SDMD Portfolio Report

100013901, Samara Dionne

Overview

I will be creating an application to aid in playing a card game by the name of Resistance. Using this application will allow people to be privately assigned characters that they will be able to use in the game. The game re-enacts good guys vs. bad guy scenarios, where the bad guys know who is on their team but it is up to the good guys to find out who is on theirs and out the bad guys. The application will allow users to choose settings for the game, input their names and possibly view the rules. The app will randomly assign a character to each person, and they will see a description of the role of their character and possibly the lore associated with them if they wanted to read more.

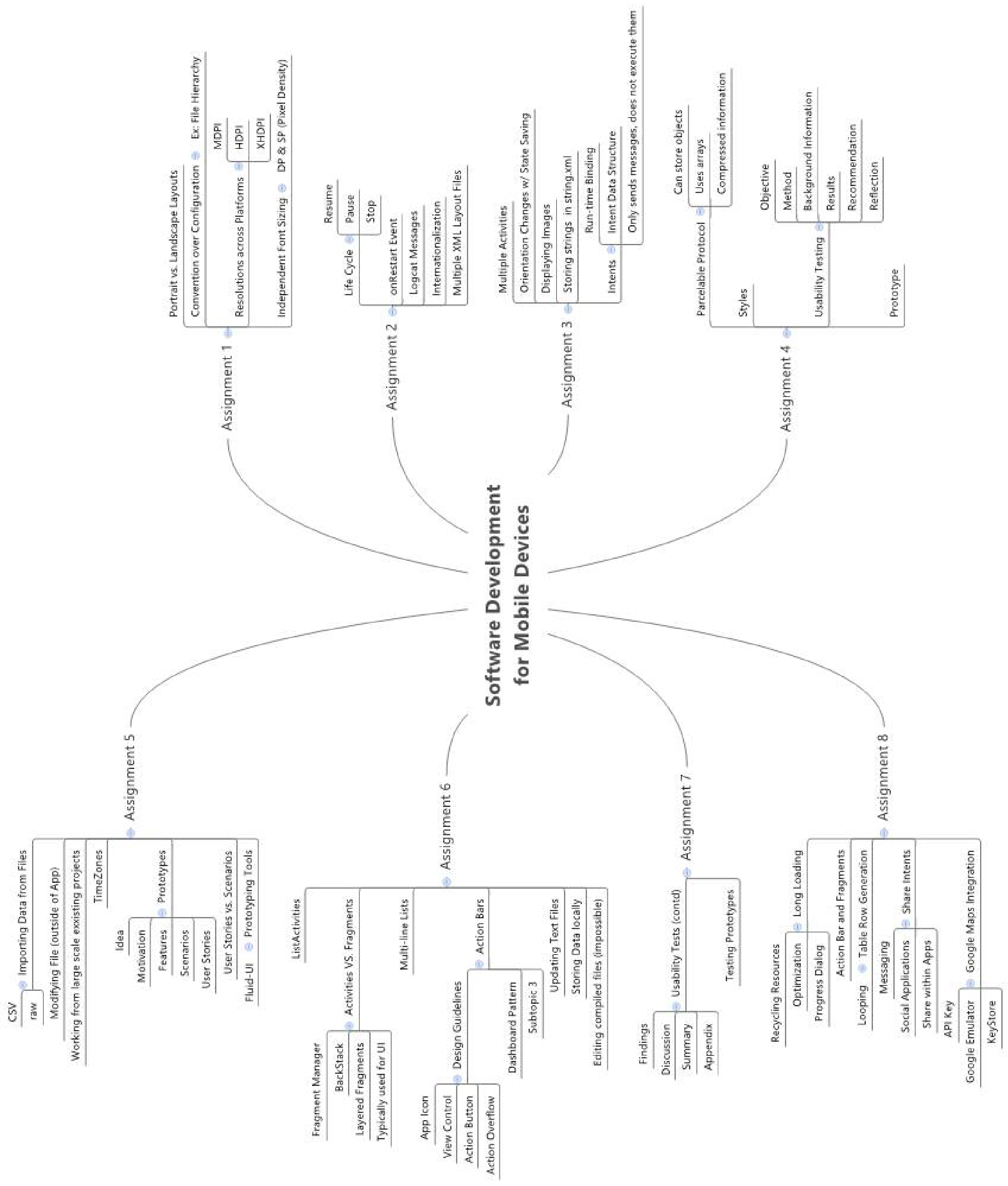
Evidence (in Portfolio Pieces)

I have completed the following assignments and the evidence is presented as part of the portfolio pieces.

Assessment	Completed
Core Assignments (for Pass)	✓
Extension Tasks (for Credit)	✓
Custom Application (for Distinction)	Planned
Research Report (for HD)	

Reflection

Concept Map



Mobile Application Development Process

An idea is the first step in creating any application. It doesn't need to be particularly fleshed out, it can be pretty broad in its initial form. The idea will become very quickly refined once we start to narrow down the problem the application it will solve, or define the motivation behind it. This is achieved by creating hypothetical scenarios of people using the application. Scenarios should be personable but realistic, they should use real names and create a clear context for what needs to be done. They should be goal oriented, but they should not dictate the steps in which one would use to achieve the outcome.

Once the initial motivation is defined, it is then time to decide how the first designs will look. Prototypes and sketched mockups should be the next step in the process. No coding is required at this stage as quickly sketching layouts will be more effective than recoding every time it is decided that something doesn't belong or should be moved. The first stage should concentrate more on functionality than form. Are all the features accessible? Does the flow make sense? Does it require too many clicks to get from point A to point B? Once decided, these sketches will be given to a developer who will start the construction of the application.

Will it be based off an existing library or API? Is it necessary that the code is proprietary? Will it be open source so that the development community can contribute more important? What platform will it be on? Will it be available in multiple languages? These are questions the programmer must consider before he will write any code. Once these questions are answered, it is then time to make a plan on how to complete the application. During this process, the developer will be required to create an aesthetically pleasing UI (possibly working with a designer), and test it out on emulators or the phones themselves if available. Constant error-checking and debugging will be going on throughout this process. Once the application is bug free, and designed effectively it is time to figure out where the application will be hosted.

Will it be cloud based? Will the developers maintain the servers themselves, or have someone else host it for them? With these choices made, the developer will be finally able to deploy their application and have it beta tested. However, before any beta testing can go on, the application will need to be approved and certified before it can go into the Google Play store. When it is in the store and ready to be used, it is then necessary to manage and package different versions. Not every user has the same model phone, and it is important to offer versions to everyone in order to properly expand the application's audience.

Once the final application is ready to go, how to promote the application is the next step to research. Will it be through banner ads on Google or other sites? Are conventions an appropriate place to present and sell the application? Should it be advertised through other applications? These are all ways to expand the audience of the application, and ways to decide which venue will be the most valuable and the most efficient way to spend your advertising budget .

Now it is time to track who is using the application, and what aspects of the application need to be improved. Through user ratings and analytics software, the *developer is able to manage updates for the applications to support the ever changing audience and platform. As with all technology, it is constantly being upgraded, and to continue to be relevant, an application must upgrade with it.*

Analysis and Problem Solving Approaches

The problem: You are given the Amazon website to analyze and improve upon the user interface. The UI is cluttered, there is an information overload and it is not clear on how to navigate the application.

The solution: Doing a usability test is a good start to fix any application (or website). The test group should be diverse, but hopefully upon background checks we could group those who have similar qualities. Differentiating these groups of people will allow us to discern if certain opinions and results correlate those qualities, or if it is just a rare occurrence. The test itself should contain scenarios that force the tester to go through different tasks to achieve a goal. Recording everything during the testing is very important. This can be done through simply watching the user and timing how long it takes them to complete each task. During the testing, the user should be asked to think aloud and walk us through their thought process. It is important to note scenarios that were not clear, or paths that were not linear enough, or options that just felt too similar.

At the end of the test, it is important to ask the user their opinion on different aspects of the application. While we may have observed them completing a task that seemed simple, it may have actually been difficult for the user. Was the interface clear and easy to navigate? Was the text legible or too large? Does the application allow for the completion of all scenarios?

Once this test is completed, it is then up to the developer to collect all the data and make decisions based on it. For example, if the users believe there are too many clicks, what will happen if you remove a few pages? Will reducing the amount of clicks lose too much information? Do those pages contain ads that produce revenue? These are all choices or sacrifices the developer will need to make in order to effectively create an application.

Comparison and Contextual Placement

Android is the first Mobile platform I have worked with. I assumed that it would be far more complicated to create an application. I didn't realize there would be a software (Android Studio) would so easily facilitate the creation of applications. Android has a predefined hierarchy of storing information, making the learning curve of jumping into developing a preexisting application all that much easier. Unlike developing for the web, there is no specific file hierarchy on where to store your JavaScript, image or html files. While Android and Web development contrast in this regard, they are actually still very similar.

Due to prior experience in web development, I find that Mobile development is incredibly similar to Web development. They are both network based and rely heavily on the user interaction(specifically non-developer audiences). They both *work with a relatively simple front-end structure (HTML and XML) and have very powerful backend capabilities. Because Android uses Java, and the web uses JavaScript (as well as other programming languages like PHP), it made the transition very easy. JavaScript is derived from Java, and therefore uses a very similar syntax. Having learnt PHP, I already understood the basic properties of loops and variables which is common element in most programming languages.*

Generalization

A very simple but powerful technique is the how Convention over Configuration. Specifically, I will mention how it can be applied to web development. Alot of new coders are taught to create websites from scratch so that they can learn every element of the code. This is great in reinforcing the use of different tags and styles, however it is not very effective once you understand the basics. In every HTML document, you are required to have head and body tags. After doing a few sites, this becomes muscle memory. If the coder used a premade template that had this already written, they would maximize their time (if only by a little).

While this is a pretty simple example, we can apply this on a larger scheme. If every web developer created a template folder with their website, that already contained JavaScript folders, images folders and generic names for the pages they typically used (index.html, about.html, contact.html, etc.) they would be streamlining their work flow. When developers use different names for folders, it can be confusing to memorize directories. For example, when referring to an image source, a developer may need to think was the URL img/flowers/lily.jpg or was it images/flowers/lily.jpg? By having a standard structure, developers make not only their lives easier, but possibly the lives of other developers who may jump in to help on the project.

Challenges in Mobile Development

Coming into this class I assumed I would have a very hard time as I had some difficulties in Java programming prior to this. The Java community is large, and there are plenty of questions on sites such as StackOverflow to push you in the right direction, but there is no one tutorial that brings every aspect of the Java language together. For example, the Java API Specifications created by Oracle are very detailed in explaining all their packages and classes, but they are incredibly dense. It can be difficult to decipher syntaxes when you are just a beginner. The Java Docs lack the easy to read quality that the Android API Guides deliver. By using laymen's terms and simple explanations, Google created a great resource to learning Android development. They have made the task of creating an application a lot less daunting.

I was surprised to actually quite like Android Studio. It is very user friendly, the interface relatively simple to navigate, and the generated code is actually effecient. I expected this program to be similar to the early stages of Adobe Dreamweaver that used convoluted templates and very dense unnecessary code to achieve simple tasks (onHover in JavaScript for example). If you know nothing about programming, Dreamweaver is fine, but for those who understand the language, it is cringe worthy to see what is being produced.

Explorations

I did not do too much exploring outside of the syllabus as I did have difficulty with some of the assignments. I focused on understanding why I used certain code, and not only copy and pasting it from the learning materials or otherwise. I plan to look more into Google Maps and how specific I can get on a user's location. My original idea was to create an app that shows the floor plan of the inside of major shopping centers. Shoppers could walk into the centre and it would either discern what centre the shopper was in and set the map to be that mall, or the shopper might have to connect manually. Either way, the app would eliminate or reduce the need for Information Kiosks, and allow shoppers to have an easier time navigating the jungles that are these shopping centers.

If I was successful in that, I would certainly look into monetizing the application. I would create a business model where the centers could upload their floor plans for a fee. This would enable users to only need a single application for all shopping centers (as opposed to now, that has applications that are proprietary to only their shopping center).