Software Project Management

Comprehensive Plan

Group Members:

Lauren Robbins

Mark Zeagler

Nygel Jones

Bilash Paul

Sharmell Smith

****Those who contributed are highlighted and underlined****

Due: October 8, 2017

Table of Content

Project Description

The project management system will include the following set of requirements:

I. A general section where the user will be able to input specific information.

- **a.** High level description of the project
- **b.** The project manager's name
- **c.** A list of project team members (This should be able to be updated as the project progresses)
- **d.** A list of risk and risk status

II. A section where project requirements can be recorded.

- **a.** A feature will allow the user to enter functional and nonfunctional requirements.

III. A section where the user can monitor and track the project

- **a.** A feature shall be included that will allow the user to enter the number of hours (daily or weekly) that a person or persons has contributed to requirements analysis, designing, coding, testing, and project management.
- **b.** A feature that allows the user to view the total hours that has been contributed to requirements analysis, designing, coding, testing, and project management.

Deliverables for the Project Management System:

I. Project plan

- **a.** A description of the product, goals, and deliverables
- **b.** An assessment of the total cost in terms of resources
- **c.** A schedule for the project
- **d.** Risks and assumptions

II. Executable code

<u>Product Goals</u>

I. Allow functional and nonfunctional requirements to be tracked properly

- **a.** This includes their description, source, acceptance status, and completion status.
  - **i.** 98% of what is entered is correctly displayed

II. Requirements tracking, hour logging, and risk management will be centralized into a single utility that is user friendly

- **a.** This includes response timeliness, and ease of flow
  - **i.** 98% conformance to standards on the screen

III. Data will be saved and loaded so we can maintain a record of our progress over the course of the project

- **a.** This includes any data being entered will have the option to be saved. Also, any past information concerning projects can be loaded.
  - **i.** 98% of saves will be successful and 98% of loaded information will be successful and available to the customer in a timely manner.

<u>Project goals</u>

I. The project will remain on schedule and all deliverables will be completed by their due dates

- **a.** This includes all deliverables that need to be turned in by their due date
  - **i.** We will measure our progress by setting weekly check-ins (two at most) with all team members to make sure all deliverables are on track

II. The project will maintain good productivity levels

- **a.** Team members will be in good morale and all team members will equally contribute to deliverables
  - **i.** This will be measured through open communication in the form of discussion boards and group me. Everyone will have a safe

environment to express any opinions on the deliverables being created.

Processes and Methods used

We decide to use the Test-Driven Development (TDD) as it will be a practical method to develop and test as we go.Test-driven development (TDD) is a software development process that relies on the repetition of a very short development cycle: first the developer writes an (initially failing) automated test case that defines a desired improvement or new function, then produces the minimum amount of code to pass that test, and finally refractors the new code to acceptable standards.

Possible Risks/ Assumptions for Project

Reviewing our model for the project management system software, there appears to be a particular risk associated with our design. In the modules of classes, the effort type “ANY” may be superfluous and not used often in the code. Besides that, the modules are dependent on one another, but a positive is that many are loosely coupled, allowing software to be modified without any major changes in classes.

Cost and Expenses

Analyzing the requirements of this project, we have created two instances of how much it will cost depending on the client’s preferences.

In the first instance, we will take the minimum approach that requires less money spent but could factor the extent we can go in the project.

- **I.** Tools used would be the free version of Intellij IDE
- **II.** Hiring members that would have to be trained to produce our product.

III. Have meetings via online/ Telephone to minimize travel expenses.

IV. Create monthly goals for phases to be complete and keep team members focused at tasks

V. If lucky, we can contact a professional to oversee the team's progress and correct anything, free of charge.

In the second instance, we would create a budget so we can have an effective and larger support system in the development of the program.

I. To have all available coding and support for Intellij, it would cost: $700 per year. In case of a fallout in creating the program in a certain language, this will allow programs to convert to any fitting language.

II. Hire a Lead to help guide team members to develop a sturdy program. This lead would have previous experience which means they would need to be paid reasonably for their services. ($30 per hour).

III. Although it is great to have meetings online, it is more efficient to meet in person so things can be done in real time and problems are explained better than text message or video chat. We would try to keep the team members local so we will measure based on gas prices. At most the employees would drive 25 miles for meeting. ($0.12 gas price per mile X 25 miles)= 3 gallons required. Team members would pay $2.70 x 3= $8 for each trip to the meeting.

IV. Goals would be set with a reward at the end of phases to keep members motivated and give back for their hard work. (Company Credit and/or Gift Card). About $30.

□	Information
■	No/meaningless value
▨	Calculated value
🟨	Input/manual entry

Hours Multiplier: 1.3

PERT Analysis

Expected Hours: 345.8

Standard Deviation: 68.46666667

🟩 Starter Task	🟧 No slack (Critical Path)
🟥 Final Task (Critical path)	

				Labor Hours			Schedule Hours			PERT Average (Rounded Up)	Activity Tables				
Task	Description	Required Skills	Prerequisites	Optimistic Hours	Normal Hours	Pessimistic Hours	Optimistic Hours	Normal Hours	Pessimistic Hours		Early Start	Early Finish	Late Start	Late Finish	Slack
1 GUI	The visual representation and editor of the project data. It will contain a General Pane, a Requirements Pane, and an Hour Log Pane	JavaFX	■	68	150	263	88.4	195	341.9	202	■	■	■	■	■
1.1 General Page	A pane to display the current team, the project description, the project manager, and the risks involved in the project	JavaFX	■	23	54	107	29.9	70.2	139.1	75	■	■	■	■	■
1.1.1 Description Panel	Displays and edits the description of the project	JavaFX	■	2	4	9	2.6	5.2	11.7	6	■	■	■	■	■
1.1.1.1 Display Description	Display the description in an editable textbox. If the description is changed, indicate that it has been adjusted. Remove the indicator once the changes have been saved	JavaFX	2.8	1	2	5	1.3	2.6	6.5	4	34	38	46	50	12
1.1.1.2 Edit Description	Allow the user to update the description by typing in and editing the textbox. Allow the changes to be saved or reverted	JavaFX	1.1.1.1	1	2	4	1.3	2.6	5.2	3	38	41	50	53	12
1.1.2 Manager Panel	Allow the user to select a team member to be made manager. Setting a new person as manager will remove the old manager from the position. Team members will be displayed in a scrollpane as a list of buttons.	JavaFX	■	2	7	14	2.6	9.1	18.2	10	■	■	■	■	■
1.1.2.1 Display Manager	Indicate which team member is manager by printing their name at the top of the pane and highlighting the button of the appropriate person	JavaFX	1.9, 2.7	1	5	10	1.3	6.5	13	7	29	36	43	50	14
1.1.2.2 Edit Manager	Select a new manager by clicking the button with the person's name on it, and selecting "yes" on the confirmation pop-up window.	JavaFX	1.1.2.1	1	2	4	1.3	2.6	5.2	3	36	39	50	53	14
1.1.3 Members Panel	Displays and edits the members of the project team	JavaFX	■	11	24	53	14.3	31.2	68.9	35	■	■	■	■	■
1.1.3.1 Display Team Members	Display the team members in a list of buttons in a scrollpane	JavaFX	1.9, 2.7	3	5	10	3.9	6.5	13	8	29	37	29	37	0
1.1.3.2 Add Team Member	Click a button that prompts the user to add a new member by name	JavaFX	1.1.3.1	1	4	8	1.3	5.2	10.4	6	37	43	37	43	0
1.1.3.3 Edit Team Members	Click the button of the corresponding team member to edit their information	JavaFX	1.1.3.2	1	4	10	1.3	5.2	13	6	43	49	47	53	4
1.1.3.4 Display Team Member's WorkedHours	Click the button of the corresponding team member to display their reported hours	JavaFX	1.1.3.2	3	6	15	3.9	7.8	19.5	10	43	53	43	53	0
1.1.3.5 Delete Team Member	Click the button of the corresponding team member to prompt the user to delete the member	JavaFX	1.1.3.2	3	5	10	3.9	6.5	13	8	43	51	45	53	2
1.1.4 Risks Panel	Displays and edits the risks of the project	JavaFX	■	8	19	31	10.4	24.7	40.3	25	■	■	■	■	■
1.1.4.1 Display Risks	Display the risks in a scrollpane which shows a short description of the risk, the percentage likelihood that the issue may occur, and the dollar amount associated with correcting the risk	JavaFX	2.6	3	10	15	3.9	13	19.5	13	9	22	29	42	20
1.1.4.2 Add Risk	Click a button that prompts the user to add a new risk by description, percentage likelihood, and recovery cost	JavaFX	1.1.4.1	2	4	8	2.6	5.2	10.4	6	22	28	42	48	20
1.1.4.3 Edit Risk	Click the appropriate risk (label? Button?) to display a pop-up window that allows the risk to be edited	JavaFX	1.1.4.2	2	3	5	2.6	3.9	6.5	5	28	33	48	53	20
1.1.4.4 Delete Risk	Click the appropriate risk to display a pop-up window that prompts the user to delete the risk	JavaFX	1.1.4.2	1	2	3	1.3	2.6	3.9	3	28	31	50	53	22
1.2 Requirements Page	Displays and edits the project requirements	JavaFX	■	11	26	34	14.3	33.8	44.2	33	■	■	■	■	■
1.2.1 Display Requirements	Displays the requirements by item number, description, source, priority, acceptance status, and completion status	JavaFX	2.5	4	12	15	5.2	15.6	19.5	15	12	27	22	37	10
1.2.2 Add Requirements	Allows the user to add requirements based on a description, source, priority, acceptance status, and completion status	JavaFX	1.2.1	4	8	10	5.2	10.4	13	10	27	37	37	47	10
1.2.3 Edit Requirements (Including to mark complete?)	Click the requirement (label?) to display a pop-up that allows the user to edit its information	JavaFX	1.2.2	2	4	6	2.6	5.2	7.8	6	37	43	47	53	10

□	Information				Hours Multiplier:	1.3	**PERT Analysis**					Starter Task	No slack (Critical Path)			
■	No/meaningless value						Expected Hours:	345.8				Final Task (Critical path)				
▨	Calculated value						Standard Deviation:	68.46666667								
🟨	Input/manual entry															

				Labor Hours			Schedule Hours			PERT Average (Rounded Up)	Activity Tables				
Task	Description	Required Skills	Prerequisites	Optimistic Hours	Normal Hours	Pessimistic Hours	Optimistic Hours	Normal Hours	Pessimistic Hours		Early Start	Early Finish	Late Start	Late Finish	Slack
1.2.3 Delete Requirements	Click the requirement (label?) to display a pop-up that prompts the user to delete the requirement	JavaFX	1.2.2	1	2	3	1.3	2.6	3.9	3	37	40	50	53	13
1.3 Hour Log Pane	Displays the project hourlog distribution by activity, and allows new hours to be logged based on a daily orweekly submission timetable	JavaFX		7	18	33	9.1	23.4	42.9	25					
1.3.1 Display expended WorkedHours	Displays the distribution of worked hours by activity (i.e. requirements, designing, coding, etc.)	JavaFX		4	9	18	5.2	11.7	23.4	13					
1.3.1.1 Display Total WorkedHours	Displays the total worked hours for the project. This will be printed as a value above where the hours of each type are listed	JavaFX	2.1.1, 2.1.2, 2.2	2	4	8	2.6	5.2	10.4	6	23	29	40	46	17
1.3.1.2 Display hours by WorkedHourType	Displays the how many hours of each type have been worked. This displays the value, as well as a progress bar that shows the percentage of hours by type divided by the total hours	JavaFX	1.3.1.1	2	5	10	2.6	6.5	13	7	29	36	46	53	17
1.3.2 Submit new WorkedHours	Allows the user to edit some fields to create a new WorkedHours for submission. Will catch and handle errors thrown by the WorkedHours constructor when invalid WorkedHourTypes are used. The user will select the submitting member via a PersonButtonScrollPane, which will display the person's name and ID in the submission form. Then the user will be allowed to complete the form by specifying the type of work done and the duration.	JavaFX		3	9	15	3.9	11.7	19.5	12					
1.3.2.1 Force by schedule standards (day or week)	Allows the user to select a standard time-frame for WorkedHours submission. This can either be done on the first overall submission of WorkedHours for a Project, or can be done in the Settings window. After this, if the user attempts to switch timeframes, an error is displayed (?)	JavaFX	2.1, 2.2	1	5	7	1.3	6.5	9.1	7	23	30	40	47	17
1.3.2.2 Add effort by contributing Person and WorkedHourType	Use the entered data to push a new WorkedHour submission to the Team, which will register it in the ProjectHourLog	JavaFX	1.3.2.1	2	4	8	2.6	5.2	10.4	6	30	36	47	53	17
1.4 Help Page	Used to display helpful information to the user	JavaFX		5	10	18	6.5	13	23.4	14					
1.4.1 Pop-up Stage	Will use the PopupStage library added to this project	JavaFX	--	2	5	10	2.6	6.5	13	7	0	7	39	46	39
1.4.2 Display helpful information	Will display helpful information like "Contact us if you need help"????	JavaFX	1.4.1	3	5	8	3.9	6.5	10.4	7	7	14	46	53	39
1.5 About Page	Will display information about us	JavaFX		3	5	9	3.9	6.5	11.7	7					
1.5.1 Pop-up Stage	Will use the PopupStage library added to this project	JavaFX	--	2	3	5	2.6	3.9	6.5	5	0	5	45	50	45
1.5.2 Display information about us! :D	Will provide an email to contact????	JavaFX	1.5.1	1	2	4	1.3	2.6	5.2	3	5	8	50	53	45
1.6 Settings Page	Will allow the user to edit information about how the GUI handles the Project	JavaFX		7	18	35	9.1	23.4	45.5	25					
1.6.1 Pop-up Stage	Will use the PopupStage library added to this project	JavaFX	--	3	10	20	3.9	13	26	14	0	14	26	40	26
1.6.2 Allow settings to be changed	Certain settings can only be changed on initialization of the project, and will be locked in once the first submissions are made. There will need to be hover-text or a pop-up that says this	JavaFX	1.6.1	1	2	5	1.3	2.6	6.5	4	14	18	40	44	26
1.6.3 Settings for:	Settings will include the Day/Week standard, ...	JavaFX		3	6	10	3.9	7.8	13	9					
1.6.3.1 Day/Week standard	Will force the user to maintain the Day/Week standard they adopted for the project. If they attempt to submit by a different standard, an error will be displayed	JavaFX	1.6.2, 2.2.2	3	6	10	3.9	7.8	13	9	18	27	44	53	26
1.7 Load/Save Pages	Will allow the user to save the current project or load a new one	JavaFX		7	11	15	9.1	14.3	19.5	15					
1.7.1 Pop-up Stage	This will use the FileChooser library from the Java standard library	JavaFX	--	2	3	4	2.6	3.9	5.2	4	0	4	45	49	45
1.7.2 If loading or exiting, prompt user to save first	If the current project is about to be discarded, prompt the user to save first	JavaFX	1.7.1, 2.8.5	2	3	4	2.6	3.9	5.2	4	34	38	49	53	15

□	Information
■	No/meaningless value
▨	Calculated value
🟨	Input/manual entry

Hours Multiplier: 1.3

PERT Analysis	
Expected Hours:	345.8
Standard Deviation:	68.46666667

Starter Task	No slack (Critical Path)
Final Task (Critical path)	

				Labor Hours			Schedule Hours			PERT Average (Rounded Up)	Activity Tables				
Task	Description	Required Skills	Prerequisites	Optimistic Hours	Normal Hours	Pessimistic Hours	Optimistic Hours	Normal Hours	Pessimistic Hours		Early Start	Early Finish	Late Start	Late Finish	Slack
1.7.3 Save	On first “Save” or “Save As”, prompt user to specify file name/location. On subsequent uses of “Save”, use that location and filename (persist across opening/closing of GUI)	JavaFX	1.7.4, 2.8.4	2	3	4	2.6	3.9	5.2	4	33	37	49	53	16
1.7.4 Save As	On “Save As”, save as the specified filename/location each time	JavaFX	1.7.1, 2.8.4	1	2	3	1.3	2.6	3.9	3	30	33	50	53	20
1.8 PersonButton	A Button that holds an instance of a Person, and displays their name. Different actions can be provided, so that it can fill the needed requirements of the TeamMembersPane, ManagerPane, and the TeamPane on the HourLog pane	JavaFX, Inheritance	2.1.2.1	2	3	4	2.6	3.9	5.2	4	3	7	18	22	15
1.9 PersonButtonScrollPane	A scrollpane specifically made to handle PersonButtons. Can handle the buttons by either Person or by PersonButton. If just a Person is used, it either locates the button that holds it, or will create a new button for that Person	JavaFX, Inheritance	1.8	3	5	8	3.9	6.5	10.4	7	7	14	22	29	15
2 Back-end System	The internal mechanics that make the GUI work			56	108	177	72.8	140.4	230.1	145					
2.1 Team organization	All classes pertaining to the team, including the Team.class, Person.class, and the HourLog package			18	27	36	23.4	35.1	46.8	36					
2.1.1 Team class	A class used to represent and act as a team. It holds the Person team members, can identify the manager, and should handle most interactions between the Team package and outside classes			11	16	21	14.3	20.8	27.3	21					
2.1.1.1 Add Person instances	Adds a new Person to the team		2.1.2.1	2	3	4	2.6	3.9	5.2	4	3	7	9	13	6
2.1.1.2 Remove Person instances	Removes the Person from the team. If the Person is still represented in the HourLog as a contributor, the instance will not be removed from memory, though the Person will no longer be directly accessible		2.1.1.1	3	4	5	3.9	5.2	6.5	6	7	13	13	19	6
2.1.1.3 Provide Total Team WorkedHours	Provide a way for outside classes to get information from the HourLog via the Team class		2.2.1.2.1	2	3	4	2.6	3.9	5.2	4	18	22	19	23	1
2.1.1.4 Provide Team WorkedHours for given WorkedHourType	Provide a way for outside classes to get information from the HourLog via the Team class		2.1.2.1, 2.2.1.2.1	2	3	4	2.6	3.9	5.2	4	18	22	19	23	1
2.1.1.5 Provide Team WorkedHours for given Person	Provide a way for outside classes to get information from the HourLog via the Team class		2.1.2, 2.2.1.2.2	2	3	4	2.6	3.9	5.2	4	19	23	19	23	0
2.1.2 Person class	A class to represent a member on a Team			7	11	15	9.1	14.3	19.5	15					
2.1.2.1 Person data	Holds the person's name, a unique ID (for people with the same name), and whether or not the person is a manager. Reports hours to the Team, which registers it in the HourLog		--	1	2	3	1.3	2.6	3.9	3	0	3	6	9	6
2.1.2.2 Report WorkedHours to ProjectWorkedHours	Reports hours to the Team. (Throw exception if submission is not in accordance with Weekly/Daily timeframe?)		2.1.1, 2.2.1.1	2	3	4	2.6	3.9	5.2	4	15	19	19	23	4
2.1.2.3 Promote to Manager	Promotes the Person to a Manager		2.1.2.1	2	3	4	2.6	3.9	5.2	4	3	7	49	53	46
2.1.2.4 Demote to Programmer	Demotes this Person from Manager		2.1.2.1	2	3	4	2.6	3.9	5.2	4	3	7	49	53	46
2.2 Hour Log	Keeps track of the submitted hours, and provides information on how many hours certain members have worked, or how many hours have been worked on a certain type of job type (i.e. requirements, designing, etc.)			9	19	31	11.7	24.7	40.3	26					
2.2.1 ProjectHourLog class	Stores WorkedHour instances, and retrieves them based on specified parameters			7	14	22	9.1	18.2	28.6	19					
2.2.1.1 Stores all WorkedHours instances for the project	Store them in the order they are submitted. Do not worry about sorting		2.2.2	3	5	7	3.9	6.5	9.1	7	8	15	9	16	1
2.2.1.2 Can provide WorkedHours sum of	Retrieves the number of hours worked by the below parameters			4	9	15	5.2	11.7	19.5	12					
2.2.1.2.1 WorkedHourType	Retrieves the number of hours worked by the given WorkedHourType		2.2.1.1, 2.2.3	1	2	3	1.3	2.6	3.9	3	15	18	16	19	1
2.2.1.2.2 Person	Retrieves the number of hours worked by the given Person		2.2.1.1, 2.1.2.1	1	2	3	1.3	2.6	3.9	3	15	18	16	19	1

□	Information
■	No/meaningless value
▨	Calculated value
🟨	Input/manual entry

Hours Multiplier: 1.3

PERT Analysis	
Expected Hours:	345.8
Standard Deviation:	68.46666667

Starter Task	No slack (Critical Path)
Final Task (Critical path)	

				Labor Hours			Schedule Hours			PERT Average (Rounded Up)	Activity Tables				
Task	Description	Required Skills	Prerequisites	Optimistic Hours	Normal Hours	Pessimistic Hours	Optimistic Hours	Normal Hours	Pessimistic Hours		Early Start	Early Finish	Late Start	Late Finish	Slack
2.2.2 WorkedHours class	Records who worked what number of hours for a given time period, and what type of work they were doing		2.2.3	1	4	8	1.3	5.2	10.4	6	2	8	3	9	1
2.2.3 WorkedHourType enum	Any - Requirements - Design - Coding - Testing - Project_Management		--	1	1	1	1.3	1.3	1.3	2	0	2	1	3	1
2.3 Load/Save Configuration	Load/save the GUI/Project configuration	File Input/Output Steams	■	9	19	27	11.7	24.7	35.1	25	■	■	■	■	■
2.3.1 Initialize new config file if none present	Creates a new config file, and loads it with the default settings	File	--	2	4	5	2.6	5.2	6.5	5	0	5	32	37	32
2.3.2 Read configuration from config file	Reads the settings from an existing config file	Scanner	2.3.3	2	3	4	2.6	3.9	5.2	4	21	25	49	53	28
2.3.3 Save configuration	Saves the current configuration to the config file		■	5	12	18	6.5	15.6	23.4	16	■	■	■	■	■
2.3.3.1 Save Configuration	Writes the current configuration to the file	FileOutputStream	2.3.1	1	4	6	1.3	5.2	7.8	5	5	10	37	42	32
2.3.3.2 Do not lose information from current file	Ensure that any other text is not overwritten (i.e. just replace old values with the new ones)		2.3.3.1	4	8	12	5.2	10.4	15.6	11	10	21	42	53	32
2.4 Logging	Output descriptive status updates to the log	FileOutputStream	■	5	13	33	6.5	16.9	42.9	20	■	■	■	■	■
2.4.1 Print updates to log w/ timestamp	Print out the status message with a date/time token in front to indicate when the message was printed		--	3	5	10	3.9	6.5	13	8	0	8	37	45	37
2.4.2 Redirect System.out and System.err to log	Redirect System.out and System.err to the log so that any crash messages or other messages using the System printstream will be displayed in the log		2.4.1	1	5	15	1.3	6.5	19.5	8	8	16	45	53	37
2.4.3 Clear old logs to prevent build-up	Make sure that log files do not build up beyond a certain limit, probably 5		--	1	3	8	1.3	3.9	10.4	5	0	5	48	53	48
2.5 Requirements	Store the requirements and requirement data		■	6	10	14	7.8	13	18.2	13	■	■	■	■	■
2.5.1 Requirement class	Store the requirement data. Specifically, the Source, description, item number, acceptance status, completion status, and priority		2.5.3, 2.5.4, 2.5.5	1	3	5	1.3	3.9	6.5	4	2	6	15	19	13
2.5.2 RequirementsList	Store all of the requirements. Also retrieves a separate list of requirements based on a parameter, such as Priority level or Acceptance status	Inheritance	■	5	7	9	6.5	9.1	11.7	10	■	■	■	■	■
2.5.2.1 Extend LinkedList<Requirement>			2.5.1	1	2	3	1.3	2.6	3.9	3	6	9	19	22	13
2.5.2.2 Return Requirements based on parameter			2.5.2.1	1	2	3	1.3	2.6	3.9	3	9	12	50	53	41
2.5.3 Priority Enum	Unassigned - Critical - High - Medium - Low		--	1	1	1	1.3	1.3	1.3	2	0	2	13	15	13
2.5.4 Status Enum	Pending_Consideration - Accepted_Current - Accepted_Post_Release - Rejected		--	1	1	1	1.3	1.3	1.3	2	0	2	13	15	13
2.5.5 Completion Status Enum	Not_Started - In_Progress - Awaiting_QA - Completed		--	1	1	1	1.3	1.3	1.3	2	0	2	13	15	13
2.6 Risks			■	2	5	11	2.6	6.5	14.3	8	■	■	■	■	■
2.6.1 Risk class			--	1	3	6	1.3	3.9	7.8	5	0	5	33	38	33
2.6.2 RiskList class			2.6.1	1	2	5	1.3	2.6	6.5	4	5	9	38	42	33
2.7 TeamPresenter interface	An interface used by JavaFX objects that display the Team and its members. Specifies methods AddMember, RemoveMember, and UpdateTeam		2.1	2	4	6	2.6	5.2	7.8	6	23	29	23	29	0
2.8 Project class			■	5	11	19	6.5	14.3	24.7	15	■	■	■	■	■
2.8.1 Initialize empty Team			2.1.1	1	2	3	1.3	2.6	3.9	3	23	26	42	45	19
2.8.2 Intialize new RequirementsList			2.5.2	1	2	3	1.3	2.6	3.9	3	12	15	42	45	30
2.8.3 Initialize new RisksList			2.6.2	1	2	3	1.3	2.6	3.9	3	9	12	42	45	33
2.8.4 Save current Project		File Input/Output Steams	2.8.1, 2.8.2, 2.8.3	1	2	5	1.3	2.6	6.5	4	26	30	45	49	19
2.8.5 Load saved Project from file		File Input/Output Steams	2.5.4	1	3	5	1.3	3.9	6.5	4	30	34	45	49	15
			Totals:	**124**	**258**	**440**	**161.2**	**335.4**	**572**		**MAX:**	**53**			

Scheduling Assignments

Hour	Mark	Nygel	Lauren	Sharmell	Bilash
0	2.2.3	2.1.2.1	2.5.3	2.5.4	2.4.3
1					
2	2.2.2		2.5.5	▨	
3		2.1.2.3		▨	
4			2.1.1.1	2.5.1	
5					2.6.1
6					
7		2.1.2.4			
8	2.2.1.1		2.1.1.2	2.5.2.1	
9					
10					2.6.2
11		1.8		2.5.2.2	
12					
13					
14			▨	2.8.2	2.8.3
15	2.1.2.2	2.2.1.2.1	2.2.1.2.2		
16					
17				1.9	2.4.1
18		2.1.1.3	2.1.1.4		
19	2.1.1.5				
20					
21					
22		▨	1.7.1		
23	2.7	1.3.1.1			
24					
25					2.4.2
26					
27					
28					

29		1.3.1.2	1.3.2.1	1.2.1	2.4.2
30	1.1.3.1				
31					
32					
33			1.3.2.2		2.8.1
34					
35					
36		1.1.2.1			2.8.4
37					
38	1.1.3.2				
39			1.4.1	1.2.2	
40					1.6.1
41					
42					
43		2.3.1			
44	1.1.3.4				
45					
46			1.1.3.5		
47					
48		2.3.3.1			
49				1.1.4.1	
50					
51					
52					
53		2.3.3.2			
54	2.8.5		1.5.1		1.6.2
55					
56					
57					
58	1.1.1.1				
59					

60	1.1.1.1		1.7.4		1.6.3.1
61					
62	1.1.1.2		1.7.3	1.1.4.2	
63					
64		2.3.2			
65	1.2.3				
66			1.1.2.2		
67					1.7.2
68		1.4.2			
69			1.1.3.3	1.1.4.3	
70					
71	1.1.4.4				
72					
73					
74					