CICD

- CICD stands for Continuous Integration & Continuous Delivery (Automated release)/Deployment (manual production release)
- Automated steps involved during the CICD pipeline:

CI: Code Build -> Unit Case execution -> Static Code analysis -> Vulnerability Test

CD: Infra Provision -> Infra setup & configure -> Database migration -> Deploying application -> PROD release -> Smoke Testing

Finally, Celebrations

CICD Benefits

Associated Cost Benefits	Associated Delivery Benefits	Technology Benefits
Reduce Cost	Less developer time on issues from new developer code	Catch Compile Errors After Merge
Avoid Cost	Less bugs in production and less time in testing	Catch Unit Test Failures
Avoid Cost	Prevent embarrassing or costly security holes	Detect Security Vulnerabilities
Avoid Cost	Less human error, Faster deployments	Automate Infrastructure Creation
Reduce Cost	Less infrastructure costs from unused resources	Automate Infrastructure Cleanup
Increase Revenue	New value-generating features released more quickly	Faster and More Frequent Production Deployments
Increase Revenue	Less time to market	Deploy to Production Without Manual Checks
Protect Revenue	Reduced downtime from a deploy-related crash or major bug	Automated Smoke Tests
Protect Revenue	Quick undo to return production to working state	Automated Rollback Triggered by Job Failure

Customer satisfaction

- Customers are happy with fast turnaround & roll-out of new features and bug fixes. Product release cycles are shorter with targeted feature releases and this only blocks fewer features that aren’t ready for release.
- Keeps Customer happy as they end up finding zero to none errors in your product.
- Add new requirements based on customer’s needs/feedbacks on a daily basis during continuous development, which leads to usability improvements.
- Seamless & targeted introduction of new production features by toggles and blue-green deploys features
- Less disruptive service when upgrades introduce smaller units of change and identifying many errors in your product before release

Smaller backlog

- Backlog of non-critical defects is lower because defects are often fixed before other feature pressures arise.
  - More time for developers/QAs to focus on the big picture instead of spending their valuable time on fixes
  - More time for QAs to find larger problems before being a product is released.
- Product release velocity is high.
  - High velocity means improved feature release rate, as failures are detected faster and as such, can be repaired faster
  - Release cycles are shorter with targeted releases and this blocks fewer features that aren’t ready for release.

Business Metrics Produced

Metric	Formula	Impact
Lead Time to Production	(*Time at Successful Prod Deployment*) - (*Time at Completion of Feature Grooming*)	Shows how CI/CD is impacting overall delivery time, from the point the team hears about the feature to the point at which it is done (deployed to production). Easy metric to collect if using task management system to track feature grooming and deployments.
Rollback Rate	(*Total Rollbacks*) / (*Total Deployments*)	Shows the quality of our deployments. Of course, rate should be low because previous stages should filter out defected builds. This metric is a leading indicator for the confidence of the business in the dev team's ability to delivery.
Time to Failure	(*Time at Failure Discovery*) - (*Time at Build Start*)	Shows how quickly we find failures. The lower the better.
Production Uptime	(*Total Production Working Time*) / (*Total Time*)	Shows the amount of time we are taking production down because of botched deployments or due to our chosen deployment strategy.
Failed Pipeline Cost	Various calculations including job run time and resources created	Shows the estimated amount of money spent on a failed build. Encourages us to put cheaper jobs earlier in the pipeline.