Overview

The project took place over a three-month period and consisted of five key deliverables, the objectives of these deliverables got more significant and required more input the further through the project the team got. The group was made up of seven members, four of which had previous experience in application development and three who either had little or no background knowledge in the area. Because of the gap in required knowledge the project group was split into the development team and documentation team with the group leader (Sumedh) being part of the development team. Due to Covid-19 and lockdown in place during a large chunk of the project’s life-cycle the team had little to no opportunities to meet in person and so all meetings or communication was done online via, Zoom, Google Meet, WhatsApp and GitHub.

Team Structure

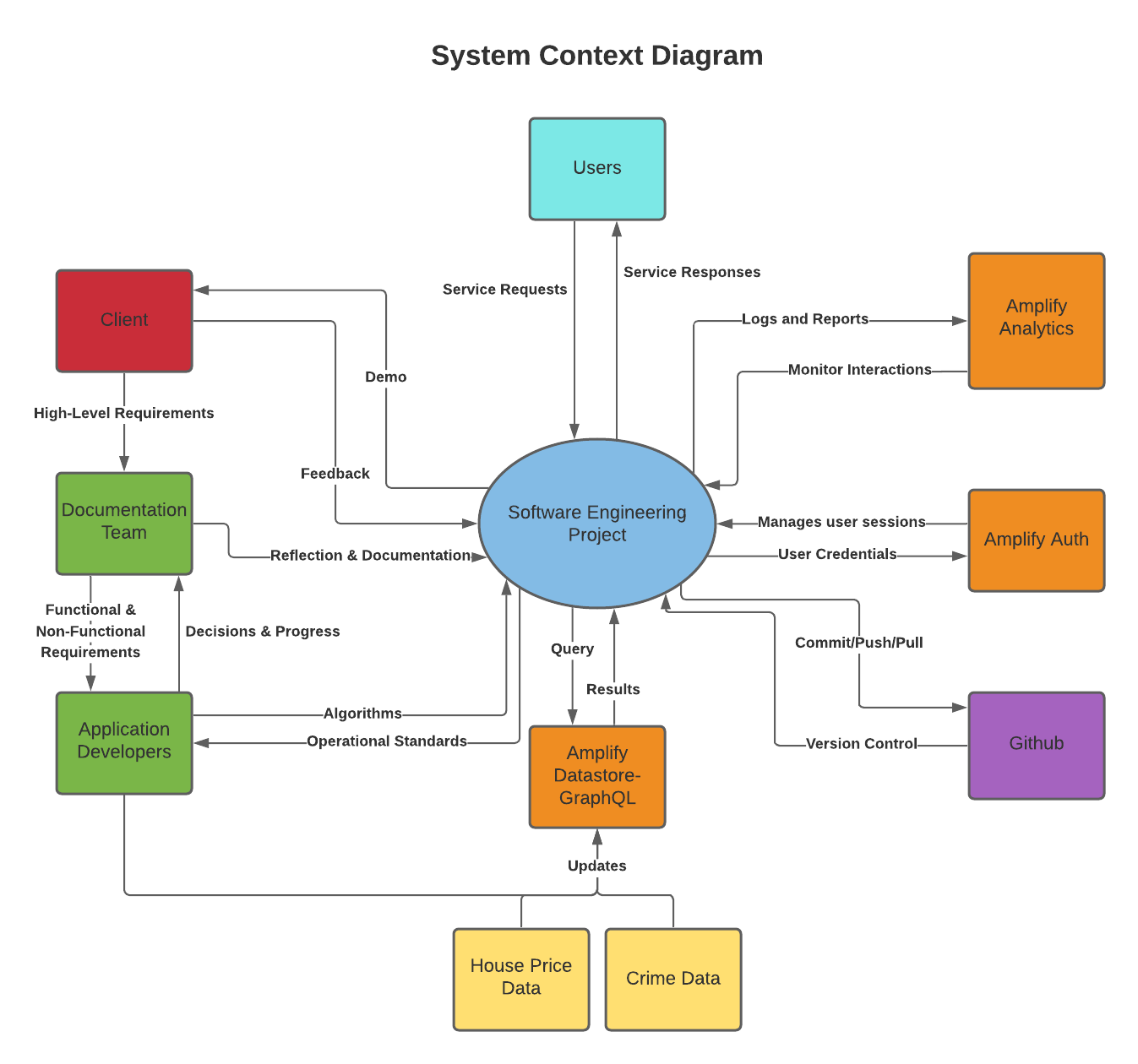
Team allocation was vital to the project as there was a clear divide in those who had experience in application development and those who hadn’t. Team allocation allowed three of the seven team members to be allocated to the documentation team and play a different role in the project, these three team members would still be kept in the development loop via the team leader but would mainly be tasked with research and writing. To keep the development team and documentation team in sync, frequent team meetings were held and all team members were able to communicate their problems, ideas, progression and general thoughts on the project. During the development of the first deliverable the varying levels of personal understanding around application development and presenting became clearer, this resulted in some changes in the team structure. The most significant change was made before the presentation of the first deliverable, one team member felt more confident presenting the software developed and therefore took on the role of team leader, replacing another member of the development team.

Software Development Process

The software development process mainly revolved around an agile approach. One of the main reasons for us to use agile over waterfall was because of the gradual release of tasks. Upon undertaking the first task we had no idea what the future tasks were as they had yet to be released. Our only understanding was that the tasks would somewhat connect to the previous task, this allowed us to recognise our application would better suit an iterative approach. Due to the complexity of some tasks we could’ve switched our method and tried a waterfall approach however, we didn’t feel this was completely necessary. For these more complex tasks we used a mix of both waterfall and agile. Because of the size of the tasks we felt it was better to plan how we were going to complete it, and highlight the different stages. Once we had decided on a plan members were able to work on their designated tasks. Group members remained flexible throughout the task as there were often times in which things needed to change, for example, the customer giving us more details or something wasn’t working correctly so we had to change our approach. Having the fixed process and outcome, complemented with a flexible approach played a crucial role of the completion of our project. The developers worked in sprints, this meant they’d have a week, or longer, to work on a specific part of the application. This was effective as it meant they could focus on their specific task without any distractions.

System Context Diagram

The System Context Diagram shown below describes the context of the project in high-level terms so it can be understood by all stakeholders of the project. The context bubble in the middle, named *Software Engineering Project,* masks a group of interconnected processes and activities within development. The diagram is a useful tool for describing the project’s scope and identifying all external factors and events that would need to be considered, such as the technology available from Amazon Amplify and the Data required for such a project, as well as the needs of the users and clients, as shown in the diagram. This is most useful for business partners i.e. the client, as it can indicate areas which require budgeting (Amazon Cloud Services) and can help decipher the domain that is being investigated at first glance. While the diagram serves little purpose to the development team as it lacks detail, areas of strength and weakness, which will be discussed in this report, can be subtly deduced from this diagram.



Bad Feature 1: Divide in communication between two teams

The biggest issue within our software engineering process was the lack of synergy between our two sub-groups; Development and Documentation. One of the biggest mistakes we made was to have one main group chat for the whole of the group and smaller chats, with only the members of the sub groups in them. This meant that the majority of communication happened with the members of the sub group. We felt this was the wrong approach to take, by having all of the communications in one place would’ve allowed everyone to ascertain what each group was doing, and how they were doing it. We also found that the majority of the communication between both groups was through the team leader and not always through other members, this sometimes caused the documentation group to feel slightly out of the loop when it came to the progress towards different deliverables.

Possible Solution

An alternative approach could’ve been to use an app, such as, Discord, this would’ve allowed us to create different channels for specific conversations, but would’ve still allowed us to see what's happening with every part of the project. In hindsight WhatsApp probably wasn’t the best platform for communications on this project. Another approach we could’ve taken to ensure that everybody was on the same page would’ve been to have one member of the team working in both of the groups. This approach would’ve made it easier to share information between every member. Another approach could be to add a team leader to the documentation team, this would allow for decisions to be made quicker and facilitate better communication between the documentation group as a whole.

Conflict that was resolved by a good feature

Another issue we encountered involved one of our group members failing to interact and respond to any messages. Although this did not make a huge impact it undoubtedly affected all but one of our tasks. From the final solutions to each of the tasks you couldn’t see we’d lost 25% of our development team, this is a testament to them as they had to put in the extra work to make up for our loss.

Bad Feature 2: Meeting / communication difficulties

Despite the fact we were able to complete all of the tasks given to us, we did encounter some issues along the way. Some of the problems were completely out of our control, for example, the Covid-19 restrictions.

Not being able to meet with other group members in-person was one element of these restrictions. Although not perfect, we made use of Google Meet, a video communication service that allowed us to communicate as a group. This was not a problem for our whole group meetings, in which we’d discuss topics at a higher level. The problems arose when we’d be in smaller groups and we wanted to share things. Features such as screen sharing and shared documents that update in real-time were utilised, however, these did not match being able to meet-up and work collectively on a single machine. Early on in the project the documentation team occasionally ended up working on the same activity, resulting in duplicates, after this we had specific tasks to complete and would highlight anything we were working on in the main task list.

Good Feature 1: Meetings

When researching and developing the application there were various methods and features to the project process that allowed all team members to successfully contribute towards the product regardless of their previous experience. These methods also facilitated a productive team environment and allowed for good communication, clear goal setting, better understanding of tasks and more. Although the team used various methods, the key process method used by the group surrounded quality co-operation, this method largely took the form of frequent and efficient team meetings often before or after timetabled lectures and seminars. This method was helpful as it allowed the team leader to communicate the current group goals to both the documentation and development team.

Good Feature 2: Efficiency of Team meetings

The purpose of the whole group team meetings were usually to update everyone on the tasks that had been completed by each member and to assign new ones where possible. All team members are prompted to prepare topics they would like to discuss prior to the meeting. Meetings were fairly efficient as after the release of a new deliverable, the documentation team would create a requirements document and compile a list of functional and non-functional requirements, to which the development team breaks up into lower-level requirements for the development process. Usually, the meetings would end after all conflicts or questions have been resolved and everyone knows what they need to do in time for the next meeting. Where the team leader is required to present the product to the customer, the team has a practice presentation in order to prepare for all questions the customer might ask.

Change to peer assessment

Our peer assessment described a system, where the contribution of each team member was supposed to be assessed after every deliverable. This was meant to be done by keeping a log, which outlined tasks to be completed for each deliverable for each team member and comparing what has been done to what should have been done, through the Github commits, which allows us to see the contributor. The log was regularly updated, however, the nature of the deliverables, which most of the time each required different skill sets, and the structure of the team as previously described, made our intentions unfeasible. Certain deliverables required very little documentation, in which case the documentation team was left to research the technology used in order to stay in the loop of the development process and be able to document the use of such technology in later deliverables, however did not actually contribute to development. In other deliverables, the documentation team only required feedback and guidance from the development team. Within the development team, there were smaller deliverables, which were developed by one or two team members, while others required everyone’s contribution. As mentioned, the current Covid situation limited the group’s opportunities to collaborate as often as possible, but having active team members and regular meetings allowed us to deliver all units before the deadlines. The group has opted to individually assess all other team members’ contribution using Github and the log after the completion of the final project instead and each member is set to receive the mean score assigned by the others.

Change of original planned role and why

Our initial plan was to have the most experienced member of the team (MirageLee) mentor a couple of other members, the idea behind this was to allow everyone to learn something new from the module. This idea never came to fruition as it was, again, difficult to do remotely. If two members were able to meet in-person a paired-programming approach could’ve taken place. This ultimately didn't have an effect on the outcome of the project, however, it would’ve been beneficial to the learning of the developers. In this case, developers each took on a task which utilised technology they were most familiar with. However, this still resulted in the group mentor having to do more work on bigger deliverables.

Task Breakdown

**Task 1**

This task was simple yet essential for all upcoming tasks. We were required to create Github accounts and a joint repository, where the project files will be stored.

**Task 2**

The second task was fairly simple. Our implementation of the task was a map displaying the users’ current location along with the longitude and latitude coordinates. To develop this we used Android Studio.

**Task 3**

The second task required us to improve our initial application by adding client-server communication. There were many different cloud-based services suitable for this project. After some research we decided we’d use Amazon Web Services and make particular use of AWS Amplify. This gave us access to useful tools such as; Amplify Analytics, Amplify Auth and Amplify Datastore with GraphQL. Our implementation stores the clients location in a database in the backend server. For this task we have also implemented accounts, these are created by the user and authenticated using Amplify Auth. The user's location is sent to the database every five minutes.

**Task 4**

The main aim in task four was to display house prices in a specific location. We were given a large data set from the Land Registry, containing all of the data we needed. One of the main challenges was how we were going to process and display the data in an efficient way. We made sure to only use the data we needed to display, this was important as using the whole data set would have been extremely inefficient. Within our implementation we have also included graphs that display the change in house prices in the specific postcode area as well as a heat map feature to visualise the amount of houses that have been sold.

**Task 5**

For the final task we were told by the customer to implement another feature, the customer was fairly vague in what he wanted, this allowed us to decide on a feature that we felt would improve on our current implementation. We decided on displaying crime figures, we thought as the users were viewing the house prices in a specific area they would also like to know some information about crime. Our application displays the locations of the recorded crime in the area, as well as a small description.

Our Implementations

Upon launching the application, the interface will let the user know that some data is being downloaded using a loading bar.

Once this is done, the user will be prompted to either login or register. If the user needs to register, the system requires a valid email and a password of sufficient complexity. The app awaits email verification before registering the user’s credentials.

What was being downloaded at the start is the positional data of all the houses in the database, onto the device’s local storage.

While this uses up approximately 60mb and requires around 15 seconds to download, it massively increases the search speed later on in the application. Our initial approach was to load the data, for all houses within a 5km radius of the device’s current location or from a location the user has searched for, from the server during run-time. However, the implementation chosen was found to decrease the search time from 5 seconds to 0.5 seconds and supported the creation of a much more interactive application at the cost of a slight increase in space.

Once the user logs in, they will be able to see all the houses in the database in the area covered by the screen on google maps. The number of houses displayed adjusts depending on how much the user zooms in or out. The tab on the top right-hand corner shows a “Heat Map” feature. Once selected, it displays a layer on top of the current map, showing the density of houses in all areas in the UK as requested by the client.

By clicking on a certain house on the normal map, the app will direct the user to a page showing a graph of the transaction history for that house from the years 1995 to 2020. From here the user can choose whether they would also like to view additional data about the property throughout those years from the “House History” tab or if they would like to see details of crime incidents in the area by clicking the “Crime” Tab. Displaying the crime rates was the additional feature our team chose to include for the last task, as this is a feature we thought would be of most interest to someone looking at properties in a certain area. “Crime” directs the user to another map, showing the current property’s location and the locations of all crimes in the area that happened over the years. The user would then also be able to see each crime’s details such as the nature of that crime and who reported it, by clicking on the tab on the top right-hand corner of the screen.

To view data for houses away from the user’s current location, there is a search bar on the top right-hand corner of the main screen, where they can search google maps. Note that this search can direct the user to areas where there is no property data.

The tab on the top left-hand corner can be selected to show the user their current location in terms of coordinates, street number and postcode. This is where the user can also logout.

Conclusion

Overall we feel the project was a success. We completed everything that the client asked for in all of the tasks and often went further than necessary to build a quality application. This report has highlighted the elements of our group structure and approach that was effective as well as what we would do differently. Most of our group members had some experience of the software development cycle, however, the remote approach was something completely new for everybody. Ultimately this didn't affect the outcome of the project although it did take some time to get used to. As previously mentioned the process of having two groups worked well, although, there were times in which there was a lack of communication between both of the groups. This communication issue may have been less apparent if we were able to regularly meet face-to -face or use a different platform for our communications. If we were to do the project again we would definitely have a couple of members in both of the groups, this would enable information to be shared much more easily.