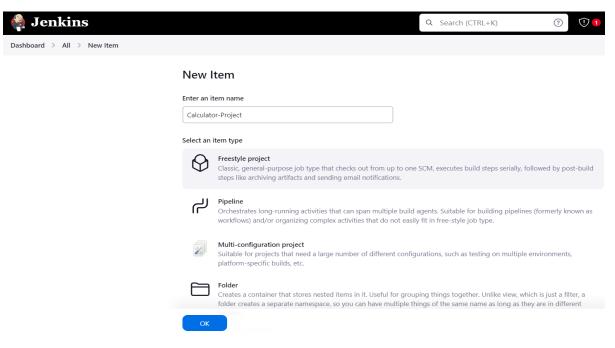
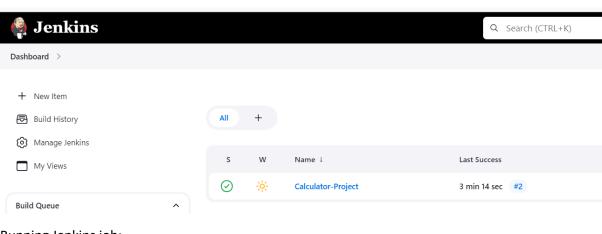
Experiment No. 5

Title: Applying CI/CD Principles to Web Development Using Jenkins, Git, and LocalHTTP Server.

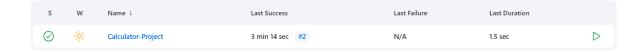
Step 1) Creating a Jenkins Job using "Freestyle Project" type:



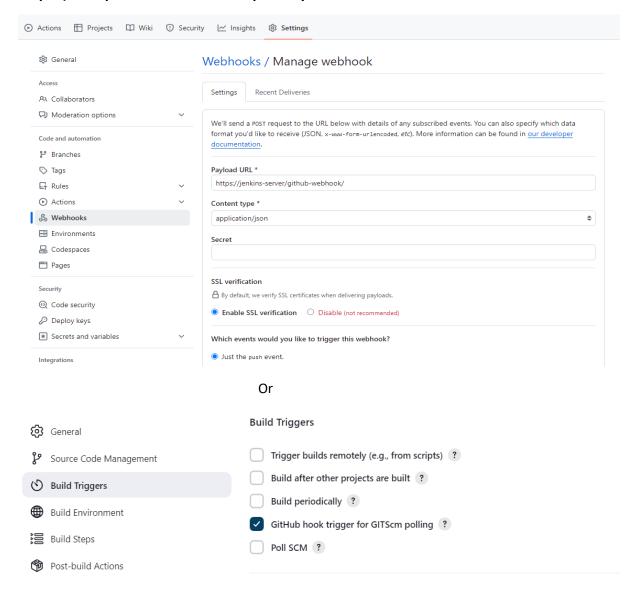
Set up the Jenkins job;



Running Jenkins job;

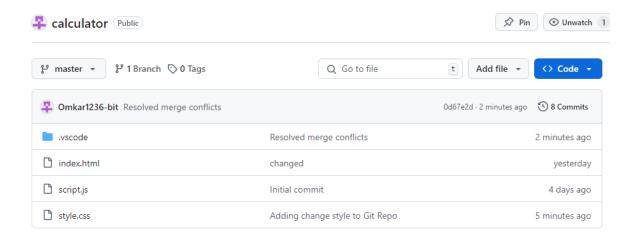


Step 2) Set up a Webhook in Git Repository



Step 3) Trigger the CI/CD Pipeline:

Git Repository before push changes;



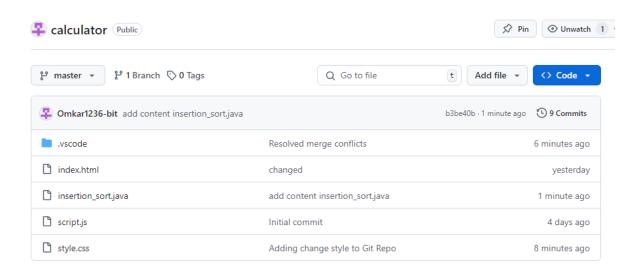
Commit the changes;

```
PS D:\Calculator> git init
Reinitialized existing Git repository in D:/Calculator/.git/
PS D:\Calculator> git add insertion_sort.java
PS D:\Calculator> git commit -m "add content insertion_sort.java"
[master b3be40b] add content insertion_sort.java
1 file changed, 40 insertions(+)
create mode 100644 insertion_sort.java
```

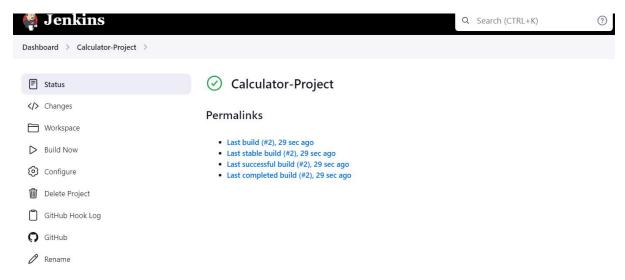
Pushing changes to Git Repository;

```
PS D:\Calculator> git push origin master
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 693 bytes | 693.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/Omkar1236-bit/calculator.git
   0d67e2d..b3be40b master -> master
```

After push Git Repository become;

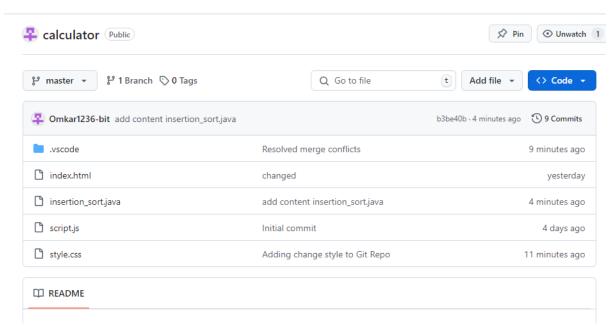


Monitor the Jenkins job's progress in the Jenkins web interface;



Step 4) Verify the CI/CD Pipeline.

The webhook should trigger the Jenkins job automatically, executing the build and deployment;



Hence the local HTTP server to verify that the web application has been updated with the latest changes.