DEVOPS TRAINING - DAY 2: Jenkins & Docker Integration

Step 1: Create a Repository in GitHub

Go to GitHub and create a new repository.

Step 2: Generate a GitHub Personal Access Token

- Navigate to Settings > Developer Settings > Personal Access Tokens
- Click Generate New Token (Classic)
- Copy and store the token securely

Step 3: Configure Jenkins for GitHub Integration

- Open Jenkins
- Click New Item > Pipeline
- In Configuration, add the GitHub repository URL
- Save the configuration

Step 4: Verify Jenkins Setup

Check the Jenkins Status Page to confirm that the setup is successful.

Step 5: Clone Git Repository

```bash
git clone https://github.com/your-username/your-repository.git
cd your-repository
```

Step 6: Add and Push Code to GitHub

```bash
git add .
git commit -m "Initial commit"
git push origin main
```

Step 7: Verify Uploaded Files in GitHub

Go to your GitHub repository and check if all files are uploaded successfully.

Step 8: Log in to Docker

```bash
docker login
```
Enter your Docker Hub username and password/token when prompted.

Step 9: Configure Global Credentials in Jenkins

- Open Jenkins > Manage Jenkins > Manage Credentials
- Copy and update the Global Credentials
- Save the changes

Step 10: Commit and Push the Jenkinsfile

```bash
git add Jenkinsfile
git commit -m "Added Jenkinsfile"
git push origin main
```

Step 11: Verify Jenkinsfile in GitHub

Go to GitHub and confirm that the Jenkinsfile is successfully pushed.

Step 12: Grant Jenkins Docker Permissions

```bash
sudo usermod -aG docker jenkins
sudo systemctl restart jenkins
```

Step 13: Build Jenkins Pipeline

- In Jenkins, click Build Now
- Monitor the Console Output

Step 14: Run Application Locally

After a successful build, start the application:

```bash
localhost:5001
```

Step 15: Check Docker Image Repository

Verify that the Docker image is built and stored in the repository:

```bash
docker images
```