| **Aspect** | **Continuous Deployment** | **Continuous Delivery** |
| --- | --- | --- |
| **Definition** | Automatically deploys every code change to production, without manual intervention. | Automatically prepares code for production but requires manual approval for deployment. |
| **Automation** | Fully automated deployment process. Once code passes tests, it is automatically deployed. | Automated build, testing, and staging but requires manual approval for release to production. |
| **Deployment Frequency** | Multiple deployments per day, with every code change. | Frequent, but at the discretion of the team, often aligned with release cycles. |
| **Deployment Control** | Minimal human intervention. Code is pushed to production as soon as it passes automated tests. | Requires manual approval or intervention for each production deployment. |
| **Risk Tolerance** | Higher risk as changes go directly to production. Extensive automated testing and monitoring are crucial. | Lower risk as changes are thoroughly tested and validated before being deployed to production. |
| **Rollback Strategy** | Quick rollback is essential, as issues can be introduced in production with every change. | Rollback is usually smoother due to extensive testing, making rollbacks less frequent. |
| **Feedback Loop** | Immediate feedback from real-world users and production environments. | Feedback loop is slower, as changes are not immediately pushed to production. |
| **Use Cases** | Suitable for web applications, SaaS, and scenarios where rapid changes are required. | Appropriate for applications that require a balance between speed and control, such as financial services. |
| **Team Collaboration** | Requires a high level of trust and collaboration among development, testing, and operations teams. | Promotes collaboration but allows for a more controlled and staged approach. |