

Software Requirements Specification

for

SplitPay

Version 1.0 approved

ZILDOR, Inc.

February 9, 2011

Prepared by:

Rick Aasen
Leland Cerauskis
Blake Matson

Ed Carlisle
Eric Jeffers
Josh Ritchey

Nick Carson
Travis Green
Phuong Vo

TABLE OF CONTENTS

1. Introduction	4
1.1 Purpose	4
1.2 Document Conventions	4
1.3 Intended Audience and Reading Suggestions	4
1.4 Project Scope.....	5
2. Overall Description.....	6
2.1 Product Perspective	6
2.2 Product Features	7
2.3 User Classes and Characteristics.....	9
2.4 Operating Environment	11
2.5 Design and Implementation Constraints.....	11
2.6 User Documentation	11
2.7 Assumptions and Dependencies.....	12
3. System Features	14
Core Features	14
3.1 User Registration and Welcome	14
3.2 Group Creation and Management.....	15
3.3 Posting a Bill	16
3.4 Member-to-Member Transactions	17
3.5 Final Debt Resolution	18
3.6 Group History	19
3.7 Show All Debts	20
3.8 Settings Menu	21
3.9 Help Menu.....	22
3.10 Push Notifications	22
Additional Features	24
3.11 Member Debt Visualization	24
3.12 PayPal.....	24
3.13 GPS Tracking.....	25
3.14 Receipt Imaging.....	26
3.15 E-mail/SMS Notifications.....	27
3.16 SplitPay Tutorial	28

4. External Interface Requirements	29
4.1 User Interface.....	29
4.2 Hardware Interfaces	37
4.3 Software Interfaces	37
4.4 Communications Interfaces	38
5. Other Nonfunctional Requirements.....	39
5.1 Performance Requirements.....	39
5.2 Safety Requirements	39
5.3 Security Requirements	39
5.4 Software Quality Attributes	39
6. Key Milestones.....	41
7. Key Resource Requirements	42
8. Other Requirements.....	43
9. <i>Appendix A</i> Glossary.....	44
10. <i>Appendix B</i> Project Proposal	45

1. INTRODUCTION

1.1 PURPOSE

The purpose of this Software Requirements Specification (SRS) document is to provide a detailed description of the functionalities of the SplitPay system. This document will cover each of the system's intended features, as well as offer a preliminary glimpse of the software application's User Interface (UI). The document will also cover hardware, software, and various other technical dependencies.

1.2 DOCUMENT CONVENTIONS

This document features some terminology which readers may be unfamiliar with. See Appendix A (Glossary) for a list of these terms and their definitions.

1.3 INTENDED AUDIENCE AND READING SUGGESTIONS

This document is intended for all individuals participating in and/or supervising the SplitPay project. Readers interested in a brief overview of the product should focus on the rest of Part 1 (Introduction), as well as Part 2 of the document (Overall Description), which provide a brief overview of each aspect of the project as a whole. These readers may also be interested in Part 6 (Key Milestones) which lays out a concise timeline of the project.

Readers who wish to explore the features of SplitPay in more detail should read on to Part 3 (System Features), which expands upon the information laid out in the main overview. Part 4 (External Interface Requirements) offers further technical details, including information on the user interface as well as the hardware and software platforms on which the application will run.

Readers interested in the non-technical aspects of the project should read Part 5, which covers performance, safety, security, and various other attributes that will be important to users. Readers who have not found the information they are looking for should check Part 8 (Other Requirements), which includes any additional information which does not fit logically into the other sections.

1.4 PROJECT SCOPE

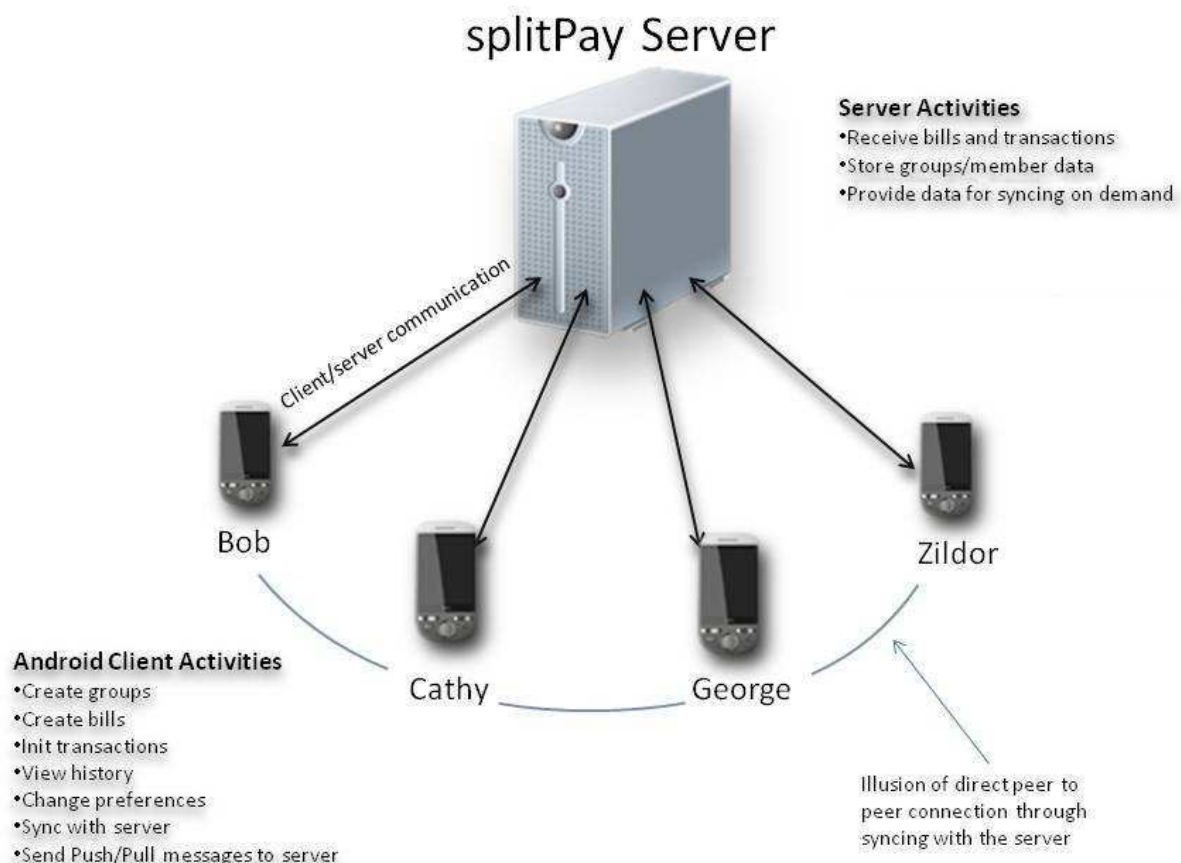
The SplitPay system is composed of two main components: a client-side application which will run on Android handsets, and a server-side application which will support and interact with various client-side features. The system is designed to facilitate the process of tracking and settling shared expenses. Potential scenarios include paying rent, splitting a check at dinner, sharing travel expenses, etc.

For more information about the project and its goals, see Appendix B (Project Proposal).

2. OVERALL DESCRIPTION

2.1 PRODUCT PERSPECTIVE

The SplitPay project is a new, self-contained product intended for use on the Android platform. While the SplitPay mobile application is the main focus of the project, there is also a server-side component which will be responsible for database and synchronization services. The scope of the project encompasses both server- and client-side functionalities, so both aspects are covered in detail within this document. Below is a diagram of the SplitPay system which illustrates the interactions between the server and client applications.



2.2 PRODUCT FEATURES

The following list offers a brief outline and description of the main features and functionalities of the SplitPay system. The features are split into two major categories: core features and additional features. Core features are essential to the application's operation, whereas additional features simply add new functionalities. The latter features will only be implemented as time permits.

CORE FEATURES

1. USER REGISTRATION & WELCOME

- Only appears once (the first time the application is run)
- Allows the user to register with the SplitPay server
- Enables the user to customize his/her account settings and preferences

2. GROUP CREATION & MANAGEMENT

- Streamlines the process of creating and organizing groups
- Provides support for multiple groups
- Allows the user to add group members manually or from contacts list

3. POSTING A BILL

- Stores and monitors the bill amount, the individuals involved, etc.
- Includes support for multiple simultaneous bills
- Efficiently distributes debt amongst the individuals responsible for the bill

4. MEMBER-TO-MEMBER TRANSACTIONS

- Enables group members to simulate transfers of debt, payments made, etc.
- Adjusts member balances accordingly
- Records relevant information (amount paid, members involved, etc.)

5. FINAL DEBT RESOLUTION

- Calculates the most efficient method of sorting out debts
- Notifies group members of unresolved debts, credits, etc.
- Offers the option to disband a group once all payments are made

6. GROUP HISTORY

- Automatically records all transactions and bills posted to each group
- Provides users with access to a detailed history of transactions
- Supports sorting transactions by date, amount, payer, etc.

7. SHOW ALL DEBTS

- Enumerates all of a user's unresolved debts across each group he/she is a part of
- Provides easy access to relevant information (past transactions, group info, etc.)
- Offers the option to resolve a debt (or debts) immediately

8. SETTINGS MENU

- Allows the user to customize his/her preferences
- Enables the user to modify certain features and functionalities
- Can be accessed at any time using the built-in Settings button on Android phones

9. HELP MENU

- Displays a list of topics covering the different components of SplitPay
- Offers detailed information on each feature, menu, etc.
- Can be accessed at any time via the Settings menu

10. PUSH NOTIFICATIONS

- Appear after any significant event occurs in a group
- Alert group members of newly incurred expenses
- Remind users of unresolved debts

ADDITIONAL FEATURES

11. MEMBER DEBT VISUALIZATION

- Presents a visual representation of current member balances
- Allows users to navigate through financial information in a more intuitive fashion
- Automatically updates as users post expenses and make transactions

12. PAYPAL INTEGRATION

- Incorporates a mechanism for initiating real transactions
- Facilitates secure, hassle-free transactions between members
- Automatically updates member balances as transactions occur

13. GPS TRACKING

- Stores location data associated with certain events
- Utilizes Google Maps to display transaction locations
- Creates an expense map which can be viewed by all members of a group

14. RECEIPT IMAGING

- Utilizes the camera built into Android handsets
- Records and stores a snapshot of receipts associated with different expenses
- Provides a method of checking/verifying expenses posted to a group

15. E-MAIL/SMS NOTIFICATIONS

- Extends the standard notifications service built into SplitPay
- Automatically delivers notifications via e-mail and/or text message
- Enables individuals without SplitPay to receive group notifications

16. SPLITPAY TUTORIAL

- Provides an abridged version of the Help menu for first-time users
- Offers a step-by-step run through of each feature, menu, etc.
- Enables any user to quickly and easily take advantage of all of SplitPay's functionalities

A major functionality present in several of these features is automatic synchronization. Using Android's internet capabilities, the application periodically communicates with the SplitPay server. This allows bills, transactions, groups, and group histories to be uploaded to a central server where the data can be shared with all other Android users in the group. This process of exchanging data between the server and the phone(s) is referred to as syncing.

For more detailed information, see Part 3 of the document (System Features).

2.3 USER CLASSES AND CHARACTERISTICS

The SplitPay project is meant to offer a shared expenses solution that is faster, easier, and more convenient than manually calculating and handling debts. Consequently, the application will have little or no learning curve, and the user interface will be as intuitive as possible. Thus, technical expertise and Android experience should not be an issue. Instead, anticipated users can be defined by how they will use the product in a particular situation. The following list categorizes the scenarios in which SplitPay is expected to be utilized:

SPLITPAY: POTENTIAL SCENARIOS

1. Long-term recurring expenses (e.g. rent, groceries, utilities)
 - Key functions:
 - Keep track of expenses
 - Notify users when debts are incurred
 - Record who has paid and who still owes
 - Requirements:
 - Method for inputting expenses into the application
 - Support for automated notifications
 - Database containing current debts, past transactions, etc.
2. Short-term recurring expenses (e.g. travel costs – gas, food, hotel)
 - Key functions:
 - Add new expenses (quickly and easily)
 - Record who is paying and what he/she is paying for
 - Update member balances on the fly
 - Requirements:
 - Simple and intuitive user interface for adding expenses
 - Expense-tracking system
 - Automated, background algorithm for calculating debts
3. Single expense (e.g., splitting a bill at dinner)
 - Key functions:
 - Create a group (quickly and easily)
 - Add non-registered individuals to the group
 - Quickly calculate each member's balance
 - Requirements:
 - Simple and intuitive user interface for adding groups
 - Support for group members without the SplitPay application
 - Option to calculate all balances on the spot

These groups are not meant to separate or categorize users, just the different situations in which SplitPay is likely to be used. In fact, a user may utilize the application for all of these scenarios simultaneously. This is another defining feature of the SplitPay system: support for multiple groups. This functionality allows a user to track expenses pertaining to several unrelated groups at the same time.

It is crucial that each of these situations be fully supported in the final product so as to maximize the overall value of the product. It is also important that the application be as user-friendly as possible, otherwise it will not be a viable alternative to handling shared expenses manually. Most importantly, the application must be reliable. Regardless of the situation, the application must accurately distribute costs. There is zero tolerance for error when dealing with financial transactions.

2.4 OPERATING ENVIRONMENT

The main component of the SplitPay project is the software application, which will be limited to the Android operating system (specifically Android 2.2 and above). The application is not resource- or graphics-intensive, so there are no practical hardware constraints. The app will rely on several functionalities built into Android's Application Programming Interface (API), so ensuring appropriate usage of the API will be a major concern. Beyond that, the application is a self-contained unit and will not rely on any other Android-related software components.

The application will, however, frequently interact with the SplitPay server, a virtual dedicated server hosted by GoDaddy.com. The server operates on a Linux CentOS platform with 1GB of RAM and 15GB of allocated storage space. The SplitPay database will be stored on the server using MySQL and will be interfaced with a wrapper written in PHP 5.

2.5 DESIGN AND IMPLEMENTATION CONSTRAINTS

The primary design constraint is the mobile platform. Since the application is designated for mobile handsets, limited screen size and resolution will be a major design consideration. Creating a user interface which is both effective and easily navigable will pose a difficult challenge. Other constraints such as limited memory and processing power are also worth considering. SplitPay is meant to be quick and responsive, even when dealing with large groups and transactions, so each feature must be designed and implemented with efficiency in mind.

2.6 USER DOCUMENTATION

The primary goal of SplitPay is to *facilitate* the process of managing shared expenses. Consequently, the application will be designed to be as simple to use as possible. Nonetheless, users may still require some supplementary information about each component of the SplitPay system. The application will contain two features that offer this: the SplitPay Tutorial and the Help menu.

The Help menu is a collection of topics covering each of the application's menus, features, etc. At any time, the user can navigate to the Help menu and select any of these topics to obtain more information. Details about the Help menu can be found in section 3.9 of this document.

The SplitPay Tutorial takes all of these topics and condenses them into a single, step-by-step demonstration that the user can access immediately after installing the application. This tutorial is meant to quickly and effectively teach new users the "ins and outs" of the application. Section 3.16 covers the tutorial in further detail.

2.7 ASSUMPTIONS AND DEPENDENCIES

TIME DEPENDENCIES

As mentioned previously, the features of SplitPay are divided into two groups: core features and additional features. Core features are crucial to the basic functionality of the SplitPay application. These features must all be implemented in order for the application to be useful.

Optional features, however, are *not* critical to the function of the application. They are usability improvements and convenience enhancements that may be added after the application has been developed. Thus, the implementation of these features is entirely dependent upon the time spent designing and implementing the core features. The final decision on whether or not to implement these features will be made during the later stages of the design phase.

HARDWARE DEPENDENCIES

Some of the additional features rely on hardware components present in Android handsets. For instance, the camera will be used to record images of receipts for digital storage. Consequently, this feature is entirely reliant upon the ability to access the camera's functionalities. In addition, the application will use the handset's location sensors (GPS) to record the location of a specific bill or transaction. Both the camera and the GPS functionalities will be achieved using the API provided by the Android operating system.

References: <http://developer.android.com/reference/android/hardware/Camera.html>
<http://developer.android.com/guide/topics/location/index.html>

EXTERNAL DEPENDENCIES

Several of the features presented in this document rely on the existence and maintained operation of several APIs. A non-exhaustive list follows.

EMAIL NOTIFICATIONS

The Android platform is not suited for sending mass emails. Thus, the central server will be responsible for this feature of the application. The smartphone client will notify the server when messages need to be sent using a custom API that is to be created. This API will use standard HTTP messaging to facilitate client-server communications. The API will be implemented using PHP.

SMS NOTIFICATIONS

This feasibility of this feature is yet to be determined. If implemented, this feature would allow offline users without an Android smartphone to receive notifications of outstanding debt and other information via text messages. A suitable and free text messaging API that can be called from the server has yet to be found. The possibility of sending text messages from the Android smartphone client itself is also being reviewed.

PAY-PAL WEB API

We will use the PayPal API in order to facilitate payment of debts that users may have incurred and wish to pay using the software.

Reference: <https://www.x.com/community/ppx/dev-tools>


GOOGLE PLACES API

The software application may integrate the ability to interact with Google Places for marking the location of a bill or purchase. Note that this API is not guaranteed to be perfectly functional, as it is currently in the beta phase of its lifecycle and is provided by an experimental branch of its host company, Google Labs. Using this API may require the use of the GPS hardware on the Android platform, where it is available.

Reference: <http://code.google.com/apis/maps/documentation/places/>

3. SYSTEM FEATURES

SplitPay's system features are divided into two main categories: core features and additional features. Core features form the body of the application and include any features that are essential to the functionality of the SplitPay system. These features must be implemented in order to have a fully-functioning application. Additional features, however, are not required for the app to function. They include any features which, if time permits, will be added to the application in order to provide extra functionality.

NOTE: Features marked by the server icon () exchange data with the SplitPay server and thus require both a functioning server system and an active internet connection. If no connection is available, the application will cache the data locally until a connection becomes available. The user will be notified if this condition arises.

CORE FEATURES

3.1 USER REGISTRATION AND WELCOME

When the application is installed and run for the very first time, the user is presented with an initial registration/welcome screen. This screen prompts the user to create an account on the SplitPay server using the email address associated with his/her Google account. The user also enters a "Display Name", which will be the name that is shown as their handle within the groups. Completing this process will create and store an account for the user on the SplitPay server, enabling all of the application's synchronization capabilities.

3.1.1 STIMULUS/RESPONSE SEQUENCES

- Step 1** SplitPay application launched from the Android home screen
- Step 2** The user is prompted to enter an email address and a display name
 - The user's Google account info is entered by default
- Step 3** This information is sent to server and stored in the database
- Step 4** Registration is completed and user is taken to main screen

3.1.2 USER REQUIREMENTS

VALID EMAIL ADDRESS

The user cannot proceed until a valid email address is entered. The application will verify that the user's input is consistent with the format of an email address (i.e. xxxxx@xxxxx.xxx).

3.1.3 SYSTEM REQUIREMENTS

SECURE DATABASE SYSTEM

The application must insure that the user's information is encrypted and safely stored.

3.2 GROUP CREATION AND MANAGEMENT



The "Groups" screen will be the main screen of the application. From this screen, users will be able to view and manage existing groups. Groups may be created by adding members from the user's contacts or by manually entering an email address and name. The creator of a group is designated as the "Leader" of that group. The Leader is responsible for confirming transactions submitted by other members of the group and will also have the ability to stop or disband the group.

3.2.1 STIMULUS/RESPONSE SEQUENCES: GROUP CREATION

- Step 1** The user selects "Create a New Group" from the main Groups screen
- Step 2** The user must choose a unique name to be used as the group's identifier (e.g., "D.C. Road Trip"). The user can add members to the group from contacts stored on the phone or via manual entry (email address and display name are required).
 - Users who are not registered with the SplitPay server are designated on the server as "offline users"
 - These users cannot participate in transactions directly, so the leader is responsible for reconciling their balances
 - At any time, these offline users may register a SplitPay account (using the same information stored on the server) and take control of their transactions
- Step 3** The user finishes selecting members and confirms that the group is complete
- Step 4** The user is then designated as the leader of the group

3.2.2 STIMULUS/RESPONSE SEQUENCES: GROUP MANAGEMENT

From the management Screen, the Leader can choose from the following options:

- Remove member(s)
- Add member(s)
- Disband group (leave all debts hanging)
- Disband and Remove Group
 - Final Balance sheet is posted to emails and notifications
 - Group is removed from “Groups” screen
- STOP Group and Reconcile Group
 - The Group still persists in the main screen but new charges cannot be added.
 - Transactions to resolve the final balances may still be processed within the group
- Manage Transactions (Confirm/Deny member to member payments)
- Notify All (sends current balance notification to all members in group)

3.2.3 USER REQUIREMENTS

VALID EMAIL ADDRESS(ES)

The group leader must enter valid e-mail addresses for group members in order to obtain the full functionality of the application.

3.2.4 SYSTEM REQUIREMENTS

DATA STRUCTURE(S) FOR REPRESENTING BILLS

Bills must be discrete objects that the application/server can store and manipulate.

3.3 POSTING A BILL



On several screens, users are given the option to create a new expense, or “Bill”, and post it to a particular group. Bills created within the application are meant to represent expenses incurred by the group in real life. Users must input a display name (e.g., “Groceries”), the associated cost, and select the individuals responsible for the expense. Once the Bill is confirmed, it is synced to the server, enabling all other SplitPay users in the group to view it.

3.3.1 STIMULUS/RESPONSE SEQUENCES: POSTING A BILL TO THE GROUP

- Step 1** The user is presented with a screen where they enter the following information:
- Expense title
 - Bill amount
 - Optional information includes
 - Location
 - Type of expense (e.g., Travel, Groceries, Food, etc.)
 - Comments (i.e., any other information the user finds relevant)
- Step 2** The user then selects the members of the group involved in the Bill
- User can choose “Select All” or pick members individually
- Step 3** An algorithm calculates the distribution of debt amongst each member involved in the bill
- Step 4** The Bill is pushed to the server, which notifies other group members of the Bill and also updates their running balances

3.3.2 USER REQUIREMENTS

NONE

Only an active internet connection is required.

3.3.3 SYSTEM REQUIREMENTS

COST-DIVISION ALGORITHM

The algorithm must divide the cost and add the correct amount to each user’s current balance

3.4 MEMBER-TO-MEMBER TRANSACTIONS



This feature represents a real-world transaction between two or more members of a group. A common scenario would involve the user resolving his/her debt by making payments to other group members. The resulting changes in balances are calculated automatically and displayed for the users involved.

3.4.1 STIMULUS/RESPONSE SEQUENCES

- Step 1** The user selects Member-to-Member Transaction and inputs the following data:
- Transaction Description (optional)
 - Member(s) to be paid
 - Amount to be paid (to each member if multiple)
- Step 2** The user confirms the transaction
- Step 3** Each involved member has his/her balance adjusted automatically
- Step 4** The information is sent to the server for approval by the Leader (unless the Leader created the transaction)

3.4.2 USER REQUIREMENTS

NONE

Only an active internet connection is required.

3.4.3 SYSTEM REQUIREMENTS

REAL-TIME BALANCING

The application must be able to update each user's balance on the fly.

3.5 FINAL DEBT RESOLUTION



This feature is utilized when the Leader of a group wants to end the group and resolve all debts. An algorithm takes into account the balances of all members and determines the most efficient method of resolving debts, minimizing the number of transactions between members.

3.5.1 STIMULUS/RESPONSE SEQUENCES

- Step 1** The group's Leader chooses to end the group and resolve all debts
- Step 2** The algorithm automatically determines the most efficient way to resolve all debts
- Step 3** E-mails and/or notifications are sent out containing the following information:
- The group's name, members, and the user's overall balance in that group
 - A list of group members who the user owes and how much he/she owes to each person
 - A list of group members who owe the user and how much each person owes him/her

- Step 4** Each member can then use Member-to-Member transactions to resolve his/her debts.
- Step 5** Once all debts are resolved, all members are notified and the group may then be removed from the main Groups page. Bill and Transaction Histories are still available, however.

3.5.2 USER REQUIREMENTS

NONE

Only an active internet connection is required.

3.5.3 SYSTEM REQUIREMENTS

ALGORITHM FOR OPTIMIZING DEBT RESOLUTION

The algorithm must be able to accurately and efficiently calculate the transactions required to solve all outstanding debts. Ideally, the algorithm will reduce the number of transactions to the absolute minimum, making things easier for the group.

3.6 GROUP HISTORY



This screen provides a view of all transactions and bills that occurred within a group. This list will be presented in chronological order by default, but can also be sorted by payer, amount, etc. It will show the names of members involved in each transaction and the amounts paid/received, and the user has the option of viewing each item in more detail by selecting it. The detailed view will display all members involved in the bill/transaction, any comments about it, and any additional information that was included when it was created (Location, Type, etc.).

3.6.1 STIMULUS/RESPONSE SEQUENCES

- Step 1** The user is presented with a list of all transactions/bills posted to the current group
- The user may sort these items by date, amount, payer, etc.
- Step 2** The user may select any one of the transactions/bills for detailed viewing
- Upon selection, a dialog is presented with the details of the transaction/bill

3.6.2 USER REQUIREMENTS

NONE

Only an active internet connection is required.

3.6.3 SYSTEM REQUIREMENTS

NONE

Only an operational server is required.

3.7 SHOW ALL DEBTS



This is a global feature that will enumerate an individual's debt across all groups of which he/she is a member. Visual cues will be used to provide a distinction between positive, negative, and even balances within each group. There will also be an option to reconcile all debts from this screen. Selecting this option will show the user a list of transactions which are required before his/her debts can be resolved. The user will be able to initiate these transactions within the app from this screen.

3.7.1 STIMULUS/RESPONSE SEQUENCES

- Step 1** The user enters this screen from the popup context menu from any other screen in the app
- Step 2** The user can see his/her status in each of the groups he/she is a member of
 - Selecting a specific group from this list displays that group's History page
- Step 3** The user has the option to resolve his/her debts with a transaction for each group

3.7.2 USER REQUIREMENTS

NONE

Only an active internet connection is required.

3.7.3 SYSTEM REQUIREMENTS

NONE

Only an operational server is required.

3.8 SETTINGS MENU

This menu allows the user to modify more advanced settings within the application. The menu is accessed from any screen by the Preferences button, a hardware button built into all Android handsets. Modifiable settings include:

- User settings (e.g., e-mail and other personal information can be modified)
- Notifications preferences (how and when the user is notified, if at all)
- Feature preferences—this includes the following options:
 - Interface preferences (allows the user to choose what information is displayed)
 - PayPal account preferences (see section 3.11 for further details)
 - GPS preferences (see section 3.12 for further details)

From the Settings menu, the user may also access the Help screen (see section for details).

3.8.1 STIMULUS/RESPONSE SEQUENCES

- Step 1** From any menu, the user may press the Settings button
- This is a physical button featured on all Android phones
- Step 2** From a pop-up menu, the user may select what he/she wishes to modify
- Step 3** After making the desired changes, the user may save his/her preferences and exit the menu, or simply exit without saving
- Step 4** The user is then returned to the screen he/she was on before accessing the Settings menu

3.8.2 USER REQUIREMENTS

NONE

The Settings menu may be accessed at any time, even if there is no connection with the server.

3.8.3 SYSTEM REQUIREMENTS

DYNAMIC, MODIFIABLE PREFERENCES FOR KEY FEATURES

The software must be implemented in such a way that certain features can be customized to each user's preferences. This requires that certain features be dynamic and flexible.

3.9 HELP MENU

The Help menu is meant to answer any questions the user may have while using SplitPay. The menu displays a list of topics related to features, menus, and the app in general. The user can select any of these topics to access further information and explanations.

3.9.1 STIMULUS/RESPONSE SEQUENCES

- Step 1** From any screen, the user may press the Settings button built into the phone
- Step 2** This displays the Settings menu, where he/she can then select the Help menu
- Step 3** The user is presented with a list of help topics which he/she can scroll through
- Step 4** Once the user selects a topic, more information on that topic is displayed
- Step 5** The user can navigate back to the Help screen or exit the Help menu altogether

3.9.2 USER REQUIREMENTS

NONE

The Help menu may be accessed at any time, even if there is no connection with the server.

3.9.3 SYSTEM REQUIREMENTS

DATABASE OF HELP TOPICS/INFORMATION

The application must contain a small database containing detailed information pertaining to each of the menus, features, and any other relevant aspects of the SplitPay system.

3.10 PUSH NOTIFICATIONS



Push notifications is an added mechanism to provide updates and alerts from the application server to the user's android device. Whenever a new expense is posted in a group, the server will broadcast notifications to all group members' phones of the new bill and each user's respective balances. Members are also notified when all debts in a group are resolved or when the leader chooses to notify all members of the group's balance. Priority: medium

3.10.1 STIMULUS/RESPONSE SEQUENCES: NEW BILL NOTIFICATION

- Step 1** A member of the group creates a bill and is confirmed by the leader.
- Step 2** The server pushes notifications to all members' phones. The pop-up notification will contain information about the new bill, similar to the following example:

New Bill: \$50 gas
Paid by: Blake
Group: 'Trip to DC'
Your balance: \$10

3.10.2 STIMULUS/RESPONSE SEQUENCES: GROUP NOTIFICATION

- Step 1** The leader selects to notify all members of the group's current balance.
- Step 2** Server pushes a notification to all members' devices with a pop-up.
- e.g., "Current Balance of the Group 'Trip to DC' is \$100."

3.10.3 STIMULUS/RESPONSE SEQUENCES: FINAL DEBT RESOLUTION

- Step 1** The group leader opts to stop the group and resolve all debts
- Step 2** After payments are calculated, the server notifies each group member of the transactions he/she must make to resolve all debts

3.10.4 USER REQUIREMENTS

NONE

Only an active internet connection is required.

3.10.5 SYSTEM REQUIREMENTS

NONE

Only an operational server is required.

ADDITIONAL FEATURES

3.11 MEMBER DEBT VISUALIZATION



This feature allows users to access group balances as bar graphs, rather than the default text-only form. These visuals can be obtained at any time by selecting the Member Debt Visualization option from any group. The visuals are updated any time a group member's balance changes.

3.11.1 STIMULUS/RESPONSE SEQUENCES

- Step 1** The user activates this visualization via a button on a particular group page
- Step 2** From here, the user can view details about individual members by selecting them

3.11.2 USER REQUIREMENTS

NONE

Only an active internet connection is required.

3.11.3 SYSTEM REQUIREMENTS

GRAPHICS IMPLEMENTATION

The application must have a method of converting raw data into rich, dynamic visuals.

3.12 PAYPAL



This feature will offer users the option to resolve their debts through using PayPal. This will enable users to make real financial transactions through the app, which will then update all balances accordingly.

3.12.1 STIMULUS/RESPONSE SEQUENCES

- Step 1** The user accesses this feature from any transactions-related screen
- Step 2** The user enters his/her PayPal account information
 - The user has the option of being automatically logged in next time he/she uses this feature
- Step 3** The user is taken to a screen where he/she must fill out all of the necessary details of the transaction
 - If this information was already entered in the transactions screen, it will be filled in automatically
- Step 4** The user confirms the payment, and the transaction is completed
 - All relevant balances are updated to reflect this

3.12.2 USER REQUIREMENTS

PAYPAL ACCOUNT

Only users with a PayPal account will be able to utilize its functionalities. Obtaining a PayPal account is done independently of the SplitPay system.

3.12.3 SYSTEM REQUIREMENTS

PAYPAL INTEGRATION

The application must utilize the PayPal API to provide the appropriate functionalities.

3.13 GPS TRACKING



When a Bill is posted or a transaction is made, this feature records the GPS location of the phone(s) involved. At any point, a user may access this information in the form of a map with each saved location displayed. This feature provides a visual record of where the group has been, what payments were made at specific locations, and what path on the map the group has taken. The application will use Google Places as its maps tool.

3.13.1 STIMULUS/RESPONSE SEQUENCES

- Step 1** When posting a bill or transaction, the user has an option to add the GPS location
- Step 2** If this option is selected, the phone records the GPS location and also sends it to the server, where it is synced to all other SplitPay-enabled phones
- Step 3** The user may also manually enter in an address if the GPS location is not desired
- Step 4** From any group screen, the user may access the group's associated map
- Step 5** Each address and/or GPS location is pinned to the map, with a line tracing the path between each pin

3.13.2 USER REQUIREMENTS

NONE

Only an active internet connection is required.

3.13.3 SYSTEM REQUIREMENTS

GOOGLE PLACES INTEGRATION

The application must utilize the Google Places API to provide the appropriate functionalities.

3.14 RECEIPT IMAGING



Receipt imaging allows users to store the receipt associated with a Bill for later viewing. Whenever a user posts a Bill, he/she will have the option to take a photo of the receipt using the phone's built-in camera. This image is then sent to the server and synced with all SplitPay users in the group.

3.14.1 STIMULUS/RESPONSE SEQUENCES

- Step 1** While creating a new Bill, the user may select the Receipt Imaging feature.
- Step 2** This activates the camera and creates a window on the screen to act as a viewfinder.
- Step 3** The user then takes a picture of the receipt by pressing a button on the app.
- Step 4** The user has the option to retake the picture until he/she is satisfied with the result.
- Step 5** The picture of the receipt is then saved and becomes accessible by viewing the Bill.
- Step 6** After the new bill is created, the data will be pushed to the server and synced with other phones

3.14.2 USER REQUIREMENTS

FUNCTIONING CAMERA

The phone's built-in camera must be functioning properly.

3.14.3 SYSTEM REQUIREMENTS

CAMERA INTEGRATION

The application must utilize the Android API to gain access to the camera.

3.15 E-MAIL/SMS NOTIFICATIONS

Since groups may involve individuals without SplitPay (i.e., offline users), there must be an alternative method of notifying these individuals. The simplest way to achieve this is via e-mail or text message (SMS). This will allow these individuals to enjoy some of the same functionalities as SplitPay users.

3.15.1 STIMULUS/RESPONSE SEQUENCES

- Step 1** When a Bill is posted or a transaction is made, the server is updated and any user involved is sent a notification
- The group leader may modify offline users' notification preferences, which will determine whether the user receives an e-mail, a text message, or no notification at all
- Step 2** SplitPay users are notified within the application, but offline users are sent an e-mail or text message containing information about the update.

3.15.2 USER REQUIREMENTS

A VALID E-MAIL ADDRESS OR PHONE NUMBER

The offline user must have some method of being contacted electronically.

3.15.3 SYSTEM REQUIREMENTS

E-MAIL AND SMS CAPABILITIES

The phone must be able send a request to the server to compose and send e-mails/SMS.

3.16 SPLITPAY TUTORIAL

The SplitPay Tutorial is a structured, condensed version of the Help menu. This feature will allow the user to view brief explanations of each menu and feature. These explanations will be kept brief, as the user interface will be designed to be as intuitive as possible. The tutorial will be designed to answer any questions a first-time user may have about the application.

3.16.1 STIMULUS/RESPONSE SEQUENCES

- Step 1** At the initial registration screen, the user will have the option to view the tutorial
- Step 2** The tutorial will then direct the user to each of the menus within the app, explaining each feature in some detail. The user will not be able to do anything on these screens besides navigating through the steps of the tutorial.
- Step 3** Once each menu/feature has been explained, the tutorial will return the user to the welcome screen and allow him/her to complete the registration process.
 - Alternatively, the user may quit at any point in order to return to the welcome/registration screen

3.16.2 USER REQUIREMENTS

NONE

The user does not have to satisfy any conditions in order to access the tutorial.

3.16.3 SYSTEM REQUIREMENTS

TUTORIAL SYSTEM

The application must have a tutorial system in place that is able to navigate through each menu without letting the user modify any fields. This system will also include pop-ups which briefly explain each feature, as well as navigation buttons that allow the user move back and forth between steps.

4. EXTERNAL INTERFACE REQUIREMENTS

4.1 USER INTERFACE

4.1.1 GENERAL LAYOUT



- From every menu, the user can access the built-in settings and back/previous button
 - These are physical buttons built into the phone, so they are not shown here
- Every menu will have a descriptive title
- Menus will be readable, but will make efficient use of space so the user isn't forced to navigate through too many menus

4.1.2 WELCOME SCREEN

The image shows a mobile application welcome screen. At the top, it says "Welcome to splitPay" in a large, dark blue font. Below this, there is a blue rounded rectangle containing the text "First time user". Underneath, there are three input fields: "Display name:" with the value "Bob", "Email:" with the value "bob@gmail.com", and "Mobile:" with the value "1-800-999-9999". At the bottom of this blue rectangle are two buttons: "Tutorial" and "Done".

- For first-time users only
- Prompts the user to enter account information to be stored on the server
- Notifies the user if their information is invalid
- Provides a quick tutorial which explains the different menus and options

4.1.3 MAIN MENU



- Shows all groups, and how much the user owes the group collectively
- User can select a group to open a View Group (in a separate menu)
- Create a Bill option (Separate menu)
- Create group option (Separate menu)

4.1.4 VIEW GROUP



- Shows (graphically) how far ahead/behind each member is to the group in terms of collective debt to the group
- Shows (graphically) how much the users owes or is owed by each member of the group
- Create a Bill option (separate menu)
- Users can make a "Member to Member" transaction by selecting a member and entering a payment amount (in a separate menu)
- Transaction History (separate menu)
- Group leaders have additional options
 - Add / Remove members
 - Clear debts (individuals or entire group)
 - Designate a new Leader
 - Resolve debts (entire group)
 - Disband group

4.1.5 CREATE / MANAGE GROUP



- Prompts user to name the group
- Allows the user to select members from contacts or enter their information manually
- The group creator will initially assume the role of "Group Leader", however they will have the option to pick someone else in this menu

4.1.6 CREATE A BILL



- User is prompted to enter the bill amount
- Members of the group splitting the bill (including the user) will be displayed below, along with the option to add and remove members.
- The user has the option to select payers from the list of members (Initially the app will assume the user is paying)

4.1.7 TRANSACTION HISTORY



- Shows list of transactions/split bills between the user and members of the group
- The group leader can confirm/deny a transaction's validity

4.1.8 MEMBER TO MEMBER TRANSACTION



The image shows a mobile application screen titled "Transaction". Below the title is a blue rounded rectangle containing the text "From Bob To Todd". Underneath this, there is a checkbox, a dollar sign, and a text input field showing "0.00". Below the input field is another checkbox labeled "Clear all debt". At the bottom of the blue rectangle are two buttons: "PayPal" and "Cash/Other".

- Prompts the user to enter an amount to pay the selected individual
- (Optionally) User can select different payment options

4.2 HARDWARE INTERFACES

SplitPay is intended as a mobile application for the Android platform and hence is solely supported on Android-powered devices. Messages, updates, and data exchanged between Android devices are transmitted to and handled by the SplitPay server. This produces the illusion of peer-to-peer interactivity between group members, however this is not the case as all interactions always run through the central server first.

SplitPay is being developed specifically for Android 2.2 (Froyo) and all versions released after it. The Android platform supports push messages that will be used to synchronize data between the local application and the main application server. Information will be sent using TCP/IP and the HTTP protocol.

The Android platform provides abstractions for all network communication interfaces and thus the hardware as well.

4.3 SOFTWARE INTERFACES

The SplitPay app is to be developed under the Android operating systems using the Java JDK (Java Development Kit) and the Android SDK (software development kit) tools.

4.3.1 INCOMING AND OUTGOING ITEMS

- Outgoing data consists of group information, bills, transactions, and confirmations sent by users to the server.
- Incoming data consists of updates from the server regarding member balances, as well as any notifications deemed necessary.

4.3.2 SERVICES AND COMMUNICATIONS

- SplitPay relies on server push and pull protocols to be fully functional
- Communication will occur in occasional, short bursts between a user's phone and the server in the following situations:
 - Whenever a user creates/confirms a new bill or transaction
 - Whenever the server finishes processing group/member debts to update users
 - The application will notify the server when it successfully receives an update

4.4 COMMUNICATIONS INTERFACES

The SplitPay application will have a network server that is web-based and created using the PHP (Hypertext Processing) language. The server exists to retrieve information from the database and calculate payments. The product also calls for a database system that stores user information and transaction history between users. The HTTP server will use a push protocol to push notifications of updates onto the Android phones. Furthermore, whenever a user opens the SplitPay app from their phone, a pull protocol will be used to retrieve and sync the latest transaction updates from the server.

5. OTHER NONFUNCTIONAL REQUIREMENTS

5.1 PERFORMANCE REQUIREMENTS

Performance should not be an issue because all of our server queries involve small pieces of data. Changing screens will require very little computation and thus will occur very quickly. Server updates should only take a few seconds as long as the phone can maintain a steady signal. The cost-division algorithms used by in application will be highly efficient, taking only a fraction of a second to compute.

5.2 SAFETY REQUIREMENTS

SplitPay will not affect data stored outside of its servers nor will it affect any other applications installed on the user's phone. It cannot cause any damage to the phone or its internal components. The only potential safety concern associated with this application applies to virtually all handset apps: SplitPay should not be used while operating a vehicle or in any other situation where the user's attention must be focused elsewhere.

5.3 SECURITY REQUIREMENTS

This application assumes that only the user or whoever he/she allows will have access to his/her Android handset. With that being said, only a Google email address is required to verify the identity of the user upon opening the app. Since it is not password protected, there is no method to authenticate the user's identity. This could only pose a thread if a user has set up PayPal functionality, however any transaction involving real currency must be authorized and confirmed before becoming final. The PayPal API provides all of the security checks needed to ensure that no fraudulent transactions occur.

5.4 SOFTWARE QUALITY ATTRIBUTES

The graphical user interface of SplitPay is to be designed with usability as the first priority. The app will be presented and organized in a manner that is both visually appealing and easy for the user to navigate. There will be feedbacks and visual cues such as notifications to inform users of updates and pop-ups to provide users with instructions.

To ensure reliability and correctness, there will be zero tolerance for errors in the algorithm that computes and splits expenses between group members. To maintain flexibility and adaptability, the app will take into account situations in which a user loses internet connection or for whatever reason cannot establish a connection with the server. These users will still be able to use the application, but any Bills, transactions, etc. posted while disconnected will be cached until the connection is restored.

Furthermore, the group leader also has the option to add members who do not own an Android phone and add transactions on their behalf. With SplitPay being ported solely for the Android platform, this software application has the advantage of being portable and convenient to use whenever and wherever. Overall, the app balances both the ease of use and the ease of learning. The layout and UI of the app will be simple enough that users will take no time to learn its features and navigate through it with little difficulty.

6. KEY MILESTONES

Milestone	Deadline	Comments
<i>SRS document</i>	2/8	Abstract done by Jan 31 so a presentation can be created
<i>Finalized Interface Design</i>	2/19	Designs should be done a week in advance of abstract for revisions
<i>Finalized Algorithm Design</i>	2/19	Designs should be done a week in advance of abstract because of bugs
<i>SDD document</i>	3/3	Abstract done by Feb 26 so a presentation can be created
<i>Server Setup</i>	3/10	This should be done in the beginning of the implementation phase to sync with other features
<i>Interface & Algorithm Implementation</i>	4/13	This should be completed two weeks in advance of complete project so app can be thoroughly tested and bugs fixed
<i>Software Festival</i>	4/27	Completion of project

7. KEY RESOURCE REQUIREMENTS

Major Project Activities	Skill/Expertise Required	Internal Resource	External Resource	Issues/Constraints
Create a Server application	Database systems & servers experience	Nick and Eric have general knowledge in this field	Internet	No physical server; limited number of team members with SQL/PHP experience
Sync the Server to the Application	Application networking with online servers	Nick and Eric have general knowledge in this field	Internet	Possible compatibility issues
Design the Interface	Design & usability experience; familiarity with Android GUI design	Josh has extensive digital design experience; Rick has experience with human-computer interaction (HCI)	Android Developers community; published examples	Schedule conflicts; physical constraints of handset screen resolution
Design an algorithm	Knowledge & experience designing mathematical algorithms	All members have taken data structures and algorithms	Android Developers community; published examples	Potential schedule conflicts
Implement an algorithm	Familiarity with the Android development environment	All team members have Java programming experience	Android Developers community; published examples Android emulators are publicly available for testing	Potential schedule conflicts
Implement the interface	Familiarity with Android GUI implementation	All team members have Java programming experience	Android Developers community; published examples Android emulators are publicly available for testing	Potential schedule conflicts

8. OTHER REQUIREMENTS

A database for SplitPay calls for a server side implementation that holds information for the users, groups, bills, transactions, as well as all the relationships involved. The database will be using MySQL. The following provides an example of information that may be stored in the database:

- **Users:** ID, Name, phone number, email, PayPal username
- **Groups:** ID, Name, Members
- **Bills:** ID, Name, Owner, Group, Cost, Receipt, Date, Time
- **Transactions:** ID, Members Involved, Group, Date, Time, Amount

The server will be configured on a Linux platform, and through use of PHP will allow interaction and processing in conjunction with the database. Processes to be done on the server include: pushing/pulling data, updating data, and generating notifications.

9. *APPENDIX A* GLOSSARY

OFFLINE USER

A user that has been added to the SplitPay server but does not have a copy of the application. These users are contacted exclusively through email and/or SMS. An offline user may become a SplitPay user by downloading the application and claiming their account.

GROUPS

Groups in the context of the SplitPay application are a collection of users that can create Bills and Transactions amongst each other.

BILLS

Bills are a method of distributing an expense among some or all members of a group.

TRANSACTIONS

Transactions are used for re-allocating debt to different members of the group. Transaction amounts may be positive or negative and may involve two or more members of a group.

DISBAND AND DELETE GROUP

Terminates the Group. All debts are left hanging and no final debt resolution notice is issued. The group is removed from the Groups screen.

STOP GROUP

Bills and transactions that add debt can no longer be posted to the group. The group enters the debt resolution state where transactions may only be applied to resolve all debts. The group still persists in the Groups screen.

STOP AND DELETE GROUP

The group is removed from the groups screen, and a final debt resolution notice is issued to all members via email.

10. *APPENDIX B* PROJECT PROPOSAL

Group 1492: Software Application Proposal

Working title: SplitPay

Platform: Android (2.2+), predominantly Java programming

MOTIVATION

We are designing an application that facilitates the process of paying off shared expenses. In other words, we want to make it easier to pay for things as a group. This is relevant in many different situations, ranging from every day transactions (friends splitting the cost of dinner), to recurring payments (roommates paying rent), to a more professional setting (a team of professionals making business transactions). Depending on the number of people and/or the amount of money involved, these situations can get very complicated.

PROBLEM STATEMENT

The problems we aim to alleviate with this app include the following:

- Difficulty of calculating divided costs
- Unfair or imbalanced payments amongst individuals
- Lack of accountability for money owed
- Difficulty remembering payments already made, outstanding debts, etc.

The app we design will solve these problems and several more. Consequently, we will include the following functionalities:

- Automatic calculation of money owed by each member of the group
- Customizable formula for splitting costs
- Transaction history (including all payments made & costs incurred by each individual)

We will also add the following features, which extend beyond the simple task of dividing costs:

- Support for multiple transactions which can take place over extended periods of time
- Automatic notifications, alerting users of outstanding debts and/or other relevant info
- Transaction synchronization, keeping each group member up to date at all times
- PayPal integration, allowing for easier transactions
- Receipt storage, which allows the user(s) to save photos of receipts associated with particular purchases/expenses
- E-mail and print functions, allowing non-Android users to obtain copies of transactions

NOTE: Some of these features are not part of our core design and will only be added if time permits.

OBJECTIVES

Beyond implementing the above features, there are a number of additional objectives we must accomplish as a team. First and foremost, as a group we must familiarize ourselves with the Android platform, as this will lay the groundwork for future development. Additional objectives include the following:

- Develop a timeline detailing each stage of development

- Establish well-defined roles for each member of the group

- Create detailed software specifications

- Ensure each member is up to speed and completing work on time

- Form a testing framework to simplify the debugging process

- Create a fully-functional, bug-free application

METHODOLOGY

We will be dividing the team into 3-4 subgroups, including some or all of the following: Research/Documentation, Android/Java Implementation, Server/Web, and GUI & Graphics. In addition, we will divide the development process into several discrete stages. The first stage will be dedicated to getting everyone in the group acclimated to the new development platform. We will do this by finding and sharing resources (documentation, sample code, etc.), as well as running through the set-up process for the Android SDK as a group. Before beginning development, we will establish documentation guidelines and testing frameworks in order to enhance maintainability and prevent bugs in the long run.

Next, we will work up specifications, establish deadlines, define roles, and allocate tasks to each group member. We plan on dividing up the development process into two central phases, core development and feature development. Core development will involve implementing the essential aspects of our application (calculator, transaction history, auto-sync, etc.), whereas feature development will encompass any additions which extend the app's capabilities (PayPal, advanced GUI, e-mail/print functions, etc.). We will implement these additional features in order of priority (which we will establish before beginning development).

HISTORY

There are several applications "on the market" which are similar to our proposed app (examples include Bills Are In, Share a Bill, Split a Bill, Fair Share, and Xpense Split). A majority of these are web-based applications with little or no mobile functionality. In addition, they all require users to set up an account before they can utilize their services. The remaining apps we found were limited to the iOS platform, leaving the Android market open. Furthermore, we plan on adding some features which are not currently available in any of these applications.