Agile-

Agile software development refers to software development methodologies centered round the idea of iterative development, where requirements and solutions evolve through collaboration between self-organizing cross-functional teams. The ultimate value in agile development is that it enables teams to deliver value faster, with greater quality and predictability, and greater aptitude to respond to change. Plan-driven development A plan-driven approach to software engineering is based around separate development stages with the outputs to be produced at each of these stages planned in advance. Not necessarily waterfall model: plan-driven, incremental development is possible. Iteration occurs within activities. Agile development Specification, design, implementation and testing are inter-leaved and the outputs from the development process are decided through a process of negotiation during the software development process. Most projects include elements of plan-driven and agile processes. Deciding on the balance depends on many technical, human, and organizational issues.

Agile Manifesto

Better ways of developing software by doing it and helping others do it.

Agile Says value to:

Individuals and interactions over processes and tools × Working software over comprehensive documentation × Customer collaboration over contract negotiation × Responding to change over following a plan

Agile Methodology:

Types × Scrum eXtreme Programming (XP) × Feature Driven Development (FDD) × Dynamic Systems Development Method (DSDM)

SCRUM

Scrum is a subset of Agile. It is a lightweight process framework for agile development, and the most widely-used one.

A “process framework” is a particular set of practices that must be followed in order for a process to be consistent with the framework.

Scrum process framework requires the use of development cycles called Sprints

“Lightweight” means that the overhead of the process is kept as small as possible, to maximize the amount of productive time available for getting useful work done.

Essential components of Agile Project

1. An agile project plan is divided into releases and sprints

   Agile planners define a release, which involves creating a new product or substantially updating an existing product. Each release is broken down into several iterations, also called sprints. Each sprint has a fixed length, typically 1-2 weeks, and the team has a predefined list of work items to work through in each sprint. The work items are called user stories. The release plan is broken down into several iterations (sprints) that include user stories (items).

2. Planning is based on user stories A user story briefly describes a need experienced by your users. For example: • "As a team member, I need to know which tasks are currently assigned to me" • "As a team leader, I need to receive email notification when a task is stuck or behind schedule" Unlike in traditional project management methodologies like waterfall, in which teams would create detailed technical specifications of exactly what they would build, in agile planning, the team only documents what the user needs. Throughout the sprint, the team figures out together how to address that specific need in the best way possible

3. Planning is iterative and incremental
   The agile process is focused on the concept of iteration. All sprints are of equal length, and an agile team repeats the same process over and over again in every sprint. Each sprint should result in working features that can be rolled out to end users. An iterative process allows the team to learn what they are capable of estimate how many stories they can complete in a given timeframe (the team's velocity) and learn about problems that impede their progress. These problems can be taken care of in subsequent sprints.
4. Estimation is done by team members themselves A core ethic of agile planning is that development teams should participate in planning and estimation, and not have the work scope "dictated" to them by management. In this spirit, agile planning allows teams to assign story points to user stories in the release plan.

What is JIRA?

JIRA is a tool developed by Australian Company Atlassian. This software is used for bug tracking, issue tracking, and project management. The name "JIRA" is actually inherited from the Japanese word "Gojira" which means "Godzilla". The basic use of this tool is to track issue and bugs related to your software It is also used for project management. The JIRA dashboard consists of many useful functions and features which make handling of issues easy. Some of the key features are listed below. Important Points to Note The following points explain some interesting details of JIRA. JIRA is an incident management tool.

• JIRA is developed by Atlassian Inc., an Australian Company.

• JIRA is a platform independent tool; it can be used with any OS. • JIRA is multi-lingual tool – English, French, German, Japanese, Spanish, etc.

• JIRA supports MySQL, Oracle, PostgreSQL and SQL server in the backend.

• JIRA can be integrated with many other tools – Subversion, GIT, Clearcase, Team Foundation Software, Mercury, Concurrent Version System and many more. Following are some of the most

significant uses of JIRA:

• JIRA is used in Bugs, Issues and Change Request Tracking.

• JIRA can be used in Helpdesk, Support and Customer Services to create tickets and

• track the resolution and status of the created tickets. JIRA is useful in Project Management, Task Tracking and Requirement Management.

• JIRA is very useful in Workflow and Process management

JIRA – Workflow

In JIRA, workflow is used to track the lifecycle of an Issue. Workflow is a record of statuses and transitions of an issue during its lifecycle. A status represents the stage of an issue at a particular point. An issue can be in only one status at a given point of time like Opened, To Do, Done, Closed, Assigned, etc. A transition is a link between two statuses when an issue moves from one status to another. For an issue to move between two statuses, a transition must exist. In a simple way, a transition is some kind of work done on the issue, while status is the impact of work on that issue. Example As of now, an issue is created and opened. When the assignee starts working on the issue, the issue moves to the In Progress status. Here, the transition is starting the work, while the status of the issue is now progressive. JIRA workflow has the following stages to track as soon as an issue is created:

Open Issue: After creation, the issue is open and can be assigned to the assignee to

• Start working on it. In Progress Issue: The assignee has actively started to work on the issue.

• Resolved Issue: All sub-tasks and works of that Issue are completed. Now, the issue is waiting to be verified by the reporter. If verification is successful, it will be closed or reopened, if any further changes are required.

• Reopened Issue: This issue was resolved previously, but the resolution was either incorrect or missed a few things or some modifications are required. From Reopened stage, issues are marked either as assigned or resolved. •Close Issue: The issue is considered as finished; resolution is correct as of now. Closed issues can be re-opened later based on the requirement.