Plan of approach

DHI1V.SQ_3

Background

Our group is going to work on developing the software for the "The Rather Rudimentary Operated Lunar Vehicle" shortened to Frog. We are going to develop two separate applications one called “The Pilot” and the other “Ground Control”.

The Pilot application will have the functionality of controlling the rover manually through buttons, it will display the “Mission log” which is a history of all the executed commands by the rover so far and it will also be able to send this list to the Ground Control application. In the pilot there will be an option to run a preprogrammed mission by the Ground control. Passive functionalities are sending the collision points and current location to the Ground Control and the algorithm for path finding.

The Ground Control application will be able to give coordinates to the pilot application for which the pilot using the path fining algorithm will control the frog to set coordinates, it will also display a map with the current location of the rover and a map with the current coordinates detected. This application will have the ability to create the preprogrammed missions and send them to the rover. It will also export a mission or collision points.

Both applications and the rover itself will communicate with the SaSa communication library and both applications will have a Postgre database to store all the necessary data.

Results

We are going to deliver a Functional design with wireframes and use case diagrams, test plans and tests for the application. A technical design with a complete breakdown of the application. In our repository we are going to push all our code using branches. Product backlog and sprint backlogs can be found in the Gitlab repository or in the Sprint Report files.

Organization

For our group organization we will be using several tool provided to us from the school. We will use Gitlab to share our code with each other. Everyone will have their own branches to keep thing organized and avoid push conflicts as much as possible. Sebastian will be responsible for maintaining our repository since he has the most amount of knowledge of all of us when it comes to git. Yani will be responsible for the SCRUM boards and creating the user stories. We will use Gitlab for Scrum as well. For communication between the team members, we will use Discord and WhatsApp. Our team meetings will mainly be conducted through Discord. We will use the One drive folder to share any important documentation. Viviana will be responsible for maintaining the folder organized. We will have a stand-up meeting at least once a week to keep up with each other's work and help each other if need be. The meeting will be held when all team members are available.

As far as tools go, we used Java swing for the UI, PgAdmin and Postgre for our database and JDBC for communication between the code and the DB. We used SaSa communication library for interaction between the two applications and the rover.

The layout of the sprints is split this way: For the first sprint we will familiarize ourselves with the environment provided to us by the client and establish some organizational structure for our team. For the Second Sprint we split the work and mainly focus our efforts on creating the common frame within which everyone will work on. This includes common UI elements for the front end, DB structure and classes and a common folder structure to keep our code tidy. We decided to use MVC model for that, and if we have time, we might start working on some of the tasks handed out already. For the third sprint we finished all the codding tasks fix any bugs left and refactor our code and the UI. We will also create the test plan and at the very end test our application.

Borders and Risks

When it comes to the project itself, we will mainly focus on the requirements marked with a must and should. For this project we have created a list of simple rules and everyone should follow, however we all agree that the core principle everyone will follow is that if you accept responsibility for something you will have to keep your word and do it. Of course, since this document is written before we began work on this project we expect thing to go wrong so the way we plan to handle this is by consulting each other when a task proves to be too difficult or complex.

Definition of done

This section will provide clarity for our team and the product owner when it comes to when the project can be considered completed. With this checklist we clearly define our scope of our tasks so there will not be any misunderstanding between team members or between the owner and the team.

User story is completed when:

1. User story is integrated with the Main code base, and everything is working
2. Exceptions are handled properly
3. The integrated code is tested on different machine
4. Features are being peer-reviewed in a team meeting
5. Features are tested against the acceptance criteria
6. All functionalities meet the user requirements
7. The documentation is updated
8. Feature OK-ed by the product owner
9. Refactoring, if necessary, completed
10. Any configuration or build changes documented
11. UI is OK-ed by UI designer (Sebastien)
12. TODO's are completed
13. DoD of each user story are met (Description)

Team Agreement

1. We should be joining class at least once a week
2. We will use WhatsApp and Discord to keep in touch and communicate with each other
3. When a group member takes a task and promises to deliver that task, he should do it

    3.1. If a group member sets a deadline for that task, he should do it on time
    3.2. If a group member cannot do the task on time, he should communicate it with the rest of the team
    3.3. If a group member did not complete his task and he didn't mention it to the rest of the group, he will get a Strike

4. If a group member gets 3 strikes, he will be reported to the teacher.
5. The response time should not be longer than 24 hours
6. We will be sharing our documentation in One drive
7. We will be using Gitlab with a separate branch for our code
8. The expected response time for a message should not exceed one working day

8.1. If a team member fails to respond 3 times within the above-mentioned time frame, he will get a strike

9. We will be conducting 1 team meeting at least once a week

    9.1. During team meeting we will be giving each other feedback on the work we have done
    9.2. All team members are required to join the team meeting
    9.3. If a team member agrees to a meeting and doesn't show up he gets an immediate strike

10. If a conflict within the group occurs, one group member will join in and mediate between the 2 sides.
11. Every team member will put at minimum 8 working hours a week outside of school hours.