LEARNOPS – Self Learning Challenge

Name: Geo Mathew George
Challenge Duration: 15 Days
Domain: DevOps
Tools Covered: Git & GitHub, Docker, CI/CD Concepts, Jenkins, Terraform

1. Introduction

The LEARNOPS – Self Learning Challenge gave me the opportunity to learn DevOps tools in a practical and realistic way by taking responsibility for my own learning. Instead of just following instructions, I explored documentation, experimented with tools, and learned through hands-on practice, which helped me understand how things work in real projects. This report shares my overall learning experience, the key concepts I picked up, how each tool is used in real DevOps workflows, and the personal growth I achieved through this self-driven journey.

For my studies, I relied on a mix of high-quality YouTube channels and well-structured online learning platforms that helped me understand concepts clearly and practically. YouTube channels such as **freeCodeCamp**, **Web Dev Simplified**, and **Mosh Hamedani** provided easy-to-follow tutorials covering web development, programming fundamentals, and real-world coding practices. For DevOps-specific learning, channels like **Abhishek Veeramalla** (DevOps Zero to Hero), **Edureka**, **Bret Fisher**, **TechWorld with Nana**, and **KodeKloud** helped me understand DevOps tools, workflows, and real industry scenarios through hands-on demonstrations.

Along with video learning, I frequently referred to trusted learning websites such as **W3Schools** and **GeeksforGeeks** for clear explanations, examples, and quick references on programming and DevOps concepts.

2. Objectives

- To learn how to study and work independently
- To get hands-on experience with DevOps tools
- To understand how DevOps tools work together
- To improve problem-solving skills

TOOLS

1)Git & Gitlab

TOPICS

- Installing and configuring Git
- Creating and managing repositories in GitLab
- Understanding the Git commit lifecycle and commit history
- Creating, switching, and managing branches
- Using merge requests and merging branches
- Cloning repositories and working locally
- Using common Git commands in daily development
- Understanding authentication methods such as HTTPS, SSH, and access tokens
- Collaborating with teams using GitLab features
- Using Git and GitLab in CI/CD and DevOps workflows

KEY CONCEPTS

Git

- Git is a distributed version control system used to track changes in files
- Each commit represents a snapshot of the project at a specific time
- Branches allow developers to work on different features independently
- Developers can experiment safely without affecting the main code
- Commands like add, commit, pull, push, and merge form the basic Git workflow
- Most Git operations are done locally, which makes development fast and reliable

GitLab

- GitLab is a centralized platform built on Git for team collaboration
- Repositories store shared source code and complete commit history
- Merge Requests allow code review, approval, and safe merging of changes

- Branch visibility helps teams manage multiple features at the same time
- GitLab supports authentication using HTTPS, SSH keys, and access tokens
- GitLab provides the base for creating and managing CI/CD pipelines

Practical Works Performed as Part of Git and GitLab Learning

The following practical tasks were performed to gain hands-on experience with Git and GitLab as part of the learning process.

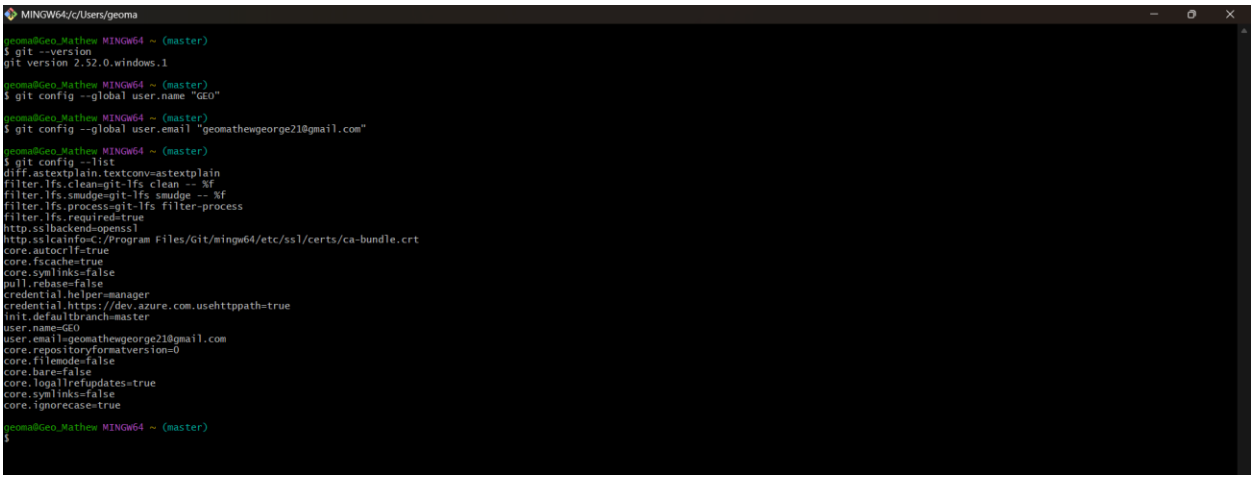
Tools Required

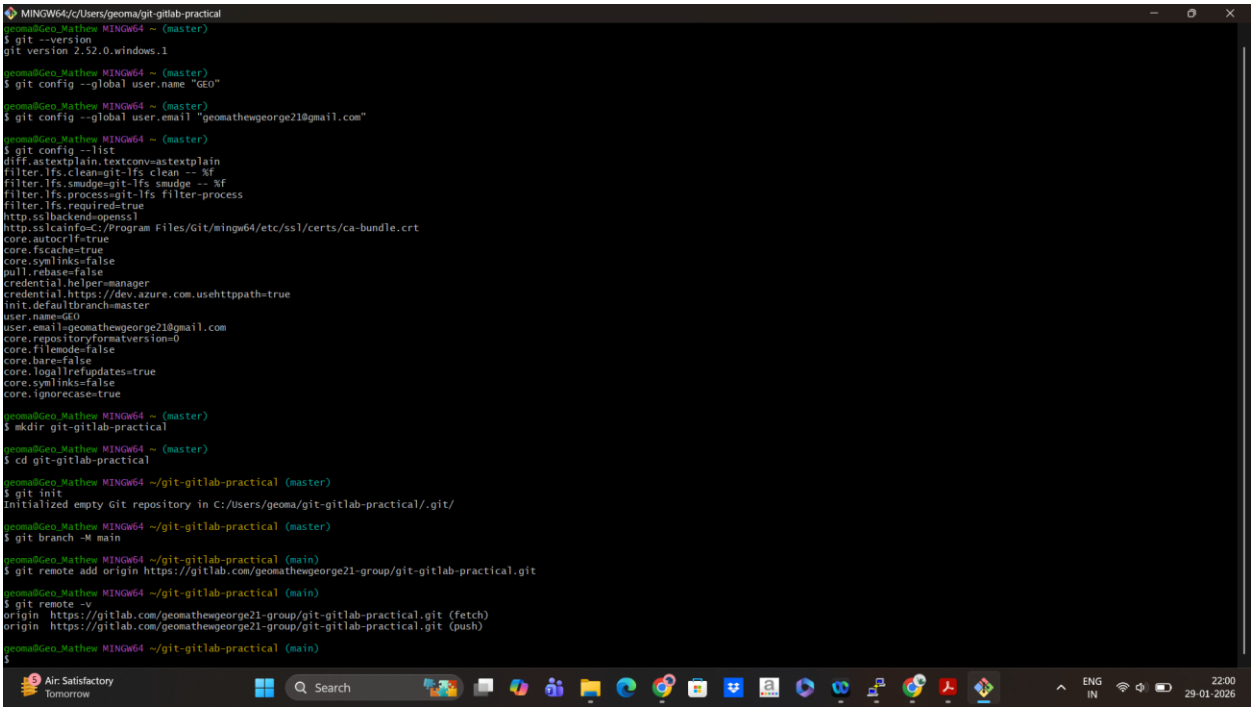
- Git
- GitLab account
- Git Bash or Linux Terminal
- Internet connection

Practical 1: Git Installation and Configuration

Steps performed

- Verified Git installation using the version command
- Configured global username and email
- Checked Git configuration settings





Outcome

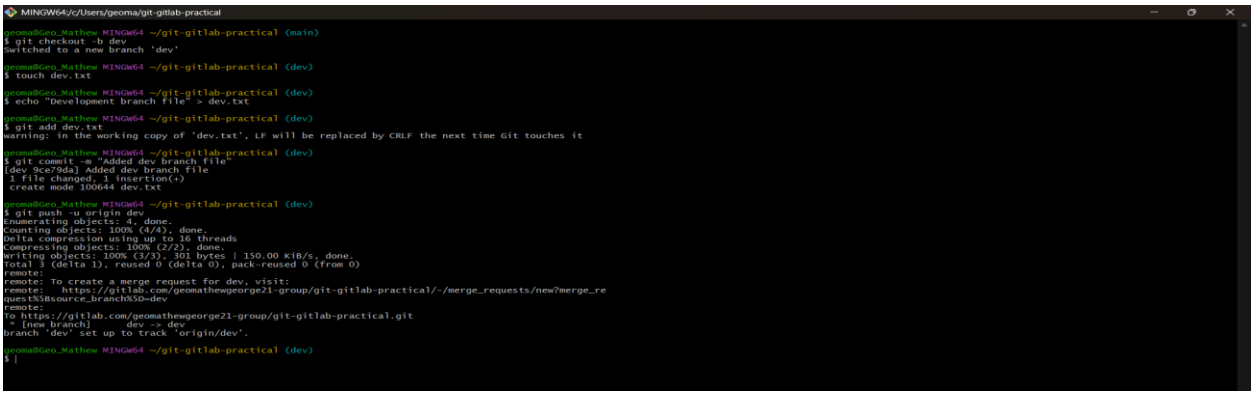
Git was successfully installed and user details were configured.

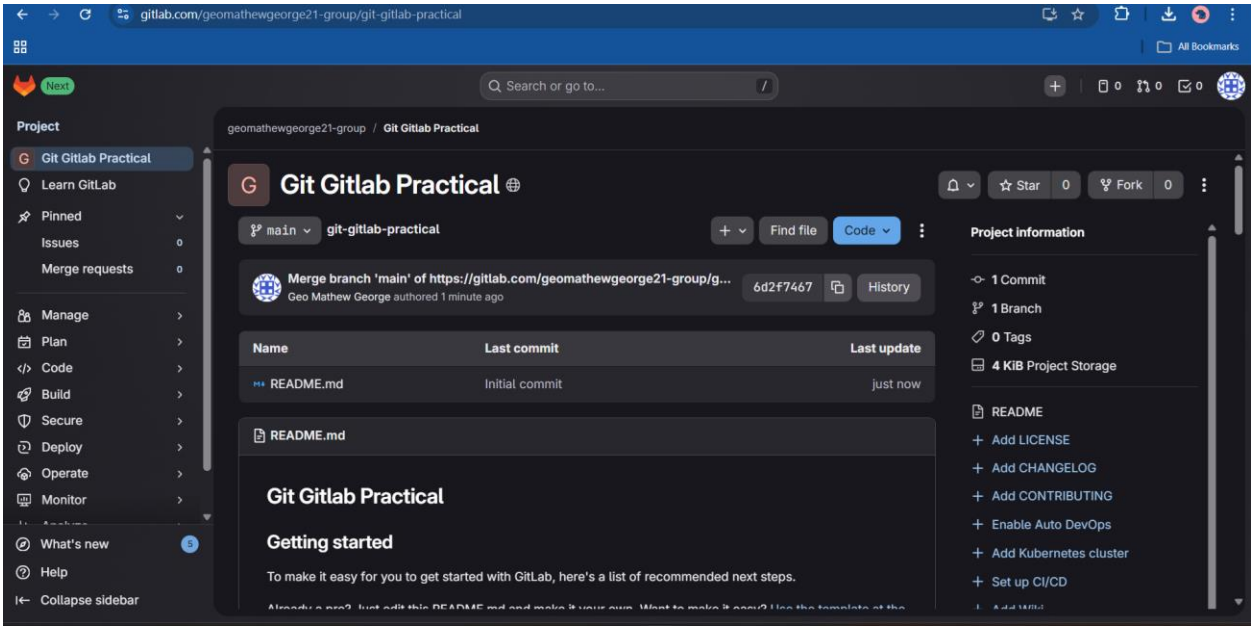
Practical 2: Create Repository in GitLab

Steps performed

- Logged into GitLab

- Created a new blank project
- Set repository name and visibility





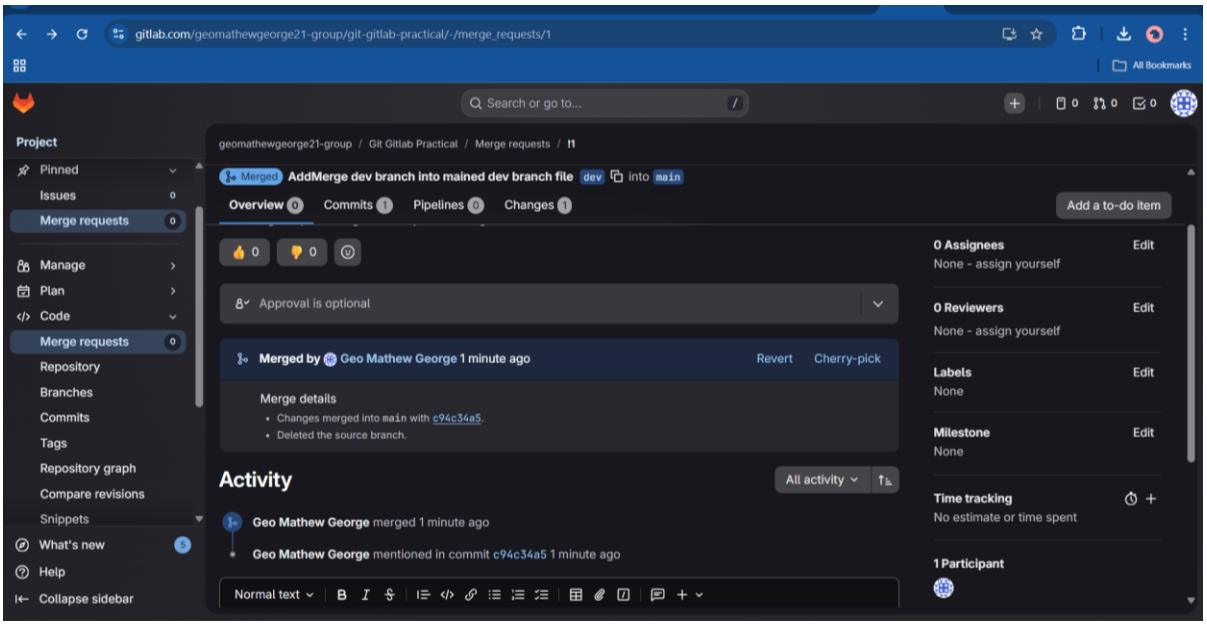
Outcome

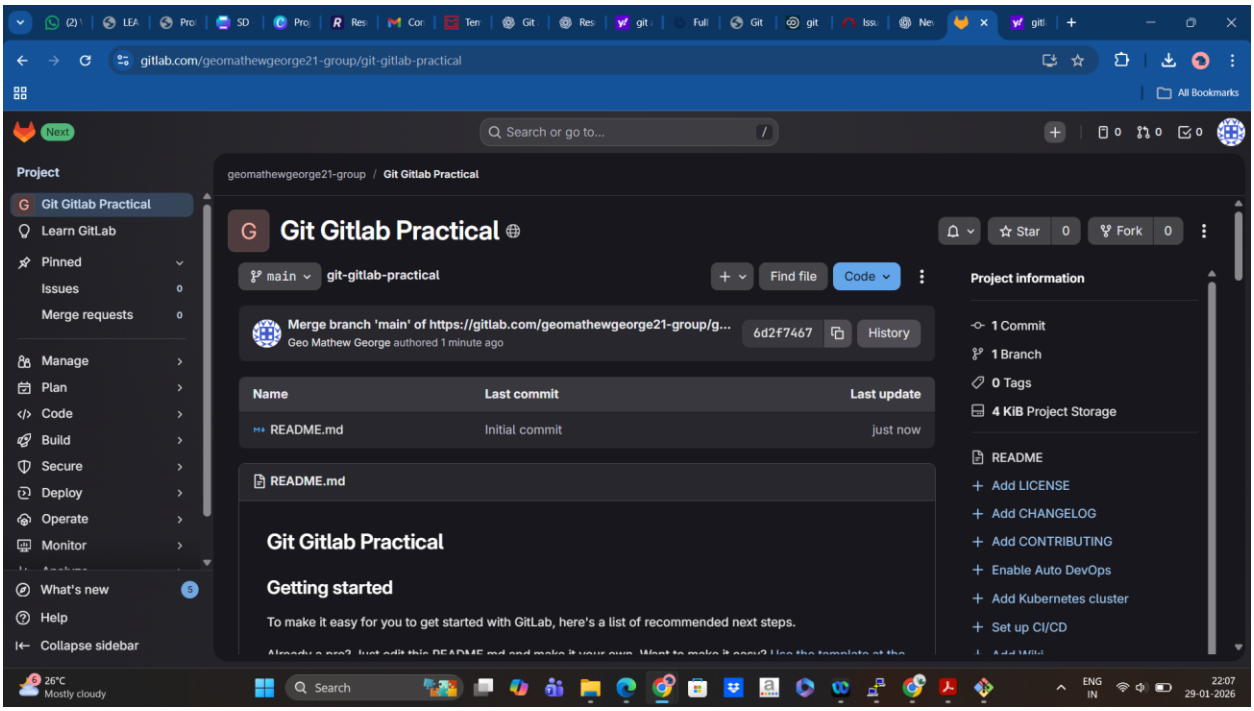
An empty GitLab repository was successfully created.

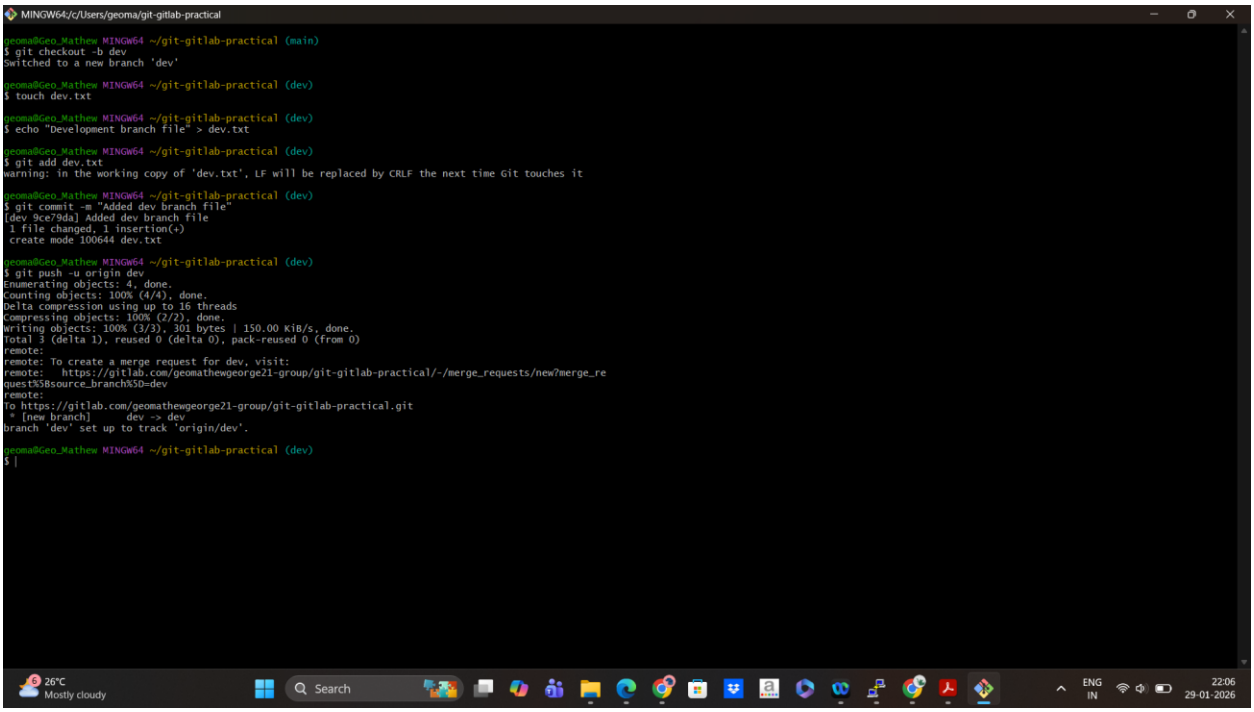
Practical 3: Initialize Local Git Repository

Steps performed

- Created a local project directory
- Initialized Git repository

- Set main branch
- Connected local repository to GitLab remote repository
- 





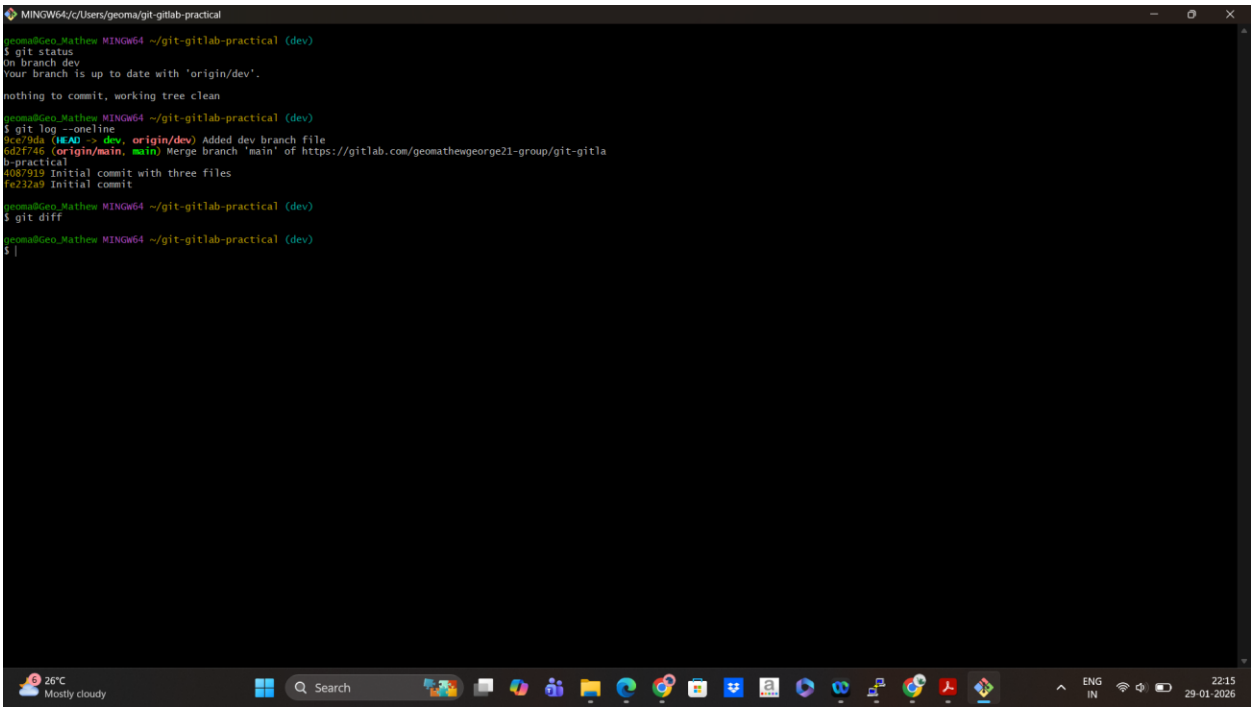
Outcome

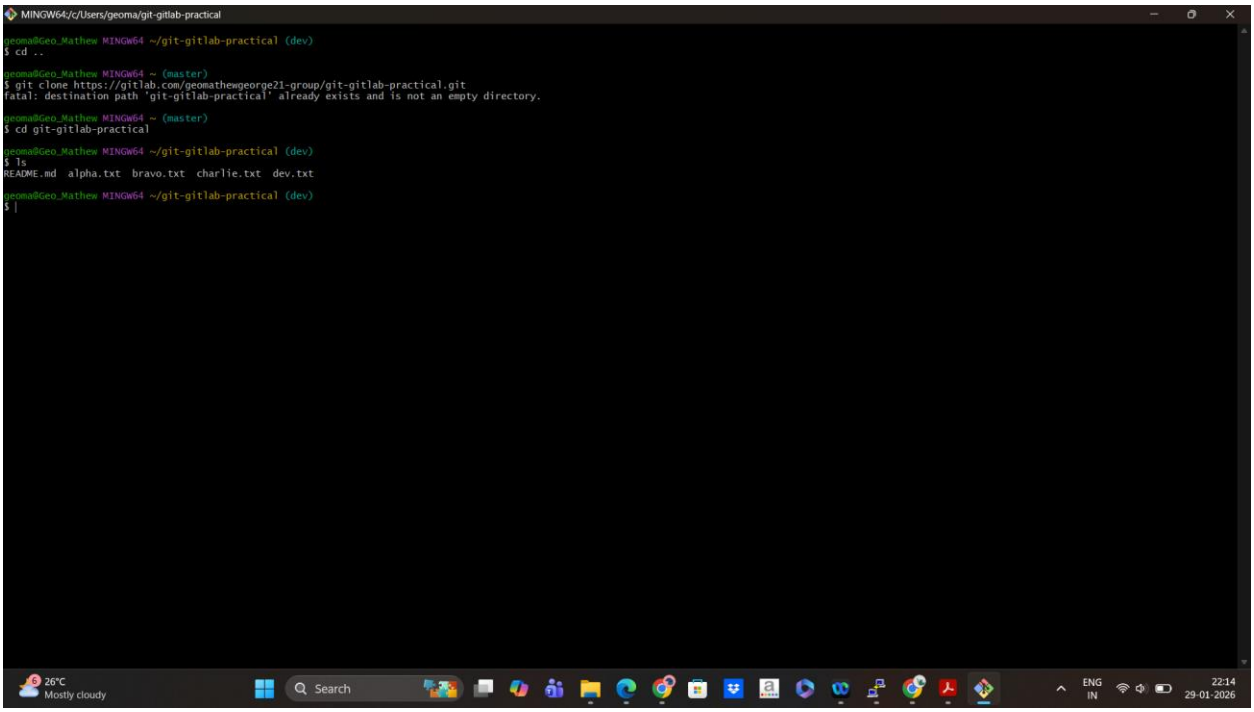
Local Git repository was successfully linked with GitLab.

Practical 4: Add Files and Create Commits

Steps performed

- Created multiple text files
- Added content to files
- Added files to staging area
- Committed changes
- Pushed commits to GitLab





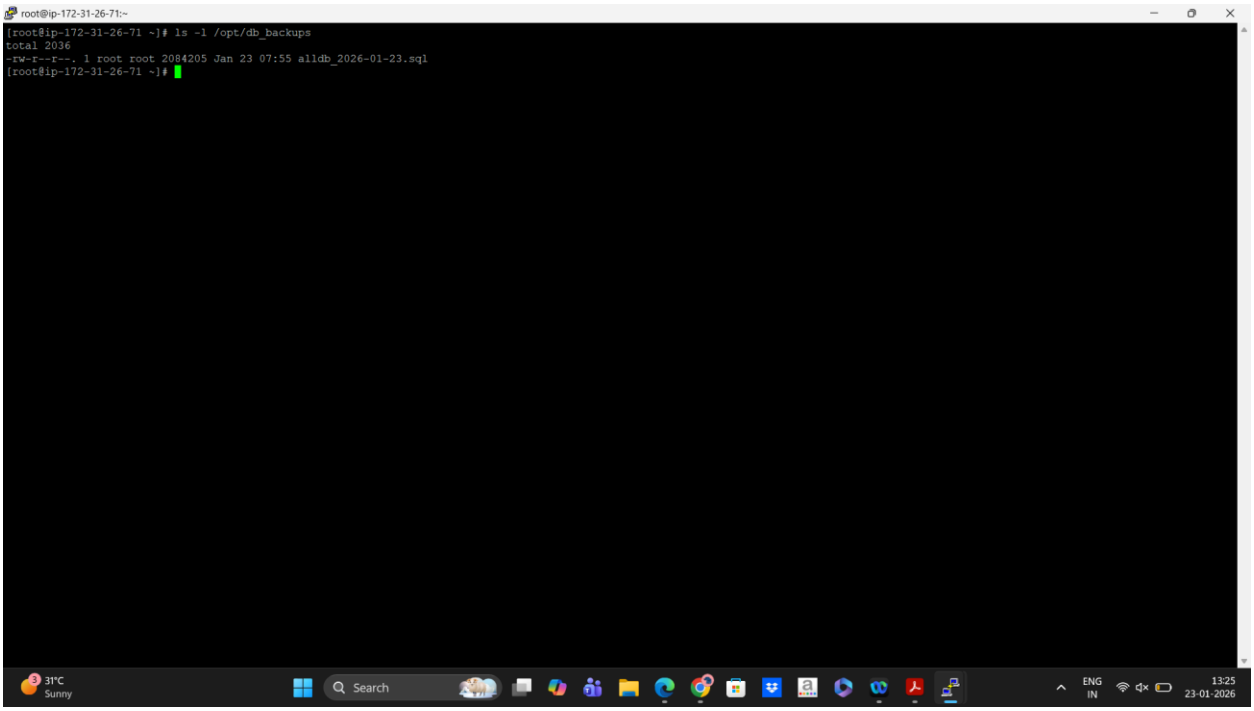
Outcome

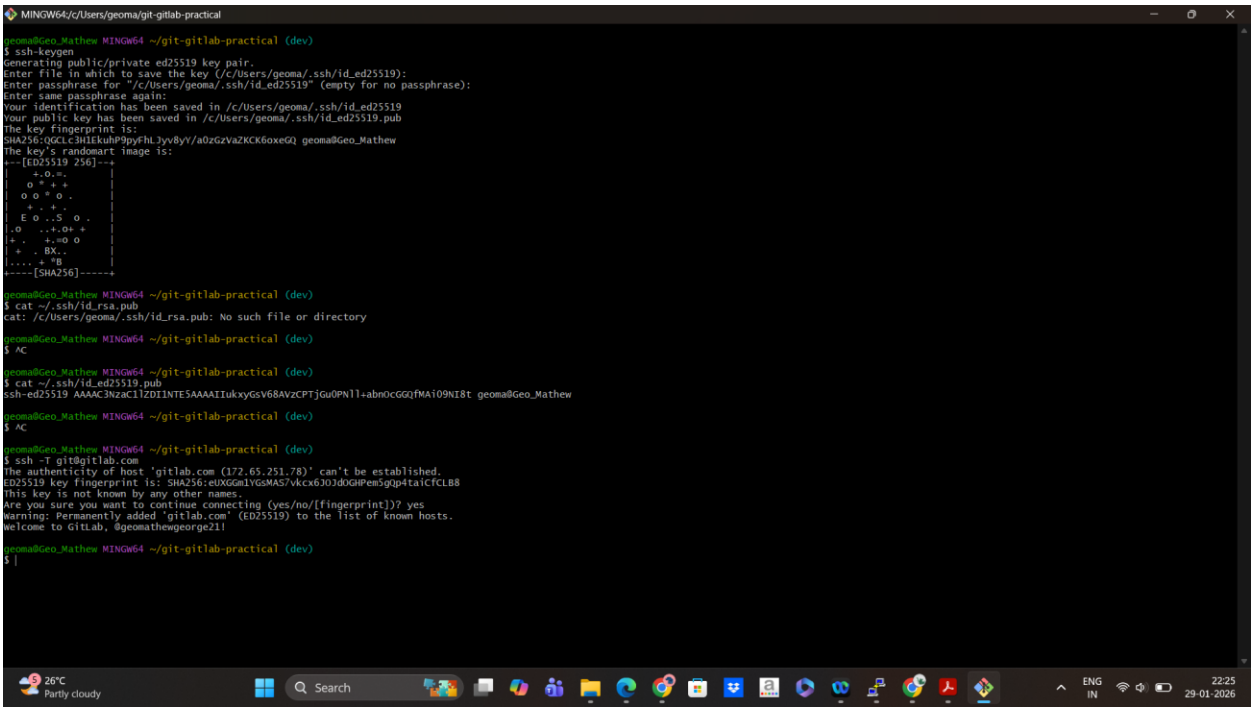
Files were uploaded to GitLab with proper commit history.

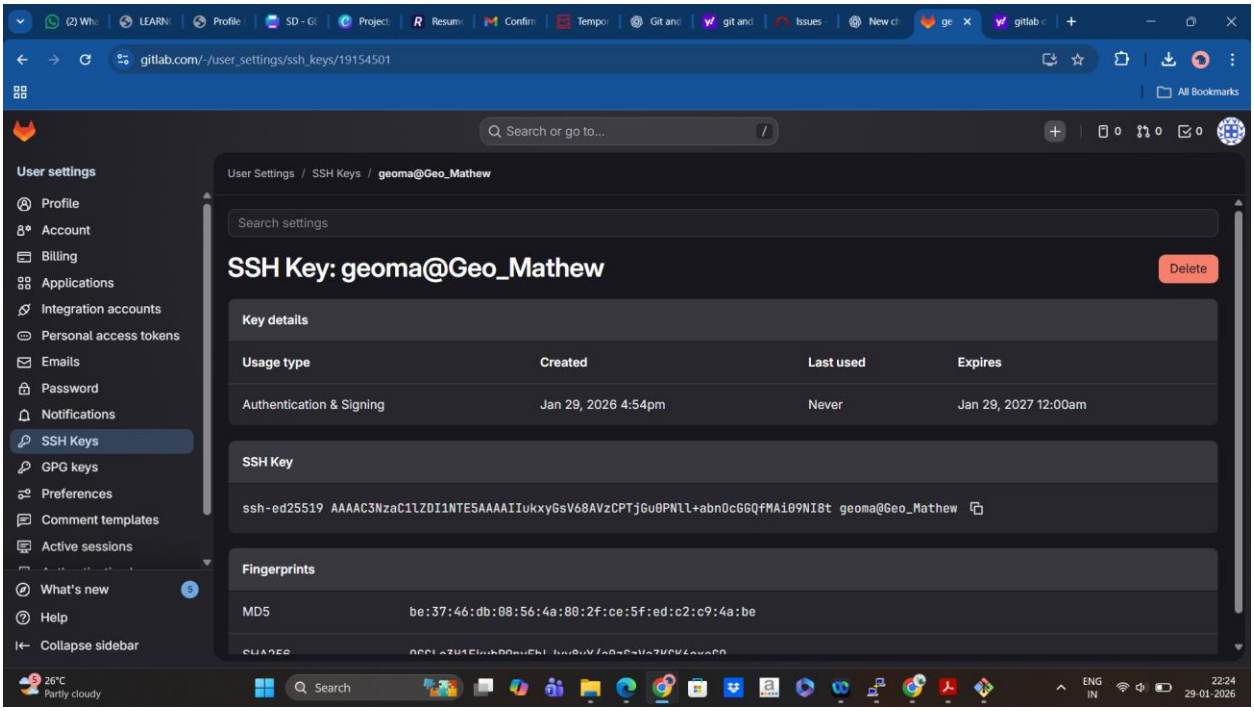
Practical 5: Branch Creation and Development

Steps performed

- Created a new development branch
- Added a new file in the branch
- Committed and pushed branch changes







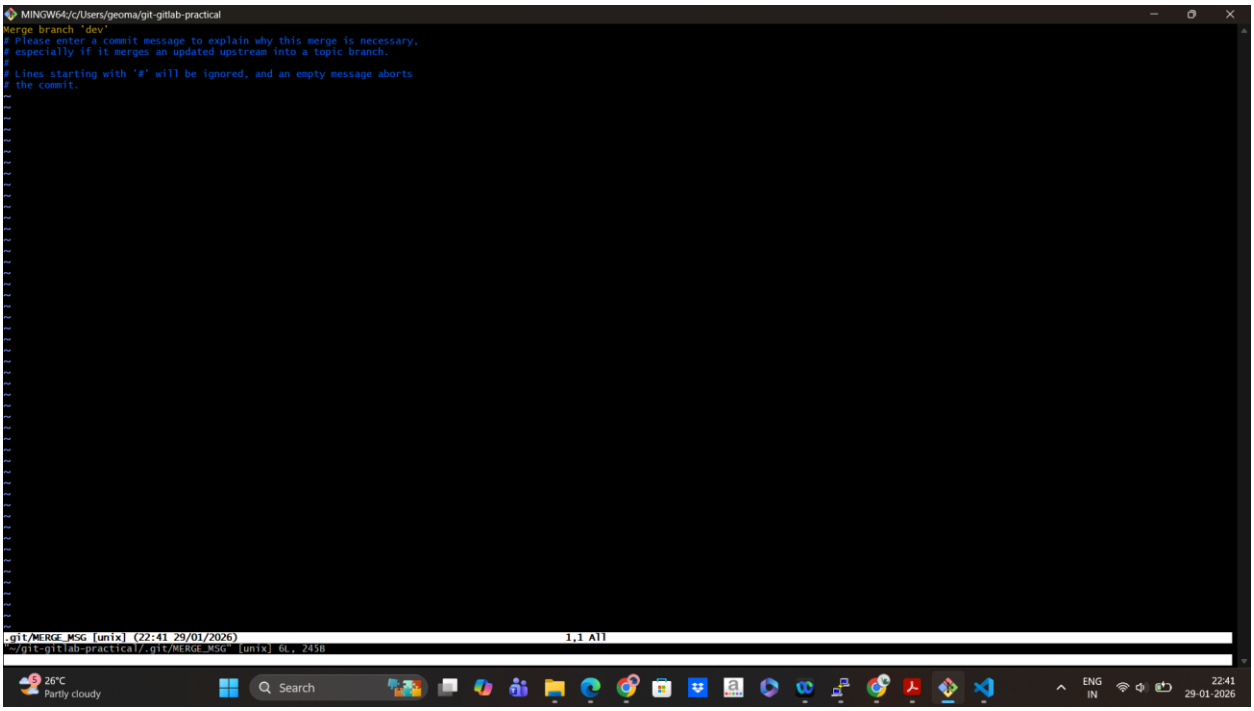
Outcome

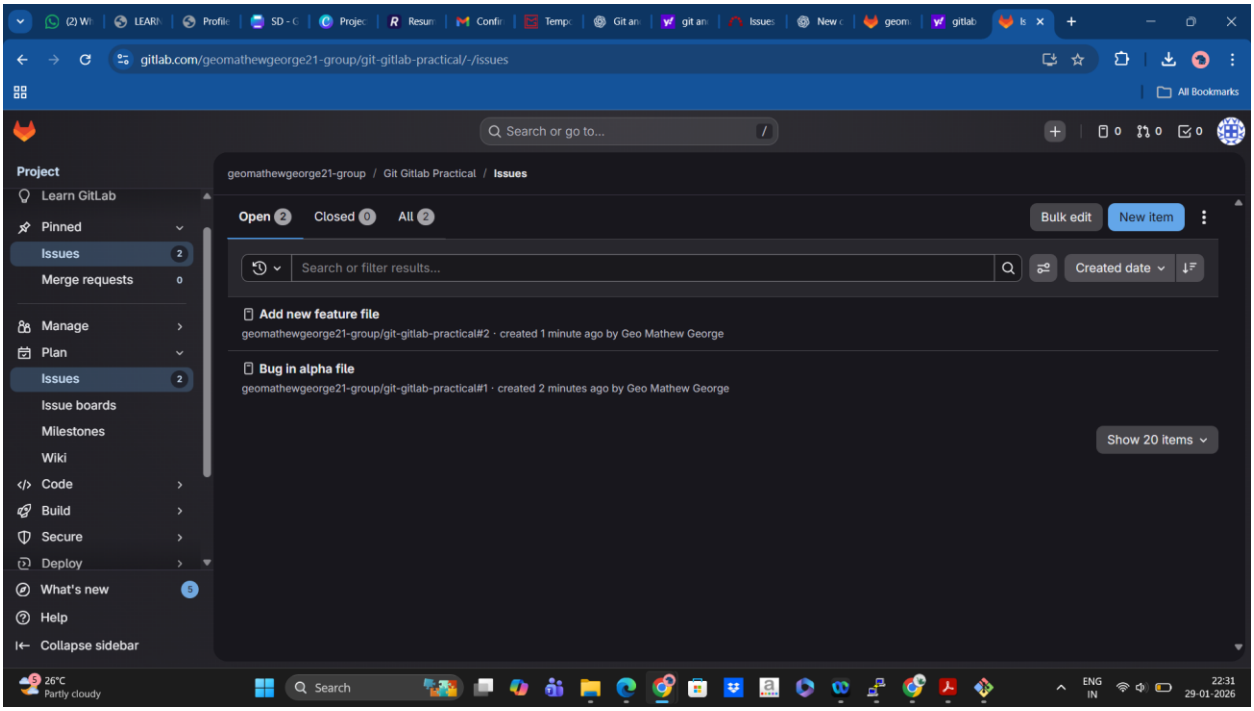
A new branch was created and updated independently.

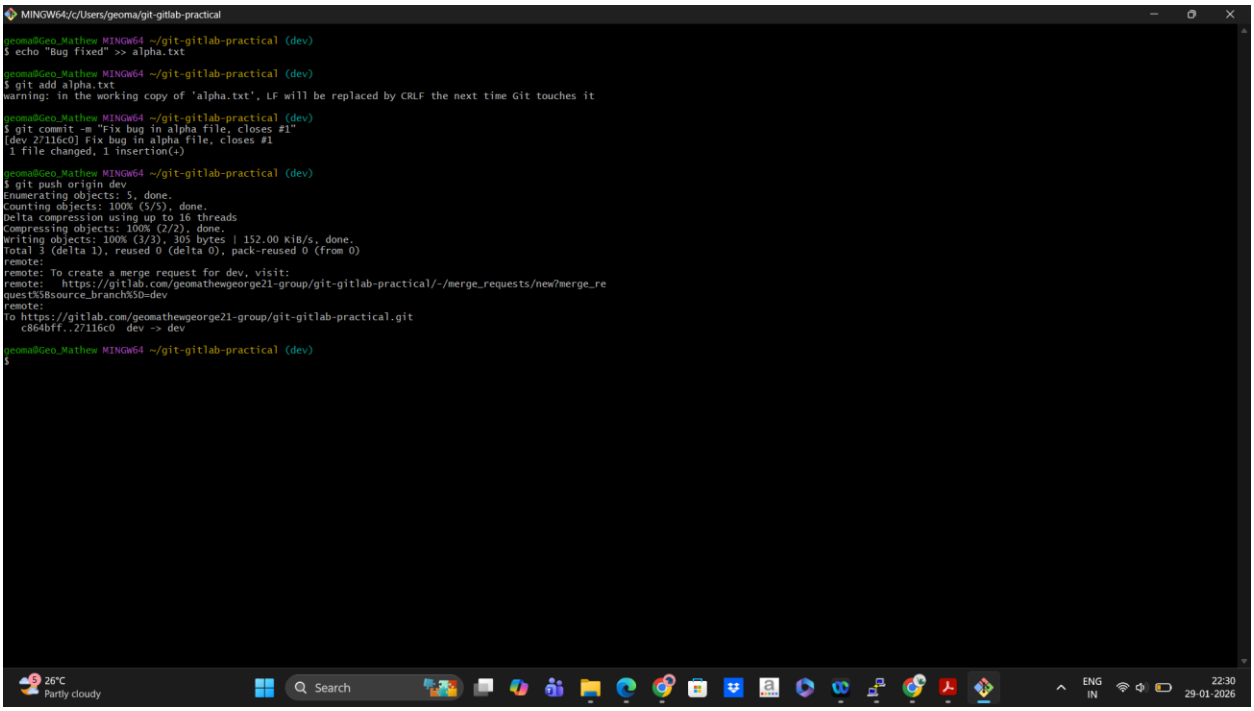
Practical 6: Create Merge Request in GitLab

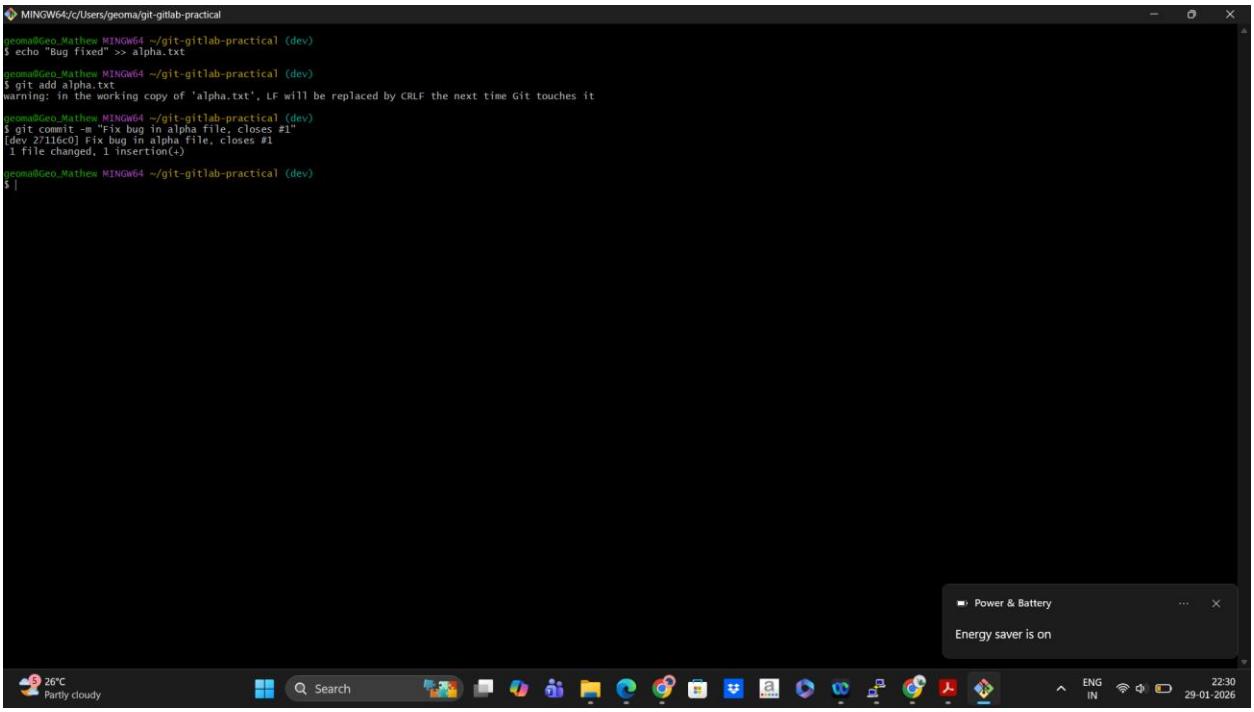
Steps performed

- Created a merge request from dev branch to main branch
- Reviewed and approved the merge request
- Merged changes and deleted the dev branch


MINGW64:/c/Users/geoma/git-gitlab-practical
Merge branch 'dev'
Please enter a commit message to explain why this merge is necessary,
especially if it merges an updated upstream into a topic branch.
#
Lines starting with '#' will be ignored, and an empty message aborts
the commit.
.git/MERGE_MSG [unix] (22:41 29/01/2026) 1,1 All
~/git-gitlab-practical/.git/MERGE_MSG [unix] 6L, 245B
26°C
Partly cloudy
Search
ENG
IN
22:41
29-01-2026


gitlab.com/geomathewgeorge21-group/git-gitlab-practical/-/issues
All Bookmarks
Search or go to...
Project
Learn GitLab
Pinned
Issues 2
Merge requests 0
Manage
Plan
Issues 2
Issue boards
Milestones
Wiki
Code
Build
Secure
Deploy
What's new 6
Help
Collapse sidebar
geomathewgeorge21-group / Git Gitlab Practical / Issues
Open 2
Closed 0
All 2
Bulk edit
New item
Search or filter results...
Created date
Add new feature file
geomathewgeorge21-group/git-gitlab-practical#2 · created 1 minute ago by Geo Mathew George
Bug in alpha file
geomathewgeorge21-group/git-gitlab-practical#1 · created 2 minutes ago by Geo Mathew George
Show 20 items
26°C
Partly cloudy
Search
ENG
IN
22:31
29-01-2026





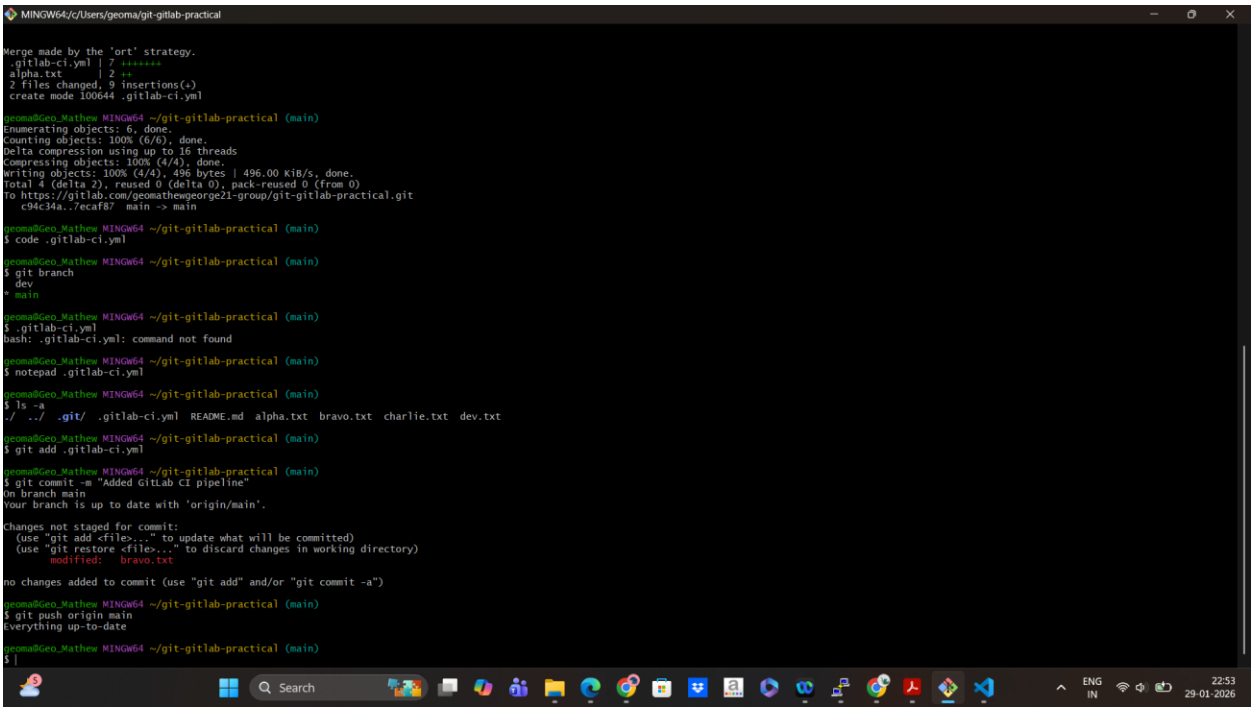
Outcome

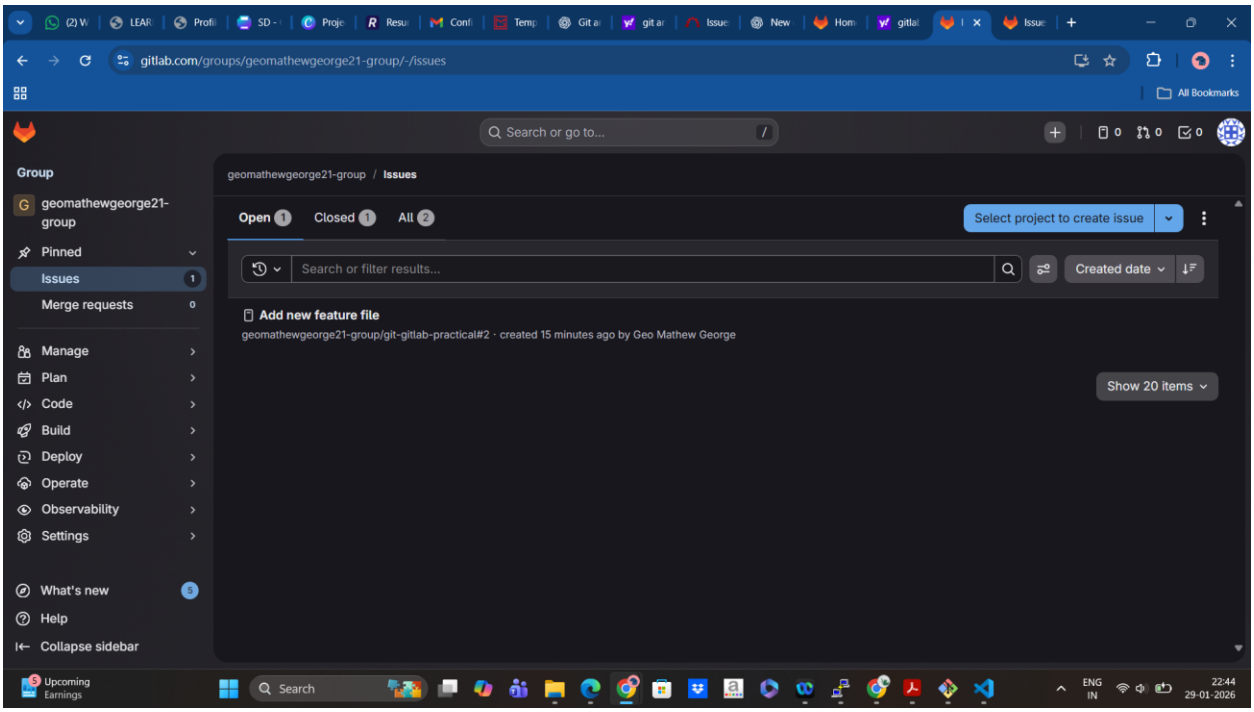
Changes from dev branch were successfully merged into main.

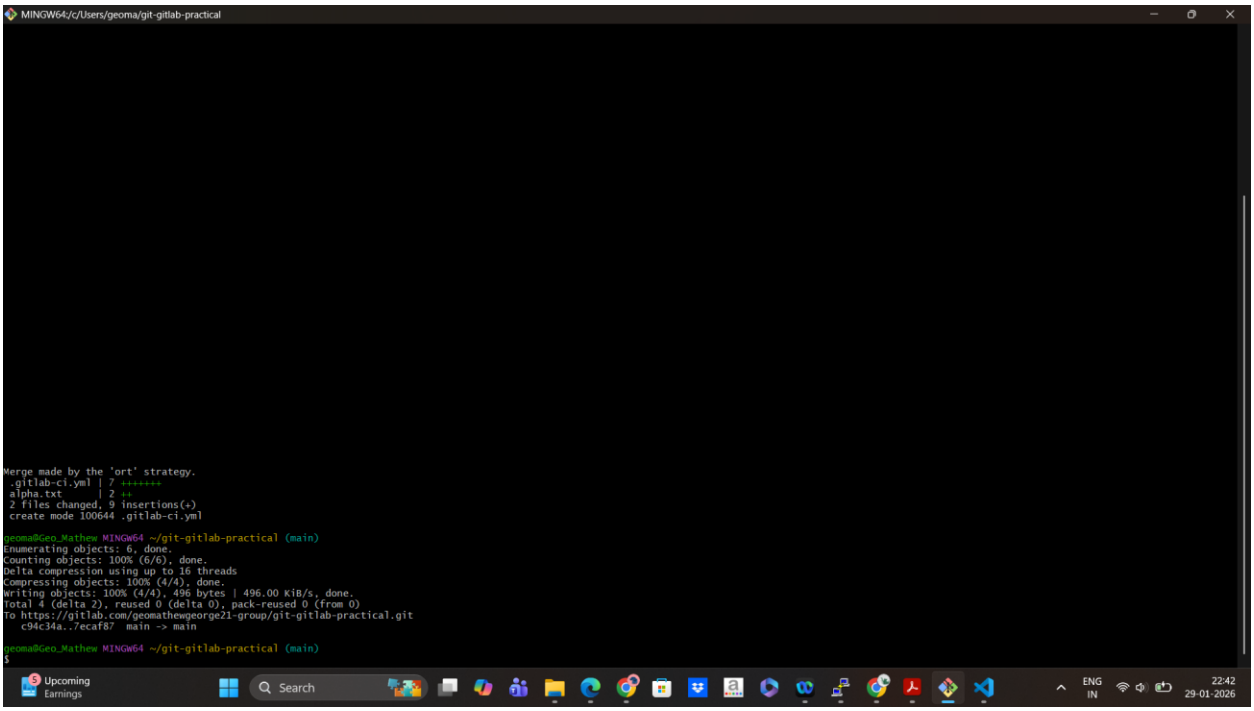
Practical 7: Clone Repository

Steps performed

- Cloned the GitLab repository to local system
- Verified cloned files


MINGW64:/c/Users/geoma/git-gitlab-practical
Merge made by the 'ort' strategy.
.gitlab-ci.yml | 7 +++++++
alpha.txt | 2 ++
2 files changed, 9 insertions(+)
create mode 100644 .gitlab-ci.yml
geoma@Geo_Mathew MINGW64 ~/git-gitlab-practical (main)
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 16 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 496 bytes | 496.00 KiB/s, done.
Total 4 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
To https://gitlab.com/geomathewgeorge21-group/git-gitlab-practical.git
c94c34a..7ecaf87 main -> main
geoma@Geo_Mathew MINGW64 ~/git-gitlab-practical (main)
$ code .gitlab-ci.yml
geoma@Geo_Mathew MINGW64 ~/git-gitlab-practical (main)
$ git branch
dev
* main
geoma@Geo_Mathew MINGW64 ~/git-gitlab-practical (main)
$ .gitlab-ci.yml
bash: .gitlab-ci.yml: command not found
geoma@Geo_Mathew MINGW64 ~/git-gitlab-practical (main)
$ notepad .gitlab-ci.yml
geoma@Geo_Mathew MINGW64 ~/git-gitlab-practical (main)
$ ls -a
./ ../ .git/ .gitlab-ci.yml README.md alpha.txt bravo.txt charlie.txt dev.txt
geoma@Geo_Mathew MINGW64 ~/git-gitlab-practical (main)
$ git add .gitlab-ci.yml
geoma@Geo_Mathew MINGW64 ~/git-gitlab-practical (main)
$ git commit -m "Added GitLab CI pipeline"
On branch main
Your branch is up to date with 'origin/main'.
Changes not staged for commit:
(use "git add <file>..." to update what will be committed)
(use "git restore <file>..." to discard changes in working directory)
modified: bravo.txt
no changes added to commit (use "git add" and/or "git commit -a")
geoma@Geo_Mathew MINGW64 ~/git-gitlab-practical (main)
$ git push origin main
Everything up-to-date
geoma@Geo_Mathew MINGW64 ~/git-gitlab-practical (main)
$
Search
ENG
IN
22:53
29-01-2026


gitlab.com/groups/geomathewgeorge21-group/-/issues
All Bookmarks
Search or go to...
Group
geomathewgeorge21-group
Pinned
Issues 1
Merge requests 0
Manage
Plan
Code
Build
Deploy
Operate
Observability
Settings
What's new 6
Help
Collapse sidebar
geomathewgeorge21-group / Issues
Open 1
Closed 1
All 2
Select project to create issue
Search or filter results...
Created date
Add new feature file
geomathewgeorge21-group/git-gitlab-practical#2 · created 15 minutes ago by Geo Mathew George
Show 20 items
Upcoming Earnings
Search
ENG
IN
22:44
29-01-2026



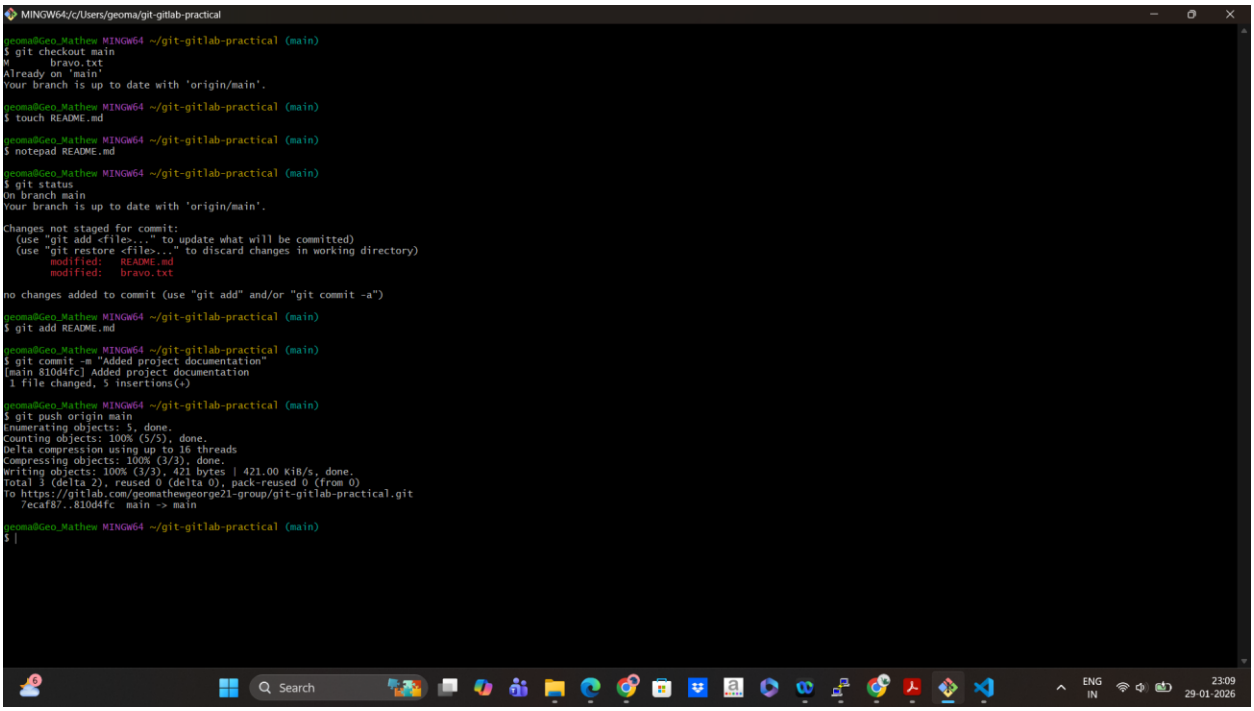
Outcome

Complete repository was successfully cloned locally.

Practical 8: Git Status, Log, and Diff

Steps performed

- Checked repository status
- Viewed commit history
- Compared file changes using diff
- Modified files and committed updates



Outcome

File changes were tracked and pushed successfully.

Practical 9: Advanced Git Operations

Steps performed

- Used Git stash to temporarily save changes
- Used Git revert to undo a commit safely

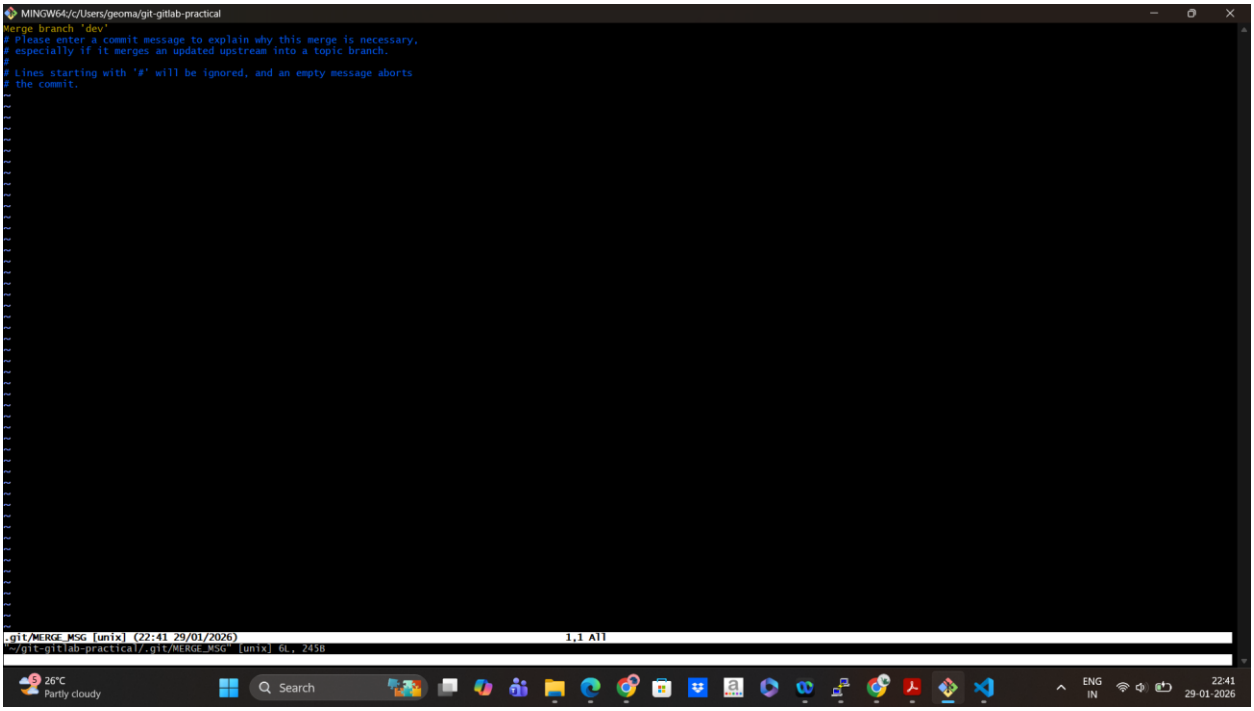
Outcome

Advanced Git operations were executed successfully.

Practical 10: SSH Key Configuration

Steps performed

- Generated SSH key pair
- Added public key to GitLab account
- Tested SSH connection



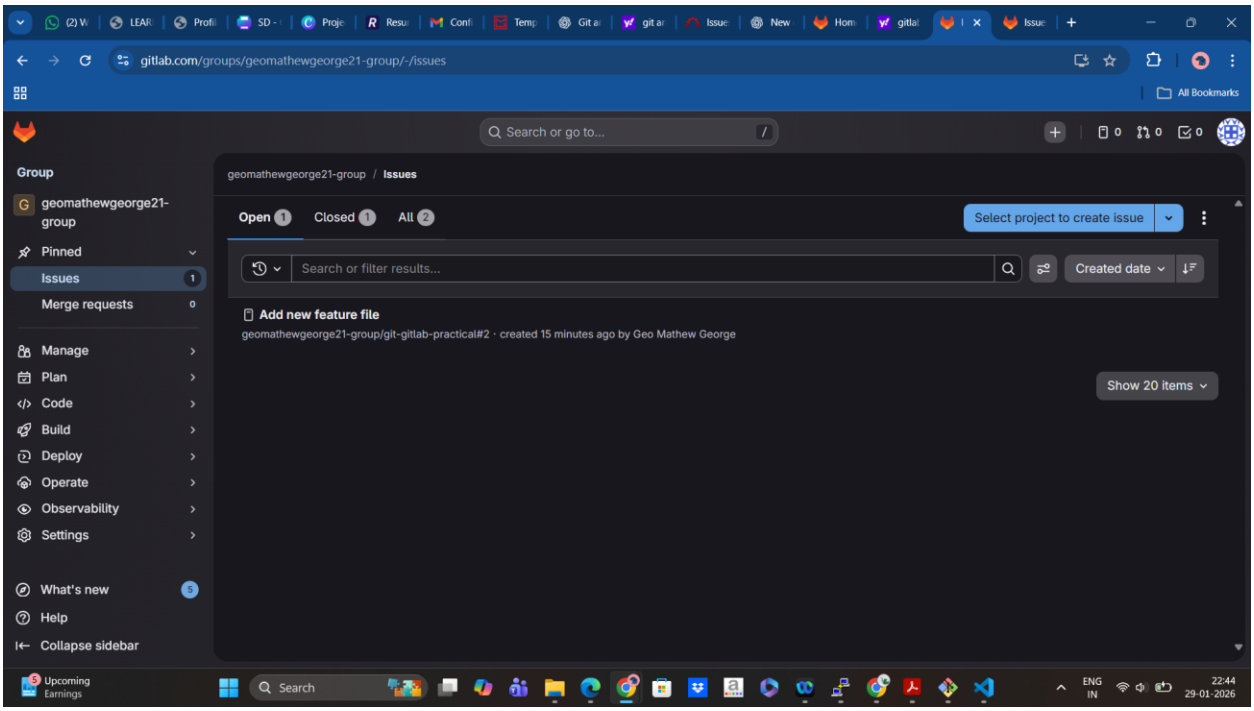
Outcome

SSH authentication was successfully configured.

Practical 11: GitLab Issues

Steps performed

- Created multiple issues in GitLab
- Linked commits to issues
- Closed issues after resolution



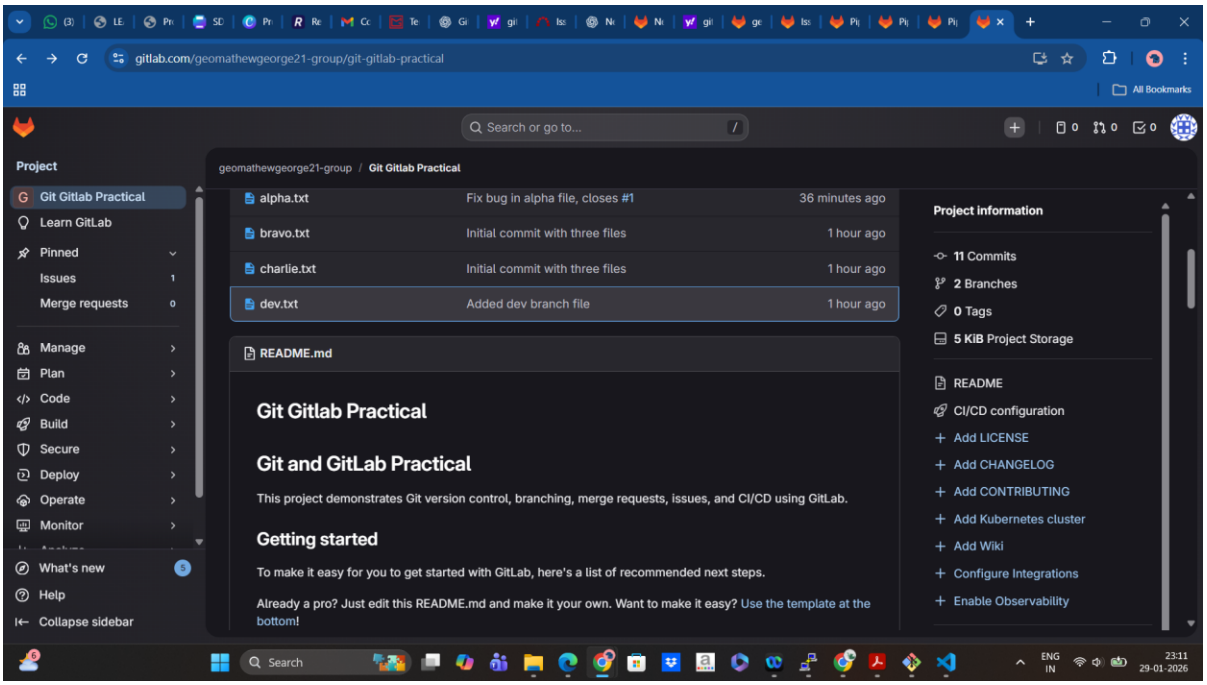
Outcome

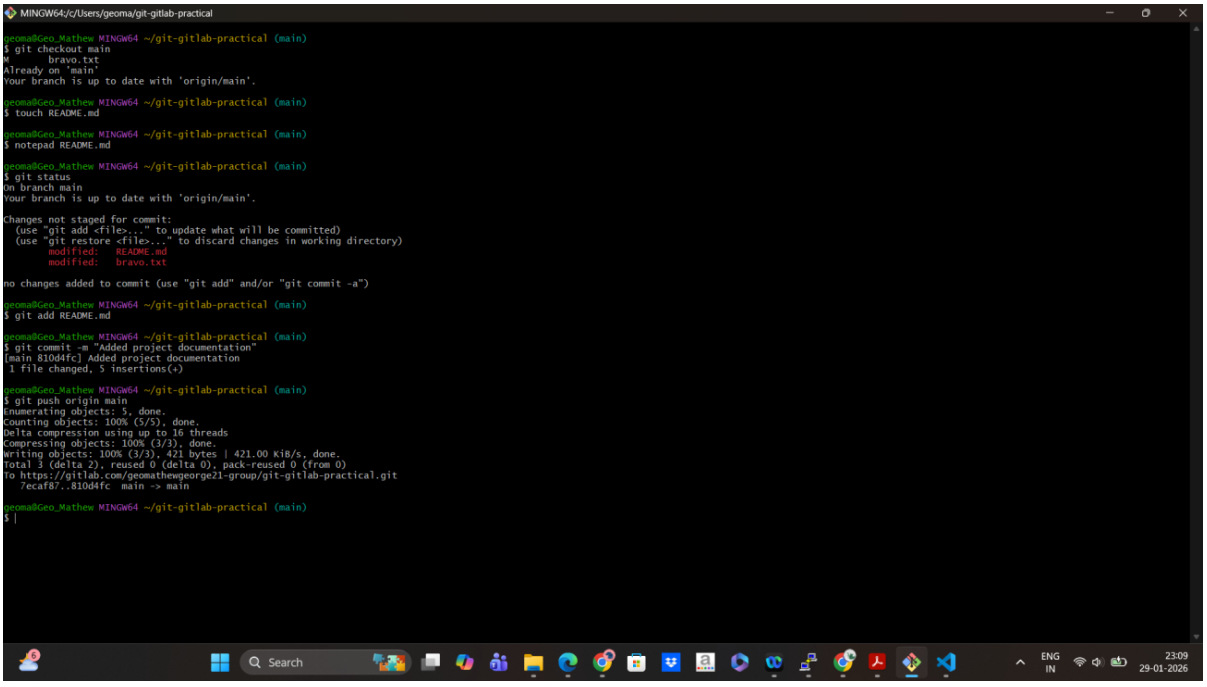
Issue tracking and management were demonstrated.

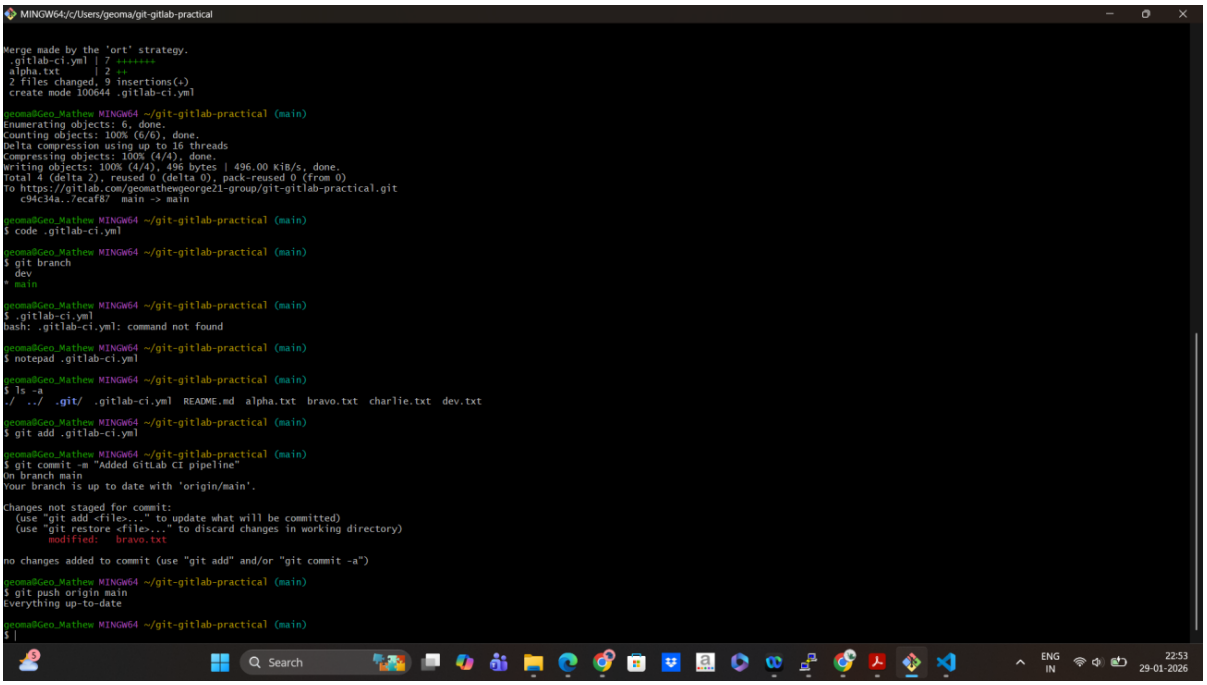
Practical 12: GitLab CI/CD Pipeline

Steps performed

- Created GitLab CI configuration file
- Defined test stage and job
- Committed and pushed pipeline configuration
- Verified pipeline execution

- 
gitlab.com/geomathewgeorge21-group/git-gitlab-practical
Search or go to...
Project
Git Gitlab Practical
Learn GitLab
Pinned
Issues 1
Merge requests 0
Manage
Plan
Code
Build
Secure
Deploy
Operate
Monitor
What's new 5
Help
Collapse sidebar
geomathewgeorge21-group / Git Gitlab Practical
alpha.txt Fix bug in alpha file, closes #1 36 minutes ago
bravo.txt Initial commit with three files 1 hour ago
charlie.txt Initial commit with three files 1 hour ago
dev.txt Added dev branch file 1 hour ago
README.md
Git Gitlab Practical
Git and GitLab Practical
This project demonstrates Git version control, branching, merge requests, issues, and CI/CD using GitLab.
Getting started
To make it easy for you to get started with GitLab, here's a list of recommended next steps.
Already a pro? Just edit this README.md and make it your own. Want to make it easy? Use the template at the bottom!
Project information
11 Commits
2 Branches
0 Tags
5 KiB Project Storage
README
CI/CD configuration
Add LICENSE
Add CHANGELOG
Add CONTRIBUTING
Add Kubernetes cluster
Add Wiki
Configure Integrations
Enable Observability
Search
ENG IN
23:11
29-01-2026


MINGW64:/c/Users/geoma/git-gitlab-practical
geoma@Geo_Mathew MINGW64 ~/git-gitlab-practical (main)
$ git checkout main
M bravo.txt
Already on 'main'
Your branch is up to date with 'origin/main'.
$ touch README.md
$ notepad README.md
$ git status
On branch main
Your branch is up to date with 'origin/main'.
Changes not staged for commit:
(use "git add <file>..." to update what will be committed)
(use "git restore <file>..." to discard changes in working directory)
modified: README.md
modified: bravo.txt
no changes added to commit (use "git add" and/or "git commit -a")
$ git add README.md
$ git commit -m "Added project documentation"
[main 810d4fc] Added project documentation
1 file changed, 5 insertions(+)
$ git push origin main
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 16 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 421 bytes | 421.00 KiB/s, done.
Total 3 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
To https://gitlab.com/geomathewgeorge21-group/git-gitlab-practical.git
7ecaf87..810d4fc main -> main
Search
ENG IN
23:09
29-01-2026



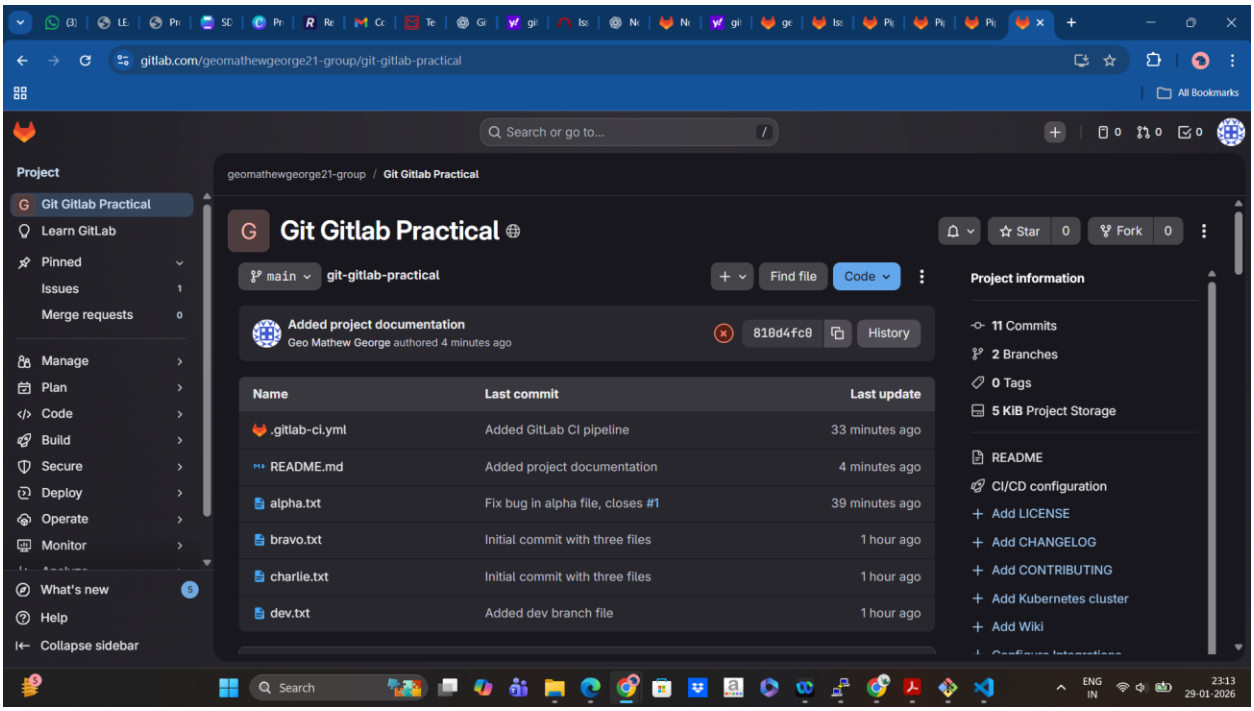
Outcome

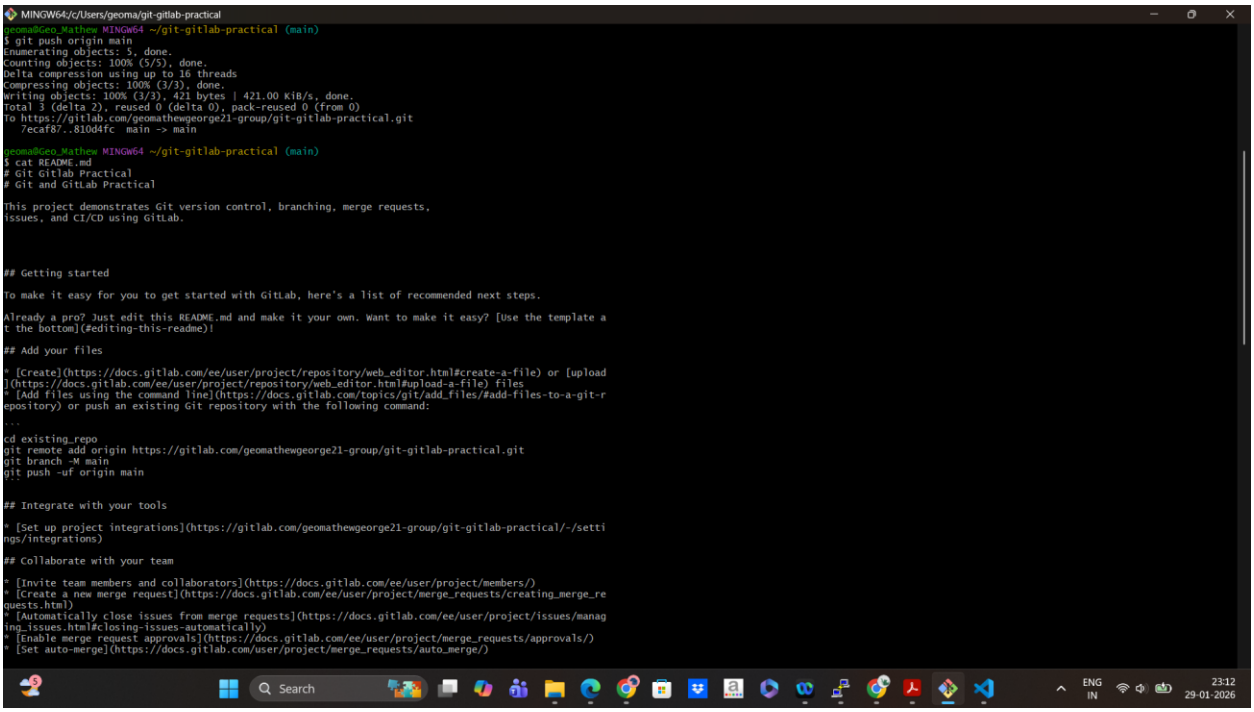
CI/CD pipeline executed successfully.

Practical 13: Documentation

Steps performed

- Created project documentation using README file
- Committed and pushed documentation to repository





Outcome

Project documentation was successfully added and maintained.

Git and GitLab Are Used in Real-World DevOps Environments

In real companies, Git and GitLab are used together to help teams build, test, and deliver software smoothly and safely.

Git is Used (Developer Side)

Developers use Git to manage their daily work.

- Git keeps track of every change made to the code, just like saving different versions of a document
- Multiple people can work on the same project without overwriting each other's work
- Developers create branches to work on new features separately, so the main project stays safe
- If something breaks, Git allows the team to go back to an older working version quickly
- Every change has a history, so it's easy to know who changed what and why

GitLab Is Used (Team and DevOps Side)

GitLab is where the team comes together.

- It stores the main project code in one safe, central place
- Team leads control who can view or change the code
- Issues are used to track bugs, tasks, and new features, just like a to-do list
- Merge Requests allow code to be checked and approved before it becomes part of the main project
- GitLab automatically runs tests and builds using CI/CD pipelines
- Applications can be deployed to servers automatically without manual effort

Real-Life Example

Imagine a team building a **food delivery app**.

One developer works on the login page, another on payments, and another on the admin panel.

- Each developer uses **Git** to save their work and experiment safely on their own branch

- All code is pushed to **GitLab**, where the team can see everyone's progress
- When a feature is ready, a Merge Request is created so the team lead can review it
- GitLab automatically tests the code before allowing it to be merged
- Once approved, the app is automatically updated on the server

If a problem appears after deployment, the team can quickly go back to the last working version using Git.

Why Is Important in DevOps

By using Git and GitLab together:

- Work becomes faster and more organized
- Mistakes are reduced because changes are reviewed and tested
- Teams communicate better and work more transparently
- Software is delivered more often and with better quality

My Personal Understanding and Learning Outcomes

This learning experience helped me understand how modern software development actually works. At first, version control felt confusing and difficult, but with regular practice, I became comfortable using Git commands and working in GitLab.

I learned how real development teams work together without disturbing each other's work. I understood why clear commit messages, proper use of branches, and code reviews are important. I also learned that CI pipelines act like a safety check by finding mistakes early before the code goes live.

Working with Merge Requests showed me that software development is not only about writing code. It also involves teamwork, responsibility, and maintaining good quality. Learning GitLab CI helped me see how automation saves time and reduces manual work.

Overall, this learning changed the way I look at DevOps tools. Git and GitLab are no longer just tools for me. They are important systems that help teams work together smoothly and deliver reliable software.

2)Terraform

Topics

During this learning period, I studied Terraform as a tool used to manage infrastructure using code. The learning mainly focused on creating and managing AWS cloud resources using Terraform.

The topics covered include:

- Basics of Terraform and the concept of Infrastructure as Code
- Installing Terraform and setting up an AWS account
- Understanding Terraform configuration language (HCL)
- Using providers and resources to create cloud infrastructure
- Using Terraform commands such as init, plan, apply, and destroy
- Working with variables, outputs, and local values
- Using data sources to fetch existing infrastructure details
- Organizing Terraform projects using proper folder structure and best practices
- Creating and using Terraform modules for reusable infrastructure
- Managing Terraform state and using remote backends
- Using workspaces to manage multiple environments like development and production
- Understanding Terraform planning and execution flow
- Using provisioners and null resources when required
- Learning production-level Terraform concepts
- Understanding how Terraform is used in team-based DevOps environments

These topics were learned through structured Terraform tutorials and practical AWS-based examples.

Key Concepts Understood from Terraform

Terraform (Infrastructure as Code Tool)
Terraform allows us to create and manage cloud infrastructure using code instead of doing everything manually. We describe what the infrastructure should look like, and

Terraform takes care of creating it. When the same code is run multiple times, it gives the same result, which keeps the infrastructure consistent.

Providers

Providers are connectors that allow Terraform to talk to cloud platforms like AWS. For example, the AWS provider lets Terraform create resources such as virtual machines, storage buckets, and networks.

Resources

Resources are the actual cloud components that Terraform creates. Examples include virtual servers, storage buckets, and security rules.

Terraform State

Terraform keeps a record of the infrastructure it manages in a state file. This helps Terraform compare what already exists with what is defined in the code. For team work, this state file is usually stored remotely to avoid conflicts and data loss.

Modules

Modules help reuse the same infrastructure code in multiple environments like development and production. This keeps the code clean and avoids repetition.

Variables and Outputs

Variables allow us to change values without changing the main code, making it more flexible. Outputs show important information such as server IP addresses or URLs after the infrastructure is created.

Plan and Apply

Terraform plan shows a preview of the changes before they are made. Terraform apply then creates or updates the infrastructure after confirmation.

Workspaces

Workspaces allow the same Terraform code to be used for different environments, with each environment having its own separate infrastructure state.

Terraform Practical Sessions – Topics Covered

Practical Session 1: Terraform Installation and Initial Setup

Activities performed

- Downloaded and installed Terraform
- Verified installation using the version command
- Created an AWS account and IAM user
- Configured AWS credentials for Terraform
- Created the first Terraform working directory

Outcome

Terraform was successfully installed and connected to AWS.

Practical Session 2: Creating the First Terraform Project

Activities performed

- Created Terraform configuration files
- Configured the AWS provider
- Created an S3 bucket using Terraform
- Ran init, plan, and apply commands
- Verified the resource in AWS Console
- Destroyed the resource after testing

Concepts practiced

- Providers
- Resources
- Basic Terraform workflow

Practical Session 3: Terraform Resource Management

Activities performed

- Created EC2 instances using Terraform
- Created and attached security groups
- Managed multiple resources in one project
- Learned how Terraform handles dependencies

Concepts practiced

- Resource blocks
- Resource dependency handling
- EC2 and security groups

Practical Session 4: Using Terraform Variables

Activities performed

- Defined input variables
- Used variables inside resource blocks
- Passed values using command line, variable files, and environment variables

Concepts practiced

- Variable types
- Default values
- Code reusability

Practical Session 5: Terraform Outputs and Locals

Activities performed

- Defined output values
- Displayed public IP and DNS of EC2 instances
- Used local values to simplify configurations

Concepts practiced

- Outputs

- Local values
- Clean and readable code

Practical Session 6: Terraform Data Sources

Activities performed

- Used data sources to fetch existing AWS resources
- Retrieved AMI IDs and VPC details
- Used fetched data in resource creation

Concepts practiced

- Data blocks
- Dynamic infrastructure configuration

Practical Session 7: Project Structure

Activities performed

- Split code into multiple files
- Organized variables and outputs properly
- Followed naming conventions and folder structure

Concepts practiced

- Project organization
- Maintainability
- Best practices

Practical Session 8: Terraform Modules

Activities performed

- Created a reusable EC2 module

- Passed variables into modules
- Used module outputs
- Reused the same module for multiple environments

Concepts practiced

- Module creation
- Module reuse
- DRY principle

Practical Session 9: Terraform State Management

Activities performed

- Studied local state file behavior
- Configured remote state using S3
- Enabled state locking using DynamoDB
- Observed state updates during execution

Concepts practiced

- Terraform state
- Remote backend
- State locking

Practical Session 10: Terraform Workspaces

Activities performed

- Created multiple workspaces
- Switched between environments
- Deployed infrastructure separately for each environment

Concepts practiced

- Workspace isolation

- Environment management

Practical Session 11: Terraform Plan and Lifecycle Management

Activities performed

- Generated and reviewed execution plans
- Used lifecycle rules to control resource behavior
- Performed safe updates

Concepts practiced

- Planning before execution
- Resource lifecycle control

Practical Session 12: Provisioners and Null Resources

Activities performed

- Used local and remote provisioners
- Ran scripts after resource creation
- Used null resources for orchestration

Concepts practiced

- Provisioners
- Post-creation configuration

Practical Session 13: Terraform Destroy and Cleanup

Activities performed

- Destroyed infrastructure safely
- Verified cleanup in AWS Console
- Learned cost-saving practices

Concepts practiced

- Resource cleanup
- Cost control

Practical Session 14: Terraform with Version Control

Activities performed

- Stored Terraform code in Git repository
- Tracked changes using commits
- Rolled back infrastructure when needed
- Collaborated using Git workflow

Concepts practiced

- Git and Terraform integration
- Team collaboration

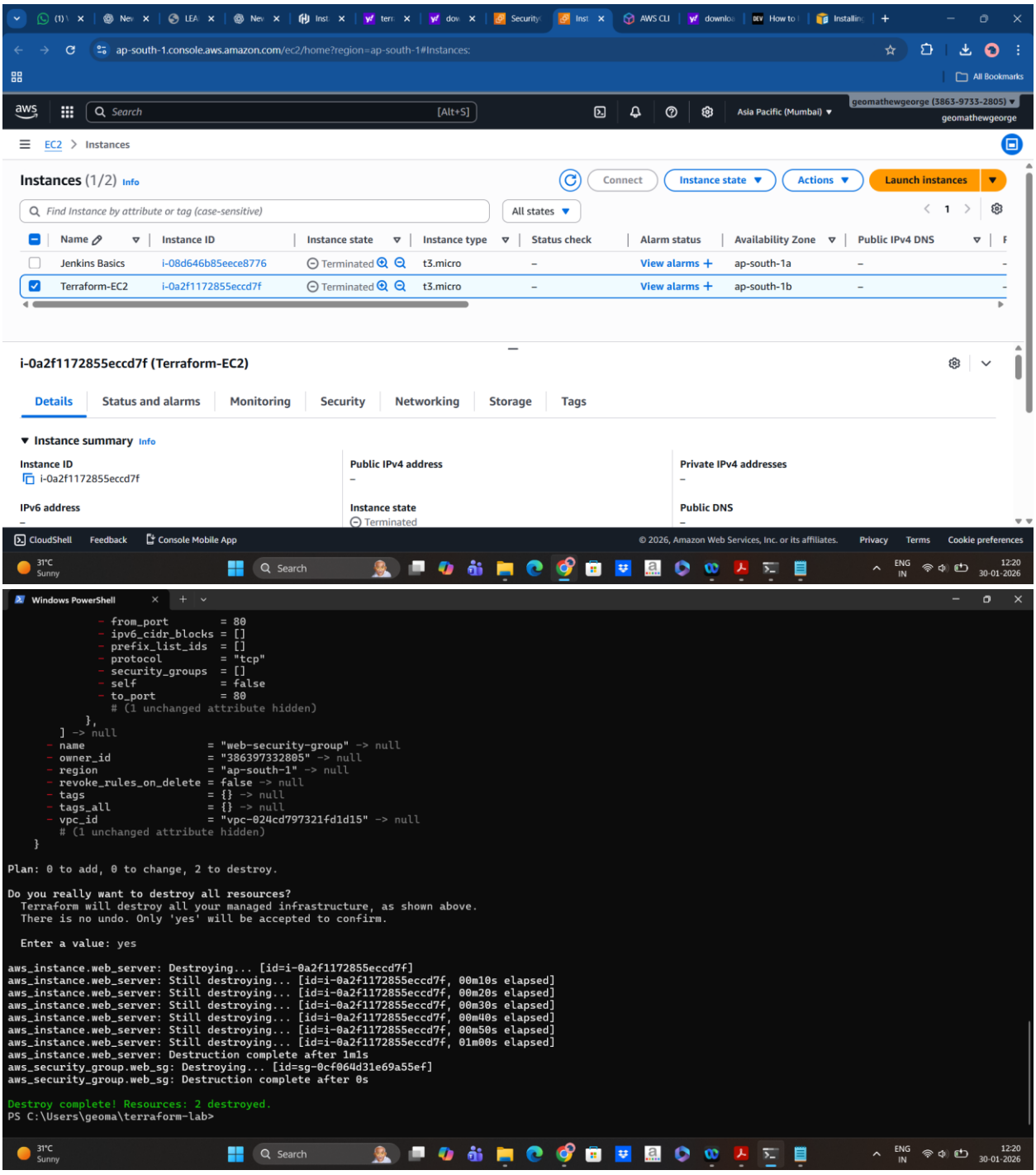
Practical Session 15: Terraform in CI/CD

Activities performed

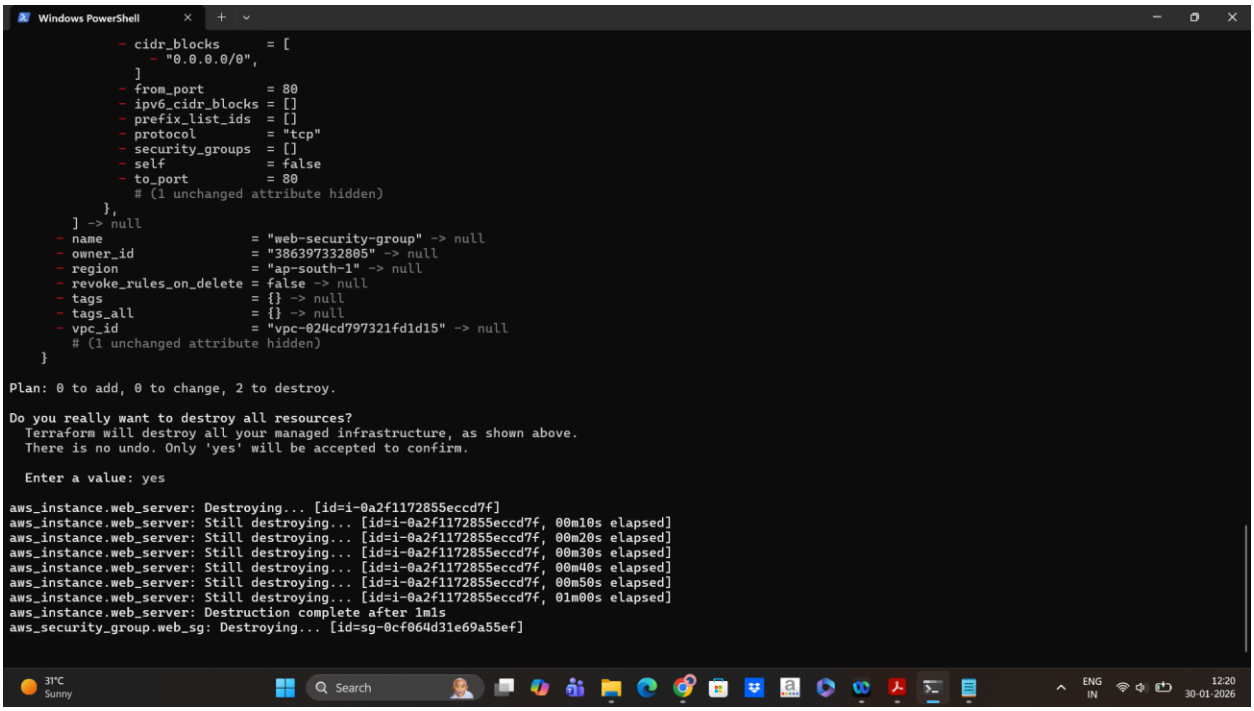
- Integrated Terraform with CI/CD tools
- Automated plan and apply steps
- Used approval-based deployments

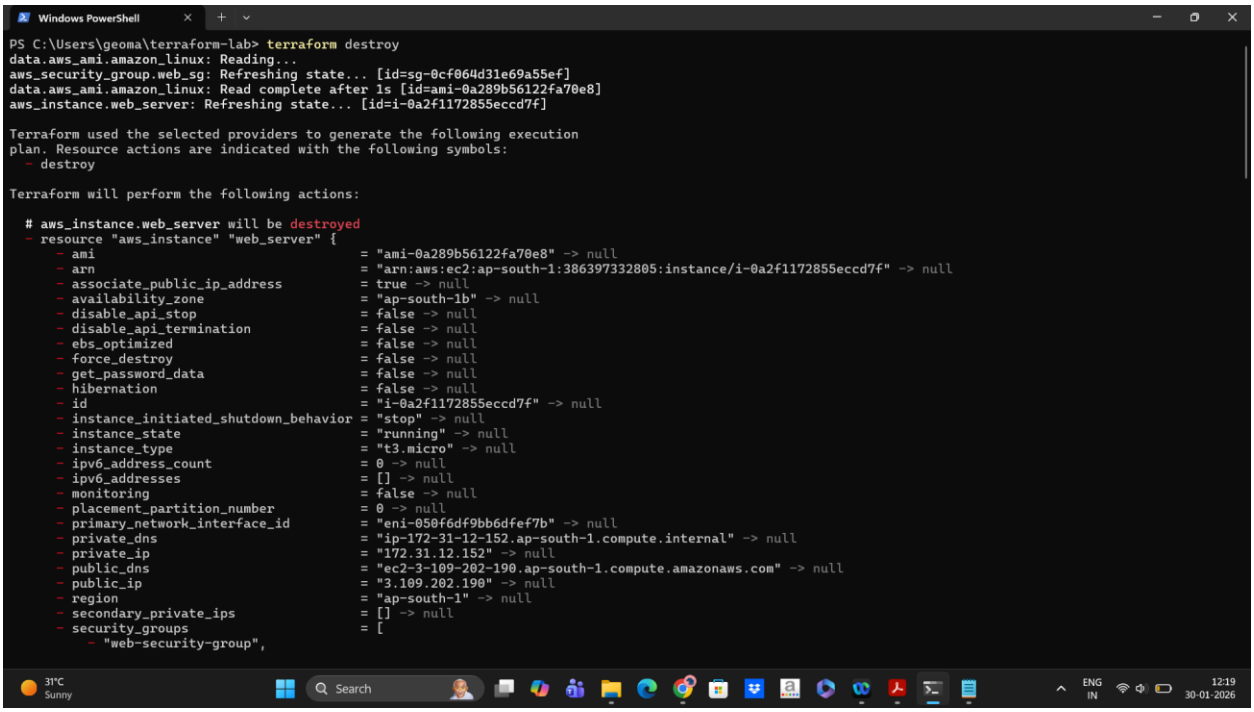
Concepts practiced

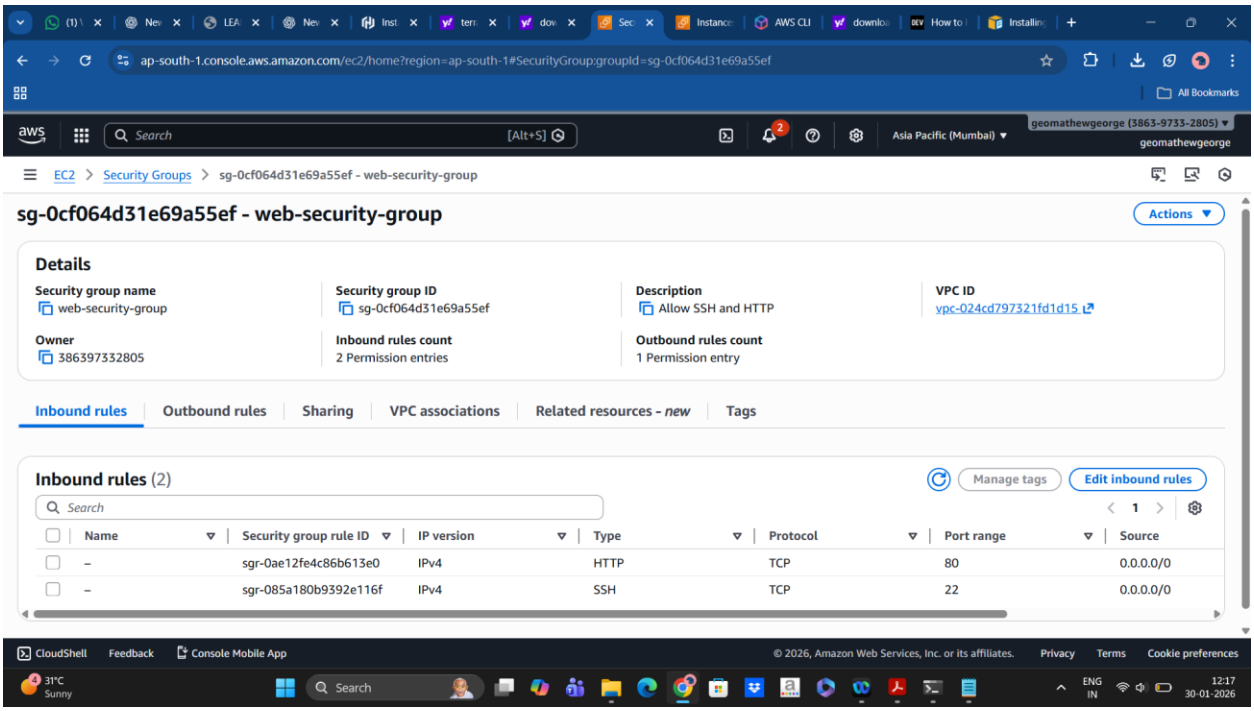
- DevOps automation
- CI/CD integration

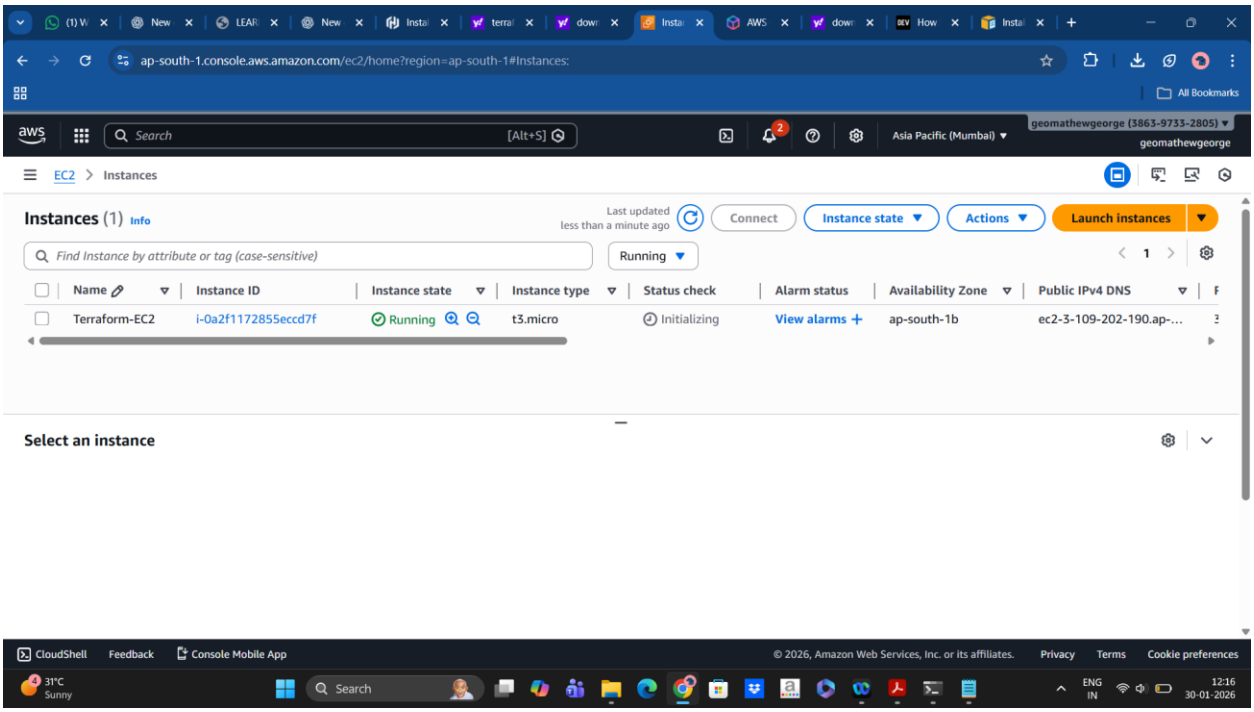

ap-south-1.console.aws.amazon.com/ec2/home?region=ap-south-1#Instances:
Search [Alt+S]
Asia Pacific (Mumbai)
geomathewgeorge (3863-9733-2805)
EC2 > Instances
Instances (1/2) Info
Connect
Instance state
Actions
Launch instances
Find Instance by attribute or tag (case-sensitive)
All states
Name
Instance ID
Instance state
Instance type
Status check
Alarm status
Availability Zone
Public IPv4 DNS
Jenkins Basics
i-08d646b85eece8776
Terminated
t3.micro
View alarms
ap-south-1a
Terraform-EC2
i-0a2f1172855eccd7f
Terminated
t3.micro
View alarms
ap-south-1b
i-0a2f1172855eccd7f (Terraform-EC2)
Details
Status and alarms
Monitoring
Security
Networking
Storage
Tags
Instance summary Info
Instance ID
i-0a2f1172855eccd7f
Public IPv4 address
Private IPv4 addresses
IPv6 address
Instance state
Terminated
Public DNS
CloudShell
Feedback
Console Mobile App
© 2026, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
31°C
Sunny
Search
ENG
IN
12:20
30-01-2026

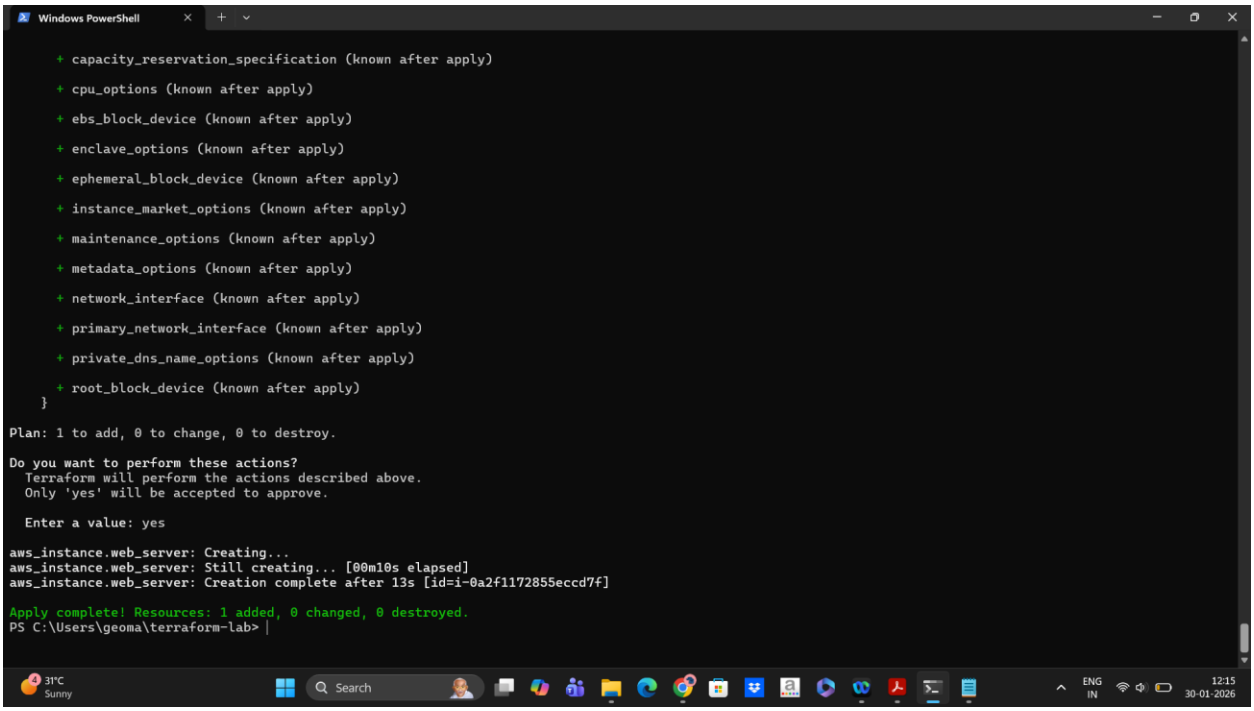
Windows PowerShell
- from_port = 80
- ipv6_cidr_blocks = []
- prefix_list_ids = []
- protocol = "tcp"
- security_groups = []
- self = false
- to_port = 80
(1 unchanged attribute hidden)
},
] -> null
- name = "web-security-group" -> null
- owner_id = "386397332805" -> null
- region = "ap-south-1" -> null
- revoke_rules_on_delete = false -> null
- tags = {} -> null
- tags_all = {} -> null
- vpc_id = "vpc-024cd797321fd1d15" -> null
# (1 unchanged attribute hidden)
}
Plan: 0 to add, 0 to change, 2 to destroy.
Do you really want to destroy all resources?
Terraform will destroy all your managed infrastructure, as shown above.
There is no undo. Only 'yes' will be accepted to confirm.
Enter a value: yes
aws_instance.web_server: Destroying... [id=i-0a2f1172855eccd7f]
aws_instance.web_server: Still destroying... [id=i-0a2f1172855eccd7f, 00m10s elapsed]
aws_instance.web_server: Still destroying... [id=i-0a2f1172855eccd7f, 00m20s elapsed]
aws_instance.web_server: Still destroying... [id=i-0a2f1172855eccd7f, 00m30s elapsed]
aws_instance.web_server: Still destroying... [id=i-0a2f1172855eccd7f, 00m40s elapsed]
aws_instance.web_server: Still destroying... [id=i-0a2f1172855eccd7f, 00m50s elapsed]
aws_instance.web_server: Still destroying... [id=i-0a2f1172855eccd7f, 01m00s elapsed]
aws_instance.web_server: Destruction complete after 1m1s
aws_security_group.web_sg: Destroying... [id=sg-0cf064d31e69a55ef]
aws_security_group.web_sg: Destruction complete after 0s
Destroy complete! Resources: 2 destroyed.
PS C:\Users\geoma\terraform-lab>
31°C
Sunny
Search
ENG
IN
12:20
30-01-2026

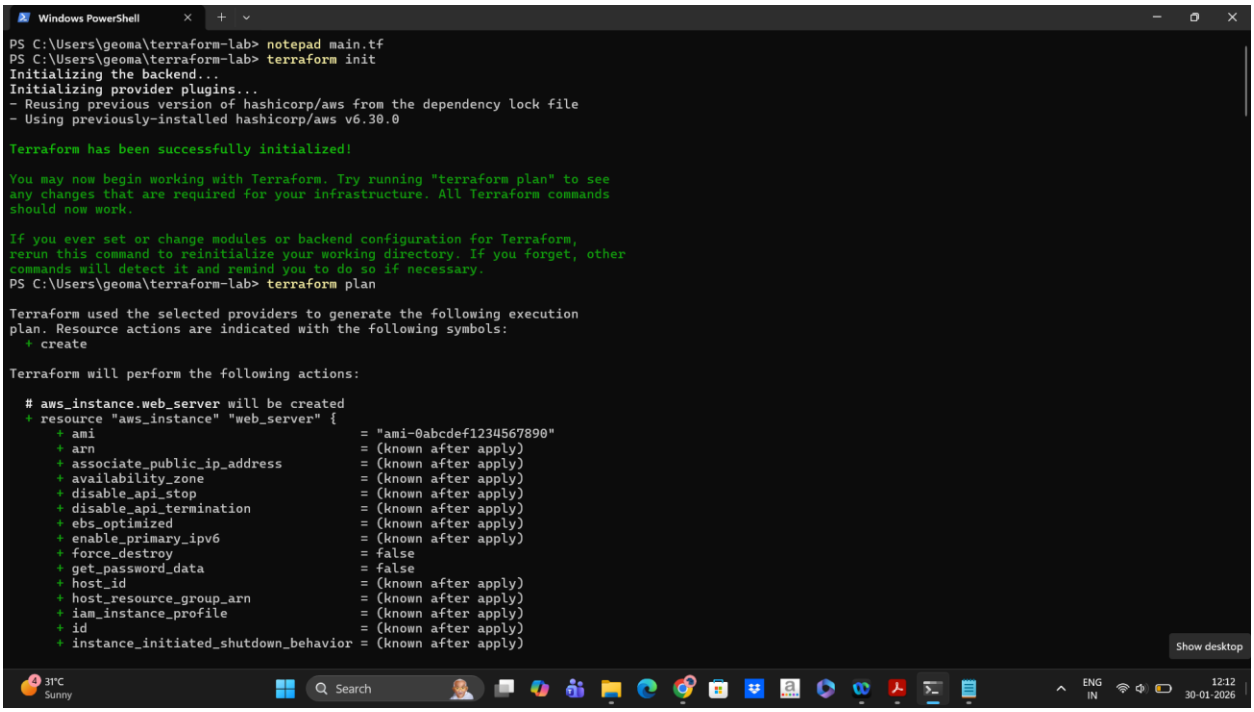

Windows PowerShell
- cidr_blocks = [
"0.0.0.0/0",
]
- from_port = 80
- ipv6_cidr_blocks = []
- prefix_list_ids = []
- protocol = "tcp"
- security_groups = []
- self = false
- to_port = 80
(1 unchanged attribute hidden)
},
] -> null
- name = "web-security-group" -> null
- owner_id = "386397332805" -> null
- region = "ap-south-1" -> null
- revoke_rules_on_delete = false -> null
- tags = {} -> null
- tags_all = {} -> null
- vpc_id = "vpc-024cd797321fd1d15" -> null
# (1 unchanged attribute hidden)
}
Plan: 0 to add, 0 to change, 2 to destroy.
Do you really want to destroy all resources?
Terraform will destroy all your managed infrastructure, as shown above.
There is no undo. Only 'yes' will be accepted to confirm.
Enter a value: yes
aws_instance.web_server: Destroying... [id=i-0a2f1172855eccd7f]
aws_instance.web_server: Still destroying... [id=i-0a2f1172855eccd7f, 00m10s elapsed]
aws_instance.web_server: Still destroying... [id=i-0a2f1172855eccd7f, 00m20s elapsed]
aws_instance.web_server: Still destroying... [id=i-0a2f1172855eccd7f, 00m30s elapsed]
aws_instance.web_server: Still destroying... [id=i-0a2f1172855eccd7f, 00m40s elapsed]
aws_instance.web_server: Still destroying... [id=i-0a2f1172855eccd7f, 00m50s elapsed]
aws_instance.web_server: Still destroying... [id=i-0a2f1172855eccd7f, 01m00s elapsed]
aws_instance.web_server: Destruction complete after 1m1s
aws_security_group.web_sg: Destroying... [id=sg-0cf064d31e69a55ef]
31°C
Sunny
Search
ENG
IN
12:20
30-01-2026

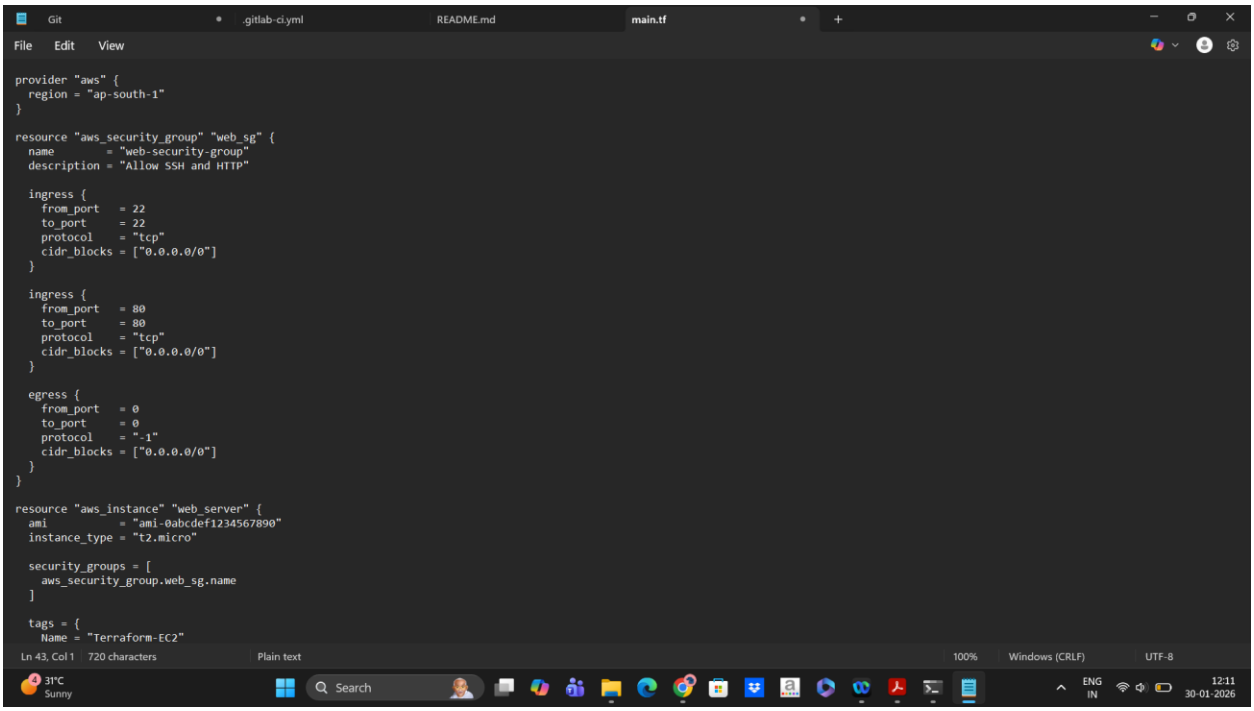

Windows PowerShell
PS C:\Users\geoma\terraform-lab> terraform destroy
data.aws_ami.amazon_linux: Reading...
aws_security_group.web_sg: Refreshing state... [id=sg-0cf064d31e69a55ef]
data.aws_ami.amazon_linux: Read complete after 1s [id=ami-0a289b56122fa70e8]
aws_instance.web_server: Refreshing state... [id=i-0a2f1172855eccd7f]
Terraform used the selected providers to generate the following execution
plan. Resource actions are indicated with the following symbols:
- destroy
Terraform will perform the following actions:
# aws_instance.web_server will be destroyed
- resource "aws_instance" "web_server" {
- ami = "ami-0a289b56122fa70e8" -> null
- arn = "arn:aws:ec2:ap-south-1:386397332805:instance/i-0a2f1172855eccd7f" -> null
- associate_public_ip_address = true -> null
- availability_zone = "ap-south-1b" -> null
- disable_api_stop = false -> null
- disable_api_termination = false -> null
- ebs_optimized = false -> null
- force_destroy = false -> null
- get_password_data = false -> null
- hibernation = false -> null
- id = "i-0a2f1172855eccd7f" -> null
- instance_initiated_shutdown_behavior = "stop" -> null
- instance_state = "running" -> null
- instance_type = "t3.micro" -> null
- ipv6_address_count = 0 -> null
- ipv6_addresses = [] -> null
- monitoring = false -> null
- placement_partition_number = 0 -> null
- primary_network_interface_id = "eni-050f6df9bb6dfef7b" -> null
- private_dns = "ip-172-31-12-152.ap-south-1.compute.internal" -> null
- private_ip = "172.31.12.152" -> null
- public_dns = "ec2-3-109-202-190.ap-south-1.compute.amazonaws.com" -> null
- public_ip = "3.109.202.190" -> null
- region = "ap-south-1" -> null
- secondary_private_ips = [] -> null
- security_groups = [
"web-security-group",
31°C
Sunny
Search
ENG
IN
12:19
30-01-2026

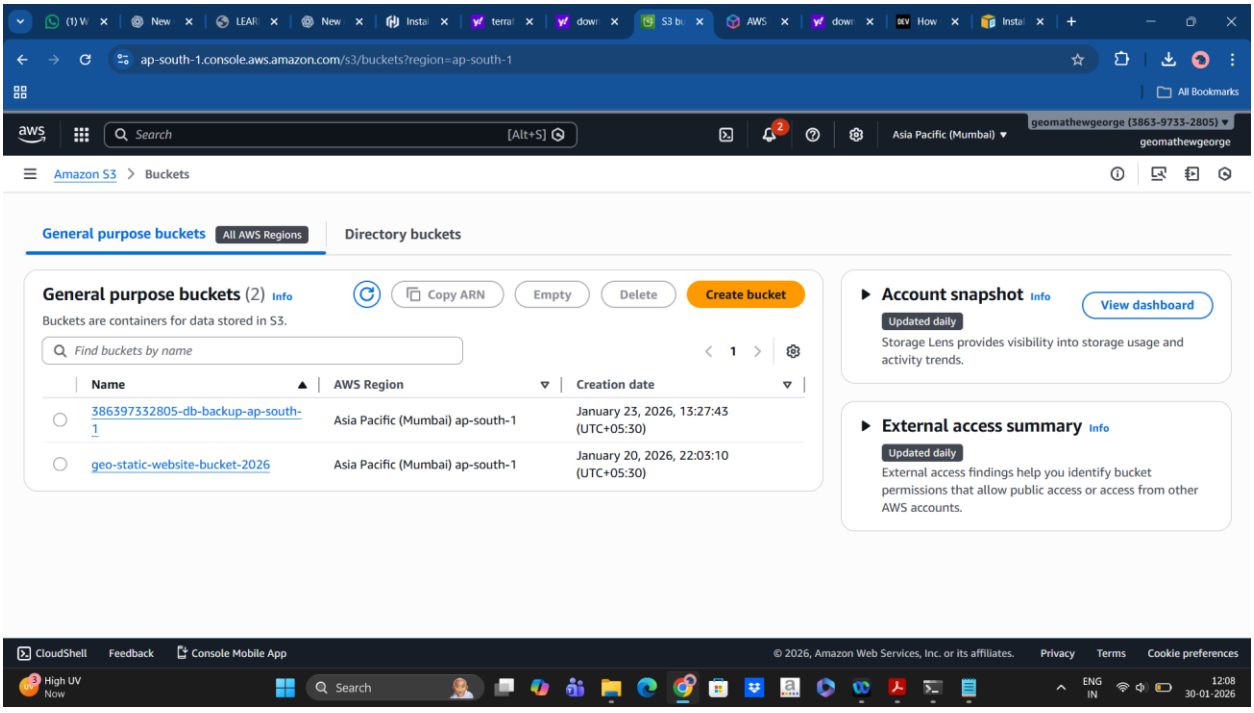

ap-south-1.console.aws.amazon.com/ec2/home?region=ap-south-1#SecurityGroup:groupId=sg-0cf064d31e69a55ef
Search
[Alt+S]
Asia Pacific (Mumbai)
geomathewgeorge (3863-9733-2805)
geomathewgeorge
EC2 > Security Groups > sg-0cf064d31e69a55ef - web-security-group
sg-0cf064d31e69a55ef - web-security-group
Actions
Details
Security group name
web-security-group
Security group ID
sg-0cf064d31e69a55ef
Description
Allow SSH and HTTP
VPC ID
vpc-024cd797321fd1d15
Owner
386397332805
Inbound rules count
2 Permission entries
Outbound rules count
1 Permission entry
Inbound rules
Outbound rules
Sharing
VPC associations
Related resources - new
Tags
Inbound rules (2)
Manage tags
Edit inbound rules
Search
Name
Security group rule ID
IP version
Type
Protocol
Port range
Source
sgr-0ae12fe4c86b613e0
IPv4
HTTP
TCP
80
0.0.0.0/0
sgr-085a180b9392e116f
IPv4
SSH
TCP
22
0.0.0.0/0
CloudShell
Feedback
Console Mobile App
© 2026, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
31°C
Sunny
Search
ENG
IN
12:17
30-01-2026

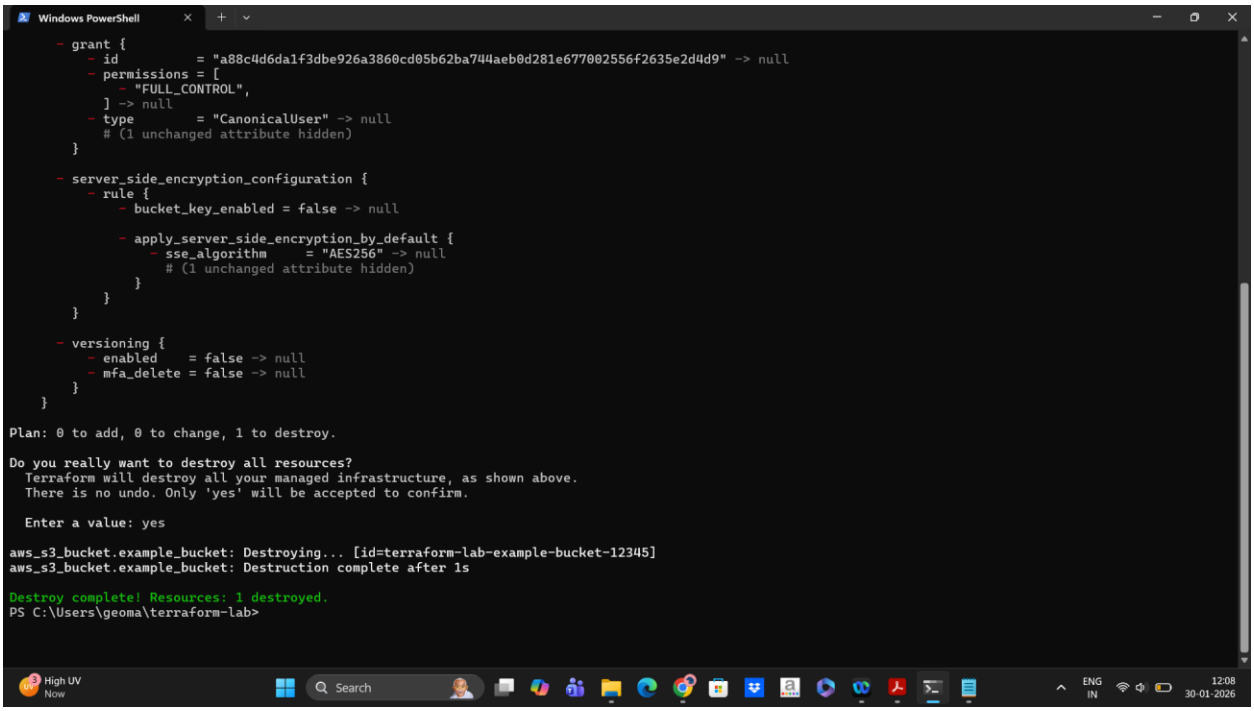

ap-south-1.console.aws.amazon.com/ec2/home?region=ap-south-1#Instances:
Search
[Alt+S]
Asia Pacific (Mumbai)
geomathewgeorge (3863-9733-2805)
geomathewgeorge
EC2 > Instances
Instances (1) Info
Last updated
less than a minute ago
Connect
Instance state
Actions
Launch instances
Find Instance by attribute or tag (case-sensitive)
Running
Name
Instance ID
Instance state
Instance type
Status check
Alarm status
Availability Zone
Public IPv4 DNS
Terraform-EC2
i-0a2f1172855eccd7f
Running
t3.micro
Initializing
View alarms
ap-south-1b
ec2-3-109-202-190.ap-...
Select an instance
CloudShell
Feedback
Console Mobile App
© 2026, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
31°C
Sunny
Search
ENG
IN
12:16
30-01-2026

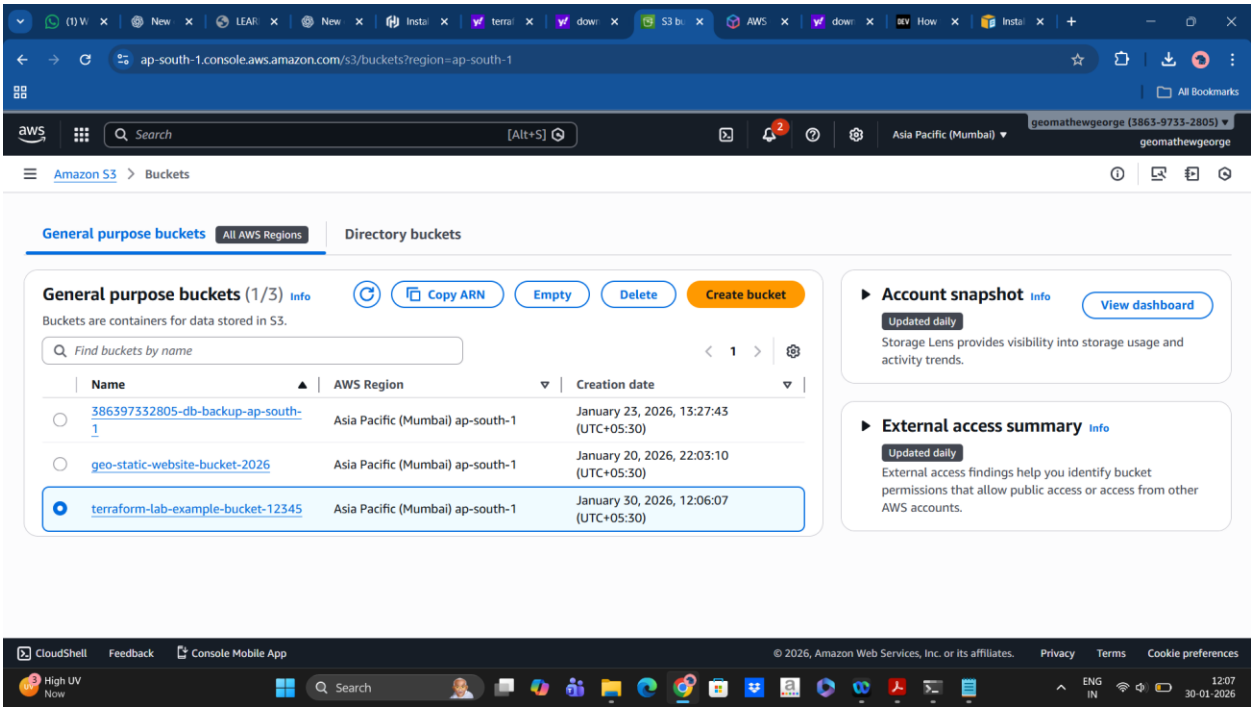

Windows PowerShell
+ capacity_reservation_specification (known after apply)
+ cpu_options (known after apply)
+ ebs_block_device (known after apply)
+ enclave_options (known after apply)
+ ephemeral_block_device (known after apply)
+ instance_market_options (known after apply)
+ maintenance_options (known after apply)
+ metadata_options (known after apply)
+ network_interface (known after apply)
+ primary_network_interface (known after apply)
+ private_dns_name_options (known after apply)
+ root_block_device (known after apply)
}
Plan: 1 to add, 0 to change, 0 to destroy.
Do you want to perform these actions?
Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.
Enter a value: yes
aws_instance.web_server: Creating...
aws_instance.web_server: Still creating... [00m10s elapsed]
aws_instance.web_server: Creation complete after 13s [id=i-0a2f1172855eccd7f]
Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
PS C:\Users\geoma\terraform-lab>
31°C
Sunny
Search
ENG
IN
12:15
30-01-2026

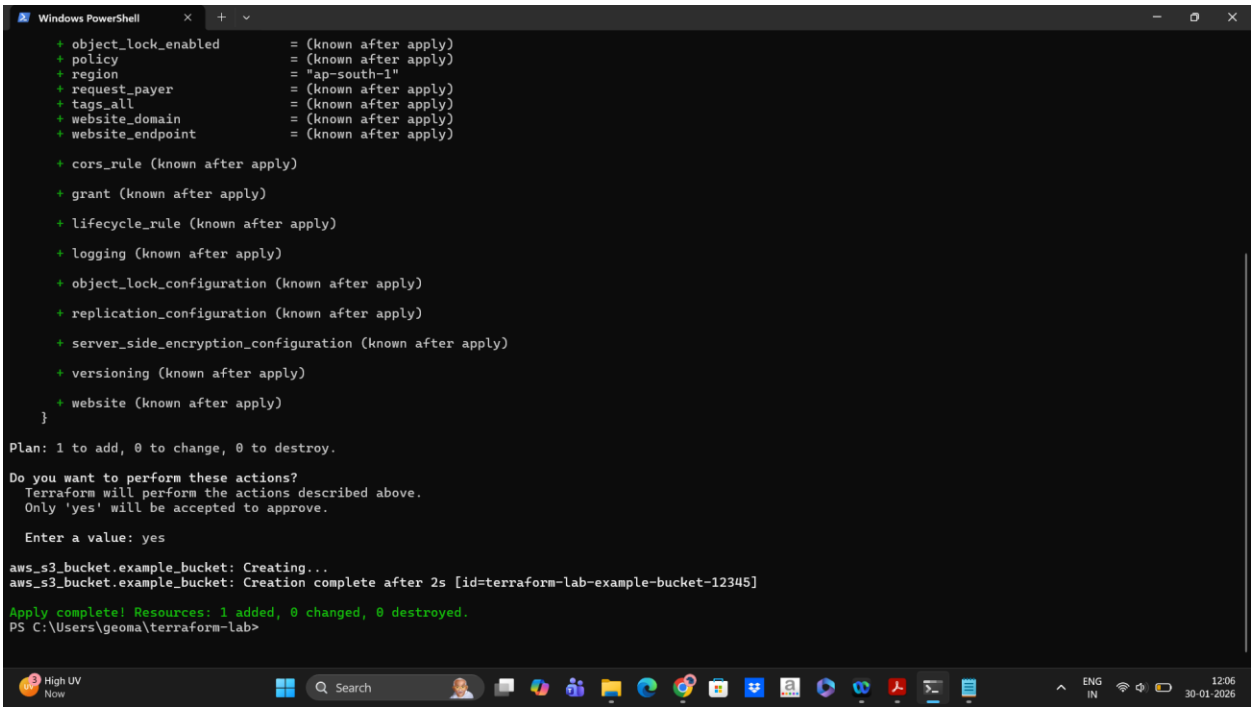

Windows PowerShell
PS C:\Users\geoma\terraform-lab> notepad main.tf
PS C:\Users\geoma\terraform-lab> terraform init
Initializing the backend...
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v6.30.0
Terraform has been successfully initialized!
You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.
If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
PS C:\Users\geoma\terraform-lab> terraform plan
Terraform used the selected providers to generate the following execution
plan. Resource actions are indicated with the following symbols:
+ create
Terraform will perform the following actions:
# aws_instance.web_server will be created
+ resource "aws_instance" "web_server" {
+ ami = "ami-0abcdef1234567890"
+ arn = (known after apply)
+ associate_public_ip_address = (known after apply)
+ availability_zone = (known after apply)
+ disable_api_stop = (known after apply)
+ disable_api_termination = (known after apply)
+ ebs_optimized = (known after apply)
+ enable_primary_ipv6 = (known after apply)
+ force_destroy = false
+ get_password_data = false
+ host_id = (known after apply)
+ host_resource_group_arn = (known after apply)
+ iam_instance_profile = (known after apply)
+ id = (known after apply)
+ instance_initiated_shutdown_behavior = (known after apply)
Show desktop
31°C
Sunny
Search
ENG
IN
12:12
30-01-2026

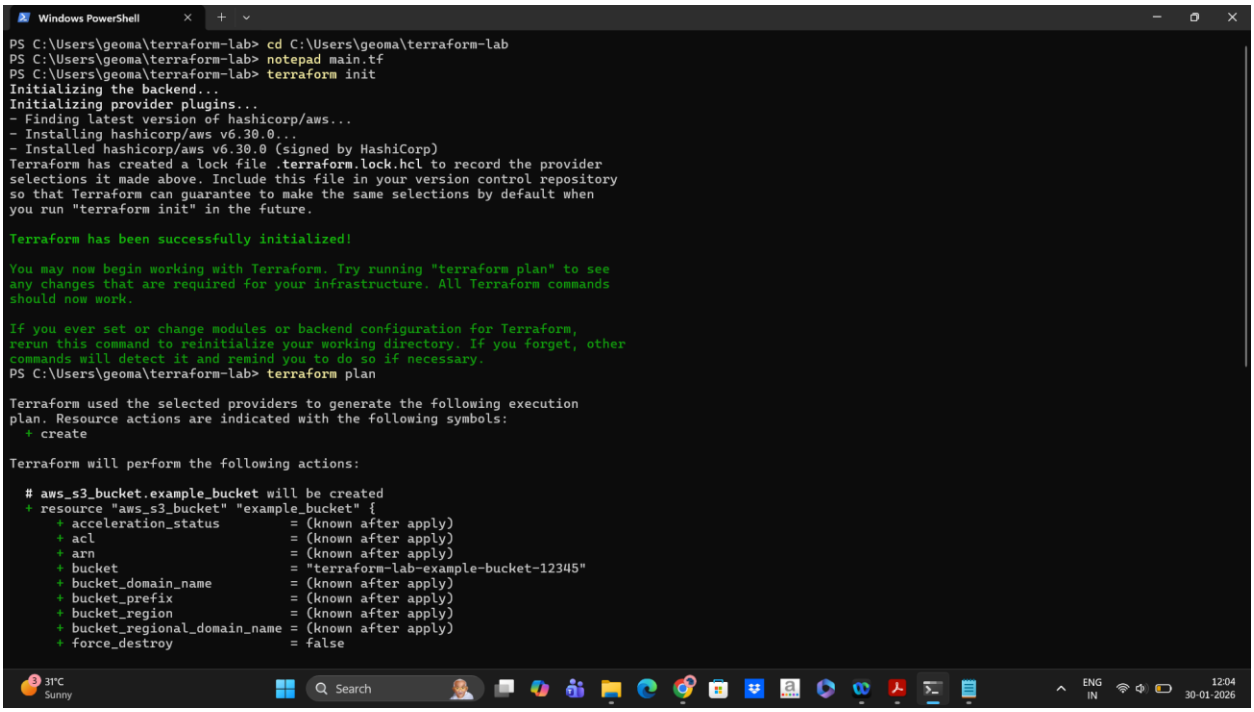

Git
.gitlab-ci.yml
README.md
main.tf
File
Edit
View
provider "aws" {
region = "ap-south-1"
}
resource "aws_security_group" "web_sg" {
name = "web-security-group"
description = "Allow SSH and HTTP"
ingress {
from_port = 22
to_port = 22
protocol = "tcp"
cidr_blocks = ["0.0.0.0/0"]
}
ingress {
from_port = 80
to_port = 80
protocol = "tcp"
cidr_blocks = ["0.0.0.0/0"]
}
egress {
from_port = 0
to_port = 0
protocol = "-1"
cidr_blocks = ["0.0.0.0/0"]
}
}
resource "aws_instance" "web_server" {
ami = "ami-0abcdef1234567890"
instance_type = "t2.micro"
security_groups = [
aws_security_group.web_sg.name
]
tags = {
Name = "Terraform-EC2"
Ln 43, Col 1
720 characters
Plain text
100%
Windows (CRLF)
UTF-8
31°C
Sunny
Search
ENG
IN
12:11
30-01-2026

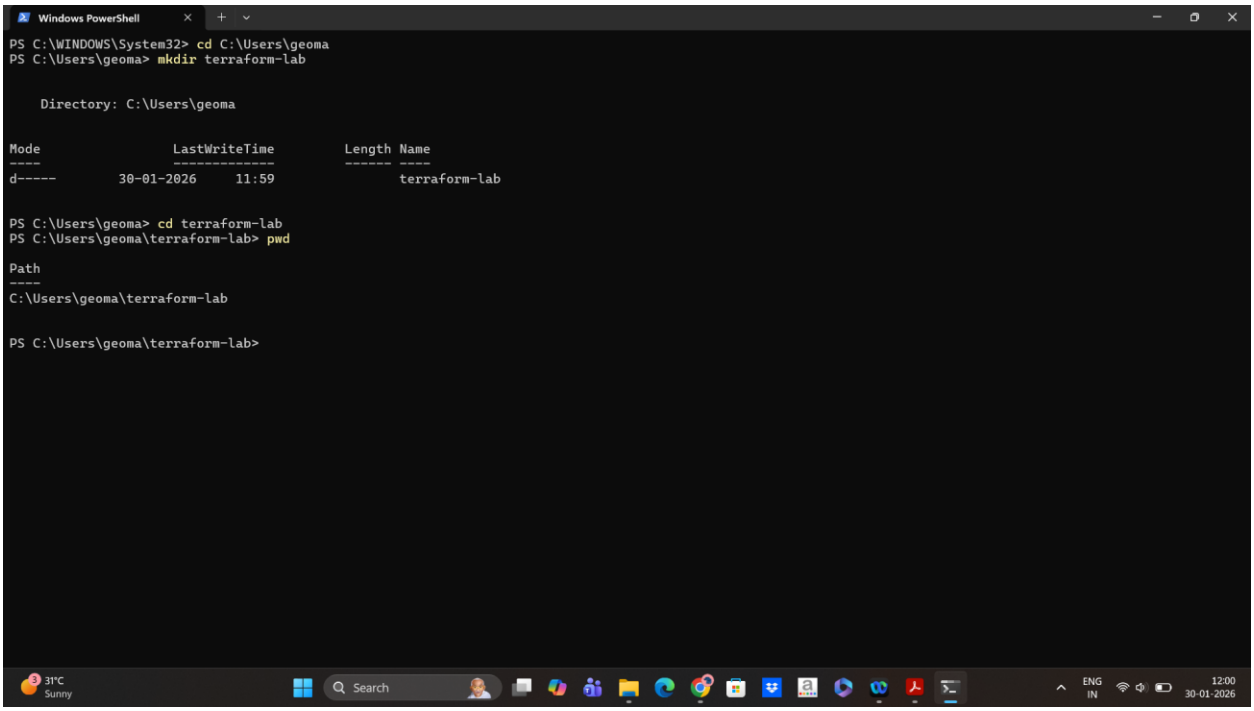

ap-south-1.console.aws.amazon.com/s3/buckets?region=ap-south-1
All Bookmarks
aws
Search
[Alt+S]
Asia Pacific (Mumbai)
geomathewgeorge (3863-9733-2805)
geomathewgeorge
Amazon S3
Buckets
General purpose buckets
All AWS Regions
Directory buckets
General purpose buckets (2) Info
Buckets are containers for data stored in S3.
Copy ARN
Empty
Delete
Create bucket
Find buckets by name
Name
AWS Region
Creation date
386397332805-db-backup-ap-south-1
Asia Pacific (Mumbai) ap-south-1
January 23, 2026, 13:27:43 (UTC+05:30)
geo-static-website-bucket-2026
Asia Pacific (Mumbai) ap-south-1
January 20, 2026, 22:03:10 (UTC+05:30)
Account snapshot Info
View dashboard
Updated daily
Storage Lens provides visibility into storage usage and activity trends.
External access summary Info
Updated daily
External access findings help you identify bucket permissions that allow public access or access from other AWS accounts.
CloudShell
Feedback
Console Mobile App
© 2026, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
High UV
Now
Search
ENG
IN
12:08
30-01-2026

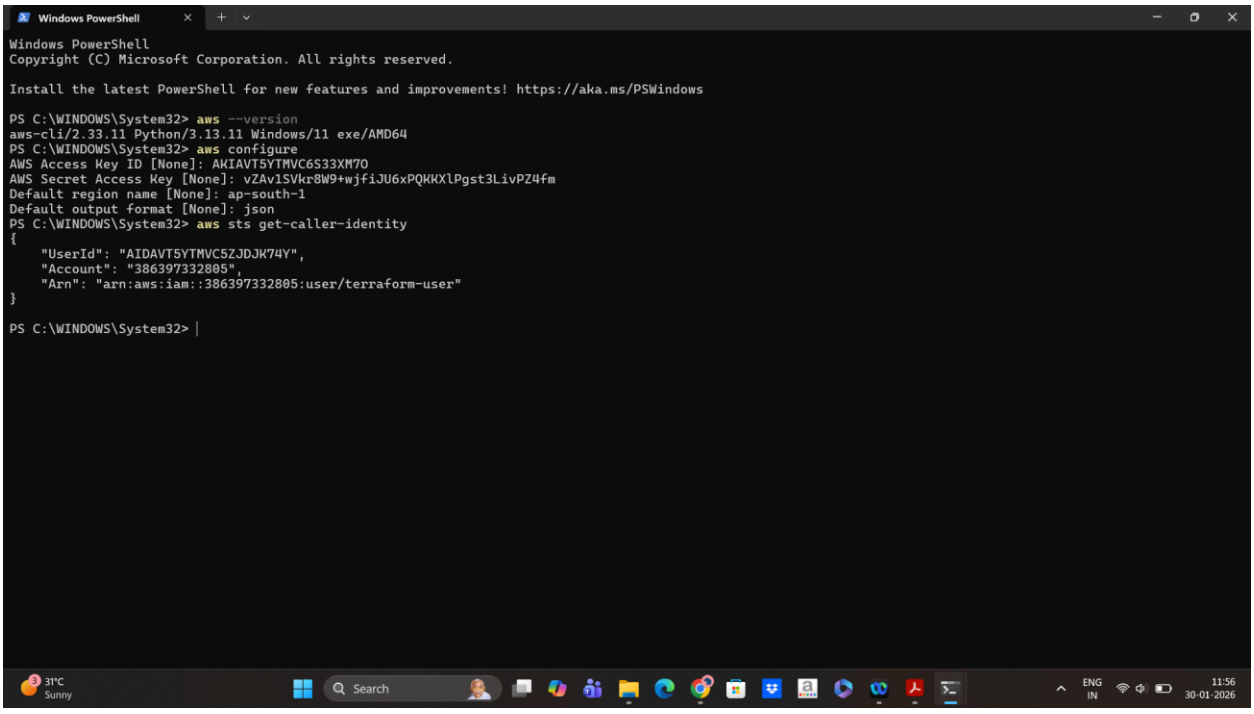

Windows PowerShell
- grant {
- id = "a88c4d6da1f3dbe926a3860cd05b62ba744aeb0d281e677002556f2635e2d4d9" -> null
- permissions = [
- "FULL_CONTROL",
] -> null
- type = "CanonicalUser" -> null
# (1 unchanged attribute hidden)
}
- server_side_encryption_configuration {
- rule {
- bucket_key_enabled = false -> null
- apply_server_side_encryption_by_default {
- sse_algorithm = "AES256" -> null
# (1 unchanged attribute hidden)
}
}
}
- versioning {
- enabled = false -> null
- mfa_delete = false -> null
}
}
Plan: 0 to add, 0 to change, 1 to destroy.
Do you really want to destroy all resources?
Terraform will destroy all your managed infrastructure, as shown above.
There is no undo. Only 'yes' will be accepted to confirm.
Enter a value: yes
aws_s3_bucket.example_bucket: Destroying... [id=terraform-lab-example-bucket-12345]
aws_s3_bucket.example_bucket: Destruction complete after 1s
Destroy complete! Resources: 1 destroyed.
PS C:\Users\geoma\terraform-lab>
High UV Now
Search
ENG IN
12:08 30-01-2026

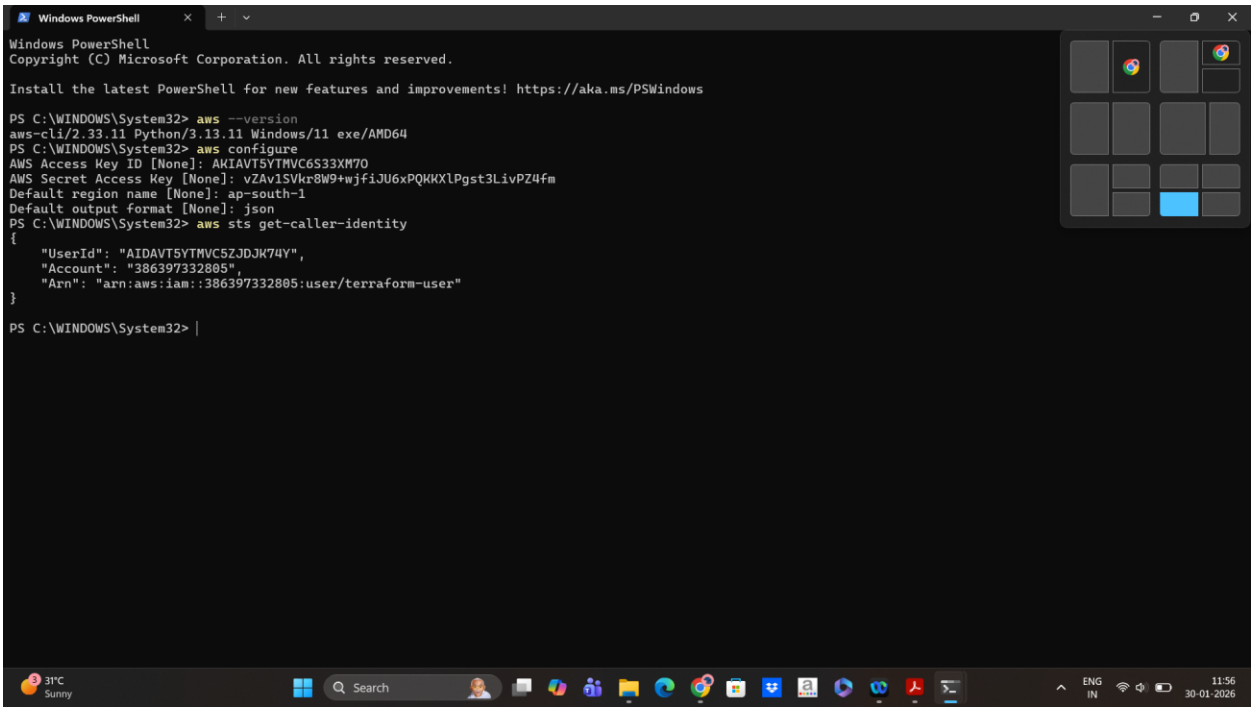

ap-south-1.console.aws.amazon.com/s3/buckets?region=ap-south-1
All Bookmarks
aws
Search [Alt+S]
Asia Pacific (Mumbai)
geomathewgeorge (3863-9733-2805)
geomathewgeorge
Amazon S3 > Buckets
General purpose buckets
All AWS Regions
Directory buckets
General purpose buckets (1/3) Info
Copy ARN
Empty
Delete
Create bucket
Buckets are containers for data stored in S3.
Find buckets by name
Name
AWS Region
Creation date
386397332805-db-backup-ap-south-1
Asia Pacific (Mumbai) ap-south-1
January 23, 2026, 13:27:43 (UTC+05:30)
geo-static-website-bucket-2026
Asia Pacific (Mumbai) ap-south-1
January 20, 2026, 22:03:10 (UTC+05:30)
terraform-lab-example-bucket-12345
Asia Pacific (Mumbai) ap-south-1
January 30, 2026, 12:06:07 (UTC+05:30)
Account snapshot Info
View dashboard
Updated daily
Storage Lens provides visibility into storage usage and activity trends.
External access summary Info
Updated daily
External access findings help you identify bucket permissions that allow public access or access from other AWS accounts.
CloudShell
Feedback
Console Mobile App
© 2026, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
High UV Now
Search
ENG IN
12:07 30-01-2026

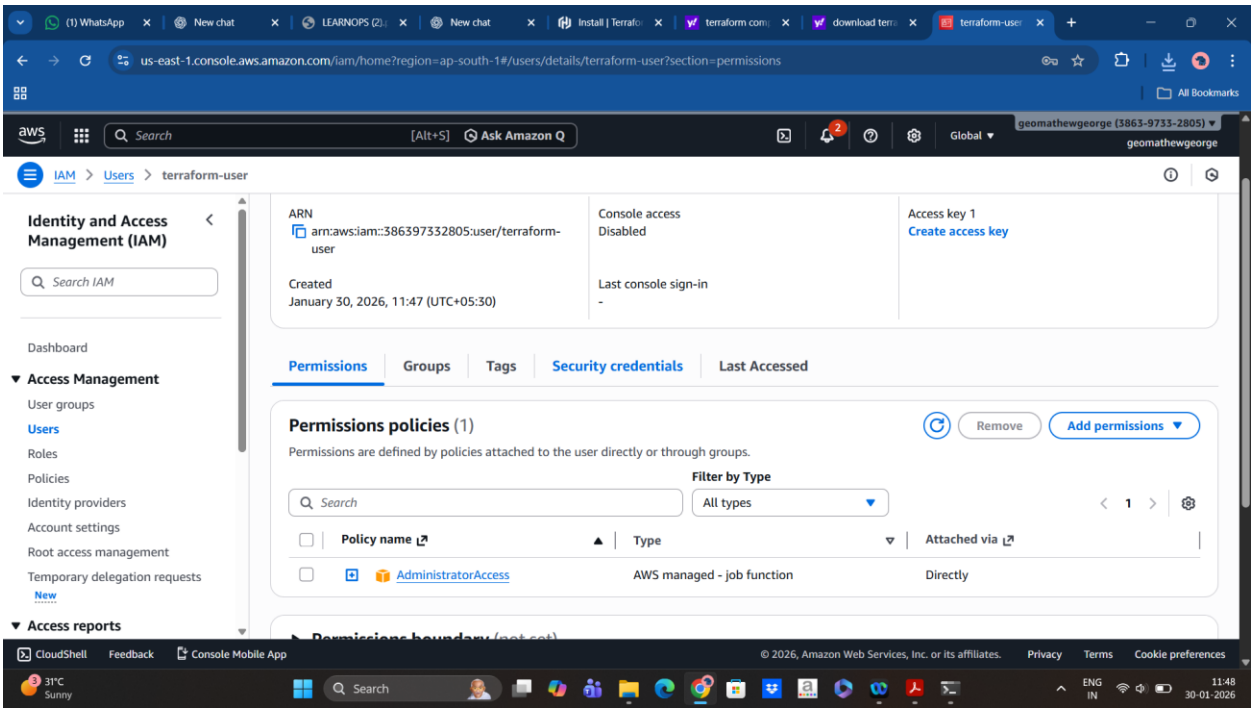

Windows PowerShell
+ object_lock_enabled = (known after apply)
+ policy = (known after apply)
+ region = "ap-south-1"
+ request_payer = (known after apply)
+ tags_all = (known after apply)
+ website_domain = (known after apply)
+ website_endpoint = (known after apply)
+ cors_rule (known after apply)
+ grant (known after apply)
+ lifecycle_rule (known after apply)
+ logging (known after apply)
+ object_lock_configuration (known after apply)
+ replication_configuration (known after apply)
+ server_side_encryption_configuration (known after apply)
+ versioning (known after apply)
+ website (known after apply)
}
Plan: 1 to add, 0 to change, 0 to destroy.
Do you want to perform these actions?
Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.
Enter a value: yes
aws_s3_bucket.example_bucket: Creating...
aws_s3_bucket.example_bucket: Creation complete after 2s [id=terraform-lab-example-bucket-12345]
Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
PS C:\Users\geoma\terraform-lab>
High UV
Now
Search
ENG
IN
12:06
30-01-2026

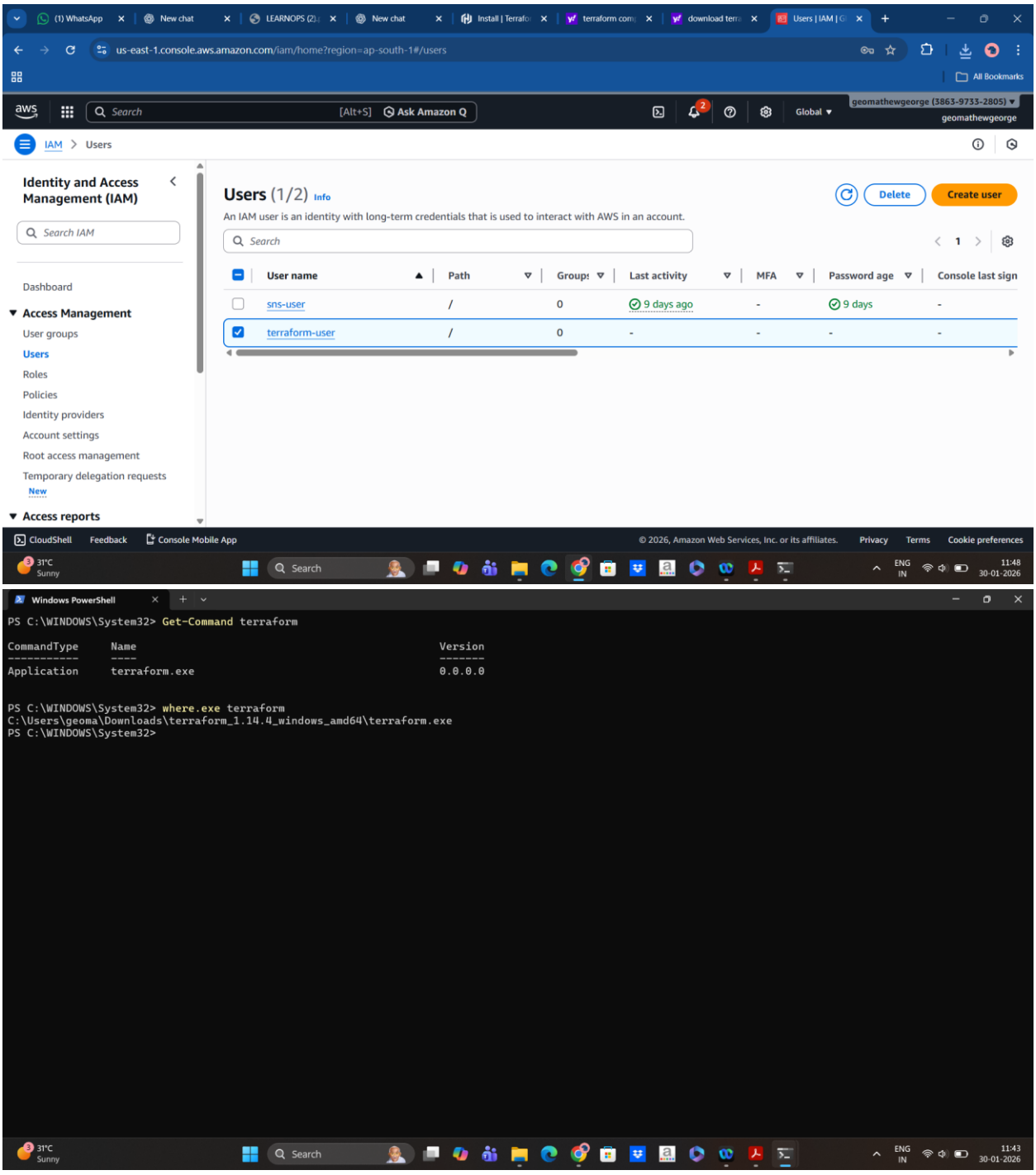

Windows PowerShell
PS C:\Users\geoma\terraform-lab> cd C:\Users\geoma\terraform-lab
PS C:\Users\geoma\terraform-lab> notepad main.tf
PS C:\Users\geoma\terraform-lab> terraform init
Initializing the backend...
Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v6.30.0...
- Installed hashicorp/aws v6.30.0 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.
Terraform has been successfully initialized!
You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.
If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
PS C:\Users\geoma\terraform-lab> terraform plan
Terraform used the selected providers to generate the following execution
plan. Resource actions are indicated with the following symbols:
+ create
Terraform will perform the following actions:
# aws_s3_bucket.example_bucket will be created
+ resource "aws_s3_bucket" "example_bucket" {
+ acceleration_status = (known after apply)
+ acl = (known after apply)
+ arn = (known after apply)
+ bucket = "terraform-lab-example-bucket-12345"
+ bucket_domain_name = (known after apply)
+ bucket_prefix = (known after apply)
+ bucket_region = (known after apply)
+ bucket_regional_domain_name = (known after apply)
+ force_destroy = false
31°C
Sunny
Search
ENG
IN
12:04
30-01-2026

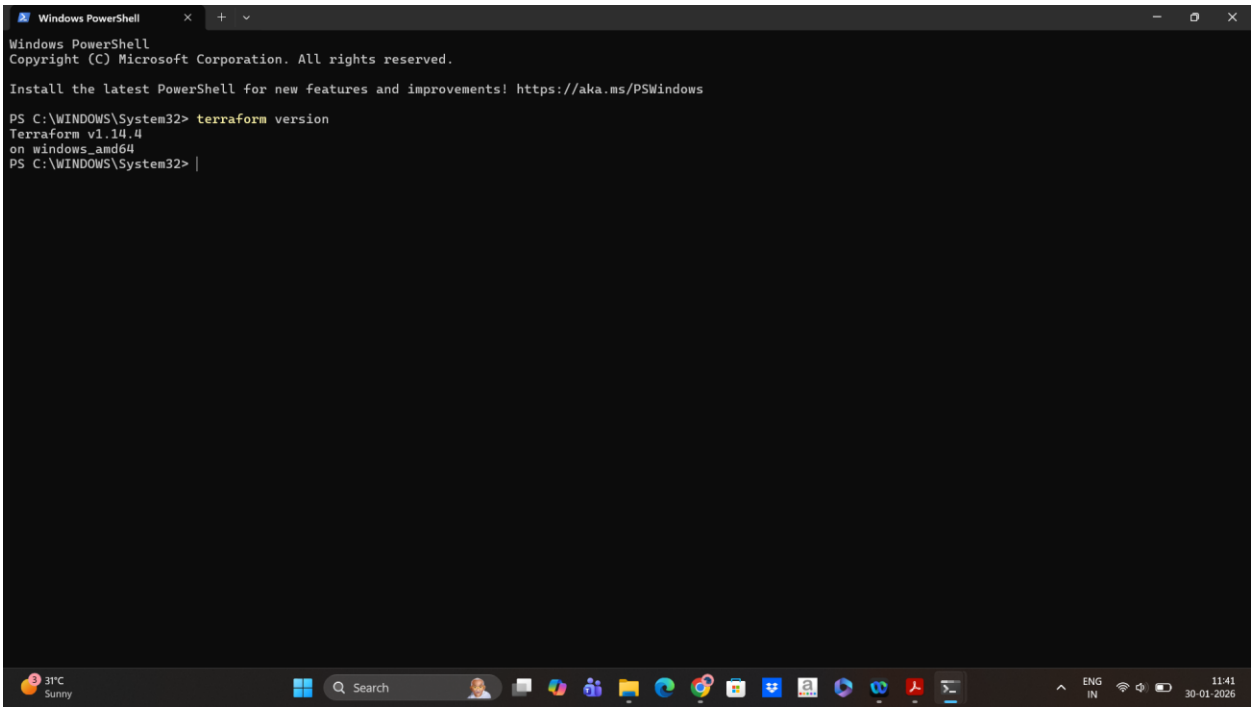

Windows PowerShell
PS C:\WINDOWS\System32> cd C:\Users\geoma
PS C:\Users\geoma> mkdir terraform-lab
Directory: C:\Users\geoma
Mode LastWriteTime Length Name
d----- 30-01-2026 11:59 terraform-lab
PS C:\Users\geoma> cd terraform-lab
PS C:\Users\geoma\terraform-lab> pwd
Path
C:\Users\geoma\terraform-lab
PS C:\Users\geoma\terraform-lab>
31°C Sunny
Search
ENG IN
12:00 30-01-2026


Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.
Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows
PS C:\WINDOWS\System32> aws --version
aws-cli/2.33.11 Python/3.13.11 Windows/11 exe/AMD64
PS C:\WINDOWS\System32> aws configure
AWS Access Key ID [None]: AKIAVT5YTMVC6S33XM7O
AWS Secret Access Key [None]: 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
Default region name [None]: ap-south-1
Default output format [None]: json
PS C:\WINDOWS\System32> aws sts get-caller-identity
{
"UserId": "AIDAVT5YTMVC5ZJDJK74Y",
"Account": "386397332805",
"Arn": "arn:aws:iam::386397332805:user/terraform-user"
}
PS C:\WINDOWS\System32>
31°C Sunny
Search
ENG IN
11:56 30-01-2026


Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.
Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows
PS C:\WINDOWS\System32> aws --version
aws-cli/2.33.11 Python/3.13.11 Windows/11 exe/AMD64
PS C:\WINDOWS\System32> aws configure
AWS Access Key ID [None]: AKIAVT5YTMVC6S33XM7O
AWS Secret Access Key [None]: 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
Default region name [None]: ap-south-1
Default output format [None]: json
PS C:\WINDOWS\System32> aws sts get-caller-identity
{
"UserId": "AIDAVT5YTMVC5ZJDJK74Y",
"Account": "386397332805",
"Arn": "arn:aws:iam::386397332805:user/terraform-user"
}
PS C:\WINDOWS\System32>
31°C
Sunny
Search
ENG
IN
11:56
30-01-2026


(1) WhatsApp
New chat
LEARNOPS (2)
New chat
Install | Terraf
terraform com
download terra
terraform-user
us-east-1.console.aws.amazon.com/iam/home?region=ap-south-1#/users/details/terraform-user?section=permissions
All Bookmarks
aws
Search
[Alt+S]
Ask Amazon Q
Global
geomathewgeorge (3863-9733-2805)
geomathewgeorge
IAM > Users > terraform-user
Identity and Access Management (IAM)
Search IAM
Dashboard
Access Management
User groups
Users
Roles
Policies
Identity providers
Account settings
Root access management
Temporary delegation requests
New
Access reports
ARN
arn:aws:iam::386397332805:user/terraform-user
Console access
Disabled
Access key 1
Create access key
Created
January 30, 2026, 11:47 (UTC+05:30)
Last console sign-in
-
Permissions
Groups
Tags
Security credentials
Last Accessed
Permissions policies (1)
Permissions are defined by policies attached to the user directly or through groups.
Remove
Add permissions
Filter by Type
Search
All types
Policy name
Type
Attached via
AdministratorAccess
AWS managed - job function
Directly
CloudShell
Feedback
Console Mobile App
© 2026, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
31°C
Sunny
Search
ENG
IN
11:48
30-01-2026


us-east-1.console.aws.amazon.com/iam/home?region=ap-south-1#/users
Search
[Alt+S]
Ask Amazon Q
Global
geomathewgeorge (3863-9733-2805)
geomathewgeorge
IAM > Users
Identity and Access Management (IAM)
Search IAM
Dashboard
Access Management
User groups
Users
Roles
Policies
Identity providers
Account settings
Root access management
Temporary delegation requests
New
Access reports
Users (1/2) Info
An IAM user is an identity with long-term credentials that is used to interact with AWS in an account.
Delete
Create user
Search
User name
Path
Groups
Last activity
MFA
Password age
Console last sign
sns-user
/
0
9 days ago
-
9 days
-
terraform-user
/
0
-
-
-
-
CloudShell
Feedback
Console Mobile App
© 2026, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
31°C
Sunny
Search
ENG
IN
11:48
30-01-2026
Windows PowerShell
PS C:\WINDOWS\System32> Get-Command terraform
CommandType Name Version
Application terraform.exe 0.0.0.0
PS C:\WINDOWS\System32> where.exe terraform
C:\Users\geoma\Downloads\terraform_1.14.4_windows_amd64\terraform.exe
PS C:\WINDOWS\System32>
31°C
Sunny
Search
ENG
IN
11:43
30-01-2026



Terraform Is Used in Real-World DevOps Environments

In real companies, Terraform is used to **create, manage, and control cloud infrastructure automatically** instead of doing everything manually in the cloud console.

Terraform Is Used in Daily DevOps Work

Automating cloud infrastructure

Terraform is used to create servers, networks, and storage automatically.

Example: A company uses Terraform to create EC2 servers, load balancers, and databases with one command instead of clicking many options in AWS.

Keeping all environments the same

Terraform helps make sure development, testing, and production environments look the same.

Example: The same Terraform code creates identical setups for Dev, QA, and Production, reducing “works on my system” problems.

Working with CI/CD pipelines

Terraform is often connected to CI/CD tools.

Example: When code is pushed to GitHub, Jenkins runs Terraform to check changes and deploy infrastructure automatically after approval.

Using Git for infrastructure
Terraform files are stored in Git just like application code.
Example: If an infrastructure change causes a problem, the team can roll back to an earlier version using Git.

Reducing human mistakes
Manual setup can cause errors. Terraform avoids this by using tested code.
Example: Security rules are written once in Terraform and reused everywhere, preventing wrong firewall settings.

Supporting multiple cloud platforms
Terraform can manage resources in AWS, Azure, and Google Cloud together.
Example: A company runs servers in AWS and backups in another cloud using the same Terraform workflow.

Fast creation and cleanup of environments
Terraform allows quick setup and removal of infrastructure.
Example: A testing environment is created in the morning and destroyed at night to save cloud costs.

Enforcing standards and security
Terraform ensures rules are followed every time.
Example: Every server created must have encryption and monitoring enabled by default.

Tools Terraform Works With in Real Projects

Terraform does not work alone. It is used together with other DevOps tools.

- Git is used to track and review infrastructure code
- Jenkins or GitLab CI runs Terraform automatically
- Docker and Kubernetes handle application deployment
- AWS services like EC2, S3, Load Balancers, and Auto Scaling are managed using Terraform

Real-World Scenario Example

A company building an **online shopping website** uses Terraform like this:

- Developers push Terraform code to Git
- Jenkins runs Terraform plan to show upcoming changes
- After approval, Terraform creates or updates AWS infrastructure
- Kubernetes deploys the application on those servers
- If traffic increases, infrastructure can be scaled quickly using Terraform

Personal Understanding and Learning Outcomes

From this learning, I understood that infrastructure can be managed just like software using code. This study helped me build a strong foundation in Terraform and DevOps practices.

My key learning outcomes are:

- I understood why Infrastructure as Code is important in DevOps
- I learned how to write and understand Terraform code using HCL
- I gained hands-on experience in creating AWS cloud resources using Terraform
- I understood how Terraform state works and how environments are kept separate
- I learned best practices for building reusable and scalable infrastructure
- I became confident in reading, editing, and running Terraform configurations
- I gained awareness of real-world DevOps problems and how Terraform helps solve them

Overall, this learning helped me connect development and operations work. It also strengthened my skills in cloud automation and DevOps engineering.

3)Jenkins

Topics

During this learning phase, I studied **Jenkins**, a popular DevOps automation tool. The focus was on how Jenkins is used for Continuous Integration (CI) and Continuous Delivery or Deployment (CD).

The main topics covered include:

- Basics of DevOps and Continuous Integration
- Jenkins architecture and overall workflow
- Installing Jenkins on different operating systems
- Using the Jenkins dashboard and managing jobs
- Configuring system tools such as JDK, Git, and Maven
- Creating and managing different types of build jobs
- Integrating Jenkins with Git and GitHub
- Build triggers and scheduling jobs
- Running automated tests using Jenkins
- Using post-build actions and notifications
- Jenkins plugins and extensibility
- Distributed builds using master and agent nodes
- Basic security concepts in Jenkins
- Jenkins pipelines and CI/CD concepts
- Jenkins maintenance, backup, and upgrades

2. Key Concepts Understood from Jenkins

Continuous Integration (CI)

I learned that Continuous Integration means developers regularly push their code to a shared repository, and Jenkins automatically builds and tests the code. This helps find errors early and reduces problems when combining code from multiple developers.

Jenkins Architecture

Jenkins follows a master and agent model.

- The master manages jobs, schedules builds, and provides the user interface
- Agent nodes run the actual build jobs, which allows faster and parallel execution

Jenkins Installation and Configuration

I learned different ways to install Jenkins, such as using a WAR file, native installers, or running it as a service. I also understood the importance of the Jenkins home directory, which stores job data, plugins, logs, and build history.

Build Jobs

Jenkins supports different job types including freestyle jobs, Maven jobs, and pipeline jobs. Each job can perform tasks like pulling code, building the project, running tests, and performing post-build actions.

Source Code Management Integration

Jenkins integrates with Git using plugins. It automatically pulls code from repositories and starts builds when changes are made, ensuring continuous integration.

Build Triggers

I learned that Jenkins builds can be triggered in multiple ways, such as automatically when code changes, at scheduled times, manually, or through other Jenkins jobs.

Automated Testing

Jenkins supports automated testing tools. Test results are shown clearly in the dashboard, making it easy to identify failed builds.

Plugins and Extensibility

Jenkins has a large plugin ecosystem. Plugins allow Jenkins to work with tools like Git, Maven, Docker, email services, and messaging tools.

Notifications and Reporting

Jenkins can notify teams about build success or failure through email or other notification tools, which improves communication.

Security

I learned how Jenkins supports user authentication and role-based access control to protect CI/CD pipelines from unauthorized access.

Pipelines

Jenkins pipelines allow CI/CD workflows to be written as code using a Jenkinsfile. This makes automation more reliable, version-controlled, and consistent across environments.

Jenkins Practical Work for DevOps Project

Project Title

Implementation of Continuous Integration using Jenkins

Practical 1: Jenkins Installation and Initial Setup

Objective

To install Jenkins and verify its successful operation.

Practical Work

In this practical, Jenkins was installed on the system to act as a Continuous Integration server. Java was first installed since Jenkins requires a Java runtime environment. After downloading Jenkins, it was started as a service and accessed through a web browser.

Once Jenkins was launched, the initial setup wizard was completed. The administrator password was retrieved from the Jenkins home directory, basic plugins were installed, and an admin user was created. After setup, the Jenkins dashboard was accessed successfully.

Outcome

Jenkins server was installed, configured, and ready for CI/CD operations.

- Jenkins running in browser
- Jenkins dashboard
- Admin user creation screen

Practical 2: Jenkins Tool Configuration

Objective

To configure required build tools in Jenkins.

Practical Work

In this practical, global tools required for builds were configured. Java Development Kit (JDK), Git, and Maven were configured using **Manage Jenkins → Global Tool Configuration**.

Jenkins was configured either to use local installations or automatically install tools during builds. This ensures consistent build environments across different jobs and systems.

Outcome

Jenkins was successfully configured with required build tools.

- Global Tool Configuration page
- JDK configuration
- Git and Maven configuration

Practical 3: GitHub Integration with Jenkins

Objective

To integrate Jenkins with GitHub for source code management.

Practical Work

A GitHub repository containing a sample project was created. Jenkins Git plugin was installed and configured. A new Jenkins job was created and connected to the GitHub repository using the repository URL.

Credentials were added securely in Jenkins to allow authenticated access. Jenkins was tested by pulling source code from GitHub successfully.

Outcome

Jenkins was successfully integrated with GitHub for automated source code retrieval.

- GitHub repository

- Jenkins job SCM configuration
- Credentials configuration

Practical 4: Creating a Jenkins Build Job

Objective

To create and execute a Jenkins build job.

Practical Work

A **Freestyle Project** was created in Jenkins. The job was configured to:

- Pull source code from GitHub
- Execute build steps using Maven
- Compile the application

The job was manually triggered and Jenkins executed the build successfully. Build logs were reviewed to verify correct execution.

Outcome

A working Jenkins build job was created and executed successfully.

- Job configuration page
- Build console output
- Successful build status

Practical 5: Automated Build Trigger Using SCM Polling / Webhook

Objective

To automatically trigger builds when code changes occur.

Practical Work

The Jenkins job was configured with **SCM Polling** or **GitHub Webhook**. Whenever a developer commits code to the repository, Jenkins automatically detects the change and triggers a new build.

A test commit was made to the repository, and Jenkins automatically started the build without manual intervention.

Outcome

Automatic CI pipeline was successfully implemented.

- Webhook or SCM polling configuration
- Git commit history
- Auto-triggered Jenkins build

Practical 6: Automated Testing with Jenkins

Objective

To execute automated tests and display results.

Practical Work

JUnit test cases were added to the project. Jenkins was configured to execute tests as part of the build process. Test results were published using post-build actions.

Jenkins displayed test reports and clearly indicated passed and failed test cases using visual indicators.

Outcome

Automated testing was successfully integrated into the CI pipeline.

- Test execution logs
- Test report page

- Build status indicators

Practical 7: Post-Build Actions and Notifications

Objective

To configure notifications after build completion.

Practical Work

Post-build actions were configured to archive build artifacts and send email notifications. Jenkins was connected to an SMTP server, and notifications were sent for both successful and failed builds.

This helped improve team communication by instantly informing developers about build results.

Outcome

Post-build automation and notifications were implemented.

- Post-build action configuration
- Email notification setup
- Sample notification

Practical 8: Jenkins Pipeline Implementation

Objective

To implement CI using Jenkins Pipeline (Pipeline as Code).

Practical Work

A Jenkins Pipeline job was created using a **Jenkinsfile** stored in the GitHub repository. The pipeline defined stages such as:

- Source code checkout
- Build
- Test
- Deployment (optional)

The pipeline executed automatically and displayed stage-wise progress.

Outcome

Pipeline-based CI/CD workflow was successfully implemented.

- Jenkinsfile
- Pipeline execution view
- Stage-wise output

Practical 9: Jenkins Security Configuration

Objective

To secure Jenkins using authentication and authorization.

Practical Work

Security was enabled in Jenkins by configuring user authentication and role-based access control. Different permissions were assigned to users to restrict access.

This ensured that only authorized users could modify jobs and configurations.

Outcome

Jenkins was secured against unauthorized access.

- Security configuration page
- User management
- Role assignment

Practical 10: Jenkins Backup and Maintenance

Objective

To understand Jenkins maintenance and backup.

Practical Work

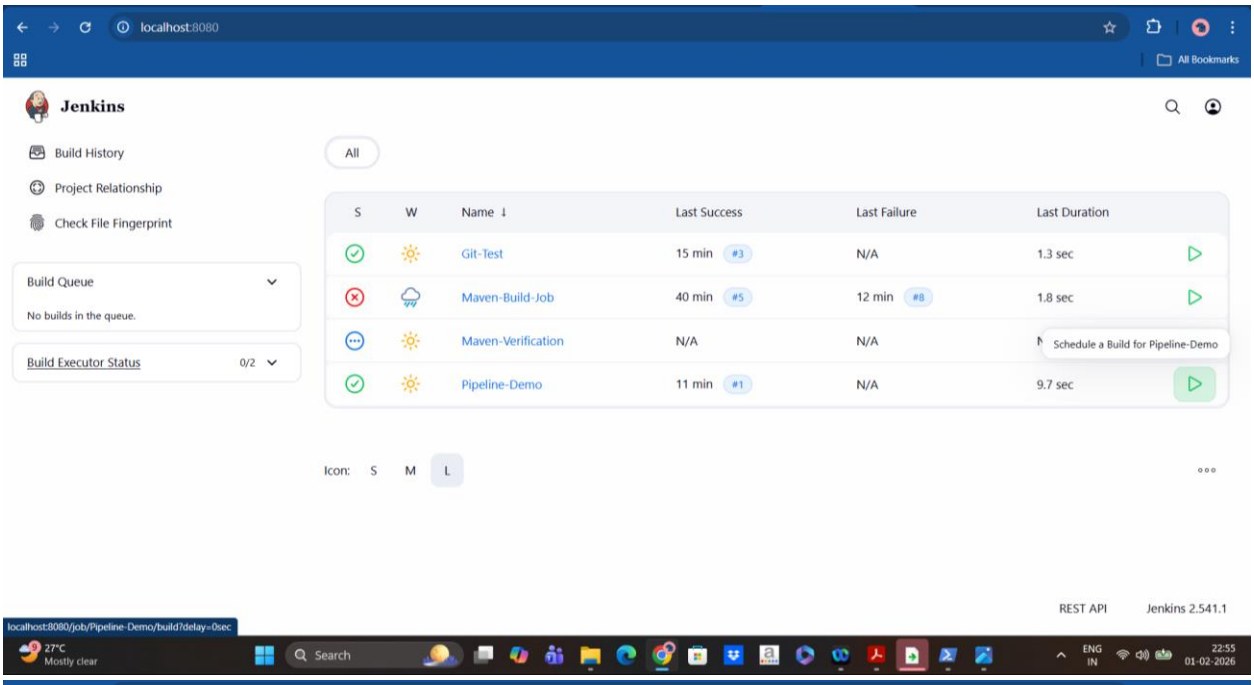
The Jenkins home directory was backed up manually to prevent data loss. Plugin updates were performed, and Jenkins logs were reviewed for maintenance purposes.

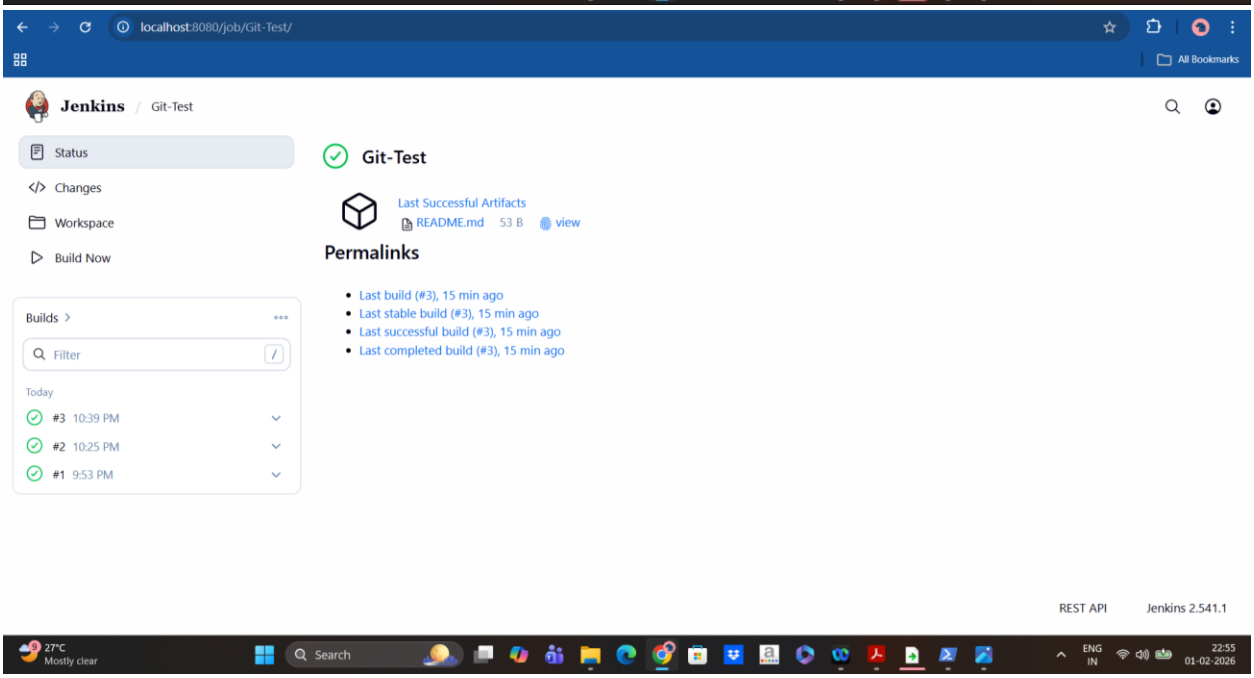
This practice ensures Jenkins reliability in production environments.

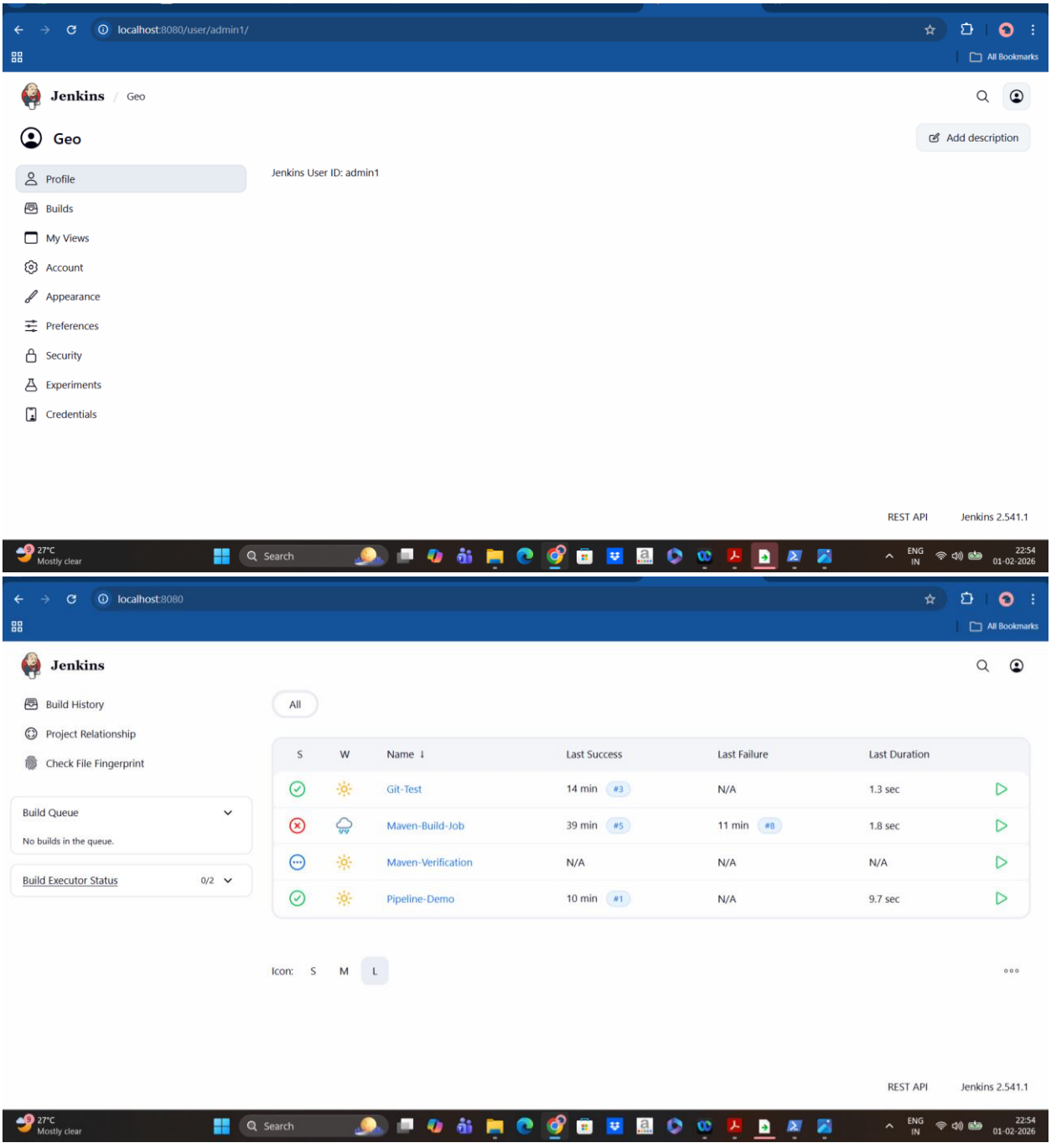
Outcome

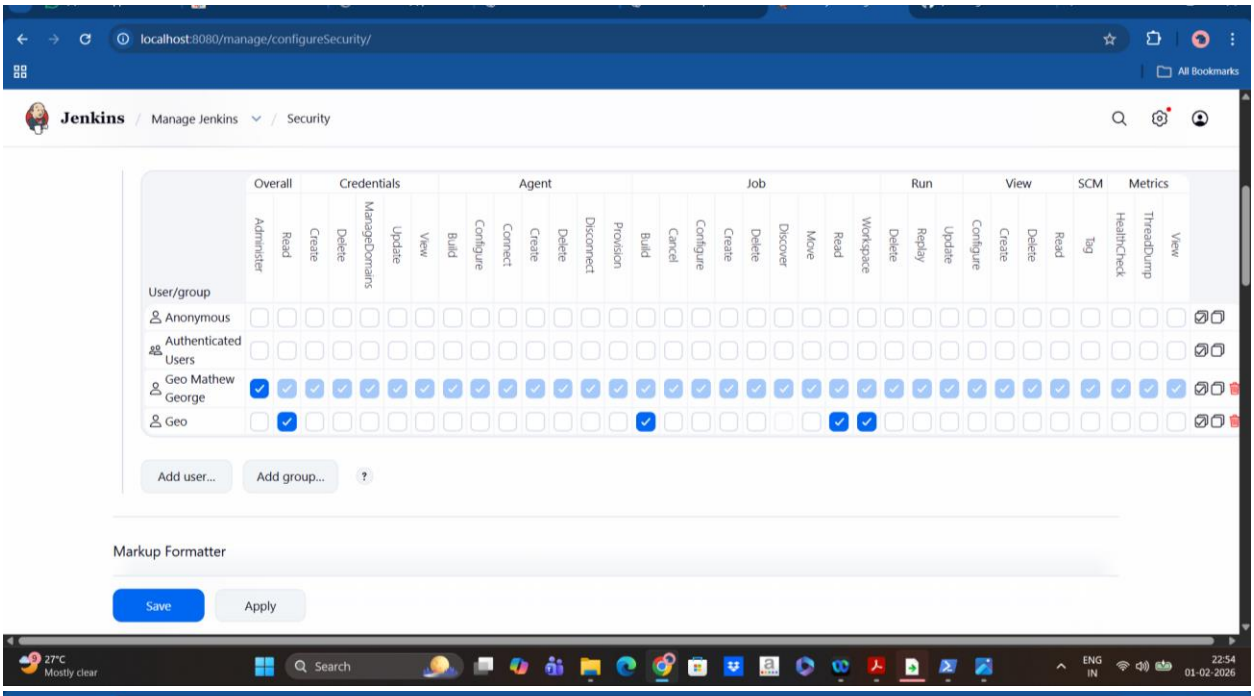
Basic Jenkins maintenance and backup strategies were implemented.

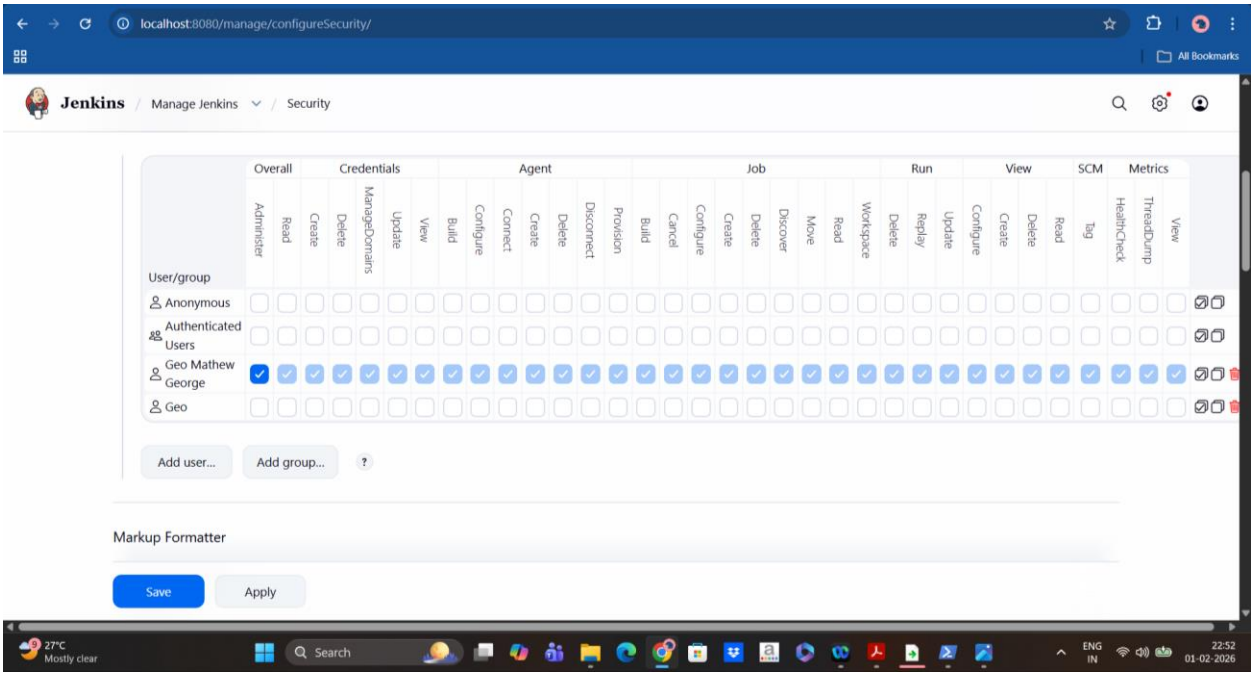
- Jenkins home directory
- Backup files
- Plugin update page

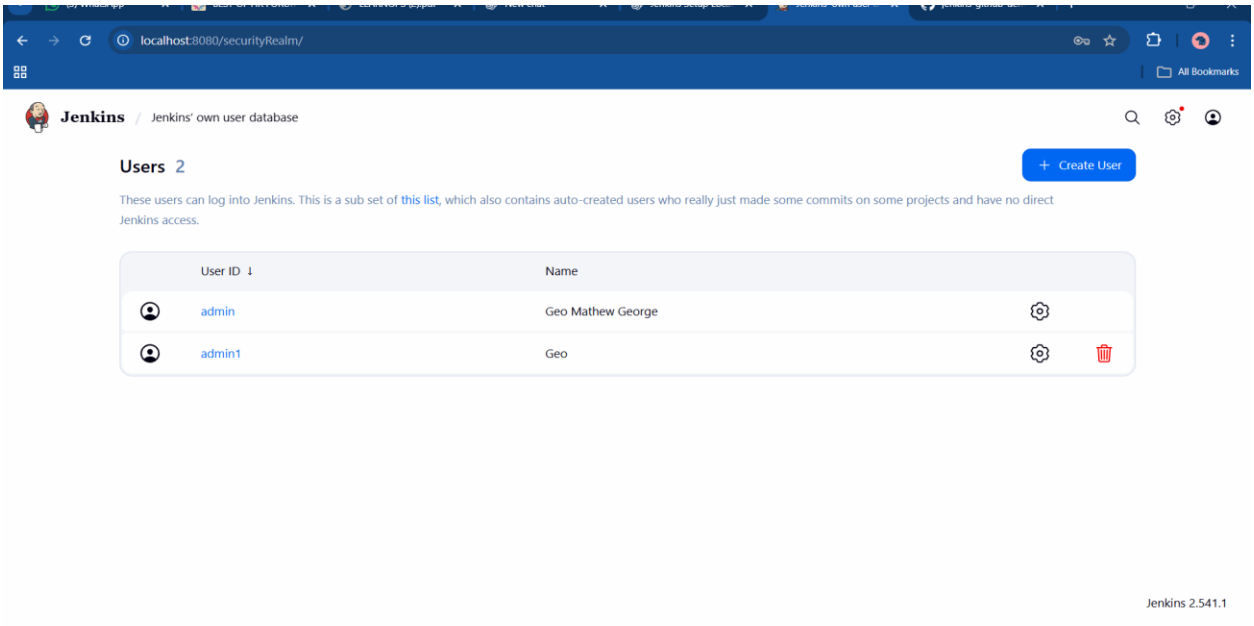

localhost:8080
All Bookmarks
Jenkins
Build History
Project Relationship
Check File Fingerprint
Build Queue
No builds in the queue.
Build Executor Status 0/2
All
S W Name ↓ Last Success Last Failure Last Duration
Git-Test 15 min #3 N/A 1.3 sec
Maven-Build-Job 40 min #5 12 min #8 1.8 sec
Maven-Verification N/A N/A
Schedule a Build for Pipeline-Demo
Pipeline-Demo 11 min #1 N/A 9.7 sec
Icon: S M L
REST API Jenkins 2.541.1
localhost:8080/job/Pipeline-Demo/build?delay=0sec
27°C Mostly clear
Search
ENG IN
22:55 01-02-2026

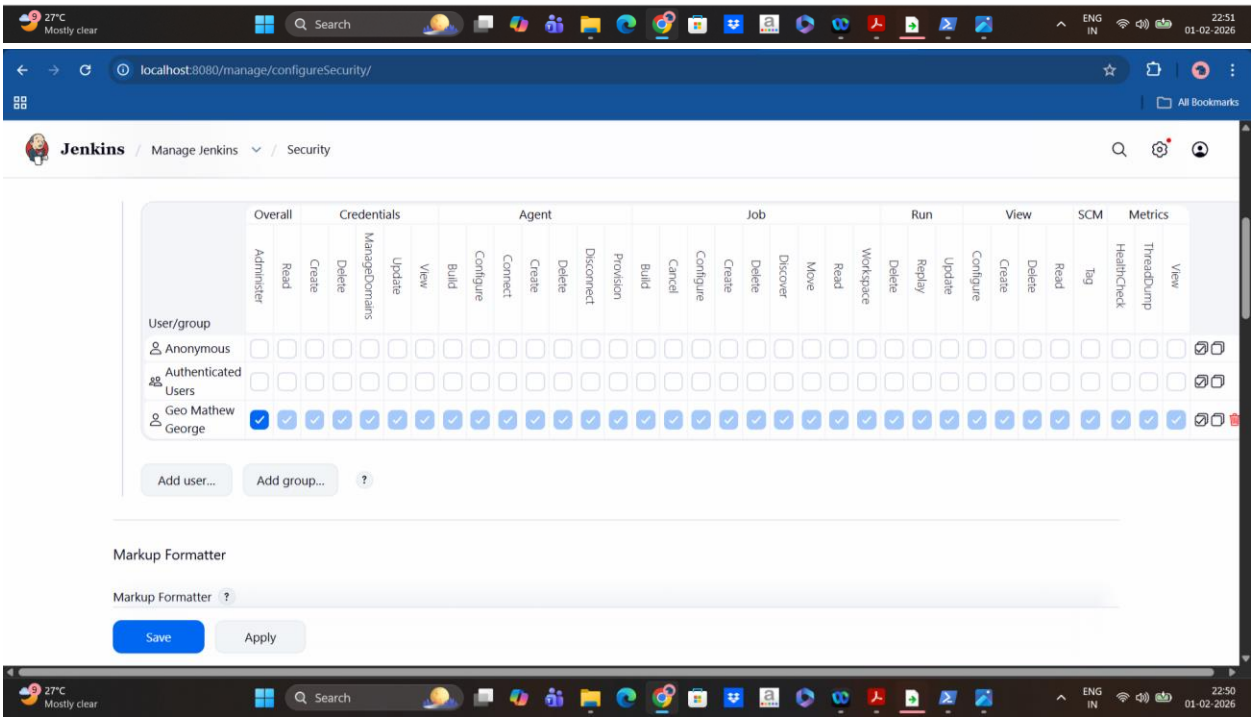

localhost:8080/job/Git-Test/
All Bookmarks
Jenkins / Git-Test
Status
Changes
Workspace
Build Now
Builds
Filter
Today
#3 10:39 PM
#2 10:25 PM
#1 9:53 PM
Git-Test
Last Successful Artifacts
README.md 53 B view
Permalinks
Last build (#3), 15 min ago
Last stable build (#3), 15 min ago
Last successful build (#3), 15 min ago
Last completed build (#3), 15 min ago
REST API Jenkins 2.541.1
27°C Mostly clear
Search
ENG IN
22:55 01-02-2026

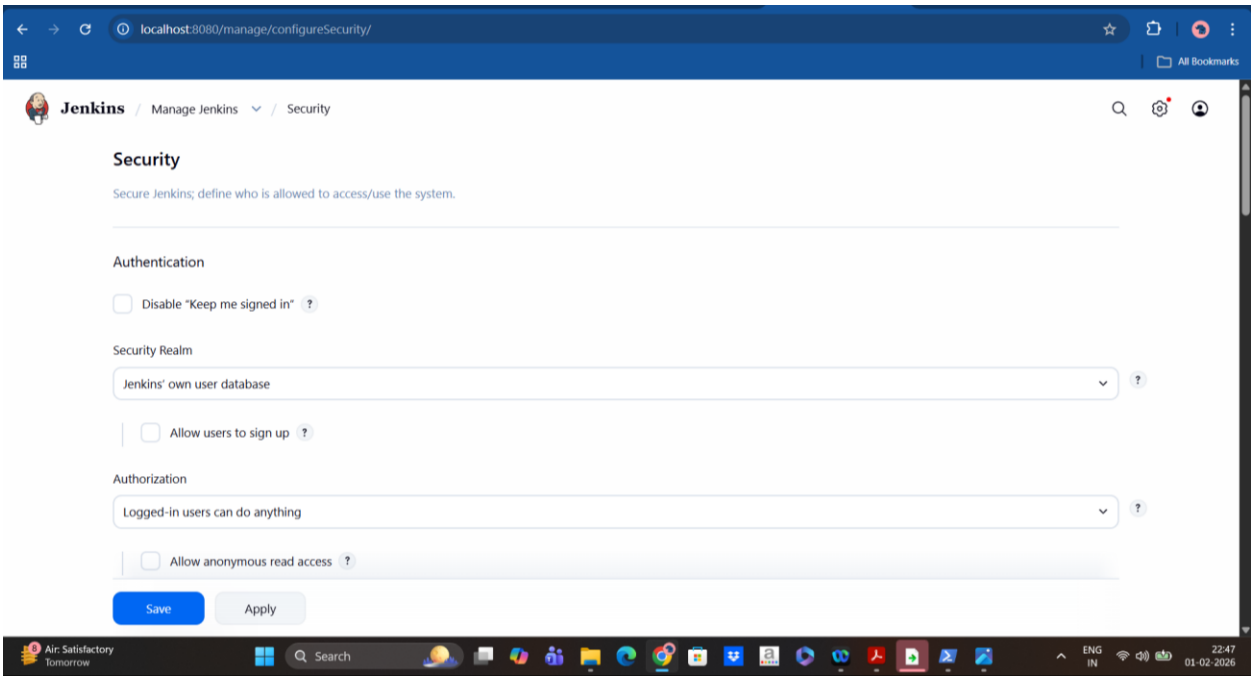

localhost:8080/user/admin1/
All Bookmarks
Jenkins / Geo
Geo
Add description
Profile
Builds
My Views
Account
Appearance
Preferences
Security
Experiments
Credentials
Jenkins User ID: admin1
REST API
Jenkins 2.541.1
27°C
Mostly clear
Search
ENG
IN
22:54
01-02-2026
localhost:8080
All Bookmarks
Jenkins
Build History
Project Relationship
Check File Fingerprint
Build Queue
No builds in the queue.
Build Executor Status
0/2
All
S
W
Name
Last Success
Last Failure
Last Duration
Git-Test
14 min #3
N/A
1.3 sec
Maven-Build-Job
39 min #5
11 min #8
1.8 sec
Maven-Verification
N/A
N/A
N/A
Pipeline-Demo
10 min #1
N/A
9.7 sec
Icon: S M L
REST API
Jenkins 2.541.1
27°C
Mostly clear
Search
ENG
IN
22:54
01-02-2026

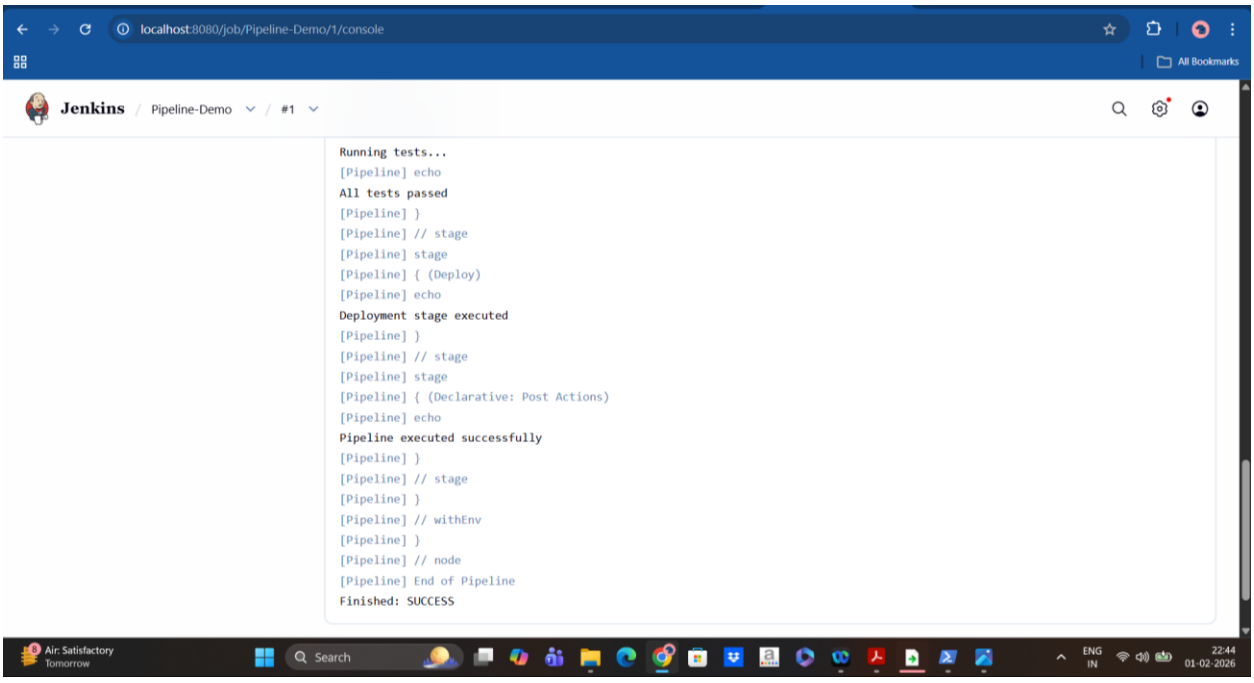

localhost:8080/manage/configureSecurity/
Jenkins
Manage Jenkins
Security
Overall
Credentials
Agent
Job
Run
View
SCM
Metrics
Administer
Read
Create
Delete
ManageDomains
Update
View
Build
Configure
Connect
Create
Delete
Disconnect
Provision
Build
Cancel
Configure
Create
Delete
Discover
Move
Read
Workspace
Delete
Replay
Update
Configure
Create
Delete
Read
Tag
HealthCheck
ThreadDump
View
User/group
Anonymous
Authenticated Users
Geo Mathew George
Geo
Add user...
Add group...
Markup Formatter
Save
Apply
27°C
Mostly clear
Search
ENG
IN
22:54
01-02-2026

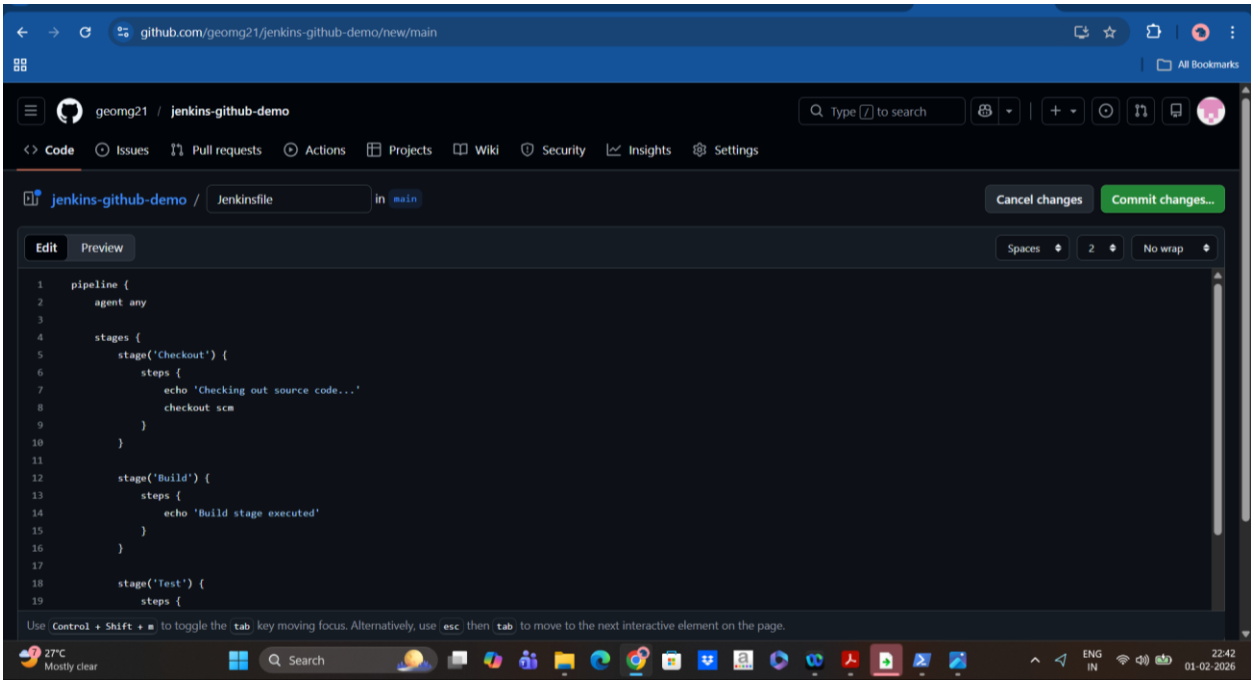

localhost:8080/manage/configureSecurity/
Jenkins
Manage Jenkins
Security
Overall
Credentials
Agent
Job
Run
View
SCM
Metrics
Administer
Read
Create
Delete
ManageDomains
Update
View
Build
Configure
Connect
Create
Delete
Disconnect
Provision
Build
Cancel
Configure
Create
Delete
Discover
Move
Read
Workspace
Delete
Replay
Update
Configure
Create
Delete
Read
Tag
HealthCheck
ThreadDump
View
User/group
Anonymous
Authenticated Users
Geo Mathew George
Geo
Add user...
Add group...
Markup Formatter
Save
Apply
27°C
Mostly clear
Search
ENG
IN
22:52
01-02-2026

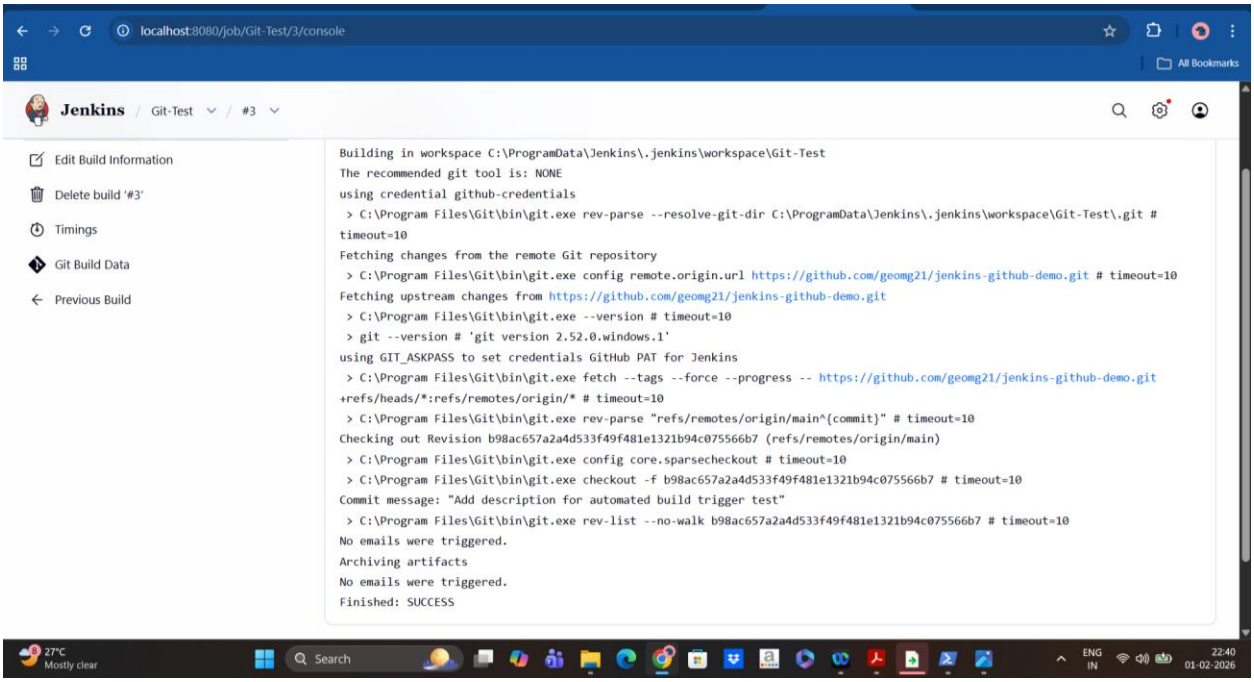

localhost:8080/securityRealm/
All Bookmarks
Jenkins / Jenkins' own user database
Users 2
+ Create User
These users can log into Jenkins. This is a sub set of this list, which also contains auto-created users who really just made some commits on some projects and have no direct Jenkins access.
User ID ↓
Name
admin
Geo Mathew George
admin1
Geo
Jenkins 2.541.1

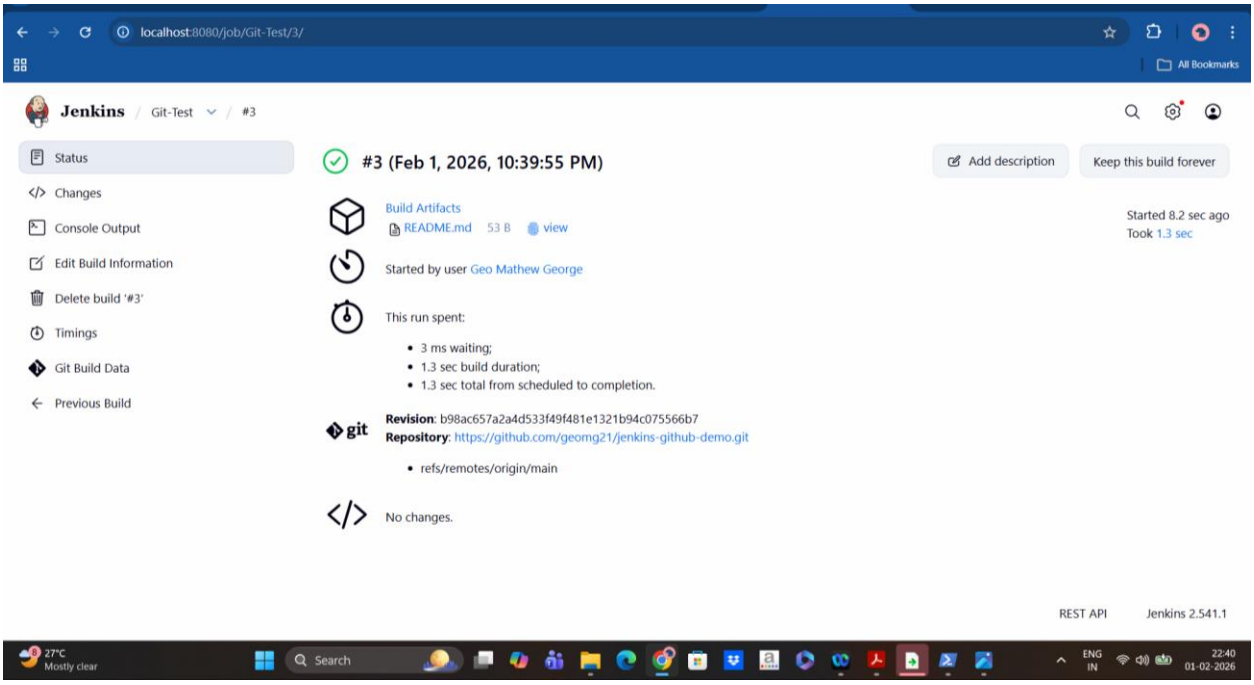

localhost:8080/manage/configureSecurity/
All Bookmarks
Jenkins / Manage Jenkins / Security
Overall
Credentials
Agent
Job
Run
View
SCM
Metrics
Administer
Read
Create
Delete
ManageDomains
Update
View
Build
Configure
Connect
Create
Delete
Disconnect
Provision
Build
Cancel
Configure
Create
Delete
Discover
Move
Read
Workspace
Delete
Replay
Update
Configure
Create
Delete
Read
Tag
HealthCheck
ThreadDump
View
User/group
Anonymous
Authenticated Users
Geo Mathew George
Add user...
Add group...
Markup Formatter
Markup Formatter
Save
Apply

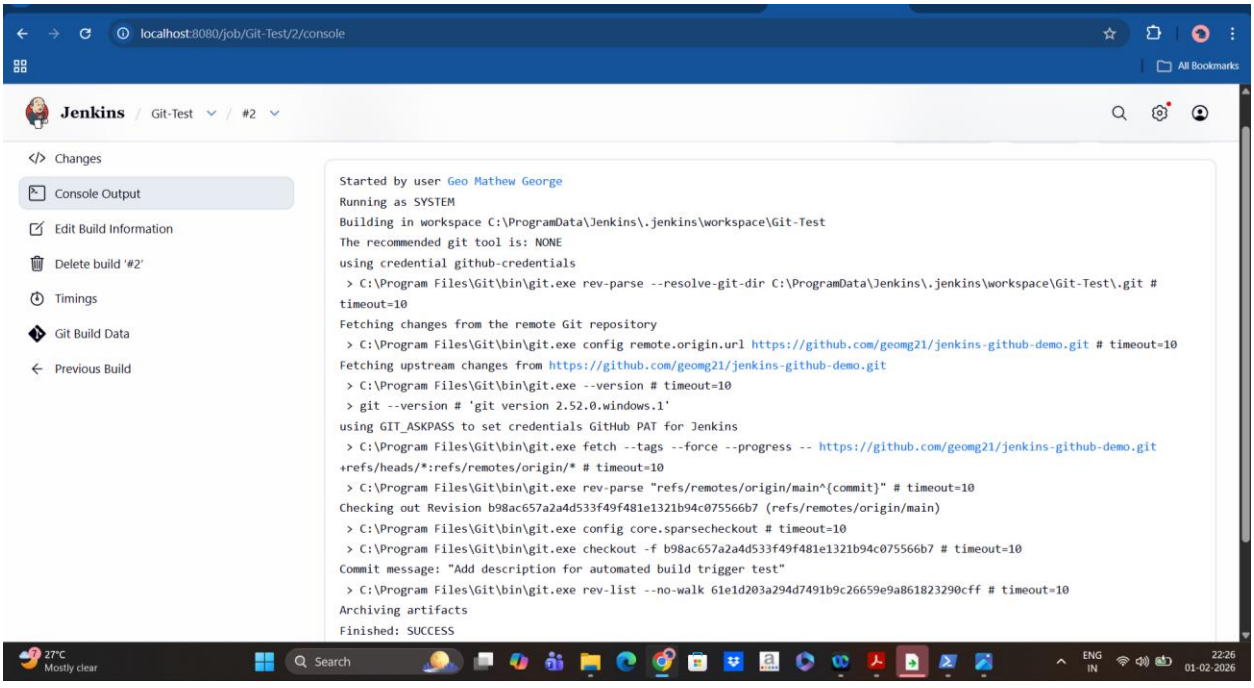

localhost:8080/manage/configureSecurity/
All Bookmarks
Jenkins / Manage Jenkins / Security
Security
Secure Jenkins; define who is allowed to access/use the system.
Authentication
Disable "Keep me signed in"
Security Realm
Jenkins' own user database
Allow users to sign up
Authorization
Logged-in users can do anything
Allow anonymous read access
Save
Apply
Air: Satisfactory
Tomorrow
Search
ENG
IN
22:47
01-02-2026

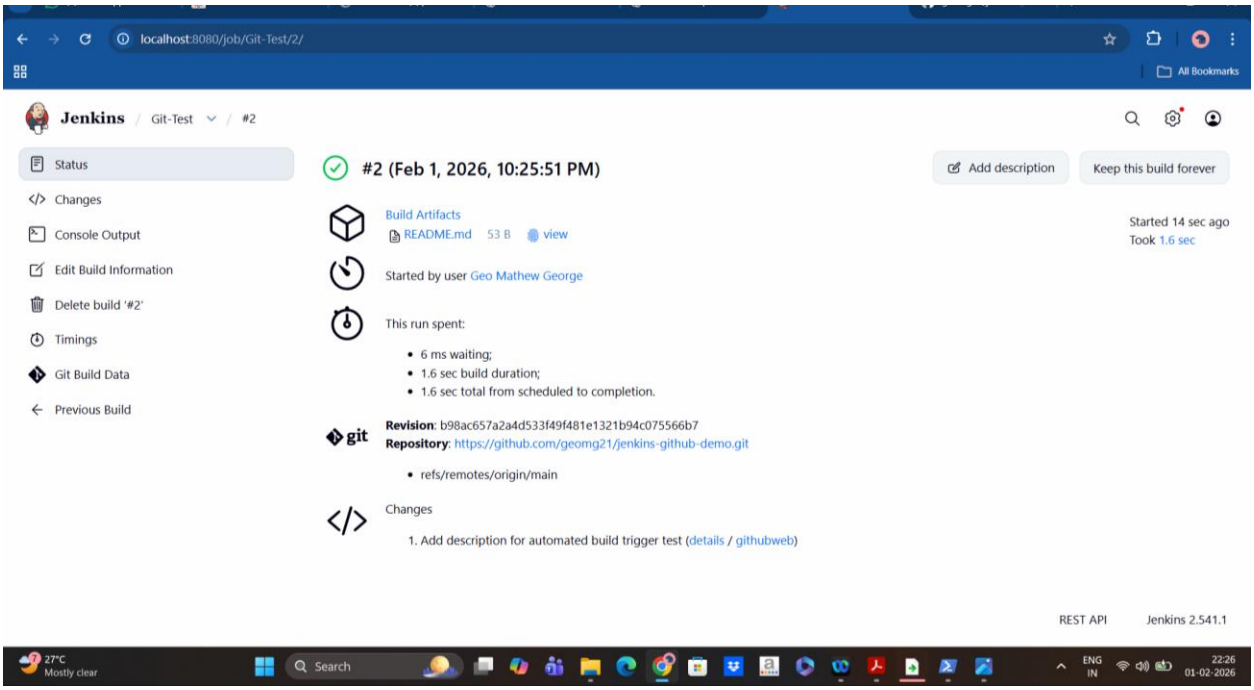

localhost:8080/job/Pipeline-Demo/1/console
All Bookmarks
Jenkins / Pipeline-Demo / #1
Running tests...
[Pipeline] echo
All tests passed
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Deploy)
[Pipeline] echo
Deployment stage executed
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Declarative: Post Actions)
[Pipeline] echo
Pipeline executed successfully
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
Air: Satisfactory
Tomorrow
Search
ENG
IN
22:44
01-02-2026

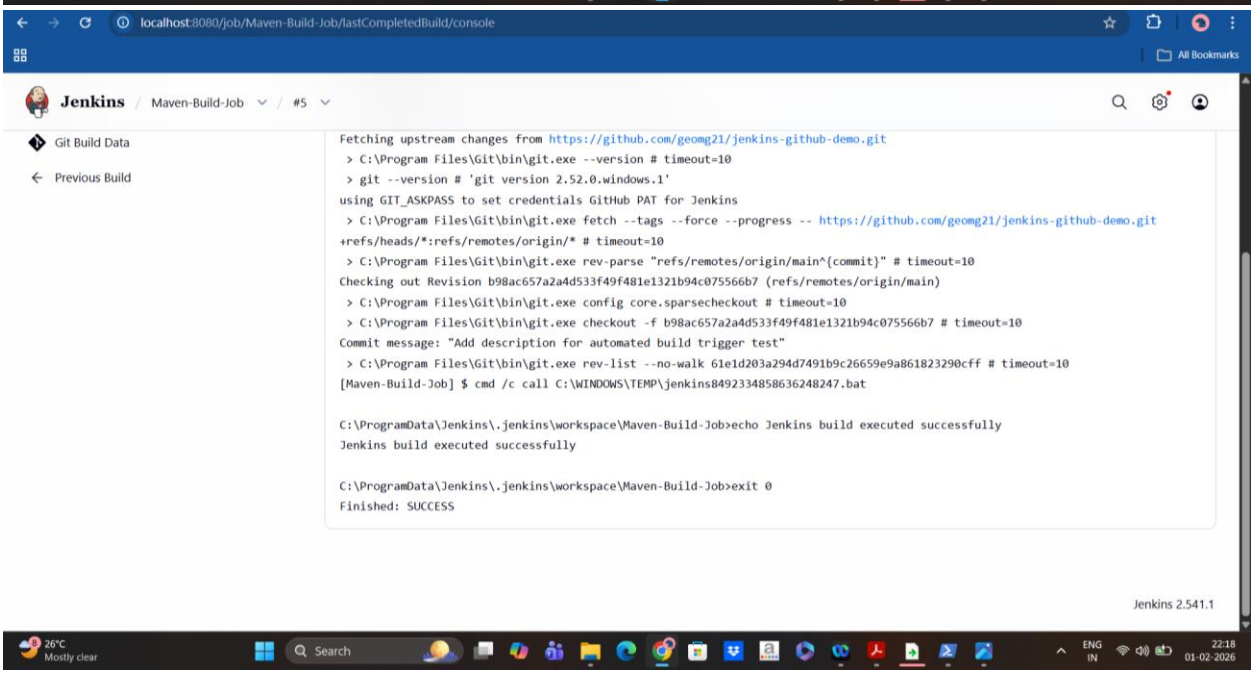

github.com/geomg21/jenkins-github-demo/new/main
geomg21 / jenkins-github-demo
Code Issues Pull requests Actions Projects Wiki Security Insights Settings
jenkins-github-demo / Jenkinsfile in main
Cancel changes
Commit changes...
Edit Preview
Spaces 2 No wrap
pipeline {
agent any
stages {
stage('Checkout') {
steps {
echo 'Checking out source code...'
checkout scm
}
}
stage('Build') {
steps {
echo 'Build stage executed'
}
}
stage('Test') {
steps {
Use Control + Shift + m to toggle the tab key moving focus. Alternatively, use esc then tab to move to the next interactive element on the page.
27°C
Mostly clear
Search
ENG
IN
22:42
01-02-2026

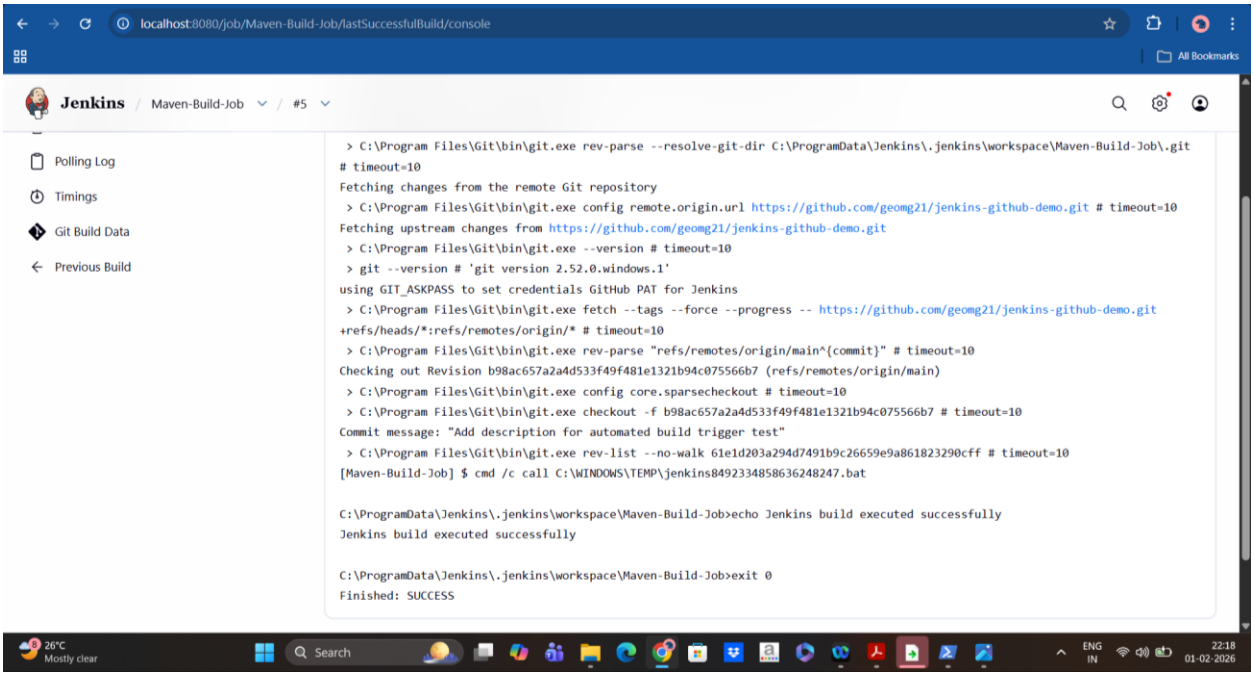

localhost:8080/job/Git-Test/3/console
Jenkins / Git-Test / #3
Edit Build Information
Delete build '#3'
Timings
Git Build Data
Previous Build
Building in workspace C:\ProgramData\Jenkins\.jenkins\workspace\Git-Test
The recommended git tool is: NONE
using credential github-credentials
> C:\Program Files\Git\bin\git.exe rev-parse --resolve-git-dir C:\ProgramData\Jenkins\.jenkins\workspace\Git-Test\.git # timeout=10
Fetching changes from the remote Git repository
> C:\Program Files\Git\bin\git.exe config remote.origin.url https://github.com/geomg21/jenkins-github-demo.git # timeout=10
Fetching upstream changes from https://github.com/geomg21/jenkins-github-demo.git
> C:\Program Files\Git\bin\git.exe --version # timeout=10
> git --version # 'git version 2.52.0.windows.1'
using GIT_ASKPASS to set credentials GitHub PAT for Jenkins
> C:\Program Files\Git\bin\git.exe fetch --tags --force --progress -- https://github.com/geomg21/jenkins-github-demo.git +refs/heads/*:refs/remotes/origin/* # timeout=10
> C:\Program Files\Git\bin\git.exe rev-parse "refs/remotes/origin/main^{commit}" # timeout=10
Checking out Revision b98ac657a2a4d533f49f481e1321b94c075566b7 (refs/remotes/origin/main)
> C:\Program Files\Git\bin\git.exe config core.sparsecheckout # timeout=10
> C:\Program Files\Git\bin\git.exe checkout -f b98ac657a2a4d533f49f481e1321b94c075566b7 # timeout=10
Commit message: "Add description for automated build trigger test"
> C:\Program Files\Git\bin\git.exe rev-list --no-walk b98ac657a2a4d533f49f481e1321b94c075566b7 # timeout=10
No emails were triggered.
Archiving artifacts
No emails were triggered.
Finished: SUCCESS
27°C
Mostly clear
Search
ENG
IN
22:40
01-02-2026

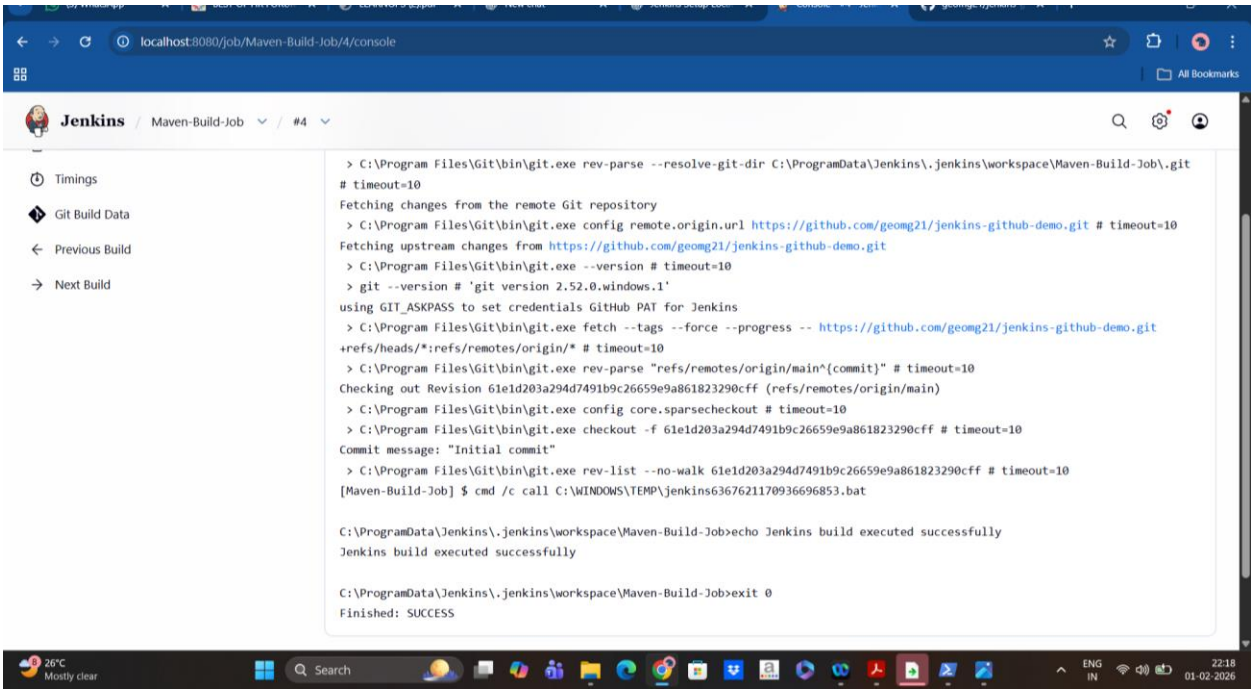

localhost:8080/job/Git-Test/3/
All Bookmarks
Jenkins / Git-Test / #3
Status
Changes
Console Output
Edit Build Information
Delete build '#3'
Timings
Git Build Data
Previous Build
#3 (Feb 1, 2026, 10:39:55 PM)
Add description
Keep this build forever
Build Artifacts
README.md 53 B view
Started 8.2 sec ago
Took 1.3 sec
Started by user Geo Mathew George
This run spent:
3 ms waiting;
1.3 sec build duration;
1.3 sec total from scheduled to completion.
Revision: b98ac657a2a4d533f49f481e1321b94c075566b7
Repository: https://github.com/geomg21/jenkins-github-demo.git
refs/remotes/origin/main
No changes.
REST API
Jenkins 2.541.1
27°C
Mostly clear
Search
ENG
IN
22:40
01-02-2026

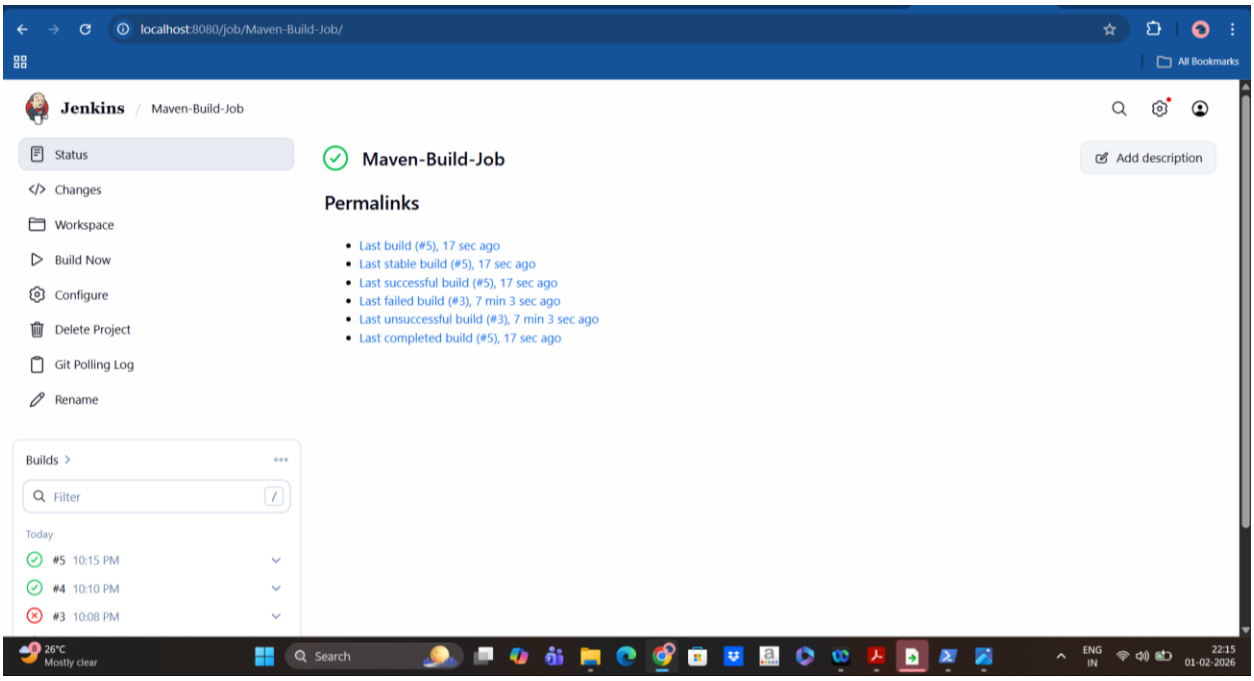

localhost:8080/job/Git-Test/2/console
All Bookmarks
Jenkins / Git-Test / #2
Changes
Console Output
Edit Build Information
Delete build '#2'
Timings
Git Build Data
Previous Build
Started by user Geo Mathew George
Running as SYSTEM
Building in workspace C:\ProgramData\Jenkins\.jenkins\workspace\Git-Test
The recommended git tool is: NONE
using credential github-credentials
> C:\Program Files\Git\bin\git.exe rev-parse --resolve-git-dir C:\ProgramData\Jenkins\.jenkins\workspace\Git-Test\.git # timeout=10
Fetching changes from the remote Git repository
> C:\Program Files\Git\bin\git.exe config remote.origin.url https://github.com/geomg21/jenkins-github-demo.git # timeout=10
Fetching upstream changes from https://github.com/geomg21/jenkins-github-demo.git
> C:\Program Files\Git\bin\git.exe --version # timeout=10
> git --version # 'git version 2.52.0.windows.1'
using GIT_ASKPASS to set credentials GitHub PAT for Jenkins
> C:\Program Files\Git\bin\git.exe fetch --tags --force --progress -- https://github.com/geomg21/jenkins-github-demo.git +refs/heads/*:refs/remotes/origin/* # timeout=10
> C:\Program Files\Git\bin\git.exe rev-parse "refs/remotes/origin/main^{commit}" # timeout=10
Checking out Revision b98ac657a2a4d533f49f481e1321b94c075566b7 (refs/remotes/origin/main)
> C:\Program Files\Git\bin\git.exe config core.sparsecheckout # timeout=10
> C:\Program Files\Git\bin\git.exe checkout -f b98ac657a2a4d533f49f481e1321b94c075566b7 # timeout=10
Commit message: "Add description for automated build trigger test"
> C:\Program Files\Git\bin\git.exe rev-list --no-walk 61e1d203a294d7491b9c26659e9a861823290cff # timeout=10
Archiving artifacts
Finished: SUCCESS
27°C
Mostly clear
Search
ENG
IN
22:26
01-02-2026

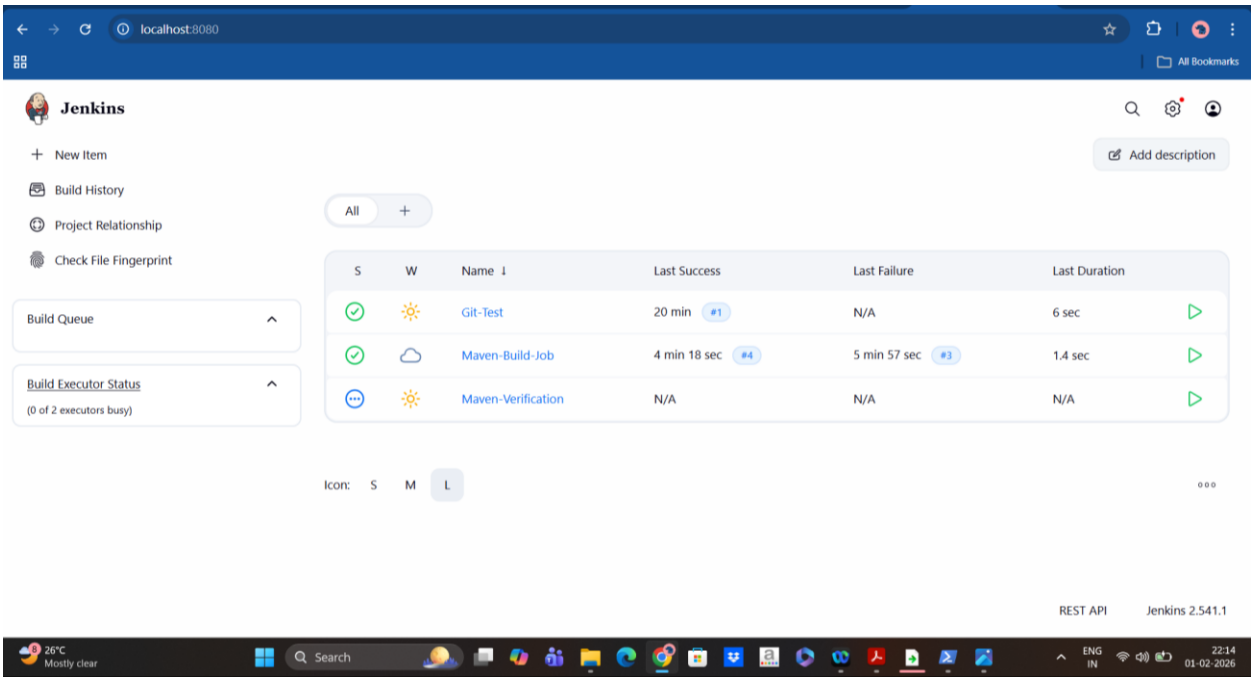

localhost:8080/job/Git-Test/2/
Jenkins / Git-Test / #2
Status
Changes
Console Output
Edit Build Information
Delete build '#2'
Timings
Git Build Data
Previous Build
#2 (Feb 1, 2026, 10:25:51 PM)
Add description
Keep this build forever
Build Artifacts
README.md 53 B view
Started 14 sec ago
Took 1.6 sec
Started by user Geo Mathew George
This run spent:
6 ms waiting;
1.6 sec build duration;
1.6 sec total from scheduled to completion.
Revision: b98ac657a2a4d533f49f481e1321b94c075566b7
Repository: https://github.com/geomg21/jenkins-github-demo.git
refs/remotes/origin/main
Changes
1. Add description for automated build trigger test (details / githubweb)
REST API
Jenkins 2.541.1

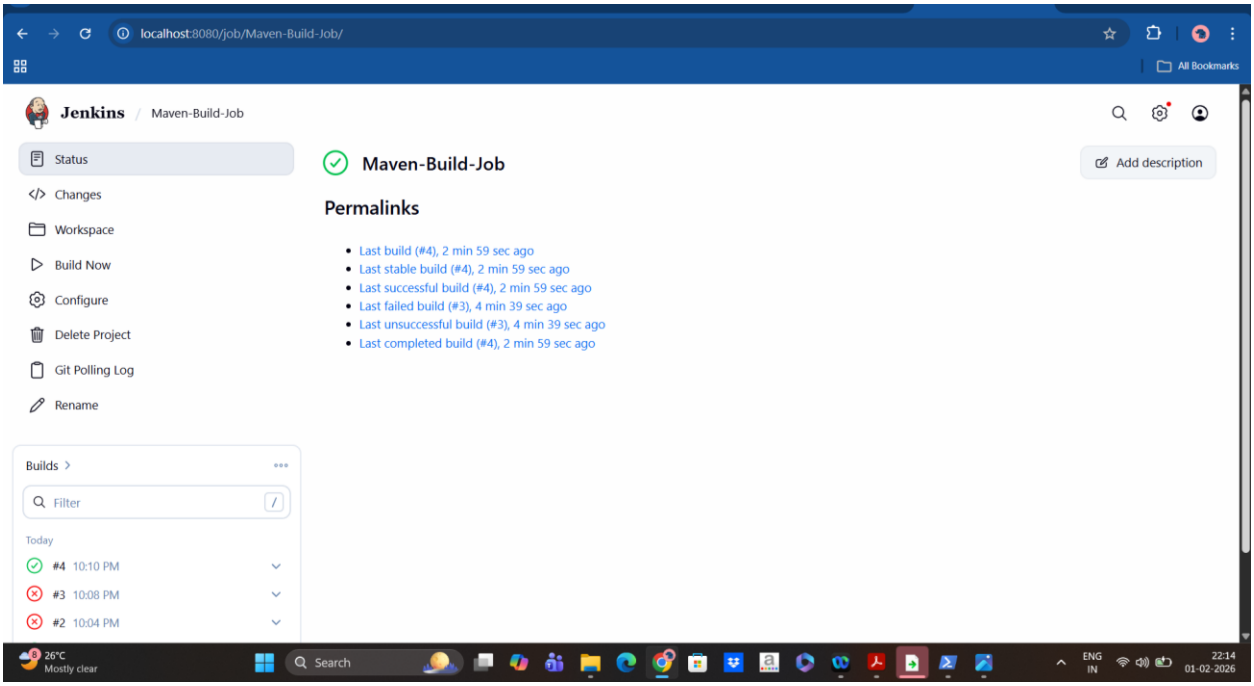

localhost:8080/job/Maven-Build-Job/lastCompletedBuild/console
Jenkins / Maven-Build-Job / #5
Git Build Data
Previous Build
Fetching upstream changes from https://github.com/geomg21/jenkins-github-demo.git
> C:\Program Files\Git\bin\git.exe --version # timeout=10
> git --version # 'git version 2.52.0.windows.1'
using GIT_ASKPASS to set credentials GitHub PAT for Jenkins
> C:\Program Files\Git\bin\git.exe fetch --tags --force --progress -- https://github.com/geomg21/jenkins-github-demo.git
+refs/heads/*:refs/remotes/origin/* # timeout=10
> C:\Program Files\Git\bin\git.exe rev-parse "refs/remotes/origin/main^{commit}" # timeout=10
Checking out Revision b98ac657a2a4d533f49f481e1321b94c075566b7 (refs/remotes/origin/main)
> C:\Program Files\Git\bin\git.exe config core.sparsecheckout # timeout=10
> C:\Program Files\Git\bin\git.exe checkout -f b98ac657a2a4d533f49f481e1321b94c075566b7 # timeout=10
Commit message: "Add description for automated build trigger test"
> C:\Program Files\Git\bin\git.exe rev-list --no-walk 61e1d203a294d7491b9c26659e9a861823290cff # timeout=10
[Maven-Build-Job] $ cmd /c call C:\WINDOWS\TEMP\jenkins8492334858636248247.bat
C:\ProgramData\Jenkins\.jenkins\workspace\Maven-Build-Job>echo Jenkins build executed successfully
Jenkins build executed successfully
C:\ProgramData\Jenkins\.jenkins\workspace\Maven-Build-Job>exit 0
Finished: SUCCESS
Jenkins 2.541.1

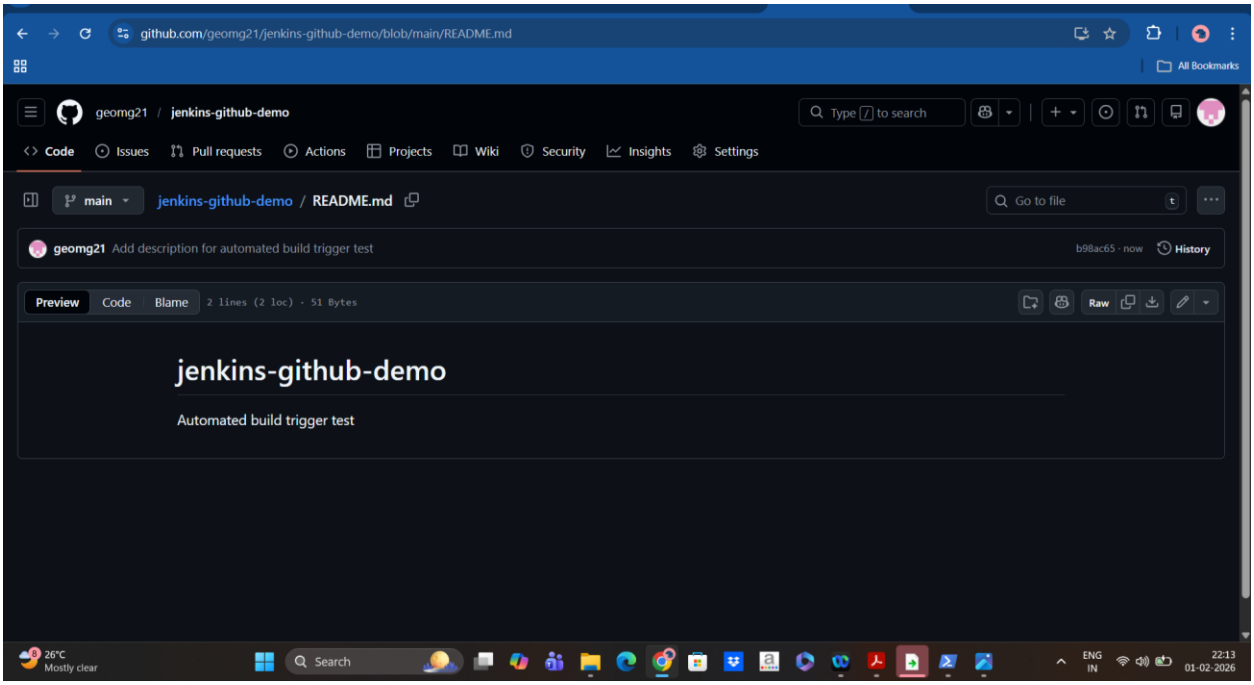

localhost:8080/job/Maven-Build-Job/lastSuccessfulBuild/console
All Bookmarks
Jenkins
Maven-Build-Job
#5
Polling Log
Timings
Git Build Data
Previous Build
> C:\Program Files\Git\bin\git.exe rev-parse --resolve-git-dir C:\ProgramData\Jenkins\.jenkins\workspace\Maven-Build-Job\.git # timeout=10
Fetching changes from the remote Git repository
> C:\Program Files\Git\bin\git.exe config remote.origin.url https://github.com/geomg21/jenkins-github-demo.git # timeout=10
Fetching upstream changes from https://github.com/geomg21/jenkins-github-demo.git
> C:\Program Files\Git\bin\git.exe --version # timeout=10
> git --version # 'git version 2.52.0.windows.1'
using GIT_ASKPASS to set credentials GitHub PAT for Jenkins
> C:\Program Files\Git\bin\git.exe fetch --tags --force --progress -- https://github.com/geomg21/jenkins-github-demo.git +refs/heads/*:refs/remotes/origin/* # timeout=10
> C:\Program Files\Git\bin\git.exe rev-parse "refs/remotes/origin/main^{commit}" # timeout=10
Checking out Revision b98ac657a2a4d533f49f481e1321b94c075566b7 (refs/remotes/origin/main)
> C:\Program Files\Git\bin\git.exe config core.sparsecheckout # timeout=10
> C:\Program Files\Git\bin\git.exe checkout -f b98ac657a2a4d533f49f481e1321b94c075566b7 # timeout=10
Commit message: "Add description for automated build trigger test"
> C:\Program Files\Git\bin\git.exe rev-list --no-walk 61e1d203a294d7491b9c26659e9a861823290cff # timeout=10
[Maven-Build-Job] $ cmd /c call C:\WINDOWS\TEMP\jenkins8492334858636248247.bat
C:\ProgramData\Jenkins\.jenkins\workspace\Maven-Build-Job>echo Jenkins build executed successfully
Jenkins build executed successfully
C:\ProgramData\Jenkins\.jenkins\workspace\Maven-Build-Job>exit 0
Finished: SUCCESS
26°C
Mostly clear
Search
ENG
IN
22:18
01-02-2026

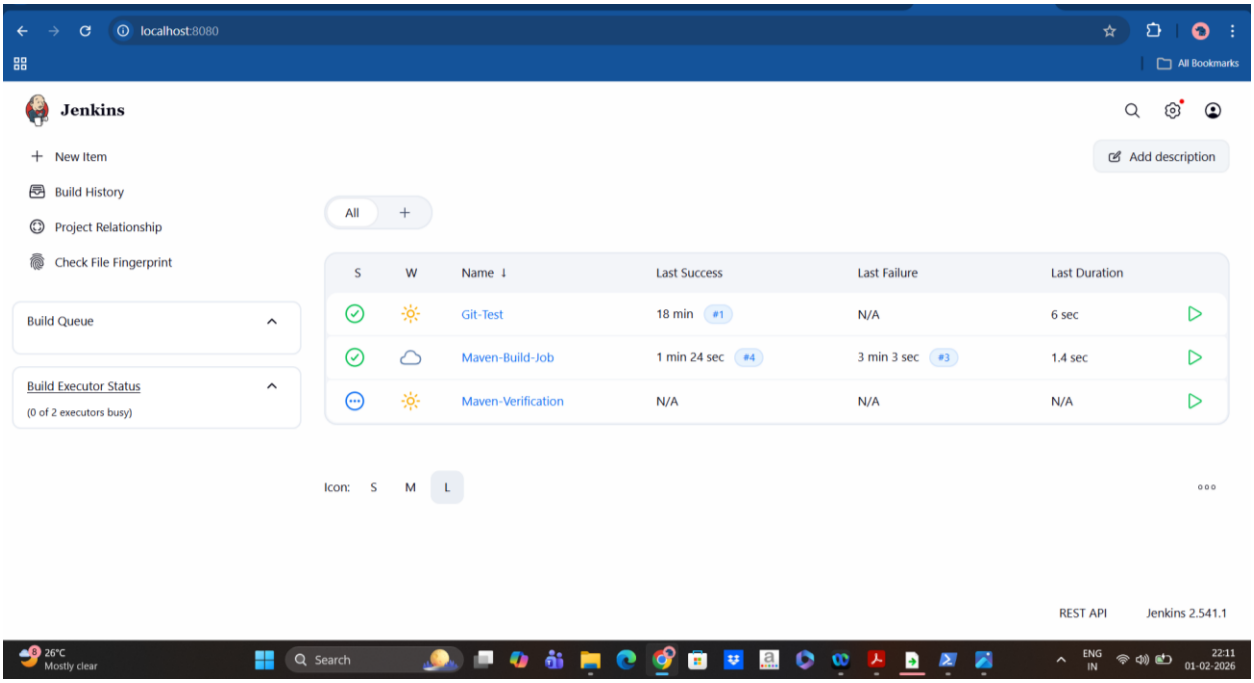

localhost:8080/job/Maven-Build-Job/4/console
All Bookmarks
Jenkins
Maven-Build-Job
#4
Timings
Git Build Data
Previous Build
Next Build
> C:\Program Files\Git\bin\git.exe rev-parse --resolve-git-dir C:\ProgramData\Jenkins\.jenkins\workspace\Maven-Build-Job\.git # timeout=10
Fetching changes from the remote Git repository
> C:\Program Files\Git\bin\git.exe config remote.origin.url https://github.com/geomg21/jenkins-github-demo.git # timeout=10
Fetching upstream changes from https://github.com/geomg21/jenkins-github-demo.git
> C:\Program Files\Git\bin\git.exe --version # timeout=10
> git --version # 'git version 2.52.0.windows.1'
using GIT_ASKPASS to set credentials GitHub PAT for Jenkins
> C:\Program Files\Git\bin\git.exe fetch --tags --force --progress -- https://github.com/geomg21/jenkins-github-demo.git +refs/heads/*:refs/remotes/origin/* # timeout=10
> C:\Program Files\Git\bin\git.exe rev-parse "refs/remotes/origin/main^{commit}" # timeout=10
Checking out Revision 61e1d203a294d7491b9c26659e9a861823290cff (refs/remotes/origin/main)
> C:\Program Files\Git\bin\git.exe config core.sparsecheckout # timeout=10
> C:\Program Files\Git\bin\git.exe checkout -f 61e1d203a294d7491b9c26659e9a861823290cff # timeout=10
Commit message: "Initial commit"
> C:\Program Files\Git\bin\git.exe rev-list --no-walk 61e1d203a294d7491b9c26659e9a861823290cff # timeout=10
[Maven-Build-Job] $ cmd /c call C:\WINDOWS\TEMP\jenkins6367621170936696853.bat
C:\ProgramData\Jenkins\.jenkins\workspace\Maven-Build-Job>echo Jenkins build executed successfully
Jenkins build executed successfully
C:\ProgramData\Jenkins\.jenkins\workspace\Maven-Build-Job>exit 0
Finished: SUCCESS
26°C
Mostly clear
Search
ENG
IN
22:18
01-02-2026

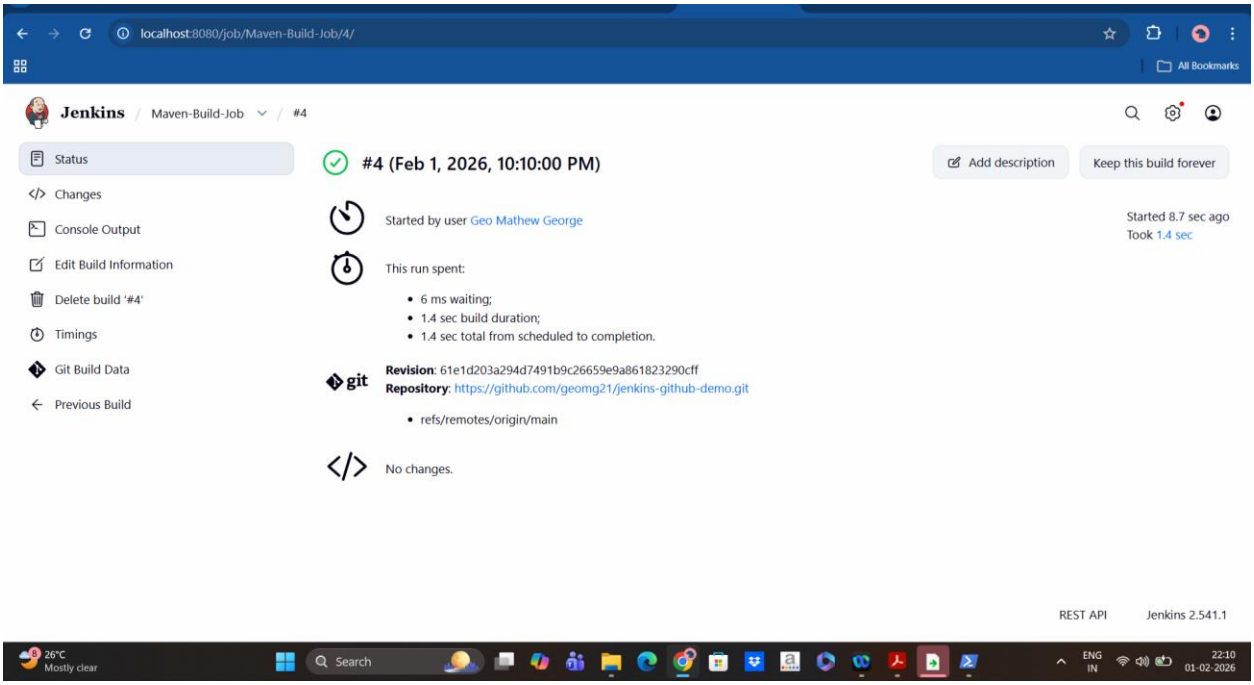

localhost:8080/job/Maven-Build-Job/
All Bookmarks
Jenkins / Maven-Build-Job
Status
Changes
Workspace
Build Now
Configure
Delete Project
Git Polling Log
Rename
Maven-Build-Job
Add description
Permalinks
Last build (#5), 17 sec ago
Last stable build (#5), 17 sec ago
Last successful build (#5), 17 sec ago
Last failed build (#3), 7 min 3 sec ago
Last unsuccessful build (#3), 7 min 3 sec ago
Last completed build (#5), 17 sec ago
Builds
Filter
Today
#5 10:15 PM
#4 10:10 PM
#3 10:08 PM
26°C
Mostly clear
Search
ENG
IN
22:15
01-02-2026

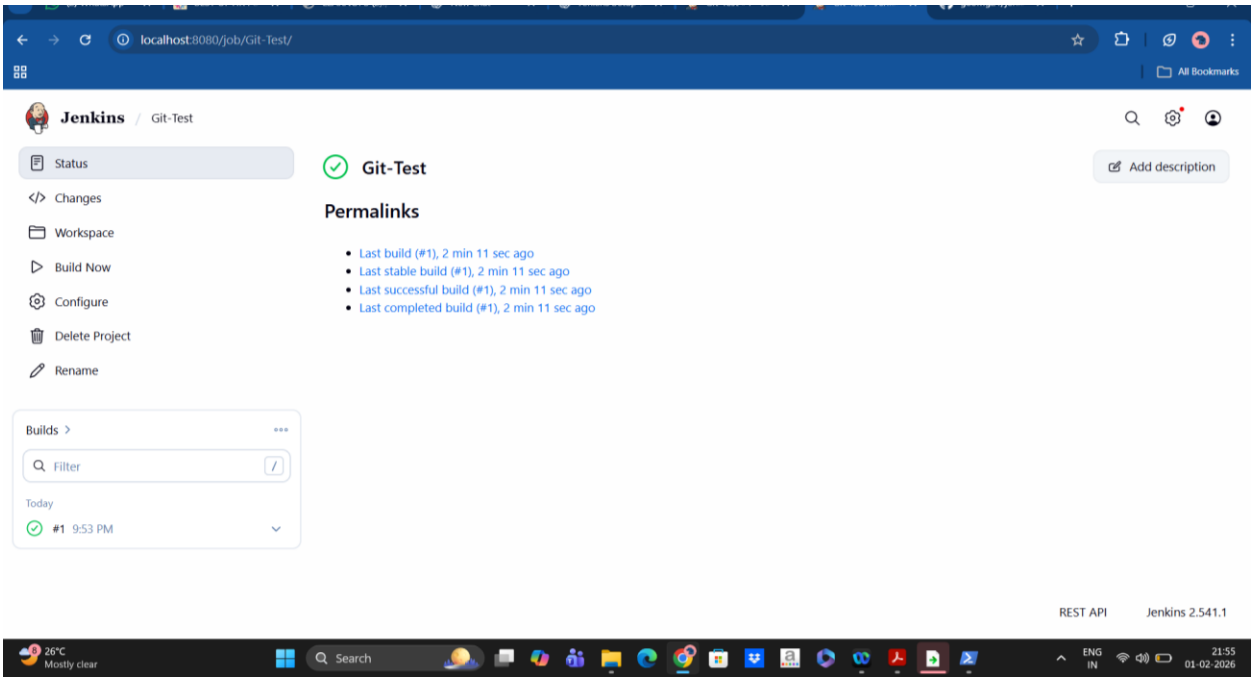

localhost:8080
All Bookmarks
Jenkins
New Item
Build History
Project Relationship
Check File Fingerprint
Add description
Build Queue
Build Executor Status
(0 of 2 executors busy)
All
S
W
Name
Last Success
Last Failure
Last Duration
Git-Test
20 min #1
N/A
6 sec
Maven-Build-Job
4 min 18 sec #4
5 min 57 sec #3
1.4 sec
Maven-Verification
N/A
N/A
N/A
Icon: S M L
REST API
Jenkins 2.541.1
26°C
Mostly clear
Search
ENG
IN
22:14
01-02-2026

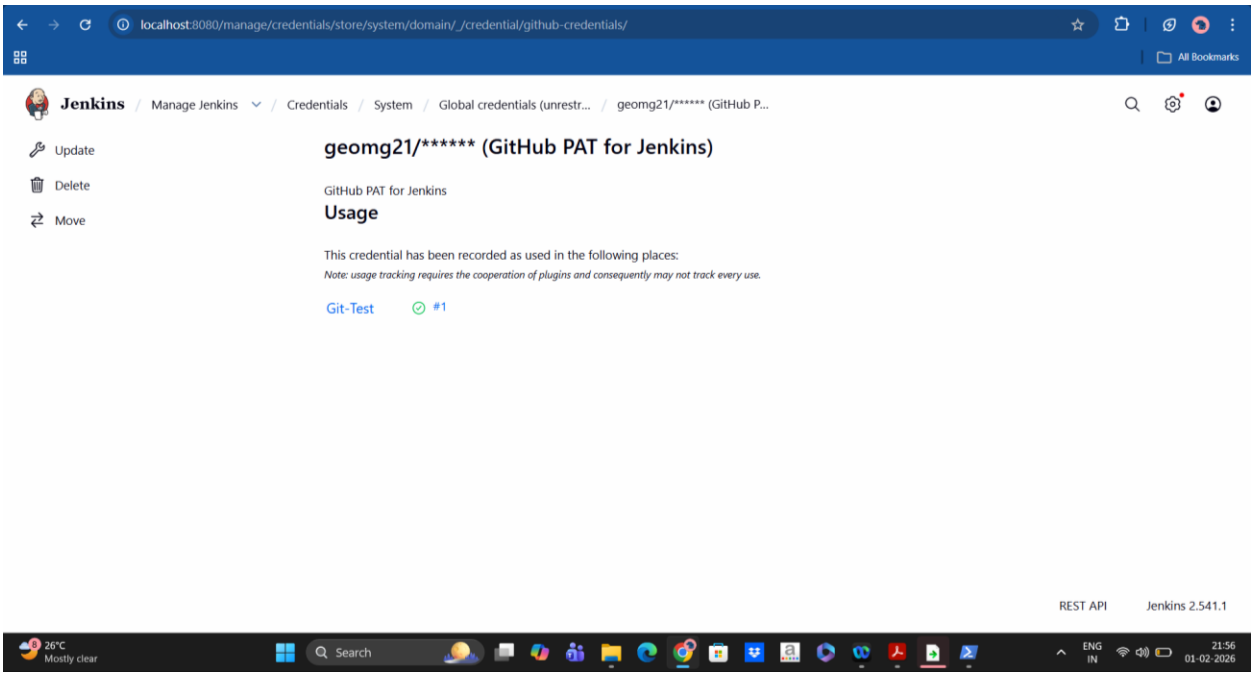

localhost:8080/job/Maven-Build-Job/
Jenkins
Maven-Build-Job
Status
Changes
Workspace
Build Now
Configure
Delete Project
Git Polling Log
Rename
Maven-Build-Job
Add description
Permalinks
Last build (#4), 2 min 59 sec ago
Last stable build (#4), 2 min 59 sec ago
Last successful build (#4), 2 min 59 sec ago
Last failed build (#3), 4 min 39 sec ago
Last unsuccessful build (#3), 4 min 39 sec ago
Last completed build (#4), 2 min 59 sec ago
Builds
Filter
Today
#4 10:10 PM
#3 10:08 PM
#2 10:04 PM
Search
22:14
01-02-2026

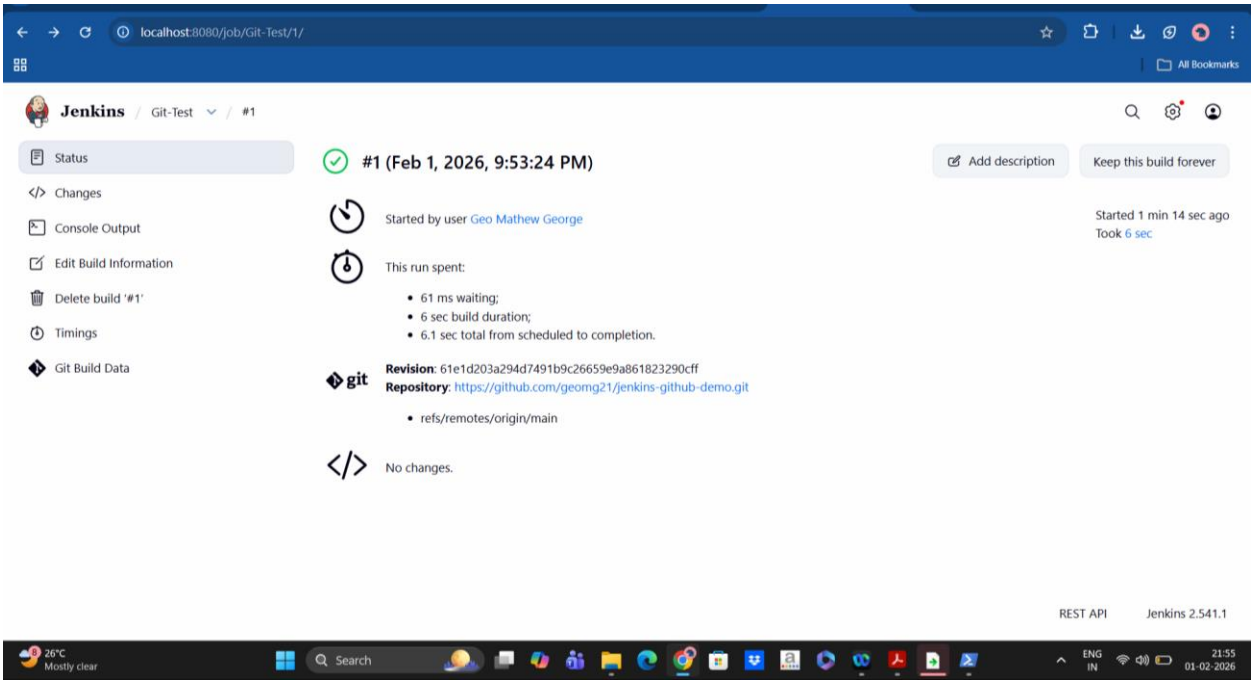

github.com/geomg21/jenkins-github-demo/blob/main/README.md
geomg21 / jenkins-github-demo
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
main
jenkins-github-demo / README.md
Go to file
geomg21 Add description for automated build trigger test
b98ac65 · now
History
Preview
Code
Blame
2 lines (2 loc) · 51 Bytes
Raw
jenkins-github-demo
Automated build trigger test
Search
22:13
01-02-2026


localhost:8080
All Bookmarks
Jenkins
New Item
Build History
Project Relationship
Check File Fingerprint
Add description
All
S
W
Name ↓
Last Success
Last Failure
Last Duration
Git-Test
18 min #1
N/A
6 sec
Maven-Build-Job
1 min 24 sec #4
3 min 3 sec #3
1.4 sec
Maven-Verification
N/A
N/A
N/A
Build Queue
Build Executor Status
(0 of 2 executors busy)
Icon: S M L
REST API
Jenkins 2.541.1
26°C
Mostly clear
Search
ENG
IN
22:11
01-02-2026

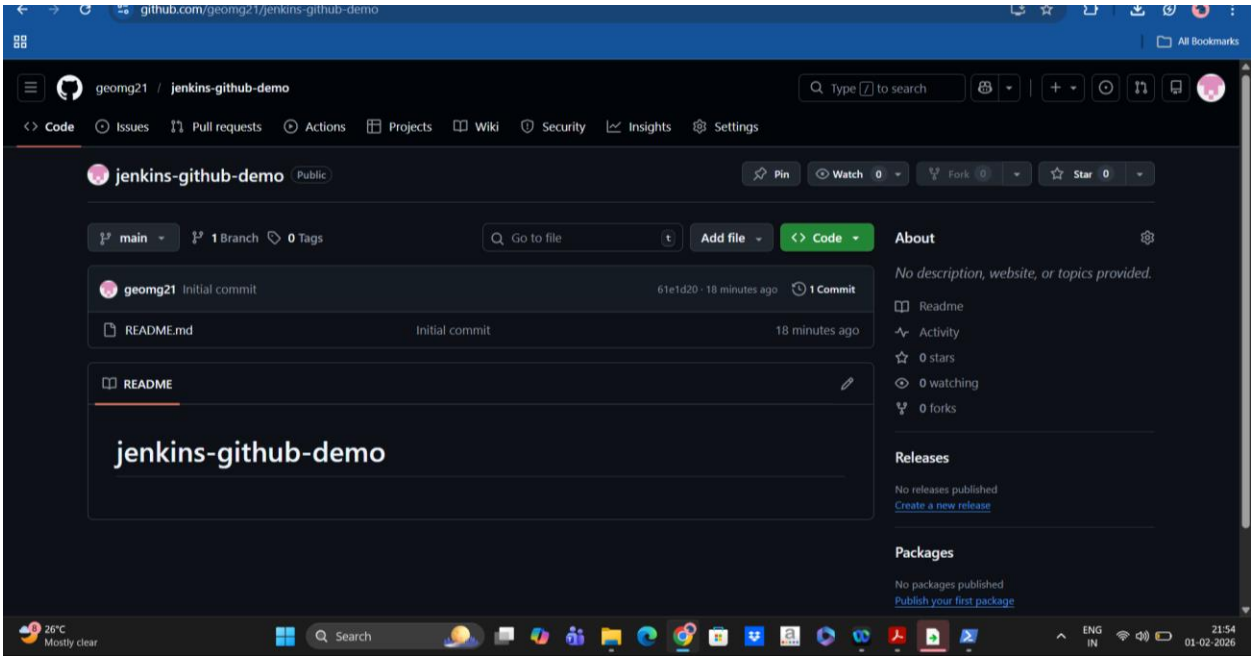

localhost:8080/job/Maven-Build-Job/4/
All Bookmarks
Jenkins
Maven-Build-Job
#4
Status
Changes
Console Output
Edit Build Information
Delete build '#4'
Timings
Git Build Data
Previous Build
#4 (Feb 1, 2026, 10:10:00 PM)
Add description
Keep this build forever
Started by user Geo Mathew George
Started 8.7 sec ago
Took 1.4 sec
This run spent:
6 ms waiting;
1.4 sec build duration;
1.4 sec total from scheduled to completion.
Revision: 61e1d203a294d7491b9c26659e9a861823290cff
Repository: https://github.com/geomg21/jenkins-github-demo.git
refs/remotes/origin/main
No changes.
REST API
Jenkins 2.541.1
26°C
Mostly clear
Search
ENG
IN
22:10
01-02-2026

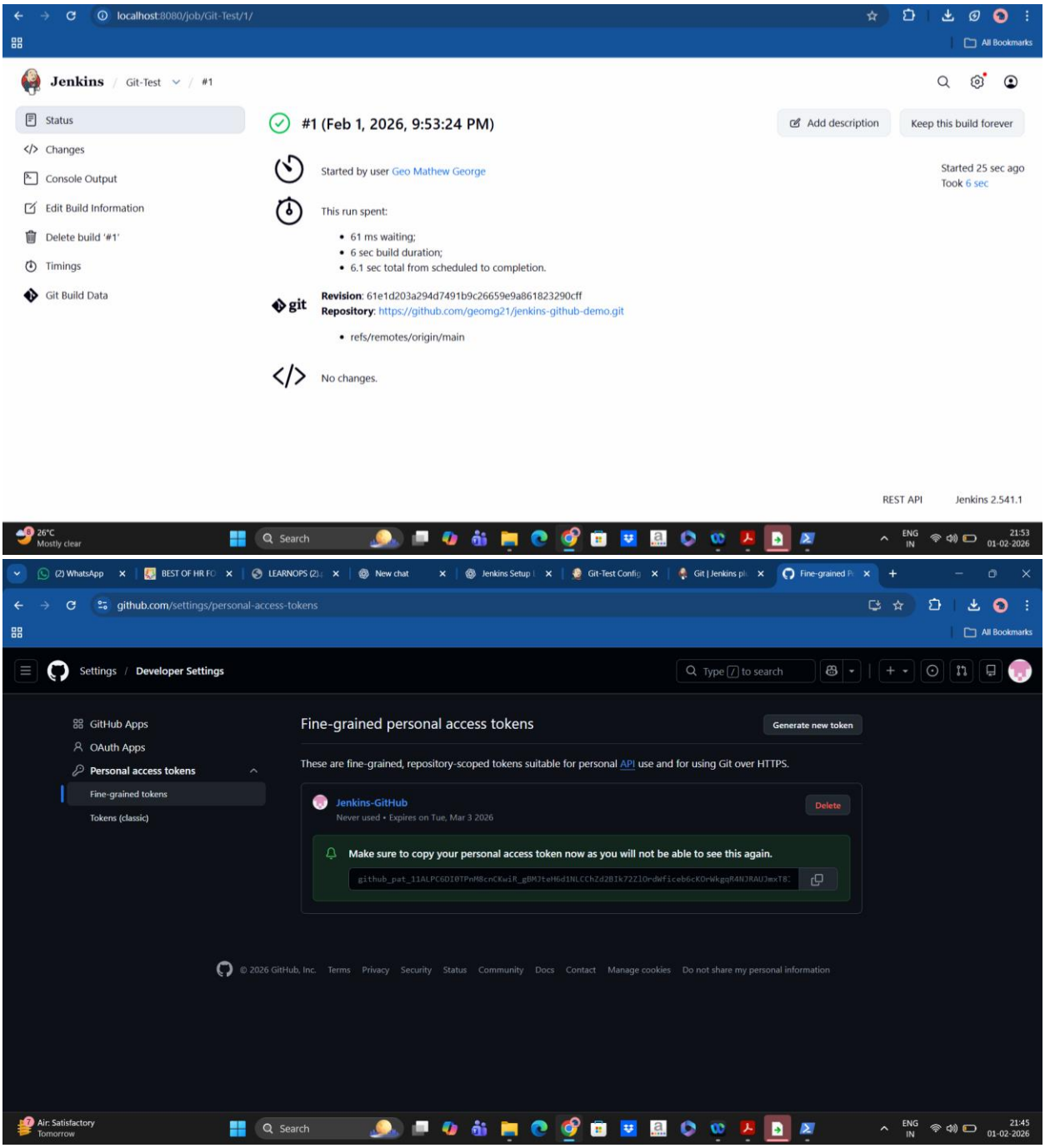

localhost:8080/job/Git-Test/
All Bookmarks
Jenkins
Git-Test
Status
Changes
Workspace
Build Now
Configure
Delete Project
Rename
Builds
Filter
Today
#1 9:53 PM
Git-Test
Add description
Permalinks
Last build (#1), 2 min 11 sec ago
Last stable build (#1), 2 min 11 sec ago
Last successful build (#1), 2 min 11 sec ago
Last completed build (#1), 2 min 11 sec ago
REST API
Jenkins 2.541.1
26°C
Mostly clear
Search
ENG
IN
21:55
01-02-2026


localhost:8080/manage/credentials/store/system/domain/_/credential/github-credentials/
All Bookmarks
Jenkins
Manage Jenkins
Credentials
System
Global credentials (unrestr...
geomg21/****** (GitHub P...
Update
Delete
Move
geomg21/****** (GitHub PAT for Jenkins)
GitHub PAT for Jenkins
Usage
This credential has been recorded as used in the following places:
Note: usage tracking requires the cooperation of plugins and consequently may not track every use.
Git-Test
#1
REST API
Jenkins 2.541.1
26°C
Mostly clear
Search
ENG
IN
21:56
01-02-2026

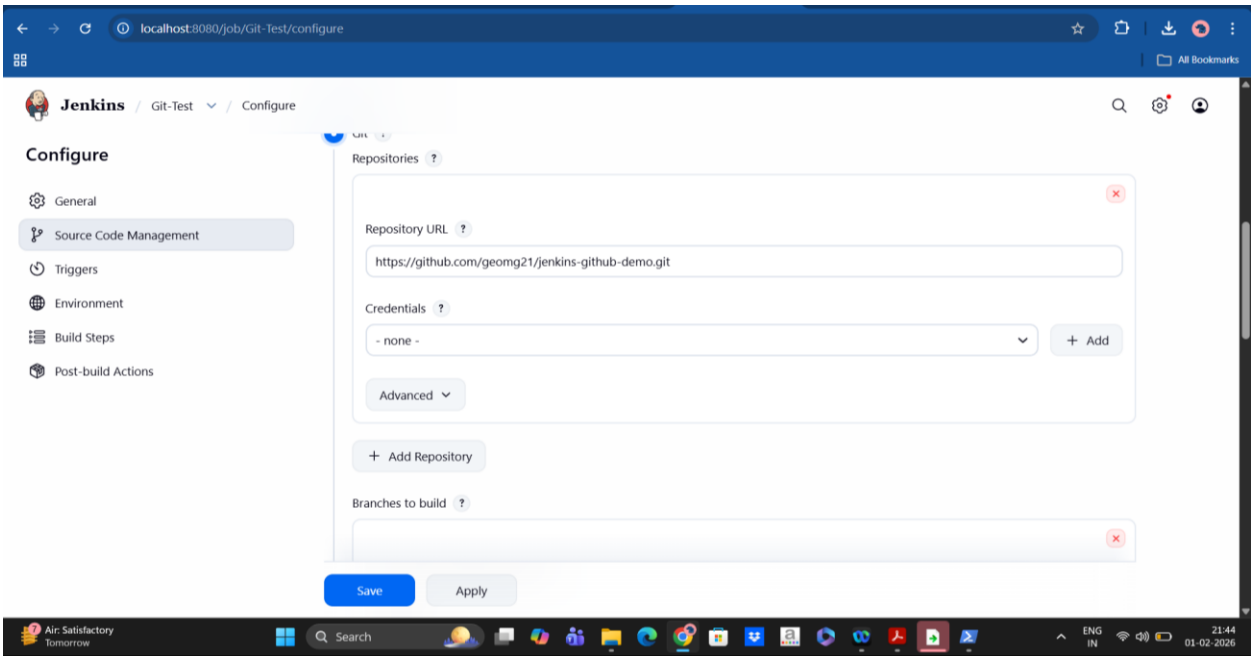

localhost:8080/job/Git-Test/1/
Jenkins / Git-Test / #1
Status
Changes
Console Output
Edit Build Information
Delete build '#1'
Timings
Git Build Data
#1 (Feb 1, 2026, 9:53:24 PM)
Add description
Keep this build forever
Started by user Geo Mathew George
Started 1 min 14 sec ago
Took 6 sec
This run spent:
61 ms waiting;
6 sec build duration;
6.1 sec total from scheduled to completion.
Revision: 61e1d203a294d7491b9c26659e9a861823290cff
Repository: https://github.com/geomg21/jenkins-github-demo.git
refs/remotes/origin/main
No changes.
REST API
Jenkins 2.541.1

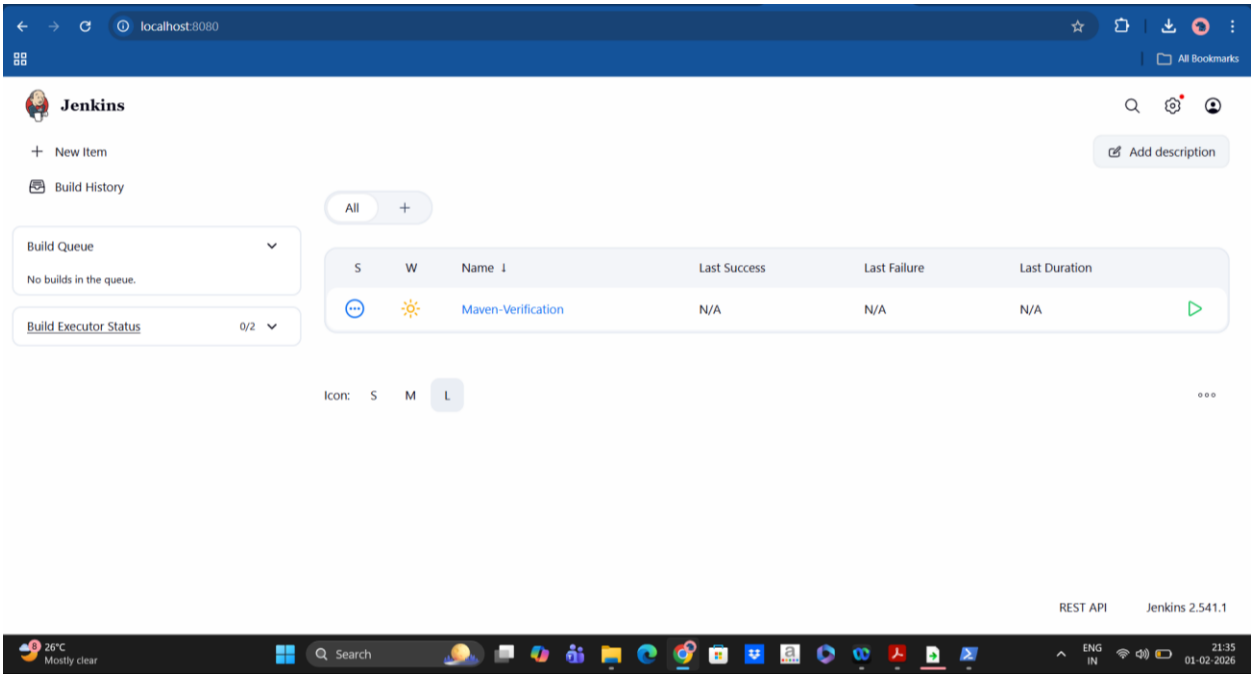
26°C
Mostly clear
Search
ENG
IN
21:55
01-02-2026

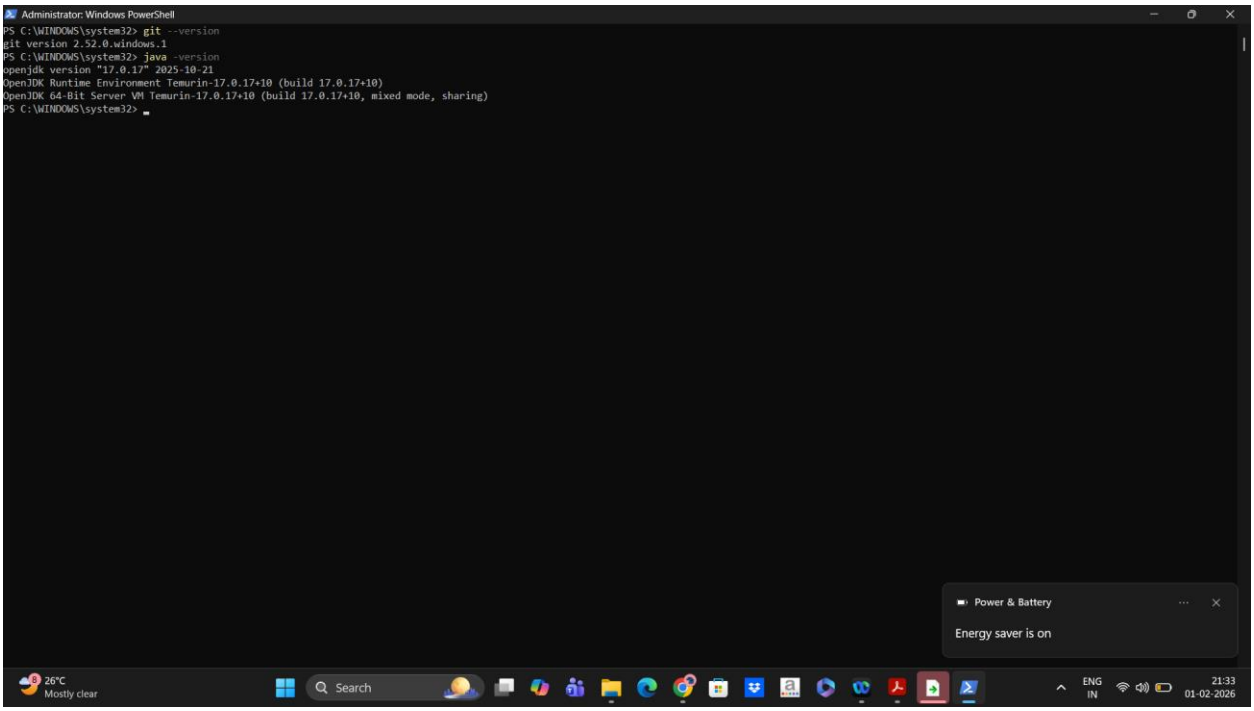

github.com/geomg21/jenkins-github-demo
geomg21 / jenkins-github-demo
Type / to search
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
jenkins-github-demo
Public
Pin
Watch 0
Fork 0
Star 0
main
1 Branch
0 Tags
Go to file
Add file
Code
geomg21 Initial commit
61e1d20 · 18 minutes ago
1 Commit
README.md
Initial commit
18 minutes ago
README
jenkins-github-demo
About
No description, website, or topics provided.
Readme
Activity
0 stars
0 watching
0 forks
Releases
No releases published
Create a new release
Packages
No packages published
Publish your first package
26°C
Mostly clear
Search
ENG
IN
21:54
01-02-2026

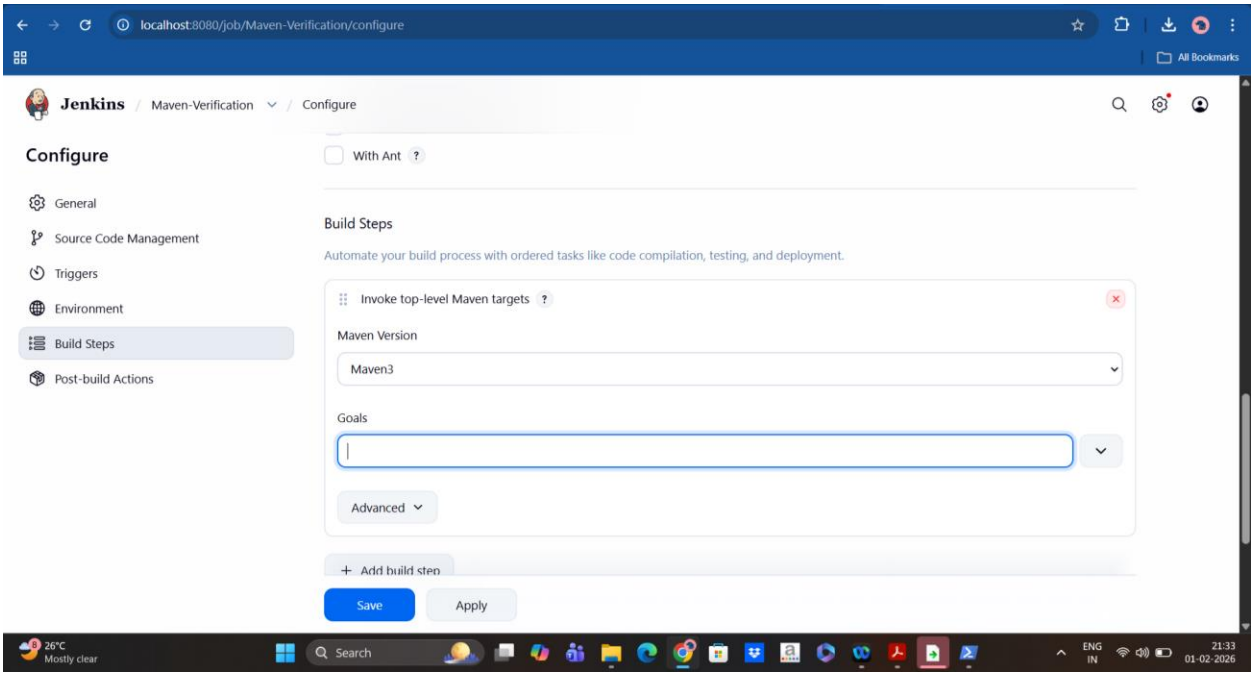

localhost:8080/job/Git-Test/1/
Jenkins / Git-Test / #1
Status
Changes
Console Output
Edit Build Information
Delete build '#1'
Timings
Git Build Data
#1 (Feb 1, 2026, 9:53:24 PM)
Add description
Keep this build forever
Started by user Geo Mathew George
Started 25 sec ago
Took 6 sec
This run spent:
61 ms waiting;
6 sec build duration;
6.1 sec total from scheduled to completion.
Revision: 61e1d203a294d7491b9c26659e9a861823290cff
Repository: https://github.com/geomg21/jenkins-github-demo.git
refs/remotes/origin/main
No changes.
REST API
Jenkins 2.541.1

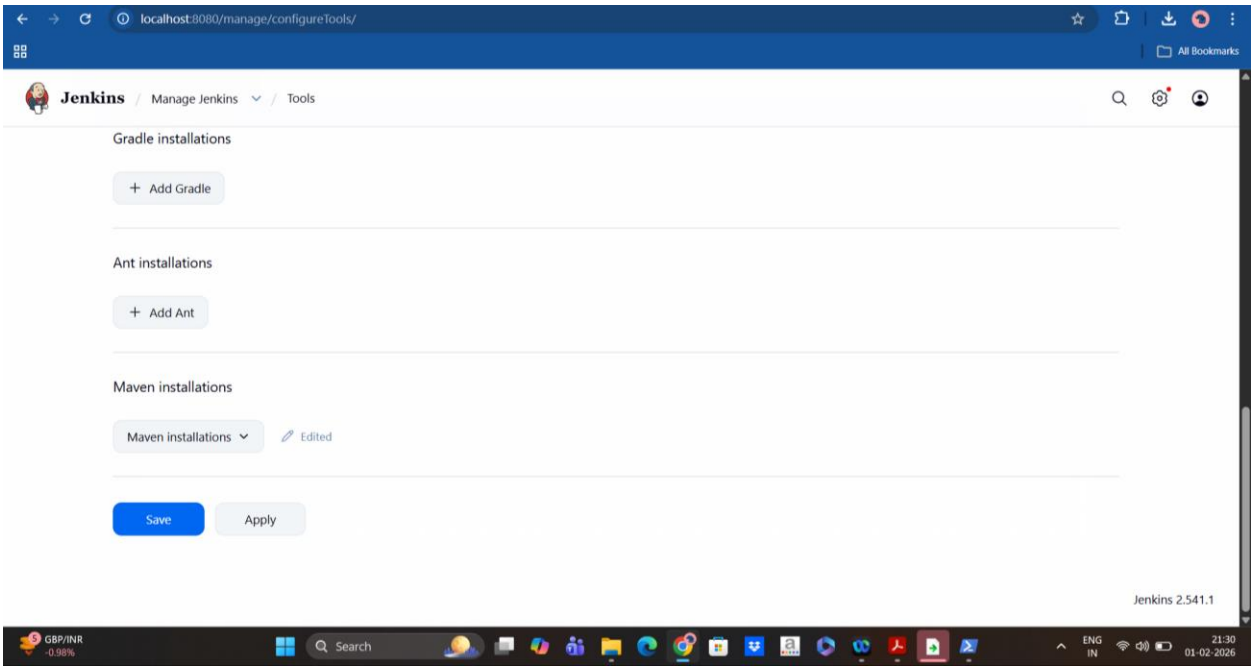
github.com/settings/personal-access-tokens
Settings / Developer Settings
GitHub Apps
OAuth Apps
Personal access tokens
Fine-grained tokens
Tokens (classic)
Fine-grained personal access tokens
Generate new token
These are fine-grained, repository-scoped tokens suitable for personal API use and for using Git over HTTPS.
Jenkins-GitHub
Never used • Expires on Tue, Mar 3 2026
Delete
Make sure to copy your personal access token now as you will not be able to see this again.
© 2026 GitHub, Inc. Terms Privacy Security Status Community Docs Contact Manage cookies Do not share my personal information

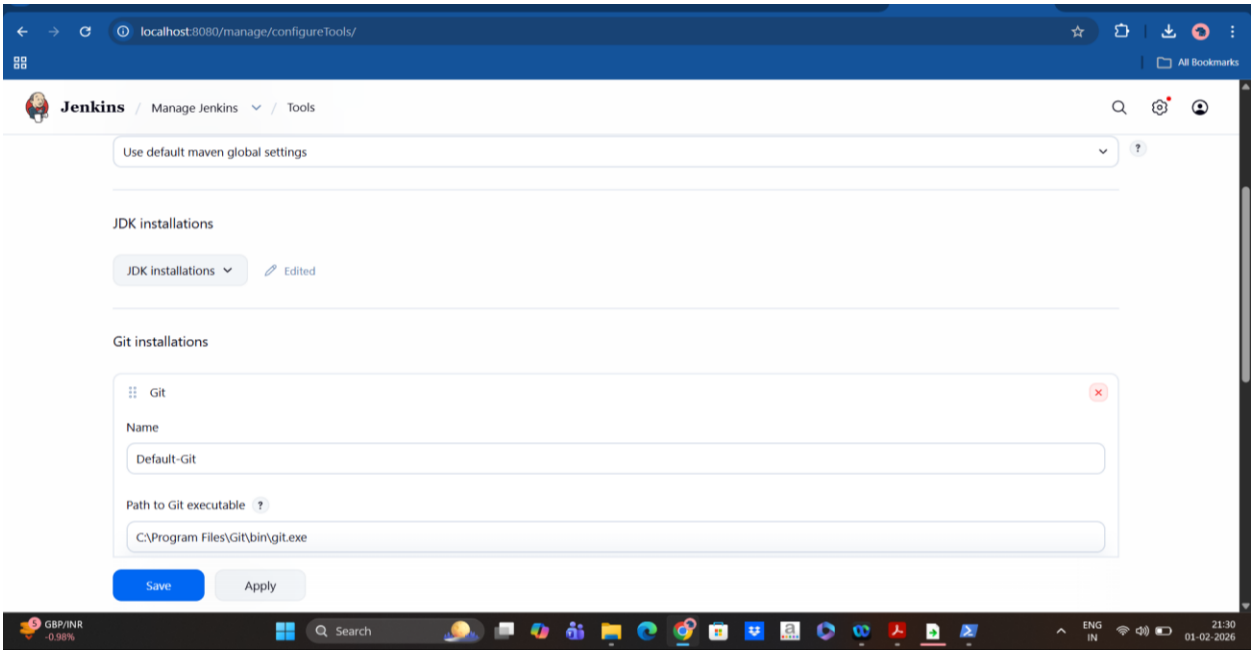

localhost:8080/job/Git-Test/configure
Jenkins
Git-Test
Configure
Configure
General
Source Code Management
Triggers
Environment
Build Steps
Post-build Actions
Repositories
Repository URL
https://github.com/geomg21/jenkins-github-demo.git
Credentials
- none -
Add
Advanced
Add Repository
Branches to build
Save
Apply

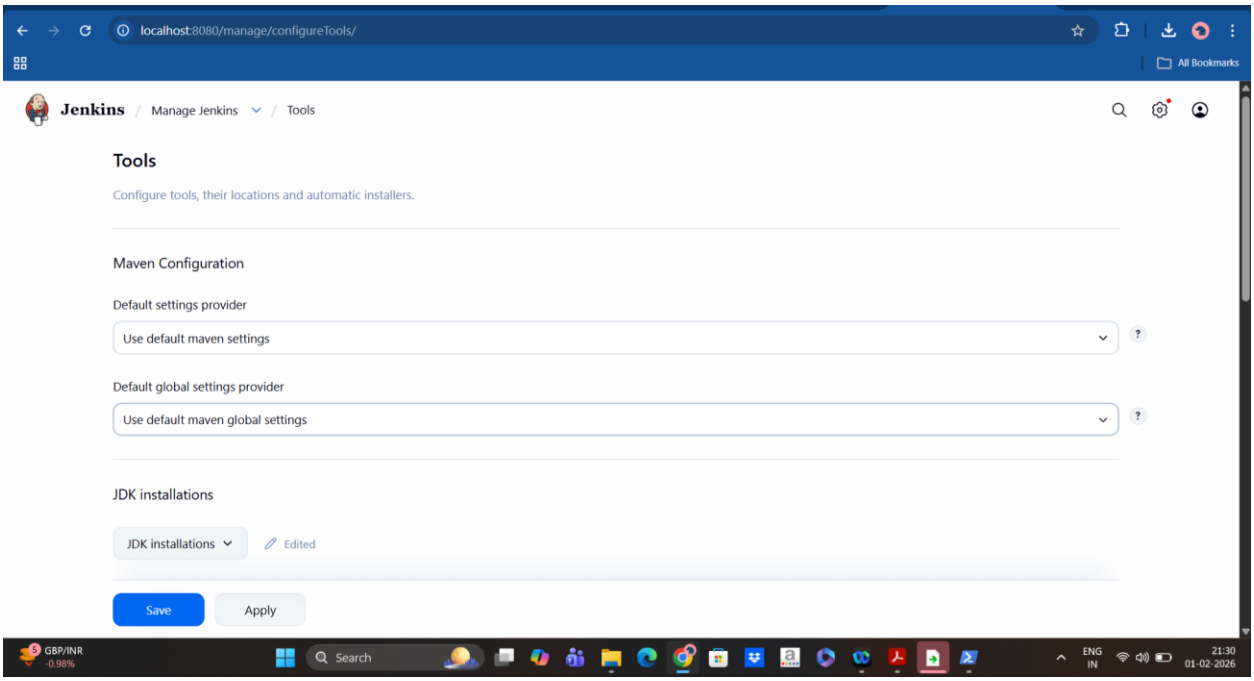

localhost:8080
Jenkins
New Item
Build History
Add description
All
Build Queue
No builds in the queue.
Build Executor Status
0/2
S
W
Name
Last Success
Last Failure
Last Duration
Maven-Verification
N/A
N/A
N/A
Icon: S M L
REST API
Jenkins 2.541.1

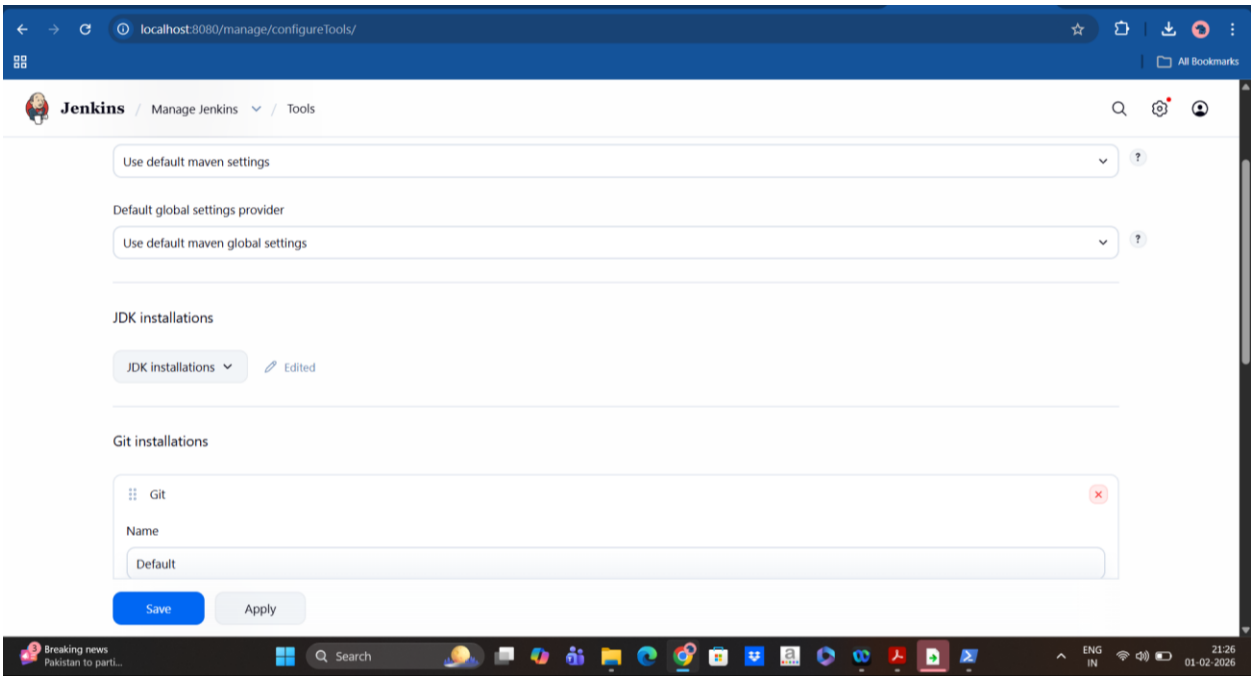

Administrator: Windows PowerShell
PS C:\WINDOWS\system32> git --version
git version 2.52.0.windows.1
PS C:\WINDOWS\system32> java -version
openjdk version "17.0.17" 2025-10-21
OpenJDK Runtime Environment Temurin-17.0.17+10 (build 17.0.17+10)
OpenJDK 64-Bit Server VM Temurin-17.0.17+10 (build 17.0.17+10, mixed mode, sharing)
PS C:\WINDOWS\system32>
Power & Battery
Energy saver is on
26°C
Mostly clear
Search
ENG
IN
21:33
01-02-2026

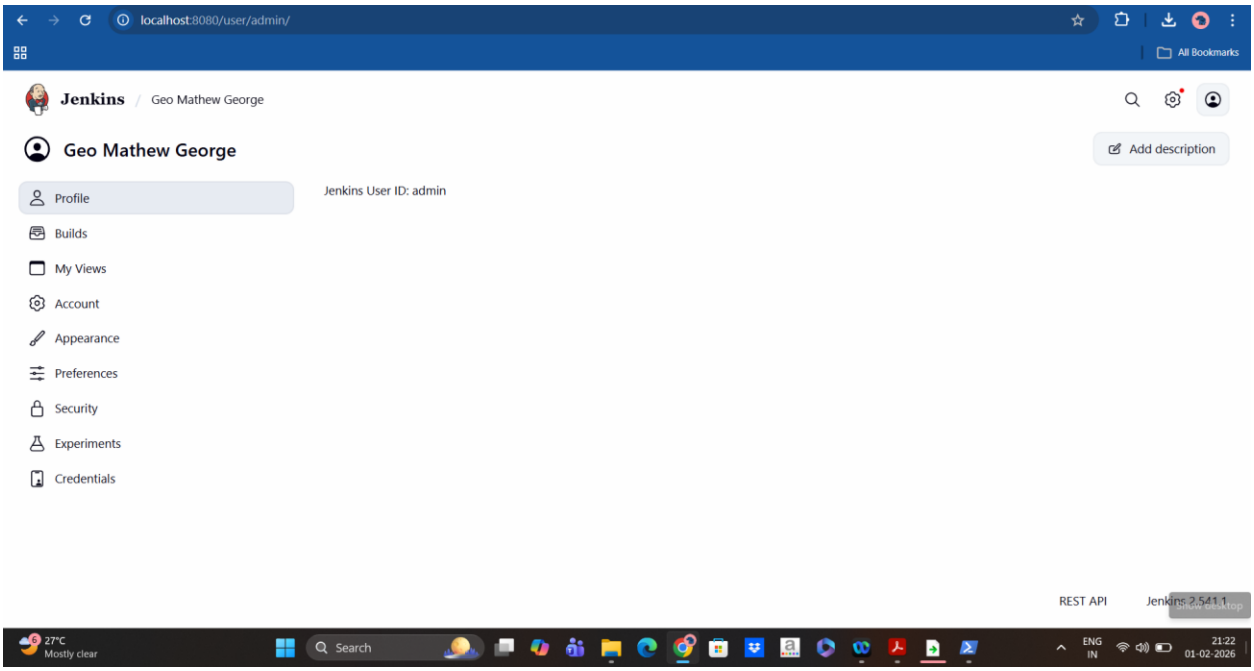

localhost:8080/job/Maven-Verification/configure
All Bookmarks
Jenkins
Maven-Verification
Configure
Configure
General
Source Code Management
Triggers
Environment
Build Steps
Post-build Actions
With Ant
Build Steps
Automate your build process with ordered tasks like code compilation, testing, and deployment.
Invoke top-level Maven targets
Maven Version
Maven3
Goals
Advanced
Add build step
Save
Apply
26°C
Mostly clear
Search
ENG
IN
21:33
01-02-2026


localhost:8080/manage/configureTools/
All Bookmarks
Jenkins / Manage Jenkins / Tools
Gradle installations
+ Add Gradle
Ant installations
+ Add Ant
Maven installations
Maven installations
Edited
Save
Apply
Jenkins 2.541.1
GBP/INR
-0.98%
Search
ENG
IN
21:30
01-02-2026


localhost:8080/manage/configureTools/
All Bookmarks
Jenkins / Manage Jenkins / Tools
Use default maven global settings
JDK installations
JDK installations
Edited
Git installations
Git
Name
Default-Git
Path to Git executable
C:\Program Files\Git\bin\git.exe
Save
Apply
GBP/INR
-0.98%
Search
ENG
IN
21:30
01-02-2026


localhost:8080/manage/configureTools/
All Bookmarks
Jenkins / Manage Jenkins / Tools
Tools
Configure tools, their locations and automatic installers.
Maven Configuration
Default settings provider
Use default maven settings
Default global settings provider
Use default maven global settings
JDK installations
JDK installations
Edited
Save
Apply
GBP/INR
-0.98%
Search
ENG
IN
21:30
01-02-2026


localhost:8080/manage/configureTools/
All Bookmarks
Jenkins / Manage Jenkins / Tools
Use default maven settings
Default global settings provider
Use default maven global settings
JDK installations
JDK installations
Edited
Git installations
Git
Name
Default
Save
Apply
Breaking news
Pakistan to parti...
Search
ENG
IN
21:26
01-02-2026



Final Project Outcome

Through this project:

- A complete CI pipeline was implemented
- Jenkins was integrated with GitHub
- Builds, tests, and notifications were automated
- DevOps best practices were applied in a real project

Real-World Use of Jenkins in DevOps

In real companies, Jenkins is used to **automatically build, test, and deploy software** whenever developers make changes to the code. This removes manual work and makes software delivery faster and safer.

Jenkins Is Used in Real Life

Automating builds, tests, and deployments
Jenkins automatically runs builds and tests instead of developers doing it manually.
Example: When a developer pushes code, Jenkins builds the application and checks if it works.

Handling work from multiple developers
Many developers work on the same project at the same time. Jenkins helps combine their work safely.
Example: If one developer breaks the code, Jenkins detects it immediately.

Maintaining code quality
Jenkins runs tests and checks rules before allowing code to move forward.
Example: If tests fail, Jenkins stops the deployment and notifies the team.

Reducing manual mistakes
Manual deployments can cause errors. Jenkins automates the process to avoid mistakes.
Example: Jenkins deploys the app the same way every time.

Supporting continuous delivery and deployment
Jenkins allows teams to release updates frequently.
Example: Small updates are released daily instead of large risky releases.

Faster and safer releases
Automation helps release software quickly without breaking production.
Example: Jenkins tests every change before it reaches users.

Working with cloud and containers
Jenkins works with cloud platforms and tools like Docker and Kubernetes.
Example: Jenkins builds a Docker image and deploys it to a Kubernetes cluster.

Scaling with distributed agents
Jenkins can run multiple builds at the same time using agents.
Example: One agent runs tests while another builds the application.

Real-World Example

Imagine a company running an **online banking application**.

- Developers push code changes to GitHub
- Jenkins automatically builds and tests the application
- If tests pass, Jenkins deploys the update to a test server
- After approval, Jenkins deploys the update to production
- If anything fails, Jenkins stops the process and alerts the team

This ensures that **only tested and safe code reaches customers**.

Personal Understanding and Learning Outcomes

Through learning and working with Jenkins, I gained a clear and practical understanding of how DevOps automation works in real-world projects. Earlier, software development and deployment felt like two separate activities, but this learning showed me how they are connected through CI/CD pipelines.

I clearly understood how **Continuous Integration and Continuous Delivery** help teams deliver software faster and with better quality. Instead of waiting until the end to test and deploy, Jenkins allows teams to automatically build and test code every time a developer makes a change. This helps identify errors early and avoids bigger problems later.

I learned the importance of **automation** in reducing human errors. Manual builds and deployments often lead to mistakes such as missing files, wrong configurations, or skipped steps. Jenkins removes these risks by following the same automated process every time, which makes software delivery more reliable.

Jenkins helped me understand how a **central CI/CD tool** connects different stages of software development. It pulls code from version control systems, runs builds, executes tests, and can deploy applications automatically. This showed me how development, testing, and deployment work together as a single flow rather than separate tasks.

Working with **Jenkins pipelines** taught me the value of defining workflows as code. Pipelines make the process transparent, reusable, and consistent across environments. If

a pipeline works in one environment, it can be reused in others with minimal changes, which improves stability.

I also learned how DevOps tools like Jenkins support **team collaboration**. Jenkins provides clear feedback through build logs, test reports, and notifications, allowing teams to quickly understand the status of their work. This improves communication and speeds up decision-making.

Overall, this learning experience strengthened my practical understanding of DevOps principles. Jenkins helped me bridge the gap between development and operations by automating repetitive tasks, improving consistency, and ensuring reliable software delivery. I now feel confident working with real-world CI/CD pipelines and DevOps workflows.

4)CI/CD Pipelines

Topics

During this learning period, I studied the core concepts of DevOps and CI/CD to understand how modern software development and delivery work.

The main topics covered include:

- Basics of DevOps and DevOps culture
- Software Development Life Cycle and how DevOps fits into it
- Continuous Integration and its role in development
- Continuous Delivery and how it prepares software for release
- Continuous Deployment and automatic production releases
- Structure and flow of a CI/CD pipeline
- Different stages of a CI/CD pipeline such as source, build, test, and deploy
- Version control systems and Git-based workflows
- Common CI/CD tools like Jenkins, GitHub Actions, GitLab CI/CD, and CircleCI
- Supporting tools such as Docker, Kubernetes, and Infrastructure as Code tools
- Best practices for building effective CI/CD pipelines
- Common challenges and limitations in CI/CD
- Real-world CI/CD workflows used in software teams

Key Concepts

DevOps

DevOps is a way of working where development and operations teams work together instead of separately. The main goal of DevOps is to deliver software faster, with better quality and fewer failures.

From DevOps, I understood the importance of:

- Automation to reduce manual work
- Continuous feedback to improve software quickly
- Shared responsibility between teams
- Faster and more frequent releases

DevOps removes the gap between developers and operations teams. This helps organizations build, test, and deploy software in a smooth and reliable way.

Continuous Integration (CI)

Continuous Integration is the practice of frequently merging code changes into a shared repository. Each time code is pushed, automated builds and tests are triggered.

Key points I learned:

- Developers commit small and frequent code changes
- Every commit triggers an automatic build
- Automated tests run immediately
- Errors are found early in the development process

Benefits of CI:

- Bugs are detected early
- Fewer conflicts when merging code
- Faster feedback to developers
- Better teamwork and collaboration

CI ensures that the application stays stable even when many developers work on it at the same time.

Continuous Delivery (CD)

Continuous Delivery means keeping the application ready for deployment at all times. Code that passes CI is automatically prepared for release.

Key points I learned:

- Only tested and verified builds move forward
- Deployment to production may need manual approval
- Commonly used in enterprise and critical systems
- Focuses on control, safety, and reliability

Continuous Delivery reduces deployment risks and allows organizations to release software whenever needed.

Continuous Deployment

Continuous Deployment is an advanced form of automation where every successful build is automatically deployed to production.

Key points I learned:

- No manual approval is needed after tests pass
- Requires strong automated testing
- Monitoring and rollback are very important
- Mostly used in web and cloud-based applications

This approach allows very fast feature delivery but needs mature DevOps practices to avoid failures.

CI/CD Pipeline Stages

A typical CI/CD pipeline consists of multiple stages, and each stage checks the quality of the code.

Source Stage

- Code is pushed to a Git repository
- The pipeline starts automatically

Build Stage

- Source code is compiled
- Dependencies are downloaded
- Docker images may be created

Test Stage

- Unit tests are executed
- Integration and regression tests are run
- Security tests may be performed

Deploy Stage

- Application is deployed to staging or production
- Deployment methods like rolling, blue-green, or canary are used

Each stage acts as a quality checkpoint and stops faulty code from moving forward.

CI/CD Tools and Supporting Technologies

The main CI/CD tools studied include:

- Jenkins
- GitHub Actions
- GitLab CI/CD
- CircleCI
- Travis CI

Supporting technologies learned:

- Docker for packaging applications
- Kubernetes for managing containers
- Terraform and Ansible for Infrastructure as Code

These tools work together to automate building, testing, deploying applications, and managing infrastructure.

CI/CD Practical Work

Practical 1: Version Control with Git and GitHub

Objective:
To understand version control and team collaboration.

Work Done:
Git was installed, a GitHub repository was created, a local repository was initialized, files were committed, and code was pushed to GitHub.

Concepts Covered:

- Version control
- Git workflow
- Branching and commits

Real-World Use:
Used by all DevOps teams to track code changes and collaborate.

Practical 2: Continuous Integration Using GitHub Actions

Objective:
To implement basic Continuous Integration.

Work Done:
A CI workflow file was created to automatically run build and test steps whenever code was pushed to the repository.

Concepts Covered:

- CI pipelines
- Automated builds
- Event-based triggers

Practical 3: Jenkins Installation and Basic Configuration

Objective:
To install and set up Jenkins for CI/CD.

Work Done:
Java and Jenkins were installed, Jenkins was unlocked, plugins were installed, and an admin user was created.

Concepts Covered:

- CI/CD automation server
- Jenkins architecture
- Plugin management

Practical 4: Jenkins Freestyle Job for CI

Objective:
To create an automated build job in Jenkins.

Work Done:
A Freestyle job was created, connected to GitHub, configured with build triggers, and used to run build commands.

Concepts Covered:

- Automated CI jobs

- Build triggers
- Jenkins workspace

Practical 5: Automated Testing in CI Pipeline

Objective:
To include testing in the CI process.

Work Done:
Unit tests were added, and Jenkins or GitHub Actions was configured to run tests automatically and fail the build if tests failed.

Concepts Covered:

- Unit testing
- Quality checks
- Test automation

Practical 6: Dockerizing an Application

Objective:
To containerize an application.

Work Done:
A Dockerfile was created, a Docker image was built, the container was run locally, and the image was pushed to Docker Hub.

Concepts Covered:

- Docker
- Containerization
- Image lifecycle

Practical 7: CI/CD with Docker Using Jenkins

Objective:
To integrate Docker with Jenkins pipelines.

Work Done:
Docker was installed on the Jenkins server, images were built using Jenkins, and pushed to Docker Hub automatically.

Concepts Covered:

- Jenkins and Docker integration
- Artifact management
- Image automation

Practical 8: Continuous Delivery to Staging Environment

Objective:
To deploy applications automatically to staging.

Work Done:
SSH access was configured and Jenkins was used to deploy Docker containers to a staging server.

Concepts Covered:

- Continuous Delivery
- Environment separation
- Automated deployment

Practical 9: Continuous Deployment to Production

Objective:
To fully automate production deployment.

Work Done:
Successful builds were automatically deployed to production, and rollback strategies were implemented.

Concepts Covered:

- Continuous Deployment
- Zero-downtime deployment
- Rollback handling

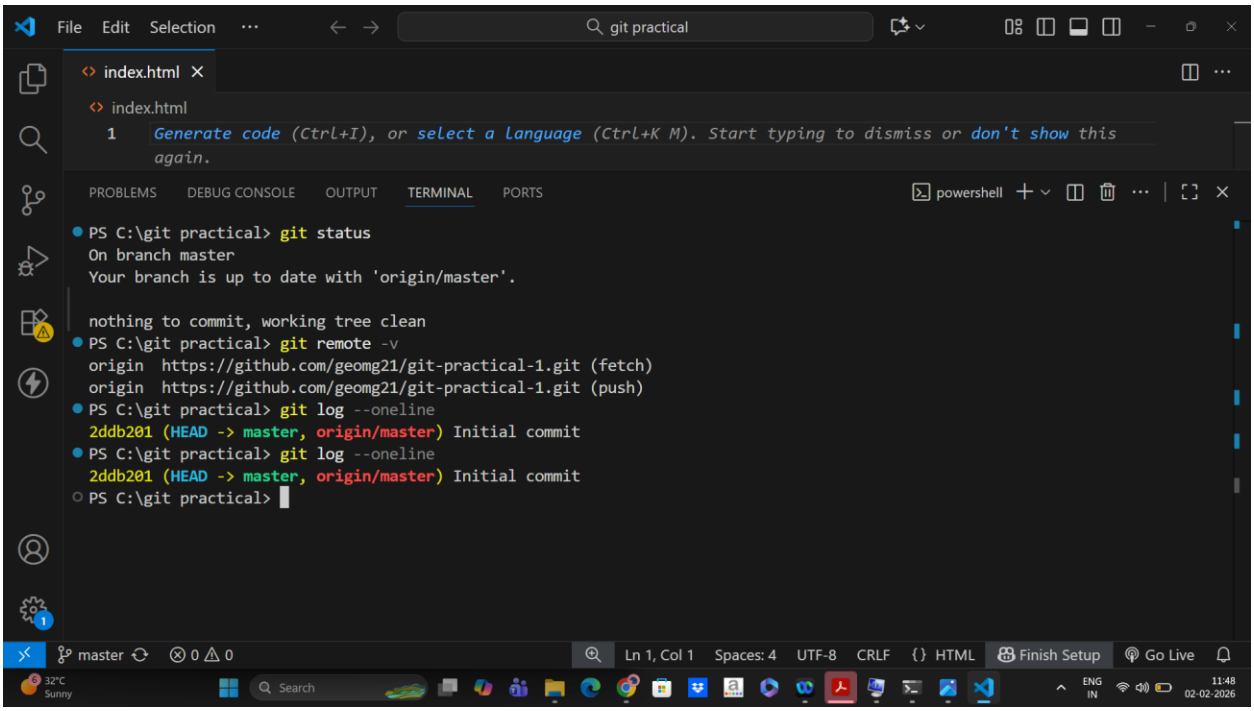
Practical 10: Monitoring, Notifications, and Optimization

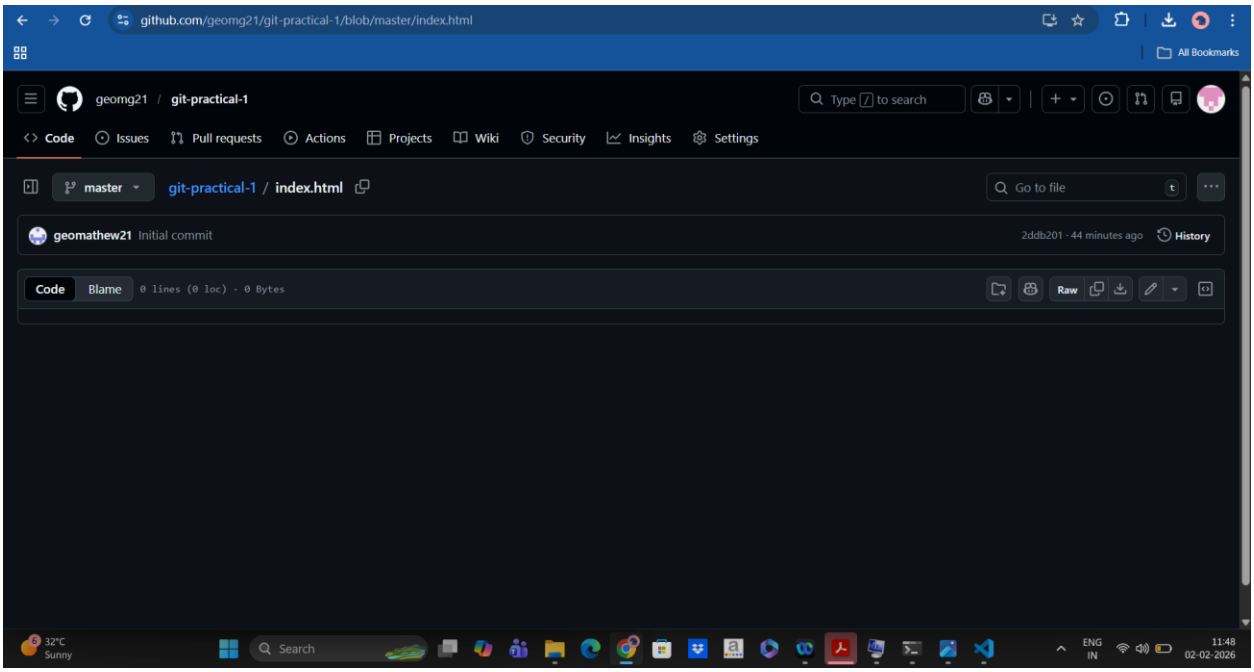
Objective:
To monitor pipelines and improve reliability.

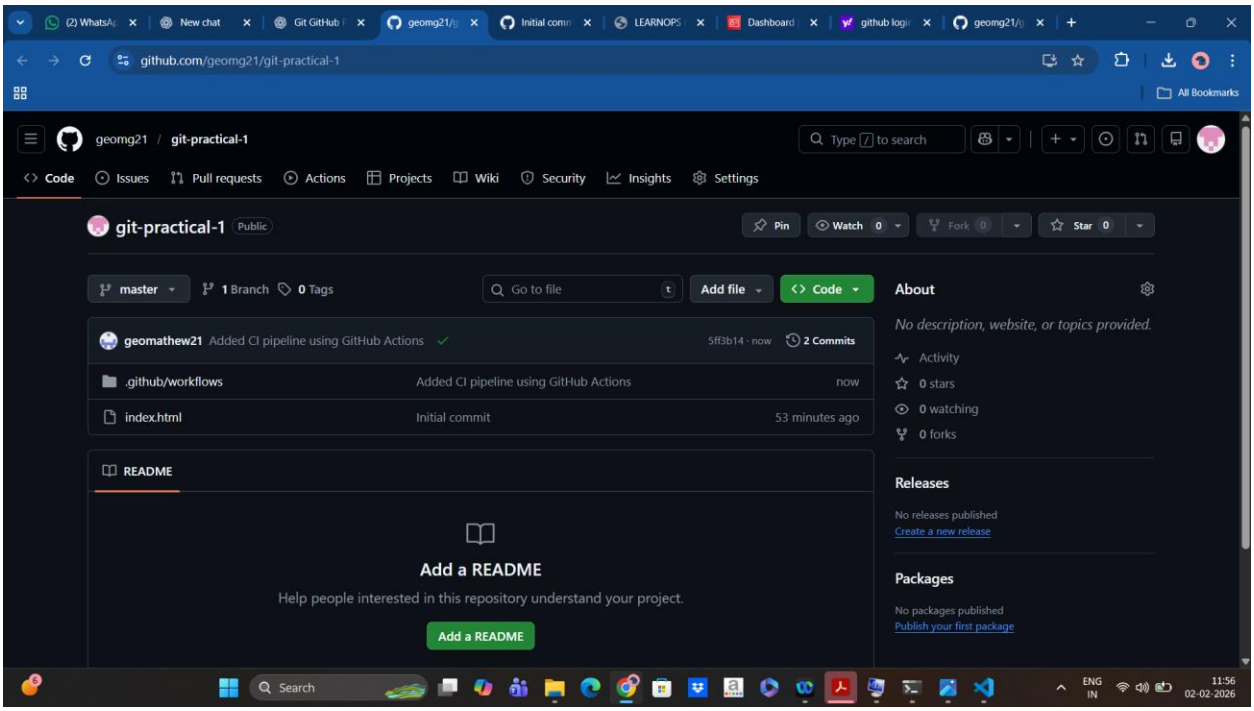
Work Done:
Notifications were configured using email or Slack, logs and metrics were enabled, and pipeline stages were optimized.

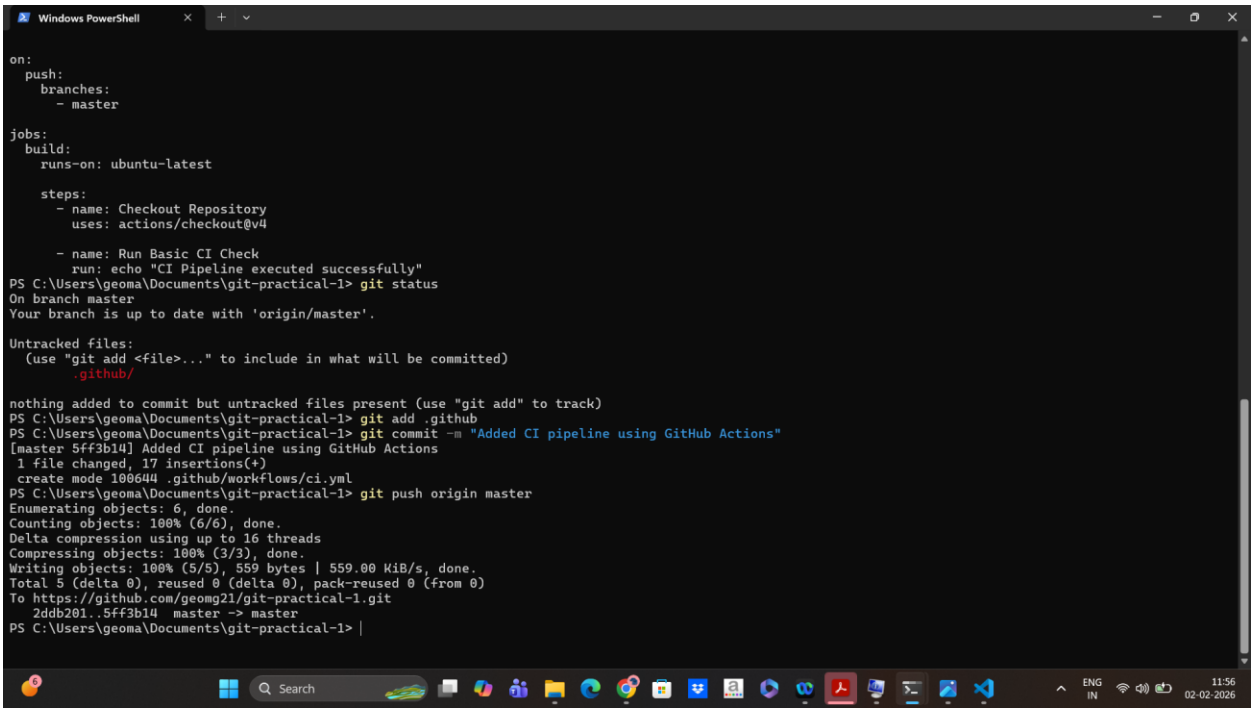
Concepts Covered:

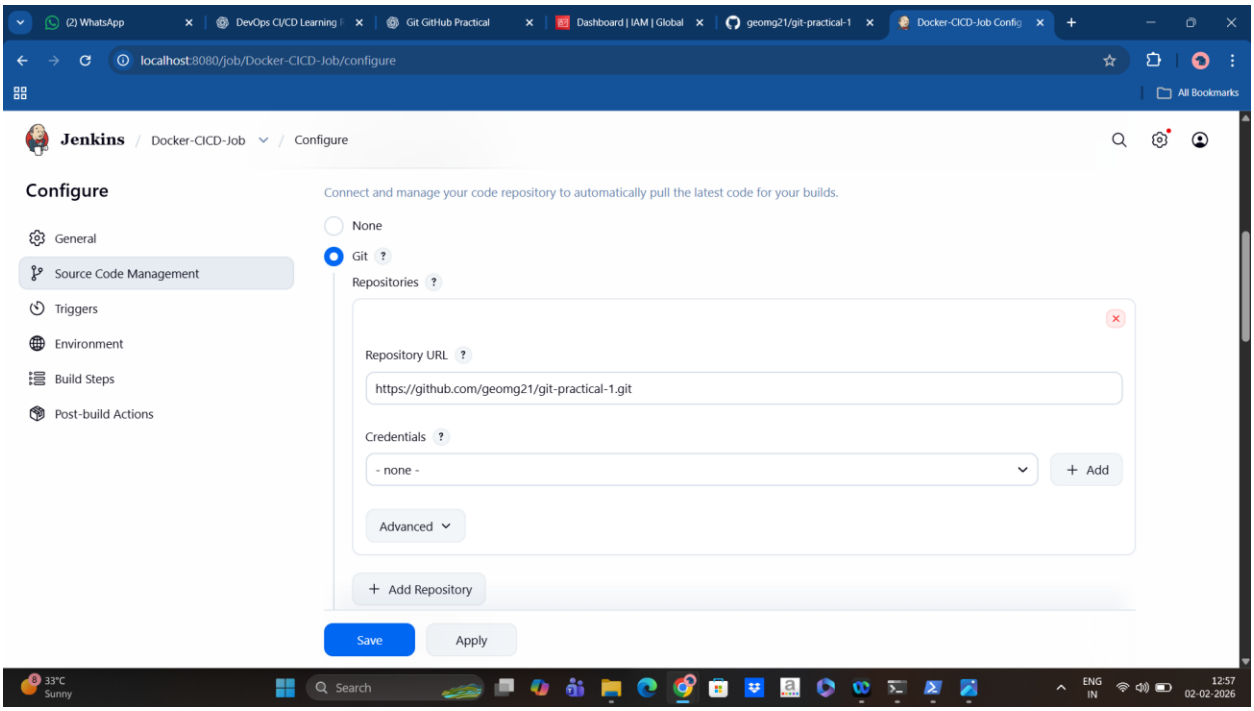
- Monitoring and observability
- Notifications
- CI/CD optimization

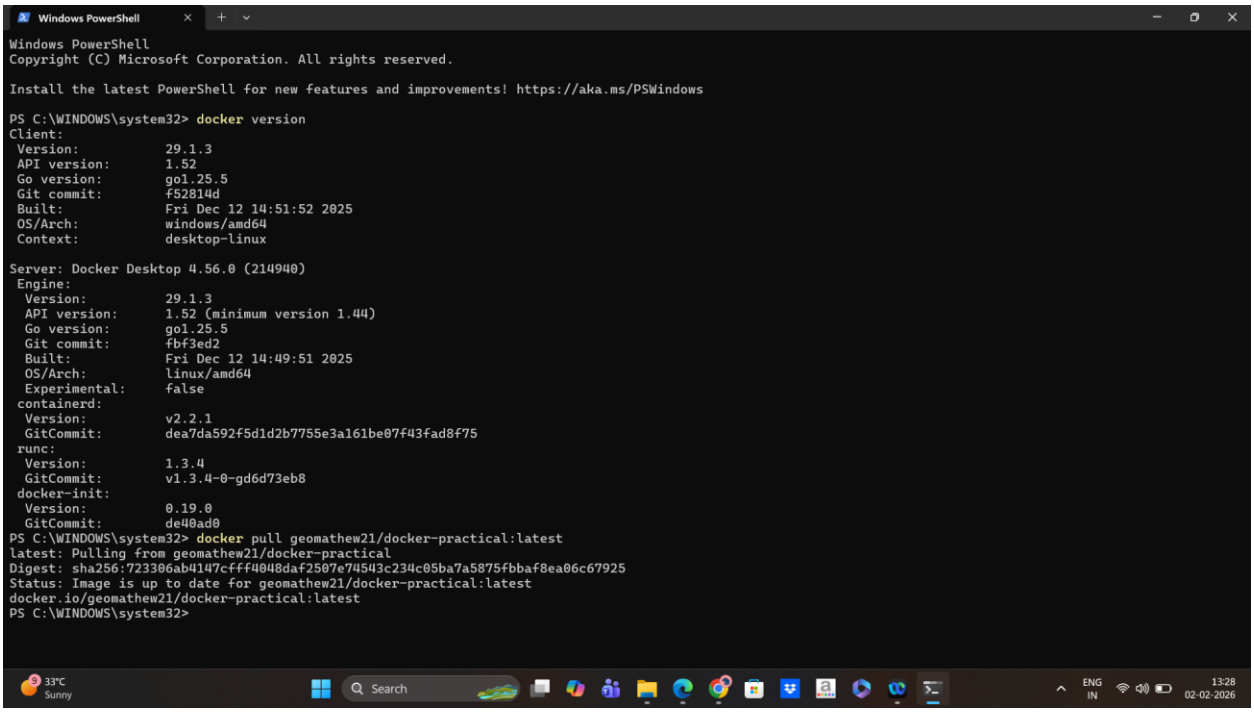

File Edit Selection
git practical
index.html
Generate code (Ctrl+I), or select a language (Ctrl+K M). Start typing to dismiss or don't show this again.
PROBLEMS DEBUG CONSOLE OUTPUT TERMINAL PORTS
powershell
PS C:\git practical> git status
On branch master
Your branch is up to date with 'origin/master'.
nothing to commit, working tree clean
PS C:\git practical> git remote -v
origin https://github.com/geomg21/git-practical-1.git (fetch)
origin https://github.com/geomg21/git-practical-1.git (push)
PS C:\git practical> git log --oneline
2ddb201 (HEAD -> master, origin/master) Initial commit
PS C:\git practical> git log --oneline
2ddb201 (HEAD -> master, origin/master) Initial commit
PS C:\git practical>
master
Ln 1, Col 1 Spaces: 4 UTF-8 CRLF HTML Finish Setup Go Live

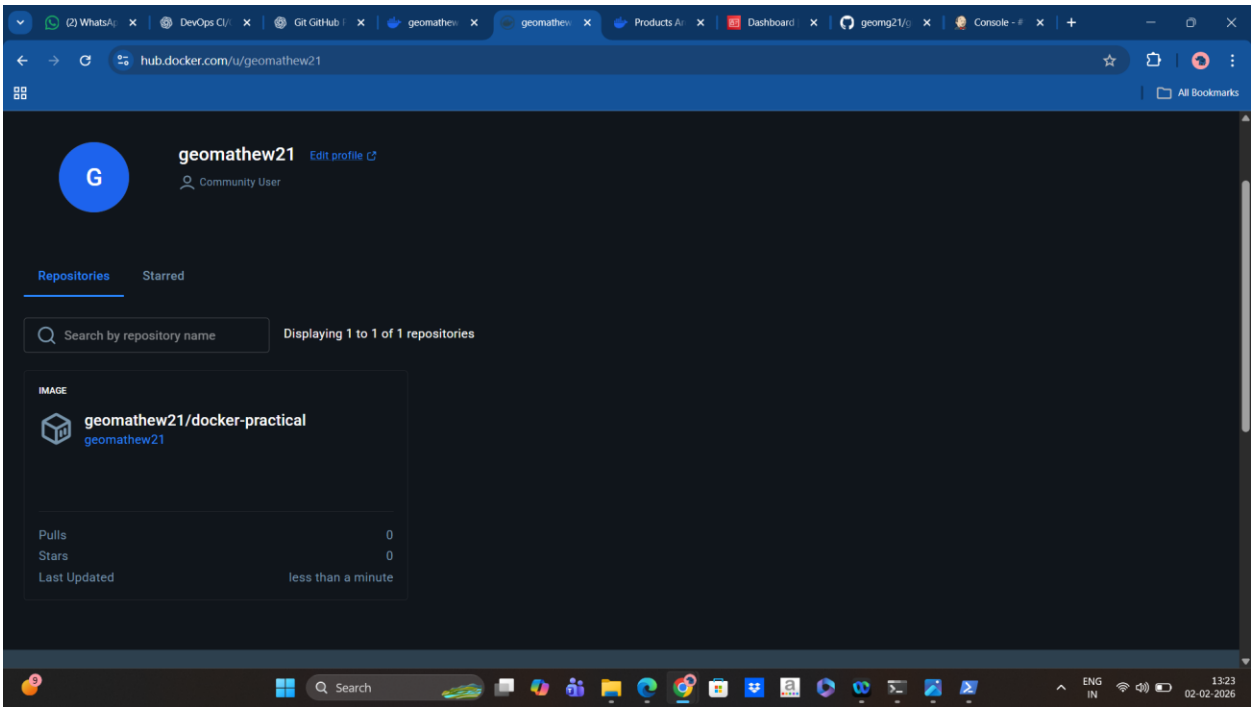

github.com/geomg21/git-practical-1/blob/master/index.html
geomg21 / git-practical-1
Code Issues Pull requests Actions Projects Wiki Security Insights Settings
master
git-practical-1 / index.html
geomathew21 Initial commit
2ddb201 · 44 minutes ago History
Code Blame 0 lines (0 loc) · 0 Bytes
Raw

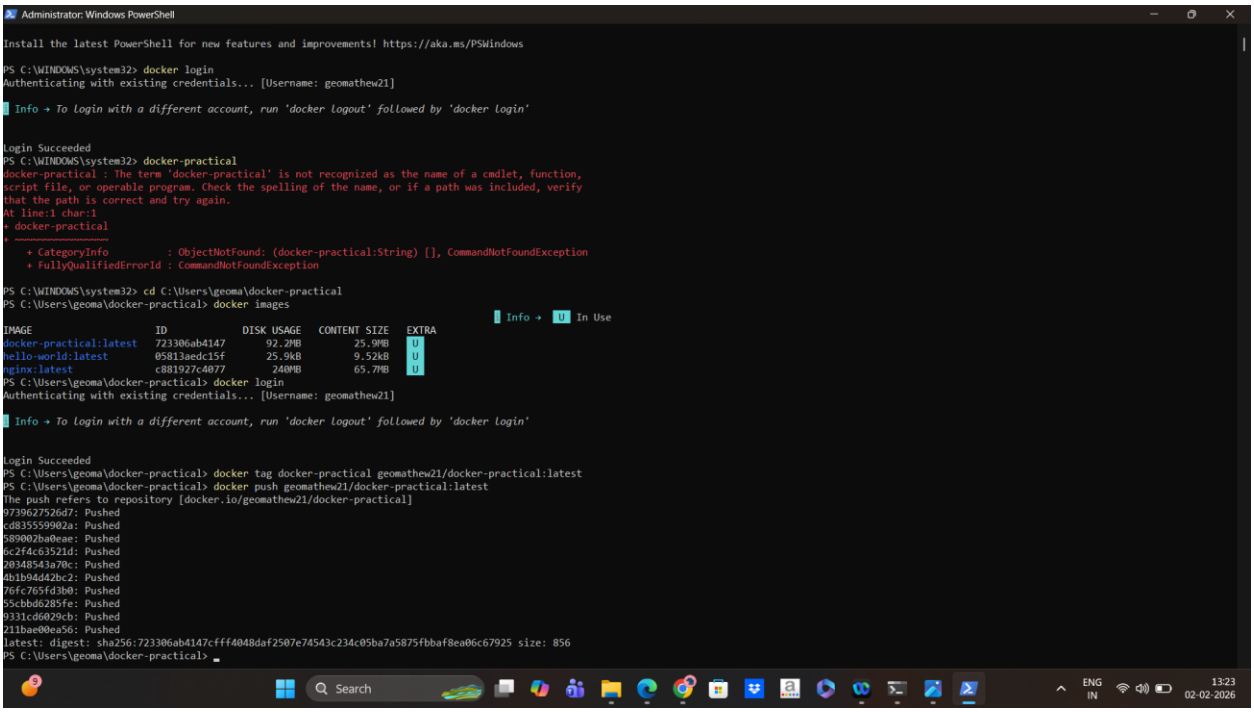

github.com/geomg21/git-practical-1
geomg21 / git-practical-1
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
git-practical-1
Public
master
1 Branch
0 Tags
Go to file
Add file
Code
geomathew21 Added CI pipeline using GitHub Actions
5ff3b14 · now
2 Commits
.github/workflows
Added CI pipeline using GitHub Actions
now
index.html
Initial commit
53 minutes ago
README
Add a README
Help people interested in this repository understand your project.
Add a README
About
No description, website, or topics provided.
Activity
0 stars
0 watching
0 forks
Releases
No releases published
Create a new release
Packages
No packages published
Publish your first package


Windows PowerShell
on:
push:
branches:
- master
jobs:
build:
runs-on: ubuntu-latest
steps:
- name: Checkout Repository
uses: actions/checkout@v4
- name: Run Basic CI Check
run: echo "CI Pipeline executed successfully"
PS C:\Users\geoma\Documents\git-practical-1> git status
On branch master
Your branch is up to date with 'origin/master'.
Untracked files:
(use "git add <file>..." to include in what will be committed)
.github/
nothing added to commit but untracked files present (use "git add" to track)
PS C:\Users\geoma\Documents\git-practical-1> git add .github
PS C:\Users\geoma\Documents\git-practical-1> git commit -m "Added CI pipeline using GitHub Actions"
[master 5ff3b14] Added CI pipeline using GitHub Actions
1 file changed, 17 insertions(+)
create mode 100644 .github/workflows/ci.yml
PS C:\Users\geoma\Documents\git-practical-1> git push origin master
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 16 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (5/5), 559 bytes | 559.00 KiB/s, done.
Total 5 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/geomg21/git-practical-1.git
2ddb201..5ff3b14 master -> master
PS C:\Users\geoma\Documents\git-practical-1>


(2) WhatsApp
DevOps CI/CD Learning
Git GitHub Practical
Dashboard | IAM | Global
geomg21/git-practical-1
Docker-CICD-Job Config
localhost:8080/job/Docker-CICD-Job/configure
All Bookmarks
Jenkins / Docker-CICD-Job / Configure
Configure
General
Source Code Management
Triggers
Environment
Build Steps
Post-build Actions
Connect and manage your code repository to automatically pull the latest code for your builds.
None
Git
Repositories
Repository URL
https://github.com/geomg21/git-practical-1.git
Credentials
- none -
Add
Advanced
Add Repository
Save
Apply
33°C
Sunny
Search
ENG
IN
12:57
02-02-2026


Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.
Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows
PS C:\WINDOWS\system32> docker version
Client:
Version: 29.1.3
API version: 1.52
Go version: go1.25.5
Git commit: f52814d
Built: Fri Dec 12 14:51:52 2025
OS/Arch: windows/amd64
Context: desktop-linux
Server: Docker Desktop 4.56.0 (214940)
Engine:
Version: 29.1.3
API version: 1.52 (minimum version 1.44)
Go version: go1.25.5
Git commit: fbf3ed2
Built: Fri Dec 12 14:49:51 2025
OS/Arch: linux/amd64
Experimental: false
containerd:
Version: v2.2.1
GitCommit: dea7da592f5d1d2b7755e3a161be07f43fad8f75
runc:
Version: 1.3.4
GitCommit: v1.3.4-0-gd6d73eb8
docker-init:
Version: 0.19.0
GitCommit: de40ad0
PS C:\WINDOWS\system32> docker pull geomathew21/docker-practical:latest
latest: Pulling from geomathew21/docker-practical
Digest: sha256:723306ab4147cfff4048daf2507e74543c234c05ba7a5875fbbaf8ea06c67925
Status: Image is up to date for geomathew21/docker-practical:latest
docker.io/geomathew21/docker-practical:latest
PS C:\WINDOWS\system32>
33°C
Sunny
Search
ENG
IN
13:28
02-02-2026





Overall Outcome

- Complete CI/CD pipeline implemented
- Automation from code commit to deployment achieved
- Integration of Git, Jenkins, Docker, and cloud servers

- Real-world DevOps workflow understood and practiced

Real-World Use of These Tools in DevOps Environments

- In real-world DevOps environments, CI/CD pipelines are used to:
- Automatically build and test applications on every code change
- Deploy applications across multiple environments (dev, test, staging, production)
- Reduce human error in deployments
- Enable faster time-to-market
- Improve system stability and reliability
- Support microservices and cloud-native architectures
- Enable rollback and disaster recovery
- Large organizations use CI/CD to release features multiple times per day while maintaining high quality and security standards
- CICD_Pillar_PDFdownload
- .

Personal Understanding and Learning Outcomes

- Through this learning process, I gained a strong understanding of how CI/CD pipelines transform traditional software development.

- **Key learning outcomes:**

- Understood how automation improves speed and reliability
- Learned the importance of small, frequent code changes
- Gained clarity on CI vs Continuous Delivery vs Continuous Deployment
- Understood real-world pipeline design and quality gates
- Recognized the role of testing and monitoring in DevOps success
- Developed confidence in designing and explaining CI/CD workflows
- This learning has helped me understand how modern software companies deliver high-quality applications continuously and reliably.

4) Docker

Topics

During this learning period, I studied the following DevOps concepts and tools:

- DevOps fundamentals and lifecycle
- Version Control using Git and GitHub
- Continuous Integration and Continuous Deployment (CI/CD)
- Jenkins for automation
- Docker and containerization
- Infrastructure as Code (IaC) using Terraform
- Cloud basics with AWS (IAM, EC2)
- Monitoring and maintenance concepts

Key Concepts

DevOps

- DevOps is a combination of development and operations practices.
- Focuses on faster delivery, automation, collaboration, and reliability.
- Key stages: Plan, Code, Build, Test, Release, Deploy, Operate, Monitor.

Git & GitHub

- Git is a distributed version control system.
- GitHub is used to host and manage repositories.
- Concepts learned:
    - Repositories, commits, branches, merge
    - Push, pull, clone
    - Collaboration using remote repositories

Jenkins

- Jenkins is an automation server used for CI/CD.
- Key concepts:
    - Jobs and pipelines
    - Build triggers
    - Plugins
    - Console output and logs
- Jenkins automates build, test, and deployment processes.

Docker

- Docker is a containerization platform.
- Key concepts:
    - Images and containers
    - Dockerfile
    - Docker Hub and repositories
    - Volumes and networking
- Containers are lightweight, portable, and consistent across environments.

Terraform

- Terraform is an Infrastructure as Code (IaC) tool.
- Key concepts:
    - Providers
    - Resources
    - Variables
    - State files
- Used to create and manage infrastructure using code.

AWS

- Learned basic cloud concepts.
- Key services:
    - IAM (users, roles, permissions)
    - EC2 (virtual servers)
- Understood cloud scalability and security basics

Docker Practical Work

Practical 1: Docker Installation and Verification

Objective:
To install Docker and confirm it works correctly.

Work Done:
Docker was installed on the system. The Docker version was checked, and the hello-world container was run to verify successful installation.

Concepts Covered:

- Docker installation
- Docker Engine
- Basic Docker commands

Practical 2: Working with Docker Images

Objective:
To understand Docker images and how they are managed.

Work Done:
Images were pulled from Docker Hub, listed locally, inspected, and unused images were removed.

Concepts Covered:

- Docker images
- Docker Hub
- Image lifecycle

Practical 3: Creating and Managing Containers

Objective:
To learn how to create and manage containers.

Work Done:
Containers were created in interactive and detached modes. Containers were started, stopped, restarted, deleted, and their status was checked.

Concepts Covered:

- Containers
- Container lifecycle
- Docker run options

Practical 4: Building Docker Images Using Dockerfile

Objective:
To create custom Docker images.

Work Done:
A Dockerfile was written for a simple application. The image was built and run using Docker commands.

Concepts Covered:

- Dockerfile
- Image building
- Layered architecture

Practical 5: Docker Volumes and Data Persistence

Objective:
To understand data persistence in Docker.

Work Done:
Named volumes were created and attached to containers. Data was verified to persist even after container deletion.

Concepts Covered:

- Docker volumes

- Data persistence
- Storage management

Practical 6: Docker Networking

Objective:
To understand container communication.

Work Done:
Custom Docker networks were created. Multiple containers were connected to the same network and communication was tested.

Concepts Covered:

- Docker networking
- Bridge and host networks
- Container communication

Practical 7: Docker Compose for Multi-Container Applications

Objective:
To deploy a multi-container application.

Work Done:
A docker-compose file was written to run a web application with a database. Services were started and scaled using Docker Compose.

Concepts Covered:

- Docker Compose
- Service orchestration
- Multi-container applications

Practical 8: Docker Registry and Docker Hub

Objective:
To share Docker images.

Work Done:
A Docker Hub account was created. Images were tagged, pushed to Docker Hub, and pulled on another system.

Concepts Covered:

- Docker registry
- Image tagging
- Image distribution

Practical 9: Docker Monitoring, Logs, and Debugging

Objective:
To monitor and troubleshoot containers.

Work Done:
Container logs were viewed, resource usage was monitored, commands were executed inside containers, and container details were inspected.

Concepts Covered:

- Logging
- Monitoring
- Debugging

Practical 10: Docker Security

Objective:
To apply Docker security and optimization techniques.

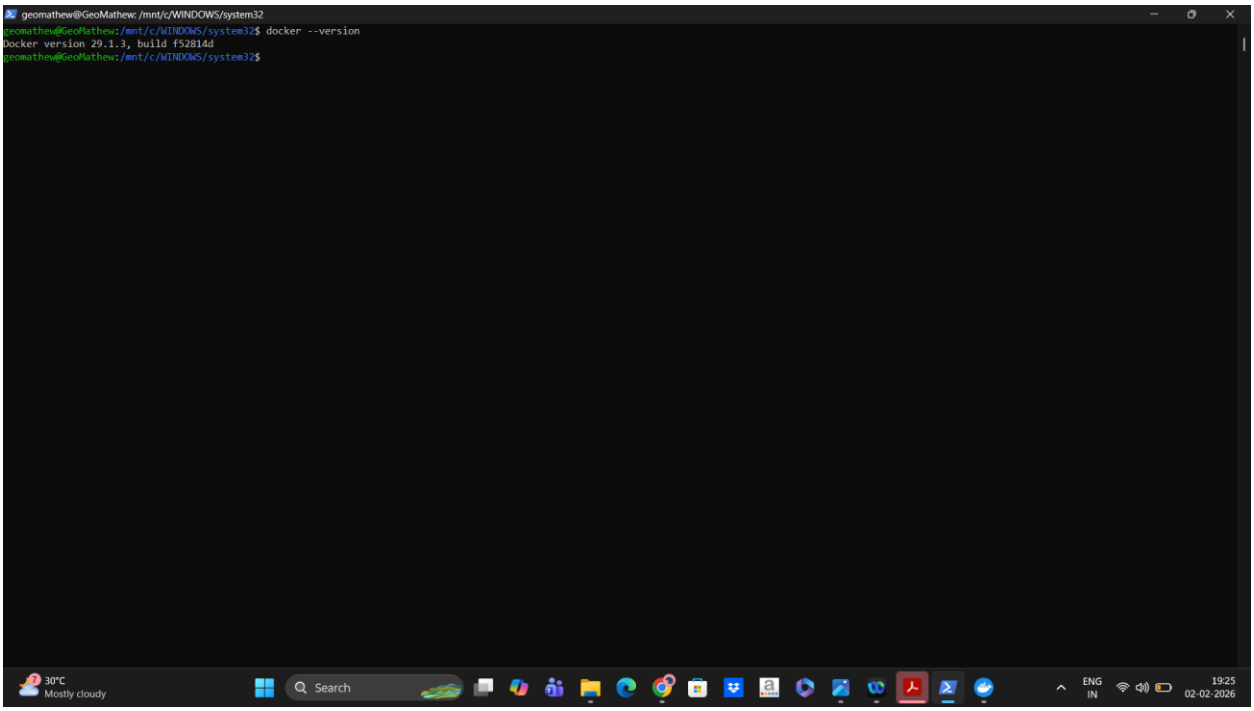
Work Done:
Official base images were used, unnecessary files were ignored, resource limits were applied, and containers were run with non-root users.

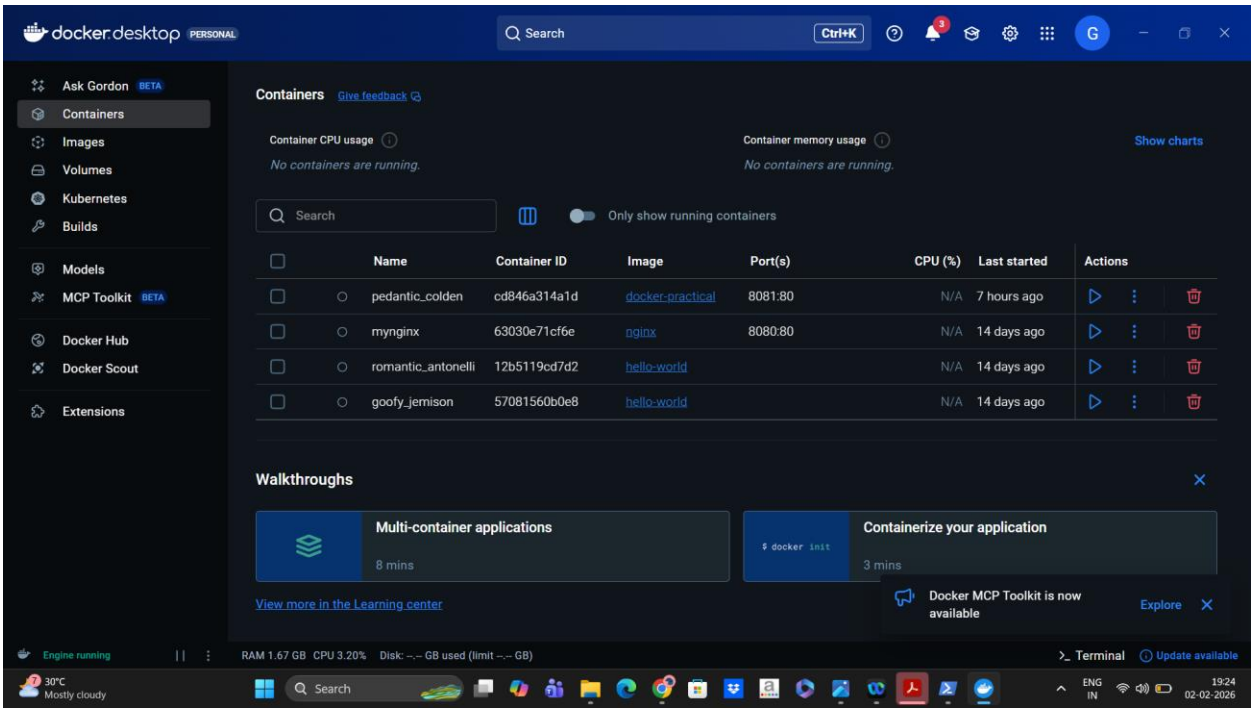
Concepts Covered:

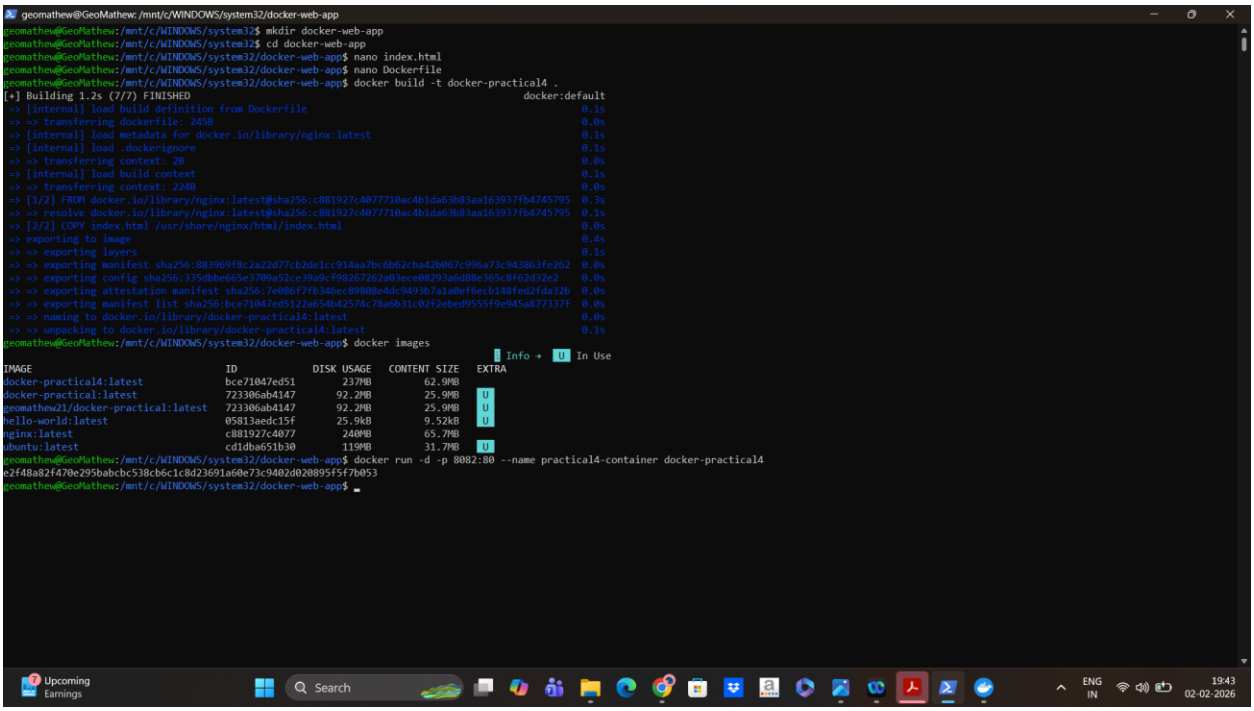
- Docker security
- Best practices
- Performance optimization

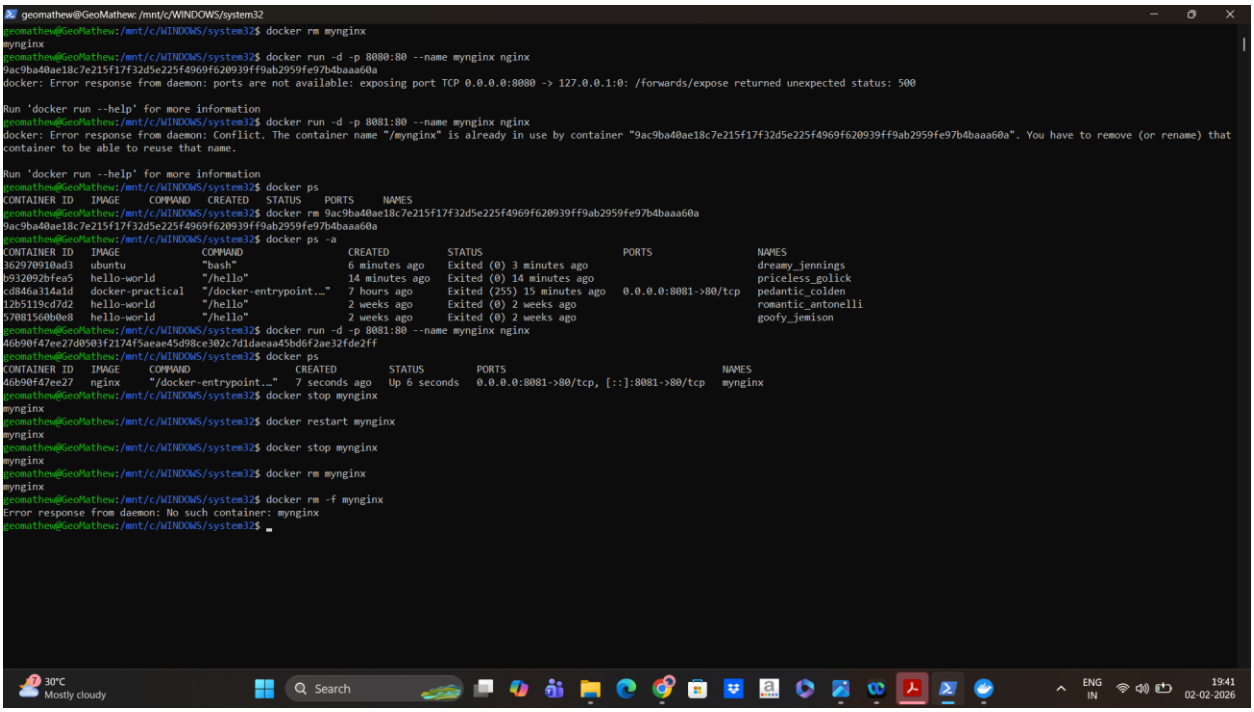
Overall Outcome

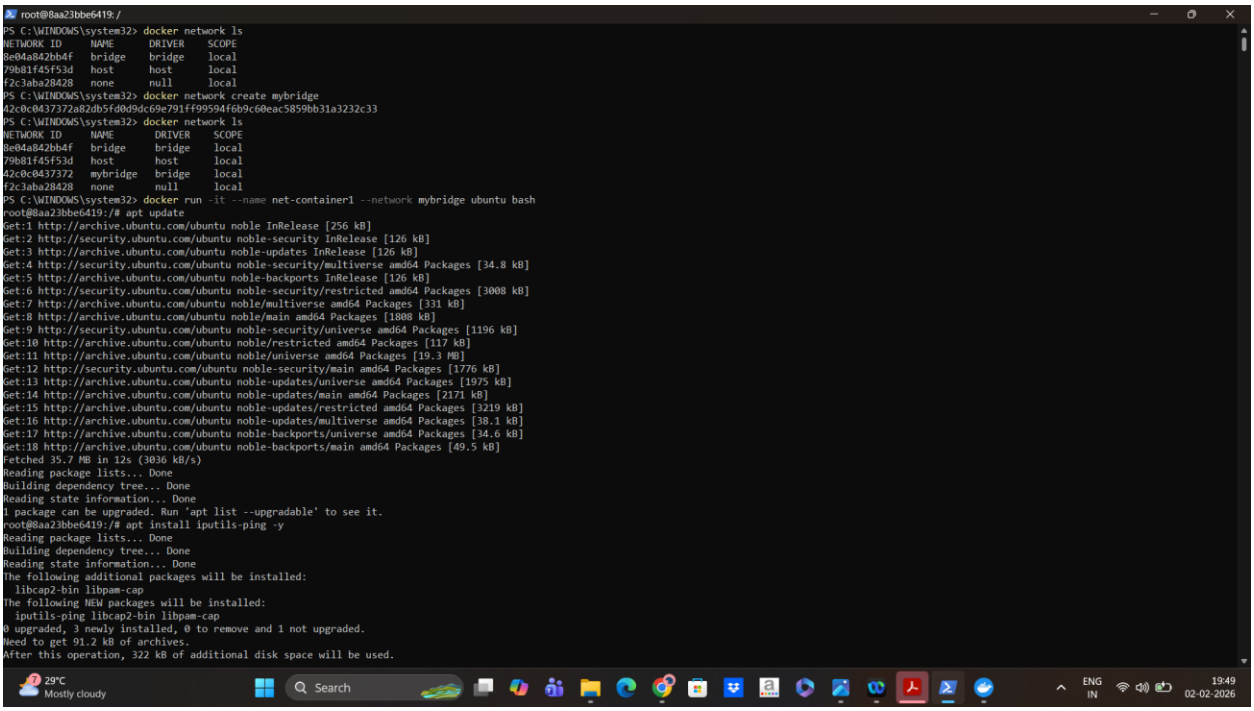
- Gained hands-on experience with Docker
- Understood image creation, container management, and networking
- Learned how to deploy multi-container applications
- Practiced Docker security and best practices

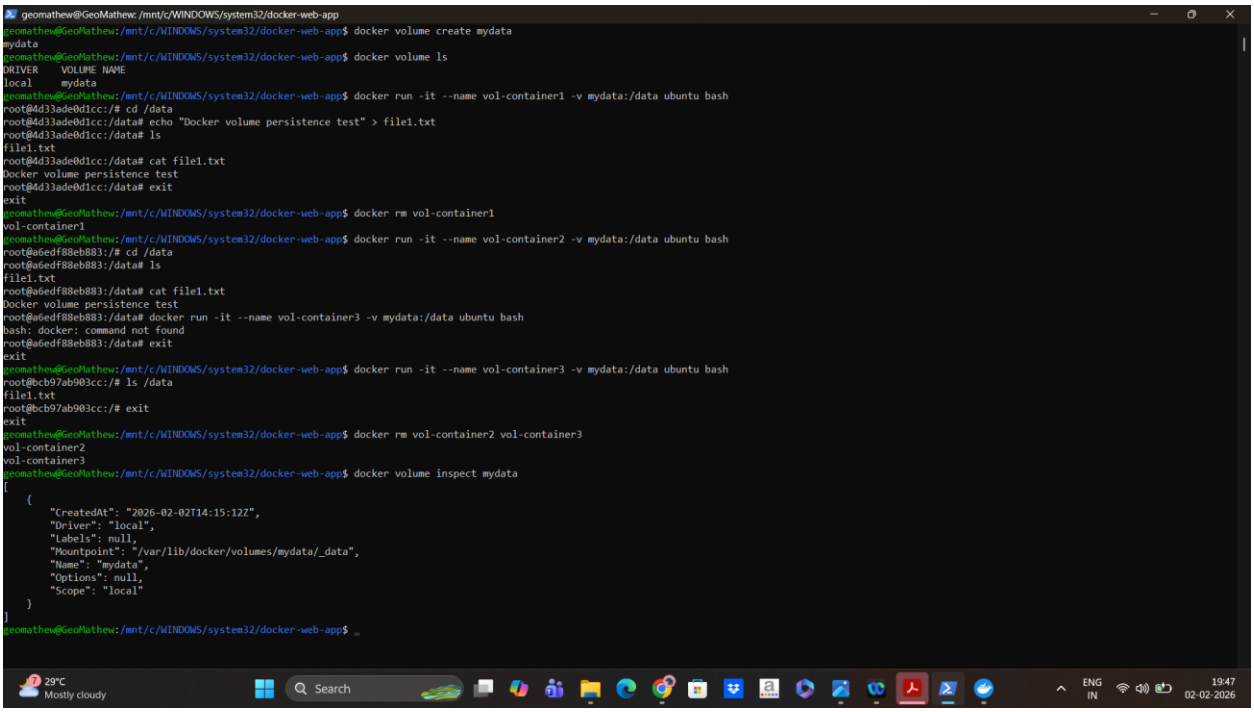

geomathew@GeoMathew: /mnt/c/WINDOWS/system32
geomathew@GeoMathew:/mnt/c/WINDOWS/system32$ docker --version
Docker version 29.1.3, build f52814d
geomathew@GeoMathew:/mnt/c/WINDOWS/system32$
30°C
Mostly cloudy
Search
ENG
IN
19:25
02-02-2026

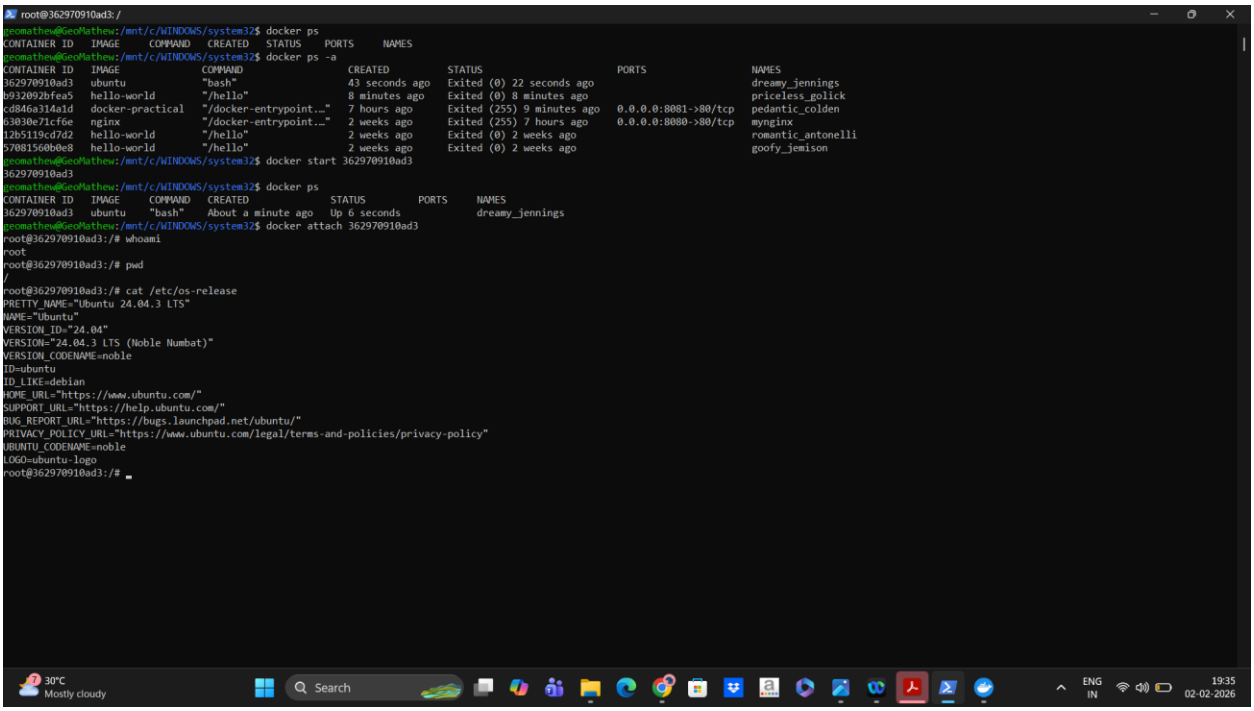

docker desktop
PERSONAL
Search
Ctrl+K
Ask Gordon
BETA
Containers
Images
Volumes
Kubernetes
Builds
Models
MCP Toolkit
BETA
Docker Hub
Docker Scout
Extensions
Containers
Give feedback
Container CPU usage
No containers are running.
Container memory usage
No containers are running.
Show charts
Search
Only show running containers
Name
Container ID
Image
Port(s)
CPU (%)
Last started
Actions
pedantic_colden
cd846a314a1d
docker-practical
8081:80
N/A
7 hours ago
mynginx
63030e71cf6e
nginx
8080:80
N/A
14 days ago
romantic_antonelli
12b5119cd7d2
hello-world
N/A
14 days ago
goofy_jemison
57081560b0e8
hello-world
N/A
14 days ago
Walkthroughs
Multi-container applications
8 mins
$ docker init
Containerize your application
3 mins
View more in the Learning center
Docker MCP Toolkit is now available
Explore
Engine running
RAM 1.67 GB CPU 3.20% Disk: --.-- GB used (limit --.-- GB)
Terminal
Update available
30°C
Mostly cloudy
Search
ENG
IN
19:24
02-02-2026

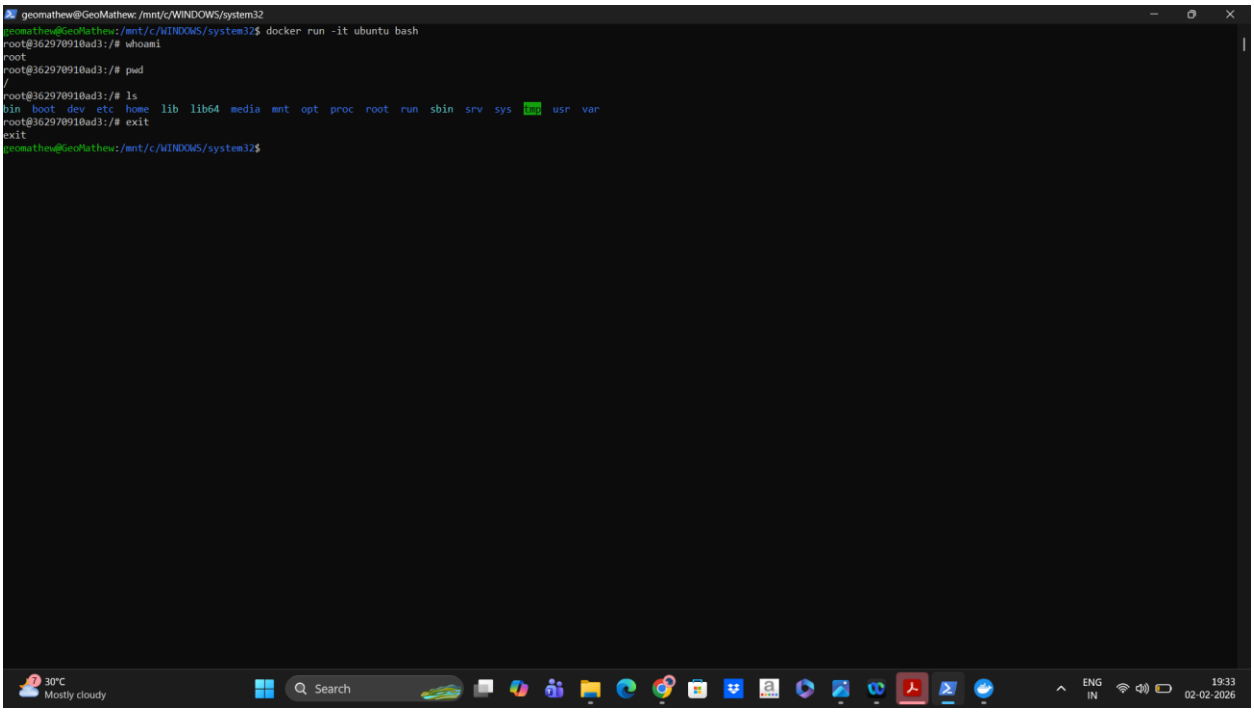

geomathew@GeoMathew: /mnt/c/WINDOWS/system32/docker-web-app
geomathew@GeoMathew:/mnt/c/WINDOWS/system32$ mkdir docker-web-app
geomathew@GeoMathew:/mnt/c/WINDOWS/system32$ cd docker-web-app
geomathew@GeoMathew:/mnt/c/WINDOWS/system32/docker-web-app$ nano index.html
geomathew@GeoMathew:/mnt/c/WINDOWS/system32/docker-web-app$ nano Dockerfile
geomathew@GeoMathew:/mnt/c/WINDOWS/system32/docker-web-app$ docker build -t docker-practical4 .
[+] Building 1.2s (7/7) FINISHED docker:default
=> [internal] load build definition from Dockerfile 0.1s
=> => transferring dockerfile: 245B 0.0s
=> [internal] load metadata for docker.io/library/nginx:latest 0.1s
=> [internal] load .dockerignore 0.1s
=> => transferring context: 2B 0.0s
=> [internal] load build context 0.1s
=> => transferring context: 224B 0.0s
=> [1/2] FROM docker.io/library/nginx:latest@sha256:c881927c4077710ac4b1da63b83aa163937fb4745795 0.3s
=> => resolve docker.io/library/nginx:latest@sha256:c881927c4077710ac4b1da63b83aa163937fb4745795 0.1s
=> [2/2] COPY index.html /usr/share/nginx/html/index.html 0.0s
=> exporting to image 0.4s
=> => exporting layers 0.1s
=> => exporting manifest sha256:883969f8c2a22d77cb2de1cc914aa7bc6b62cba42b067c996a73c943863fe262 0.0s
=> => exporting config sha256:335dbbe665e3709a52ce39a9cf98267262a03ece08293a6d88e365c8f62d32e2 0.0s
=> => exporting attestation manifest sha256:7e086f7fb346ec89808e4dc9493b7a1a0ef6ecb148fed2fda32b 0.0s
=> => exporting manifest list sha256:bce71047ed5122a654b42574c78a6b31c02f2ebed9555f9e945a877337f 0.0s
=> => naming to docker.io/library/docker-practical4:latest 0.0s
=> => unpacking to docker.io/library/docker-practical4:latest 0.1s
geomathew@GeoMathew:/mnt/c/WINDOWS/system32/docker-web-app$ docker images
Info → U In Use
IMAGE ID DISK USAGE CONTENT SIZE EXTRA
docker-practical4:latest bce71047ed51 237MB 62.9MB
docker-practical:latest 723306ab4147 92.2MB 25.9MB U
geomathew21/docker-practical:latest 723306ab4147 92.2MB 25.9MB U
hello-world:latest 05813aedc15f 25.9kB 9.52kB U
nginx:latest c881927c4077 240MB 65.7MB
ubuntu:latest cd1dba651b30 119MB 31.7MB U
geomathew@GeoMathew:/mnt/c/WINDOWS/system32/docker-web-app$ docker run -d -p 8082:80 --name practical4-container docker-practical4
e2f48a82f470e295babcbc538cb6c1c8d23691a60e73c9402d020895f5f7b053
geomathew@GeoMathew:/mnt/c/WINDOWS/system32/docker-web-app$
Upcoming Earnings
Search
ENG IN
19:43
02-02-2026

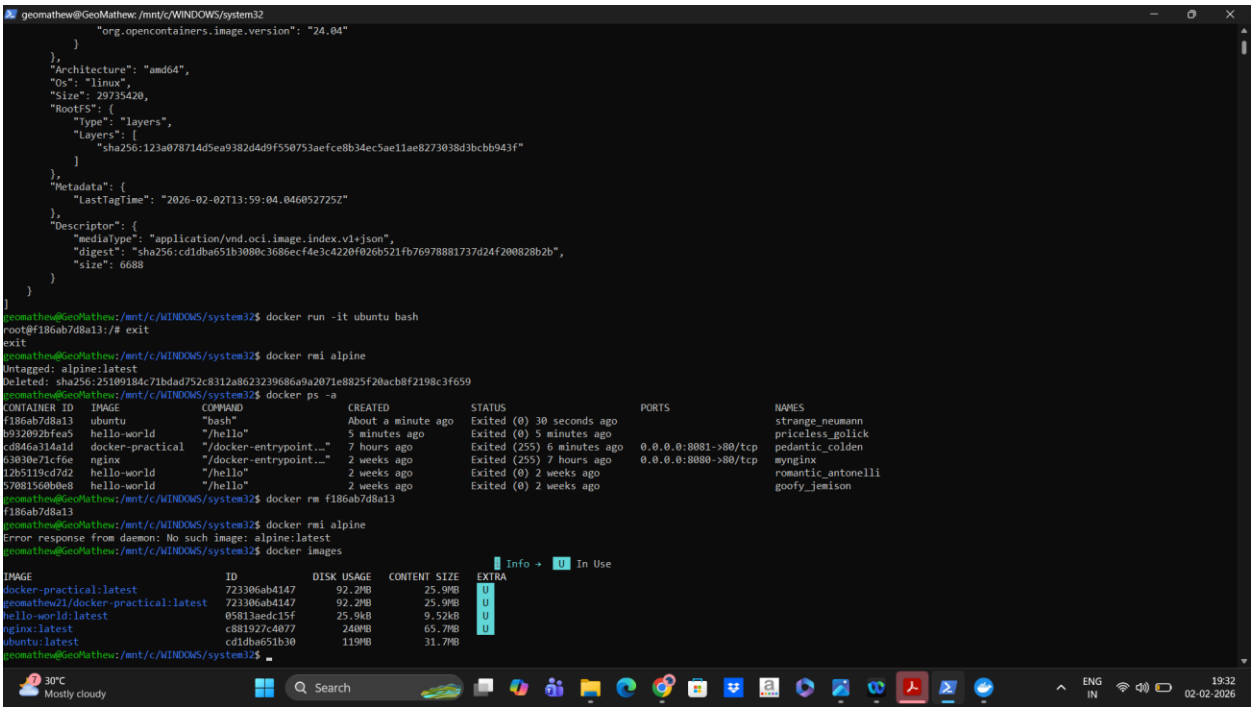

geomathew@GeoMathew: /mnt/c/WINDOWS/system32
geomathew@GeoMathew:/mnt/c/WINDOWS/system32$ docker rm mynginx
mynginx
geomathew@GeoMathew:/mnt/c/WINDOWS/system32$ docker run -d -p 8080:80 --name mynginx nginx
9ac9ba40ae18c7e215f17f32d5e225f4969f620939ff9ab2959fe97b4baaa60a
docker: Error response from daemon: ports are not available: exposing port TCP 0.0.0.0:8080 -> 127.0.0.1:0: /forwards/expose returned unexpected status: 500
Run 'docker run --help' for more information
geomathew@GeoMathew:/mnt/c/WINDOWS/system32$ docker run -d -p 8081:80 --name mynginx nginx
docker: Error response from daemon: Conflict. The container name "/mynginx" is already in use by container "9ac9ba40ae18c7e215f17f32d5e225f4969f620939ff9ab2959fe97b4baaa60a". You have to remove (or rename) that container to be able to reuse that name.
Run 'docker run --help' for more information
geomathew@GeoMathew:/mnt/c/WINDOWS/system32$ docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
geomathew@GeoMathew:/mnt/c/WINDOWS/system32$ docker rm 9ac9ba40ae18c7e215f17f32d5e225f4969f620939ff9ab2959fe97b4baaa60a
9ac9ba40ae18c7e215f17f32d5e225f4969f620939ff9ab2959fe97b4baaa60a
geomathew@GeoMathew:/mnt/c/WINDOWS/system32$ docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
362970910ad3 ubuntu "bash" 6 minutes ago Exited (0) 3 minutes ago dreamy_jennings
b932092bfea5 hello-world "/hello" 14 minutes ago Exited (0) 14 minutes ago priceless_golick
cd846a314a1d docker-practical "/docker-entrypoint.…" 7 hours ago Exited (255) 15 minutes ago 0.0.0.0:8081->80/tcp pedantic_colden
12b5119cd7d2 hello-world "/hello" 2 weeks ago Exited (0) 2 weeks ago romantic_antonelli
57081560b0e8 hello-world "/hello" 2 weeks ago Exited (0) 2 weeks ago goofy_jemison
geomathew@GeoMathew:/mnt/c/WINDOWS/system32$ docker run -d -p 8081:80 --name mynginx nginx
46b90f47ee27d0503f2174f5aeae45d98ce302c7d1daeaa45bd6f2ae32fde2ff
geomathew@GeoMathew:/mnt/c/WINDOWS/system32$ docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
46b90f47ee27 nginx "/docker-entrypoint.…" 7 seconds ago Up 6 seconds 0.0.0.0:8081->80/tcp, [::]:8081->80/tcp mynginx
geomathew@GeoMathew:/mnt/c/WINDOWS/system32$ docker stop mynginx
mynginx
geomathew@GeoMathew:/mnt/c/WINDOWS/system32$ docker restart mynginx
mynginx
geomathew@GeoMathew:/mnt/c/WINDOWS/system32$ docker stop mynginx
mynginx
geomathew@GeoMathew:/mnt/c/WINDOWS/system32$ docker rm mynginx
mynginx
geomathew@GeoMathew:/mnt/c/WINDOWS/system32$ docker rm -f mynginx
Error response from daemon: No such container: mynginx
geomathew@GeoMathew:/mnt/c/WINDOWS/system32$
30°C Mostly cloudy
Search
ENG IN
19:41
02-02-2026

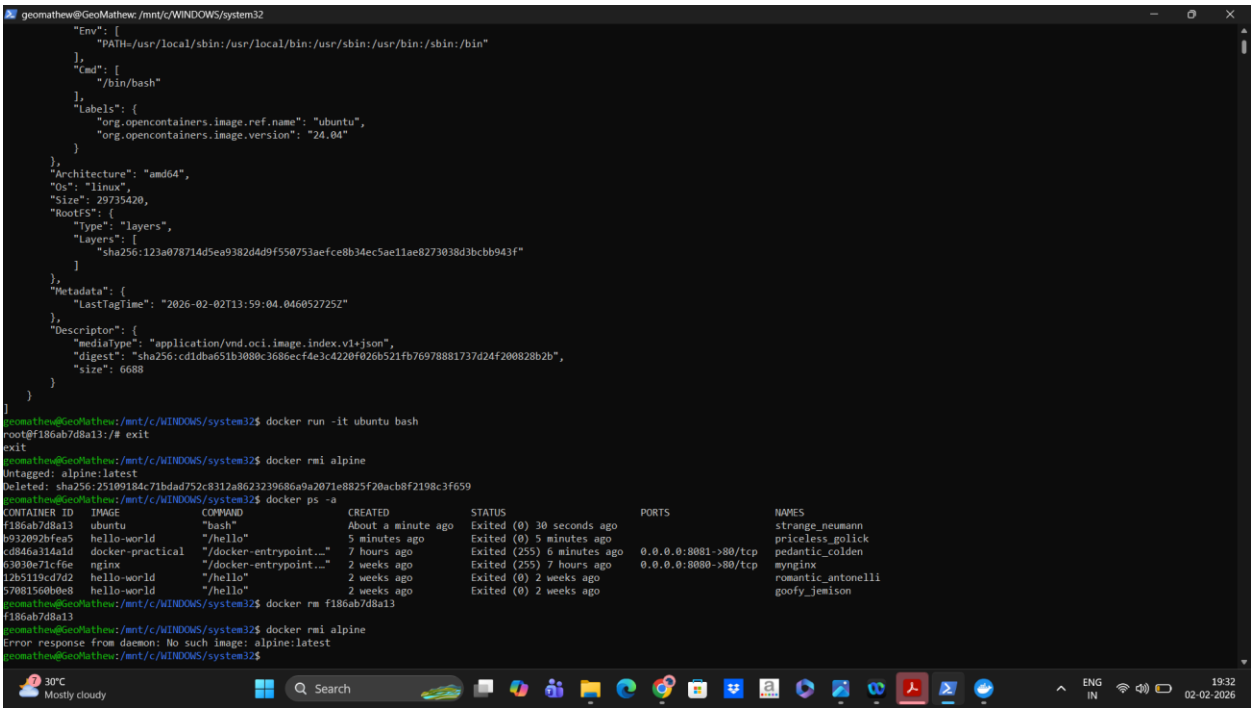

root@8aa23bbe6419: /
PS C:\WINDOWS\system32> docker network ls
NETWORK ID NAME DRIVER SCOPE
8e04a842bb4f bridge bridge local
79b81f45f53d host host local
f2c3aba28428 none null local
PS C:\WINDOWS\system32> docker network create mybridge
42c0c0437372a82db5fd0d9dc69e791ff99594f6b9c60eac5859bb31a3232c33
PS C:\WINDOWS\system32> docker network ls
NETWORK ID NAME DRIVER SCOPE
8e04a842bb4f bridge bridge local
79b81f45f53d host host local
42c0c0437372 mybridge bridge local
f2c3aba28428 none null local
PS C:\WINDOWS\system32> docker run -it --name net-container1 --network mybridge ubuntu bash
root@8aa23bbe6419:/# apt update
Get:1 http://archive.ubuntu.com/ubuntu noble InRelease [256 kB]
Get:2 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:3 http://archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:4 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Packages [34.8 kB]
Get:5 http://archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:6 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Packages [3008 kB]
Get:7 http://archive.ubuntu.com/ubuntu noble/multiverse amd64 Packages [331 kB]
Get:8 http://archive.ubuntu.com/ubuntu noble/main amd64 Packages [1808 kB]
Get:9 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Packages [1196 kB]
Get:10 http://archive.ubuntu.com/ubuntu noble/restricted amd64 Packages [117 kB]
Get:11 http://archive.ubuntu.com/ubuntu noble/universe amd64 Packages [19.3 MB]
Get:12 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [1776 kB]
Get:13 http://archive.ubuntu.com/ubuntu noble-updates/universe amd64 Packages [1975 kB]
Get:14 http://archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [2171 kB]
Get:15 http://archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Packages [3219 kB]
Get:16 http://archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Packages [38.1 kB]
Get:17 http://archive.ubuntu.com/ubuntu noble-backports/universe amd64 Packages [34.6 kB]
Get:18 http://archive.ubuntu.com/ubuntu noble-backports/main amd64 Packages [49.5 kB]
Fetched 35.7 MB in 12s (3036 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
1 package can be upgraded. Run 'apt list --upgradable' to see it.
root@8aa23bbe6419:/# apt install iputils-ping -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
libcap2-bin libpam-cap
The following NEW packages will be installed:
iputils-ping libcap2-bin libpam-cap
0 upgraded, 3 newly installed, 0 to remove and 1 not upgraded.
Need to get 91.2 kB of archives.
After this operation, 322 kB of additional disk space will be used.
29°C
Mostly cloudy
Search
ENG
IN
19:49
02-02-2026

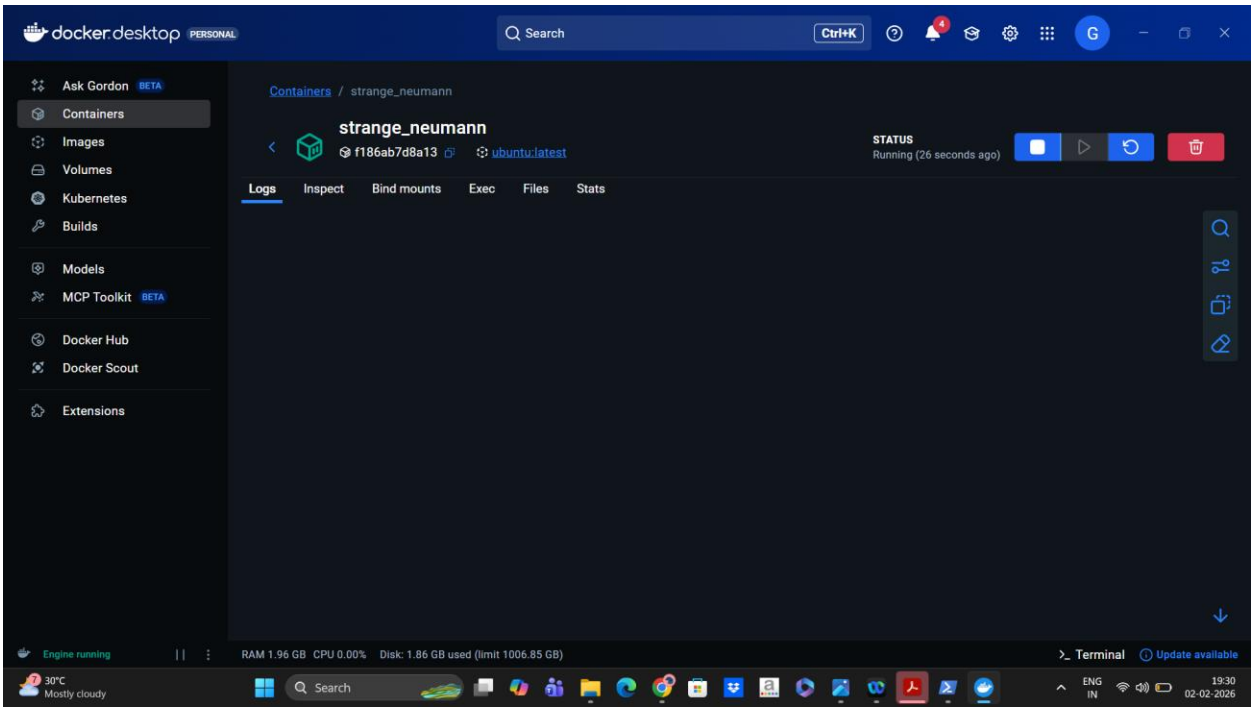

geomathew@GeoMathew: /mnt/c/WINDOWS/system32/docker-web-app
geomathew@GeoMathew:/mnt/c/WINDOWS/system32/docker-web-app$ docker volume create mydata
mydata
geomathew@GeoMathew:/mnt/c/WINDOWS/system32/docker-web-app$ docker volume ls
DRIVER VOLUME NAME
local mydata
geomathew@GeoMathew:/mnt/c/WINDOWS/system32/docker-web-app$ docker run -it --name vol-container1 -v mydata:/data ubuntu bash
root@4d33ade0d1cc:/# cd /data
root@4d33ade0d1cc:/data# echo "Docker volume persistence test" > file1.txt
root@4d33ade0d1cc:/data# ls
file1.txt
root@4d33ade0d1cc:/data# cat file1.txt
Docker volume persistence test
root@4d33ade0d1cc:/data# exit
exit
geomathew@GeoMathew:/mnt/c/WINDOWS/system32/docker-web-app$ docker rm vol-container1
vol-container1
geomathew@GeoMathew:/mnt/c/WINDOWS/system32/docker-web-app$ docker run -it --name vol-container2 -v mydata:/data ubuntu bash
root@a6edf88eb883:/# cd /data
root@a6edf88eb883:/data# ls
file1.txt
root@a6edf88eb883:/data# cat file1.txt
Docker volume persistence test
root@a6edf88eb883:/data# docker run -it --name vol-container3 -v mydata:/data ubuntu bash
bash: docker: command not found
root@a6edf88eb883:/data# exit
exit
geomathew@GeoMathew:/mnt/c/WINDOWS/system32/docker-web-app$ docker run -it --name vol-container3 -v mydata:/data ubuntu bash
root@bcb97ab903cc:/# ls /data
file1.txt
root@bcb97ab903cc:/# exit
exit
geomathew@GeoMathew:/mnt/c/WINDOWS/system32/docker-web-app$ docker rm vol-container2 vol-container3
vol-container2
vol-container3
geomathew@GeoMathew:/mnt/c/WINDOWS/system32/docker-web-app$ docker volume inspect mydata
[
{
"CreatedAt": "2026-02-02T14:15:12Z",
"Driver": "local",
"Labels": null,
"Mountpoint": "/var/lib/docker/volumes/mydata/_data",
"Name": "mydata",
"Options": null,
"Scope": "local"
}
]
geomathew@GeoMathew:/mnt/c/WINDOWS/system32/docker-web-app$
29°C
Mostly cloudy
Search
ENG
IN
19:47
02-02-2026


root@362970910ad3: /
geomathew@GeoMathew:/mnt/c/WINDOWS/system32$ docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
geomathew@GeoMathew:/mnt/c/WINDOWS/system32$ docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
362970910ad3 ubuntu "bash" 43 seconds ago Exited (0) 22 seconds ago dreamy_jennings
b932092bfea5 hello-world "/hello" 8 minutes ago Exited (0) 8 minutes ago priceless_golick
cd846a314a1d docker-practical "/docker-entrypoint.…" 7 hours ago Exited (255) 9 minutes ago 0.0.0.0:8081->80/tcp pedantic_colden
63030e71cf6e nginx "/docker-entrypoint.…" 2 weeks ago Exited (255) 7 hours ago 0.0.0.0:8080->80/tcp mynginx
12b5119cd7d2 hello-world "/hello" 2 weeks ago Exited (0) 2 weeks ago romantic_antonelli
57081560b0e8 hello-world "/hello" 2 weeks ago Exited (0) 2 weeks ago goofy_jemison
geomathew@GeoMathew:/mnt/c/WINDOWS/system32$ docker start 362970910ad3
362970910ad3
geomathew@GeoMathew:/mnt/c/WINDOWS/system32$ docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
362970910ad3 ubuntu "bash" About a minute ago Up 6 seconds dreamy_jennings
geomathew@GeoMathew:/mnt/c/WINDOWS/system32$ docker attach 362970910ad3
root@362970910ad3:/# whoami
root
root@362970910ad3:/# pwd
/
root@362970910ad3:/# cat /etc/os-release
PRETTY_NAME="Ubuntu 24.04.3 LTS"
NAME="Ubuntu"
VERSION_ID="24.04"
VERSION="24.04.3 LTS (Noble Numbat)"
VERSION_CODENAME=noble
ID=ubuntu
ID_LIKE=debian
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
UBUNTU_CODENAME=noble
LOGO=ubuntu-logo
root@362970910ad3:/#
30°C
Mostly cloudy
Search
ENG
IN
19:35
02-02-2026


geomathew@GeoMathew: /mnt/c/WINDOWS/system32
geomathew@GeoMathew:/mnt/c/WINDOWS/system32$ docker run -it ubuntu bash
root@362970910ad3:/# whoami
root
root@362970910ad3:/# pwd
/
root@362970910ad3:/# ls
bin boot dev etc home lib lib64 media mnt opt proc root run sbin srv sys tmp usr var
root@362970910ad3:/# exit
exit
geomathew@GeoMathew:/mnt/c/WINDOWS/system32$
30°C
Mostly cloudy
Search
ENG
IN
19:33
02-02-2026


geomathew@GeoMathew: /mnt/c/WINDOWS/system32
"org.opencontainers.image.version": "24.04"
}
},
"Architecture": "amd64",
"Os": "linux",
"Size": 29735420,
"RootFS": {
"Type": "layers",
"Layers": [
"sha256:123a078714d5ea9382d4d9f550753aefce8b34ec5ae11ae8273038d3bcbb943f"
]
},
"Metadata": {
"LastTagTime": "2026-02-02T13:59:04.046052725Z"
},
"Descriptor": {
"mediaType": "application/vnd.oci.image.index.v1+json",
"digest": "sha256:cd1dba651b3080c3686ecf4e3c4220f026b521fb76978881737d24f200828b2b",
"size": 6688
}
}
]
geomathew@GeoMathew:/mnt/c/WINDOWS/system32$ docker run -it ubuntu bash
root@f186ab7d8a13:/# exit
exit
geomathew@GeoMathew:/mnt/c/WINDOWS/system32$ docker rmi alpine
Untagged: alpine:latest
Deleted: sha256:25109184c71bdad752c8312a8623239686a9a2071e8825f20acb8f2198c3f659
geomathew@GeoMathew:/mnt/c/WINDOWS/system32$ docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
f186ab7d8a13 ubuntu "bash" About a minute ago Exited (0) 30 seconds ago strange_neumann
b932092bfea5 hello-world "/hello" 5 minutes ago Exited (0) 5 minutes ago priceless_golick
cd846a314a1d docker-practical "/docker-entrypoint.…" 7 hours ago Exited (255) 6 minutes ago 0.0.0.0:8081->80/tcp pedantic_colden
63030e71cf6e nginx "/docker-entrypoint.…" 2 weeks ago Exited (255) 7 hours ago 0.0.0.0:8080->80/tcp mynginx
12b5119cd7d2 hello-world "/hello" 2 weeks ago Exited (0) 2 weeks ago romantic_antonelli
57081560b0e8 hello-world "/hello" 2 weeks ago Exited (0) 2 weeks ago goofy_jemison
geomathew@GeoMathew:/mnt/c/WINDOWS/system32$ docker rm f186ab7d8a13
f186ab7d8a13
geomathew@GeoMathew:/mnt/c/WINDOWS/system32$ docker rmi alpine
Error response from daemon: No such image: alpine:latest
geomathew@GeoMathew:/mnt/c/WINDOWS/system32$ docker images
Info → U In Use
IMAGE ID DISK USAGE CONTENT SIZE EXTRA
docker-practical:latest 723306ab4147 92.2MB 25.9MB U
geomathew21/docker-practical:latest 723306ab4147 92.2MB 25.9MB U
hello-world:latest 05813aedc15f 25.9kB 9.52kB U
nginx:latest c881927c4077 240MB 65.7MB U
ubuntu:latest cd1dba651b30 119MB 31.7MB
geomathew@GeoMathew:/mnt/c/WINDOWS/system32$
30°C
Mostly cloudy
Search
ENG
IN
19:32
02-02-2026


geomathew@GeoMathew: /mnt/c/WINDOWS/system32
"Env": [
"PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"
],
"Cmd": [
"/bin/bash"
],
"Labels": {
"org.opencontainers.image.ref.name": "ubuntu",
"org.opencontainers.image.version": "24.04"
}
},
"Architecture": "amd64",
"Os": "linux",
"Size": 29735420,
"RootFS": {
"Type": "layers",
"Layers": [
"sha256:123a078714d5ea9382d4d9f550753aefce8b34ec5ae11ae8273038d3bcbb943f"
]
},
"Metadata": {
"LastTagTime": "2026-02-02T13:59:04.046052725Z"
},
"Descriptor": {
"mediaType": "application/vnd.oci.image.index.v1+json",
"digest": "sha256:cd1dba651b3080c3686ecf4e3c4220f026b521fb76978881737d24f200828b2b",
"size": 6688
}
}
]
geomathew@GeoMathew:/mnt/c/WINDOWS/system32$ docker run -it ubuntu bash
root@f186ab7d8a13:/# exit
exit
geomathew@GeoMathew:/mnt/c/WINDOWS/system32$ docker rmi alpine
Untagged: alpine:latest
Deleted: sha256:25109184c71bdad752c8312a8623239686a9a2071e8825f20acb8f2198c3f659
geomathew@GeoMathew:/mnt/c/WINDOWS/system32$ docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
f186ab7d8a13 ubuntu "bash" About a minute ago Exited (0) 30 seconds ago strange_neumann
b932092bfea5 hello-world "/hello" 5 minutes ago Exited (0) 5 minutes ago priceless_golick
cd846a314a1d docker-practical "/docker-entrypoint.…" 7 hours ago Exited (255) 6 minutes ago 0.0.0.0:8081->80/tcp pedantic_colden
63030e71cf6e nginx "/docker-entrypoint.…" 2 weeks ago Exited (255) 7 hours ago 0.0.0.0:8080->80/tcp mynginx
12b5119cd7d2 hello-world "/hello" 2 weeks ago Exited (0) 2 weeks ago romantic_antonelli
57081560b0e8 hello-world "/hello" 2 weeks ago Exited (0) 2 weeks ago goofy_jemison
geomathew@GeoMathew:/mnt/c/WINDOWS/system32$ docker rm f186ab7d8a13
f186ab7d8a13
geomathew@GeoMathew:/mnt/c/WINDOWS/system32$ docker rmi alpine
Error response from daemon: No such image: alpine:latest
geomathew@GeoMathew:/mnt/c/WINDOWS/system32$
30°C
Mostly cloudy
Search
ENG
IN
19:32
02-02-2026



Real-World Use of Docker

Docker is used to **run applications the same way everywhere**.

- Docker packs the application and everything it needs into one container
- The app works the same on a developer's laptop, test server, and production server
- It removes the problem of "it works on my system but not on the server"
- Applications are easy to start, stop, and update using Docker
- Many small services can run separately using Docker containers
- Docker is fast and uses less system resources
- It is commonly used in CI/CD pipelines for building and testing apps

Simple example:
A developer creates an app in Docker. The same Docker container is used for testing and then deployed to the server without changes.

In short, **Docker makes applications easy to move, run, and manage**

Personal Understanding and Learning Outcomes

Through learning Docker, I understood how applications can be packaged and run in a simple and reliable way. Earlier, setting up applications on different systems took a lot of time, but Docker showed me how this process can be made easy and consistent.

From this learning, I understood:

- How Docker helps applications run the same on all systems
- Why containers are better than manual installation
- How Docker reduces setup time and configuration problems
- The importance of consistency between development, testing, and production
- How Docker is used in real DevOps pipelines for building and deploying applications
- How containers make application deployment faster and more reliable

Overall, this learning helped me understand how modern DevOps teams use Docker to simplify application delivery. Docker improved my confidence in handling deployments and strengthened my understanding of container-based development.

Conclusion

This project helped me understand how DevOps works in real software companies. By working hands-on with tools like Git, CI/CD pipelines, Jenkins, Docker, Terraform, and AWS, I learned how software is created, tested, deployed, and maintained in an efficient way.

I understood why automation is so important. Automation reduces manual work, avoids mistakes, and helps teams deliver software faster. By using CI/CD pipelines, I learned how even small code changes can be tested and deployed quickly without affecting the stability of the application.

Tools like Jenkins and Docker helped me see how the same application can run smoothly in development, testing, and production environments. This made software delivery more reliable and predictable.

Learning Terraform showed me how cloud infrastructure can be created and managed using code instead of manual setup. This makes infrastructure easy to scale and repeat. Working with AWS gave me a basic understanding of real cloud platforms used by companies today.

Overall, this project improved my technical skills, problem-solving ability, and confidence in DevOps workflows. It also helped me understand how developers and operations teams work together in real projects. This learning experience has given me a strong base to grow further in cloud computing and DevOps engineering.