PROJECT PLAN

MEet and You

Brent Nishioka (Leader)
Gideon Essel
Joshua Ramos
Raymond Guevara
Vivian Dinh
Team Pentaskilled
October 21st, 2021

Table of Contents

Risk Management

Project Level Risks

1. Project grows beyond original expectations
    a. Mitigation: In order to minimize scope creep, team members must thoroughly understand the project requirements and make sure these requirements align with the client. Team members must also create a detailed project plan and uphold the plan during the duration of the project. Discuss how any change would affect the project plan and create a plan on how to tackle any changes to the scope of the project.

2. Misunderstanding of project requirements and vision between client and developers
    a. Mitigation: Get feedback from the client to gain clear definitions of the project requirements and vision. Getting constant feedback from the client improves project clarity and ensures progress is made in the right direction.

3. Product does not meet client expectation
    a. Mitigation: Involve the client throughout every step of development and make sure to fully question the client on their expectations and requirements to ensure a mutual understanding between developers and the client.

4. Not sticking to/lack of understanding of agile principles
    a. Mitigation: Educate team members on agile principles and make sure everyone is on the same page with the principles of an agile software development process.

PROJECT PLAN

Iteration/Sprint level Risks

1. Spending more than allocated budget
    a. Mitigation: Once all requirements are thoroughly discussed, the team recalculates the cost of completing the requirements to make the cost as accurate as possible. The team negotiates with the client to find the best solution that fits the client's needs and requirements.

2. Team members are unable to complete a task or work item due to time, class constraints, or unforeseen personal circumstances
    a. Mitigation: To accommodate for a team member's inability to complete a work item, the team needs to reallocate tasks to other capable team members to ensure the work gets completed.

3. Inability to accommodate changing requirements in the project
    a. Mitigation: Involve the client for feedback during each stage of the development process. Design a product backlog that can accommodate for changes or client feedback.

4. Developers become blocked and unable to continue on a work item
    a. Mitigation: The Scrum master will aid in finding applicable resources to alleviate the obstacle. Assigning additional team members may be necessary to make progress on the work item.

5. Features grow beyond the expected scope due to increasing complexity
    a. Mitigation: Team members must discuss the complexity of each feature in great detail before committing. In order to combat increasing complexity, lessen specific functionalities of a feature until it is within scope. Reprioritize by removing features as a last resort if exclusions are approved by the client.

6. Poor code quality to achieve sprint goals
    a. Mitigation: If the task is achievable but the code is poor, then ask the team for help on how to refine the code. If the task is NOT achievable, being straightforward and asking the team for help is still advised. If possible, do not take on tasks that are unachievable within the sprint timeline.

7. Setting unrealistic timelines/deadlines for work items
    a. Mitigation: Thoroughly discuss work item details and requirements in order to fully understand the amount of time required for the timelines/deadlines. Outline these details to make them as granular as possible in order to set realistic timelines.

8. Falling behind expected schedule for tasks or work items
    a. Mitigation: Update and refine the sprint backlog daily to see if the team is falling behind schedule as soon as possible. Adjust the plan for the sprint accordingly to identify if the team is ahead of schedule or behind schedule. Increase volatility in order to get more work accomplished to meet the expected schedule.