Community Costs

Ethan Goldman-Kirst (egk35)
Tiffany Lin (lintiff)

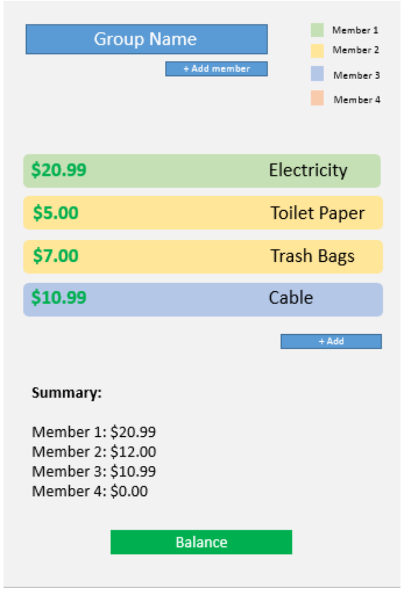
Overview

When going out, working with, or living with a group of people, bills and money are bound to be an issue. For example, calculating how to split a check evenly is complicated especially when a few people are carrying cash and other credit cards. How much do you owe? And to whom do you owe that to? Community Costs will keep track of that so that time isn't wasted on calculations and payments aren't forgotten. When living with roommates, every roommate usually chips in to buy and pay for different items/services, such as toilet paper and electricity. Instead of wasting time to notify every roommate of what was paid for, how much was spent, and how much they owe, Community Costs keep each roommate updated on what others have bought, how much each person has paid, and outstanding totals.

Vision

We want to make an app that makes keeping track of all purchases and running payments to your circles of friends, roommates, and coworkers easier. Current ways people use to do this include random notes and lists to remember payments in the future, and messaging threads or Facebook groups to communicate and update their friends on new purchases and payments.

To ease this process, Community Costs is an app that combines both payment information storage as well as group notification and updates. To work well, it requires users in your circle to all have the app so that they can communicate within group by adding their purchases. Each user can create new groups they'll be a part of. Then, every individual in this group can put down items they have purchased which will update the rest of the group. The app will keep track of how much each group member has spent and will let each group member know how much they currently owe.



$20.99	Electricity
$5.00	Toilet Paper
$7.00	Trash Bags
$10.99	Cable

Summary:

Member 1: $20.99
Member 2: $12.00
Member 3: $10.99
Member 4: $0.00

Unfortunately, there are apps that do similar bill calculations. There's Venmo, a growing app that uses Facebook and email to identify users, and allows them to charge their friends and pay their friends using their debit cards or bank accounts. There's also smaller apps named, Shared Bills and Bill Splitter targeted for calculating costs. However, these apps don't allow group viewing and sharing, which is a strong concentration of Community Costs. These apps focus only on your money relation with other individuals, or forces only one person to keep track of totals in a group, whereas our app encourages communication and shared information amongst everyone.

Software Architecture

Users will need a homepage where they can view their groups. This page could also display information for the user about how much they owe other people and how much is owed to them. Homepages will be unique to each user whereas group pages will appear the same for every member of that group.

The app will be organized using the model-view-controller pattern. The view consists of group pages displaying purchase information and totals for each member of the group and individual pages. Individual pages will allow users to create groups, see the groups they are a part of, and accept invitations into other groups. The controller will respond to user actions such as adding or deleting purchase information. This will then be sent to the model, which will calculate totals and store group and individual information. Users will need to have identification stored in the model so that they can be verified as part of a group and their groups can be remembered. Users will also need passwords to prevent arbitrary people manipulating data of a group they are not a part of.

One useful tool in creating this app could be the android plugin for eclipse. This would be helpful for creating an android app using a familiar language in Java. Another option could be using HTML, CSS, and JavaScript.

Challenges / Risks

One challenge for building this app is concurrency problems. Multiple users will be able to modify group pages at the same time. These requests will need to be handled using locking. One easy way to do this would be to make use of Java’s synchronization libraries. Another challenge is the usability of the app. It needs to be simple to use and clear what the purpose of every feature is. Finding other people and adding them to groups also needs to be easy. This could possibly be done by connecting users to Facebook profiles. The app also requires that everyone in a group downloads the same app, adding to the difficulty of usability. A large risk of the app is security. Users need to be protected with passwords so malicious clients cannot change group information to which they do not belong.