❖ Kanban model

Kanban is a popular Agile Software Development Methodology. It is basically a signaling device that instructs the moving of parts in a 'pull' production system, developed as part of the TPS (Toyota Production System). Kanban is about envisioning the existing workflow in terms of steps. These steps can be created on the whiteboard.

The main aim of Kanban is to reduce WIP (Work-In-Progress), or inventory, between processes by ensuring the upstream process creates parts as long as its downstream process needs it. The goal of the Kanban execution is to ensure work items move to the next steps quickly to realize business value faster.

The Kanban method is an approach to evolutionary and incremental systems and process change for organizations. A work-in-progress limited pull system is used as the central mechanism to uncover system operation (or process) complications and encourage collaboration to improve the system continuously.

Electronic Kanban boards are also available in ALM tools like Rally (CA Agile), Jira, SwiftKanban, LeanKit Kanban, etc. stages could be configured in these tools, and movement of tickets between stages could be viewed in these tools.

Kanban is best suited in the below scenarios:

- Dynamic/ frequent changing requirements which need to be delivered faster.
- In case of changing priorities, the prioritized work can be pulled by the team as soon as the WIP limit drops.
- Frequent releases are there (Periodically).
- When incoming work is continuous.
- Where task priority needs to be decided dynamically based upon task nature and type.
- The best suit is for Ticket or Production support projects.

- Kanban could be used by any function of an organization as well, for instance in Marketing, Sales, and HR.

However, Kanban might not be the right fit for projects where:

- Tasks could remain in the 'wait' state for long.
- Mainly research-oriented takes are there.
- For enhancements where requirements are evolving/ unclear.
- No prior scope is not defined and tasks keep on evolving.
- Too much dependency is there between tasks.
- If all the items across work stages need to be collated, then only deployed.

Kanban Board/Card

It is critical to understand the visualization of workflow stages in the task execution pipeline. Kanban board provides a simple way to understand the process. It can be explained as follows:

Every request received is put on the Kanban board.

A column on the board represents a stage (these stages are termed as the Work stage) during the lifecycle of bugs/ tickets. For instance, the Kanban board can have 4 stages- Received/ Acknowledged, In-progress, UAT & Done.

The received stage could be called a "Backlog" also.

The team could decide the names for the phases based on the terminology used by their respective teams.

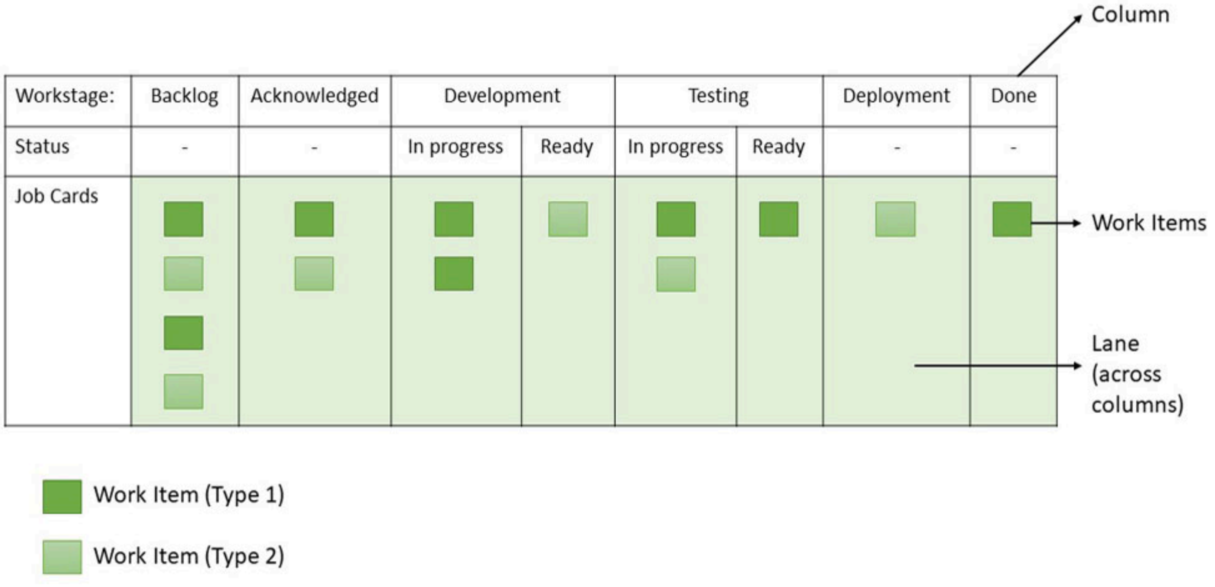
Kanban board could be a simple whiteboard on which sticky notes could be used with ticket details or an electronic Kanban board could be used.

ALM tools like Rally/ Jira could be configured to use the Kanban board.

The board can give a signal in case the bugs/ tickets are stuck in one stage for a long.

For electronic boards, one can configure the Kanban board in a way that tickets/ user stories along with the time stamp are visible.

For whiteboards that are maintained manually, the team can enter the date/ time.



Workstage:	Backlog	Acknowledged	Development		Testing		Deployment	Done
Status	-	-	In progress	Ready	In progress	Ready	-	-
Job Cards	■ □ ■ □	■ □	■ ■	□	■ □	■	□	■

From the sample board above, these can be inferred about Kanban Boards or Cards:

- Workflow: Backlog -> Acknowledged -> Development -> Testing -> Deployment/UAT -> Done
- Work stages: Acknowledged, Development (In Progress), Development (Ready), Testing (In Progress), Testing (Ready), Deployment/UAT, Backlog (Arrival Queue), Done (Finished Stage)
- Work Items: Two types of tasks (Type 1 and Type 2) are represented by work items

Principles of Kanban

Kanban is based on four key principles which are mentioned below:

- Start with the existing process: It is a change management method that starts with the existing process. Changes are done in the system in incremental and evolutionary ways. Unlike Scrum, there's no specific process or roles defined in Kanban.

- Agree to continue evolutionary and incremental changes: After starting with the existing process, the team must agree on continuous, incremental, and evolutionary changes. The changes should be small and incremental. Rapid and substantial changes may be effective but they will be subjected to larger resistance as well by the Team.

- Admire current roles, processes, responsibilities & titles: Though Kanban suggests continuous incremental changes in the process, it respects current roles, responsibilities, and job titles. This helps the team to gain confidence as they get started with Kanban.

- Leadership at all levels: Kanban does not expect leadership from a specific set, rather the actions of leadership at all levels in the organization are very much encouraged.

Kanban Practices

The following are the six core Kanban practices:

- ○ Limit WIP: Limiting Work-In-Process (WIP) implies that a pull system is executed on either parts or the whole workflow. It (PULL system) will act as one of the key stimuli for incremental, continuous, and evolutionary changes to the system. Limit WIP assigns explicit limits to the number of items that may be in progress at each workflow state.

- ○ Visualize: Visualizing the workflow and making it visible is important so as to know how work proceeds. Without understanding the flow of work, incorporating the right changes is difficult. Usually, a card wall with columns and cards is used to visualize the flow of work. Different states or steps within the workflow are represented by the columns on the card wall.

- ○ Manage flow: Flow of work through every state within the workflow should be observed, measured, and informed. By managing the flow vigorously, the incremental, continuous, and evolutionary modifications to the system can be assessed to have negative or positive effects on the system.

- ○ Improve Collaboratively, Evolve Experimentally: Kanban encourages small incremental, continuous, and evolutionary changes. Whenever teams have a common understanding of concepts about work, process, workflow, and risk, they are more likely to be able to form a shared understanding of a problem and suggest enhancement actions that could achieve a consensus.

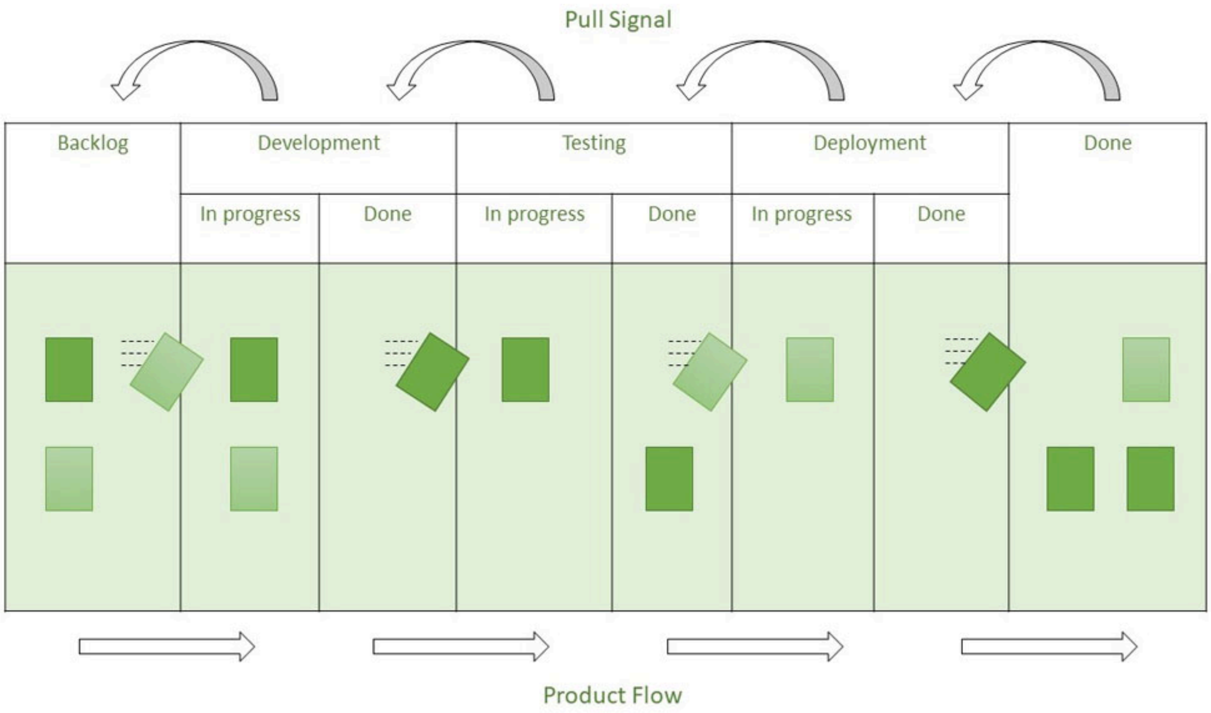
- Implement Feedback Loops: Early feedback from clients and the pull system are important in Kanban. If we get feedback from different stakeholders and processes, it will help to eliminate risk and optimize the delivery process.
- Make Policies Explicit: Until the mechanism of a process is not made clear, it is difficult to hold a debate and discuss ways to improve it. Without a clear understanding of how work is truly done and how things actually work, any conversation of complications tends to be anecdotal, emotional, and subjective. With a clear understanding, it is possible to hold a more rational, empirical, objective discussion of issues. It is more likely to facilitate consensus around improvement suggestions.

Kanban Workflow

1. Visualize your Workflow:

- Identify work stages and the work items.
- Work items: The max effort of 2 days to keep it short and moving.
    - Write work items on cards and stick them in columns, under corresponding work stages, based on system workflow – From left to right.
    - Workflow can be depicted physically or with a tool like Jira.
- Sample Work stages: Design, Development, Test, Production, Deployment, Done.
    - Each Stage can be split into- Ready and In Progress.
    - Initially, these two sub-stages may not be present.
    - Later they are introduced to check the waiting time in each stage.

2. Establish a Pull System:



Rather than pushing tasks into the process, teams pull work only when there is a need for it and they have the capacity to meet it. 'In progress (IP)' and 'Done' columns appear in every process state.

For example, when a development task is finished, it is moved to the 'Development (Done)' column.

Queue statuses are what 'Done' columns are referred to be. No one works on tasks in the queue states; hence these are passive.

In our example, 'Development (Done)', is the queue from which testing teams pull tasks ready for testing once they have the capacity to handle new work.

Teams can avoid multitasking by pulling work and focusing on the most important tasks.

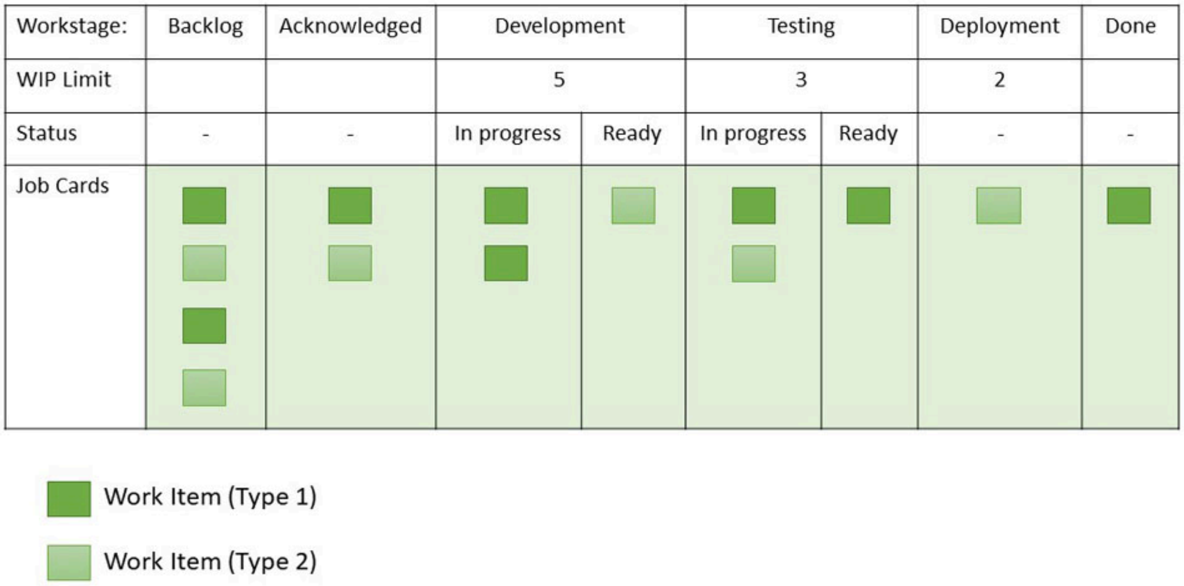
This results in higher throughput and shorter cycle times, which implies happy customers and higher profits.

3. Limiting WIP

Create a Pull system upstream by setting WIP limits, and also ensure that the team "Stops beginning and Starts Finishing." After finishing a task in the current stage, team members can pull a task from the previous stage, thereby freeing up capacity in the preceding stage. This keeps going till the input of the bugs/ tickets.

- WIP limits could be set based on historical data and capacity planning.
- In case many tickets get piled up in one stage other team members help their team members so that tickets movement is smooth, thereby increasing collaboration.
- WIP limits defined could be observed for 3 to 4 weeks and updated based on the team's experience.

The sample Kanban board with WIP limits introduced is given below:



Workstage:	Backlog	Acknowledged	Development		Testing		Deployment	Done
WIP Limit			5		3		2	
Status	-	-	In progress	Ready	In progress	Ready	-	-
Job Cards	🟩 ▨ 🟩 ▨	🟩 ▨	🟩 🟩	▨	🟩 ▨	🟩	▨	🟩

🟩 Work Item (Type 1)

▨ Work Item (Type 2)

Let us understand setting up a WIP limit for a phase in Bug-fix lifecycle with an example:

Project ABC is a maintenance project with a Bug fix in scope. This project needs to set up a Kanban board with WIP limits for the flow of Bug fixes. Let us try to understand how WIP limits are set for the Impact Analysis phase.

History data says the team is spending an average of 4 hours on impact analysis.

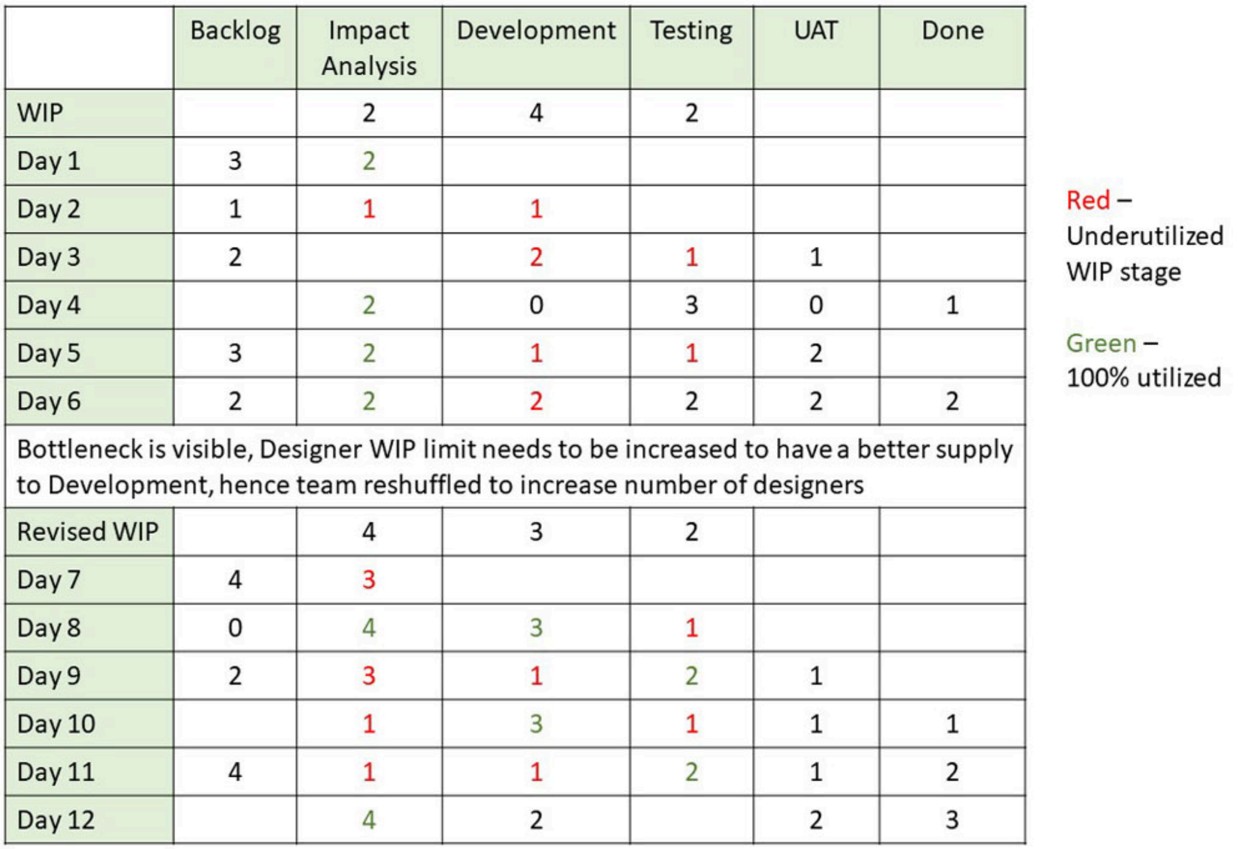
There are 2 designers working in the Impact analysis phase.

They are allocated to this project for 5 hours a day.

The capacity for this phase is 10 hours (5*2: No of hours * No of resources).

Hence WIP limit should be 2 for the Impact Analysis phase.

Similarly compute WIP limits for Development, Testing phases. As a preferred rule WIP limit should not be more than the number of people working on the stage.



	Backlog	Impact Analysis	Development	Testing	UAT	Done
WIP		2	4	2		
Day 1	3	2				
Day 2	1	1	1			
Day 3	2		2	1	1	
Day 4		2	0	3	0	1
Day 5	3	2	1	1	2	
Day 6	2	2	2	2	2	2
Bottleneck is visible, Designer WIP limit needs to be increased to have a better supply to Development, hence team reshuffled to increase number of designers						
Revised WIP		4	3	2		
Day 7	4	3				
Day 8	0	4	3	1		
Day 9	2	3	1	2	1	
Day 10		1	3	1	1	1
Day 11	4	1	1	2	1	2
Day 12		4	2		2	3

Red – Underutilized WIP stage

Green – 100% utilized

In case the WIP limit is breached one could record them as well on the board and provide permissible limits for breaches. In case the WIP limit is breached more than the limit set, WIP limits could be re-computed.

Kanban board can include Blocked tickets information also. WIP limit should be computed based on the project's context.

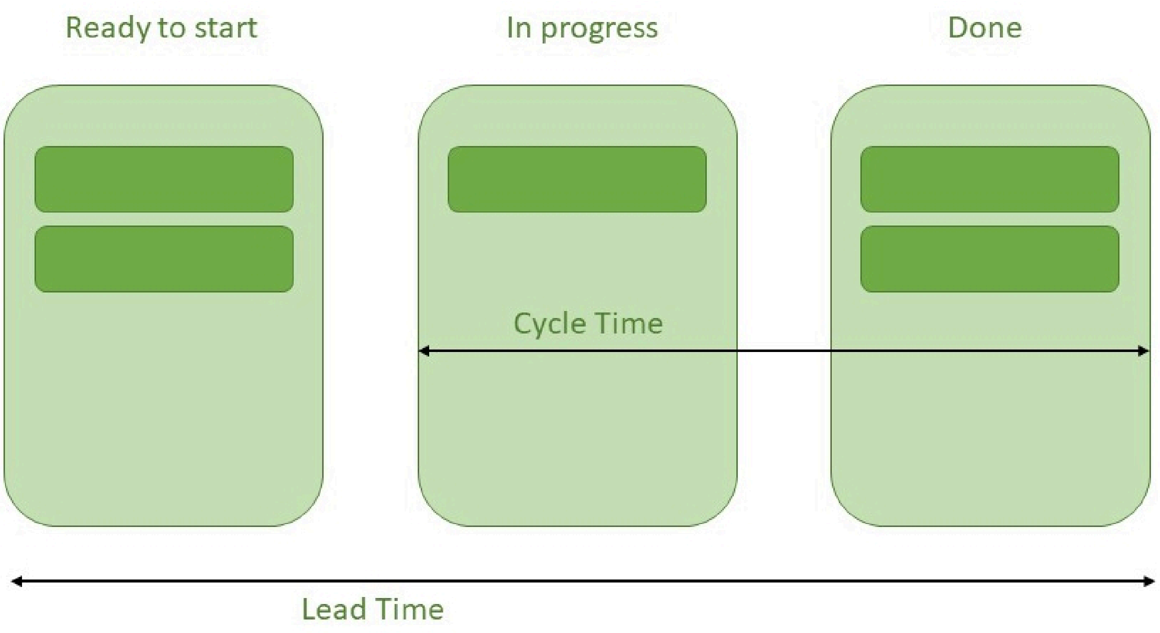
4. Apply Pull Signals

The use of pull signals to indicate that fresh tasks are ready to be handled is an important part of a pull system. When the quantity of cards in a column falls below the specified limit in a Kanban pull system, a pull signal is generated. This tells the previous column that a new job is ready to move forward. No further tasks may be pulled once the work in progress limit has been reached unless an outstanding one has been completed first. This helps to avoid team burnout by ensuring that they only have as much work as they can handle. It also aids in the avoidance of jobs being overlooked.

Lead Time and Cycle Time

Lead Time: The time span between the time a task enters the work system and the time it is completed is known as lead time. Lead time is the time it takes for an input to pass through all of the operations and arrive at the finish line. In Kanban terminology, the overall time it takes for a delegated task to reach the right-most column.

Cycle Time: The cycle time displays how much time the team spends working on a prioritized task. Cycle time begins when any team member begins working on the task and transfers it to the 'in progress' column, and it continues until the task is completed.



Cycle Time and Lead Time Relationship: The most important, though often overlooked, the distinction between cycle time and lead time is their measurement units.

Cycle time is measured in amount of time per unit/process/task

Lead time is measured in elapsed time (weeks, hours, seconds)

Little's Law gives the best description of the link between Cycle time and Lead time-

Cycle time x WIP = Lead time (Work-In-Progress)

Benefits of using Kanban Framework

- Limiting Work in Progress and setting policies will result in a better focus on quality and, as a result, increased customer satisfaction.
- Use transparency to drive process improvement
- Minimized Waste
- Less process overhead

- A more precise and predictable pace guarantees that team members are never overburdened.
- Allows for quick reprioritization in order to adapt changes according to market demand.
- Better task flow
- The composition of the team can also be changed.
- Helps achieve improved productivity of teams
- Prioritization helps in streamlining processes and workflow.
- Enhanced quality of work
- Identification and elimination of bottlenecks
- Reduced queue time
- Reduction of wasted effort