**Deliver Jenkins job configuration instructions for building Docker image on code commits.**

**Jenkins Freestyle Pipeline Setup:**

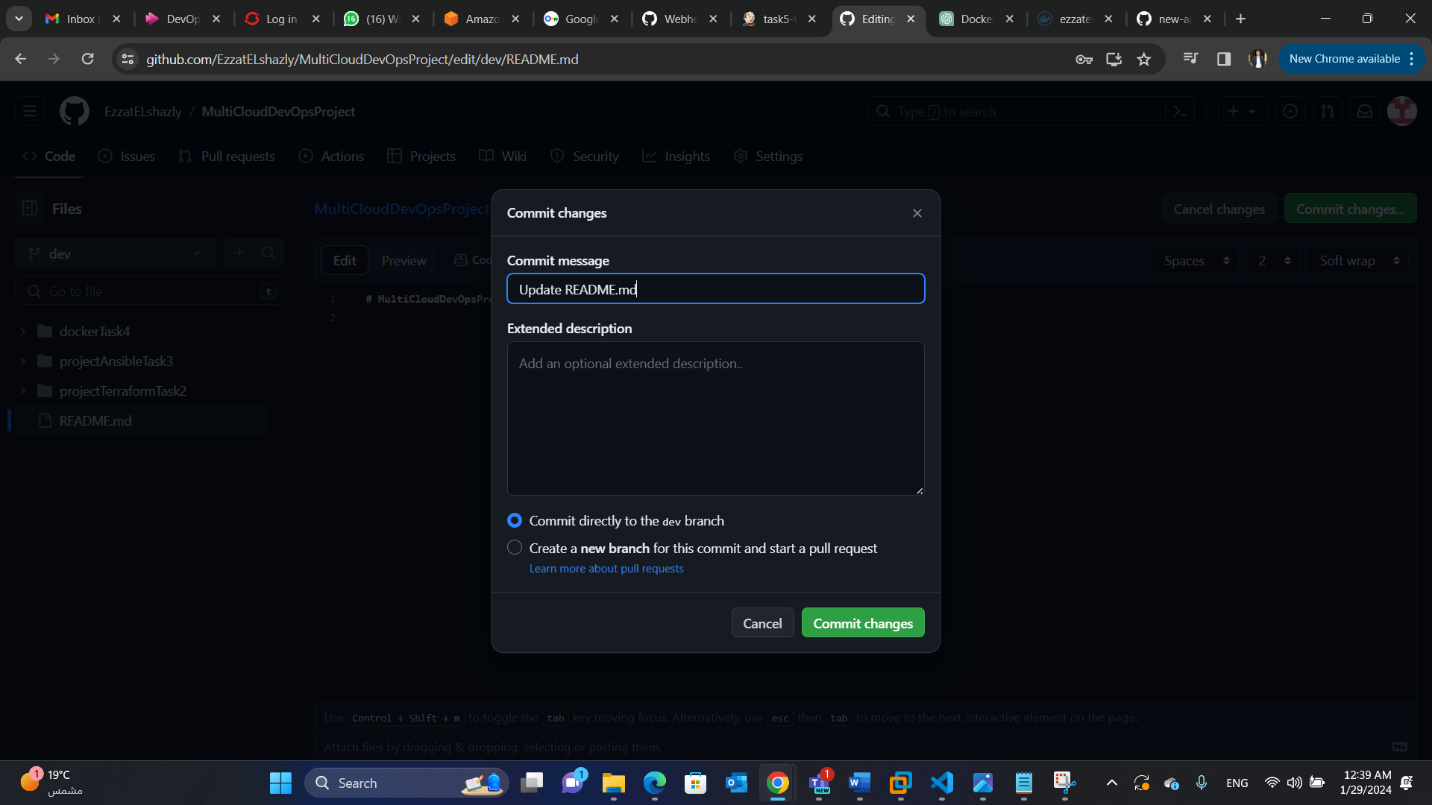
1. **Open Jenkins Console:**
   * Navigate to "Manage Jenkins."
   * Under "Security," select "Credentials."
2. **Add Global Credentials:**
   * Click on "Add Credentials."
   * Add DockerHub and GitHub credentials (username/password or secret text with login token).
3. **Create a Freestyle Project:**
   * Click on "New Item."
   * Enter a name for the project and choose "Freestyle project."
4. **Configure SCM (Source Code Management):**
   * Scroll to SCM section and choose "Git."
   * Add the repository URL.
   * Select GitHub credentials.
5. **Configure Build Triggers:**
   * In the "Configure" tab, scroll down to the "Build Triggers" section.
   * Check "GitHub hook trigger for GITScm polling."
6. **Add Execute Shell Build Step:**
   * Add a build step of type "Execute shell."
   * Enter the desired shell commands for example:
     + export DOCKER\_REGISTRY=ezzatelshazly
     + export DOCKER\_IMAGE=application-test1-ee
     + rm -rf MultiCloudDevOpsProject
     + git clone <https://github.com/EzzatELshazly/MultiCloudDevOpsProject.git>
     + cd dockerTask4/spring-boot-app-main
     + chmod +x gradlew
7. **Add Post-Build Actions:**
   * Add post-build actions of type "Build other projects."
   * Select the project to build.
   * Choose "Trigger only if builds are stable."

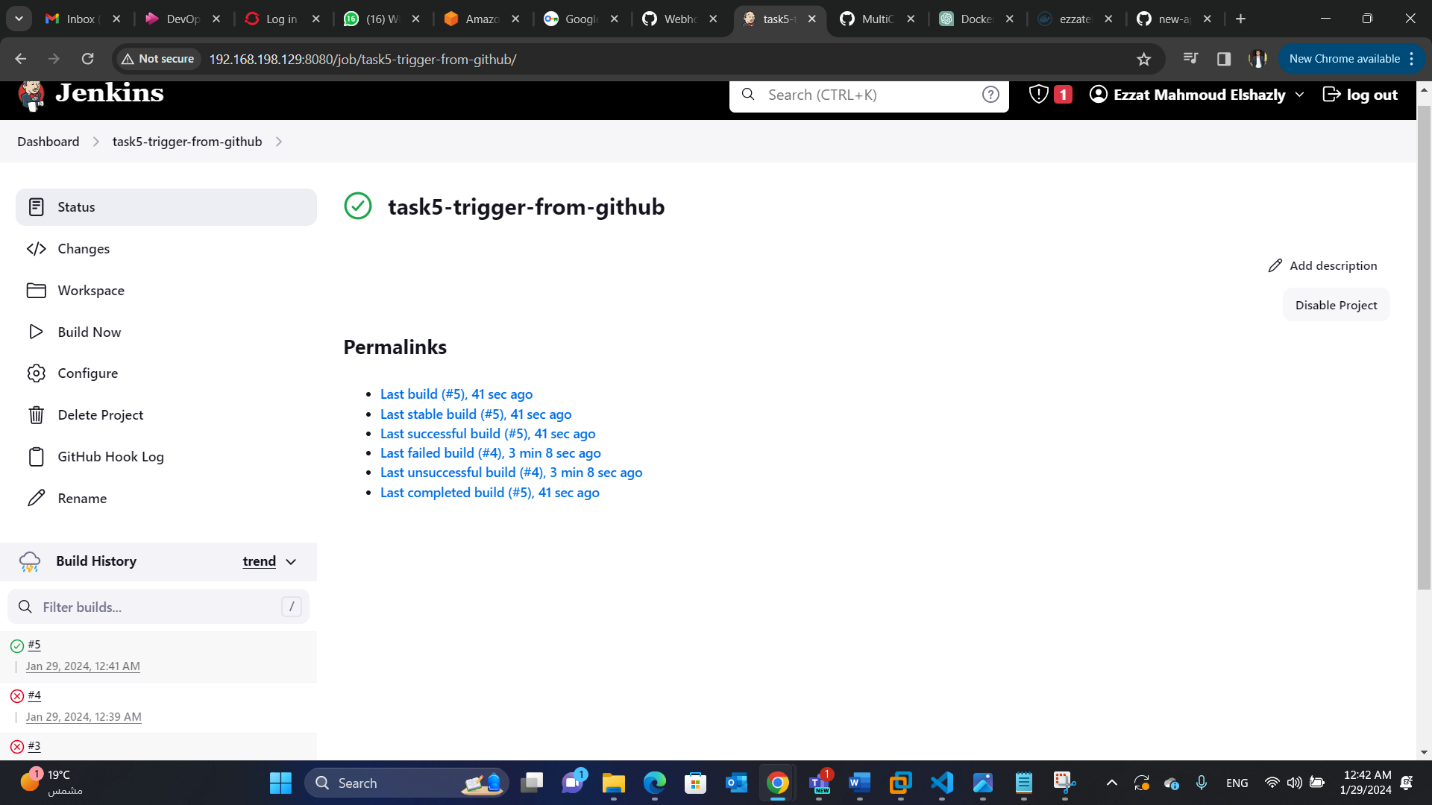
**GitHub Webhook Configuration:**

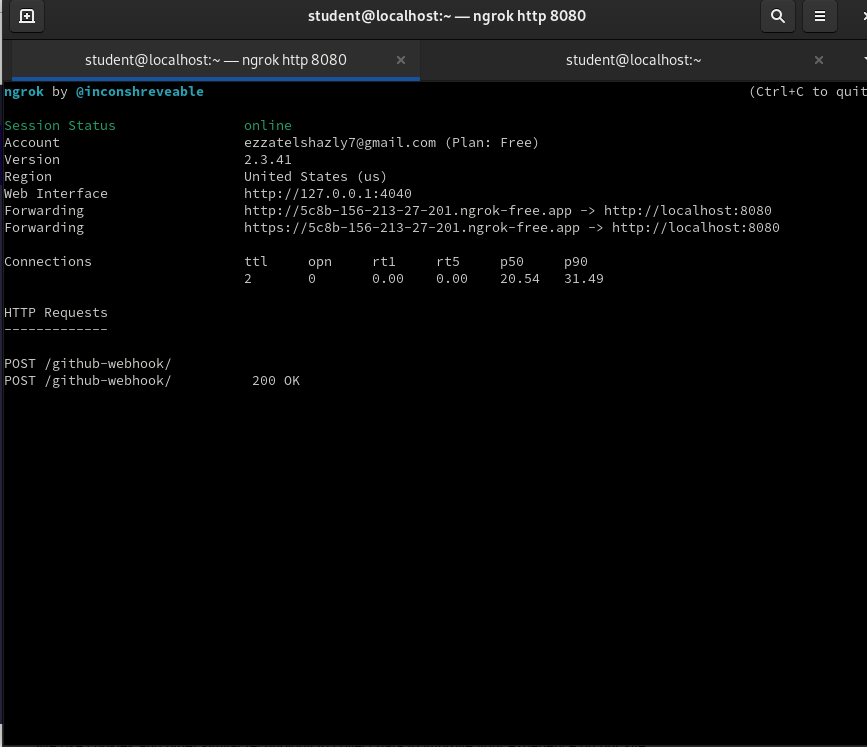
1. **Open GitHub Repository:**
   * Navigate to Settings > Webhooks > Add webhook.
2. **Configure Webhook:**
   * Enter **http://<public-ip>/github-webhook** in the Payload URL.
   * Options for **<public-ip>**:
     + For testing locally: Use ngrok or similar tools.
     + For production: Get the public IP of the EC2 instance.
3. **Set Content Type and Events:**
   * Choose "application/json" as the content type.
   * Select "Just the push event" as the trigger event.
4. **Add Webhook:**
   * Click on "Add webhook" to save the configuration.

Now, any changes committed and pushed to the GitHub repository should automatically trigger the Jenkins job to build. Make sure to adapt the placeholders like **<public-ip>** with the actual values and configure the pipeline steps according to your project's requirements.

**Screenshots:**

**Commit to GitHub.**

**Pipeline will build automatic whenever a commit to GitHub happens.**

**Ngrok for testing locally (not recommended).**