SWE20001: Managing Software Projects

Name: S M Ragib Rezwan

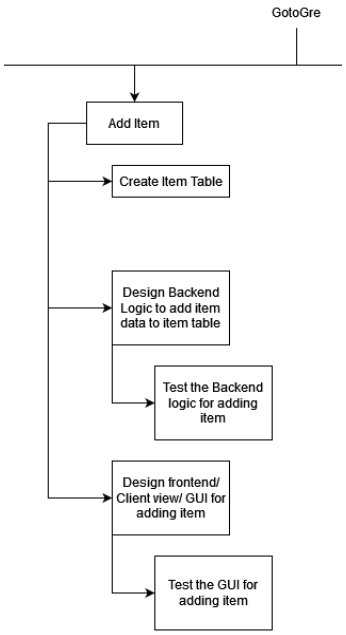
ID: 103172423

Tutor: Naveed Ali | 12.30pm Tuesday | EN31

Project Proposal1 : GotoGro :

In the group work task (08P), we had decided upon the following WBS where we have decided to use product based approach to ensure that no product planned to be developed in sprint 1 is missing from the task breakdown structure/chart.

Now here, since the task only asked for WBS and effort estimation of a ***single backlog item***, I am only giving the partial for the WBS and speaking about that part only in the effort estimation. Here that part is the "Add Item" functionality.



Here, you can see that we have the following leaf nodes the Backlog item of "Add Item" functionality:

1. Creating Item Table

2. Designing the backend functionality and logic to add Item to the item table
3. Testing the backend functionality and logic to add Item to the item table
4. Designing the GUI feature for adding the Items
5. Testing the GUI feature for adding the Items

Now here, considering Ideal Time, Ideal Effort and Effective effort, we can notice the following table:

[Note: here I am using the fact the each person will only be able to productively work for 70% of their time and using that in my calculation of real effort for each of the leaf nodes.]

[Note: the leaf node numbers refer to their task names here and I have not written their names as it would be too long and their brief abbreviations would be confusing]

Task no (following the list created above)	Ideal Time	Ideal Effort	Real Effort	Justification
1	60 mins	60 mins	120 Mins	Here we are creating the Item table in the database in C# and then verify that it works by using a test case data (this will be removed later on after troubleshooting is over). Although this should ideally take 60 mins; considering the fact that we are working in C# and the different types of errors that usually pop up, we have decided to keep some leeway time for troubleshooting and research. Furthermore, there is also the fact that each person will only be effectively working 70% productively.
2	60 mins	60 mins	120 Mins	Here we are creating the functionality and logic to add Items to the Item table in C# and then verify that it works by using a test case data (this will be removed later on after troubleshooting is over). Although this should ideally take 60 mins; considering the fact that we are working in C# and the different types of errors that usually pop up, we have decided to keep some leeway time for troubleshooting and research. Furthermore, there is also the fact that each person will only be effectively working 70% productively.
3	60 mins	60 mins	120 Mins	Here we are vigorously testing the functionality and logic to add Items to the Item table in C# following our Definition of done (noted before in previous tasks) Although this should ideally take 60 mins; considering the

				fact that we are working in C# and the different types of errors that usually pop up, we have decided to keep some leeway time for troubleshooting and research. Furthermore, there is also the fact that each person will only be effectively working 70% productively.
4	60 mins	60 mins	120mins	Here we are creating the GUI feature for adding the Items in C# and then verify that it works by using a test case table (this will be removed later on after troubleshooting is over). Although this should ideally take 60 mins; considering the fact that we are working in C# and the different types of errors that usually pop up, we have decided to keep some leeway time for troubleshooting and research. Furthermore, there is also the fact that each person will only be effectively working 70% productively.
5	60 mins	60 mins	120 Mins	Here we are vigorously testing the GUI feature for adding the Items in C# following our Definition of done (noted before in previous tasks) Although this should ideally take 60 mins; considering the fact that we are working in C# and the different types of errors that usually pop up, we have decided to keep some leeway time for troubleshooting and research. Furthermore, there is also the fact that each person will only be effectively working 70% productively.

Total estimation time: **10 hrs** (=600 mins)

Now, in our project we are planning on using pair programming method where two people will be tasked to code for a single functionality. This is done for three reasons:

a) Following the information in the lecture about “trunk factor” in risk and by also having firsthand experience of it (as a member of our team had been in a car crash in the early weeks and thus had missed for several later weeks due to recuperation), we have decided to proceed with the pair programming method in order to ensure that even if one person is unable to deliver, the other one can and thus the functionalities present in sprint 1 can be delivered in time (whilst fulfilling the definition of done related to them)

b) Different people have different way of approaching a single problem and thus can come up with different unique backend logic to provide the same outcome. Thus by using this system we can

ensure that we can have a variety of method logics and thus can choose the best one in a short period of time

c) Last but not the least, our team has issue in the skills required for the task as almost half of our members have never done such a project before from end to end, let alone in C#. So by following this technique, we can provide them with the time and guidance needed to hone their skills, whilst not delaying the progress of the project itself.

Thus, we end up with the time of **20hrs** (= 2 * Total time taken). Since our intended team time is **96hrs**, this backlog item is covering just **20.83%** of it. This is adequate as it is 1 of the 4 backlog items *(Adding members, Adding Item, Managing Item and Adding User Accounts)* we are doing in the Sprint-1 and so it covering around **20%** in terms of effective time is adequate.