Name:	Deepanshu Aggarwal, Kunal Bhatia
UID	2021300002 , 2021300010
Project Name	SecureSpot:Crime Management Tool
Experiment Number	9
Aim	Continuous Integration using DevOp

EXPERIMENT NO. 9

AIM: Continuous Integration using DevOp

THEORY:

DevOps Model:

DevOps is a set of practices, tools, and a cultural philosophy that automates and integrates the processes between software development and IT teams. It emphasizes team empowerment, cross-team communication and collaboration, and technology automation.

How does DevOps work?

A DevOps team includes developers and IT operations working collaboratively throughout the product lifecycle, in order to increase the speed and quality of software deployment. It's a new way of working, a cultural shift, that has significant implications for teams and the organizations they work for.

Under a DevOps model, development and operations teams are no longer "siloed." Sometimes, these two teams merge into a single team where the engineers work across the entire application lifecycle — from development and test to deployment and operations — and have a range of multidisciplinary skills.
DevOps teams use tools to automate and accelerate processes, which helps to increase reliability. A DevOps toolchain helps teams tackle important DevOps fundamentals including continuous integration, continuous delivery, automation, and collaboration.

DevOps values are sometimes applied to teams other than development. When security teams adopt a DevOps approach, security is an active and integrated part of the development process. This is called DevSecOps.

Benefits of DevOps:

Some benefits of DevOps include:

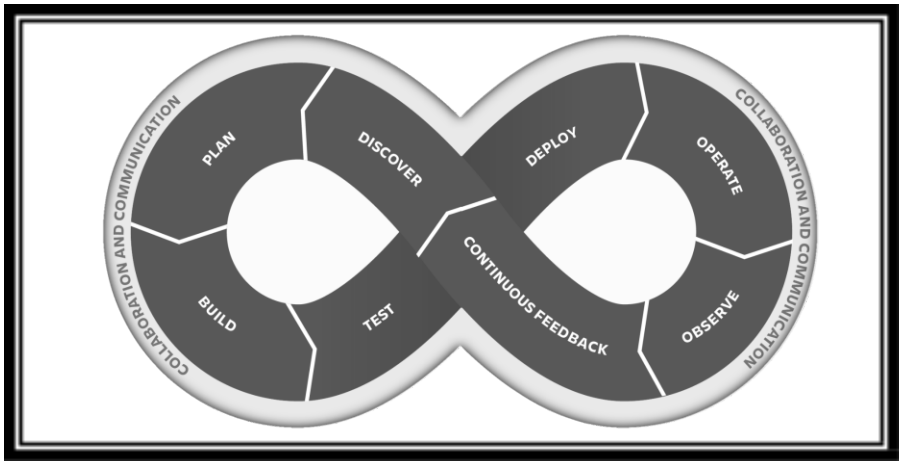
- Faster, better product delivery
- Faster issue resolution and reduced complexity
- Greater scalability and availability

More stable operating environments

- Better resource utilization
- Greater automation
- Greater visibility into system outcomes
- Greater innovation

The DevOps lifecycle:

DevOps lifecycle is a combination of different phases of continuous software development, integration, testing, deployment, and monitoring. A competent DevOps lifecycle is necessary to build superior quality software through the system.



Here are some important DevOps Lifecycle phases / Key components of DevOps:

1. **Continuous Development:** The planning and coding of the software are involved in this phase. It is the planning phase that decides the vision of the project.

2. **Continuous Integration:** The new feature code is continuously integrated with the existing code. It is therefore a continuous development of software. The updated code is then integrated continuously and smoothly with the systems to reflect changes to the end users.

3. **Continuous Testing:** In this phase, the developed software is continuously tested for bugs. For continuous testing, test automation tools such as TestNG, JUnit, Selenium, etc are used.

4. **Continuous Deployment:** In this phase, the code is deployed to the production servers. It is essential to make sure that the code is correctly used on all the servers.

5. **Continuous Monitoring:** This is a very crucial stage of the DevOps life cycle where you continuously monitor the performance of your application. The system errors such as low memory, server not reachable, etc are resolved in this phase.

6. **Continuous Feedback:** The application development is consistently improved by analyzing the results from the operations of the software. During this phase, customer behaviour is evaluated regularly on each release to improve future releases and deployments.

7. **Continuous Operations:** The last phase in the DevOps lifecycle is crucial for reducing planned downtime, such as scheduled maintenance. Continuous operation automates the process of launching the app and its updates. It uses container management systems like Kubernetes and Docker to eliminate downtime.

What is CI/CD?

"CI/CD" stands for the combined practices of Continuous Integration (CI) and Continuous Delivery (CD). It falls under DevOps (the joining of development and operations) and combines the practices of continuous integration and continuous delivery. CI/CD automates much or all of the manual human intervention traditionally needed to get new code from a commit into production such as build, test, and deploy, as well as infrastructure provisioning. With a CI/CD pipeline, developers can make changes to code that are then automatically tested and pushed out for delivery and deployment.

Steps involved in CI/CD:

Each change on the master Git branch performs the following steps:

- Build code
- Run unit tests
- If the tests pass, a Deploy block updates the production code that runs in the cloud

PROCEDURE:

Netlify Setup:

- Setup netlify account and add your project files
- Generate a new access token
- Go to Site Overview => Site Settings => General => Copy site ID

Gitlab Setup:

- Commit and push project files to gitlab remote repository branch
- Go to settings => ci/cd => add these two variables: NETLIFY_AUTH_TOKEN, NETLIFY_SITE_ID
- Clone the project on your local machine

Local Machine Setup And Changes:

- Checkout your branch and make the changes in your project
- Commit and push the changes to remote repository and wait for pipeline to succeed
- Merge the created branch to the main branch and wait for pipeline to get succeed

Jenkins Definition:

Jenkins is an automation server written in Java that helps build, test, and continually deploy software. The open-source system is currently one of the leading automation servers.

In general, Jenkins provides support for:

- Various version control tools such as Git.
- Building Ant and Apache Maven-based projects.
- Running Bash scripts and Windows batch files.

A Short History of Jenkins

The Jenkins project started in 2004 under the name Hudson. The developer Kohsuke Kawaguchi, who worked at Sun systems, wanted to create a method to perform continuous integration. The idea was to test the code before committing to avoid breaking builds.

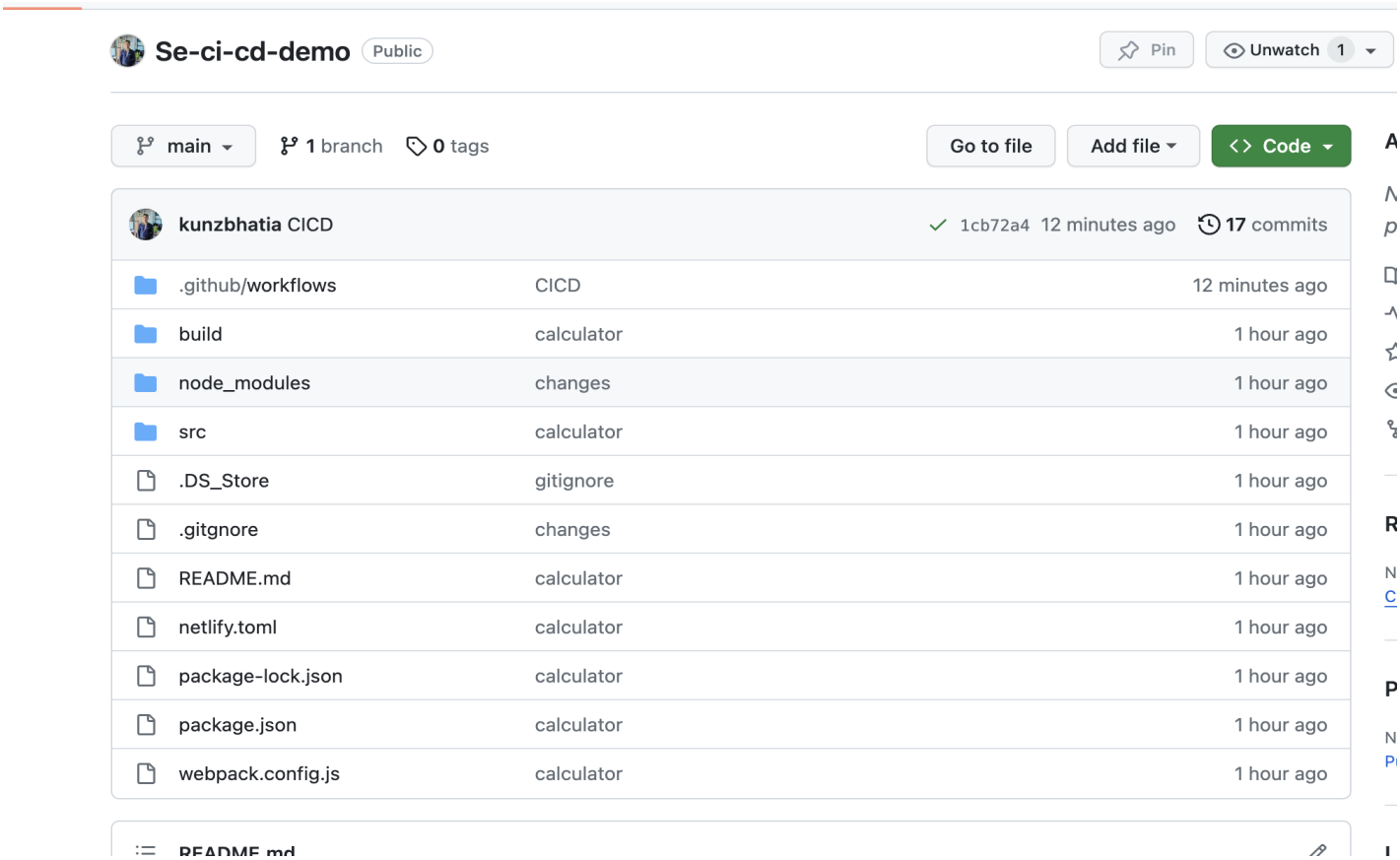
The idea proved successful and quickly spread to the rest of his team. As a result, Kohsuke Kawaguch created the Jenkins project and open-sourced the program. The usage spread across the world with a current estimate of 1.6 million users.

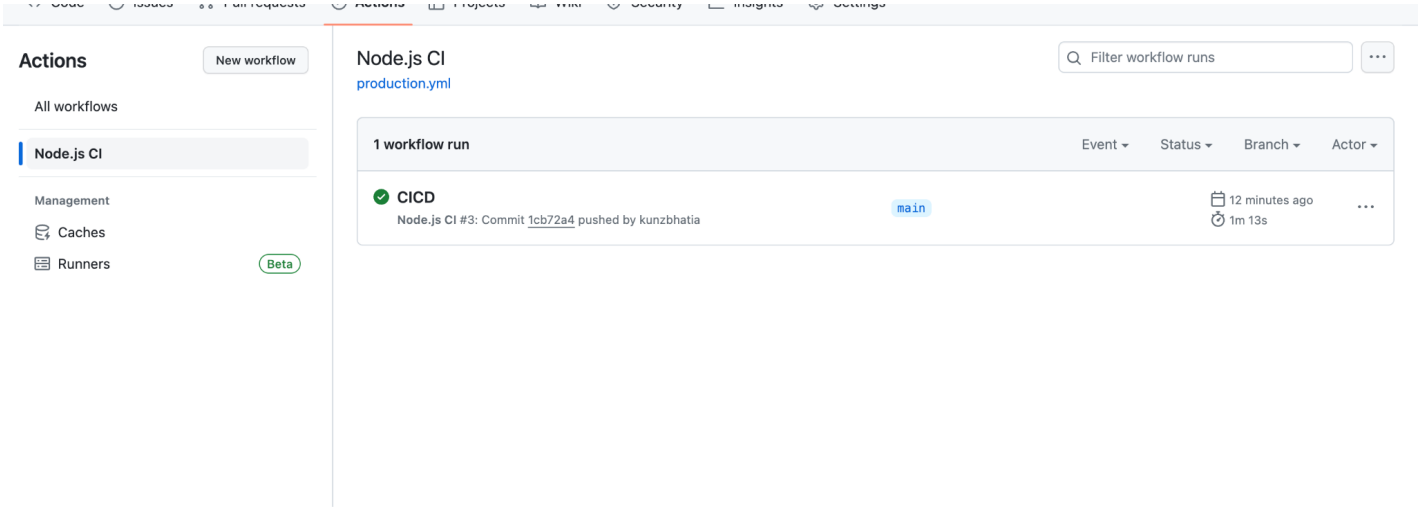
What Is Jenkins Used For?

Although Jenkins started as a continuous integration tool, the current use covers the whole software delivery pipeline, including deployment.

The program runs web containers and plugins, such as Apache Tomcat, and helps manage lifecycle and access rights requests. Over 1700 plugins for Jenkins enrich the software integration, automation, and delivery processes and provide a customizable environment.

Deployment on Netlify:





Triggered via push 12 minutes ago

kunzbhatia pushed 1cb72a4 main

Status

Success

Total duration

1m 13s

Artifacts

1

production.yml

on: push

build 10s — test 10s — deploy 31s

Annotations

3 warnings

eam / kunal-bhatia-cicd

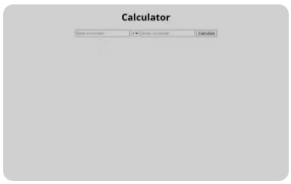
Search Netlify...

kunal-bhatia-cicd

- https://kunal-bhatia-cicd.netlify.app

Manual deploys.

Published at 1:00 PM.



Site configuration

Favorite site

Production deploys

Production Published

1:00 PM: No deploy message

Production

12:52 PM: No deploy message

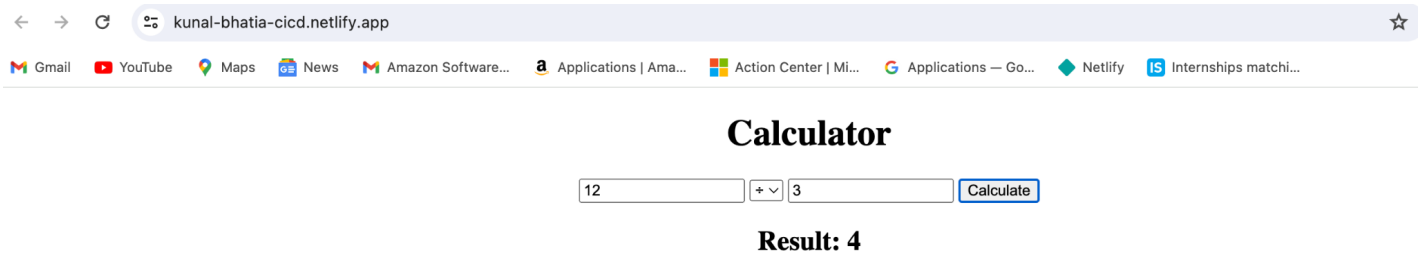
Deploy Previews

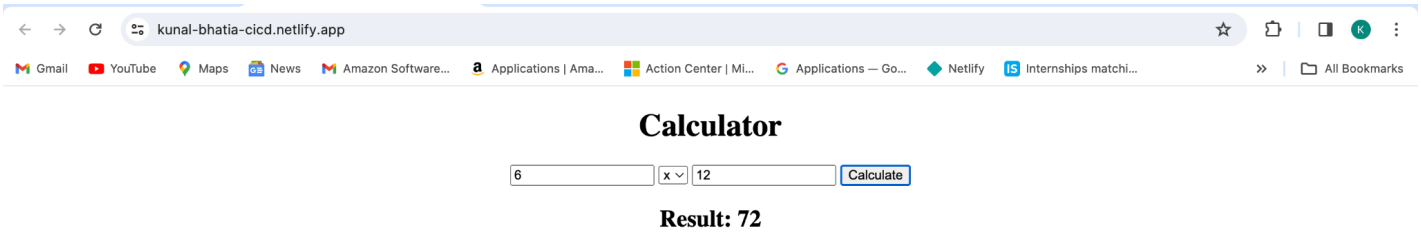
Auto-preview your PI

Github Link: https://github.com/kunzbhatia/Se-ci-cd-demo

Deployed Website:

Link : https://kunal-bhatia-cicd.netlify.app





CONCLUSION:

Thus, in this experiment, the DevOps concepts and its lifecycle was studied. CI/CD pipeline was successfully implemented using tools such as Gitlab and Netlify