

1.1

- Requirements Gathering
- High-Level Design
- Low-Level Design
- Development
- Testing
- Deployment
- Maintenance
- Wrap-up

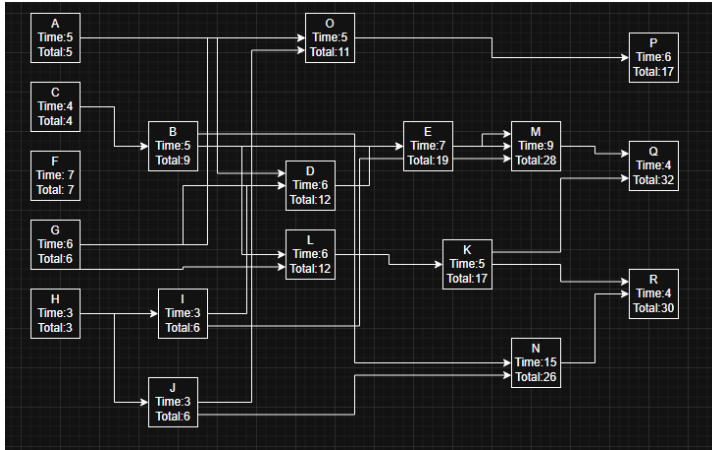
1.2

- Requirements Gathering: Learning what the customer want and needs.
- High-Level Design: Create a general overview of the product components and functionality.
- Low-Level Design: Give more detail to each component and function of the application as guidance for implementing them.
- Development: Write code to implement the application.
- Testing: Test the product for bugs and flaws by putting it under different circumstances.
- Deployment: Release the application to the users.
- Maintenance: Implement updates, and bug fixes to the application throughout its life cycle.
- Wrap-up: Evaluate the project to determine what was good and what was bad to learn from the mistakes and do better in future projects.

2.4 The changes in the document's history is distinct when going through the version history. Changes are highlighted and removal are strikethrough. I can also see who made the changes and when the changes were made. GitHub version control is similar but could works with multiple files and has branches of versions instead of the linear versions in google docs.

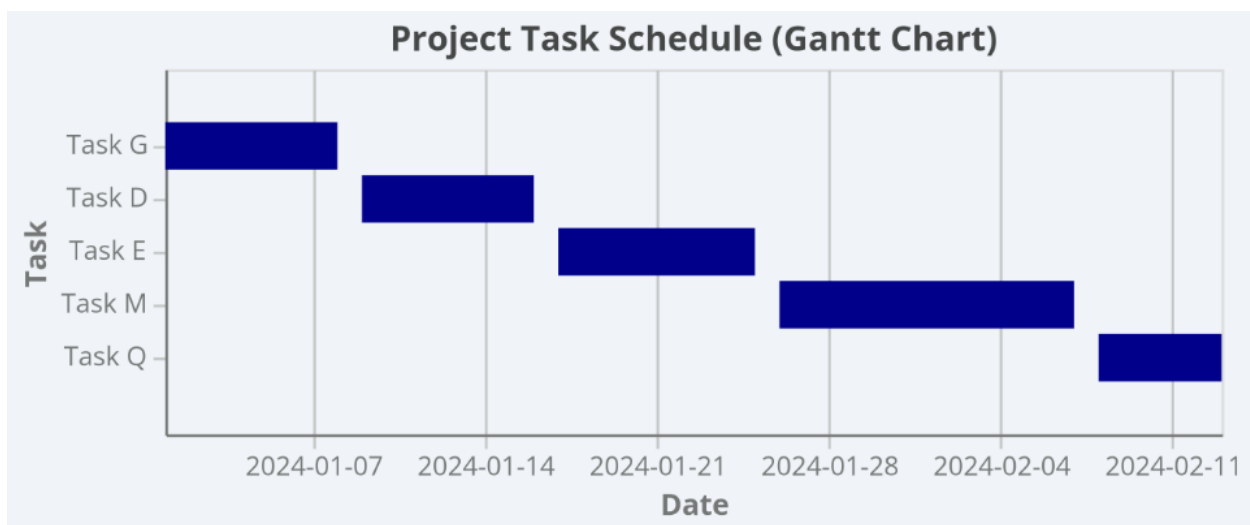
2.5 JBGE stands for "Just Barely Good Enough". It means to only write code and comments that is absolutely necessary

4.2



-
- G → Q, The tasks in the path is G D E M Q
- 32 days

4.4



4.6 When a problem occurs, add more time to the schedule.

4.8 A big mistake is not taking action immediately when a task does not go according to plan. The second mistake is thinking that bringing more manpower will cut the total time of a task.

5.1 A good requirement should be easy to understand, consistent, testable, prioritized, and unambiguous.

5.3 There are requirements for all categories except implementation. This could be because there is no updates that are required.

- 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)
- c) Let the user specify upload/download parameters such as number of retries if there's a problem. (User)
- d) Let the user select an Internet location, a local file, and a time to perform the upload/download. (User)
- e) Let the user schedule uploads/downloads at any time. (Non-Functional)
- f) Allow uploads/downloads to run at any time. (Non-Functional)
- g) Make uploads/downloads transfer at least 8 Mbps. (Non-Functional)
- h) Run uploads/downloads sequentially. Two cannot run at the same time. (Non-Functional)
- i) If an upload/download is scheduled for a time when another is in progress, the new task waits until the other one finishes. (Non-Functional)
- 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)
- m) Display reports of upload/download attempts. (Functional)
- n) Let the user view the log reports on a remote device such as a phone. (User)
- o) Send an e-mail to an administrator if an upload/download fails more than its maximum retry number of times. (Functional)
- p) Send a text message to an administrator if an upload/download fails more than its maximum retry number of times. (Functional)

5.9

- Must: The application needs a way to earn money by either having built-in advertisement or be not free.
- Should: The application could have a win-streak feature and point count instead of the binary win or lose.
- Could: The application could have a random rotation of hangman picture.
- Wont: The application could have different difficulties or word categories.