Task ID	Description	Complexity (S/M/L)	Justification
SPRINT 1			
1	Code reading for comprehension	L	This is the task everyone spent the most time on. Since we wanted to get a detailed understanding of the system we felt this was the only way of achieving this.
2	Individual Class diagrams	L	Creation of two class diagrams according to “Individual Milestone 1” to better understand the system.
3	Debugging the system	M	We thought it would contribute a lot to our comprehension if we could run the program using a debugger. We also spent a lot of time trying to make this happen.
4	Final Class diagram	M	In the beginning, some of us created a class diagram as a part of the individual milestone, and we decided to create one common class diagram to cover the important functionality and combine all thoughts from the first diagrams. This required us to get on the same page comparing all the existing diagrams and agree on the scope of the new diagram. The new class diagram is a big help for the system comprehension.
5	Setup libSMCE and smce-gd on our machines	L	Each member of the group encountered multiple different errors and obstacles of varying complexity before managing to compile/launch the projects.
6	Class documentation	M	We have documented most of the classes of the libsmce library ourselves. We selected the most important ones based on our understanding of the flow of the system. Based on this information we also continued to further specify the classes using the common class diagram.
7	Brainstorming/ Discussions/ Sharing gained knowledge	M	Most of our understanding of the system came from multiple working sessions where we read and discussed code together.
8	Creating the presentation slides	S	Presentation slides for the first milestone to better present the used techniques and first requirements.
9	Testing out the	S	We thought it would be nice to see the

	program/trying to compile different robots		system in action and that it might help our comprehension. Also making sure we can compile robots confirmed that the program was actually working.
10	Doxygen documentation	S	In the beginning we generated the doxygen documentation while also using GraphViz. It helped with understanding the scale of the system, but further code inspection was needed.
11	Flow diagram	S	We created a flow diagram based on how libSMCE is used in the example program stduart.
SPRINT 2			
12	Explain test cases (Polyfills)	S	Explain tests in Polyfills-file in detail and add to the drive document.
13	Explain test cases (BoardView)	S	Explain tests in BoardView-file in detail and add to the drive document.
14	Explain test cases (LibManagement)	S	Explain tests in LibManagement-file in detail and add to the drive document.
15	Explain test cases (BoardDevice)	S	Explain tests in BoardDevice-file in detail and add to the drive document.
16	Explain test cases (Toolchain and Board)	S	Explain tests in Toolchain and Board-files in detail and add to the drive document.
17	Design tests (Board)	M	Create tests for the Board-file in libSMCE. Increased test coverage from 88% to 95% by adding three tests that check the conditions for seven methods in Board when these are allowed to be called on the board.
18	Design tests (Ardrivo)	M	Create tests for the Ardrivo-part in libSMCE. Did look into the Ardrivo folder, and came up with possible test case scenarios. However, the tests were not implemented yet for this sprint.
19	Design tests (BoardView)	M	Create tests for the BoardView-file in libSMCE.
20	Design tests (Toolchain)	M	Create tests for the Toolchain-file in libSMCE. Focusing on covering all sketch build/configuration Toolchain-errors.

21	Investigate CodeCov usage for our fork	XS	Figure out how we can see the CodeCov report for our own fork and branches.
22	Activate GitHub Actions + enable Sonarcloud	XS	Configure the GitHub Actions and enable Sonarcloud for our repository.
23	Code Review	S	Review tests - only done for a few pull requests this time, so only a small task for now. Will become a big and important task in the next iterations.
24	Implement test cases for Arduino in Ardrivo	S	Implemented test cases for 15 functions in Ardrivo/Arduino. This will not be seen in the test coverage due to how testing for Ardrivo is done.
25	Meetings	M	Working sessions, planning sessions, and meetings with stakeholder and TA
26	Test doc review.	S	Making sure that our test documentation was correct.

Name	**Task ID**	**Contribution in %**
Max	1	20%
	3	100%
	5	20%
	6	20%
	7	20%
	9	60%
Sprint 2		
	14	100%
	19	100%(not done)
	25	20%
	26	25%
Tim	Sprint 1	
	1	20%

	2	33%
	5	20%
	6	20%
	7	15%
	8	100%
	10	25%
	Sprint 2	
	16	100%
	17	100%
	22	100%
	23	50%
	25	20%
	26	25%
Pontus	**SPRINT 1**	
	1	20%
	5	20%
	6	20%
	7	25%
	9	20%
	10	25%
	SPRINT 2	
	15	100%
	20	100%
	21	100%
	25	20%
	26	20%
Marek	Sprint 1	
	1	20%
	2	33%

	4	100%
	5	20%
	6	20%
	7	20%
	10	25%
	Sprint 2	
	12	100%
	18	50% (not finished)
	25	20%
Lukas	Sprint 1	
	1	20%
	2	33%
	5	20%
	6	20%
	7	20%
	9	20%
	10	25%
	11	100%
	Sprint 2	
	13	100%
	18	50% (not finished)
	23	50%
	24	100%
	25	20%
	26	25%