Product Backlogs

Requirements Analysis

1. Identify the stakeholders
2. Elicit the requirements
3. Specify the requirements (functional and non-functional)
4. Prioritize the requirements
5. Documenting SRS

Software Design

1. Making design decision
2. Develop initial use case
3. Develop a high-level architectural design
4. Develop a low-level architectural design
5. Documenting SDD
6. Cross-checking SDD against SRS
7. Schedule the Sprints
8. Assigning the user stories of the Sprint (features)

Implementation:

1. Implementing the Core Design Standard Template (Sprint #1)
2. Implementing the Log In by utilizing single sign-on (SSO) (Sprint #1)
3. Implementing the User Profile Module (Sprint #1)
4. Implementing the Matching Engine (Sprint #2)
5. Implementing the Recommendation Module (Sprint #2)
6. Implementing the Social Media Module (Sprint #3)
7. Implementing the Communication Module (Sprint #3)

Testing:

1. Specify the tools and test environment
2. Write Test Cases for Automation and Manual Testing
3. Prioritize the Test Cases
4. Specify the Expected and Actual Output
5. Write Test Script for Functional and Non-Functional Testing
6. Conduct User Acceptance Testing
7. Validate the system against SDD and SRS

Rationale: After discussing among the team members, we understand the product backlogs. We later defined possible product backlogs by following the software development life cycle followed by prioritizing the backlogs. We attempted to find whether the prioritization would follow the SDLC process, or it can be prioritized differently! After discussing further, we decided to prioritize based on the overall process that initiates with requirements analysis followed by design, development, and testing respectively. In each phase, we categorized items based on priority groups: must have, should have, and could have that reflect high, medium, and low priority respectively.

We added the backlog items of the development phase into a sprint in order to allow the developers to implement efficiently by following the process. In each sprint, we arranged backlog items initially for different sprints based on the MVP and the principle of web-application implementation. For instance, a core design standard template is crucial for a system and thus, we prioritize it on the top of the development phase while the communication module at the end of the sprint since this module is a plus, not a requirement.

Note: Our user stories are documented based on the prioritization that we performed in this product backlog.