Alan Chan 010063326

Dan Lu 011028589

Yanjun Zang 012419394

**Lab: Software Development Process and Project Metrics**

Our startup team consist of three people, Alan, Dan and Yanjun. The product of our startup is a mobile application that will help user locate the cheapest restaurants around them given the type of food input. We are funded with 2 million USD on the 2nd round of funding for this project.

**Role**:

Alan Chan - Product manager and Developer

* Design and build the necessary continuous integration infrastructure to develop our core product
* Make product backlog, evaluate the scope and importance of user story
* Estimate velocity of team, evaluate risk and develop mitigation plans
* Manage process, remove blocks
* Ensure the product process is on track
* Work with scrum master to re-adjust priorities of the backlog
* Setup the process metrics to evaluate the team’s progress
* Continue to improve the metrics/process by re-evaluating the feedback from previous metrics
* Core product developer

Dan Lu – Scrum Master and Developer

* Design the overall architectural design of our software implementation
* Make product backlog, evaluate the scope and importance of user story
* Estimate story priorities, choose story can be accomplished of each sprint
* Set sprint goal and create sprint backlog
* Work with product manager to ensure the priority to set correctly to meet the the bi-weekly goal
* Core product developer

Yanjun Zang - Developer and QA

* Design test plan to ensure our product will be deliver in the best quality
* Make product backlog, evaluate the scope and importance of user story
* Estimate story priorities, choose story can be accomplished of each sprint
* Set sprint goal and create sprint backlog
* Core product developer

**Project Metrics**:

For every quarter, there will have to be features/bugs that we target to finish. With the group size of three developer, it is important for us to not overestimate our abilities on closing bugs and adding new features. The board would lose trust on the team quickly if we miss deadline on features because we take on more than we can chew.

If a feature/bug is estimated to take longer than a quarter, it might be wise for us to break up the feature/bug to into many subtasks. That way we have a better measurement of successes and failures; it is also easier to divide up the work that needs to be done.

To view the result of each quarter’s successes, we can graph the tasks that we have set out to finish vs the tasks we are able to finish in a quarter. If the ratio is close to one, then we have set a good expectation for the quarter, but if we are way off far away from one, then we have underestimated the tasks’ difficulty or overestimated our actual abilities to solve these tasks, either way it does not look good to the stakeholders.

**Process Metrics**:

To measure the quality of our work, we can look at it in different parts.

* We can separate our bugs into each feature of the product, that we can look at the distribution of the bug spread. If they are all low, then the software is probably in good shape, but if they are all relatively high, we should really go back to the design board and figure out a way to produce better quality code. If the bugs are skewed to specific feature, we should also more resource into that feature to make it better.
* The maintainability of our software will determine how much time we need to go back to our old code and “fix” it as our software evolves. Good maintainability will require less frequency and less time for our developers to re-visit the code. This will also increase our developers’ productivity overall.
* If our software vulnerable to attack, this will create more bugs for our developers to fix, and it is harder to add in the security code, so it is always better to think ahead and design the product with security in mind.
* If our product requires a 100-page manual to use, no users will use it, no matter how great the product is, so making it easy to use is very important.