CI/CD

(Continuous Integration / Continuous Delivery or Deployment)

What is CI/CD?

- CI/CD stands for Continuous Integration and Continuous Delivery/Deployment.
- It is a DevOps practice that automates the software development lifecycle – from code integration → building → testing → deployment → monitoring.
- Goal: Deliver software faster, safer, and more reliably.

1. Continuous Integration (CI)

- Practice of frequently merging code changes into a shared repository (e.g., GitHub, GitLab).
- Each commit triggers automated builds and tests.
- Goal: Detect bugs early and ensure code is always in a deployable state.

2. Continuous Delivery (CD)

- Extension of CI.
- Ensures that the software can be released at any time.
- Automates deployment up to staging or pre-production environment.
- Final release to production is still a manual decision.

3. Continuous Deployment (CD)

- Goes one step further than Continuous Delivery.
- Every code change that passes tests is automatically deployed to production.
- No manual approval step.

CI/CD in DevOps

What is CI/CD in DevOps?

- CI/CD is a core practice in DevOps that automates the process of building, testing, and deploying software.
- It helps bridge the gap between development (Dev) and operations (Ops) by ensuring faster, reliable, and continuous software delivery.

CI/CD Pipeline in DevOps

1. **Plan** → Define features & requirements.
2. **Code** → Developers write and push code.
3. **Build** → Code is compiled and packaged.
4. **Test** → Automated unit, integration, and regression tests.
5. **Release/Deploy** → Application moved to staging/prod.
6. **Monitor** → Performance, logs, and issues tracked.
7. **Feedback** → Collected to improve next cycle.

CI/CD Environment Flow

1. Development (Dev)

- Developers write and test code locally.
- Unit tests, code quality checks.
- Goal → Ensure code works before moving forward.

2. Quality Assurance (QA / Test)

- Code deployed to QA env.
- Automated + manual testing (integration, regression, functional).
- Goal → Validate correctness of features.

3. User Acceptance Testing (UAT)

- Environment where end-users, clients, or business teams test.
- Validates that software meets business requirements**.**
- Often uses realistic data but not production data.
- Goal → Business sign-off before production.

4. Pre-Production (PPE / Staging)

- Also called Staging or Pre-Prod Environment (PPE)**.**
- Replica of Production (same infra, configs).
- Final system tests:
  - Performance testing
  - Load testing
  - Security checks
- Goal → Catch issues that might appear in Production.

5. Production

- Live environment accessed by real users**.**
- Deployment strategies: Blue-Green, Canary, Rolling.
- Monitoring + logging to ensure stability.

Benefits of CI/CD

- Faster development and release cycles.
- Early bug detection.
- Automated testing reduces manual effort.
- Consistent deployments across environments.
- Improves developer productivity and customer satisfaction.