Scrum Vs Kanban

Kanban in Japanese means “billboard”. The Kanban method got its name because it utilizes a board of tasks called a kanban board. The board represents items in a backlog and is usually split into three different sections; things that need to be done, things that are in progress, and things that are done. This part in itself isn’t necessarily unique to the kanban method, as SCRUM will often incorporate this aspect as well.

One of the main differences between SCRUM and Kanban, is that SCRUM work is done in fixed duration sprints. Whereas in Kanban, the work is continuous.

Another difference is that in SCRUM, tasks are pulled from the backlog in batches to form a new sprint. Sprint items must be completed first, before going back to the backlog for more tasks. In Kanban, teams deliver features on an as-needed basis and start new tasks only after the previous task is finished.

Generally speaking, in SCRUM, each team member has their own roles and responsibilities and will pull tasks from the sprint based on what matches their role. In Kanban roles and responsibilities are much more loosely defined, if at all, and any member of the team can grab any piece of work from the backlog.

Kanban allows work priority changes at any time in the development process, whereas in SCRUM changes are not permitted during the sprint. Reevaluation can be done before and after sprints.

**Reasons to use SCRUM**

* Stricter rules on what work will be done by who allows less confusion
* Stricter rules on the timeline of when features will be delivered ensures timeliness
* Having a defined set of work every sprint keeps the workload from seeming overwhelming

**Reasons not to use SCRUM**

* Less flexibility on who can work on what piece of work
* Tight deadlines make prioritizing large projects difficult
* Not as easy to pivot to work on something else if needed

**Reasons to use Kanban**

* Anyone can pick up any piece of work
* Less strict deadlines means it’s easier to see large projects through to the end
* Priorities can be easily shifted as needed

**Reasons not to use Kanban**

* Shifting priorities can cause confusion
* No strict deadlines means developers have to be conscious of how much time they’re spending on each item
* Backlog can seem never ending and overwhelming, leading to demoralization of the team.