Group members

1. Amrit Saini 22BSA10149
2. Anvitha.N 22BSA10128
3. Niranjan Yalamanda 22BSA10397

DevOps CI/CD Pipeline Project Report

Project Title: Automated CI/CD Pipeline Using Git, Jenkins, and Docker

Objective: The primary goal of this project is to implement a Continuous Integration and Continuous Deployment (CI/CD) pipeline that automates the software development lifecycle from code integration to deployment. This is achieved using Git for version control, Jenkins for pipeline automation, and Docker for containerization.

Tools & Technologies Used:

- **Git** – Version control system to manage code
- **GitHub/GitLab** – Remote repository for source code
- **Jenkins** – Automation server to build, test, and deploy code
- **Docker** – Containerization platform to build and run applications in isolated environments
- **Docker Hub** – Remote container registry to store Docker images

---

Pipeline Overview: The implemented pipeline is a fully automated CI/CD process that performs the following steps:

1. **Code Commit & Trigger:**
    - Developers push code to the Git repository.

   - Jenkins is configured with webhooks or polling to trigger the pipeline upon code changes.

2. **Checkout Stage:**
   - Jenkins pulls the latest code from the Git repository.
3. **Build Stage:**
   - The application is built using appropriate tools (e.g., Maven, Gradle, NPM).
4. **Testing Stage:**
   - Unit tests are executed to validate code correctness.
   - Build is aborted if tests fail.
5. **Docker Image Creation:**
   - A Dockerfile is used to create a Docker image of the application.
   - Docker image is tagged with version identifiers.
6. **Push to Docker Hub:**
   - The Docker image is pushed to a Docker Hub repository for storage and accessibility.
7. **Deployment Stage:**
   - Jenkins deploys the Docker container to a production/staging server.
   - Deployment can be done to cloud services (e.g., AWS EC2, Kubernetes cluster).

Pipeline Visualization: The following images represent different stages and flow of the pipeline:

- **Build Pipeline.png** – Visual depiction of the entire CI/CD pipeline workflow.
- **Git-Docker-Jenkins.png** – Integration between Git, Docker, and Jenkins.

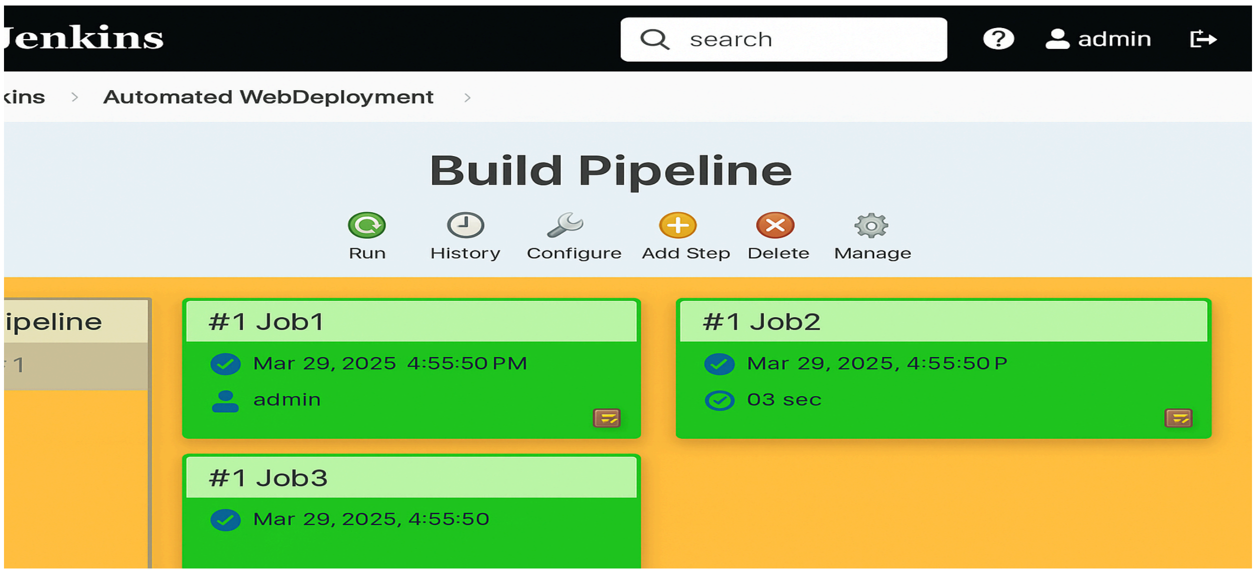
- **pipeline-1.png** – Jenkins stages: Checkout, Build, Test, Dockerize, Deploy.
- **pipeline-2.png** – Detailed process flow with enhanced stages.
- **Trigger.png** – CI trigger mechanism based on Git commits or pull requests.

Conclusion: This project demonstrates a successful implementation of a CI/CD pipeline. The automation of the build, test, and deployment process significantly reduces manual effort, ensures consistency, and speeds up the delivery cycle. The use of Docker ensures environment consistency and portability, while Jenkins orchestrates the complete workflow seamlessly.

Future Enhancements:

- Add Slack/Email notifications for pipeline success/failure.
- Implement approval gates before production deployment.
- Integrate with Kubernetes for dynamic scaling.
- Include automated integration tests and code quality checks.

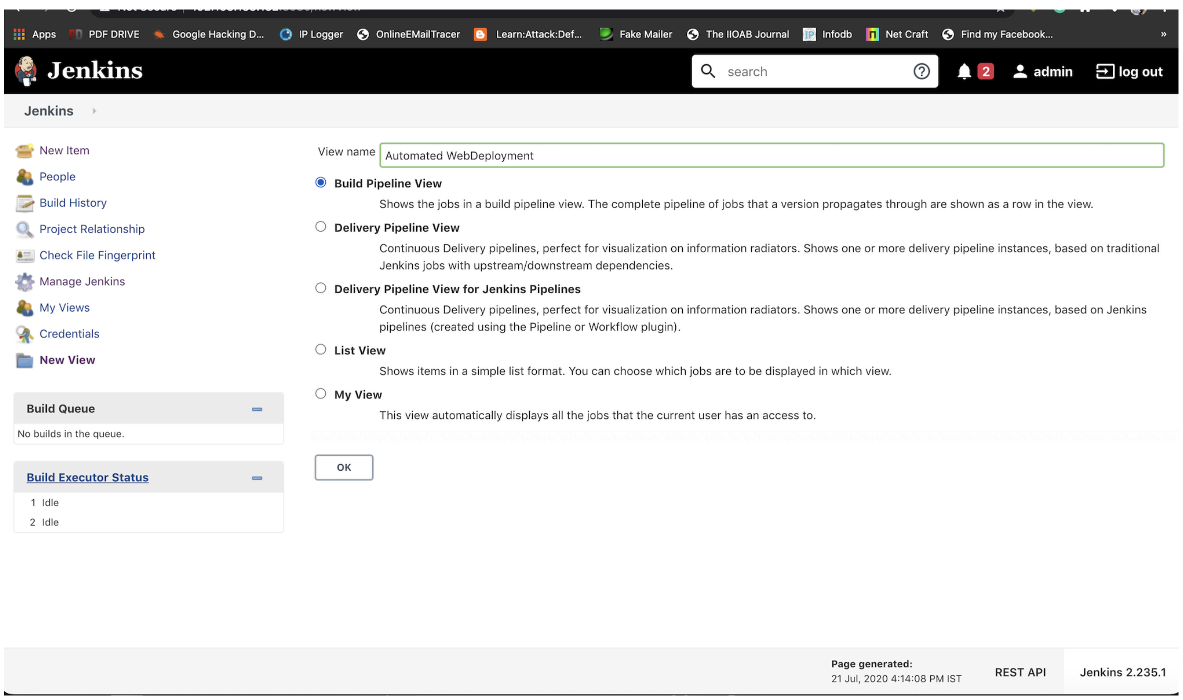
ScreenShots-



#1 Job1
Mar 29, 2025 4:55:50 PM
admin

#1 Job2
Mar 29, 2025, 4:55:50 P
03 sec

#1 Job3
Mar 29, 2025, 4:55:50


Apps | PDF DRIVE | Google Hacking D... | IP Logger | OnlineEMailTracer | Learn:Attack:Def... | Fake Mailer | The IIOAB Journal | Infodb | Net Craft | Find my Facebook...

Jenkins | search | 2 | admin | log out

Jenkins ›

- New Item
- People
- Build History
- Project Relationship
- Check File Fingerprint
- Manage Jenkins
- My Views
- Credentials
- New View

Build Queue

No builds in the queue.

Build Executor Status

1 Idle

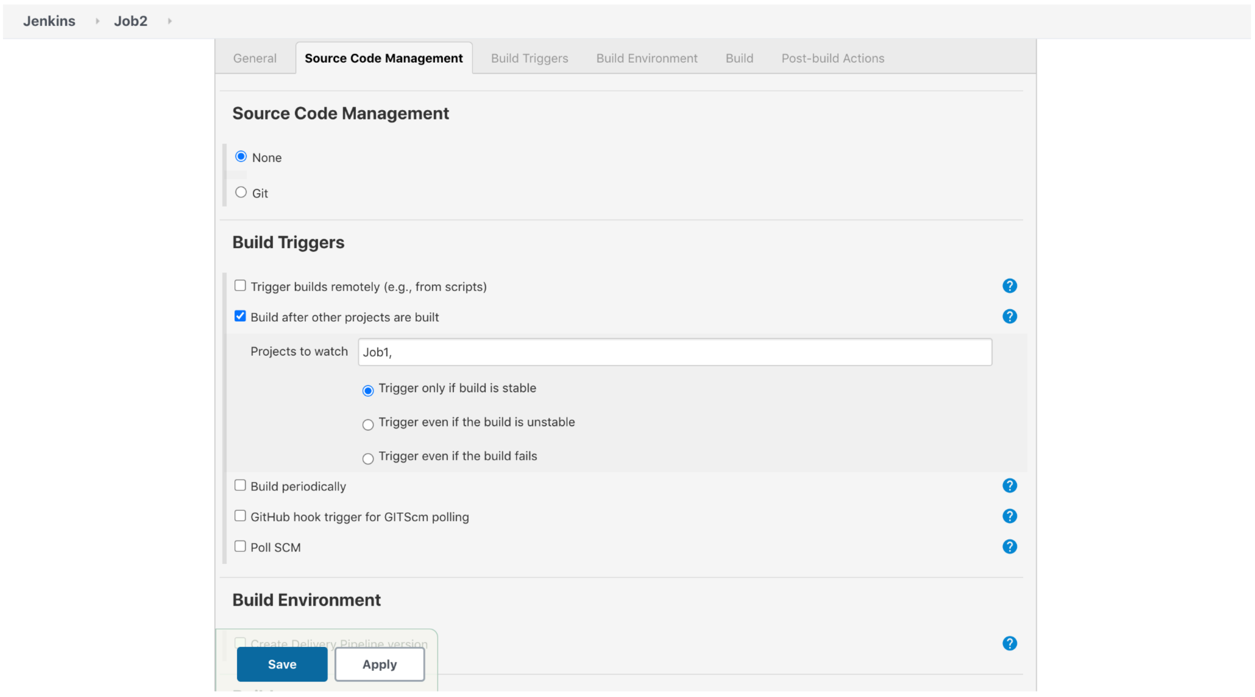
2 Idle

View name: Automated WebDeployment

- [✓] **Build Pipeline View**
  Shows the jobs in a build pipeline view. The complete pipeline of jobs that a version propagates through are shown as a row in the view.
- **Delivery Pipeline View**
  Continuous Delivery pipelines, perfect for visualization on information radiators. Shows one or more delivery pipeline instances, based on traditional Jenkins jobs with upstream/downstream dependencies.
- **Delivery Pipeline View for Jenkins Pipelines**
  Continuous Delivery pipelines, perfect for visualization on information radiators. Shows one or more delivery pipeline instances, based on Jenkins pipelines (created using the Pipeline or Workflow plugin).
- **List View**
  Shows items in a simple list format. You can choose which jobs are to be displayed in which view.
- **My View**
  This view automatically displays all the jobs that the current user has an access to.

OK

Page generated: 21 Jul, 2020 4:14:08 PM IST | REST API | Jenkins 2.235.1


Jenkins › Job2 ›

General | **Source Code Management** | Build Triggers | Build Environment | Build | Post-build Actions

Source Code Management

- [✓] None
- Git

Build Triggers

- [ ] Trigger builds remotely (e.g., from scripts)
- [✓] Build after other projects are built

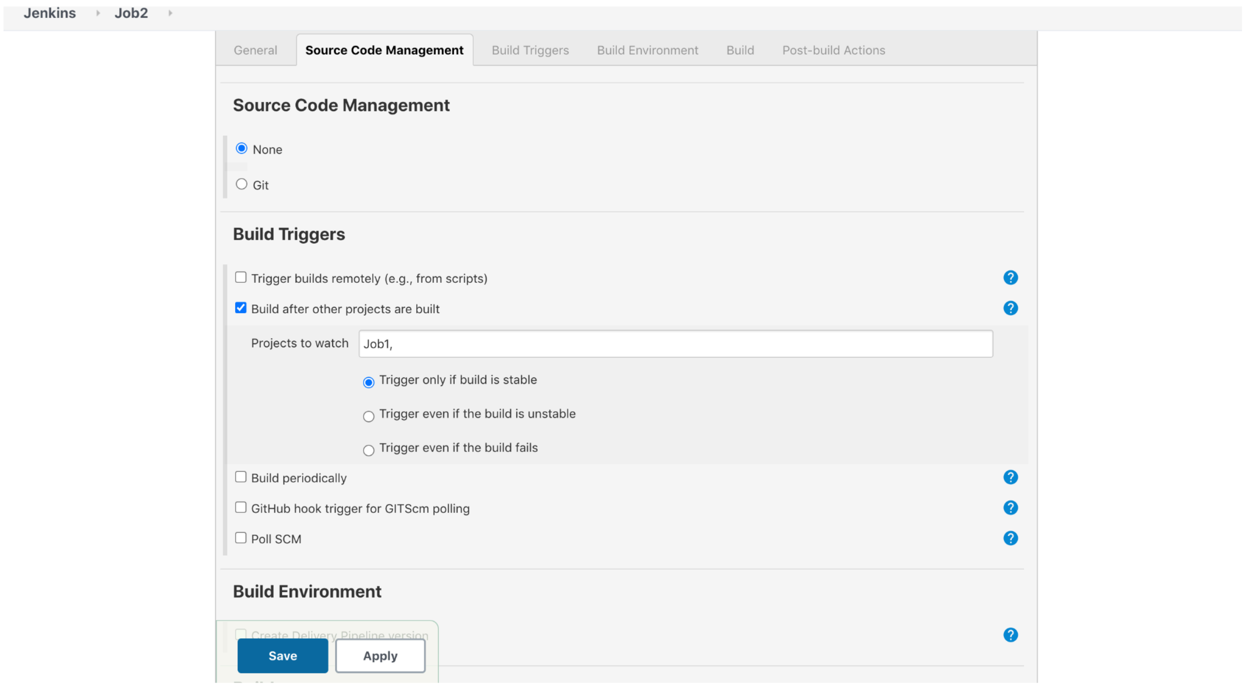
  Projects to watch: Job1,

  - [✓] Trigger only if build is stable
  - Trigger even if the build is unstable
  - Trigger even if the build fails
- [ ] Build periodically
- [ ] GitHub hook trigger for GITScm polling
- [ ] Poll SCM

Build Environment

- [ ] Create Delivery Pipeline version

Save | Apply



Jenkins › Job2 ›

General | Source Code Management | Build Triggers | Build Environment | Build | Post-build Actions

Source Code Management

- (●) None
- ( ) Git

Build Triggers

- [ ] Trigger builds remotely (e.g., from scripts)
- [✓] Build after other projects are built
  - Projects to watch: Job1,
  - (●) Trigger only if build is stable
  - ( ) Trigger even if the build is unstable
  - ( ) Trigger even if the build fails
- [ ] Build periodically
- [ ] GitHub hook trigger for GITScm polling
- [ ] Poll SCM

Build Environment

- [ ] Create Delivery Pipeline version

Save | Apply

Git hub:-