1. Fork the Project https://github.com/adikanthik/Project

2. Follow the Instructions provided and setup E2E jenkins pipeline and run the application

3. Push the built images to docker hub

4. Run the containers with the new images

5. Application should be accessible on port 9095 over a Load balancer

**Step 1: Fork the GitHub Project**

git clone https://github.com/megha270992/Project-5.git

cd Project

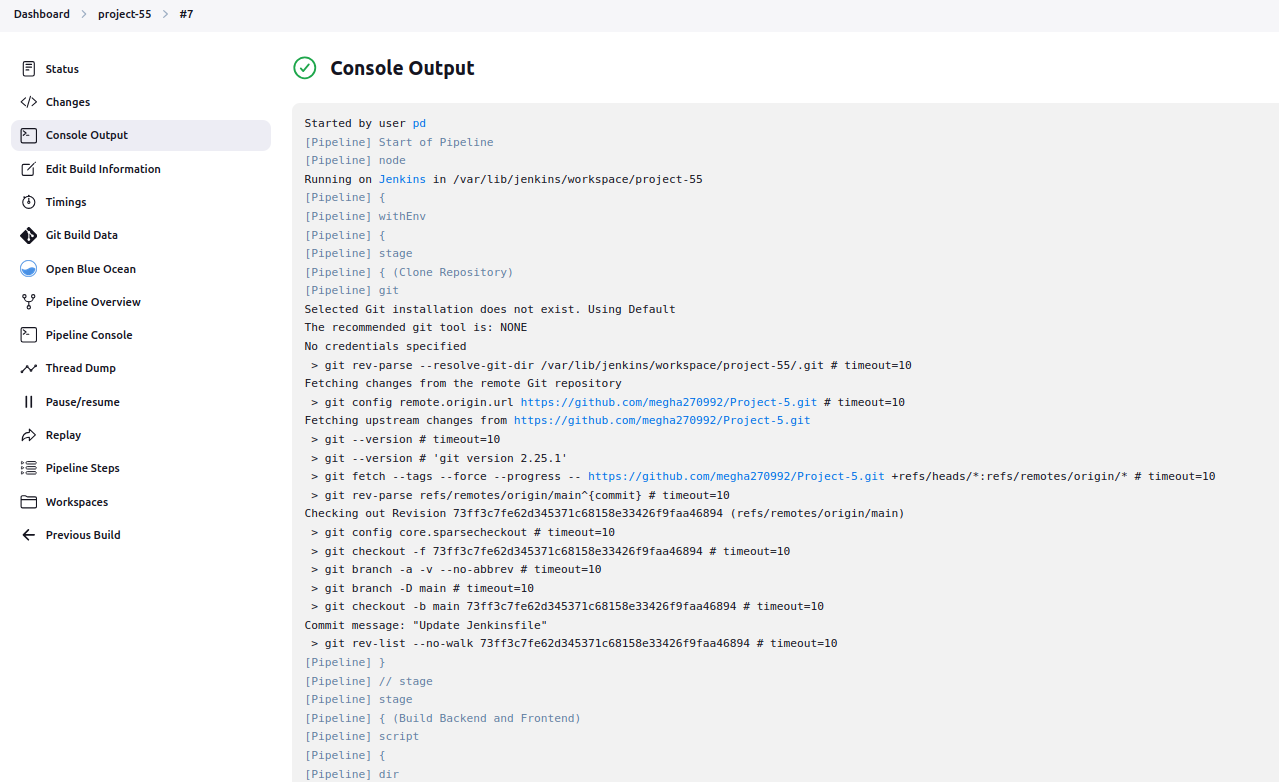
**Step 2: Set Up E2E Jenkins Pipeline**

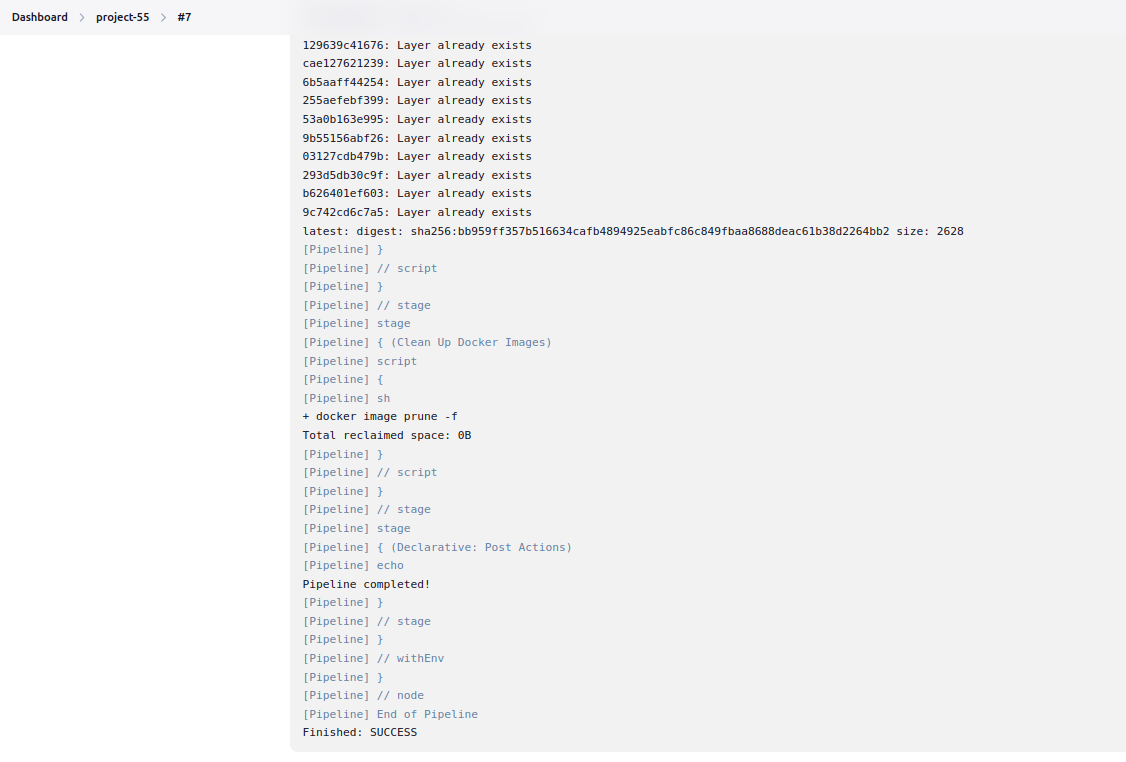
1. Installed Jenkins and required plugins (Pipeline, Docker, GitHub).
2. In Jenkins, created a new pipeline job and use the repository as the source.
3. Define a Jenkinsfile in the repository with steps like:

groovy

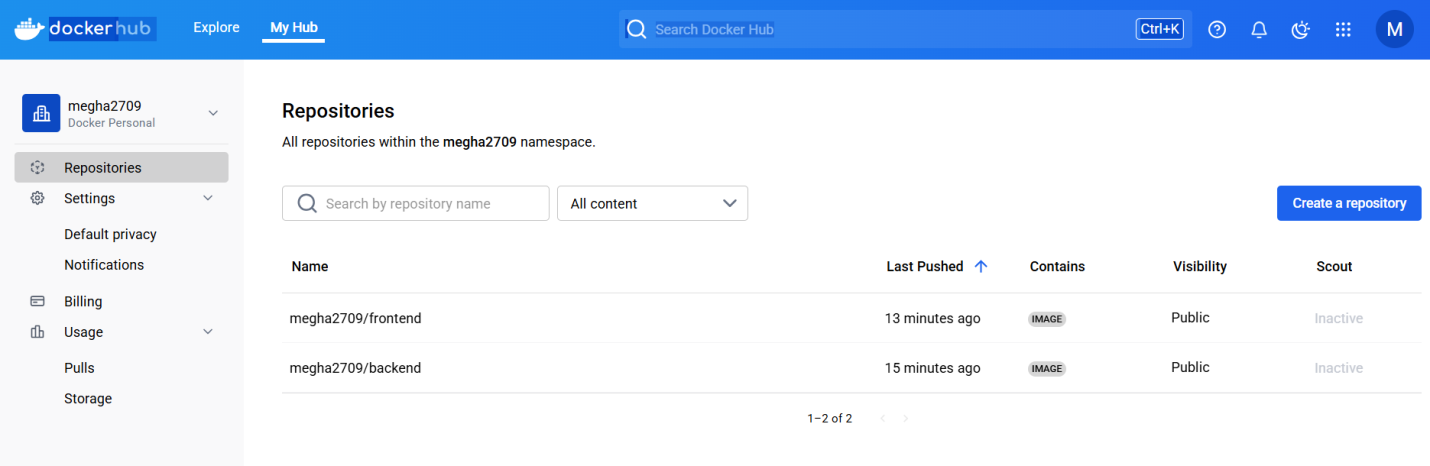
1. pipeline {
2. agent any
3. environment {
4. BACKEND\_PORT = '8085'
5. FRONTEND\_PORT = '8084'
6. DOCKER\_HUB\_REPO = 'megha2709'
7. BACKEND\_IMAGE = "${DOCKER\_HUB\_REPO}/backend"
8. FRONTEND\_IMAGE = "${DOCKER\_HUB\_REPO}/frontend"
9. IMAGE\_TAG = "${BUILD\_NUMBER}"
10. }
11. stages {
12. stage('Clone Repository') {
13. steps {
14. git branch: 'main', url: 'https://github.com/megha270992/Project-5.git'
15. }
16. }
17. stage('Build Backend and Frontend') {
18. steps {
19. script {
20. dir('backend') {
21. sh 'mvn clean install -DskipTests=true'
22. }
23. dir('frontend') {
24. sh 'mvn clean install -DskipTests=true'
25. }
26. }
27. }
28. }
29. stage('Package Applications') {
30. steps {
31. script {
32. sh 'ls backend/target'
33. sh 'ls frontend/target'
34. }
35. }
36. }
37. stage('Build Docker Images') {
38. steps {
39. script {
40. dir('backend') {
41. sh "docker build -t ${BACKEND\_IMAGE}:${IMAGE\_TAG} -t ${BACKEND\_IMAGE}:latest ."
42. }
43. dir('frontend') {
44. sh "docker build -t ${FRONTEND\_IMAGE}:${IMAGE\_TAG} -t ${FRONTEND\_IMAGE}:latest ."
45. }
46. }
47. }
48. }
49. stage('Docker Login') {
50. steps {
51. script {
52. withCredentials([usernamePassword(credentialsId: 'Dockerhub', usernameVariable: 'DOCKER\_USER', passwordVariable: 'DOCKER\_PASS')]) {
53. sh 'echo "$DOCKER\_PASS" | docker login -u "$DOCKER\_USER" --password-stdin'
54. }
55. }
56. }
57. }
58. stage('Push Docker Images to Docker Hub') {
59. steps {
60. script {
61. sh "docker push ${BACKEND\_IMAGE}:${IMAGE\_TAG}"
62. sh "docker push ${BACKEND\_IMAGE}:latest"
63. sh "docker push ${FRONTEND\_IMAGE}:${IMAGE\_TAG}"
64. sh "docker push ${FRONTEND\_IMAGE}:latest"
65. }
66. }
67. }
68. stage('Clean Up Docker Images') {
69. steps {
70. script {
71. sh 'docker image prune -f'
72. }
73. }
74. }
75. }
76. post {
77. always {
78. echo 'Pipeline completed!'
79. }
80. failure {
81. echo 'Pipeline failed.'
82. }
83. }
84. }

Saved the Jenkinsfile in the repo and configure Jenkins to trigger pipeline execution.

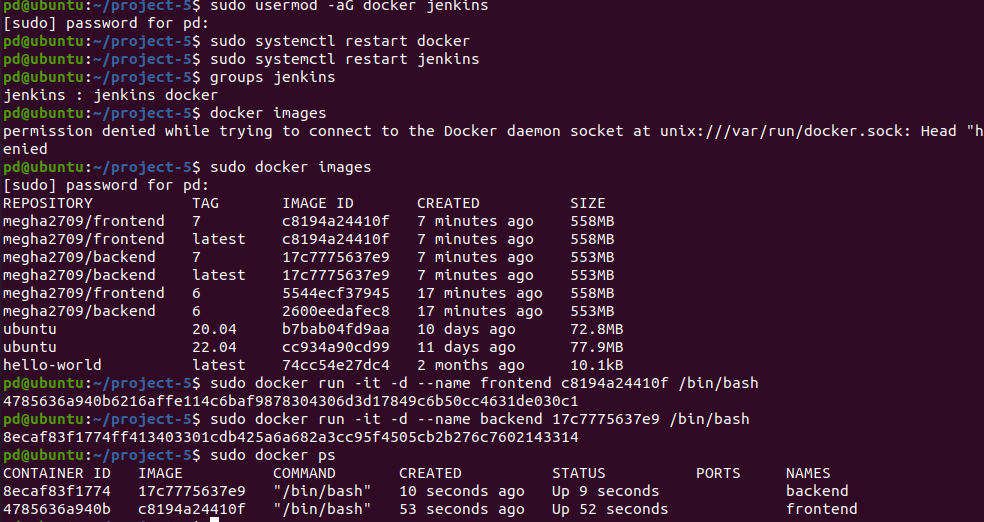




**Step 3: Push Built Images to Docker Hub**

****

**Step 4: Run Containers with New Images**



**Step 5: Configure Load Balancer**