# Introduction to JIRA & Agile Project Management

## Overview of JIRA

- **What is JIRA?**
    - JIRA is a popular project management and issue-tracking software developed by Atlassian.
    - Used for Agile software development (Scrum/Kanban), bug tracking, project planning, and product management.
- **Use Cases:**
    - Agile Sprint Planning
    - Bug/Issue Tracking
    - Requirement & Task Management
    - Reporting and Metrics
    - CI/CD integration (e.g., Jenkins)
- **Architecture:**
    - JIRA Core (Business Projects)
    - JIRA Software (Agile Boards)
    - JIRA Service Management (ITSM)
- **JIRA Ecosystem:**
    - Atlassian Confluence, Bitbucket, Trello, Bamboo

*******************************************************************************

## Agile Concepts Refresher

- Scrum vs Kanban
- Epics, Stories, Tasks, Sub-tasks
- Product Backlog, Sprint Backlog, Sprint
- Agile ceremonies: Sprint Planning, Standups, Review, Retrospective

*******************************************************************************

## JIRA Setup in Corporate Environment

- Cloud vs On-Premise (JIRA Data Center)
- Creating an Atlassian Cloud Account
- Setting up a new project
- User Roles and Permissions
- Adding team members

***********************************************************************************

## JIRA Projects

- Classic vs Next-Gen (Team Managed vs Company Managed)
- Project Types: Scrum, Kanban, Bug Tracking
- Project Settings: Workflows, Screens, Fields, Notifications

***********************************************************************************

## Creating and Managing Issues

- Issue Types: Story, Bug, Task, Sub-task, Epic
- Creating Issues manually
- Bulk Import using CSV
- Linking Issues
- Assigning issues, setting priority and estimates

***********************************************************************************

## Boards and Backlogs

- Creating Scrum/Kanban Boards
- Configuring Columns
- Managing Product and Sprint Backlogs
- Starting and Ending Sprints

***********************************************************************************

## 1. Introduction: Agile, Scrum & Why Jira?

- **Agile** is a philosophy rooted in the **Agile Manifesto**—core values include early delivery, close collaboration, adaptability, and simplicity
- **Scrum** is a lightweight Agile framework built around short iterations called **sprints**, cross-functional teams, and ceremonies like Daily Scrum, Sprint Review, and Retrospective
- **Jira**—developed by Atlassian—is the #1 tool for Agile teams. It supports backlog grooming, sprint planning, boards, reporting, integrations, and much more

## 2. Scrum Roles & Artifacts

**Roles:**

- **Product Owner (PO)**: owns backlog, prioritizes features.
- **Scrum Master (SM)**: facilitates ceremonies, removes impediments.
- **Development Team**: cross-functional and self-organizing

**Artifacts:**

- **Product Backlog**: ordered list of features/user stories.
- **Sprint Backlog**: subset of backlog items selected for a sprint.
- **Sprint Increment**: deliverable output at end of sprint.
- **Burndown Chart**, **Velocity Chart**, **Cumulative Flow Diagram**, **Control Chart** as tools to visualize progress

## 3. Jira Issue Hierarchy & Issue Types

- Hierarchies can include: **Initiatives → Epics → Stories/Tasks → Sub-tasks**. Useful for scaling or SAFe-like frameworks
- **Issue Types** in Jira: *Epic*, *Story*, *Task*, *Bug*, *Sub-task*.
- Use consistent naming and issue description practices to maintain clarity and traceability

## 4. Setting Up Jira for Scrum

- Start by creating a **Scrum project** using Jira's Scrum template. This provides Backlog, Board, Reports, Roadmap view by default .
- Choose between **Team-managed (Next-gen)** or **Company-managed (Classic)** projects based on control and scale needs.
- Configure **Workflows**, **Fields**, **Screens**, and **Notifications** to match your team's process.

- Integrate with tools like **Confluence**, **Bitbucket**, or CI/CD to close the loop between documentation and execution.

## 5. Using Jira with Scrum

**Backlog Management:**

- **Create and refine user stories** in the backlog. Good user story structure: *As a [user], I want [goal], so that [value]* .
- Prioritize stories by **drag-and-drop** in backlog.

**Sprint Planning:**

- Select backlog items into a spring-ready sprint.
- Estimate using **story points** or **time estimates** depending on team preference.
- Define **Sprint Goal** and prepare for Daily Stand-up.

**During Sprint:**

- Team members move issues across board columns: *To Do* → *In Progress* → *Code Review* → *Done*.
- Update **time estimates**, use **labels**, **comments**, and **attachments** to track progress.

**Sprint Closure:**

- Complete the sprint and view **Sprint Report**, **Burndown**, **Velocity**, **Cumulative Flow Diagrams** in Jira's Reports section to inspect progress and adoption .

## 6. Reporting & Analytics in Jira

**Standard Reports:**

| **Report** | **Purpose** |
|---|---|
| **Burndown Chart** | Visualizes remaining effort vs time |
| **Velocity Chart** | Tracks team capacity over multiple sprints |
| **Cumulative Flow Diagram** | Shows flow of issues through statuses |
| **Control Chart** | Analyzes cycle time variability |
| **Sprint Report / Pie Chart Report** | Highlights completed vs open issues, distribution by assignee/status |

## 7. Plugins, Automation & Best Practices

**Plugins to Enhance Reporting:**

- **EazyBI**, **Rich Filters**, **Custom Charts for Jira**, **Structure for Jira**—great for advanced visualizations and multi-level reporting.

**Automation Ideas:**

- Auto-assign high-priority bugs to QA Lead.
- Auto-escalate overdue tasks or send reminders.
- Use Jira's built-in Automation or marketplace add-ons.

**Best Practices:**

- Keep backlog healthy: review, groom, prune regularly.
- Keep issue descriptions clear; use checklists and Definition of Done.
- Use labels, components, and consistent estimations.
- Limit Work-in-Progress (WIP) columns to enforce flow.
- Design permissions and roles deliberately to control visibility and access

## 8. Real-Life Examples & Sample Workflow

1. **Product Backlog Grooming**
    - PO creates epics (e.g., "Checkout Flow") and user stories within. PO orders and clarifies stories.
2. **Sprint Planning**
    - Team selects stories, estimates story points (e.g. 8, 5, 3) and starts sprint.
3. **In-Sprint Tracking**
    - At Daily Stand-ups: team updates board; burndown reflects remaining work.
4. **Sprint Review & Retrospective**
    - Use **Sprint Report** to review completed work, gather feedback, and derive improvements. Add retrospective notes to Confluence, link to sprint retrospective epic.
5. **Reports & Dashboard**
    - Create a dashboard showing: Burn-down, Bugs by sprint, Velocity trend, issues by assignee.

# Why DevOps & Cloud Engineers Need Jira & Scrum?

📌 **Key Use Cases:**

- Sprint planning for **CI/CD pipeline features**
- Tracking infrastructure provisioning with **Terraform/CloudFormation**
- Managing **Incidents, Monitoring, and Alerts**
- Tracking **automation tasks**, **cloud cost optimization**, and **migrations**
- Integrating with tools like **Jenkins, Git, AWS, Azure DevOps, Docker, Kubernetes**

## Overview of Jira

**What is Jira?**

- Atlassian tool for **Agile Project Management**
- Commonly used for **issue tracking**, **sprint planning**, **bug reporting**, and **DevOps work management**

**Jira Ecosystem:**

- Jira Software (Agile)
- Jira Service Management (ITSM)
- Confluence (Documentation)
- Bitbucket (Git Repo)
- Bamboo/Jenkins (CI/CD)

**DevOps Scenarios:**

- Track tasks like “Setup ELB”, “Provision EKS Cluster”, “Write Helm Chart”, “Create Jenkins Pipeline”

✅ Live Demo:

- Create a **Company-managed Jira project** named CloudInfra-DevOps
- Define a Kanban/Scrum board
- Create an Epic: "Setup Production EKS Cluster"

## Agile Concepts Refresher

**Key Concepts:**

| Concept | DevOps Example |
|---|---|
| **Epic** | CI/CD Pipeline Setup |
| **Story** | Setup Jenkins Slave Nodes |
| **Task** | Write Terraform module |
| **Bug** | Nginx config not routing |
| **Sub-task** | Configure SSL in Nginx |

**Scrum Events for DevOps Teams:**

- Sprint Planning: Select cloud automation tasks
- Daily Stand-up: “Any blockers in creating AMI pipeline?”
- Review: “What’s the new backup automation delivered?”
- Retrospective: “Why was the IAM permission fix delayed?”

---

## Jira Setup + Work Management for DevOps & Cloud

🔹 **Jira Project Setup**

**Types of Jira Projects:**

- Scrum (good for product & sprint work)
- Kanban (ideal for continuous InfraOps & IT support)

✅ Live Setup:

- Create a Scrum project: Cloud-Migration-India
- Configure **components** like:
  - CI/CD
  - AWS
  - Monitoring
  - Terraform
  - Kubernetes

**User Roles:**

- Cloud Engineer: Contributor
- DevOps Lead: Admin
- Stakeholders: Viewer

---

## Creating Issues & Tracking Work

**Typical DevOps/Cloud Work Items:**

- Epic: “Cost Optimization on AWS”
  - Story: "Move logs to S3 Glacier"
  - Task: "Setup lifecycle policies"
  - Sub-task: "Create CloudWatch alert on S3 events"

✅ Hands-on:

- Create 3 Epics for your environment:
  - Setup Monitoring Stack
  - Setup CI/CD Pipeline
  - Implement Disaster Recovery
- Add linked stories, estimate in **Story Points (1, 3, 5, 8)**

**Use CSV Bulk Upload** for repeatable tasks like:

- “Install Prometheus Agent on 5 EC2 Instances”
- “Create IAM Roles for Jenkins”

---

## Backlogs & Sprint Tracking

🔹 **Session 5: Boards & Sprint Management**

- **Backlog View** → Prioritize upcoming tasks
- **Active Sprint** → Move tasks from *To Do* → *In Progress* → *Done*
- **Swimlanes**: Group by assignee, Epic, or priority

✅ Demo:

- Create a 1-week sprint named Sprint-Cloud-Infra-001
- Add 5–7 DevOps tasks

- Assign to team, estimate in story points

---

## Reporting & Monitoring Jira Work

🔹 **Session 6: Reporting for DevOps & Cloud Teams**

| **Report** | **DevOps Use** |
|---|---|
| **Sprint Report** | Track delivery of infrastructure setup |
| **Burndown** | Velocity of pipeline automation |
| **Cumulative Flow** | Visualize tasks stuck in "In Progress" |
| **Control Chart** | Time to complete security fixes |

✅ Dashboard Setup:

- Burndown Chart (for Infra Team)
- Pie Chart by Issue Type
- Issues by Cloud Component (AWS, Azure, Monitoring, CI/CD)

---

## Jira Automation for DevOps

Use **Automation Rules**:

- Auto-close "Done" tasks after 3 days
- Notify Slack/Teams when "CI Pipeline Failed" bug is raised
- Change priority of issues with title High CPU Alert

✅ Demo:

- Create rule: When Bug with label prod-incident is created, assign to DevOps-OnCall

---

## Integration with DevOps Tools

**Jira + Jenkins Integration**:

- Jenkins updates Jira issues on successful builds
- Link commits, branches to Jira tickets (GIT-101 in commit message)

**Jira + Confluence**:

- Link retrospectives or runbooks to issues

**Jira + GitHub/Bitbucket**:

- Automatically reference issues in PRs, commits