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 needed.
With this agile concept in thought we can use several iterations to get the finished product done. This means that we will visit typical software development activities such as; requirement specification, design, implementation, testing and so on several times. Naturally because we don't see each other face to face every day we can't hold daily sprint meetings in the normal way - this means that communication through Discord and Slack becomes more important to inform everyone on how we are doing.
At our regular team meetings we can do the "sprint planning" where we discuss what needs to be completed, what can be done, the scope of the upcoming sprint and who does what - every team members opinion is of course valued in the allocation of the work. The time between a given meeting and the next one can then function as a "sprint" where every member work on the work assigned to them, though the time limits for the sprints are subject to change.
In addition to this it is important that at every meeting the different project management activities are considered, some subjects for discussion will be; have we identified new risks?, how are we doing in relation to the risks that we identified earlier, are we falling behind on a feature?, are we on time in relation to our final deadline, who are doing what and how is it going?.

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. As mentioned above; 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 merge-tool 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. At least two people will be responsible for 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.
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. 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.