JIRA

Agile

Agile methodology emphasizes **iterative**, **flexibility**, **collaboration**, **customer satisfaction**, **improved product quality**, **faster delivery**, **team collaboration**, **adaptability**, and **continuous improvement**.

Scrum

Scrum is a framework within Agile methodology that emphasizes **iterative development**, **self-organizing teams**, **daily meetings** (stand-ups), **sprint planning**, **reviews**, **retrospectives**, and **backlog grooming**. Its benefits include **transparency**, **adaptive planning**, **faster delivery**, and **continuous improvement**.

In Agile, the **Project Owner** (or Product Owner) is responsible for **prioritizing** and **managing** the **product backlog**, **communicating** project **vision**, and **ensuring** the **team** delivers **value** to **customers**.

In Scrum, the **Scrum Master** serves as the **facilitator** and **servant-leader** for the **Scrum team**, ensuring adherence to **Scrum principles** and **practices**, **removing obstacles**, **fostering** a **collaborative environment**, and **supporting** the team in **achieving** their **goals**.

In Agile and Scrum, the **Development Team** consists of **self-organizing** individuals who collaborate to deliver **incremental** and **high-quality product increments**. They are responsible for **executing** the **work** required to **achieve** the **objectives** of each **sprint**, ensuring **product functionality** and **customer satisfaction**.

Jira

Jira is a popular **project management** tool used for **Agile** and **Scrum** methodologies. It helps teams **plan**, **track**, and **manage** their **work**, **sprints**, **backlogs**, and **issues** efficiently, promoting **collaboration** and **transparency** throughout the **development** process.

Jira Instances

Jira instances are **individual installations** of the **Jira software**, each serving as a **dedicated environment** for managing **projects**, **tasks**, and **workflows**. Organizations often maintain **multiple Jira instances** to cater to different **teams**, **departments**, or **projects**.

Key

In Jira, the key used for an **Epic** is typically a unique identifier associated with each Epic within a project.

Roadmap

It can represent a significant phase, milestone, feature set, or a specific initiative that the team plans to deliver within a certain timeframe.

include **epics** (major goals), **features** (functional requirements), **sprints** (iterative cycles), **milestones** (significant achievements), **dependencies** (task relationships), and **timelines** (progress schedules), crucial for structured project planning and execution.

Epic

an **epic** represents a **large-scale objective** or **milestone** that is **divided** into **smaller tasks** or **features**, guiding strategic planning and progress tracking within the project roadmap.

Timeline

A **timeline** in project management is a **visual representation** of **scheduled events**, **milestones**, or **deliverables** plotted against **time**, aiding in **tracking progress**, **aligning goals**, and **communicating** project **status** and **milestones** effectively.

Sprint

A **sprint** in Agile project management is a **time-boxed** iteration typically lasting **one to four weeks** where a **development team** works to complete a set of **prioritized tasks** or **user stories**.

Story

In Agile project management, a **user story** is a concise description of a feature or requirement from the perspective of the end-user. It typically follows the format: "As a [role], I want [goal] so that [reason]." **Key terms** related to user stories include **acceptance criteria** (conditions to meet), **estimation** (effort assessment), **priority** (importance level), **epic** (larger initiative), and **backlog** (list of pending items), guiding **incremental development** and **customer-centric** product **delivery**.

Task

A **task** in Agile development is a specific unit of work that contributes to completing a user story.

multiple team members collaborate on a user story, whereas tasks are typically handled by one individual.

Subtask

A **subtask** in Agile development is a smaller, detailed task that breaks down a larger task or user story into more manageable parts.

Issue

In Jira, an **issue** is a unit of work or task to be tracked and managed within a project. It can represent tasks, bugs, new features, improvements, or any work item that needs to be addressed.

In Jira, **subtask** refers to a type of issue that is smaller in scope and is created to break down a larger task or user story into more manageable parts. Subtasks are linked to their parent issue and can be assigned to different team members separately from the main task.

Board

In Jira, a **board** is a visual representation of issues from a project or a set of projects. It can be a Scrum board, Kanban board, or a combination of both. Boards help teams visualize work, track progress, and manage tasks through various stages of development or workflow.

Search

In Jira, the **Search** feature allows users to find specific issues or items within projects based on various criteria. Types of searches include basic text searches, advanced searches using Jira Query Language (JQL).

JQL

Jira Query Language (JQL) is a powerful query language used in Jira for searching and filtering issues. It allows users to construct complex queries to find specific sets of issues based on various criteria such as project, issue type, status, assignee, and custom fields.

Labels

labels are keywords or tags that can be added to issues to categorize and organize them based on common themes, topics, or attributes.

Git in jira

To connect Git with Jira, you typically use integration tools or plugins that facilitate communication and synchronization between the two platforms.

Integrating Git with Jira offers several important features that enhance development and project management workflows:

1. **Commit Integration**: Link commits in Git with Jira issues by mentioning issue keys (e.g., PROJ-123) in commit messages. This linkage provides traceability between code changes and specific tasks or issues.
2. **Branch Integration**: View and manage Git branches directly within Jira issues. Track the status of branches related to specific tasks or features, facilitating collaboration and visibility among team members.
3. **Pull Request Integration**: Link pull requests in Git with Jira issues. Review and discuss code changes directly within Jira issues, streamlining code reviews and approvals.
4. **Automated Workflows**: Automate issue transitions based on Git actions. For example, automatically transition an issue to "In Review" when a pull request is created, or close an issue upon merging a pull request.
5. **Deployment Tracking**: Track deployment statuses and related artifacts from Git within Jira issues. Monitor the deployment progress of features or fixes associated with specific Jira tasks.
6. **Version Control Integration**: Integrate Jira versions or releases with Git branches or tags. Associate releases in Jira with corresponding code versions in Git, ensuring alignment between development milestones and project releases.
7. **Development Insights**: Gain insights into development progress and activities through Git-related metrics and reports within Jira. Monitor commit trends, code review statistics, and pull request metrics to assess team productivity and performance.

These features help teams streamline collaboration, maintain visibility across development tasks and code changes, and ensure alignment between project management in Jira and development activities in Git.