trickle

let the money flow.

GROUP MEMBERS

KEVIN MOODY
kmoody@stanford.edu

BEN MITTELBERGER
bmittelb@stanford.edu

ADAM SCHEXNAYDER
schex93@stanford.edu

DESCRIPTION

Trickle is a free-to-use iOS application that allows organizations such as campus clubs, fraternities, sports teams, and small businesses to effortlessly manage their budgets, expenses, and reimbursements.

At its core, Trickle provides a powerful yet intuitive platform to first model how funds and reimbursements should flow through an organization, and then automatically enforce those flows, process reimbursements, and record expenses. Additionally, trickle uses Venmo, a well-trusted mobile peer-to-peer money transfer application, to execute all approved transfers of funds.

Fraternities, for example, are comprised of several committees, each with their own operating budget. When fraternity members make purchases on behalf of such committees, they often require reimbursements. Trickle allows committee members to effortlessly submit reimbursement requests by taking and submitting photos of their receipts. Once any reimbursement request meets custom-defined spending rules, Trickle automatically credits the proper amount to the purchasing member's Venmo account .

More generally, when someone wants to better manage their budgets, expenses, and reimbursements within an organization, they first create an account on trickle. Next, they attach a primary funding source, in the form of a Venmo account, to their organization's profile. From then on, they, and any other administrators they add, have the ability to create groups within the organization's account and transfer funds with custom-defined spending rules, such as "a group's total reimbursements cannot exceed $1,000 per week" or "any individual reimbursement over $100 requires special approval by a finance committee admin." Ultimately, members of groups within the organization then have the ability to spend and be reimbursed hassle-free within their group's designated spending limits.

NEED FOR TRICKLE

Any functional organization understands how to set up annual budgets with reasonable financial restrictions. However, we have found that most small to mid-sized organizations do not have the resources to give their members direct access to their primary bank account (like company credit cards) which means the individual members must use their own money for group purchases and get reimbursed by the organization later. Here lies the problem. Most groups have some pen and paper system that requires the buyer to keep every receipt, fill out a form, and bring both to the organization's financial officer. Then the financial officer has to eventually manually go through every reimbursement request, retroactively check that the purchase fit within the budgets, manually update the budget spreadsheets, and finally cut a check to the purchaser. This is obviously a sluggish system that requires valuable time from all members. Additionally, many logistical issues easily arise, like lost receipts or discrepancies between multiple budget spreadsheets. Even more importantly, this system has no way to regulate who buys what. The organization has already budgeted all this out and the info can be derived from some spreadsheet that not all member even have access to. Thus the organization has to spend copious amounts of manpower to monitor that they are keeping with their budget.

Trickle's simple-to-use mobile platform simplifies this endeavor by automating expenses, regulating how much people can get reimbursed, and opening up the proper communication channels between purchasers and approvers (like the financial officer).

AUDIENCE

We think that this product would be used by small to medium sized organizations that have groups or committee systems where people in said groups must report back up to some higher financial board or treasurer. For example, a soccer league may have many teams all with a set amount that each can spend specifically for traveling purposes, and every team must first get permission from an administrative financial board before purchasing plane tickets. These groups have expenses that range from equipment, to food, to miscellaneous invoices. Their size means that managing reimbursements requires significant effort. Their casual nature also denote that they are most likely unwilling to pay for a heavy-weight solution like Concur (an enterprise software solution for managing expenses, discussed more in the Competition section).

TECHNOLOGIES USED

INTRODUCTION

Ultimately, trickle would be a mobile application that is backed by a web admin interface. Due to the ten week time restriction, however, the initial product will be built as an iOS application. Its architecture will consist of five primary components: (1) iOS client, (2) API server, (3) cache server, (4) primary database server, and (5) long-term data store server. The following sections will describe each component in details and how they interact.

COMPONENT 1: IOS CLIENT

The iOS client will form the user-facing product. It is responsible for allowing the user to interact with historical data, create new payment rules, and file reimbursements. The codebase will consist primarily of presentation logic and user interaction flows. It will be written primarily in Swift, with occasional Objective-C as necessary, on xcode. The client application will be tested on iOS simulators and actual devices. It will communicate with the API server with JSON over HTTPS.

COMPONENT 2: SERVER

The API server will serve as the core business logic server, enforcing financial rules and serving as the liaison between all present and future clients (iOS, Android, web, etc.). The API server will be hosted within an Ubuntu image running on an Amazon Micro T2 EC2 instance. The web server will be written in Node.js using the Express.js framework, daemonized by the forever javascript library, and served behind the nginx reverse proxy.

COMPONENT 3: CACHE SERVER

The cache server will be used to provide reduced latency responses for common and repeated queries and operations. It will be implemented as a Redis in-memory data store hosted on Amazon ElatsiCache.

COMPONENT 4: PRIMARY DATABASE SERVER

The primary database server will serve as the central relational store for all data pertaining to users, groups, organizations, permissions, financial rules, transactions, and reimbursements. It will be implemented as a PostgreSQL database hosted on Amazon RDS.

COMPONENT 5: LONG-TERM DATA STORE SERVER

The long-term data store server will handle storage of large files, such as the images of receipts attached to transactions and reimbursement requests. Amazon S3 will serve as the object storage server.

ADDITIONAL LIBRARIES

The Venmo API will be used to process authorized payments. Node-postgres will be used as an adapter to connect to the primary database, and Sequelize will be used as the object relational mapper (ORM) that maps entities to database tables. Node_redis (backed by hiredis) will be used as an adapter to connect to the cache server. Tesseract open-source OCR engine, created by Google, may be used to provide receipt-reading capabilities to the reimbursement submission process.

COMPONENT INTERACTION

During a typical request flow, the iOS client will send a JSON API request over HTTPS to the API server. The API server will process the request, and if necessary, request the appropriate data from the cache server. If the cache server does not contain such data, the API server will then request the data from the primary database server. Finally, the API server will send the formatted JSON API response via HTTPS back to the iOS client. In some cases, such as during asset or image retrieval, the API server may direct the iOS client to request files directly from the long-term data store server.

RESOURCE REQUIREMENTS

In order to build and run trickle on the iOS platform, Apple developer accounts would be required. Additionally, Amazon AWS credits may be necessary to power the EC2, RDS, ElastiCache, and S3 instances if the product requirements exceed the respective free-tier capabilities.

COMPETITION

Venmo is a peer-to-peer payment app that utilizes Facebook users' social networks to facilitate virtual payments. Venmo users have a virtual balance attached to their account that they can use to pay other Venmo users by transferring funds between accounts. Users can link their bank accounts to their Venmo profiles in order to make withdrawals and deposits to and from their Venmo balance. Whereas Venmo only focuses on direct peer-to-peer transferring of funds, trickle focuses on helping organizations transfer funds from a central account to members' individual accounts. Thus Venmo's audience is individual consumers while trickle's audience is mid-sized organizations. That being said, both apps utilize the ability to transfer funds between users, which is why trickle will use Venmo's API's for for this exact functionality. One way to think about our app is that trickle is an organization wrapper around Venmo's core functionality; however we want to design trickle in a way such that it could wrap around any payment software (such as Paypal).

Concur is an enterprise software application that enables large corporations to better manage the creation and repayment of expense reports made by their employees. In essence, Concur has both the payment and organizational functionality that trickle would have. However, Concur is designed for large and complex corporations while trickle is designed for much smaller and simpler groups. This cause the two to have several major differences. First, Concur is enterprise software that requires hefty licensing fees to use. This makes sense because Concur has thousands of great features with customer support teams for all of them. However, medium-sized organizations would not need most of these features and would not want to waste limited funds on them. Similarly, trickle focuses on ease of use, but Concur focuses on customizability. Again this makes sense only for large companies. Overall Concur is a wonderfully powerful product but simply too complicated and expensive for our audience.

Splitwise is an app that allows members of a group to very easily track who owes what to each other. This is targeted specifically for roommates and trip members. Not only does it track expenses for the group, but you can also use Venmo or Paypal to "Settle Up" and pay whoever you owe money to. Splitwise is similar to trickle in that it tracks expenses and utilizes payment apps to facilitate the transfer of funds accordingly. However, it does not offer any hierarchical configuration for mid-sized organizations. This is strictly for groups where every member is on the same level and there is no centralized bank account.

POTENTIAL APPROACHES

Expense and reimbursement systems are complicated no matter what type of organization you are a part of; furthermore, there are countless approaches that different groups use. However at the end of the day, most groups fitting our target audience still use nothing but spreadsheets and paper forms. As discussed in the Need section, this gets tedious very quickly. Furthermore, once you have to start breaking down budgets into more detail across several different spreadsheets for different subgroups or committees, it can get very complicated very quickly. Cross checking spreadsheets takes time, most of which usually falls on a single individual. Trickle will enforce rules so that the mult-spreadsheet problem is simplified down to users submitting a request, and getting a direct answer. Additionally members of an organization almost never even look at the budget that they are suppose to be following. Trickle will show users exactly how much they can spend before they make the purchase. Ultimately, trickle simplifies and automates over the classic spreadsheet system.

RISKS

One risk is that the Venmo API's could be difficult to use. None of the group members have had to use Venmo API's before which means we are making many assumptions on their functionality. As of right now, it is very unclear if trickle will have enough permission to make automated payments on behalf of the user. Additionally, there are several security risks when storing user's Venmo information since it has access to their bank accounts. We will need to be very careful where we store Venmo account information.

There are some risks with relying on Amazon EC2 and Amazon S3 because they are not machines that we have 100% control of. That being said, Amazon does have a great reputation in this field and far more experience than we have. However, we will have to trust that Amazon's machines never go down or our app will too. Additionally, we will be storing very sensitive information on these machines, so we will also have to trust Amazon's security efforts.

NEXT STEPS

To begin work on the product, provisioning profiles for each of the developers will first need to be created. This way, the product can be downloaded and run on iOS devices. Simultaneously, all of the Amazon AWS instances will need to be initialized and configured. At this point, work can begin from multiple angles. Design sketches will be created to storyboard the client iOS application, database tables will be constructed in PostgreSQL, and business logic API calls will begin to be fleshed out.