activity Diagram, Class/Block Diagram

All: Include Error Recovery (7.1-7.2) as you code.

Week 9:

Yumiko: Finish Compatibility IF TIME ALLOWS (5.1 – 5.2.1)

Yumiko: Statuses (High Priority: Non-MVP)

Safeeyah: Scheduling (2.1)

Safeeyah: Chores(3.1 – 3.1.9)

Safeeyah and Yumiko: Error Recovery (7.1-7.2)

Ameera and Beleny: Work on the general front-end and add UI to functionalities as they are coded.

Adissem: Review code for consistency and cleanliness, assist with research, and support the backend team as needed with functionality

All: Start to finish and polish. Include Error Recovery (7.1-7.2) as you code.

Thanksgiving Break:

Safeeyah & Yumiko: Finish backend and database work and polish

Ameera and Beleny: Finish front-end and polish

Adissem: Review code for consistency and cleanliness, assist with research, and support the backend team as needed with functionality. Also, polish Error Recovery

Week 10:

Practice Presentation

Highest Priority (NON-MVP):

Expenses

- The system shall allow users to log financial transactions.
- Each transaction shall record the total amount, payer, date, description, and selected payment split method.
- The system shall display all transactions in a shared ledger view accessible to every group member.

High Priority (NON-MVP):

- The system shall allow users to set and view custom statuses.
    - Status options shall include “Available,” “Busy,” and “Do Not Disturb.”

Minimum Viable Product (MVP)

System Requirements:

1. **Account:**

1.1. The system shall allow users to create an account.

1.1.1. The system shall validate email format before submission.

1.1.2. The system shall verify that the user’s email domain matches a

registered university domain.

1.1.3. The system shall notify users of successful registration or display error messages for duplicate or invalid emails.

1.2. The system shall present a compatibility questionnaire.

1.2.1. The questionnaire shall include both multiple-choice and scaled (1–5) questions.

1.2.2. The system shall allow users to review and change responses before final submission.

1.2.3. The system shall provide visual feedback for unanswered or invalid responses.

2. Scheduling:

2.1. The system shall allow users to view a shared household cleaning schedule to do list..

3. Chores

3.1. The system shall allow users to create and manage household chores.

3.1.1. Each chore shall include a title, description, assigned member(s), due date, and optional notes.

3.1.2. Users shall be able to select from multiple assignment types when creating a chore:

a. Single Assignment: One specific roommate is permanently responsible for the task (e.g., vacuuming).

b. Rotating Assignment: The responsibility cycles among roommates on a set schedule (e.g., weekly trash duty).

c. Shared Assignment: Multiple members work collaboratively on the same task (e.g., deep cleaning the kitchen).

d. Voluntary Assignment: The chore remains open for any roommate to claim at any time.

3.1.3. When selecting a rotating assignment, the system shall prompt the user to choose a rotation frequency:

a. Daily

b. Weekly

c. Biweekly

d. Monthly

e. Custom interval (user-defined in days or weeks).

3.1.4. The system shall record the order of participants in a rotation and automatically advance the assignment at the end of each cycle.

3.1.5. Users shall be able to view and manually edit the rotation order at any time.

3.1.6. Each chore shall include an optional due time in addition to a due date (e.g., take out trash on Monday before 8 AM, for the garbage truck)

3.1.7. Users shall be able to mark chores as “complete,” “in progress,” or “skipped.”

3.1.8. The system shall visually differentiate between completed, pending, skipped, and overdue tasks using color or icons.

3.1.9. Users shall be able to manually edit chore performers in the case of inability to do a chore

3.2. The system shall allow users to view all scheduled chores.

3.2.1. The chores shall display in a list view.

4. Compatibility

4.1. The system shall calculate roommate compatibility scores.

4.1.1. Compatibility shall be based on questionnaire response similarity.

4.1.2. Matching weights shall emphasize lifestyle factors (cleanliness, bedtime, noise).

4.1.3. The algorithm shall return a percentage score for each match.

4.2. The system shall display a list of compatible matches.

4.2.1. Each match card shall display the user's name, age, and compatibility score.

5. Group/Household Management

5.1. The system shall allow users to create or join roommate groups.

5.1.1. Users shall name their group and assign a description.

5.1.2. The system shall generate a unique invitation to send through messages or email for inviting roommates.

5.1.3. Invitees shall receive a notification with a join prompt.

5.2. The system shall allow users to manage group members.

5.2.1. The group users shall have admin permissions to add or remove members.

6. Customization

6.1. The system shall allow users to personalize a roommate agreement.

6.1.1. Agreement sections shall include cleanliness, guest policy, noise level, and shared expenses.

6.1.2. The system shall highlight differences between roommates' responses.

7. Error Recovery

7.1. The system shall allow users to undo their most recent change in forms or settings.edit

7.2. In the case of an irreversible action, the system shall prompt the user for confirmation before proceeding.