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.
SPRINT 3			
27	Finish testing tasks from Sprint 2	S	Finish what was left over from the previous sprint, mainly resolving pull requests.
28	General research, for example on compilers and CMake (task 10)	M	As these topics were new to us, we had to research on several topics to be able to actually work on task 10.
29	Testing out ways to find compilers	M	Implemented ways to find compilers on a computer. This was done before the start of task 10 for practice.
30	C++ course	M	Ryan recommended looking into a C++ course to improve the general C++ knowledge and also how to create a modern API.
31	Task 10 Implementation (Backend)	L	Main implementation work of task 10. It is not completed yet, we plan to continue working on it in the next sprint.
32	Task 25 implementation	L	Implemented a navbar tab navigating to a “Help” window that fetches the entire smce-gd GitHub wiki and displays it in the frontend.
33	Task 13	L	Made it so that smce-gd automatically

			detects CMake and installs it if no installation is found.
SPRINT 4			
34	Task 10 (Backend)	L	Made it possible for CMake to use the selected compiler. During the last sprint, we implemented functionality to find compilers on a Computer. In this sprint, I have made it possible to compile a sketch with a selected compiler. This works, but ABI differences should be looked into during the next sprint.
35	CMake/Boost/Ninja tutorials	M	Spend a lot of time with tutorials for things relating to task 10.
36	Task 25 Improvements	M	Images, markdown formatting and new way of fetching wiki. Ran into a lot of trouble with displaying images and certain godot string formatting stuff.
37	Try to finish task 13	S	Since it was almost finished the last sprint it would be nice to finish it.
38	Dev-tools proposal	S	Prepare the document for the dev-tools proposal submission.
39	Task 10 (Frontend)	M - L	Start frontend implementation of task 10. The compiler can now be selected in the profile view and can also be saved. It took time to get into Godot and the collaboration of Godot and C++. In addition, much time was spent on calling the actual backend, this is still not completely working yet.
SPRINT 5			
40	Devtool planning	S-(M)	2-3h meeting plus further thinking about things like the algorithm for coloring directories and files.
41	Devtool implementation	L	Created a backend in Python with FastAPI (download git repo, executing commands on it, and generating file tree). Created frontend with React.
42	Task 10 (Backend)	L	A lot of time has been spent on adding more functionality, and fixing issues. It is now possible to dynamically look for different compiler versions. I have also resolved the issues with finding the msvc compiler.
43	Task 10 (Frontend)	L	Much time investment was necessary for connecting the frontend and the backend. Additionally making the frontend actually

			work, use the backend properly and address feedback from Ruthger.
44	Task 25 resolving issues and wrap up	M	Finishing up the task, resolving the issue with using user directory instead of res to correctly load up the images to frontend. Added even more markdown formatting - code snippets and hyperlinks (not clickable yet).
45	Finish task 13	M	Start testing the implementation and fixing issues that come up.
46	task 5 Wiki	M	Adding entries to the wiki of libSMCE and smce-gd.
47	Research on task10 backend	M	Time was spent reading and testing how to implement functionality to solve the abi issue. But in the end Ryan told us not to implement this now. Research was also made in the area of vswhere, and compilers.

Name	Task ID	Contribution in %
Max	**Sprint 1**	
	1	20%
	3	100%
	5	20%
	6	20%
	7	20%
	9	60%
	Sprint 2	
	14	100%
	19	100%(not done)
	25	20%
	26	25%
	sprint 3	
	33	80%(almost done)
	Sprint 4	
	37	100%(not done)
	Sprint 5	
	40	20%
	45	100%
	46	100%
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%
	Sprint 3	
	27	20%
	28	50
	30	100
	31	50
	Sprint 4	
	38	50%
	39	100%
	Sprint 5	
	40	20%
	43	100%
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%
	SPRINT 3	
	27	10%
	32	50%
	33	20% (Almost done)
	SPRINT 4	
	36	50%
	SPRINT 5	
	40	20%
	41	100%
	44	30%
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%
	SPRINT 3	
	27	20%
	32	50%

	SPRINT 4	
	36	50%
	38	50%
	SPRINT 5	
	40	20%
	44	70%
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%
	Sprint 3	
	27	30%
	28	50%
	29	100%
	31	50%
	Sprint 4	
	34	100%

	35	100%
	Sprint 5	
	40	20%
	42	100%
	47	100%