SimpleScan Final Documentation

CS 428/429

Xiaoming Chen

David Li

Daniel Pang

Jaehwan Park

Tyler Pearse

Tai Lin Wu

Table of Contents

Overview

Everyone has bills to take care and expenses to track. Many times, an individual's first exposure to such responsibility occurs during young adulthood. It is also highly likely during this time that those bills and expenses need to be shared or split amongst friends and/or roommates. SimpleScan is a mobile application with the intention of providing a simple and effective way of managing one's expenses, while allowing easy sharing of those expenses with others who are involved. Users can view previous expenses, maintain a budget, split bills, set bill reminders, and, utilizing our photo recognition software, can even scan receipts and bills right into the application for better management of finances.

We chose this project because of the challenges involves. A simple interface for managing personal finance seemed trivial yet beneficial for people in our current demographic. But more importantly we wanted to provide a means of easily documenting those expenses as well as sharing them. Our application's goal was to deliver these crucial features to the user, and hence became the focal point of our project.

Android is the target platform for this application because of the importance of mobile integration with our users. Similar applications and software seem to stray more towards a central web application in which users had to navigate to in order to properly utilize it. By choosing a mobile platform, users would be able to access the application whether they were at the grocery store or just leaving best buy.

The Process

For this project, our team chose extreme programming (XP) as the software development process. Our first step was to brainstorm user stories, and choose the features that we valued most in the application. We then split the user stories into use cases by narrowing them down to individual tasks. For each task, we decided upon a priority and time estimate, then planned out our future iterations on a roadmap. This step helped us to always have a goal for the set of use cases completed after each iteration, and follow XP's idea of early delivery and continuous iterations of improvement.

During the development phase, we split our group into smaller self-organizing groups based around a central theme in the application. We had three main groups: camera features, back-end data, and cloud sharing. These features were logically independent, and allowed us to split into pairs to code for a specific group. At the end of the iteration, we would then collaborate and merge code, as well as discuss what needed to be done for the following weeks. This process efficiently guided us to quickly ship out our basic features, since XP relies more on working code than heavy documentation. As we continued with each iteration, each group went back to look at previous code to refactor where appropriate so that our codebase would remain clean and concise.

One of the issues we had with collaborative development was the process of integrating our work together. Since our teams had mostly independent areas of focus, the knowledge and experience we gained was in a large part different than the rest of the group. Thus, there were sometimes difficulties in understanding the code of others when trying to merge features together. However, in the beginning we predicted that a problem like this may arise, so our team made sure to map out a clear central design and focus of the project. We planned out nearly all our features that we felt our application should have, and divided work in such a way so that the code we needed to merge was mostly exclusive from the code of others. Thus, only a basic understanding of the code of others was needed to correctly merge features together.

Another one of our issues was test-driven development. We started with the focus of having appropriate back-end tests to cover all the processing that our application required. When we reached the step of needing GUI tests, the portion of the code we had written in Android fragments did not have a viable framework for us to test with. Though we spent hours trying to figure it out, Expresso (the framework we used for testing) did not mesh well with our code design and we ended up only having simple GUI testing. However, we mitigated this damage to TDD by allocating more time to back-end testing and user testing to ensure that our application was working up to our standards.

Overall, choosing to follow the XP process really helped move our project forward. We felt the drive to put out new features as quickly as possible, and were motivated to always improve our existing codebase. Having different teams work in different areas helped us to have each part of our application stand out, since every team could focus on learning a specific part of Android that they were interested in, and make that feature as fluid as possible. At this point at the end of the semester, we are satisfied with SimpleScan and proud of our work as a whole.

Requirements & Specifications

As we have mentioned before, our goal is to make an application that handles the financial balance, keeping track of money, sharing the expense with multiple people. We use photo recognition feature to get the expense by taking photos of receipts. Below (Figure A) are the lists of user cases and associated user stories to specify what we implemented for this app.

Figure A

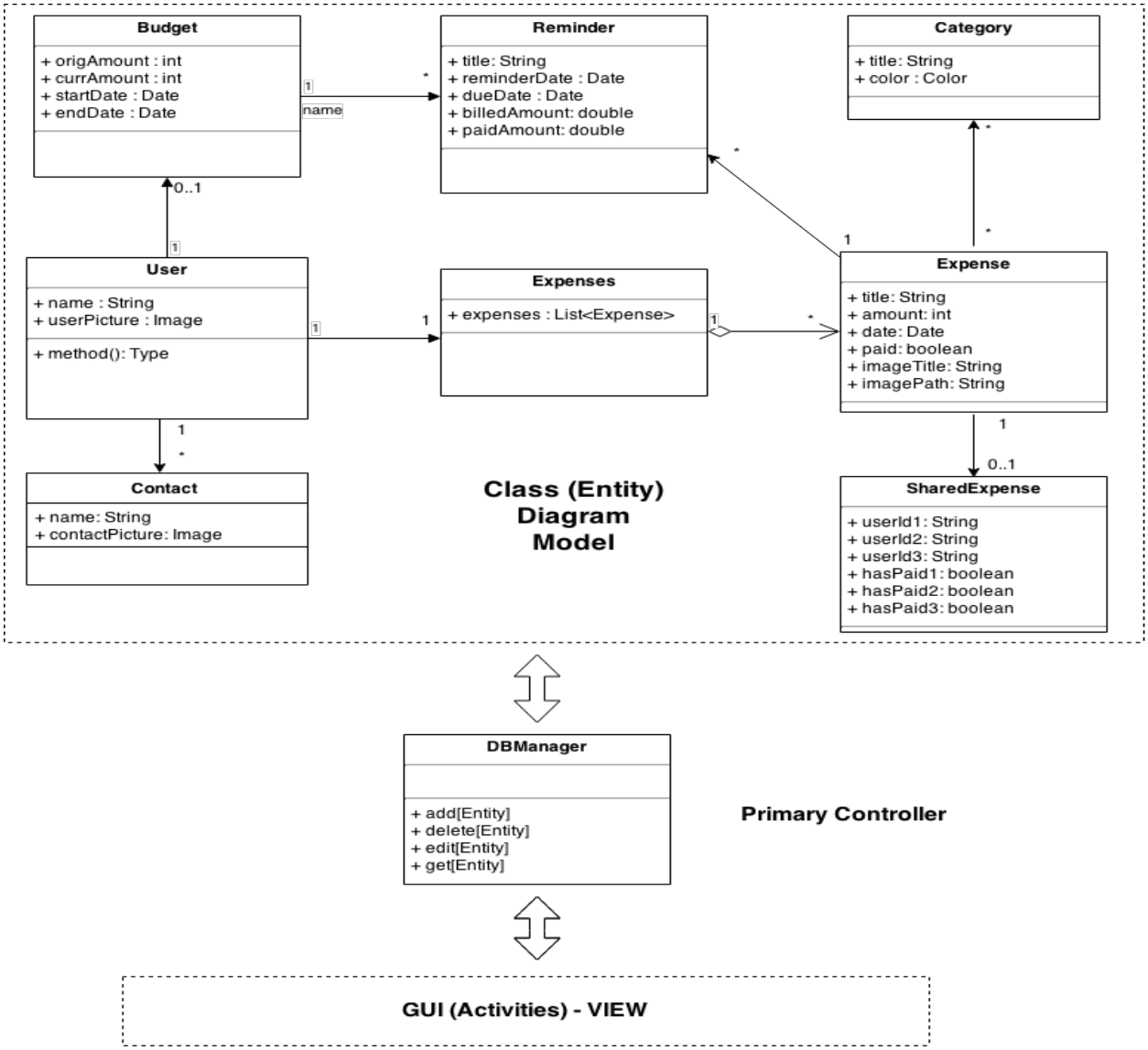
User cases	User stories
Take a picture of a bill	As a user, I want to scan my bills and receipts for expense totals.

Log an expense	As a user, I want to retrieve data from receipt picture and transfer it to the edit expense page.
View logged records	As a user, I want to input and save my gains/expenses. and images on the app.
Categorize expenses	As a user, I want to add new categories for the expenses.
Register an account	As a user, I want to set up a basic profile.
Set a budget	As a user, I want to set a budget.
Set reminders	As a user, I want to be able to set reminders for payment due dates.
Add/remove contacts	As a user, I want to view my contact list and add / remove contact to it.
Check notifications	As a user I want to view pending contact requests.
Share expenses with someone	As a user, I want to view all shared expenses.
Accept/reject payments	As a user I want to add/remove contact requests others have sent me.

Architecture & Design

The main design focus of our application was centered around the use of the MVC (Model, View, Controller) design pattern. Our "model" representation consists of database tables and entity classes for storing information. The "view"" is comprised of several activity and fragment classes with the purpose of providing interaction and viewing of information to the user. These two components are tied together through our "controller" class, `DBManager`, which provides the passing of data back and forth between the model and view. Figure B below provides a high level overview (with more detail provided for the model) of our application's architecture.

Figure B.



Being that this application is for mobile users (in this case Android), the decision for what architecture design to implement was fairly limited. All of android's user interface is handled through activities and fragments. For back-end storage, we had the option of using files or sqlite. We decided that sqlite would provide a much more cleaner way of organizing our information. These two aspects of the android framework made it an easy decision to create a centralized controller class to handle the interaction between the view and model. Not only did this make our project better organized but it also allowed ourselves to work on the different components separately without interfering with the others.

Personal Reflections

Xiaoming Chen

I am really grateful to have a semester-long android application development experience with a group of talented people. There is a doubt at the beginning of this semester that we can accomplish this complex application in one semester regarding the fact that most of our group members don't have any experience on the android development and the complex features such as ORC. However, we have a good team atmosphere and most of the group mates are self-motivated, learning while developing. The leadership has been rotated among each group members and the responsibility of the each iteration has been switched in order to fit the adjustment of the whole project. Each person in this group has their own expertise, while developing the android application, I found I have learnt a lot from each individual group member, and I have been benefited from this semester project.

Personally, I found this project is challenging and interesting in many ways. At the beginning of this semester, I don't have any experience on how to construct a good android architecture. However, through the development, I have implemented the fragment oriented architecture for our application. Understanding the lifecycle of the parent activity, child fragment activity and how to combine multiple fragment classes inside a master activity class without conflict and data misplay are challenging. However, the evolution of Android SDK seems to be advocating the use of Fragments more and more, so in order to make sure our application follow this trend, it's necessary to use "fragment oriented" architecture rather than "Activity oriented" architecture even though the life cycle is more complex. Besides, the application architecture, I work on multiple fragment classes and activity development, SQLite method implementation and some integration of OCR. With all the help from each team member, I have a clear picture of the whole android development from the SQLite database, backend, frontend and even interaction of the cloud environment and cloud base data consumption. At the end, I would say I am really luck that I could have a chance to work with and learn from a group of talented and hardworking people.

David Li

I really appreciate what I've learned throughout the course of this project. I feel like what has made this project the most exciting for me was the flexibility given to us students. With no strict requirements, each of my team members could work on the feature they most enjoyed for this project, and allowed us to enjoy the class and learn in an optimal environment. Beginning this semester, I would never have imagined myself understanding so much about Android programming. Though I was familiar with Java in the past, I didn't really know how to transition it to mobile. With the help of teammates, it made it simple and fun to get a product up and going without too much burden laid on any single member.

I personally was in charge of the interactions with our cloud environment, such as the contact lists and sharing of expenses. I learned a new Cloud database system called Android Parse which I believe will be a useful skill in the future, as well as how to integrate my code with a back-end database. I also learned general UI for the Android system, and the layouts and views that were so confusing to me in the beginning of the project now seem trivial. Most importantly, I learned how to maintain the XP process with my team, and how to work with everybody's strengths. As a team, we did a great job getting along and helping each other to where we are now, and learned much more than we could have in a traditional CS course.

Jaehwan Park

Before this project begins, I had absolutely no knowledge on either photo recognition or OCR (optical character recognition). I barely knew about the cloud system as well. I also had no idea what XP programming was since I didn't take cs427 before. Throughout the project, I have worked on those features and have learned many things about them. Although there were some troubles and mistakes, my teammates and I made it this far and now I have a better idea of how to be a team player and what to do for the success of a team.

For the future, I will look at our project and see what could've been done better. As a team, we may have not always made the best decisions but from our mistakes we can learn what to do and what not to do. I will also keep the lessons of XP in my mind. I appreciate that I could do the project with my teammates. I was lucky to have them who had both skills and passion.

Tyler Pearse

I found this project to be both interesting and challenging in several ways. First, I have never fully worked on a long-term android application. With that said, I had to rush to teach myself basic android development in a short window of time. As we were all getting up to speed with android, we all seemed to fall into different areas. Having worked with MySQL in the past, I decided I would work with the backend development of the project. This primarily included android's SQLite database system. For nearly the entirety of the project, I focused on handling the interactions between the front-end and back-end. While others were developing the user interface and camera functionality, I was focusing on making sure the passing of information back and forth between the database and UI was working properly. This greatly describes our separation of responsibility with this project. Each of us had separate components we were working on which allowed for proper encapsulation. With that said, I feel that this taught us all a good lesson in software engineering.

As far as group cohesion, our group definitely grew quite a bit throughout the semester. Most groups I have been apart of has had one member who makes all the decisions and generally is unwilling to allow others to take the lead when necessary. Our group was quite the opposite. At first, it seemed everyone had conflicting viewpoints and ideas for what direction to take with the project. However, we were all able to come to agreements and we began working together very efficiently. Throughout the semester it was apparent that we all were rotating leadership and responsibilities. One member might take

the lead for a few weeks and then someone else might step up for the next. This was all done without even discussing it which made for a great working environment. Overall, I am glad to have worked with this group and feel that we have all learned quite a bit.

Tai Lin Wu

This project experience was very fun and rewarding. At the beginning of the semester, developing a software from the ground up was still new to me. I did not know the essential parts of the project, such as necessary skills and required tools. However, through the development, I've learned a lot about how to develop software that interacts with hardware. For instance, I have implemented the optical character recognition feature for the app. Although I had some experience with image processing, applying that knowledge to a project was something I had never done. Throughout the semester, I did not only learn how to develop a simple camera app with basic features, but also some editing and recognition tools for the pictures taken. Furthermore, through the implementation of the reminder feature in the app, I've also gained much knowledge in both front-end and back-end coding; I learned how to manipulate the alarm/RTC in the phone, and I was also introduced to Google's new material design. This kind of learning experience was more rewarding than attending any lecture or reading any book.

Moreover, I was also very fortunate to become part of this project development team. It did not only present an exciting challenge for me, it also gave me a pleasant development environment and a enjoyable development process. Even though no one had a solid idea about how the project would become or how the development would end up during the final turn-in, we were able to communicate with each other to compromise everyone's idea. In addition, every member was not only hardworking, but also responsible. At the end, we are all proud of the product we have.