Process and Project Plan - Task 3 & 4

Group 5

1 Process method

In our group we have a mix of plan-driven and agile software development methods. We can not rely on only plan-driven and only agile software development because we are dependant on change. As of now we do not know what next obligatory assignment is going to contain, that is why we can not have a fully plan driven development method. But we assume that our plans and methods needs to change later on, hence a more iterative approach. In plan driven development it is harder to change the already developed plans in comparison to an agile method were all stages of development go through several iterations. Considering the information available to us, we develop initial plans as far ahead as possible and leave room for change later on in the development process.

We are inspired by the SCRUM methodology. Our meetings are regularly twice a week - once in the group meetings and once later in the week. We have planned to meet more than just twice if it will be needed in the working process. Each time we meet, we first discuss what has been done, what is in process and what needs to be done. Tasks are allocated equally. We always consider the thoughts of our teammates, what they would want to work on and what they are good with.

For the later tasks we will assign different tasks and roles for each team member based on interests, skills and wishes. Communication between team members working on the same aspects of the task will be important. We will review and follow up on the work that has been done by the different members during the team meetings where everyone are present to make sure progress is being made as planned. When we are not meeting at the school, we have communication through applications like slack and discord. Here we are keeping each other updated with tasks and resolve problems. Reallocation of human resources will be decided during team meetings if required.

2 Organization

We will hold 1-2 meetings each week in addition to the obligatory lab that are held every tuesday. At the meetings we initially start with reviewing the work that has been done, and continue refining this until every member is satisfied

with the current state of the different tasks. Before assigning more work to the team members, everyone is requested to read through the tasks that have been given to us, after this we discuss what should be included in each of these before dividing the tasks between us. The division of work is a shared effort between the members, where everyone can present what they wish to work with. As mentioned under house rules (Team Plan), members who encounter major problems or feel that they have stagnated should request help by the other member as soon as possible. Contacting other members can happen through our group chat in either Discord or Slack where we discuss work outside of the group meetings.

Every member who feel that they are not well traversed with the use of git should tread carefully when working directly with the remote repository. When creating new files or editing files that already exist in the git-repository, one should notify the rest of the team what they are working on. This is to prevent unnecessary conflicts when pulling and pushing to and from the repository. A thumb-rule is to pull any changes made in the remote repository to your local repository, before pushing your own work to the remote repository. This is to take care of merge conflicts that will be present when several people are working on the same set of files. We will use a mergetool that is included in IntelliJ to assist the team in resolving conflicts, this tool creates a view where your can see your local changes to a given file and compare this to the remote changes. Refer to the Team repository structure section in the Team Plan for an overview of the general branch structure that will be used.

Our initial plan is to divide the programming part of the project into 3 sections, AI, GUI and the game itself. Two people will be responsible each of the parts. By doing this we can assure that one person wont be alone in programming. This will improve the quality of the code we produce, decrease chance of conflicts and decrease the chance of bugs and mistakes. This will also distribute the workload evenly.

We will try to follow the class diagram made in task 2 for the basic structure of the project. We do expect some of this to change as we start the actual programming. Because of this we have tried to make the structure as modular as possible. This will make it possible to easily add new features to the basic structure without much difficulties. We are however 7 people so we still have one person who is not part of any programming team. This person can take care of documentation, and other administrative tasks. Of course the programmers will be involved in this process as well to a certain degree. We will hold regular meeting where we can communicate and see how the others are doing. If we later on discover that one of the tasks are more time consuming than expected we can redistribute members. The same goes for less time consuming tasks where we can finish faster than expected.