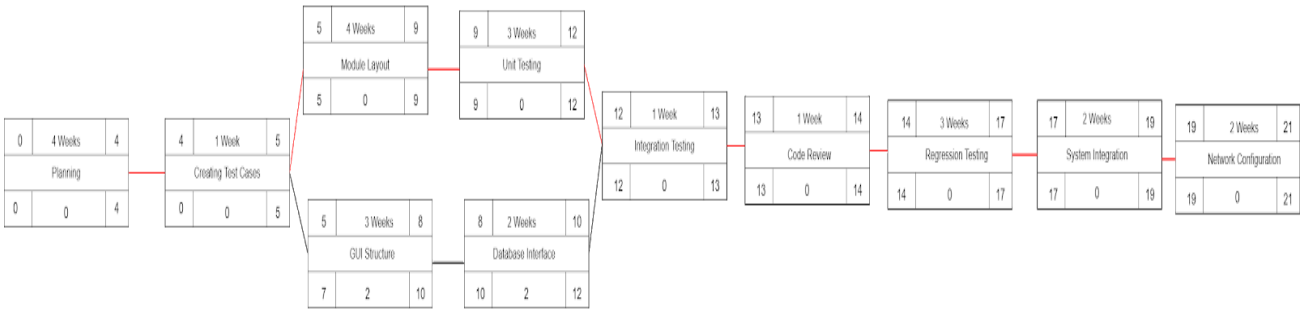
SOFE 3490U Laboratory PROJECT: LAB 4

Durreshahwar Arif (100587401)

Ibaada Arif (100655270)

Matthew Brown(100670025)

Activity Chart



Risks within Project Activities

Activity - Planning

Risk 1: Outlining wrong expectations

This activity involves setting the schedule, deliverables, and strategies that are associated with the project. If the planning is poor, the scope and direction for the project will not meet the expectations of the students and lecturers that will use it to enhance the learning experience. The countermeasure for this risk is to provide some kind of survey for students to complete to better understand what they want in a learning tool.

Risk 2: System is does not have high usability

The purpose of this project is to give students an added tool to enhance their learning, not to hinder it. Planning stages involve how to best make the system easy to interact with. This way, students are encouraged to use it and not disregard it. To mitigate the risk would be to refer to a past project such as the Blackboard course management system which may have a similar structure to our application. Blackboard is very easy to use and does not require much effort to navigate.

Activity - GUI Structure

Risk 1: The graphical user interface adds complexity and may compromise security

With computing environments becoming more sophisticated, security will suffer. Usability and security have an inverse relationship. If one of these attributes is the focus of a project, normally the other will suffer. To mitigate this risk, we could introduce some kind of security testing. Adding security testing as a part of the SDLC from beginning to end development makes finding and fixing vulnerabilities more efficient and less expensive than after the application is made available to students.

Risk 2: A poorly developed GUI would result in a poor experience for users

GUI provides users with an interface to communicate with software and various functionalities, however if there are defects in code or misunderstood requirements, it may lead poorly developed GUI. To mitigate this risk, we may conduct interviews with stakeholders to learn what they would like to see in their software. User testing would also gauge how the user interacts with the system. We can analyze how they use the system and answer questions such as “Why the user hesitates when selecting an option”. And how do their eyes scan the screen as they search.

Activity - Unit Testing

Risk 1: The code may be organized differently to conduct test

The purpose of unit testing is to break down programs into modular multi layered chunks and check for defects. The risk occurs when the code is split up so much to support testing processes, that it actually destroys system architecture and code comprehension. To mitigate this risk, good documentation and a strong understanding of code is important so the functionality is preserved and the focus of the project is not lost.

Risk 2: Tests could become more complex than actual code

Sometimes when testing software that is large, test cases tend to become harder to generate and become more complex than the actual code itself. This risk can be mitigated by making them short and concise. The principle of unit testing is testing modules, or smaller pieces, of the overall software. The code should not be broken

down so much that it changes the structure, but concise enough to conduct a reliable test.

Activity - Code Review

Risk 1: Human Error

If done badly, code reviews can have a negative impact on project performance and atmosphere. Code review consists of giving and receiving feedback. Sometimes mistakes will be made, especially when we approach it with the wrong mindset to conduct a review. To mitigate the risk of human error, reviewing fewer than 400 lines of code. The brain can only effectively process so much information at a time; beyond 400 LOC, the ability to find defects diminishes.

Risk 2: Rubber stamping code reviews

When code reviews are mandatory, developers may acquire the habit of “rubber stamp” code reviews. In these reviews, all value is lost as the code reviewer does understand the code. This defeats the purpose of the code review. To mitigate this risk, our code reviewers should select code they feel they can understand and feel comfortable writing viable solutions.

Activity - Regression Testing

Risk 1: Time Consuming

Regression testing can take quite a bit of time to complete. Regression testing also involves running tests on existing tests as well. The best way to mitigate this risk is to limit the number of regression tests following code reviews. Which means code should be written clearly and well to avoid multiple reviews to force more testing.

Risk 2: Time bound

Regression testing has strict deadlines that must be met. So testers have the tendency to finish testing quickly. The risk is that critical tests could be skipped to meet deadlines. The way to mitigate this risk, is to allocate an appropriate amount of time to testing our application. Testing is important and if want to maximize the success of our product,

Activity - System Integration

Risk 1: Coupling

Coupling is the measure of the degree of interdependence between modules. Unfortunately coupling increases the instability of a system. To mitigate this risk, decoupling can be introduced between modules by making them more independant.

Risk 2: Essentiality

Essentiality occurs when a key feature or features of the system will be unavailable if the component does not work properly. This risk is serious because all of the work on the application could go to waste if it cannot be integrated into the Blackboard course management system. The best way to mitigate this risk to come up with testing integration scenarios. This way, we can prepare the issues that may occur when integrating our application to blackboard.

Resource Allocation

Activity	Resources	Start date	End Date
Planning	Matthew Brown, Durreshahwar Arif, Ibaada Arif	February 1st 2019	March 1st 2019 -February 22nd is family day. No work will be done that day
Creating Test Cases	Durreshawar Arif	March 1st 2019	March 8th 2019
GUI Structure	Ibaada Arif, Durreshahwar Arif	March 8th 2019	March 29th 2019
Module Layout	Ibaada Arif, Durreshahwar Arif	March 8th 2019	April 6th 2019
Database Interface	Matthew Brown, Durreshahwar Arif, Ibaada Arif	March 29th 2019	April 12th 2019
Unit Testing	Matthew Brown	April 6th 2019	April 27th 2019 -April 19th is Good friday No work will be done on that day

Integration Testing	Matthew Brown	April 27th 2019	May 4th 2019
Code Review	Ibaada Arif, Durreshahwar Arif	May 4th 2019	May 11th 2019
Regression Testing	Matthew Brown	May 11th 2019	June 1st 2019 -Victoria day is May 20th No work will be done on that day
System Integration	Matthew Brown, Ibaada Arif	June 1st 2019	June 15th 2019
Network Configuration	Matthew Brown, Durreshahwar Arif, Ibaada Arif	June 15th	June 29th 2019

Gantt Chart