AGILE DEVELOPMENT CHEAT SHEET

ROLES

Scrum Master Schedules and leads the sprint meetings. Removes blockers. Keeps the team productive.

Product Owner Represents the voice of the customer and the business. In charge of story prioritization.

Software Engineer Builds new product features. Also in charge of writing unit tests.

Quality Engineer (QE) In charge of quality assurance. Programmatically and manually tests new features. Files bugs when features are broken or not built to specifications.

TESTING

Test Driven Development (TDD) A method of testing software in which the tests are written before the software is built.

Continuous Integration Software development practice in which team members integrate their work frequently (i.e. check in their code). This ensures that issues are caught early by automated testing or by other team members.

Unit Test Tests small pieces of code, such as functions and methods, typically with mock data. For example, testing the result of a function given a certain set of parameters. This should be the backbone of your testing strategy.

Functional Test Tests to make sure features work end to end. For example, testing that a user's settings were updated upon clicking a save button.

Integration Test A subset of Functional Tests. Tests to make sure that features or components are compatible with each other.

Regression Test A subset of Functional and Unit Tests. Ensures that new features do not break previously working features.

Smoke Test Tests very broad functionality, such as making sure that the software compiles, or that a service is available.

Manual Test Human testing. Very time consuming and costly. Should be minimized as much as possible.

MEETINGS

Backlog Grooming Occurs once per sprint. Add new user stories to the backlog, reprioritize stories, and break down stories into tasks.

Planning Poker Occurs once per sprint. Software engineers collectively agree on the scope for every potential task coming up in the next sprint. Tasks are typically scoped to a number of hours or days.

Sprint Planning Occurs once per sprint. Determine which stories and tasks will be added to the next sprint. Tasks should already be scoped before attending.

Daily Standup Occurs once per day. Quick status update from each team member. Identify blockers.

Demo Day (Sprint Review) Occurs once at the end of a sprint. Each member demos what they completed to the team.

TASK STATES

Pending Has not been started.

Work In Progress (WIP) Has been started. Several tasks can be WIP at the same time, especially for tasks which are related to each other.

Code Complete and In Review code complete, but it is in review by other team members. If your team does not have a code review process, you can skip this state.

Done Completed. The definition of "done" can vary from team to team, which may depend on factors such as the existence of tests written by the engineer or QE, whether or not the task has been committed to the shared code repository, etc. Whatever your definition of "done" is, make sure that the definition is clear to the entire team.

UNITS OF WORK

Epic Grand product vision. Epics can be broken down into user stories.

Sprint Lasts for 2 - 4 weeks, and is typically made up o
several user stories.

Story Large unit of work that can be completed within o
sprint. Stories are typically scoped based on a Fibonacc
scale (1, 2, 3, 5, 8), where 1 is a relatively small story ar
8 is a relatively large story. An alternative to the Fibonac
scale is "T-shirt" sizing (XS, S, M, L, XL). Can be broker
down into tasks. Typically worked on by one or more tea
members.

Task Small unit of work that can be completed within days or hours. Can be broken down into subtasks. Typically worked on by several team members.

Subtask Very small unit of work that can be completed within a day. Worked on by one team member.

Spike A story or task for answering a question or gathering information.

TOOLS

Task Board A board that organizes the tasks and their states for each team member, including tasks that are Pending, Work In Progress, Code Complete and In
Review, and Done. This can be achieved with a physica
board and sticky notes or with a software tool.

Burndown Chart Line chart trending downwards from
100% to 0% work remaining on the y axis, and from the
sprint start date to the sprint end date on the x axis.
Ideally, sprints should end with the work remaining at 0
If it's not, this may indicate that the team members nee
to improve estimates.

Burnup Chart Opposite of Burndown Chart. Line chart
trending upwards from 0% to 100% work completed on
the y axis, and from the sprint start date to the sprint en
date on the x axis. Ideally, sprints should end with the
work completed at 100%. If it's not, this may indicate tha
the team members need to improve estimates.