CS 40800: Product Backlog

**Splytr**

Sahil Pujari, Keenan Wresch, Akshit Gudoor, Hari Krishnam Raju, Aritra Samanta

horizontal line

## **Problem Statement**

Roommates, even the good ones, struggle with communication. Until now, there has been no singular solution to help roommates gently remind each other and keep track of everything in their house. Having to do math manually in order to split a bill among roommates is tedious and finances are harder to maintain in the long run. Current systems such as Venmo and SplitWise are not designed according to the consideration of a roommate scenario while instead are a cohesive system. We plan to provide a web app that would help manage finances in a house better and would be designed based on this consideration. Other solutions to this problem also exist, however ours is different in that it will be made considering the needs of a roommate.

## **Background Information**

There are many aspects of living with the roommates that can be difficult to manage sometimes. Fiscal responsibilities can be sometimes mismanaged due to poor communications. Our app, Splytr, seeks to provide a one-stop solution to help and alleviate these issues and make life simpler for roommates to live along. We seek to deliver a modular application that can be molded and updated according to the needs of all the roommates across the world.

We acknowledge that there are already services that provide similar kind of services for roommates. Our application Splytr, will provide one location for you to do everything you need to manage your finances with your roommates. With modularity as a focus we will be constantly pushing updates based on customer need and feedback. Other apps are build with the initial features in mind, ours is built with the knowledge that customers needs evolve and our applications will need to grow with these needs.

## **Environment**

For backend functionality, we would like to use Node.js to provide a faster client - server interaction. We are going to be using the Express framework to build the project and to connect the backend, frontend and the database. We are going to be using EJS and Twitter Bootstrap to combine data, HTML - 5.0 and CSS to create a user - friendly environment for easy interaction with the application. We are going to be using MySQL database to store and retrieve user information. We will be using SSL/TLS or OAuth for profile and password protection of the user’s account.

### **Requirements**

## **Functional Requirements**

|  |  |  |
| --- | --- | --- |
| ID | Functional Requirements | Hours |
| 1 | As a user, I would like to Create an personal Account | 5 |
| 2 | As a user, I would like to manage my personal account (change password, change information, change profile picture, etc.). | 4 |
| 3 | As a user, I would like to be able to create a new house account. | 5 |
| 4 | As a user, I would like to be able to manage my house Information ( change house name, change address, etc.). | 4 |
| 5 | As a user, I would like to able to invite other members into the house. | 3 |
| 6 | As a user, I want to be able to remove users from the house who no longer live in the house. | 4 |
| 7 | As a user, I would like to accept or reject an House invitation. | 2 |
| 8 | As a user, I want to be able to create a split request amongst the complete house. | 4 |
| 9 | As a user, I want to be able to create a split request amongst a particular member of the house. | 5 |
| 10 | As a user, I want to be able to request money from a particular member of the house. | 5 |
| 11 | As a user, I want to be able to pay money to a particular member of the house. | 5 |
| 12 | As a user, I want to able to see all the pending split request existing on my account. | 3 |
| 13 | As a user, I want to be able to keep track of all the financial transactions happening on my account. | 2 |
| 14 | As a user, I want to be able to keep a track of how much my roommates owe me. | 3 |
| 15 | As a user, I want to be able to accept a split request. | 2 |
| 16 | As a user, I want to able to reject a split request along with a message. | 3 |
| 17 | As a user, I want to be able to set boundaries on split request. | 2 |
| 18 | As a user, I would like to restrict a person from a split request. | 3 |
| 19 | As a user, I would like to have some sort of communications among the members of the house. | 10 |
| 20 | As a user, I want to be notified when bills are about to due. | 5 |
| 21 | As a user, I want to be able to pool money together for group expenses. | 7 |
| 22 | The app should integrate with google plus (if time allows). | 6  (if time allows) |
| 23 | The app should integrate with Twitter (if time allows). | 5  (if time allows) |
| 24 | There should be a mobile version of the application - Android / ios (if time allows). | 30 - 40  (if time allows) |

## Total Hours :- 131 - 141 hours

## 

## **Non-Functional Requirements**

* As a user, I don’t want any chances of the web service being down if I need to pay my bills.
* As a developer, I will want to easily deploy builds with no downtime.
* As a developer, we will require dedicated hosting for our server software.
* As a sysadmin, I want to be able to easily scale the service and relational database like MySQL.
* As a developer, I want the application to have a material design like UIfa.
* As a user, I want a secure service taking no chance with payment and private information.
* As a user, I want a design focused on effectiveness and ease of use.
* As a user, I want the transactions to be completed fastly.
* As a user, I want the notifications to appear on time.
* Must be compatible with all the web browsers.
* As a developer, I want the application to have an admin portal
* As a developer, I want to create a Java REST API for backend functionality.
* As a developer, I want to make sure that the application doesn't take up too much of the user’s browser cache.
* As a developer, I want like to rigorously test the logic and implementation.
* As a developer, I want to have some form of tracking system to see the status of the web app.

**Use cases**

|  |  |
| --- | --- |
| **Case:** **‘Create Account’** |  |
| **Action** | **System Response** |
| 1). Click ‘Create Account’ | 2). Load Sign-up Page |

|  |  |
| --- | --- |
| **Case: ‘Sign-In’** |  |
| **Action** | **System Response** |
| 1). Click ‘Sign-In’ | 2). Load User Sign-In Page |
|  | 3). Load User DashBoard |

|  |  |
| --- | --- |
| **Case:** **‘Create House’** |  |
| **Action** | **System Response** |
| 1). Click ‘Create House’ | 2). Load House sign-up Page |

|  |  |
| --- | --- |
| **Case: ‘Account Information’** |  |
| **Action** | **System Response** |
| 1). Click ‘<User>’ | 2). Load Profile Information Page |

|  |  |
| --- | --- |
| **Case: House Information’** |  |
| **Action** | **System Response** |
| 1). Click ‘<House>’ | 2). Load House Information Page |

|  |  |
| --- | --- |
| **Case: ‘Invite Others’** |  |
| **Action** | **System Response** |
| 1). Click ‘Send Invitation’’ | 2). Load Send Invitation Page |
|  | 3). Search for user and display it on the Send Invitation Page |
| 4). Click Confirm Button | 5). Send Invitation for that particular user. |

|  |  |
| --- | --- |
| **Case: ‘Remove User’** |  |
| **Action** | **System Response** |
| 1). Click ‘<HOUSE>’ | 2). Load House Information Page |
| 3). Click on the User to remove | 4). Show up dialog box on the House page to confirm the removing of the user from house. |
| 5). Click Confirm | 6). Remove the user from the SQL for that particular House. |

|  |  |
| --- | --- |
| **Case: ‘Accept or Reject Invitation’** |  |
| **Action** | **System Response** |
| 1). Click ‘Invitation Notifications’ | 2). Display all the invitations for the user. |
| 3). Click ‘Accept’ | 4). Add the user to the house MYSQL database. |
| 5). Click ‘Reject’ | 5). Go back to the House Information Page |

|  |  |
| --- | --- |
| **Case: ‘See Pending Request’** |  |
| **Action** | **System Response** |
| 1). Click ‘House’ | 2). Display all the pending invitations for the user. |
| 3). Click ‘Accept’ | 4). Send money to the user requested |
| 5). Click ‘Reject’ | 5). Cancel invitation. |

|  |  |
| --- | --- |
| **Case: ‘View Dashboard’** |  |
| **Action** | **System Response** |
| 1). Click ‘View Dashboard’ | 2). Load personal dashboard to display items like amount owed, amount due etc. |

|  |  |
| --- | --- |
| **Case: ‘Split Request For House’** |  |
| **Action** | **System Response** |
| 1). Click ‘Create a split request’ | 2). Load Create Split request page |
| 3). Click ‘Create split request amongst house’ | 4). Load the create request page for house |
| 5). Click on ‘Confirm to send’ | 6). Send a request to all the members of the house |

|  |  |
| --- | --- |
| **Action: ‘Split Request for a Member’** | **System Response** |
| 1). Click ‘Create a split request’ | 2). Load Create Split request page |
| 3). Click ‘Create split request amongst house’ | 4). Load the create request page for house |
| 5). Click on ‘Confirm to send’ | 6). Send a request to that particular member requested by the user. |

|  |  |
| --- | --- |
| **Case: ‘Pay <User>’** |  |
| **Action** | **System Response** |
| 1). Click ‘Pay <User>’ | 2). Open saved card information to let the user confirm the amount and payment option as well. |

|  |  |
| --- | --- |
| **Case: Click ‘Edit Profile’** |  |
| **Action** | **System Response** |
| 1). Click ‘Edit Profile’ | 2). Open editable profile page for the personal account so that user can edit information |

|  |  |
| --- | --- |
| **Case: Click ‘Save Profile’** |  |
| **Action** | **System Response** |
| 1). Click ‘Save Profile’ | 2). Save edited profile to the database. |

|  |  |
| --- | --- |
| **Case: Click ‘Reject Request’** |  |
| **Action** | **System Response** |
| 1). Click ‘Reject request’ | 2). Reject current split request by a roommate and prompt the user to send a message as well. |

|  |  |
| --- | --- |
| **Case: Click ‘Pool Money’** |  |
| **Action** | **System Response** |
| 1). Click ‘Pool Money’ | 2). Prompt the user to enter the amount of money that is needed to be pooled for the group the user is a part of. Share the amount with other members of the group as well. |

|  |  |
| --- | --- |
| **Case: Notifications** |  |
| **Action** | **System Response** |
| 1). Login | 2). Notify the user the amount of money due and the amount of money owed on the dashboard as soon as the user logs into the account. User will be notified about the due dates as well. |

|  |  |
| --- | --- |
| **Case: Communication** |  |
| **Action** | **System Response** |
| 1). Send message/notification | 2) .Prompt the user to enter a message to communicate with other members of the groups about bills and other monetary things. |