

International Institute of Information Technology, Bengaluru

Software Production Engineering Mini Project Report

Scientific Calculator

Submitted by

Mohit Gupta (MT2024049)

In the guidance of Prof. B. Thangaraju



Problem Statement:

The goal of the project is to create a Scientific Calculator with the following user menu driven options :

1. Square Root Function - (\sqrt{x})
2. Factorial Function - ($x!$)
3. Natural Logarithm (base e) - ($\ln x$)
4. Power Function - (ab)

What is DevOps:

DevOps is a software development approach that unites development and operations teams, enabling them to collaborate throughout the entire application lifecycle. The primary objective of DevOps is to accelerate the delivery of applications and services.

Why DevOps:

DevOps enhances software delivery by automating and optimizing workflows, enabling businesses to deliver software more quickly and effectively. It also fosters improved collaboration between teams.

Benefits of DevOps:

- **Faster delivery:** DevOps allows businesses to release software updates more frequently and rapidly.
- **Improved quality:** By reducing errors, DevOps contributes to enhanced product quality.
- **Enhanced collaboration:** DevOps encourages breaking down silos between development and operations teams, promoting better teamwork.
- **Increased security:** DevOps integrates security practices into the development lifecycle, improving overall security.
- **Scalability:** DevOps supports the management of scalable solutions through automation and optimized workflows.
- **Better customer experience:** By delivering high-quality software, DevOps ensures that customer needs are met effectively.
- **Cost savings:** DevOps reduces the costs associated with slow and inefficient software delivery processes.

Tools and Technologies Used:

Programming Language: Java

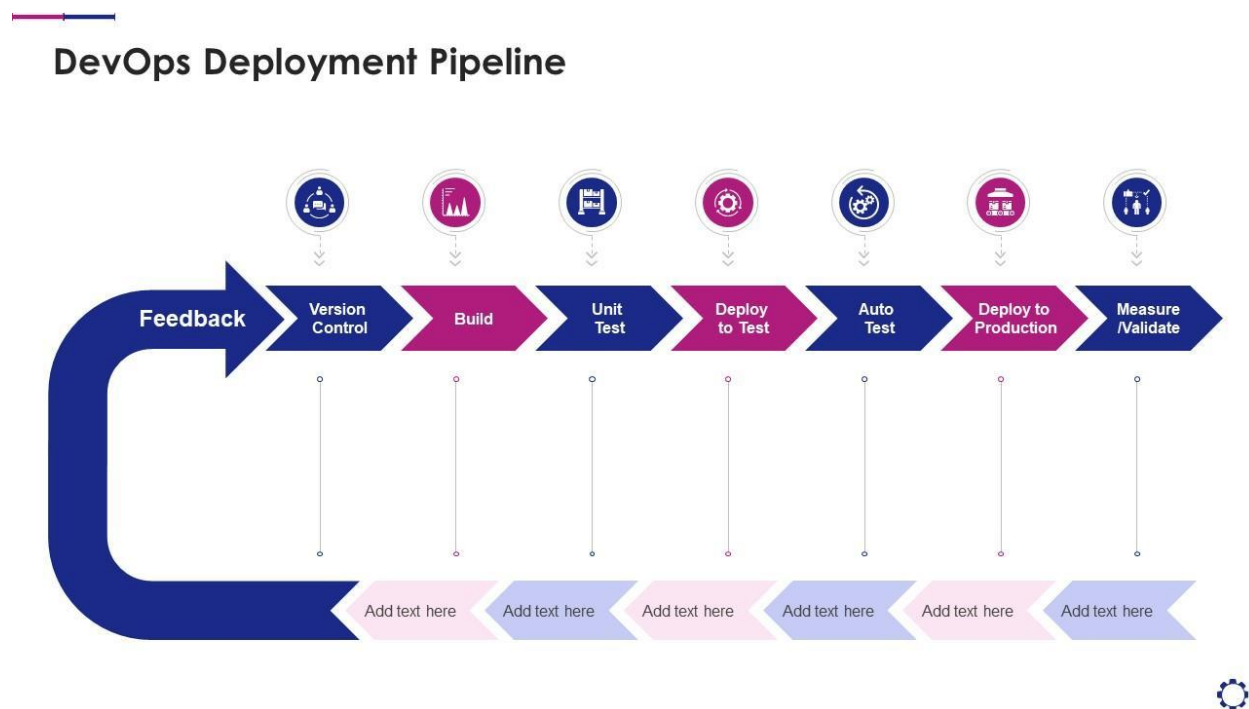
Version Control: Git & GitHub

Build Automation: Maven

CI/CD Pipeline: GitHub Actions / Jenkins

Containerization: Docker

CI/CD Pipeline:

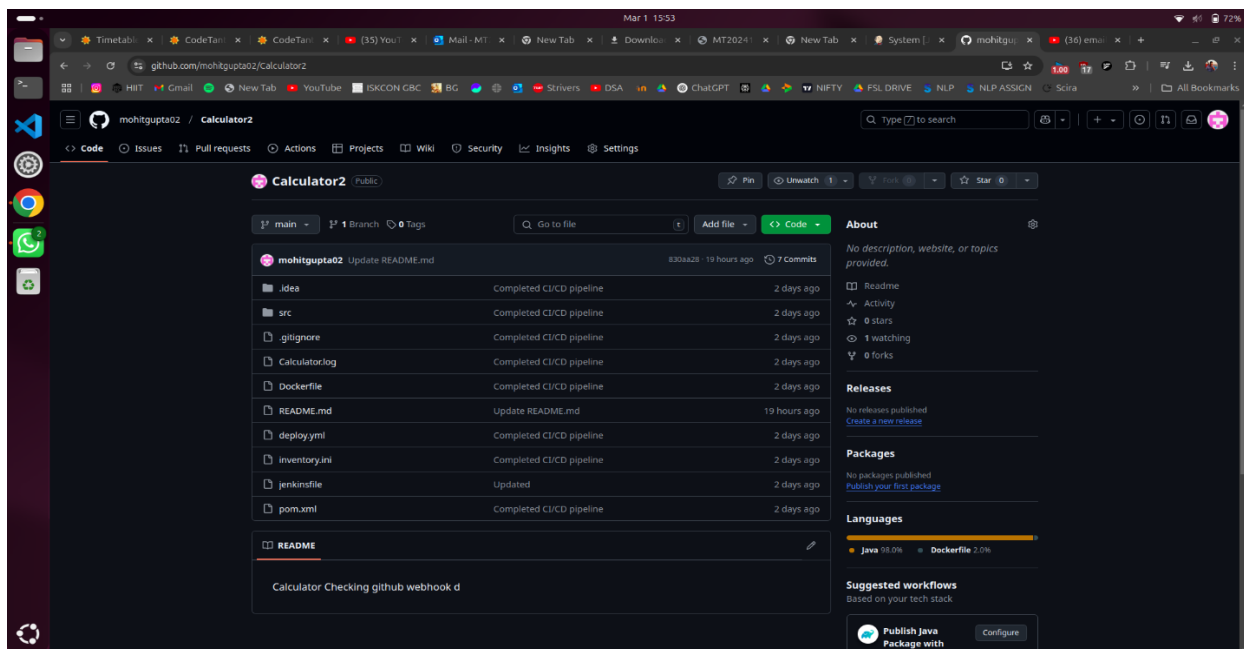


Source Code Management

SCM (Source Code Management) is a system used to track and manage changes made to a source code repository. It monitors modifications to the codebase and assists in resolving conflicts when merging updates from multiple contributors. SCM is often used interchangeably with Version Control.

The project is initially created on a local machine (local repository), and later pushed to a remote repository on platforms like GitHub. To push the local repository to the remote repository, a series of commands are executed, as outlined below:

- `git init` — Initializes a new Git repository in the local project directory.
- `git status` — Shows the current status of changes in the working directory.
- `git add` — Stages the changes to be committed.
- `git commit -m "Commit Message"` — Commits the staged changes with a descriptive message.
- `git remote add origin https://github.com/mohitgupta02/Calculator2.git` — Links the local repository to the remote GitHub repository.
- `git push -u origin main` — Pushes the committed changes to the remote repository's master branch.



Source Code Management

Docker:

- Docker is a tool for containerization that allows packaging software into a standardized unit for development, shipment, and deployment.
- It leverages kernel-level virtualization to create isolated environments known as containers.
- In this case, Docker is used to generate an image that runs the JAR file produced by the Maven build process.
- This image can then be uploaded to Docker Hub, where other machines can pull the image and instantiate containers to run the JAR file.

To build a Docker image, you can use the following command:

```
$ docker build -t <USERNAME>/<IMAGE_NAME>:<TAG>
```

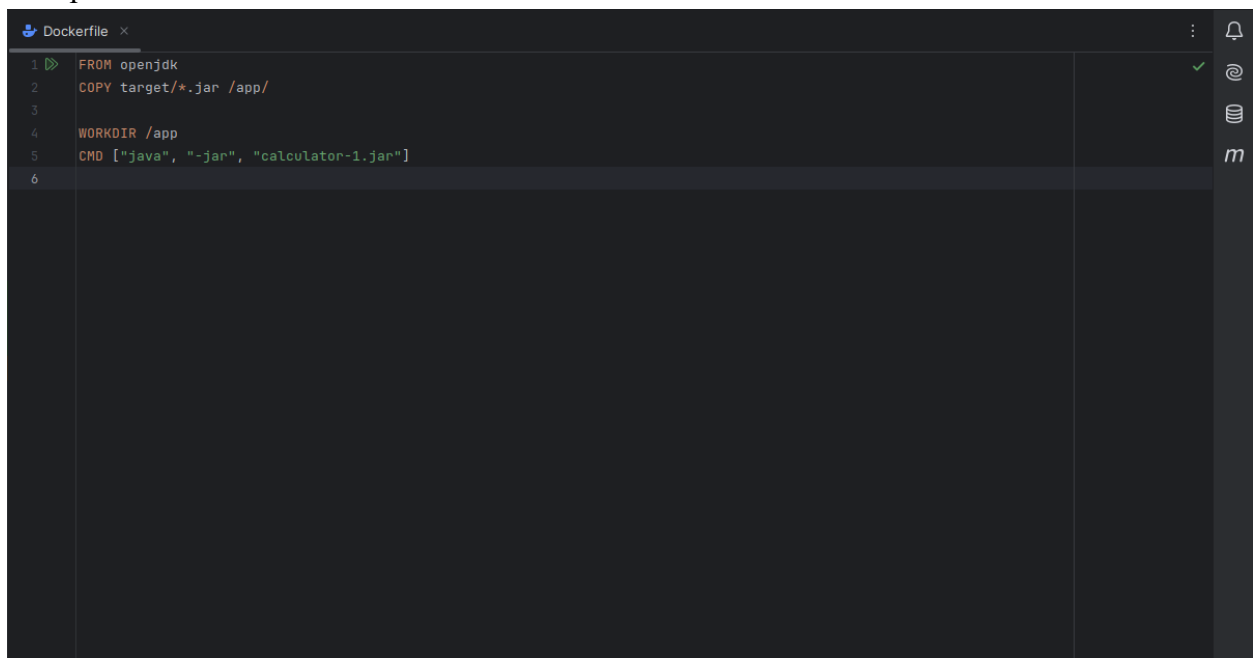
To push the Docker image to Docker Hub, use:

```
$ docker push <USERNAME>/<REPOSITORY_NAME>:<TAG>
```

The image is pushed from Jenkins after the build process and pulled during the Ansible job. You can run the Docker image with this command:

```
$ Docker run -i -t <IMAGE_NAME>
```

Once the Docker image is pushed to Docker Hub, you can verify it by checking your Docker Hub profile.

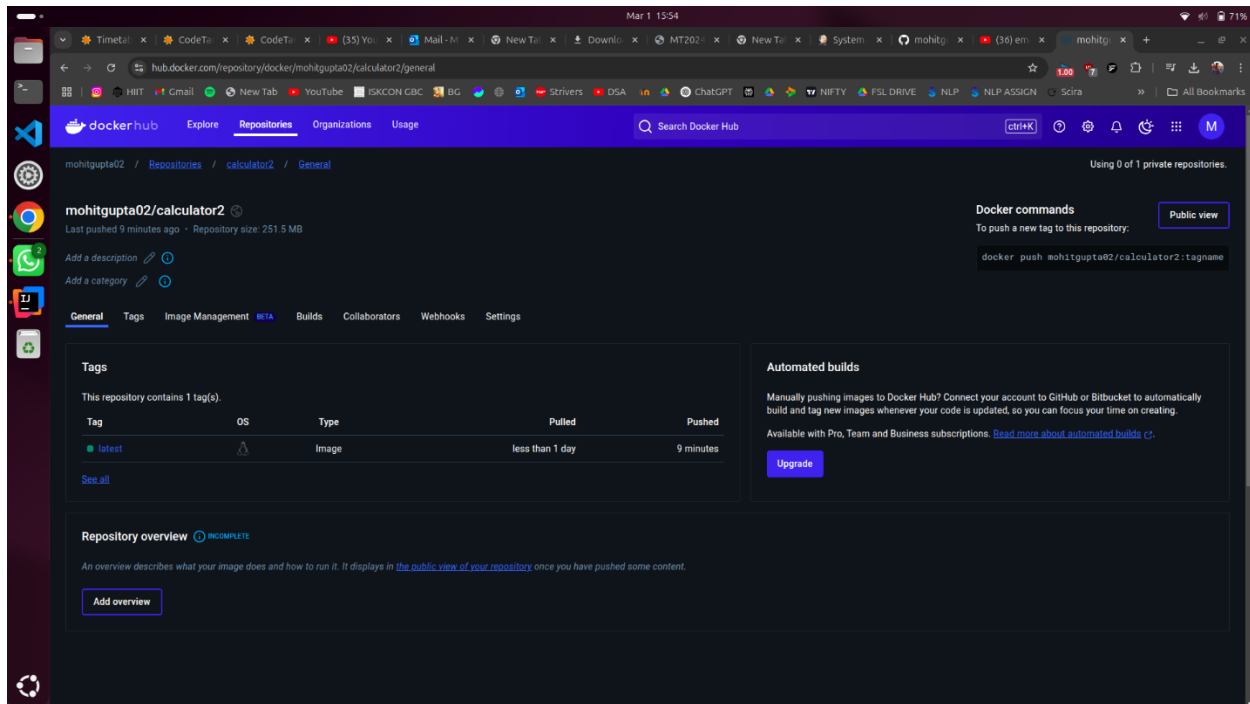
A screenshot of a code editor showing a Dockerfile. The file is named 'Dockerfile' and has a tab icon. The content of the Dockerfile is as follows:

```
1 FROM openjdk
2 COPY target/*.jar /app/
3
4 WORKDIR /app
5 CMD ["java", "-jar", "calculator-1.jar"]
6
```

The editor has a dark theme. On the right side, there are icons for a search, a list, and a terminal. A green checkmark is visible next to the first line of the Dockerfile.

DockerFile

- To create the Docker image and upload it to a remote repository on Docker Hub, we will define the steps in a Jenkins pipeline script.
- The Docker image is generated and uploaded to the user's Docker Hub repository.
- After the image is successfully uploaded to Docker Hub, the local image on the machine is removed.



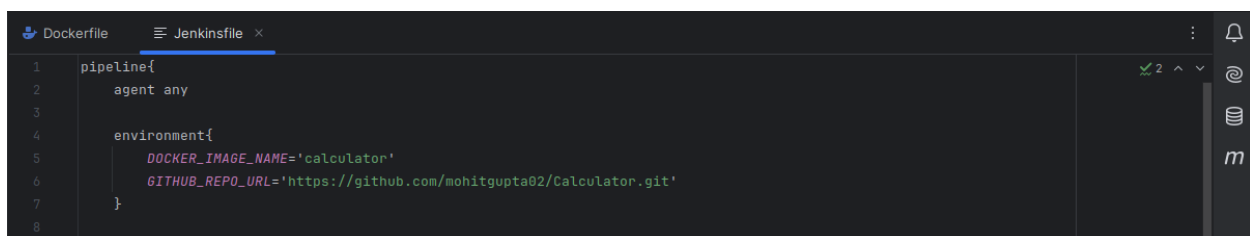
Docker Hub Repository

Jenkins:

- Jenkins is a robust tool that facilitates continuous integration and continuous delivery (CI/CD) for projects, regardless of the platform in use.
- It is an open-source application capable of handling all kinds of builds and CI processes.
- Jenkins can be integrated with a variety of testing and deployment tools.
- The pipeline is set up so that whenever a new commit is made to the GitHub repository, the pipeline is triggered, executing all steps in sequence.
- A Jenkins pipeline is an automated workflow that outlines the process of retrieving software from version control.
- Each change to the software undergoes a series of intricate processes before being released.
- This includes developing the software in a repeatable, reliable manner, and progressing through multiple stages of testing and deployment.
- All tasks are interconnected through the Jenkins pipeline.
- The pipeline pulls changes from GitHub, builds the project using Maven, creates the Docker image, pushes the image to Docker Hub, and then runs the Ansible playbook.

Create CI Pipeline

- A Jenkins pipeline is an automated representation of the steps involved in obtaining software from version control.
- Every change made to the software goes through a series of detailed processes before being released.
- This includes developing the software in a consistent and dependable way, as well as advancing the software through several stages of testing and deployment.

A screenshot of a code editor showing a Jenkinsfile configuration. The editor has a dark theme and a sidebar on the right with icons for search, settings, and other tools. The Jenkinsfile content is as follows:

```
1 pipeline{
2   agent any
3
4   environment{
5     DOCKER_IMAGE_NAME='calculator'
6     GITHUB_REPO_URL='https://github.com/mohitgupta02/Calculator.git'
7   }
8 }
```

Jenkins Pipeline

There are various stages in Jenkins Pipeline including Checkout, Build Docker Image, Push Docker Image and Run ansible playbook then there is post action to send email for acknowledge the status of pipeline.


```
9
10 stages{
11     stage('Checkout'){
12         steps{
13             script{
14                 git branch: 'main', url: "${GITHUB_REPO_URL}"
15             }
16         }
17     }
18
19     stage('Build and test') {
20         steps {
21             script {
22                 sh 'mvn clean package'
23             }
24         }
25     }
26
27     stage('Build Docker Image'){
28         steps{
29             script{
30                 docker.build("${DOCKER_IMAGE_NAME}",'.')
31             }
32         }
33     }
34
35     stage('Push Docker Images'){
36         steps{
37             script{
38                 docker.withRegistry('', 'DockerHub') {
39                     sh 'docker tag calculator mohitgupta02/calculator:latest'
40                     sh 'docker push mohitgupta02/calculator'
41                 }
42             }
43         }
44     }
45
46     stage('Run Ansible playbook') {
47         steps {
48             script {
49                 ansiblePlaybook(
50                     playbook: 'deploy.yaml'
51                 )
52             }
53         }
54     }
55 }
56
```

Stages of Pipeline

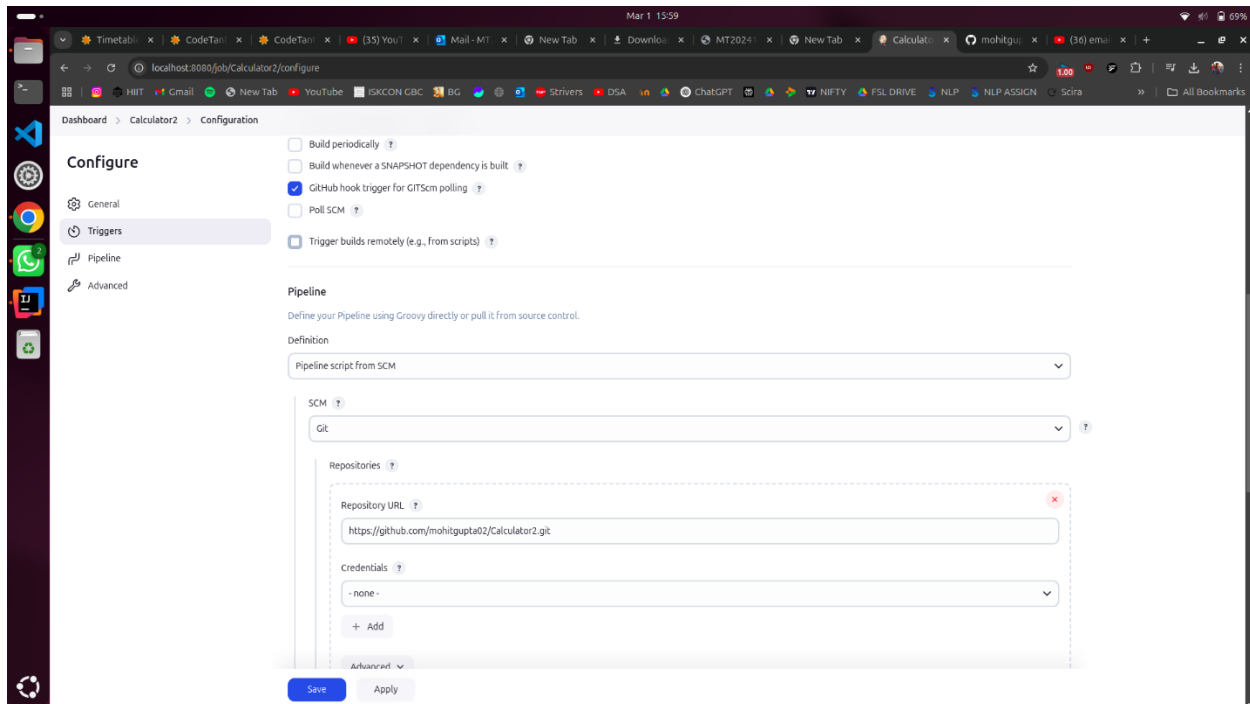
Configuration of Project in Jenkins

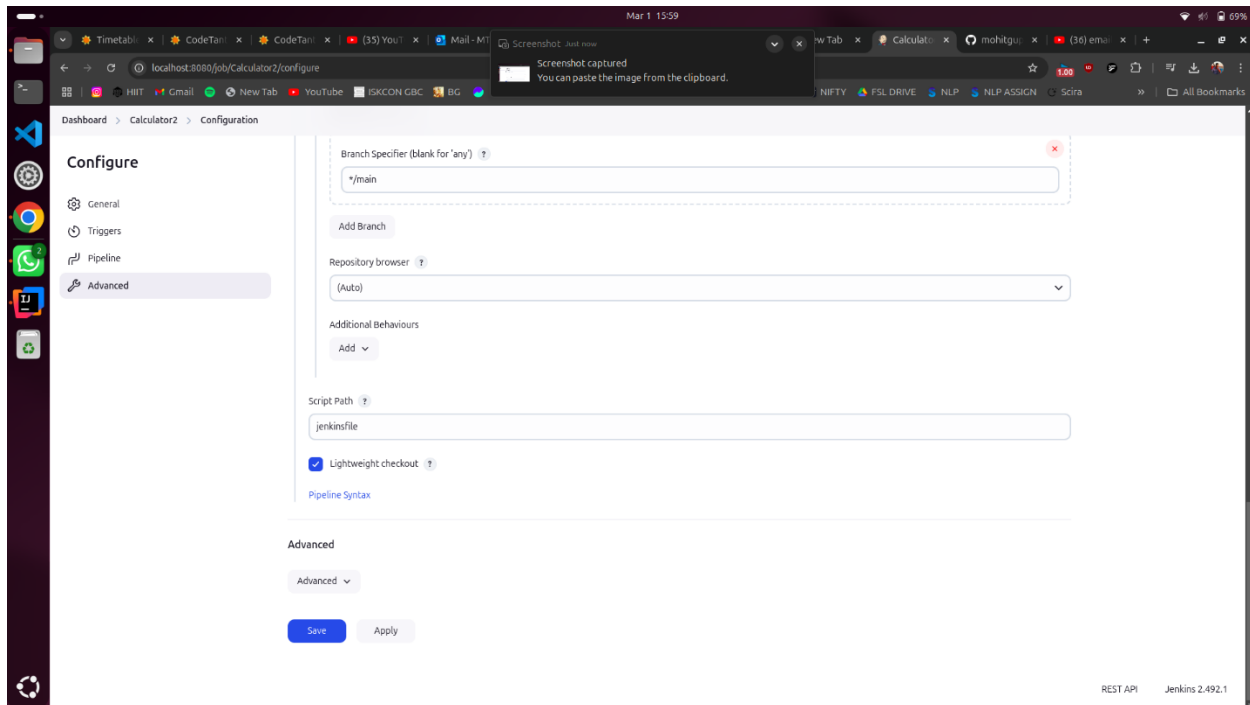
To run Jenkins in local machine there is a command to run it

```
$ sudo systemctl start jenkins
```

To check status of Jenkins

```
$ sudo systemctl status jenkins
```

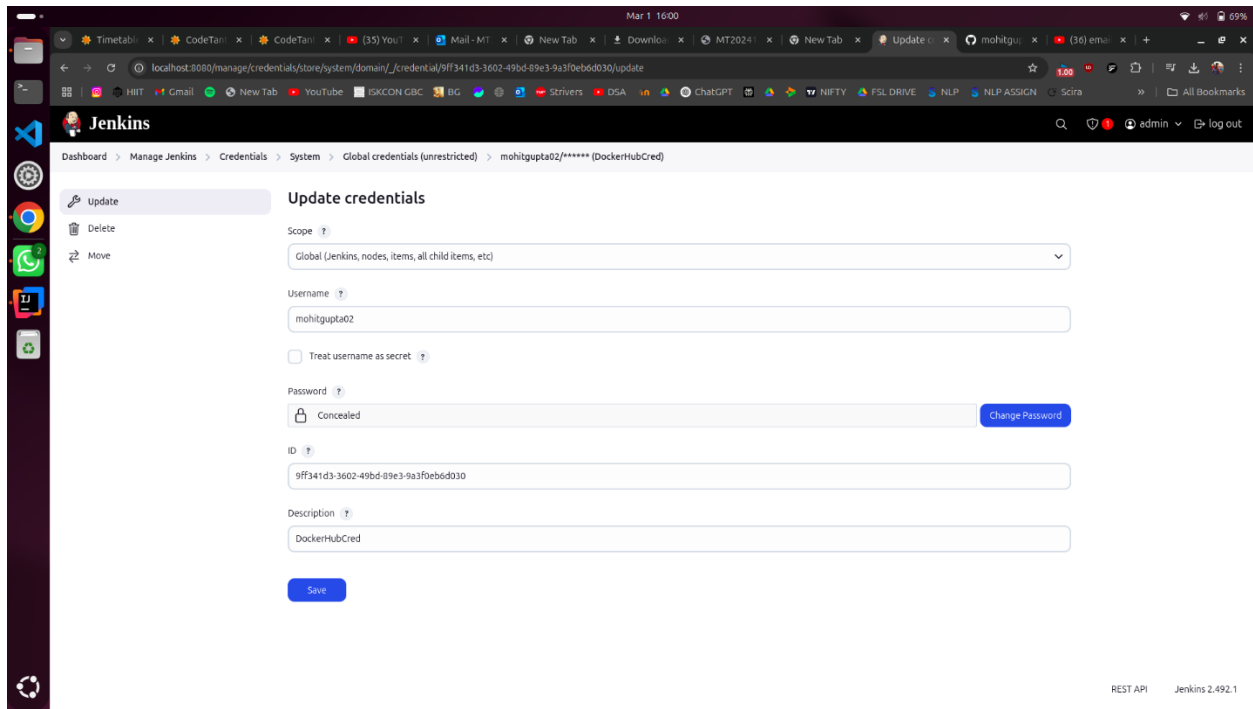




Configuration of Project

- In this step, we configure a GitHub project in Jenkins by providing the URL of our GitHub repository and specifying the path to the Jenkins file in the configuration.
- We then set up system credentials for both Docker Hub and the local machine for the Ansible playbook.
- Since the project runs on the local machine and pulls the image from Docker Hub, it's necessary to define these system configurations in Jenkins' system credentials.
- Two credentials need to be added: one for the Docker Hub repository and another for the local machine.

To configure Docker Hub credentials, we must specify the Docker Hub username from which the image will be pulled.



The screenshot shows the Jenkins web interface in a browser. The breadcrumb navigation at the top reads: Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted) > mohitgupta02/***** (DockerHubCred). On the left sidebar, there are icons for Update, Delete, and Move. The main content area is titled 'Update credentials'. It contains the following fields and controls:

- Scope:** A dropdown menu with the selected value 'Global (Jenkins, nodes, items, all child items, etc)'.
- Username:** A text input field containing 'mohitgupta02'.
- Treat username as secret:** An unchecked checkbox.
- Password:** A text input field with a lock icon and the text 'Concealed'. To its right is a blue button labeled 'Change Password'.
- ID:** A text input field containing '9ff341d3-3602-49bd-89e3-9a3foebdd030'.
- Description:** A text input field containing 'DockerHubCred'.
- Save:** A blue button at the bottom of the form.

At the bottom right of the page, the text 'REST API Jenkins 2.492.1' is visible.

Adding DockerHub Credentials

Ansible:

- Ansible is a continuous deployment tool. In which there is one control node and the nodes on which the deliverables are deployed are called managed nodes.
- Ansible need to be installed only on the control node.
- Ansible connects with the managed nodes using ssh.
- In Ansible we provide a set of commands that need to be executed on the managed node in the form of yml file.
- It is called a playbook. Along with the playbook we also provide information about the managed nodes. Which called inventory.

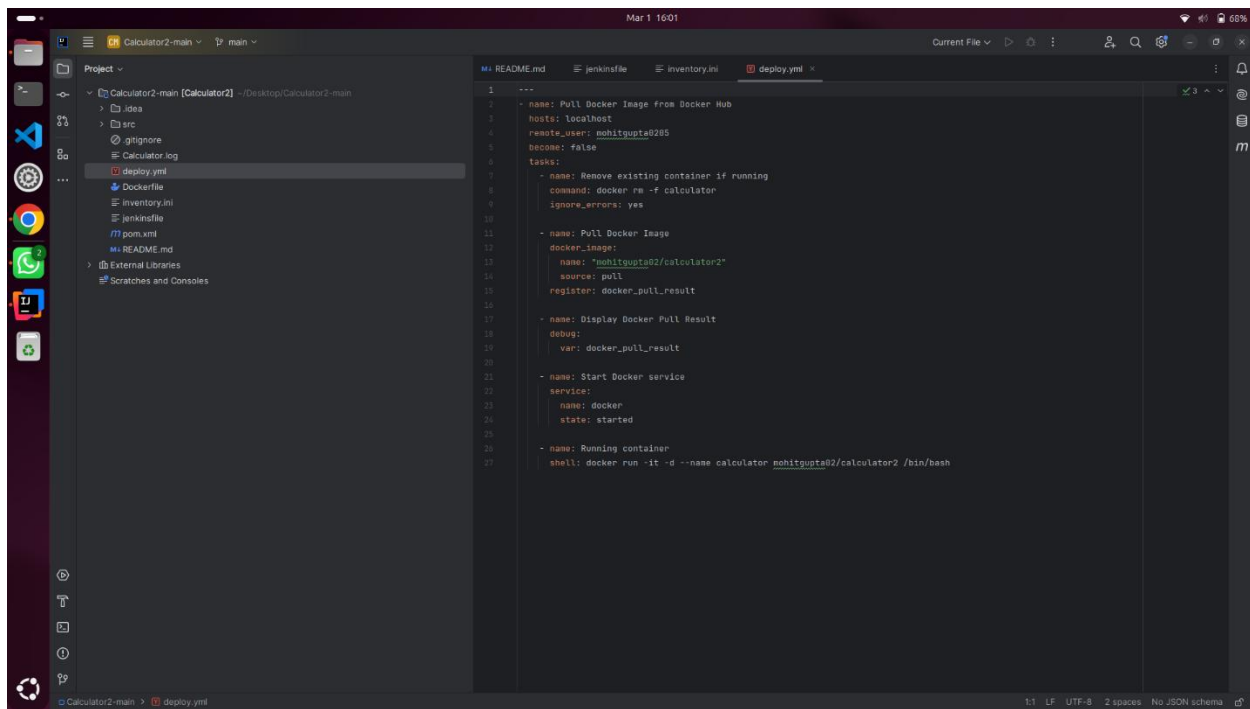
To execute this playbook we run the following command:

\$ ansible-playbook <playbook> -i <inventory>

```
1 [localhost]
2 localhost ansible_python_interpreter=/usr/bin/python3
3 localhost ansible_connection=ssh ansible_ssh_user=jenkins ansible_ssh_private_key_file=~/ssh/id_rsa
```

Inventory File

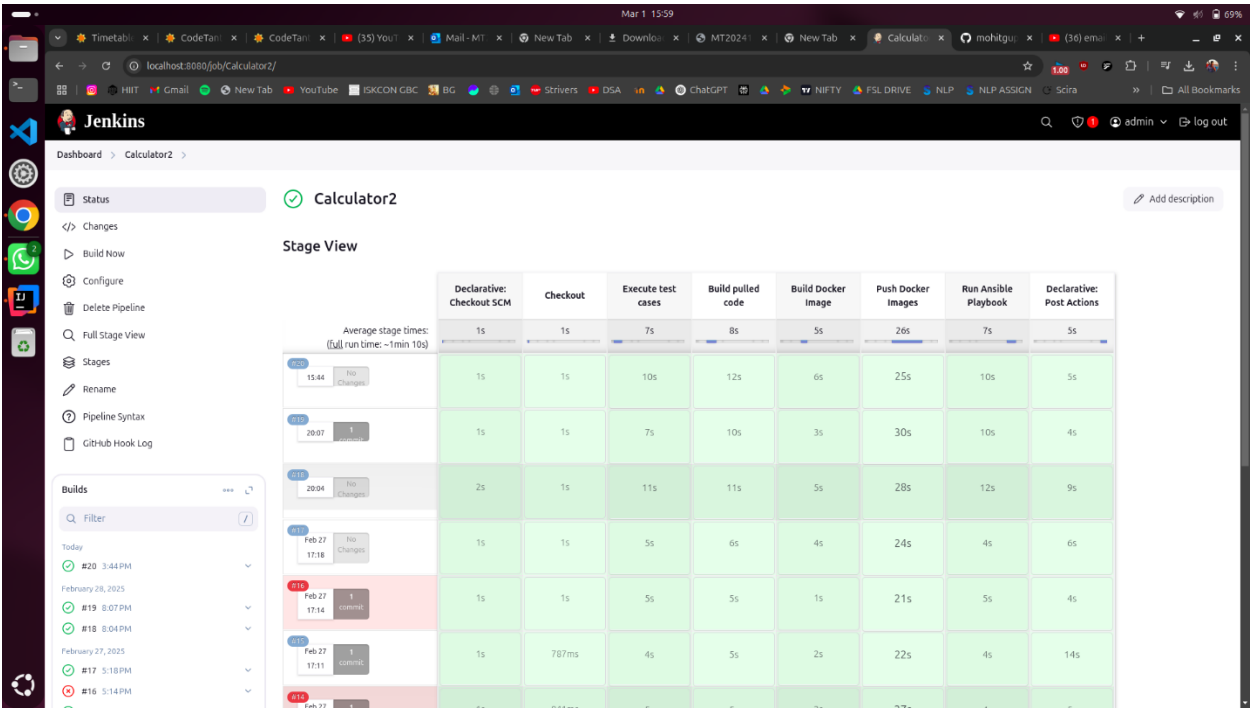
- The Ansible inventory file specifies the hosts and host groups on which tasks, modules, and commands within a playbook are executed.
- The format of the inventory file can vary based on the Ansible environment and the plugins in use.
- Ansible playbooks are used to document and carry out configuration, deployment, and orchestration tasks.
- To run the application's Docker image on the host, Python's pip and Docker must be installed on the host machine.
- The Ansible playbook pulls the Docker image from Docker Hub and launches the container on the specified hosts.
- This entire process is automated through the playbook.



Playbook file

Run Project as Pipeline in Jenkins

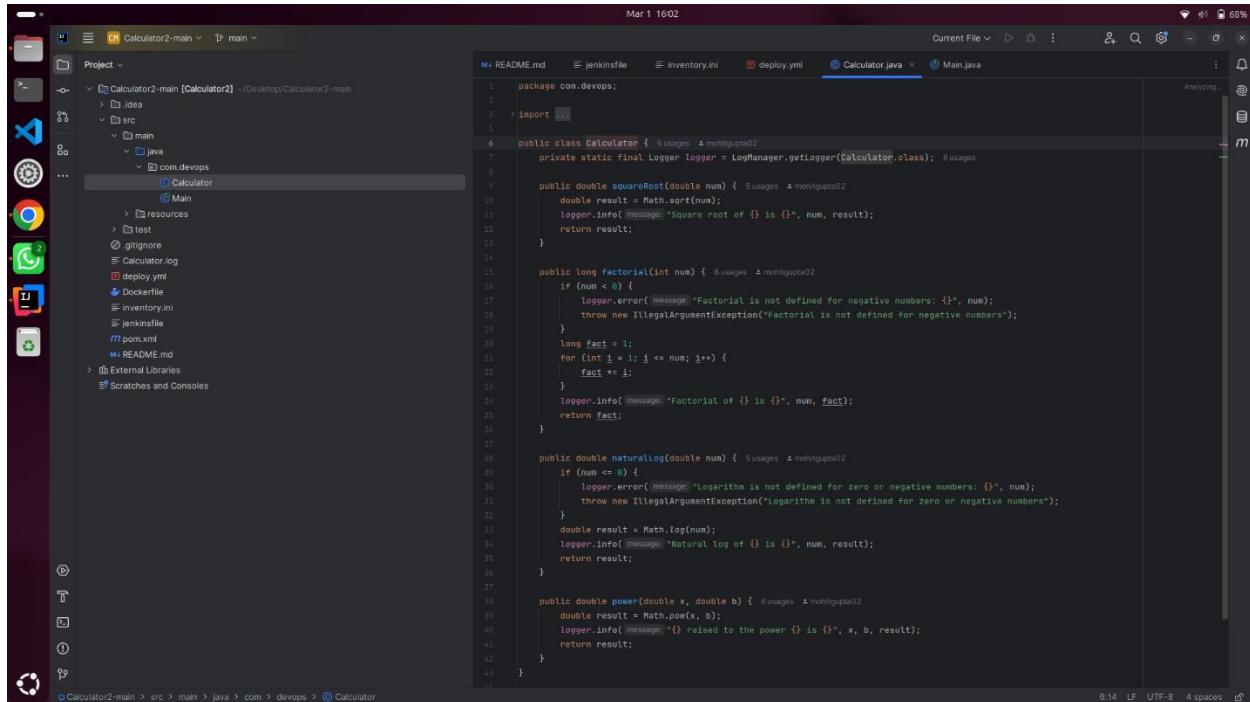
To run the project as pipeline in Jenkins we have to click build now to see each stages of pipeline and show the output of each stages.



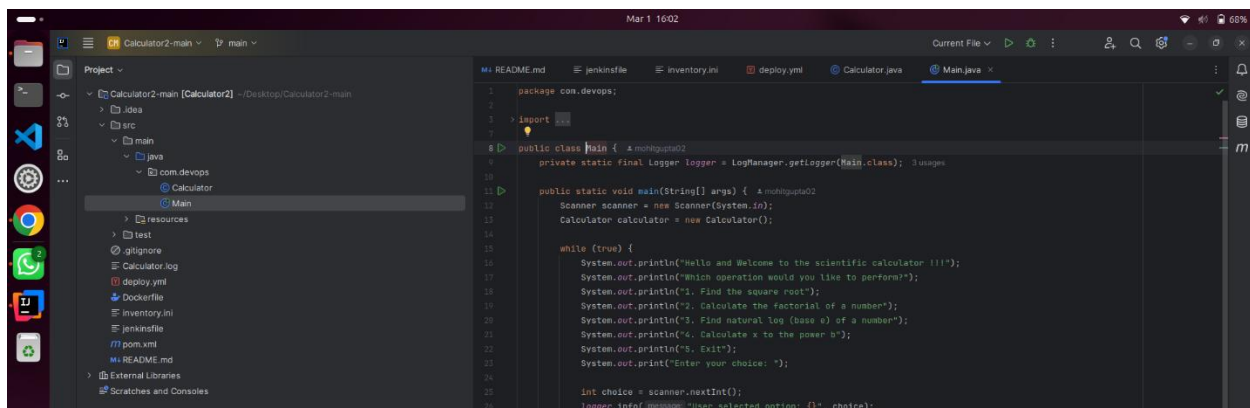
View of Pipeline Stages

Scientific Calculator Source Code:

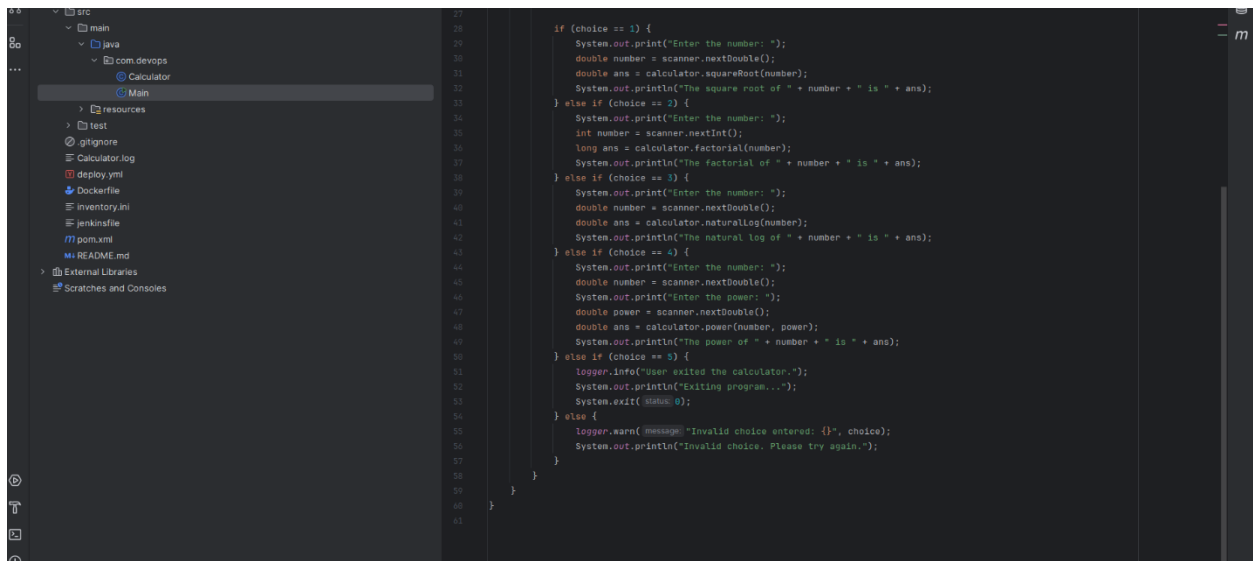
Below attached the source code of scientific calculator in java



```
1 package com.devops;
2
3 import java.util.*;
4
5 public class Calculator { @SuppressWarnings("unused")
6
7     private static final Logger logger = LogManager.getLogger(Calculator.class);
8
9     public double squareRoot(double num) {
10         double result = Math.sqrt(num);
11         logger.info("Square root of {} is {}", num, result);
12         return result;
13     }
14
15     public long factorial(int num) {
16         if (num < 0) {
17             logger.error("Factorial is not defined for negative numbers: {}", num);
18             throw new IllegalArgumentException("Factorial is not defined for negative numbers");
19         }
20         long fact = 1;
21         for (int i = 1; i <= num; i++) {
22             fact *= i;
23         }
24         logger.info("Factorial of {} is {}", num, fact);
25         return fact;
26     }
27
28     public double naturalLog(double num) {
29         if (num <= 0) {
30             logger.error("Logarithm is not defined for zero or negative numbers: {}", num);
31             throw new IllegalArgumentException("Logarithm is not defined for zero or negative numbers");
32         }
33         double result = Math.log(num);
34         logger.info("Natural log of {} is {}", num, result);
35         return result;
36     }
37
38     public double power(double x, double b) {
39         double result = Math.pow(x, b);
40         logger.info("{} raised to the power {} is {}", x, b, result);
41         return result;
42     }
43 }
```



```
1 package com.devops;
2
3 import java.util.*;
4
5 public class Main { @SuppressWarnings("unused")
6
7     private static final Logger logger = LogManager.getLogger(Main.class);
8
9     public static void main(String[] args) {
10         Scanner scanner = new Scanner(System.in);
11         Calculator calculator = new Calculator();
12
13         while (true) {
14             System.out.println("Hello and Welcome to the scientific calculator !!!");
15             System.out.println("Which operation would you like to perform?");
16             System.out.println("1. Find the square root");
17             System.out.println("2. Calculate the factorial of a number");
18             System.out.println("3. Find natural log (base e) of a number");
19             System.out.println("4. Calculate x to the power b");
20             System.out.println("5. Exit");
21             System.out.print("Enter your choice: ");
22
23             int choice = scanner.nextInt();
24             logger.info("User selected option: {}", choice);
25         }
26     }
27 }
```

The screenshot shows an IDE with a project structure on the left and a code editor on the right. The project structure includes a 'src' directory with 'main' and 'resources' subdirectories. The 'main' directory contains a 'java' package with a 'com.devops' package, which in turn contains a 'Calculator' class. The 'resources' directory contains a 'test' directory. The code editor shows the 'Calculator' class with a 'main' method that implements a menu-driven calculator. The code includes logic for calculating the square root, factorial, natural log, and power of a number, as well as handling invalid choices and exiting the program.

```
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61

if (choice == 1) {
    System.out.print("Enter the number: ");
    double number = scanner.nextDouble();
    double ans = calculator.squareRoot(number);
    System.out.println("The square root of " + number + " is " + ans);
} else if (choice == 2) {
    System.out.print("Enter the number: ");
    int number = scanner.nextInt();
    long ans = calculator.factorial(number);
    System.out.println("The factorial of " + number + " is " + ans);
} else if (choice == 3) {
    System.out.print("Enter the number: ");
    double number = scanner.nextDouble();
    double ans = calculator.naturalLog(number);
    System.out.println("The natural log of " + number + " is " + ans);
} else if (choice == 4) {
    System.out.print("Enter the number: ");
    double number = scanner.nextDouble();
    System.out.print("Enter the power: ");
    double power = scanner.nextDouble();
    double ans = calculator.power(number, power);
    System.out.println("The power of " + number + " is " + ans);
} else if (choice == 5) {
    logger.info("User exited the calculator.");
    System.out.println("Exiting program...");
    System.exit(0);
} else {
    logger.warn(message "Invalid choice entered: {}", choice);
    System.out.println("Invalid choice. Please try again.");
}
}
```

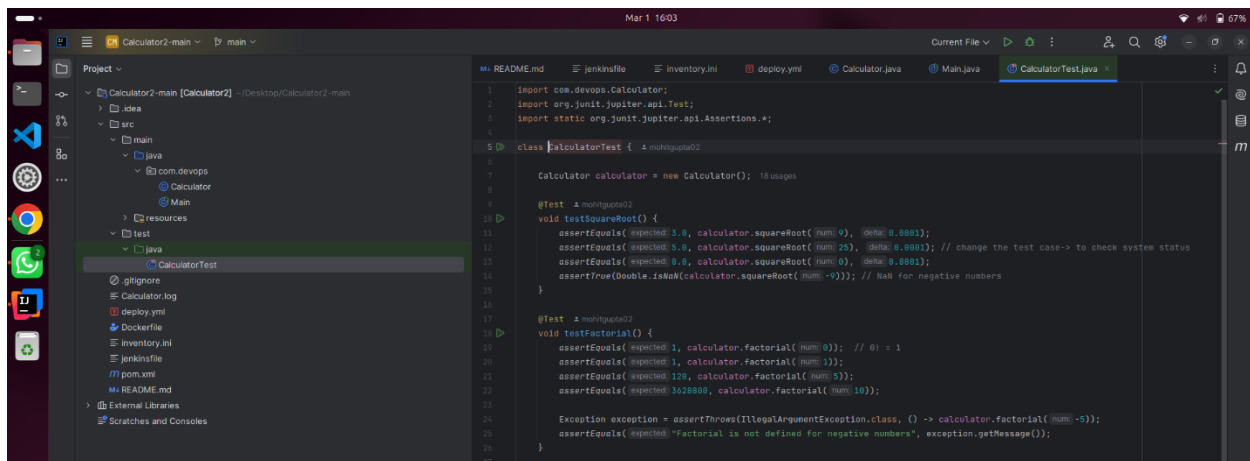
Adding Test Cases:

What is Unit Testing?

- Unit Testing is the process of testing individual components of code in isolation.
- A "unit" refers to the smallest functional part of a system, which could be a single line of code, a method, or an entire class.
- The smaller the unit being tested, the faster and more efficient the process becomes.
- This helps in gaining deeper insights into the code's functionality and overall performance.

JUnit: A Unit Testing Framework for Java

- JUnit is an open-source framework designed for unit testing in Java.
- It allows developers to write and execute automated test cases efficiently.
- In Java development, test cases must be rerun whenever new code is introduced to ensure that existing functionality remains intact.
- JUnit also provides visual feedback on test execution through graphical indicators.
- A successful test is shown in green, while a failed test appears in red.
- By leveraging JUnit, developers can create robust, error-free code with greater reliability.



```

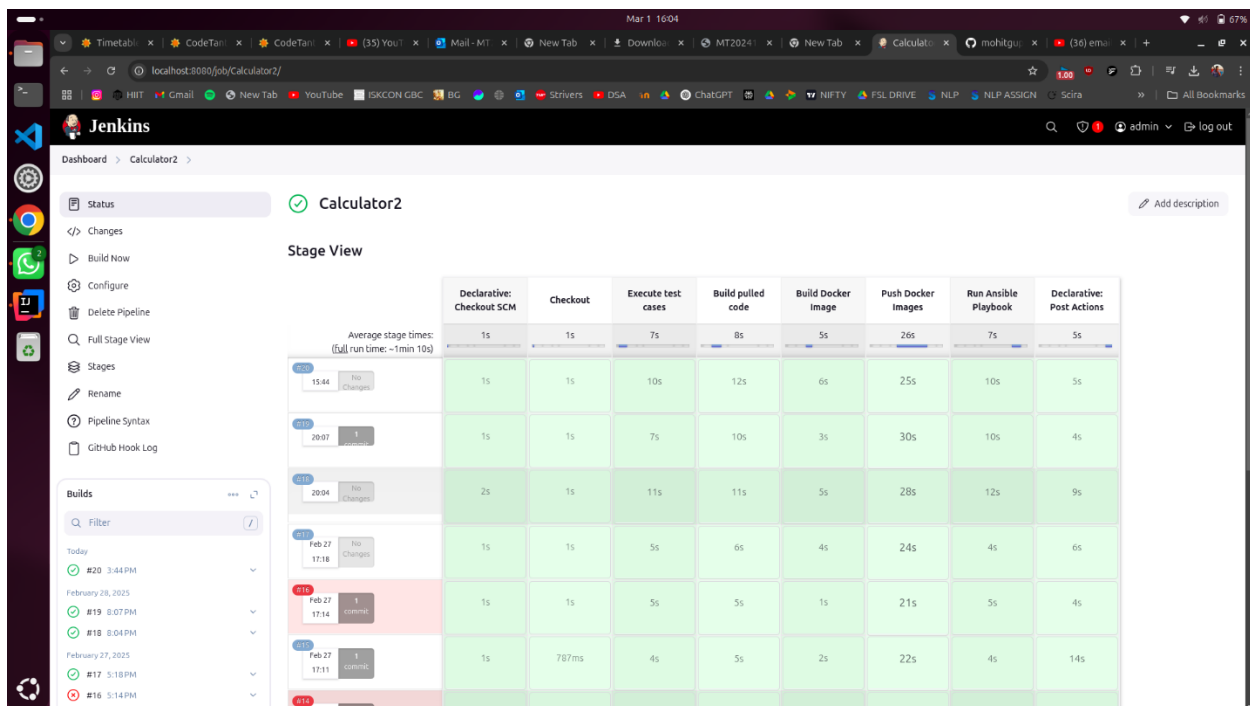
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

```

Adding Test Cases Java File

Run Project:

To run project first run pipeline in Jenkins



Pipeline View

Webhooks:

Webhooks are automated HTTP callbacks that trigger actions when an event occurs, such as a code push in a repository.

Use of Webhooks in Jenkins with Ngrok

Ngrok is used to expose Jenkins running on a local machine to the internet, enabling webhooks to communicate with it. This allows GitHub/GitLab webhooks to trigger Jenkins builds even when running on a local server.

Key Benefits:

- **Automatic Build Triggers:** Webhooks notify Jenkins of code changes.
- **Remote Accessibility:** Ngrok provides a public URL for Jenkins, making it accessible for webhook events.
- **Faster CI/CD:** Automates build and deployment without manual intervention.

Collaborators

Moderation options

Code and automation

Branches

Tags

Rules

Actions

Webhooks

Environments

Codespaces

Pages

Security

Code security

Deploy keys

Secrets and variables

Integrations

GitHub Apps

Email notifications

Settings Recent Deliveries

We'll send a POST request to the URL below with details of any subscribed events. You can also specify which data format you'd like to receive (JSON, `x-www-form-urlencoded`, etc). More information can be found in [our developer documentation](#).

Payload URL *

`https://9638-119-161-98-68.ngrok-free.app/github-webhook/`

Content type *

`application/x-www-form-urlencoded`

There is currently a secret configured for this webhook. If you've lost or forgotten this secret, you can change it, but be aware that any integrations using this secret will need to be updated.

[Change secret](#)

SSL verification

By default, we verify SSL certificates when delivering payloads.

☒ **Enable SSL verification** ☐ **Disable (not recommended)**

Which events would you like to trigger this webhook?

☒ **Just the push event.**

☐ **Send me everything.**

☐ **Let me select individual events.**

Webhooks Using Through the Git-hub

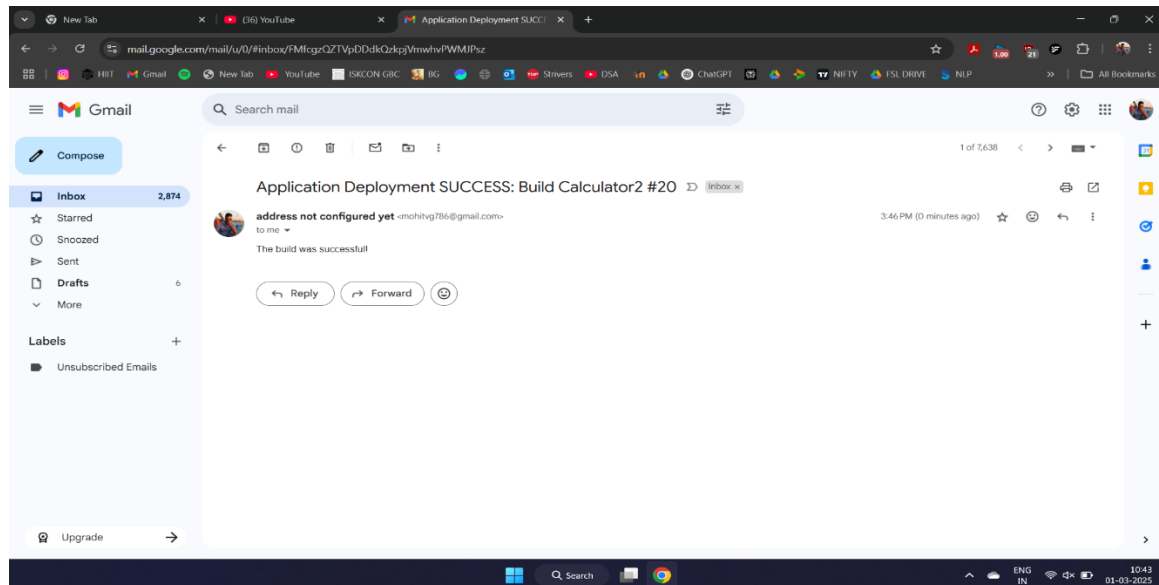
Email Notification In Jenkins Pipeline project:

What is SMTP in Jenkins?

SMTP (Simple Mail Transfer Protocol) allows Jenkins to send email notifications about build status, failures, or successes to developers and stakeholders. This ensures timely updates on project progress.

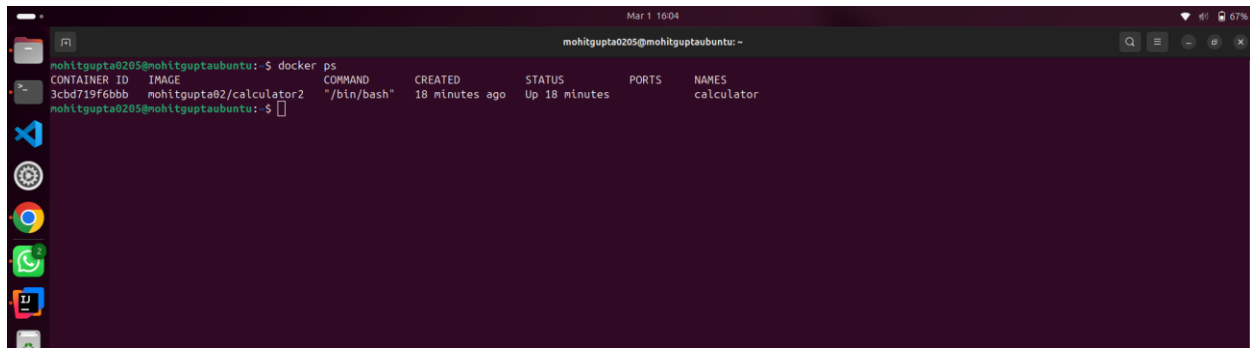
Use of SMTP in Jenkins

- Automated Email Alerts: Sends build status updates after each pipeline run.
- Failure Notifications: Alerts developers immediately if a build fails.
- Custom Messages: Can include logs, reports, and issue details in emails.



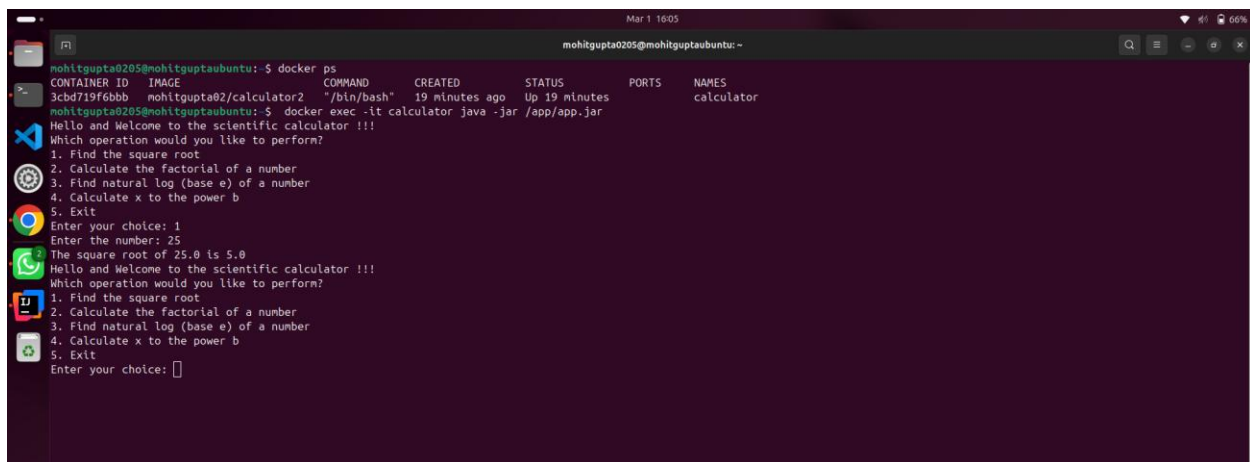
Email Photo Send by Jenkins after Deployment

Project Runs successfully and now our docker image is also pulled from docker hub. We Can run in local machine to check whether it runs or not by using below command.

A terminal window titled 'mohitgupta0205@mohitguptaubuntu: -' showing the output of the 'docker ps' command. The output is a table with columns: CONTAINER ID, IMAGE, COMMAND, CREATED, STATUS, PORTS, and NAMES. One container is listed with ID '3cbd719f6bbb', image 'mohitgupta02/calculator2', command '/bin/bash', created '18 minutes ago', status 'Up 18 minutes', and name 'calculator'.

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
3cbd719f6bbb	mohitgupta02/calculator2	"/bin/bash"	18 minutes ago	Up 18 minutes		calculator

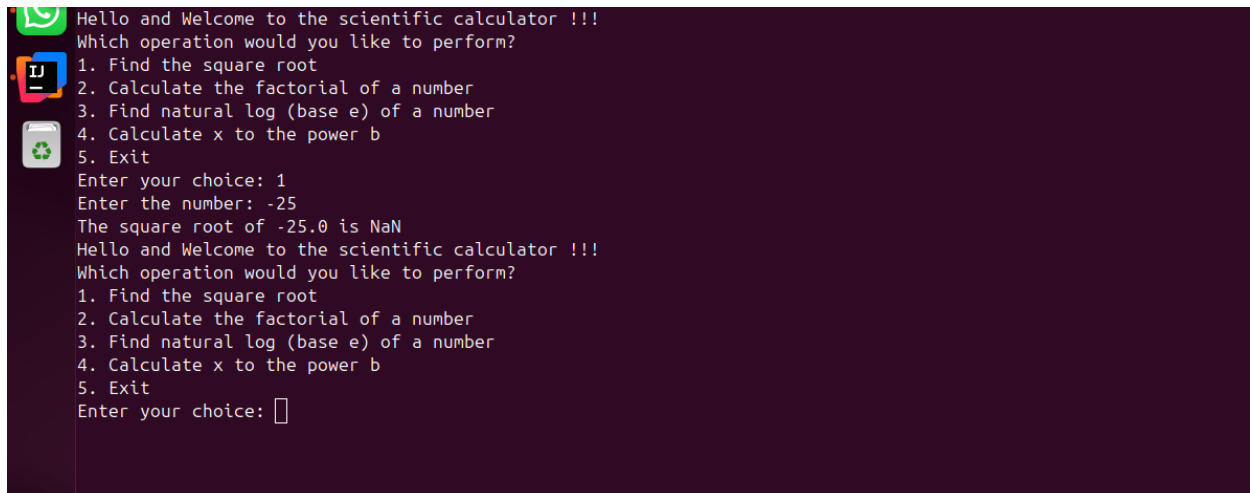
So, right now our container is running and to execute this code we are using below command to enter the container and check whether our code is working or not.

A terminal window titled 'mohitgupta0205@mohitguptaubuntu: -' showing the execution of a Docker container. The user runs 'docker exec -it calculator java -jar /app/app.jar'. The output shows a welcome message and a menu of operations. The user enters '1' for 'Find the square root' and '25' for the number. The output is 'The square root of 25.0 is 5.0'.

```
mohitgupta0205@mohitguptaubuntu:~$ docker exec -it calculator java -jar /app/app.jar
Hello and Welcome to the scientific calculator !!!
Which operation would you like to perform?
1. Find the square root
2. Calculate the factorial of a number
3. Find natural log (base e) of a number
4. Calculate x to the power b
5. Exit
Enter your choice: 1
Enter the number: 25
The square root of 25.0 is 5.0
Hello and Welcome to the scientific calculator !!!
Which operation would you like to perform?
1. Find the square root
2. Calculate the factorial of a number
3. Find natural log (base e) of a number
4. Calculate x to the power b
5. Exit
Enter your choice: 
```

Execute Container

Check for Some Test Cases:

A terminal window with a dark purple background and light green text. On the left side, there are three icons: a green speech bubble, a blue square with a white 'I', and a green recycling symbol. The text in the terminal reads:

```
Hello and Welcome to the scientific calculator !!!  
Which operation would you like to perform?  
1. Find the square root  
2. Calculate the factorial of a number  
3. Find natural log (base e) of a number  
4. Calculate x to the power b  
5. Exit  
Enter your choice: 1  
Enter the number: -25  
The square root of -25.0 is NaN  
Hello and Welcome to the scientific calculator !!!  
Which operation would you like to perform?  
1. Find the square root  
2. Calculate the factorial of a number  
3. Find natural log (base e) of a number  
4. Calculate x to the power b  
5. Exit  
Enter your choice: 
```

Checking for the Execution

GitHub Repo : <https://github.com/mohitgupta02/Calculator2>

Dockerhub Repo:

<https://hub.docker.com/repository/docker/mohitgupta02/calculator2/general>