Criterion B: Record of tasks

Task no.	Planned action	Planned outcome	Time estimated	Target completion date	Criterion
1	Initial brainstorming of possible problems to address	Arrive at a suitable problem to solve – improve the group creation process for organizer of the IB initiation course.	3 weeks	31/10/2018	A
2	Preliminary discussion with CS teacher	Discuss the suitability of the problem as an IA.	0:15	2/11/2018	A
3	Initial interview with the client	Propose a solution for the client; find out more about the group creating process and the client's requirements.	0:30	10/11/2018	A
4	Scenario establishment	Write down a transcript of the interview and success criteria.	1:00	10/11/2018	A
5	Interim discussion with CS teacher	Determine the feasibility of proposed solution in connection with success criteria.	0:15	12/11/2018	A
6	Interim discussion with the client	Confirmation of success criteria.	0:05	12/11/2018	A
7	Additional request made by the client	Address the request that created groups should be created non-deterministically.	0:10	14/11/2018	A
8	Preliminary GUI outline	List all GUI elements that should be included on individual screens.	0:30	24/11/2018	B
9	GUI sketch using computer software	Output a set of images showing the proposed GUI for consultation with the client.	4:00	1/12/2018	B
10	Consultation of GUI sketch with the client	Determine any amendments to GUI and success criteria.	0:15	2/12/2018	B
11	Updating success criteria	Success criteria reflects client's latest request from consultation – choosing activity name, group leader and adding notes to groups.	0:05	2/12/2018	A
12	Identify necessary data structures	Think about what types of files will be used to store student data and generated files	0:15	3/12/2018	B

13	Plan key algorithms	Outline all group generation algorithms in pseudo code. Determine relevant data structures. Create flowcharts for operations with text files.	3:00	2/12/2018	B
14	Updating GUI sketch using computer software	GUI follows new success criteria.	0:20	2/12/2018	B
15	Consultation of updated GUI sketch with the client	Approval from the client.	0:05	3/12/2018	B
16	Updating design documentation with annotated sketches	Link success criteria to application design.	2:00	3/12/2018	B
17	Design the classes	Create a list of classes and describe their functionality. Generate UML diagrams for the key classes.	1:40	3/12/2018	B
18	Creating a testing plan	Keep focus on the success criteria and know how they should be reflected in the product.	0:40	3/12/2018	B
19	Start working with the JavaFX GUI environment	Determine the abilities of the JavaFX API. Learn action handling, test various GUI elements to test their correct implementation.	1:00	3/12/2018	C
20	Implement JavaFX TableView to display student data	Test content loading into the table.	0:30	3/12/2018	C
21	Learn JavaFX object organisation	Be able to switch between multiple JavaFX screens.	0:10	3/12/2018	C
22	Create a dynamic list of panes (TitledPanes) with titled buttons	Each pane represents one group. The students in a group are shown as buttons belonging to a pane. Button caption is the student’s name.	0:40	4/12/2018	C
23	Handle actions for a dynamic number of buttons. Create class “StudentButton”.	Selecting group captains. Handle button clicks for buttons contained in panes (groups). By clicking on a button, the group’s leader is set to the name on that button.	0:20	4/12/2018	C

24	Plan JavaFX elements to constitute the “Create Groups” screen	Create the screen in Scenebuilder. Design appropriate back-end and action handling.	0:30	4/12/2018	C
25	Set a CSS stylesheet as the project’s root style	Provide a neat UI, as requested by the user. Test if all GUI elements have been altered with the CSS design.	0:10	4/12/2018	C
26	Implement and test switching between JavaFX Scene instances	Enhance the user experience.	0:15	5/12/2018	C
27	Save the program properly. Test if saved data is readable for the program again	Prevent the user from quitting the application without saving important data.	0:05	5/12/2018	C
28	Consume the mouse drag event on JavaFX TableView	Prevent unnecessary interaction with the application.	1:00	5/12/2018	C
29	CSS styling for TableView	Enhance user experience.	1:10	6/12/2018	C
30	Add students to the table. Test student adding	Add GUI components necessary to input data when adding students. Create interface for deleting students.	0:20	7/12/2018	C
31	Enable in-place editing on the table and test if it works	Enable the user to edit data in the table. Test if the data is updated in the underlying data structures.	1:30	7/12/2018	C
32	Create class “DataManager” and link it to front-end	Ensure proper manipulation of data.	0:30	7/12/2018	C
33	Create a custom dialog. Test if it is displayed correctly	Warn the user that the student to be added already exists.	0:30	7/12/2018	C
34	Load data from a text file	Test if data containing UTF-8 characters are loaded properly. User can read from a simple database.	0:20	7/12/2018	C
35	Store data to a text file	Test proper storage of UTF-8 characters. User can edit and add to a simple database.	0:10	7/12/2018	C
36	Create screen “Create students”	Add relevant GUI elements with action listeners to handle user input. Do CSS styling to enhance user experience.	5:00	8/12/2018	C

		Prepare function calls to back-end.			
37	Convert to non-static class representation in JavaFX.	Improve coupling traceability and thus maintainability.	1:00	9/12/2018	C
38	Implement own data structure “MyHashMap”	Add bespoke functionality (alerts about when the list of classes should be updated).	0:40	10/12/2018	C
39	Handle errors associated with the table’s backward data propagation after content edit	Prevent incorrect functionality of the program. Test if everything works correctly.	0:40	10/12/2018	C
40	Make adding students more intuitive	(De-)activate a text field based on which element is chosen in an associated combo box.	0:45	10/12/2018	C
41	Create a custom alert dialog	Make the user confirm if they want to quit the application.	0:10	11/12/2018	C
42	Create class “GroupGenerator”	Have all group creating algorithms in this class.	0:30	14/12/2018	C
43	Implement the first algorithm and integrate it with GUI elements	Test proper display of created groups.	0:30	14/12/2018	C
44	Clear created groups list after leaving “Create Groups” screen	Switch between a message and a scroll pane on “Create Groups” screen. Test if it works.	0:40	14/12/2018	C
45	Prevent the need for unnecessary alerts when adding students	Implement advanced input validation using Boolean Binding to support multiple interdependence patterns between GUI elements. Disable buttons accordingly. Test if it works.	1:10	17/12/2018	C
46	Implement and debug the 2nd algorithm	–	0:30	17/12/2018	C
47	Nest a combo box in the table	Simplify gender editing – a combo box simplifies gender editing compared to typing it manually.	1:20	18/12/2018	C
48	Implement and debug the 3rd and 4th algorithm	–	0:30	18/12/2018	C
49	Create class “Group”	Group students into groups after group creation to simplify	0:15	18/12/2018	C

		displaying and exporting of the groups created.			
50	Export groups after groups were created	Try to use a custom library for PDF file creation. ***[Note: The standard library for PDF generation in Java, iText, only offers paid licenses. Therefore, exporting to a plain text file was chosen instead.]***	3:00	18/12/2018	C
51		Prepare export to a text file. Format the file to enhance readability.	0:10	18/12/2018	C
52	Proposal presented to the client	Decision made to add online accessibility.	0:05	28/12/2018	A
53	Create a GitHub repository	Provide a platform for future distribution and ensure the safety of the source code.	0:10	19/12/2018	C
54	Add the field “note” in class “Group”	Enable the user to add notes to groups after groups were created.	0:30	20/12/2018	C
55	Prevent the user from clicking the button “delete” when no student is selected	Disable the remove button instance using Boolean Binding with the table’s selection property.	0:10	20/12/2018	C
56	Proposal presented to the client regarding data protection	Decision made to add encryption and user validation to protect data.	0:05	29/12/2018	A
57	Add user validation	Each user has to present him-/herself with credentials to gain access to the program.	1:25	30/12/2018	C
58	Add text file encryption	Implement AES for text-file encryption. ***[Note: Problems were encountered during implementation. Therefore, a different algorithm was used.]***	2:00	31/12/2018	C
59		Implement Caesar cypher for encryption. File with user data is unreadable outside the app. Test if it works properly.	1:00	31/12/2018	C
60	Create and test executable jar file. Test functionality on a	Test if the app works on different platforms and is	0:10	5/1/2018	C

	computer with a different OS.	ready to be shared among organizers.			
61	Fill the Development file	Include code snippets, explanations and justifications for the use of specific techniques.	5:30	10/1/201	C
62	Record footage	Demonstrate the program’s functionality and ideas for future development.		7/2/2019	D
63	Final consultation with the client	Determine which success criteria were met, as well as the product’s strengths and weaknesses.	0:15	8/2/2019	E
64	Evaluation	Evaluate the product, outline future course of development.	1:00	9/2/2019	E