Verification and Validation Report: Housemates

Team #9, Housemates
Justin Dang - dangj15
Harris Hamid - hamidh1
Fady Morcos - morocof2
Rizwan Ahsan - ahsanm7
Sheikh Afsar - afsars

April 4, 2024

1 Revision History

Date	**Version**	**Notes**
March 6	1.0	Version for Rev 0
April 4	2.0	Version for Rev 1

2 Symbols, Abbreviations and Acronyms

symbol	description
API	Application Programming Interface
F	Fail
P	Pass

Contents

1 Revision History i

2 Symbols, Abbreviations and Acronyms ii

3 Purpose 1

4 Functional Requirements Evaluation 1
- 4.1 System Tests . . . 1
  - 4.1.1 Bill Management . . . 2
  - 4.1.2 Task Management . . . 3
  - 4.1.3 Scheduling System . . . 3
  - 4.1.4 Account System . . . 4
- 4.2 Unit Testing . . . 4
  - 4.2.1 Automated Testing . . . 4
  - 4.2.2 Bill Management . . . 5
  - 4.2.3 Task Management . . . 6
  - 4.2.4 Scheduling . . . 7
  - 4.2.5 Account . . . 8

5 Nonfunctional Requirements Evaluation 8
- 5.1 Usability . . . 8
- 5.2 Performance . . . 12
- 5.3 Other Non-functional Tests . . . 14

6 Conclusions based off VnV Data 16

7 Changes Due to Testing 16
- 7.1 Planned Changes due to Revision 0 Feedback . . . 16
- 7.2 Planned Changes due to Usability Feedback . . . 17

8 Trace to Requirements 18

9 Trace to Modules 20

List of Tables

1 Bill Management System Tests . . . 2

2 **Task Management System Tests** 3
3 **Scheduling System Tests** 3
4 **Account System Tests** 4
5 **Bill Management Unit Tests** 5
6 **Task Management Unit Tests** 6
7 **Scheduling Management Unit Tests** 7
8 **Account Management Unit Tests** 8
9 Non-Functional System Tests 15
10 **Functional Requirements Traceability Part 1** 18
11 **Functional Requirements Traceability Part 2** 18
12 **Non-Functional Requirements Traceability Part 1** . . . 19
13 **Non-Functional Requirements Traceability Part 2** . . . 19
14 **Module Traceability Part 1** 20
15 **Module Traceability Part 2** 21

List of Figures

1 Ease of locating main features 9
2 Clear instructions 10
3 Bugs encountered 10
4 App feedback to user errors 11
5 Performance of app 11
6 Visual appeal 12
7 Average Response Times of Housemates API 13

3 Purpose

This document describes the results of verification and validation plan that was created earlier on in the software lifecycle of Housemates. The purpose of this verification and validation is

1. Ensure that the requirements listed in the SRS for the Housemates application are correct.
2. Ensure that the design of the Housemates application satisfies the requirements listed in the SRS.
3. Ensure the usability of the Housemates application for its prospective users.

To find the the details of each test look at the VnV Plan and search for the associated test id.

4 Functional Requirements Evaluation

4.1 System Tests

The details of the system tests can be found in section 3.1 of the VnV Plan.

4.1.1 Bill Management

Test ID	P	F	Test
test-BM1-1	×		Verify bill creation and equal splitting among users.
test-BM2-1	×		Ensure editing of an existing bill is possible.
test-BM3-1	×		Check if bills can be categorized (e.g., food, utilities).
test-BM4-1	×		Validate viewing past bills and their payment statuses.
test-BM5-1	×		Confirm uneven bill splitting functionality.
test-BM6-1	×		Verify functionality for settling a bill.

Table 1: **Bill Management System Tests**

All tests related to the Bill Management system have passed, indicating that users can create, edit, categorize, view, and settle bills without issues, and the split function operates correctly for both even and uneven divisions.

4.1.2 Task Management

Test ID	P	F	Test
test-TM1-1	×		Verify navigation to Task Management page via tasks icon.
test-TM2-1	×		Ensure that the Task Management page displays a list of tasks with user's chores highlighted.
test-TM3-1	×		Confirm the creation of a new task.
test-TM4-1	×		Validate assigning a task to a roommate.
test-TM5-1	×		Check marking a task as completed updates its status appropriately.

Table 2: **Task Management System Tests**

All tests related to the Task Management system have passed, confirming that users can create, assign, view, and complete tasks with ease and accuracy.

4.1.3 Scheduling System

Test ID	P	F	Test
test-SS1-1	×		Ensure users can navigate to the scheduling system page.
test-SS2-1	×		Confirm users can create and schedule new events.
test-SS3-1	×		Verify users can view all scheduled events on the calendar.

Table 3: **Scheduling System Tests**

The tests for the Scheduling System demonstrate that the application's scheduling functionalities operate as intended, allowing users to create, view, and manage events effectively.

4.1.4 Account System

Test ID	P	F	Test
test-AS1-1	×		Validate account creation with unique credentials.
test-AS1-2	×		Ensure no duplicate accounts can be created.
test-AS1-3	×		Confirm account creation adheres to specified username and password constraints.
test-AS2	×		Verify that users can log in with valid credentials.
test-AS3	×		Check that users can update profile details.
test-AS4	×		Ensure that users can delete their account.
test-AS5	×		Verify account recovery for forgotten login information.

Table 4: **Account System Tests**

All Account System tests have passed, ensuring that the account management features such as creation, updating, and deletion are functioning correctly, providing users with secure and reliable account handling capabilities.

4.2 Unit Testing

The details of the unit tests can be found in section 4 of the VnV Plan.

4.2.1 Automated Testing

Automated testing was conducted using jest and GitHub actions. Automated testing mainly covers the unit testing of Housemates and is performed automatically whenever a commit is added to the main branch. The purpose of automated testing of Housemates is to ensure that any changes made to the main branch do not break the overall functionality of Housemates.

4.2.2 Bill Management

Test ID	P	F	Test
UT-B1	×		Split expense and create bill.
UT-B2	×		Split expense with equal amounts.
UT-B3	×		Split expense with custom amounts.
UT-B4	×		Error for not found user.
UT-B5	×		Error for not found group.
UT-B6	×		Handle invalid bill ID format.
UT-B7	×		Handle null bill ID.
UT-B8	×		Handle undefined bill ID.
UT-B9	×		Handle empty string bill ID.
UT-B10	×		Delete an existing bill.
UT-B11	×		Handle deleting non-existent bill.
UT-B12	×		Get expenses for user.
UT-B13	×		Handle getting expenses for non-existent user.
UT-B14	×		Edit bill total amount.
UT-B15	×		Error editing non-existent bill.

Table 5: **Bill Management Unit Tests**

All unit tests for bill management module passed as expected. This was designed to ensure related basic functions of bill are operational.

4.2.3 Task Management

Test ID	P	F	Test
UT-T1	×		Create a new task.
UT-T2	×		Retrieve a list of tasks.
UT-T3	×		Mark a task as completed.
UT-T4	×		Reopen a completed task.
UT-T5	×		Edit details of a task.

Table 6: **Task Management Unit Tests**

All unit tests for task management module passed as expected. This was designed to ensure related basic functions of assigning tasks and organizing them are operational.

4.2.4 Scheduling

Test ID	P	F	Test
UT-S1	×		Add a new event.
UT-S2	×		Edit an existing event.
UT-S3	×		Get events for a group.
UT-S4	×		Get events for a user.
UT-S5	×		Delete an event.
UT-S6	×		Error for adding event to non-existent group.
UT-S7	×		Error for editing non-existent event.
UT-S8	×		Error for deleting non-existent event.
UT-S9	×		Error for getting events for non-existent group.
UT-S10	×		Error for getting events for a group with no events.

Table 7: **Scheduling Management Unit Tests**

All unit tests for scheduling module passed as expected. This was designed to ensure related basic functions of scheduling are operational. For example scheduling events and viewing them,

4.2.5 Account

Test ID	P	F	Test
UT-A1	×		Get user by built-in ID.
UT-A2	×		Get user by userID.
UT-A3	×		Get group details.
UT-A4	×		Get user groups.
UT-A5	×		Error for user groups retrieval.
UT-A6	×		Error for user retrieval.
UT-A7	×		Error for user by built-in ID.
UT-A8	×		Error for group retrieval.

Table 8: **Account Management Unit Tests**

All unit tests for account module passed as expected.

5 Nonfunctional Requirements Evaluation

5.1 Usability

The usability of the app was evaluated by conducting a survey with five participants who were asked to complete basic tasks within the app and provide feedback on their experience. This survey can be found in section 5.2 of the VnV Plan. The survey consisted of 13 questions, including multiple-choice, rating scale, and open-ended questions. The main findings from the survey are summarized below along with some key data quantified:

- User Profile: All the participants were students who lived with roommates or housemates and often split their bills with other people.
- First Impression: None of the participants found the app to be visually appealing, and most of them described it as bland, generic, or unpolished. They also had difficulty locating the main features or functions of the app.

- Performance: The participants were generally satisfied with the speed and performance of the app, but some of them encountered errors or bugs while using it, such as not being able to create a user or join a group.

- Feedback and Guidance: The app did not provide helpful feedback or guidance when the participants made a mistake or encountered an issue. Most of them did not know what to do when they faced a problem and had to refresh the page or try again.

- Difficulty: The participants found some features or functions to be particularly difficult to use or understand, such as joining a group, splitting an expense, or paying off a bill. They also complained about the amount of typing required and the lack of information or options available.

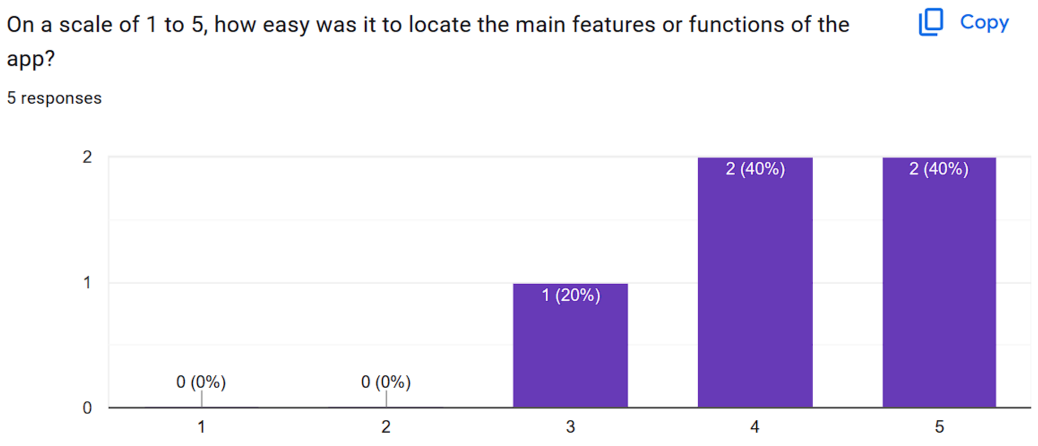


Figure 1: Ease of locating main features

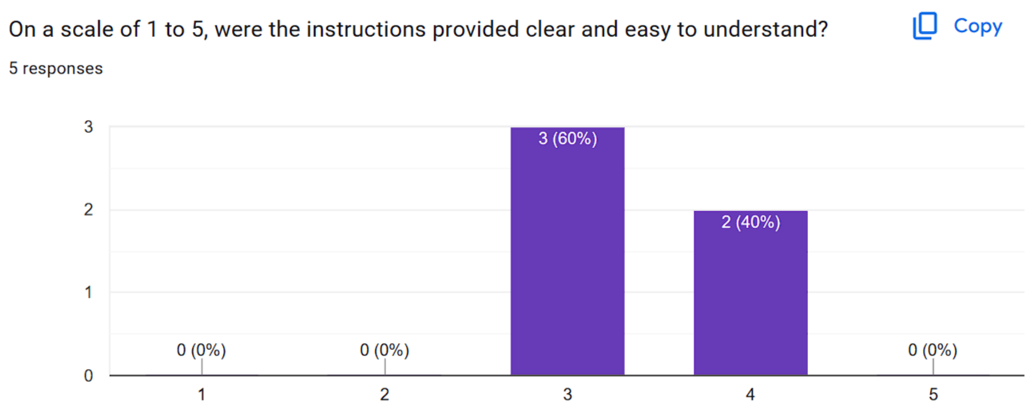


Figure 2: Clear instructions

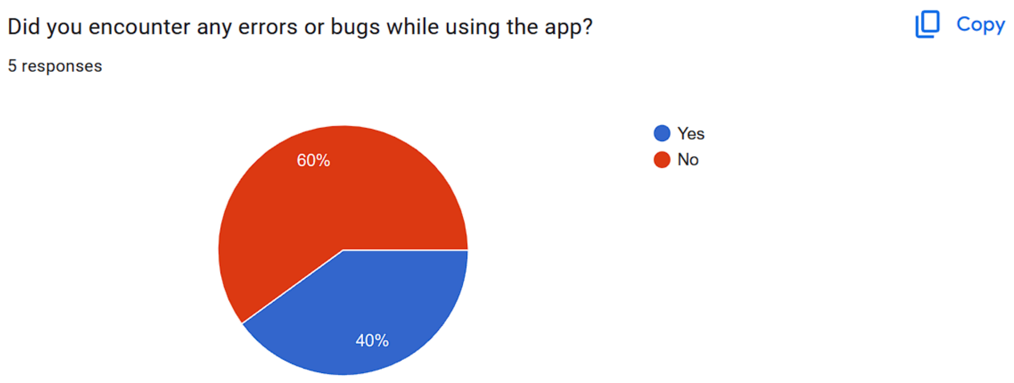
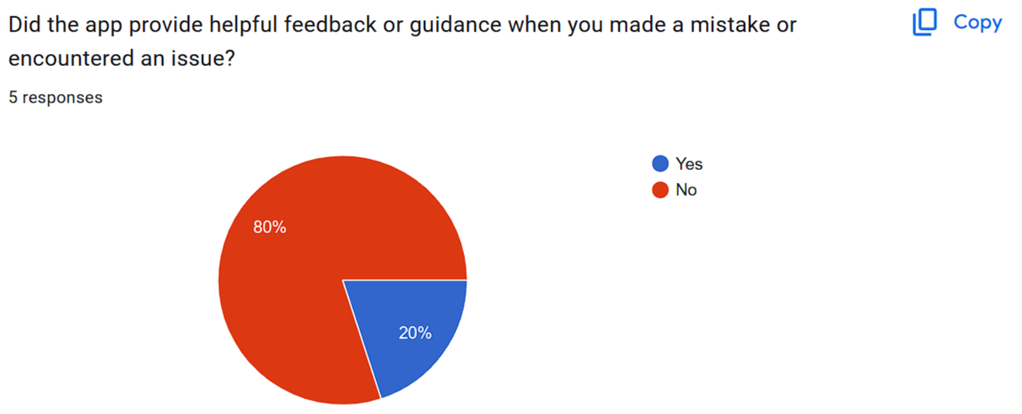
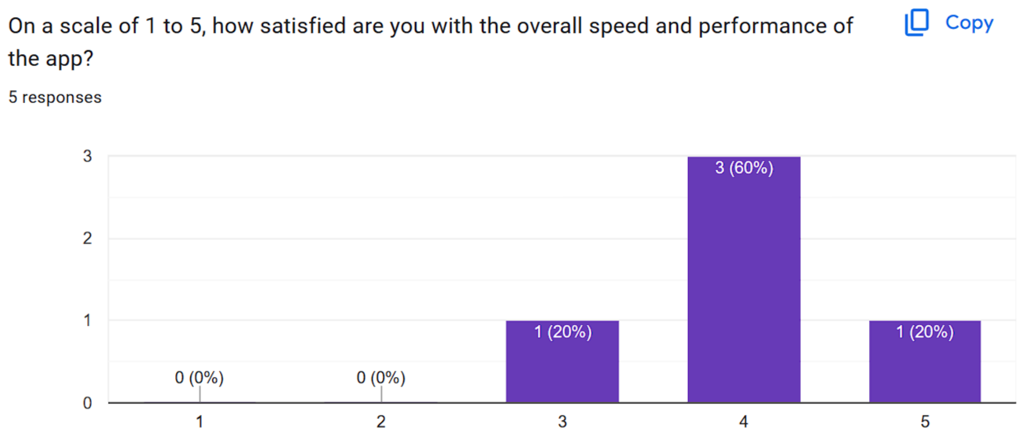


Figure 3: Bugs encountered


Did the app provide helpful feedback or guidance when you made a mistake or encountered an issue?

5 responses

Figure 4: App feedback to user errors


On a scale of 1 to 5, how satisfied are you with the overall speed and performance of the app?

5 responses

Figure 5: Performance of app

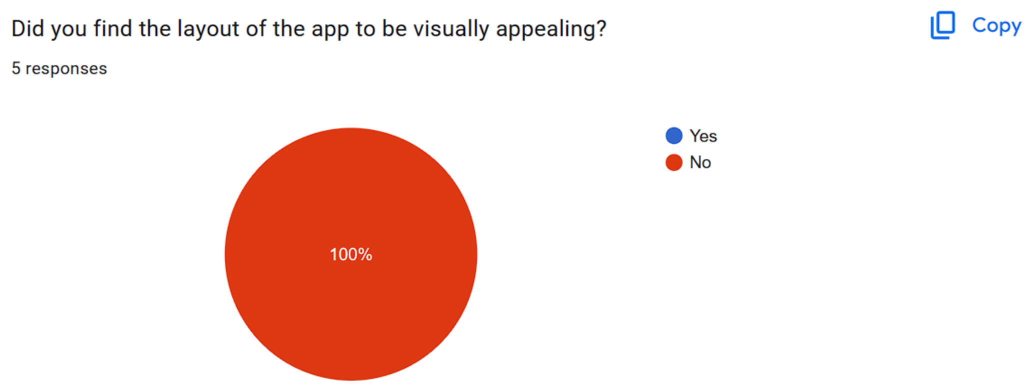


Figure 6: Visual appeal

5.2 Performance

Performance testing for Housemates was focused testing the back-end server of the application. This was done using “Jmeter”, which is a load testing tool that helps analyze the performance of the backend API of Housemates. The test plan involved creating 100 users for Housemates and having them make common requests to the backend server API (e.g getting user information after logging in). The results of performance testing can be seen in the graph below.

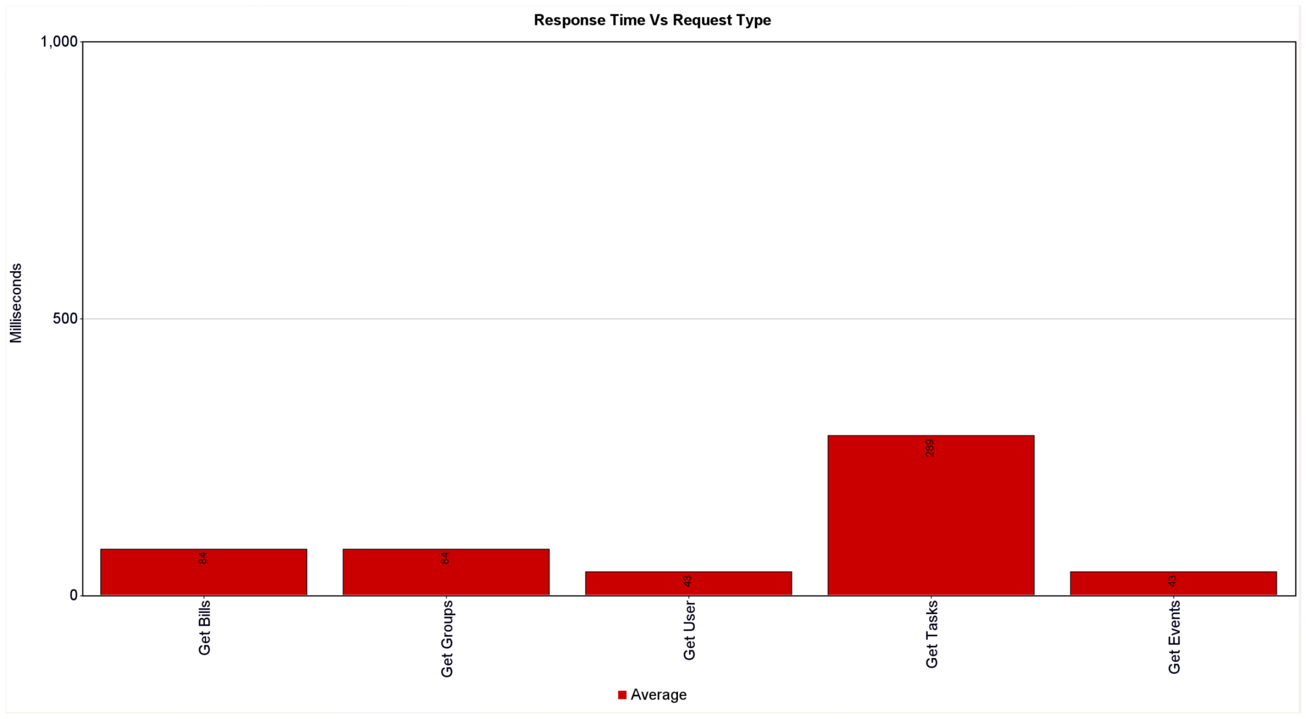


Figure 7: Average Response Times of Housemates API

As can be seen through the above graph the response times to user requests is quite fast ranging from around 40 - 300 ms. This is fast enough so that there should not be noticeable delay in Housemates on the user end. As such this indicates to us that Housemates can handle 100 concurrent users and as a result we concluded that the performance of Housemates is sufficient for our current purposes. In the future if necessary the database of Housemates could be upgraded in order to handle more users.

Some ways we could further scale up the application in the future could be:

- **Optimizing the Database:** Add some indexes on the database to speed up frequently used operation like range or equality search. For example by having an range index on the events table we can get even faster data retrievals which would be very beneficial for large databases (very busy schedules). This potential solution can speed up the user experience significantly.

- **Caching :** Integrating some caching methods for users. This can also greatly speed up the data base as user will not need to reload data.
- **Server Scaling:** Once number of users increases we can implement more servers in order to more quickly handle the greater load.

5.3 Other Non-functional Tests

The details of these non-functional tests can be found in section 3.2 of the VnV Plan.

Test ID	P	F	Description
test-LF-A1-1		×	Test application for conformance to material design guidelines.
test-LF-A1-2		×	Assess visual appeal and intuitive design.
test-UH-E1-1	×		Verify ease of navigation to key features within the app.
test-UH-E1-2	×		Confirm the application is easy to navigate according to users.
test-UH-P1	×		Ensure the app interface is presented in Canadian English.
test-UH-L1-1		×	Check if users can use a feature effectively without external assistance after a brief tutorial.
test-UH-L1-2		×	Survey to determine if the app is easy to learn.
test-UH-A1	×		Confirm the use of Android accessibility features.
test-P-SL1	×		Measure response times and confirm they are within acceptable limits.
test-P-PA1	×		Ensure bill-splitting calculations are accurate to two decimal places.
test-P-RFT1		×	Test resilience of features upon losing network connection.
test-P-C1	×		Evaluate system performance under a simulated load of multiple users.
test-OE-PE1	×		Verify the app runs on multiple phones with different Android OS versions.
test-OE-PR1		×	Check availability of the app on the Google Play Store.
test-M-M1	×		Confirm the presence of documentation on GitHub.
test-S-A1	×		Ensure user data is not accessible from other accounts.
test-S-IN1	×		Validate prevention of incorrect data entry into the database.
test-S-P1	×		Confirm display of the information collection policy on app launch.
test-C-SC1	×		Confirm the app's compliance with Google Play development standards.

Table 9: Non-Functional System Tests

For the non-functional system tests most passed, but some did not. test-OE-PR1 failed because Housemates isn't available on Google Play Store, test-P-RFT-1 failed because Housemates doesn't work well offline yet, testLF-A1-1, test-LF-A1-2, test-UH-L1-1, and test-UH-L1-2 failed because of reasons described in usability section of this VnV report.

6 Conclusions based off VnV Data

The main area of improvement for Housemates is in the usability department. As evidenced from the data from the usability survey most users are not satisfied with the current state of the Housemates application. The non-functional tests that failed also mainly have to do with the usability of Housemates. As such, the main focus for revision 1 of Housemates will be on improving the usability of Housemates. The changes that we plan to implement with revision 1 of the Housemates application are covered in the next section.

7 Changes Due to Testing

7.1 Planned Changes due to Revision 0 Feedback

For the Revision 0 demo, the main piece of feedback that we received was that the user interface of the Housemates felt very unpolished (e.g. dollar amounts not being rounded correctly, requiring a lot of typing which is undesirable for a mobile application) and that users were unlikely to use Housemates in this state. To help address these issues we plan to improve the UI of all the main features of Housemates (Bill Management, Task Management, Scheduling) by presenting the user information in a card layout, which is more user-digestible. This new UI will also require less typing (detailed plans in the next section), which should make it more efficient to the end user. Additionally, some features were suggested such as having presets for tasks and reoccurring tasks that we plan to implement in revision 1 of Housemates.
Plans to fix common problems/issues:

- **Incentivize completing tasks and paying bills** Add a reward system that rewards users for staying on track with paying bills and completing tasks. Users can be able to have a streak and be awarded according to a star system. Users may be award different statuses like "good roommate", etc.

- **Incentivize staying truthful to marking tasks complete and payed bills** Implement a photo system where users can upload photos and other roomates can check them. Also, implement a reporting system that will tarnish a users account.

7.2 Planned Changes due to Usability Feedback

One of the major feedback we got was regarding the design and layout of the application. Comments included phrases like "bland", "generic" and "not eye catching". So for revision 1 we plan to improve the current layout by introducing a more desirable color scheme rather than being black and white. Another pain point described by the user was that the application did not provide feedback if a mistake was made by the user. To remedy this problem, instead of providing feedback to the problem, we will reduce the chance of the user going into that state by introducing more guards in our application. Plans to fix/address usability issues:

- **Plans to implement less typing/manual entry:**
  - Have preset labelled buttons for certain tasks, events and bills that are quick, easy and attractive to see and use.
  - Add drop downs and scroll wheels for data and time setting when creating events.
  - quick one-click buttons for predicted bill prices to be split
- **Plans to address bland UI** Add attractive color schemes that attract users to the app.

8 Trace to Requirements

Test ID	TM1	TM2	TM3	TM4	TM5	BM1	BM2	BM3	BM4	BM5
test-TM1-1	×									
test-TM2-1		×								
test-TM3-1			×							
test-TM4-1				×						
test-TM5-1					×					
test-BM1-1						×	×			
test-BM2-1								×		
test-BM3-1									×	
test-BM4-1										×

Table 10: **Functional Requirements Traceability Part 1**

Test ID	BM6	BM7	SS1	SS2	SS3	AS1	AS2	AS3	AS4	AS5
test-BM5-1	×									
test-BM6-1		×								
test-SS1-1			×							
test-SS2-1				×						
test-SS3-1					×					
test-AS-1-1						×				
test-AS-1-2						×				
test-AS-1-3						×				
test-AS-2							×			
test-AS-3								×		
test-AS-4									×	
test-AS-5										×

Table 11: **Functional Requirements Traceability Part 2**

Test ID	LF-A1	UH-E1	UH-P1	UH-L1	UH-A1	P-SL1	P-PA1	P-RFT1
test-LF-A1-1	×							
test-LF-A1-2	×							
test-UH-E1-1		×						
test-UH-E1-2		×						
test-UH-P1			×					
test-UH-L1-1				×				
test-UH-L1-2				×				
test-UH-A1					×			
test-P-SL1						×		
test-P-PA1							×	
test-P-RFT1								×

Table 12: **Non-Functional Requirements Traceability Part 1**

NFR ID	P-C1	OE-PE1	OE-PR1	M-M1	S-A1	S-IN1	S-P1	C-SC1
test-P-C1	×							
test-OE-PE1		×						
test-OE-PR1			×					
test-M-M1				×				
test-S-A1					×			
test-S-IN1						×		
test-S-P1							×	
test-C-SC1								×

Table 13: **Non-Functional Requirements Traceability Part 2**

9 Trace to Modules

Module	Task	Bill	Scheduling	Account	Interface	Database	Network	Crypto
test-TM1-1	×				×	×	×	
test-TM2-1	×				×	×	×	
test-TM3-1	×				×	×	×	
test-TM4-1	×				×	×	×	
test-TM5-1	×				×	×	×	
test-BM1-1		×			×	×	×	
test-BM2-1		×			×	×	×	
test-BM3-1		×			×	×	×	
test-BM4-1		×			×	×	×	
test-BM5-1		×			×	×	×	
test-BM6-1		×			×	×	×	
test-SS1-1			×		×	×	×	
test-SS2-1			×		×	×	×	
test-SS3-1			×		×	×	×	
test-AS-1-1				×	×	×	×	×
test-AS-1-2				×	×	×	×	×
test-AS-1-3				×	×	×	×	×
test-AS-2				×	×	×	×	×
test-AS-3				×	×	×	×	
test-AS-4				×	×	×	×	
test-AS-5				×	×	×	×	

Table 14: **Module Traceability Part 1**

Module	Task	Bill	Scheduling	Account	Interface	Database	Network	Crypto
test-LF-A1-1					×			
test-LF-A1-2					×			
test-UH-E1-1					×			
test-UH-E1-2					×			
test-UH-P1					×			
test-UH-L1-1	×	×	×	×	×			
test-UH-L1-2	×	×	×	×	×			
test-UH-A1					×			
test-P-SL1	×	×	×	×	×			
test-P-PA1		×			×	×		
test-P-RFT1	×	×	×	×	×			×
test-P-C1	×	×	×	×		×	×	×
test-OE-PE1								
test-OE-PR1								
test-M-M1								
test-S-A1				×	×	×		×
test-S-IN1					×	×		
test-S-P1					×			
test-C-SC1								

Table 15: **Module Traceability Part 2**

Appendix — Reflection

The information in this section will be used to evaluate the team members on the graduate attribute of Reflection. Please answer the following question:

1. In what ways was the Verification and Validation (VnV) Plan different from the activities that were actually conducted for VnV? If there were differences, what changes required the modification in the plan? Why did these changes occur? Would you be able to anticipate these changes in future projects? If there weren't any differences, how was your team able to clearly predict a feasible amount of effort and the right tasks needed to build the evidence that demonstrates the required quality? (It is expected that most teams will have had to deviate from their original VnV Plan.)

There were a lot of changes that we made to the original VnV plan that we made back in November. Some of the tests in the original VnV were a bit too implementation specific (e.g. expecting a certain function to exist for the system tests) and had to be changed in order to fit the actual implementation that we made. Additionally, we had to make changes to how we did usability testing and performance testing since in the original VnV plan we didn't know exactly what we were going to do. We think that in future projects these issues would be better be able to be anticipated with more experience in creating VnV plans. Specifically, focusing more on the specifics on usability testing (e.g. determining the target user for the usability surveys) would have improved the VnV process a lot, especially with usability being an important portion of this project. Some other kinds of issues we would a anticipate better next time is that users could lie when marking tasks complete ad bills payed. Part of our VnV plan, we should have included that we should get at least 10 groups of housemates and let them try it out each for a month. This would uncovered a lot more issues like this one for example. Thus we will be better equipped to handle issues like those ones relating to usability and functionality.