# Jira Issue Types

## Standard Issue Types

|  |  |
| --- | --- |
| **Type** | **Description** |
| **Epics** | Epics are large bodies of work. You can think of them as the **big-picture goals** that your team is working towards. Epics can be further broken down into smaller, more manageable units like stories, tasks, and bugs.  Smaller Jira issues like tasks and stories are linked to the epic.  For example, we will create an epic to *launch a new mobile app*. |
| **Stories** | Stories represent **user-centric** **requirements** or **features**. They are written from the user’s perspective and describe their goals. This ensures that development stays focused on delivering value to the end users.  For example, we will create a story to *add a login UI screen in our mobile app*. |
| **Tasks** | Tasks are the **distinct pieces of work**. They are typically smaller and more granular than stories (but note that they're not children of stories), making them easier to assign and track.  For example, a task would be to *integrate the UI with backend APIs*. |
| **Bugs** | Bugs are the unwanted guests of the software development world. They represent **defects or errors** in the system. Identifying and fixing bugs is crucial for ensuring the smooth functioning of your software.  For example, a bug would be *a typo on the login page UI*. |
| **Sub-tasks** | Sub-tasks are **smallest pieces of work** that need to be completed to fulfill a story, task or bug.  For example, a task can be broken down first for *writing logic* and then for *writing unit tests*. |

## Issue Type Hierarchy

|  |  |
| --- | --- |
|  | Let’s break this down with the following hierarchy:   * An epic, being a parent issue, can have child issues such as stories, tasks, and bugs. * However, a story, task or bug can have only subtasks, referred to as child issues. * And, subtasks can’t have any child issue. * But note that stories, tasks and bugs can stand alone without being linked with an epic (not have an epic as its parent). But a subtasks cannot stand alone. * One more thing, when working with Sprint, note that stories, tasks, bugs, subtasks can belong to a Sprint, but epics do not – They're outside of Sprint. |

## Jira Issue Types Best Practices

Using Jira effectively involves adhering to best practices when managing types of issues. Here are some best practices for Jira issue types to ensure efficient project management and team collaboration:

**Define Clear Issue Type Hierarchies**

Jira issue type hierarchy is a structured way to organize issues in a project. By default, Jira hierarchy consists of Epics, Stories/Features, Tasks, Bugs, and Subtasks.

But teams can create Jira custom issue type to better align with their specific workflows and requirements. But avoid creating too many custom types that can lead to confusion.

**Consistent Naming Conventions**

Ensure consistent and clear naming conventions for all issues to make it easy to understand and search for related work. Use descriptive titles and summaries to provide context at a glance. Use tag name (enclosed in a square bracket) to group issues of the same feature.

**Detailed Descriptions and Acceptance Criteria**

Provide detailed descriptions for each issue type in Jira  to avoid ambiguity. Include acceptance criteria for stories to define when a feature or task is considered complete (aka Definition of Done).

**Proper Use of Labels and Components**

Utilize labels and components to categorize issues effectively, making it easier to filter and search for related Jira items. Ensure that labels and components are applied consistently across all Jira issue types.

**Workflow Customization**

Issue workflows define the sequence of steps an issue goes through from creation to completion. Workflows can include statuses like "To Do," "In Progress," "In Review," and "Done." These workflows ensure that issues move through a structured process, maintaining order and visibility. Always ensure transitions between states are logical and enforce required fields where necessary.

**Regular Grooming and Maintenance**

Regularly groom your backlog to prioritize and refine issues, ensuring that high-priority work is well-defined and ready for development. Archive or close outdated issues to keep the backlog manageable.

**Link Related Issues**

Linked issues are Jira issues that are related to each other in some way but do not necessarily have a parent-child relationship. They can be connected through various link types, such as "blocks," "is blocked by," "duplicates," or "relates to."

**Utilize Jira Agile Boards**

Set up [Agile boards](https://planyway.com/blog/jira-board) (Kanban, Scrum) that match your workflow to visualize and manage work in progress. Jira users often use additional specific Agile issue types above the Jira default issue type to implement Agile practices such as sprints, backlogs, and continuous delivery, allowing them to efficiently plan, track, and manage their work:

* **Feature:** A feature issue type is typically used to represent a high-level functionality that will provide significant value to the end users. It's often a part of the hierarchy that sits above user stories and tasks but below epics.
* **Spike:** A spike is an investigation or research task. It's used when there is a need to gather information or explore options before a story can be estimated or worked on.
* **Improvement:** An improvement is an enhancement or update to an existing feature, aiming to make it better or more efficient.
* **Incident:** An incident is an unexpected disruption or issue in the service that needs to be addressed immediately.
* **Change Request:** A change request is a formal proposal for altering a component or system, typically reviewed and approved before implementation.

**Effective Use of Epics**

Keep epics at a manageable size. If an epic becomes too large, consider splitting it into multiple epics. Use epic links to track the progress of related stories and tasks within the epic.

**Regular Reporting and Dashboards**

Create and use Jira dashboards to monitor project progress, issue statuses, and team performance. Use built-in reports or custom queries to track metrics like burndown charts, velocity, and cycle time.