2-1-1: Examining CI/CD Pipelines

After completing this episode, you should be able to:

- Identify and explain the significance of a CI/CD pipeline in DevOps environments, given a scenario.

Description: In this episode, the learner will examine a CI/CD pipeline and its significance to DevOps environments. We will explore continuous integration (CI), continuous deployment, automation, feedback loops, and more.

- Describe a CI/CD pipeline
    - A series of steps followed by software teams to deliver updates and improvements to software applications automatically and frequently.
    - CI/CD pipeline represents the backbone of modern DevOps practices, promoting a culture of high velocity and high-quality software development and release
- Describe continuous integration or the CI in CI/CD
    - A practice involving automatically integrating code changes from multiple contributors into a single software project several times a day.
- Describe continuous delivery or the CD in CI/CD
    - Extends the concept of CI by automatically deploying all code changes to the testing environment after the build stage.
- Describe what a workflow is within a CI/CD pipeline
    - A series of automated steps that streamline software development from code creation to deployment
    - It starts with code being pushed to a version control system, triggering the pipeline. Automated builds and tests ensure code quality and functionality.
    - Once validated, the code is merged into the main branch and deployed to testing environments for further checks.
    - Finally, the code is deployed to production, where it is continuously monitored to ensure performance and reliability.
    - This process reduces human error, enhances software quality, and speeds up the delivery of updates to users.
- Describe the common steps in a typical CI/CD process
    - CI - Pulling code from a version control system
        - Ensures that the latest code changes are tracked, peer reviewed, approved and integrated
    - CI - Running automated builds and tests to verify the code
        - Helps to catch integration errors early and improve software quality.
    - CI - Merging validated updates back into the main branch
        - Ensures that only tested and stable code is integrated, maintaining the integrity of the main branch..
    - CD - Deploying code changes to the testing and after the build stage.
        - This can ensure that new features and fixes are quickly implemented
    - CD - Deploying code changes to the production environments
        - This can ensure that new features and fixes are quickly available to users.
    - CD - Monitoring and validating post-deployment
        - Ensures deployed changes perform as expected and allows real user feedback, verifying functionality and enabling continuous improvement.
- Describe additional considerations for a CI/CD pipeline
    - Automation
        - At every stage, manual tasks are automated, from code integration to testing, building, and deploying.
        - Reduces the risk of human error, speeds up production cycles, and enables more consistent and reliable outcomes.
    - Feedback loops
        - The pipeline includes monitoring tools and feedback mechanisms that alert developers if there's a failure at any stage
        - This immediate feedback is crucial for quickly addressing problems before they affect more components or users.

- Security considerations
    - DevOps - focuses on collaboration between development and operations to streamline workflows and improve deployment speed.
    - DevSecOps - integrates security practices into the DevOps process, ensuring security is a shared responsibility and built into every stage of the pipeline, while maintaining a balance with rapid release.
    - SecDevOps - emphasizes shifting security considerations earlier in the development process, embedding security measures from the initial design through to production. This practice focuses on security, prioritizing risk mitigation over delivery speed.