Team Plan

Group 6

April 6, 2018

Skills and specializations

These roles are not final, but serve as a reminder of what people are good at.

Name	Role
Robin Grundvåg	GUI
Benjamin Dyhre Bjønnes	Code
Eirik Jørgensen	Code
Vegard Itland	Documentation, GUI
Loc Tri Le	GFX
Robin el Salim	SFX
Sverre Magnus Engø	Test suites
Stian Soltvedt	AI, version control

Version control

The repo structure is organized around the develop branch, which is where most of the changes to the code base are made. Usually when the change is small or essential to many areas of the project, commits are made directly to develop, but bigger changes which are less central to the core of the project (such as GUI or AI, which draw on the chess logic API but is not required by it) have been known to receive their own branches and later merged or rebased back in. The develop branch is merged into the master branch on release, and every commit on the master branch should be a release version of the game (v0.1, v0.2, v1.0, etc.)

Commit message titles should be brief, of the format "[Topic] short description of change". It's possible but not required to leave more detailed messages in the commit message body. Ideally, every commit should build and all the tests should be passing. Particularly the latter is not always achievable nor desirable, as other people may depend on work in progress, and more tests are good even if they don't pass, but it's an ideal to strive towards.

A more detailed explanation of our contribution guidelines can be found in `CONTRIBUTING.md` in the root project folder.

Communication

We will be using Discord for most forms of communication due to the handy ability to create multiple channels for chat. Facebook Messenger may be used when prompt response is needed.

We're using Gitlab issues for issue tracking, and have started using milestones which we connect relevant issues to. Ideally, we will create relatively small and achievable issues in order to more easily track our progress, but we will avoid creating nonsensically small or trivial issues to avoid issue board clutter. Issues are tagged with relevant labels, and commits which are made in order to advance particular issues should be marked with the issue number in the topic part of the commit title.

Risks

Feature creep

We could fall into the trap of overpromising on the number of features added to the application, delaying the project.

We mitigate this by focusing on the basic features required by the assignment in the initial sprints. If members wish to work on additional features, they will have to be done in a later sprint after the minimum required features are already implemented.

Bad time estimations

We may end up missing deadlines due to underestimating the amount of work required to implement features. We consider this the most likely, and most dangerous risk to our project.

We mitigate this by meeting bi-weekly so we can review progress on tasks. That way we can get quick feedback on whether an estimate is off or not, and take corrective measures.

Integration issues

As this is our first time cooperating on a larger software project, we may have issues combining all the separate parts individual programmers develop.

We mitigate this by discussing the interfaces we will be developing in advance, so that others can work with those while the implementations are being developed.

Missing key competence

For some technologies (git, latex) we only have one person who has a lot of experience using it. To mitigate this issue, we have created a #help channel in Discord where those people are available to help out with the respective technologies.