Sprint 3 Planning Meeting Notes:

Date: 19/4/2018

Members Present: Shan, Young, Jun, Pablo, Pedro, Brian

Photo Evidence SHA:

Timeline:

19/4/2018 to 27/4/2018

Sprint Goal:

Complete Sprint Backlog by 27 April, 2018

Roles:

Scrum Master: Shan Haupt

Product Owner: William G.J. Halfond

Team Members:

Jun Yen Leung

Pablo Chung

Young Kim

Brian Lee

Pedro Leiva

Product Backlog:

|  |  |  |  |
| --- | --- | --- | --- |
| **Task ID** | **Feature** | **Estimation** | **Priority** |
| 3 | When the Build Collage button is pressed, a collage shall be generated that fills the shape of the letters specified in the shape input with the photos that correspond to the topic input |  | 1 |
| 6 | Web application with backend written in Java |  | 2 |
| 4 | Collage must be made up of at least 30 images |  | 3 |
| 1 | Main web page has a topic input box, collage shape input box, collage options, save to history button, and build collage button. |  | 4 |
| 2 | When a collage is displayed there should be an export button that allows for download of the collage |  | 5 |
| 7 | There should be a collage history gallery to display the previously saved collages |  | 6 |
| 5 | Access to the application should be controlled by requiring a username and password |  | 7 |
| 8 | When the Build Collage button is pressed an animated busy/thinking symbol should be displayed until the collage is generated and returned |  | 8 |
|  |  |  |  |

Considerations in prioritizing tasks:

* Priority is assigned with consideration to what the “primary focus” of the app is (creating a collage) and its requirements versus what the “secondary features” are (e.g. “bells and whistles” like having a loading animation, having login functionality)
* Priority is also assigned with consideration to what features might depend on each other. It makes no sense to prioritize a feature that doesn’t work until a lower-prioritized feature is implemented.

Sprint Backlog:

|  |  |  |  |
| --- | --- | --- | --- |
| **Product Backlog Item** | **Item ID** | **Task Description** | **Task Time Estimation (Man Hours)** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Considerations in prioritizing / assigning tasks:

* Highest priority is assigned to BBOX tests, as these *must be completed before their respective implementations*. As such, the simplest approach was to get them out of the way early.
* Second highest priority is assigned to difficult implementations that will likely require large changes to the existing codebase. Most critically, this includes the “persistent history” and “deleting collages from history” requirements.
* Another consideration was dependencies – because we want the code to always work, we avoided implementing features that depended on yet unimplemented features.
* Task delegation was primarily based on the collective code responsibility requirement – we gave tasks to people based on whether they were allowed to work on the relevant code or not. Beyond this, implementation tasks and WBOX testing tasks were mostly given to Jun and Pablo while BBOX testing tasks were mostly given to Pedro and Brian.