Devops

Team Structure & Workflow (Amazon Fresh Example)

Phase 1: Requirements Gathering

Business Analyst (BA)

- Interacts directly with customers/stakeholders
- Gathers requirements
- Creates **BRD (Business Requirements Document)**

Phase 2: Prioritization

Product Manager (PM)

- Reviews BRD from BA
- Defines vision, goals, and priorities
- Decides what features to build first

Phase 3: Breaking Down Work

Product Owner (PO)

- Receives prioritized requirements from PM
- Breaks them into **Epics** (large actionable items)
- Examples: "UI redesign needed", "Backend API development", "Payment gateway integration"
- Creates these Epics in **Jira**

Phase 4: Technical Design

Software Architect

- Works with PO to validate technical feasibility
- Creates **HLD (High Level Design)** - Overall system architecture
- Creates **LLD (Low Level Design)** - Detailed component designs
- Defines frameworks and system structure

Phase 5: Development & Testing (Parallel Work)

Team members working together:

1. **Developers** - Write code (UI, APIs, databases)

2. **QA Engineers** - Test quality and performance
3. **DBA** - Design and manage databases
4. **DevOps Engineers** - Build CI/CD pipelines, manage infrastructure (Kubernetes, Docker, Git)

These 4 roles collaborate simultaneously on the same features.

Phase 6: Post-Release

SRE (Site Reliability Engineer)

- Ensures uptime, performance, and reliability
- Creates monitoring metrics and alerts
- Handles incidents

Technical Writers

- Document entire system, APIs, user guides

How DevOps Improves SDLC

Traditional SDLC phases:

1. Planning
2. Analysis
3. Design
4. Implementation
5. Testing & Integration
6. Maintenance

DevOps Engineer's Role:

- Identifies bottlenecks in SDLC
- Creates **automation** to speed up processes
- Integrates **security** (DevSecOps)
- Improves **efficiency** across all phases
- Enables faster delivery

Project Tracking with Jira

How Organizations Track Progress:

Jira - Project Management Tool

Hierarchy in Jira:

Epic (Created by PO)

↓

Stories (Created by Scrum Team from Epic)

↓

Tasks/Subtasks (Assigned to specific developers)

Agile/Scrum Process:

1. **Sprint** - Time-boxed period (usually 2-4 weeks) to complete work
2. **Backlog** - Collection of all Epics/Stories waiting to be worked on
3. **Sprint Planning** - Team selects Stories from backlog for upcoming Sprint
4. **Daily Standups** - Quick team sync on progress
5. **Sprint Retrospective** - Review what went well/poorly after Sprint ends

Example Workflow in Jira:

1. PO creates **Epic**: "Amazon Fresh - Add real-time inventory tracking"
2. Scrum team breaks Epic into **Stories**:
    - "Design inventory UI dashboard"
    - "Build inventory API endpoints"
    - "Set up database for inventory data"
    - "Write automated tests for inventory feature"
3. Stories are added to **Backlog**
4. During Sprint Planning, team moves Stories from Backlog to **Active Sprint**
5. Developers get assigned specific Stories
6. Everyone can see status: **To Do → In Progress → In Review → Done**

Key Takeaway

Visibility & Tracking: Jira provides real-time visibility into:

- Who is working on what
- What stage each requirement is at
- Bottlenecks or delays
- Sprint progress and velocity

This allows the entire organization (from BA to SRE) to stay aligned and move fast—which is the core goal of DevOps!
