Senior Project Group EVE

Professor Johnson

2-5-25

Assignment #1

Problem 1.1:

The basic tasks that all software engineering projects must handle are requirements gathering, high-level design, low-level design, development, testing, deployment, maintenance, and wrap-up.

Problem 1.2:

Requirements Gathering- Learn about the customer's wants and needs.

High-Level Design- Describe the major moving parts of the project and how they tie in with each other.

Low-Level Design- Explain in greater detail how to build the parts of the project so that they can actually be implemented by the programmers.

Development- Write the code to implement the application

Testing- Test out and try to use the application in as many ways as possible to see if there are there any noticeable bugs or other flaws.

Deployment- Open the application to be available to the users

Maintenance- Work on enhancing, adding to, bug fixes, and updates to the program

Wrap-up- Think about the history of the project to figure out the good things and the bad things that happened so you can learn from them and apply that knowledge in future endeavors.

Problem 2.4:

Google Docs crosses out what you have deleted and then shows what you have added

compared to the previous version's history while also showing the date and time that each version was modified. It also has other tracking tools to let you see who has modified a document and when if there are other collaborators. Some differences between GitHub and Google Docs regarding version control include that GitHub uses commit messages instead of edit history, GitHub can "roll back" to previous commits but not so quickly in Google Docs. Both provide version control, collaboration, and reviewing changes.

Problem 2.5:

JBGE stands for Just Barely Good Enough. It describes the idea that you shouldn't write any more code documentation or other comments than what is absolutely necessary.

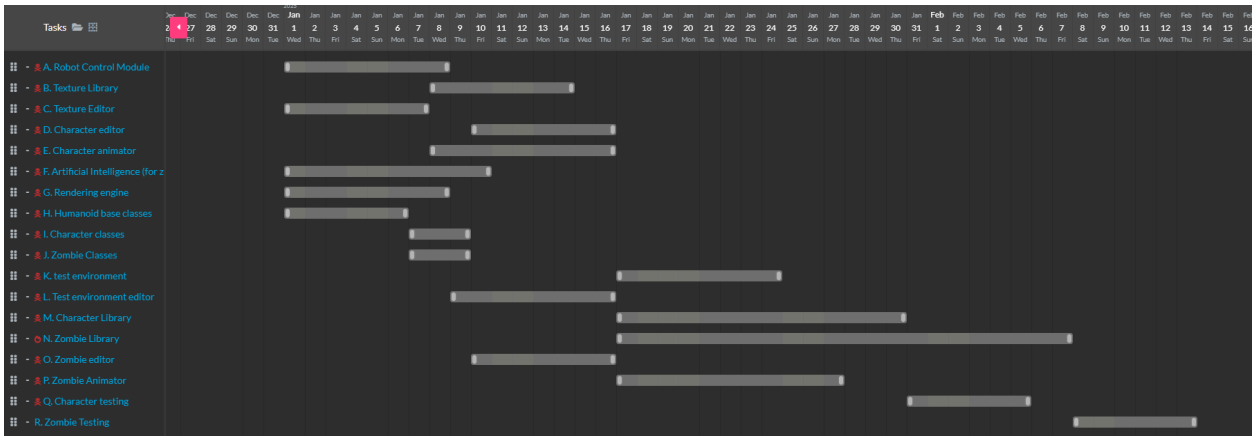
Problem 3.2:

a. The total expected project duration should be around 32 days
b. A (5) → D (6) → E (7) → M (9) → Q (4)
c. 5+6+7+9+4+14(weekends) = 44 days

Problem 4.2:

TODO

Problem 4.4:



Problem 4.6:

Deus ex machina problems can be treated the same way as something like unexpected sick leave. You can add tasks at the end of the schedule to account for these unexpected occurrences. If one of these things happens, put the lost time into the schedule.

Problem 4.8:

The first biggest mistake you can make while tracking tasks is doing nothing when a task slips. Pay close attention to the task so that you're ready to do something if it's not going well. The second biggest mistake is putting more people on a task and thinking that it will significantly cut down the total time. Unless one of the new people is a specialist, catching them up might just make the task take even longer.

Problem 5.1:

Five characteristics of good requirements are clear (easy to understand), unambiguous, consistent, prioritized, and verifiable.

Problem 5.3:

a. Allow users to monitor uploads/downloads while away from the office. **Business**
b. Let the user specify website log-in parameters such as an Internet address, a port, a username, and a password. **User, Functional**
c. Let the user specify upload/download parameters, such as number of retries if there's a problem. **User, Functional**
d. Let the user select an Internet location, a local file, and a time to perform the upload/download. **User, Functional**
e. Let the user schedule uploads/downloads at any time. **Nonfunctional**
f. Allow uploads/downloads to run at any time. **Nonfunctional**
g. Make uploads/downloads transfer at least 8 Mbps. **Nonfunctional**

h. Run uploads/downloads sequentially. Two cannot run at the same time. **Nonfunctional**

i. If an upload/download is scheduled for a time when another is in progress, the new task waits until the other one finishes. **Nonfunctional**

j. Perform scheduled uploads/downloads. **Functional**

k. Keep a log of all attempted uploads/downloads and whether they succeeded. **Functional**

l. Let the user empty the log. **User, Functional**

m. Display reports of upload/download attempts. **User, Functional**

n. Let the user view the log reports on a remote device such as a phone. **User, Functional**

o. Send an e-mail to an administrator if an upload/download fails more than its maximum retry number of times. **User, Functional**

p. Send a text message to an administrator if an upload/download fails more than its maximum retry number of times. **User, Functional**

Implementation requirements is the only one empty as you are assumed to have performed uploads and downloads already, so you have everything you need.

Problem 5.9:

You could have score/score-keeping by how many parts of Mr. Bones' skeleton appear by the time you fill in the word. You could also have it allow the user to put in a guess for the whole word all at once instead of letter by letter. You could have multiple skill levels, such as easy, medium, and hard, depending on the length of the word and its complexity.