Reflection on Development Process.

As a team we didn’t have much web and android development, so learning it together was a valuable experience and the software engineering techniques we used were very useful for developing our website and app. With every sprint, however, we improved our skills, learned new technologies and methodologies, and our understanding of each other.

In sprint 0, we started to get to know each other and our project champions. We agreed to meet our product champions regularly and to keep them up to date with our project status. As a team, we worked together on the documentation elements; together, we successfully constructed our user stories and personas, which would form the basis for our later work.

In sprint 1, we worked on the server and website, focusing in particular more on its general structure.

In sprint 2, 2 of the team members were still learning how to develop for android and so most of the work was still on the website. Thus, the android app still needed to be done and we mostly left this important part of the project for the final release.

In the final release, we managed to implement all of the android app’s core functionality and to get everything working right. The code meetings helped us a lot because we together could synchronize changes to the website and the android app.

Our final demo wasn’t well executed because we weren’t prepared enough for technical difficulties. The technical difficulties reduced our confidence and wasted time, leading to a presentation that was rushed and that took extra time. When we were asked to show the burndown chart, we hadn’t realized that we were supposed to maintain the chart actively, rather than preparing it at the end of the sprint - this was a big mistake. Nevertheless, this was a valuable experience for us because we learned an important lesson: always scope out what resources are available for a presentation in advance and prepare contingency plans. No matter what.

We used Django, heroku, eclipse for android and eclipse IDE for web development and postgresSQL. We used Django because it is very intuitive and it is in Python which is easy to understand. However, there is always a learning curve understanding how Django works but it is quite effective because we were able to finish our user stories effectively. We used heroku because it is free to use and the way to use it is essentially pushing it into git for heroku. Therefore it is really easy to use. And we used Eclipse IDE for android and for web development because we all had experience coding in an Eclipse IDE. In addition for android, there were a lot of resources for us to learn on how to use the Eclipse for android.

The way we organized our user stories was really helpful in setting our priorities so that we could get the important things done first. Our greatest regret was not working on the ability to send images from the android app in the first sprint. This is because the android app is such a vital component of the project.

Another Problem that we have encountered was in working with the database, since we have to support 2 separate platforms (android/website) . Connecting the database using postgresQL was a pain, but coding meetings together improved our understanding and helped us overcome this hurdle.

Another problem we encountered was in merging and dealing with conflicts. This took us a lot of time in the beginning because we aren't used to working together and in such large teams, but with time and an improved understanding of one another, we were able to solve such issues faster and more efficiently. Thus, this proved to be a very useful learning experience for us.

Holding frequent meetings with our product champions proved to be a very good tactic, because they gave us a clear idea on what we should do and when, and also gave us feedback on what we had already done.

We always used a scrum board to keep track of our progress because we realized how helpful it was in keeping track of the status of different tasks. Dividing tasks into things done, things being done and things to do, and making it clear to whom each task was assigned made evaluating the state of the project at any time quicker and easier.

However, we didn't use TDD effectively. The reason is that we were learning and developing at the same time and we felt a need to learn how to develop it before writing the test cases. Since we were new to these frameworks, we first made the working code then we wrote most of the test cases down. With the test cases we had encountered some problems and we learned that before learning a new language/framework we should always take half of our time learning and making test cases. And we could work more efficiently.

We also used peer programming and it was really useful and fast because we were learning and developing at the same time and hence peer programming was quite effective however there were a lot of times where we couldn't meet and therefore we had to use long distance communications to code. However, it wasn't effective in the beginning but we learned and we learned our commit messages and therefore we were able to work together really well and be very effective. However, peer programming is still very effective and we should have try to do it more instead of communicating via online.

All in all this course and the project really helped us to improve the skill of working together with strangers and it was an exciting journey for all of us.