Shortcuts Design Document **Ver 1.0**

Team: Shortcuts

Team Members: Stephen Bowder and John Hibbert

CSC305 Spring 2015

University of Rhode Island

Dr. Mello-Stark

Table of Contents

[1: Introduction 3](#__RefHeading__28976_692392234)

[1.1 Product Overview 3](#__RefHeading__28978_692392234)

[1.2 Description of Design Document 3](#__RefHeading__28980_692392234)

[2: Summary of Specifications 5](#__RefHeading__28982_692392234)

[2.1 Specification Overview 5](#__RefHeading__28984_692392234)

[2.1.1 Account Creation Workflow 5](#__RefHeading__28986_692392234)

[2.1.2 Account Update Workflow 5](#__RefHeading__28988_692392234)

[2.1.3 Account Delete Workflow 5](#__RefHeading__28990_692392234)

[2.1.4 View Pictures Workflow 6](#__RefHeading__28992_692392234)

[2.1.5 Add Pictures Workflow 6](#__RefHeading__28994_692392234)

[2.1.6 Delete Pictures Workflow 6](#__RefHeading__28996_692392234)

[2.2 Management Issues 6](#__RefHeading__28998_692392234)

[2.3 Risk Analysis 7](#__RefHeading__29000_692392234)

[2.4 Constraints 8](#__RefHeading__29002_692392234)

[3: Detailed System Design 9](#__RefHeading__29006_692392234)

[3.1 Data Entry Screen Snapshots 10](#__RefHeading__29008_692392234)

[3.1.1 Main Screen 10](#__RefHeading__35917_692392234)

[3.1.2 Create New Account 11](#__RefHeading__35919_692392234)

[3.1.3 Select Account 12](#__RefHeading__35921_692392234)

[3.1.4 Search Accounts 13](#__RefHeading__35923_692392234)

[3.1.5 View Picture Gallery 14](#__RefHeading__35925_692392234)

[3.2 Report Layouts 15](#__RefHeading__29012_692392234)

[3.3 System Interface 15](#__RefHeading__29014_692392234)

[3.4 Software Design Patterns 15](#__RefHeading__29016_692392234)

[Fig 1. Class Diagram 16](#__RefHeading__35915_692392234)

[4: Detailed Managerial Review 17](#__RefHeading__29020_692392234)

[Appendix A: CRC Cards 20](#__RefHeading__29022_692392234)

[Appendix B: Sequence Diagram 21](#__RefHeading__29026_692392234)

# 1: Introduction

# 

## 1.1 Product Overview

The application ‘Shortcuts’ is a customer relationship management application for Android tablets primarily targeting salons, barbers, and other grooming services establishments. This product fits into the user's organization by facilitating customer satisfaction. The user (the proprietor, rather than the customer of the business) can create an account for each of their clients with their name and email address. Once the account exists, the user can take pictures of their results (haircuts, nails, etc.) with the device’s integrated camera and associate that with the account. The user can review those pictures in the future by looking up the client's account.

## 1.2 Description of Design Document

This document will contain four primary parts. This first section is an introduction, which includes an overview of the product and an overview of this design document. The second section summarizes the specifications of the product. They include the tasks the system can do, the constraints the project will operate under, how the project will be managed, and an analysis of possible risks. The third section is an explanation of how the program is to be created, including an explanation of how the modules and parts relate. The fourth section is reserved for the managerial review by the team lead. The document is followed by appendices of miscellaneous diagrams.

|  |  |
| --- | --- |
| 1. Introduction | Provides information related to this document. |
| 1.1. Product Overview | Describes what services the product will provide. |
| 1.2. Description of Design Document | Explains the purpose of this design document and design documents in general |
| 2.1. Specification Overview | Describes the specifications the software must meet. |
| 2.1.1. – 2.1.6. Critical Workflows | Demonstrates the step-by-step connection between modules needed to conduct critical functions. |
| 2.2. Management Issues | Describes matters of scheduling and the division of labor. |
| 2.3. Risk Analysis | Considers possible scenarios that may put the project at risk. |
| 2.4. Constraints | Describes restrictions that the project is subject to, such as deadlines and hardware restrictions. |
| 3. Detailed System Design | Describes the modules or objects of the project in such a way that it can be created from the documents. |
| 3.1.1. – 3.1.5. Data Entry Screen Snapshots | Displays mockups of what the software will look like and describes the functionality present. |
| 3.2. Report Layouts | Explains the means by which the application will store its information. |
| 3.3. System Interface | Describes what systems the application will be interfaced with. |
| 3.4. Software Design Patterns | Explains which design pattern the project will be utilizing. |
| 4. Detailed Managerial Review | Explains the project lead’s perspective on now things went and describes the division of labor in creating this document. |
| Appendix A | Displays the CRC Cards for the project. |
| Appendix B | Displays the Sequence Diagram for the project. |

## 

# 2: Summary of Specifications

## 2.1 Specification Overview

The Android application will allow the user to navigate between three screens. The account screen will display all the existing accounts to the user, and give the user the ability to display further information about any particular account. From there, the user can navigate to the account's gallery, which will display all the existing pictures for that account. From there, the user can navigate to a full screen version of any given picture.

### 2.1.1 Account Creation Workflow

This critical system workflow is how the system creates new accounts for clients.

2.1.1.1 Main\_Activity takes the strings given by the user and sents them to Account\_Manager.

2.1.1.2 Account\_Manager gets an accID from Data\_Manager.

2.1.1.3 Account\_Mananger creates an Account object with the given information.

2.1.1.4 The Main\_Activity displays the account list.

### 2.1.2 Account Update Workflow

This critical system workflow is how the system updates the information of an existing account.

2.1.2.1 Main\_Activity takes the strings given by the user and the ID of the selected account and sends them to Account\_Manager.

2.1.2.2 Account\_Manager calls Data\_Manager and gives it the strings given by Main\_Activity.

2.1.2.3 Data\_Manager updates the fields for the given Account object.

2.1.2.4 The Main\_Activity displays the account list.

### 2.1.3 Account Delete Workflow

This critical system workflow is how the system removes an account.

2.1.3.1 Main\_Activity takes the ID of the selected account and calls Account\_Manager.

2.1.3.2 Account\_Manager calls Data\_Manager.

2.1.3.3 Data\_Manager removes the information from the fields in the selected account.

2.1.3.4 The Main\_Activity displays the account list.

### 2.1.4 View Pictures Workflow

This critical system workflow is how the system displays a gallery for a particular client.

2.1.4.1 Main\_Activity takes the ID of the selected account and calls PhotoViewer\_Activity.

2.1.4.2 PhotoViewer\_Activity takes the ID of the selected account and finds its photos.

2.1.4.3 PhotoViewer\_Activity displays the pictures.

### 2.1.5 Add Pictures Workflow

This critical system workflow is how the system adds pictures to a particular accounts.

2.1.5.1 PhotoViewer\_Activity calls the Camera\_API.

Camera\_API uses the built in camera class to take a picture.

2.1.5.2 Once the picture is taken by the user, Camera\_API returns the picture to PhotoViewer\_Activity.

2.1.5.3 PhotoViewer\_Activity displays the pictures.

### 2.1.6 Delete Pictures Workflow

This critical system workflow is how the system removes pictures from a given account.

2.1.6.1 PhotoViewer\_Activity takes the ID of the selected account and the selected photo.

2.1.6.2 Photo\_Mananger deletes the selected photo from the account.

2.1.6.3 PhotoViewer\_Activity displays the pictures.

## 2.2 Management Issues

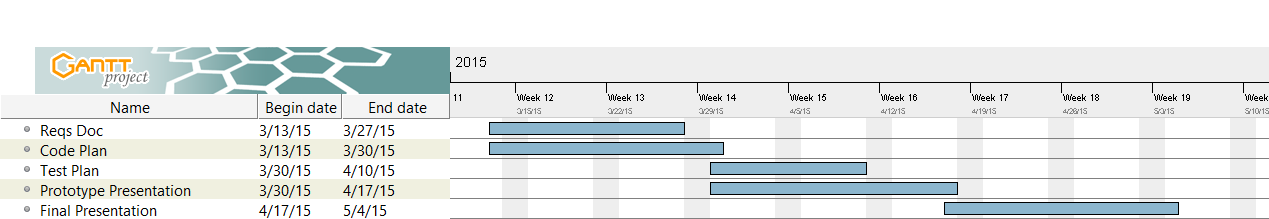
Stephen will be responsible for Account, Main\_Activity, Account\_Mananger, and Data\_Manager. John will be responsible for PhotoView\_Activity, Photo\_Manager, and Camera Wrapper.

As the team is learning Android Studio during the development time, delivery may be affected if the team cannot become proficient in the allotted time period.

**Schedule**

|  |  |
| --- | --- |
| March 27th | Requirements Document Revision |
| Mar 30th | Code Plan |
| April 10th | Test Plan |
| April 17th | Prototype |
| May 4th | Project Due |

**Gantt Chart**



## 2.3 Risk Analysis

This project could be put at risk in more than one way, but one is far more likely than the rest. As this project has two-member team, if one member were to be rendered unable to contribute further work, that could put the project in serious danger of failing to meet its deadlines. In that event, it could be mitigated by hiring a new team member or increasing the overtime budget for the solitary remaining team member.

Another possible problem that this project could face this is some sort of large scale change to the licensing of the integrated development environment that this project will use: Android Studio. Currently, Android Studio is licensed under Apache 2.0, a free software license that allows its use for any purpose without royalties. If this license were to be rescinded and replaced with a proprietary license that would incur a cost to the team, the project would be great jeopardy. The project would have to be re-written in Java. Time would have to be taken to find Java libraries to replicate certain functionality, and if they could not be found, they would have to be written by the team or the application could be redesigned to adjust for functionality changes.

## 2.4 Constraints

This project will be operating under some constraints. The deadline for this project is May 4th, 2015. It will be programmed in Android Studio using Android 4.0.3 (Ice Cream Sandwich, API 15), for Android mobile devices. Due to the nature of Android’s version-based releases, some older Android devices will have previous versions of the operating system. Those devices may or may not have the necessary hardware to run newer versions of the Android operating system. If they cannot, the older operating system may be lacking in certain necessary features and functionality to make the application work. Therefore, though the application might work for other versions of Android, the scope of this document is constrained to 4.0.3 or newer.

By the Android 4.0 Compatibility Definition, if the device is to have camera, it must have a resolution of at least 2 megapixels for rear-facing cameras and 640x480 for front-facing cameras.

# 3: Detailed System Design

This application is composed of two modules: the Main\_Activity and the PhotoView\_Activity.

**Main\_Activity:**

- Data\_Manager loads stored information pertaining to client accounts.

- Main\_Activity creates an instance of ExpandableListAdapter to display returned information from Data\_Manager instance.

- Main\_Activity calls ExpandableListAdapter’s ParentFilter subclass to search and filter list.

- Main\_Activity calls Data\_Manager to add / edit / delete Accounts.

- Main\_Activity passes a selected account index to PhotoView\_Activity module.

**PhotoView\_Activity:**

- PhotoView\_Activity interfaces with Photo\_Manager to create and display a list of photos

associated to the Account passed as initial argument.

- PhotoView\_Activity calls FullImage\_Activity to display selected image.

- FullImage\_Activity instantiates ImageAdapter to display passed (by index) image.

- PhotoView\_Activity calls Camera API instance to take a photo, Camera API instance then

passes the new photo to Photo\_Manager to be added to the associated account’s list of

photos.

- PhotoView\_Activity passes a selected photo to Photo\_Manager to be deleted from the

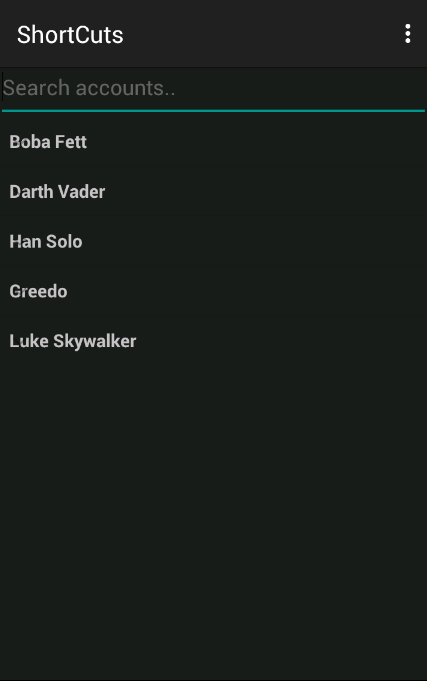
account’s list of photos.

- PhotoView\_Viewer returns back to the Main\_Activity module.

## 3.1 Data Entry Screen Snapshots

### 3.1.1 Main Screen

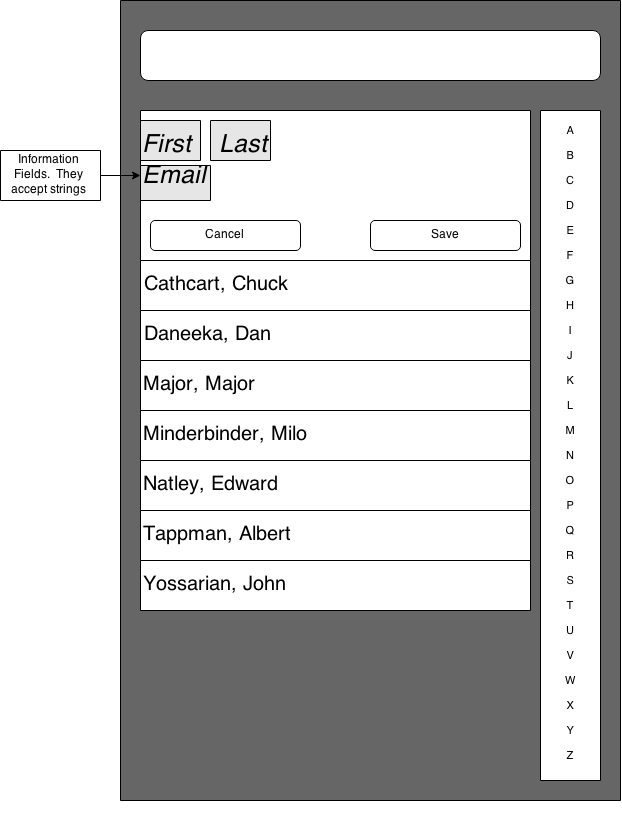
This is the main screen. It displays the existing accounts, alphabetically sorted by last name. The visible attributes of the accounts (first name and last name) are contained within each account object. It also shows the search bar at the top and the alphabetical heading shortcuts on the right. The top of the list is populated by an option to create a new account.



### 3.1.2 Create New Account

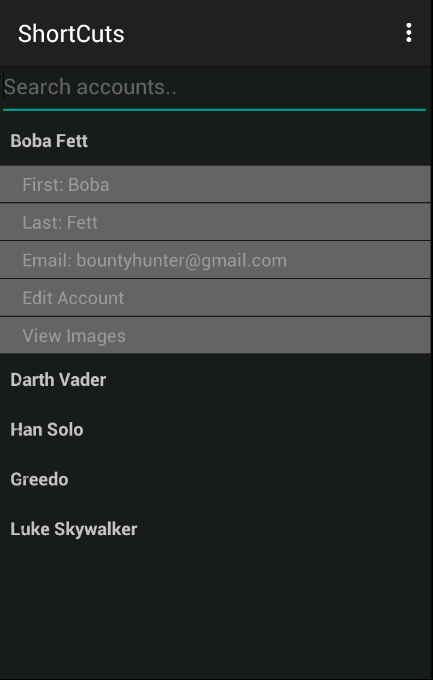
When the Create New Account heading is selected, it expands into three fields and two buttons. The fields will hold the account’s information. Each field accepts strings. The save button adds the new account to the list, while cancel discards the information. If the information is saved, the account it added to the list so that the accounts are maintained alphabetically by last name, thereby keeping the list sorted.

This feature is unimplimented as of this release, so the screenshot below is a mock up.



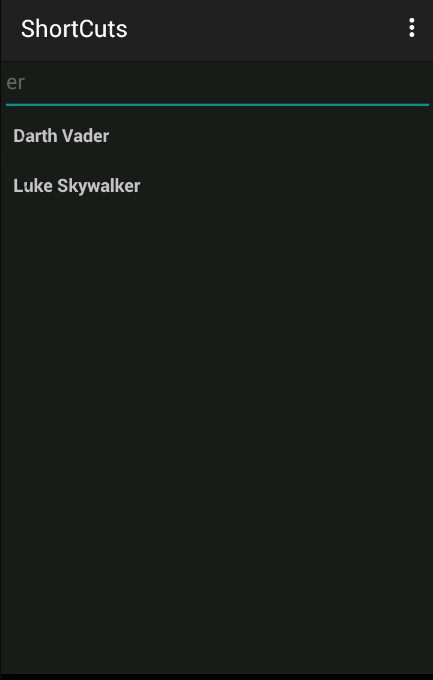
### 3.1.3 Select Account

Selecting an existing account expands to show the information in that account, as well as two buttons to interact with. Like the first and last name, the email address is kept within the account object.



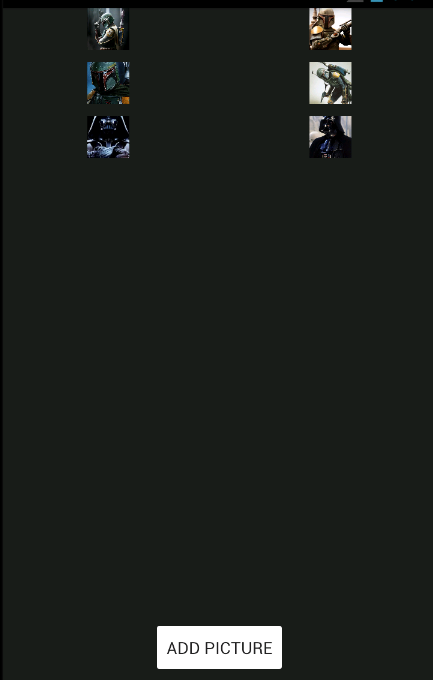
### 3.1.4 Search Accounts

Typing a string into the search bar restricts the visible accounts to accounts that contain that string in their first name, last name, or email address.



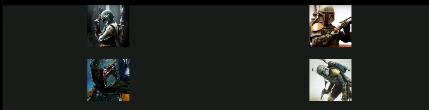
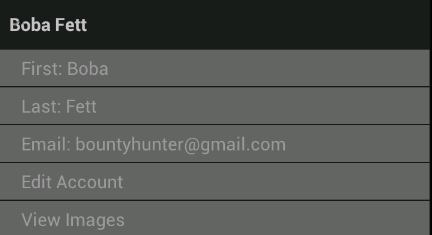
### 3.1.5 View Picture Gallery

In the picture gallery, a set of thumbnails is shown. If a user taps a picture, they gain the ability to delete the picture or to enlarge it to fill the device’s screen. Tapping an enlarged picture will restore the gallery view.



## 3.2 Report Layouts

User data will be stored on the device’s internal or external storage, dependent on availability. The scale of this project does not warrant the use of a database.



The fields for first\_name, last\_name, and email are all strings. They will exist as variables of the account object. The photos for each account will be a list of objects associated with the account object.

## 3.3 System Interface

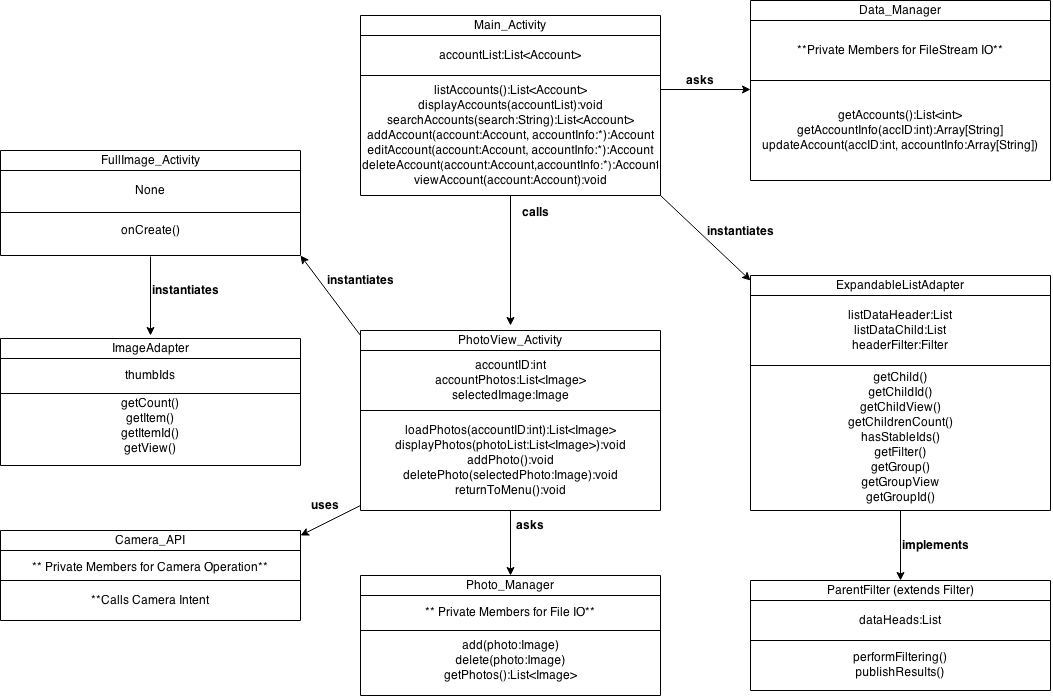
This application interfaces with mobile devices’ camera systems. A wrapper class will be used to manage the different functionalities of the camera systems, while creating an interface for the involved modules to easily access.

This application will contain no initial data. It will rely on the user to input all its data.

## 3.4 Software Design Patterns

The most commonly used design pattern in the application is the Adapter design pattern, as found in ExpandableListAdapter and ImageAdapter. Using this structural design pattern allows us to capitalize on base classes found in the Google API, by enabling us to wrap the base classes in such a way that our created classes can interface with them easily. The advantage here is not only the inherent ability to keep our code very modular by using adapters, but also the vast amount of code reuse in place.

### Fig 1. Class Diagram



# 4: Detailed Managerial Review

|  |  |  |
| --- | --- | --- |
| **Section of Document** | **Contributing Team Member(s)** | **Describe work each team member completed in detail** |
| Table of Contents | John (100%) | This doesn’t really count as work, since the word processor does it automatically. (Again, thanks to David for that tip.) |
| Introduction | John (100%) | This is adapted from the original description from the first version of the Requirements Document. |
| Specification Summary | John (75%)  Stephen (25%) | Most of section 2 was written by me, but fair chunks of it were written by Stephen during a meet up. I basically fleshed them out after commentary from David. |
| System Models reflect requirements | John (50%)  Stephen (50%) | He wrote up the Detailed System Design. I did the Data Entry Screenshots. We did 3.2, 3.3 and 3.4 collectively. Stephen did the class diagram. |
| Sequence Diagrams | Stephen (100%) | We discussed this, but he did 100% of the actual work on the sequence diagram. |
| Class Diagrams | Stephen (100%) | Same as above. We discussed this, but it would be wrong for me to take any serious amount of credit. He made the class diagram. |
| Narrative | Stephen (75%)  John (25%) | I didn't know what this was, but David described this as “document tell the programmer how to design the program.” So, for that, I guess Stephen made a greater contribution to that with the sequence diagram. |
| Quality | John (75%)  Stephen (25%) | I also didn't know what this was, but David described this as “how much value do I get from a document.” I guess since I composed the bulk of the words of the document, I suppose I contributed to the quality, for good or ill. Looking at it now, the document looks a little repetitive, but I don't think it's unnecessarily so. |
| Consistency among sections – updated requirements matrix | John (100%) | I updated the Requirements Matrix during a lab while Stephen was working on the Sequence diagram. |
| Design Pattern Implementation | Stephen (60%)  John (40%) | We hemmed and hawed over this for a while. Eventually, we came to decide on Factory. We wrote the description together and he made the class diagram. He also made the CRC Cards, though I updated them to make them plain English. |
| Detailed Managerial Review | John (100%) | Not sure why this is on the sheet. Who else would complete this part except the team lead? |

**Please give a clear description on how your team worked together and how you divided up the work. Include your team schedule.**

We met up outside of class and discussed what needed to be done every class. The design document is a bit less opaque than the requirements document, so we simply had a crack at what we felt we could do. I was a bit intimidated by the Sequence Diagram, so Stephen gave it a go, and we built the rest of it around that.

**Describe the issues your team encountered working together and how you overcame or plan to overcome them.**

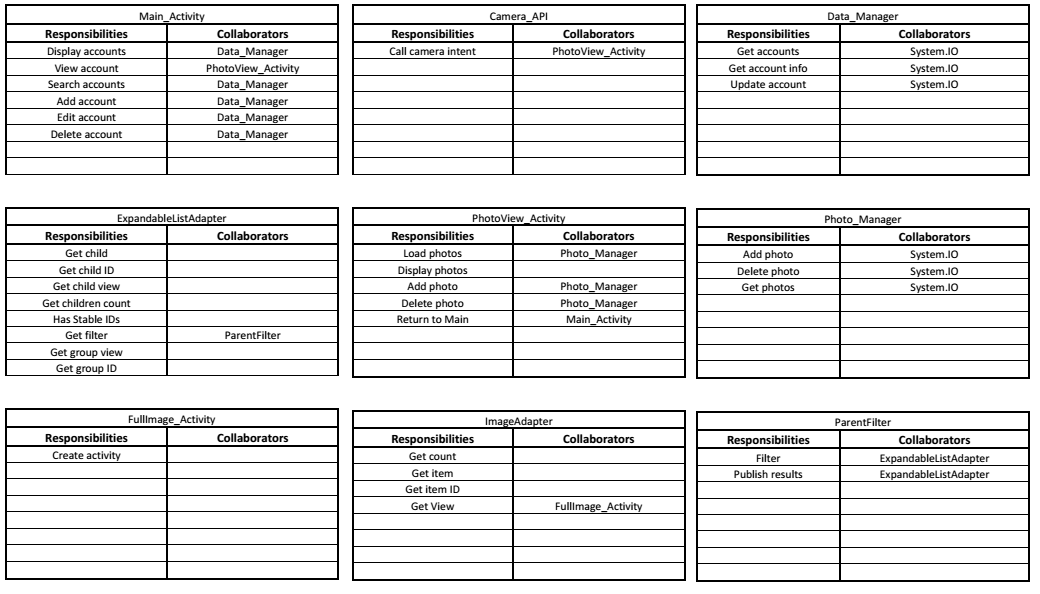
This went reasonably well, honestly. I spent the weekend before the due date doing all my other homework for the week ahead of time so I could spend all my time on this. Then, that Wednesday, we were told that the weekend and vacation wouldn’t count as late days, so we effectively had a lot longer than we thought. It kinda took the wind out of my sales, but I am glad to have had the extra time.

**Describe your team strengths and weaknesses. For your weaknesses include a plan to turn your weaknesses into strengths.**

Okay, one obvious weakness: only two people to carry the load. But it’s not shouldn’t be all that big of a deal in the long run.

One big team weakness is that I have a tendency to freeze up in fear when I am faced with one of these assignments and I don’t have someone’s help. It happened recently while trying to learn Android Studio, though I got through it by giving it some time and stopping to do dishes to let the fear subside. It might sound odd, but I really think that will be our biggest hurdle now. I know for a fact that Stephen can do this stuff. He’s plenty competent, and has done it before. I should be able to do manage, but the variable will be to summon the guts.

# Appendix A: CRC Cards



# Appendix B: Sequence Diagram