1.1 -
Requirements Gathering
High-Level Design
Low-Level Design
Development
Testing
Deployment
Maintenance
Wrap-Up

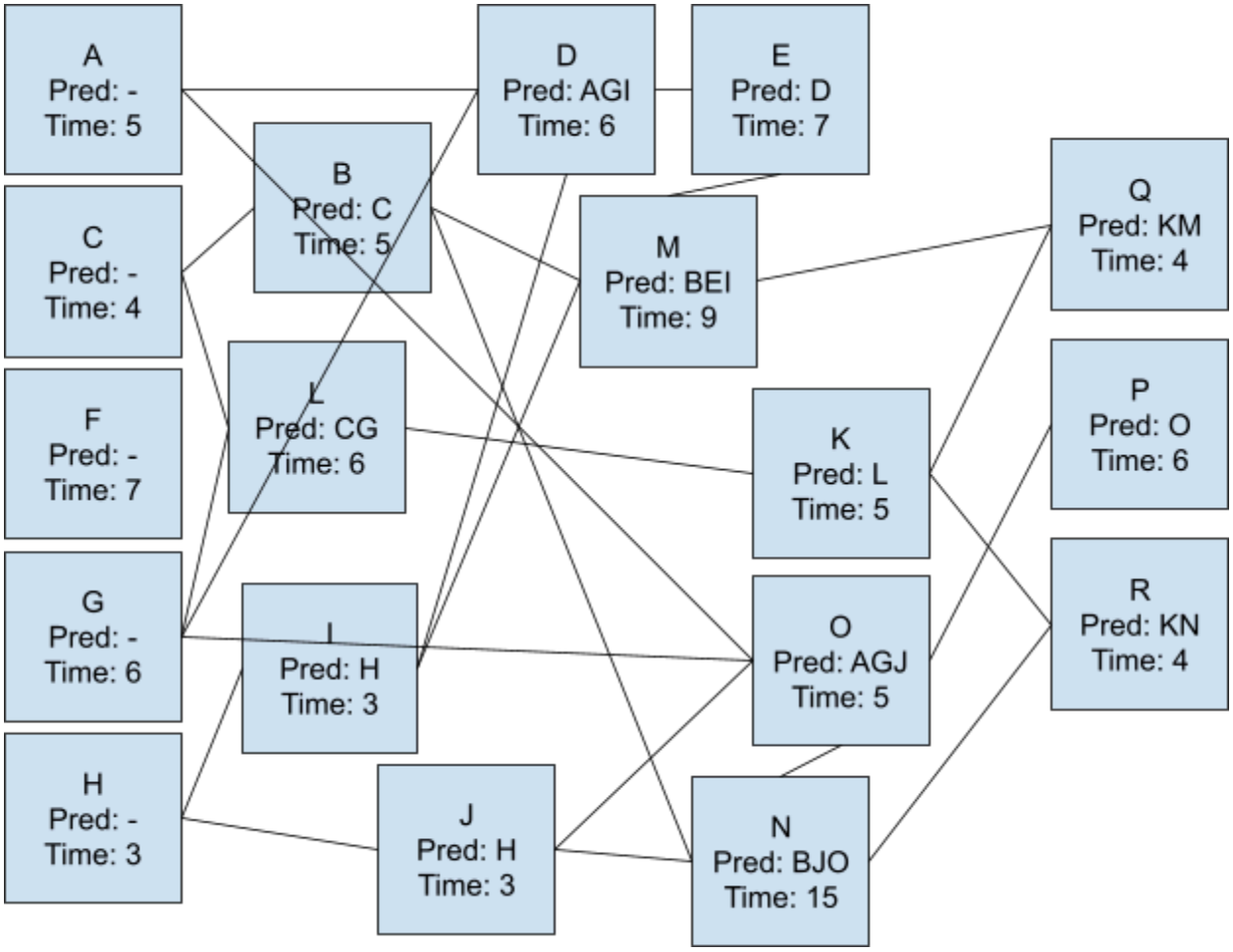
1.2 -
Requirements Gathering: Beginning step where the team finds the customers' wants and needs and supplies a document of the requirements for the product to function.
High-Level Design: The overview step of what will be used at a high level (platforms, databases, etc.) in the product.
Low-Level Design: The more intricate details of the high-level designs which explains how they will work and interact with the product.
Development: The integration step of high and low level design where developers write code and check it as they go to make sure there are no bugs.
Testing: There are multiple steps to this part of the process where developers test first then external testers who did not write code then test to see if they can find any bugs.
Deployment: The step of the process where the product is rolled out for use by consumers, but can at times be a complicated step depending on the scope of the project.
Maintenance: After deployment, maintenance is required to keep the product performing well when users find new bugs or new features are required.
Wrap-Up: This is where the developers complete a postmortem where they go over the developmental process and analyze what went well or what went wrong.

2.4 - Version 1 is a blank page with no writing on it. Version 2 then has the few sentences I wrote on it. In this version, within the version history, all of the changes between version 1 and version 2 are highlighted green. By highlighting the changes green, the user is able to clearly see what has been changed between the versions.

This is a slight difference from how GitHub displays its version history and changes between documents. Like Google Docs, GitHub highlights additions to the document in green. The green highlights show up clearly and show the user what is new and has been added to the project. The difference between GitHub and Google Docs is that GitHub highlights what was removed in red. This is a very intuitive design as the lines highlighted in red have been deleted and the lines highlighted in green have been added. This color coding helps the user easily make sense of the version history of the project.

2.5 - It stands for "Just Barely Good Enough" and it follows the idea that documentation should not be so overly detailed that it becomes cumbersome to reevaluate it between changes.

4.2 -



Critical Path - GDEMQ
Total Duration - 32

4.4 - On Gantt Chart Google Sheets File

4.6 - There are multiple ways to handle these problems that include overestimating a task's time to complete, explicitly show preapproved time off, or implement sick days into the timeline. By overestimating a task's timeline, this gives people time off if they need it but a downside is that people may take advantage of the extra time. When people request time off in advance, it can be displayed on the calendar and can then be implemented into the task's time to be completed. Lastly, by having a number of sick days as an option to be added to the timeline if someone falls ill, it will simply increase the time to complete by however long someone is out sick.

4.8 - The two biggest mistakes a developer can make while tracking tasks and being behind schedule are to avoid the problem and think you can make up the lost time later as well as adding extra developers to a task to recover lost time.

5.1 - Five characteristics of good requirements are: clear, unambiguous, consistent, prioritized, and verifiable

5.3 -

a) Business
b) User / Functional
c) User / Functional
d) User / Functional
e) Nonfunctional
f) Nonfunctional
g) Nonfunctional
h) Nonfunctional
i) Nonfunctional
j) Functional
k) Functional
l) User / Functional
m) User / Functional
n) User / Functional
o) User / Functional
p) User / Functional

Implementation requirements were not categorized because there are no temporary features displayed in the example.

5.9 -

M) It is important that the application can create revenue for the developers. Because of this, Mr. Bones should implement a bar at the bottom of the screen for ad space to monetize the game.

S) Instead of just graying out a letter, remove it from the board entirely. This way the user will have no confusion as to what letters they can pick.

C) Animate Mr. Bones to shake his head when the user loses. While it is not a totally necessary feature for the game to function, it would liven up the game and add a level of stakes for the player.

W) Allow for different app layouts. While this could include different color schemes for the letters and background, a well-designed application will eliminate the need for redesign options.