Project Concept: mycheapfriend.com

COMS 4156: Advanced Software Engineering

Team: CheapSkates

Team Members

Double Trouble: Waseem Ilahi (wki2001@columbia.edu)

Shaoqing Niu (sn2385@columbia.edu)

Dynamic Duo: Michael Glass (mgg2102@columbia.edu)

Huning "David" Dai (hd2210@columbia.edu)

The **Cheapskates** from **MyCheapFriend.com** will lend you our **Project Concept** if you give us back feedback next week.

Basic Concept:

I went out to dinner with my friends Rina and Jacob last night. At the end of the meal, Rina didn't have any money. She offered to go run to the ATM, but Jacob loaned her $20 instead. As Jacob leaves for work this morning, the fact that he's loaned Rina money is a distant memory. As Rina walks her dog this morning, she thinks to herself, "oh right, I owe Jacob $20" but failing to write it down, she might not remember the next time they see each other.

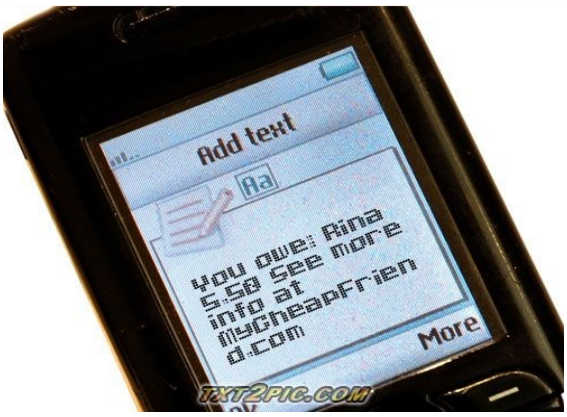
MyCheapFriend.com makes loaning small amounts of money to your friends easy to remember. Flash back to last night: As Jacob is handing $20 bill to Rina, he pulls out his cell phone and texts "rina 20" to his unique and secret e-mail address at mycheapfriend.com (saved in his address book). Rina's cell phone beeps and an e-mail from billing@mycheapfriend.com asks her, "Your cheap friend Jacob (212-555-5555) just said that you owe them 20 bucks. If this is true, reply 'Y' to this text. Otherwise, ignore this message."

The next weekend, three of us (Hi! I'm Michael) go out together. Rina opens a tab at the bar and pays for all of our drinks. She uses mycheapfriend.com to track the expenses pulls out her cell phone and texts, "drinks jacob michael 51" to her unique e-mail @ mycheapfriend.com.

The first of the month comes, and the three of us receive texts from gimmeSomeSugar@MyCheapFriend.com.

Here's what they look like:

Mine and Jacob's Texts:



Rina's Text:

Now that the gist of the application has been outlined we will enumerate all of the interactions we will have to handle:

1. Creating an account

In order to bill friends, you need to create an account. Creating an account involves texting new_account@mycheapfriend.com. new_account@mycheapfriend.com responds with a unique e-mail address to text to. The text looks like this:

"your unique address is <gop1bi@mycheapfriend.com> Please add it to your address book (as CheapFriend?)".

This authenticates users against their unique e-mail address and treats their phone number as a unique user-id.

You may also go to www.mycheapfriend.com to register with your cell phone number and a chosen unique address (as password). Our system sends a request to your cell phone asking for your password to check if you are the real owner. There you can also check your account and look into few other details about the bills.

2 Billing a friend

2.1 Requesting a bill

2.1.1 Billing a friend with a cell phone number

Billing a friend with a cell phone number involves sending a text to their unique address with a message that has an amount and a cell phone number separated by a space. Either the phone number or the amount can come first.

\$?\d{1,4}(\.\d{2})? is a regex that validates an amount

\d{10}|(\D?\d{3}\D\d{7})|(\D?\d{3}\D\d{3}\D\d{4}) is a regex that validates a phone number.

2.1.2 Billing a friend with a nickname

2.1.2.1 Billing a friend with a new/changing nickname

To bill a friend with a nickname, you must first assign a cell phone number a nickname. To do this, you must send message with a cell phone number, a nickname (defined below), and an optional amount (if you want to request a new bill in the same message).

[a-zA-Z]{1,10} validates a nickname

2.1.2.2 Billing a friend with an existing nickname

To bill a friend with an assigned nickname, you simply text the nickname, with the amount. For instance, "Jacob 24". You can also send a message with

1. Multiple nicknames with the total amount of money they owe you and our system will split the bill evenly for them; e.g. "Mike David 100"

2. Each nickname follows by an amount. e.g. "Mike 50 David 50".

2.2 Receiving a bill

When a friend bills you, you receive a text that asks you "Your cheap friend # {friend's nickname followed by cell phone # or just cell phone if you haven't assigned friend a nickname} just said that you owe them 20 bucks. If this is true, reply "Y" to this text. Otherwise, ignore this message." And Note: The grammar mistake "them" is a purposeful substitution for "him/her".

2.2.1 Accepting a bill

2.2.1.1 for non-registered users

A non-registered user can also accept a bill by replying "Y", however, our system will send an additional message asking for the user to register. Something like:"Try out the greatest

service ever - my cheap friend, reply "R" to activate your account.” If the user decided to accept the invitation, the process will be identical to ‘1. Creating an account’.

2.2.1.2 for registered users

Nothing special for registered users, a simple "Y" reply will do the magic.

2.2.2 Rejecting a bill

If no "Y" is sent back, after waiting for a certain period of time, our system will send a "Request refused: #nickname/phone number #amount" message back to your friend, who claimed that you owed him/her money, he may harass you again if he insists.

3. Settling a bill

3.1. Settling a bill by the bill-er

After the user's friend pays him/her back, he sends a message with a nickname and a negative amount to clear the bill. Note: A user might also use this to offer money to his/her friends.

3.2. Settling a bill by the bill-ee

If a user bills a friend who owes him/her money, and that bill is accepted, any balance between the two users is settled before a new "debt" is accrued, i.e., if I owe Rina $5 and I bill her $3, then I still owe her 2. If I then bill her $5, she then owes me $3

4. Bill report

A user can check the report of all non-clear bills either by logging into our website or text "REPORT" to his unique email address.

Component Framework Explanation

Team "CheapSkates" has decided to use EJB as our main framework, possibly version 3.0. We will be working on Windows OS and for the IDE, we have chosen to use Eclipse Java EE.

All the team members are familiar with the Java programming language used in developing using EJB, as compared to the languages used by various other component model frameworks.EJB is compatible across various platforms as supposed to some other well known frameworks, e.g., .net, etc. EJB is quite a well known framework, so the documentation about it and possibly a few tutorials can be found on the net, the official documentation can be found at the following address: http://java.sun.com/products/ejb/docs.html

The latest version of the framework tools can be downloaded by following the link below: http://java.sun.com/products/ejb/

Following are some other helpful links:

http://java.sun.com/developer/onlineTraining/Beans/EJBTutorial/

http://www.conceptgo.com/gsejb/index.html

http://www.roseindia.net/ejb/

We will use PHP/HTML/AJAX for the frontend programming and of course the "JAVA" programming languages for the back end programming, since we are using EJB. The Database will most likely be a MySQL.

Controversies

Originally there was a disagreement about the implementation of the user authentication in the project. There is still some uneasiness among the team regarding the user authentication process proposed to be used. It only needs some team wide explanation to clear out the workings of the whole process.

Other than that, the team has agreed on pretty much everything unanimously and there doesn't seem to be any problems so far.

THE END