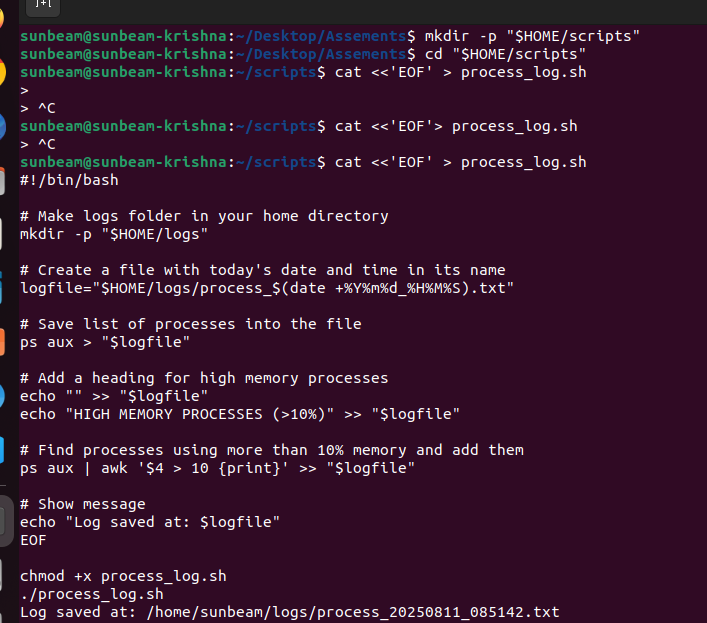
# Basic-Level Case Study Practical Questions

## A. Shell Scripting (Basics)

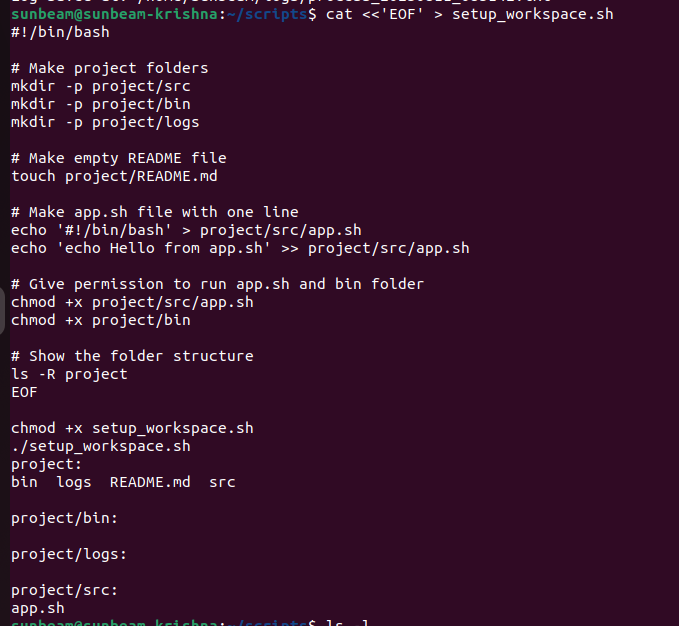
### A1) Process log + high memory check

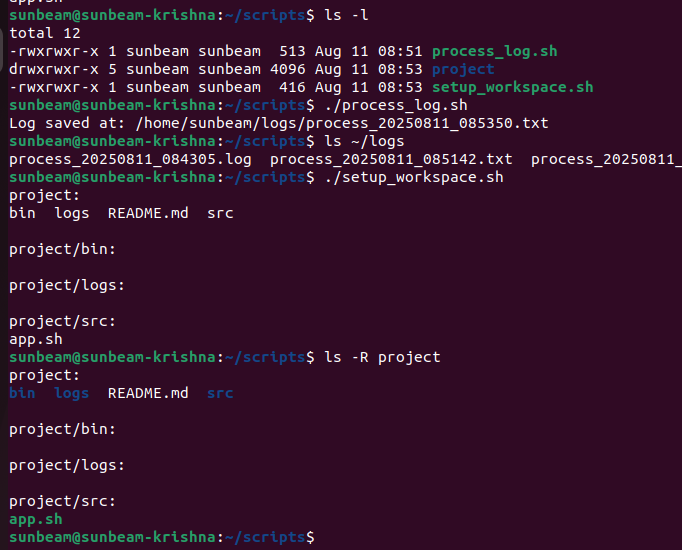
Scenario: Ops wants a snapshot of all running processes every time the script runs and a quick flag for any “memory hogs”.  
Task:  
1) Lists all processes and saves them to a dated log file under ~/logs/.  
2) Identifies any process using > 10% memory and appends those lines under a heading “HIGH MEMORY” in the same log.  
3) Creates the logs directory if it doesn’t exist.  
Deliverables: Script file, one sample log.  
Success criteria: Log exists with timestamped filename; “HIGH MEMORY” section appears when applicable.



### A2) Create files/dirs with a script

Scenario: A project bootstrap script should lay down a simple workspace.  
Task:  
1) Creates a directory tree: project/{src,bin,logs}.  
2) Creates an empty file README.md and a file src/app.sh.  
3) Adds execute permission only to bin and src/app.sh.  
Deliverables: Script + screenshot of resulting tree.  
Success criteria: Tree structure and permissions match requirement.

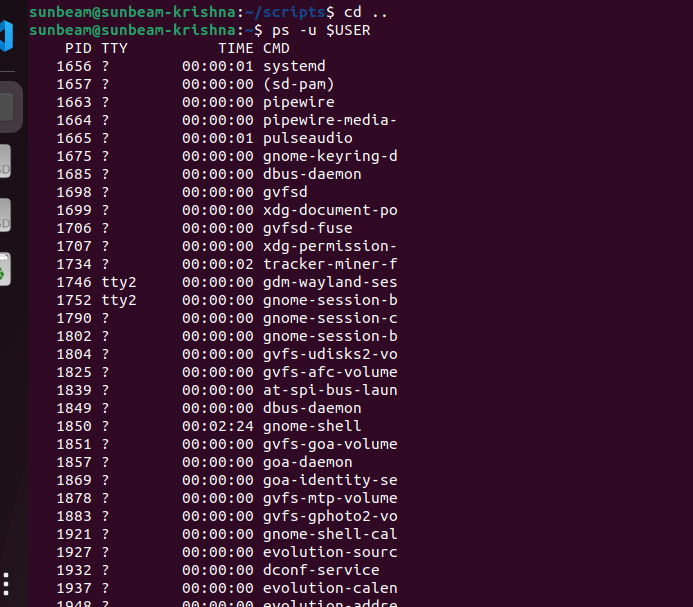


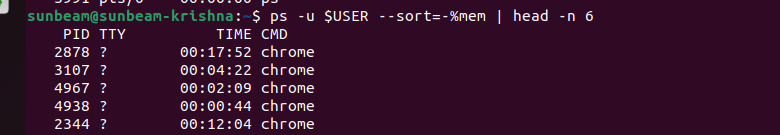


## B. Linux Basics

### B1) Checking processes

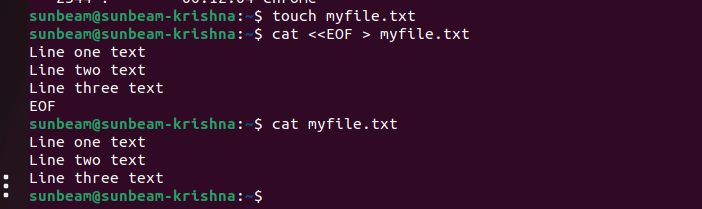
Task: List all processes for the current user, show top 5 memory-consuming processes.  
Deliverables: Commands used + screenshots.  
Success criteria: Correct process lists visible.





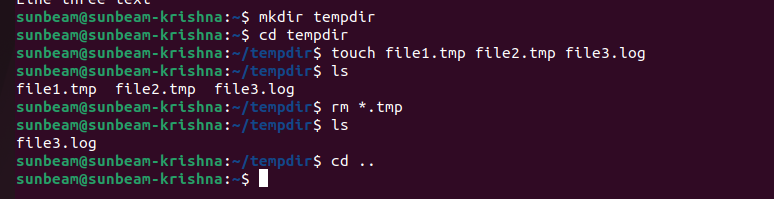
### B2) File management (cat, touch, vi, nano)

Task: Create a file, add 3 lines using nano or vi, display contents with cat.  
Deliverables: Commands + file content screenshot.  
Success criteria: File exists with expected content.



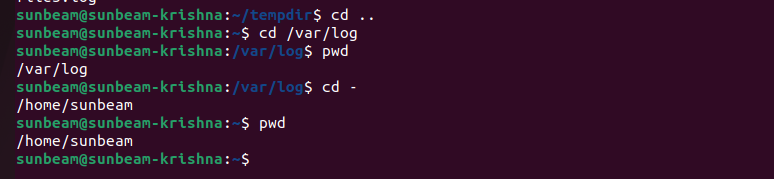
### B3) Remove command

Task: Create a temp directory with 3 files inside, then remove only files ending with .tmp.  
Deliverables: Commands + before/after listing.  
Success criteria: Only .tmp files removed.



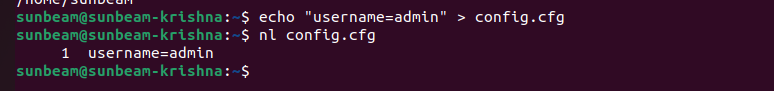
### B4) cd commandsht

Task: Navigate from home to /var/log, then back using a single command; print current directory each time.  
Deliverables: Commands + outputs.  
Success criteria: Correct paths shown.



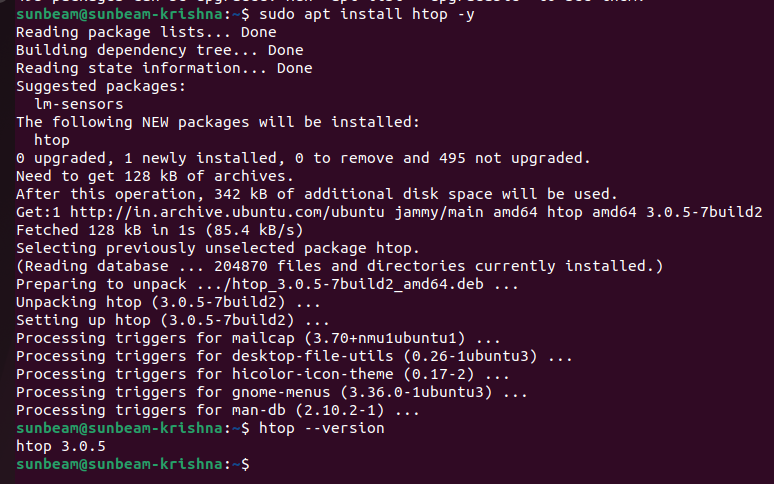
### B5) File editing

Task: Open a config file (create one), add a key=value pair, save, and show the line number where it exists.  
Deliverables: Editor steps + proof via command output.  
Success criteria: Key=value present with line number.



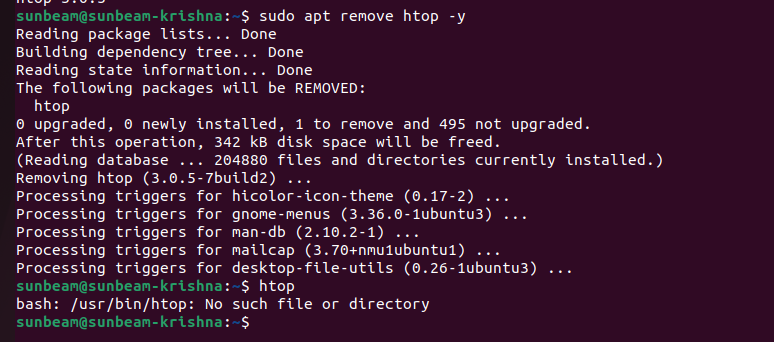
### B6) Install package

Task: Install a basic CLI tool (e.g., htop).  
Deliverables: Command + version output.  
Success criteria: Tool runs and shows version.



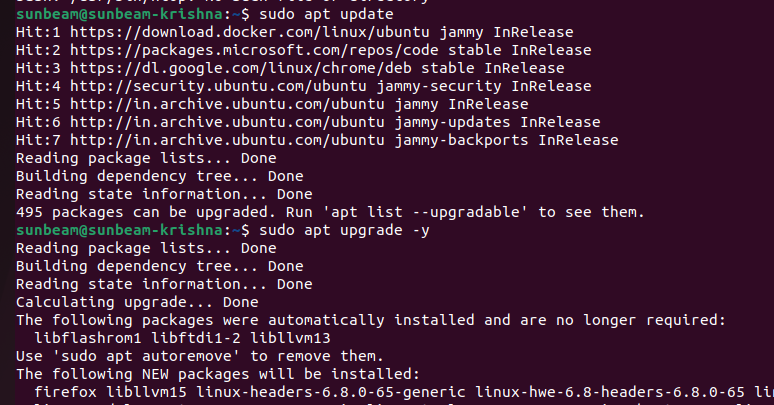
### B7) Remove package

Task: Uninstall the same package without removing unrelated deps.  
Deliverables: Commands + verification it’s gone.  
Success criteria: Command not found / removed.



### B8) Update

Task: Update package lists and upgrade only security patches (or minimal upgrade).  
Deliverables: Commands + summary output.  
Success criteria: Update/upgrade completes without errors.



## C. Jira

### C1) Create & manage project + progress

Task: Create a Kanban or Scrum project, add 4 user stories, assign to 2 users, move across statuses, generate progress report.  
Deliverables: Project screenshot, issues list, report/export.  
Success criteria: 4 stories exist, assignments correct, progress visible.

### C2) Automation rule

Task: Create an automation rule triggered when an issue is moved to Ready for QA, assigning QA lead or sending email.  
Deliverables: Rule config screenshots + a test run proof.  
Success criteria: Rule fires as expected.

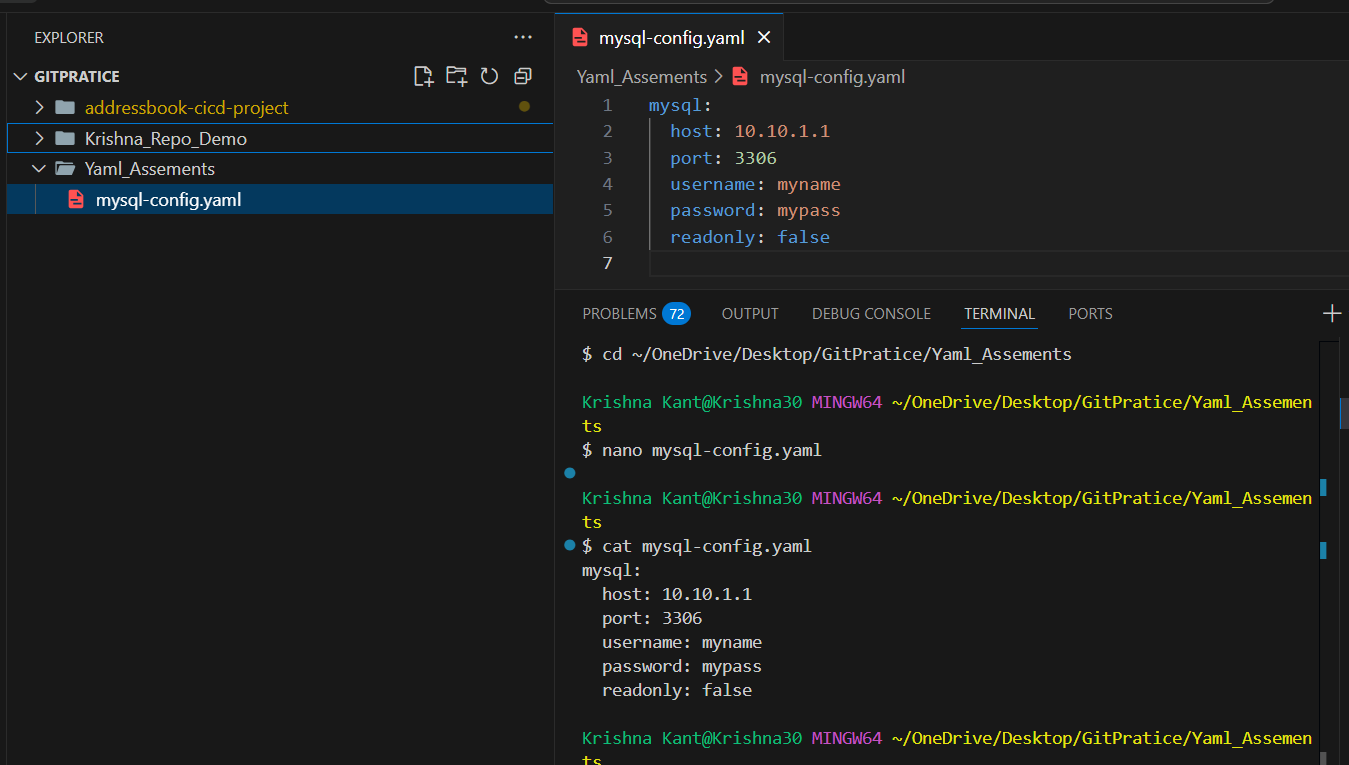
### C3) Kanban board

Task: Create a Kanban board with columns: To Do, In Progress, In Review, Done.  
Deliverables: Board screenshot with at least 3 issues across columns.  
Success criteria: Columns configured; issues visible.

## D. YAML

### D1) MySQL service descriptor

Task: Write YAML describing MySQL server at 10.10.1.1:3306, username=myname, password=mypass, readonly=false.  
Deliverables: mysql-config.yaml file.  
Success criteria: YAML is valid and human-readable.



## E. Python Basics

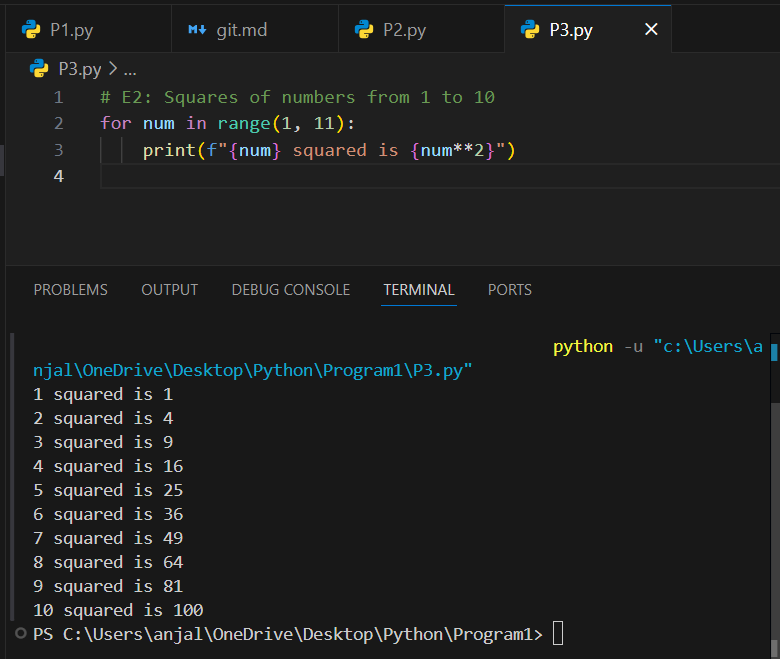
### E1) if / elif / else

Task: Script to take score (0–100) and print grade A/B/C/D/F.  
Deliverables: Script + sample runs.  
Success criteria: Correct grades printed.



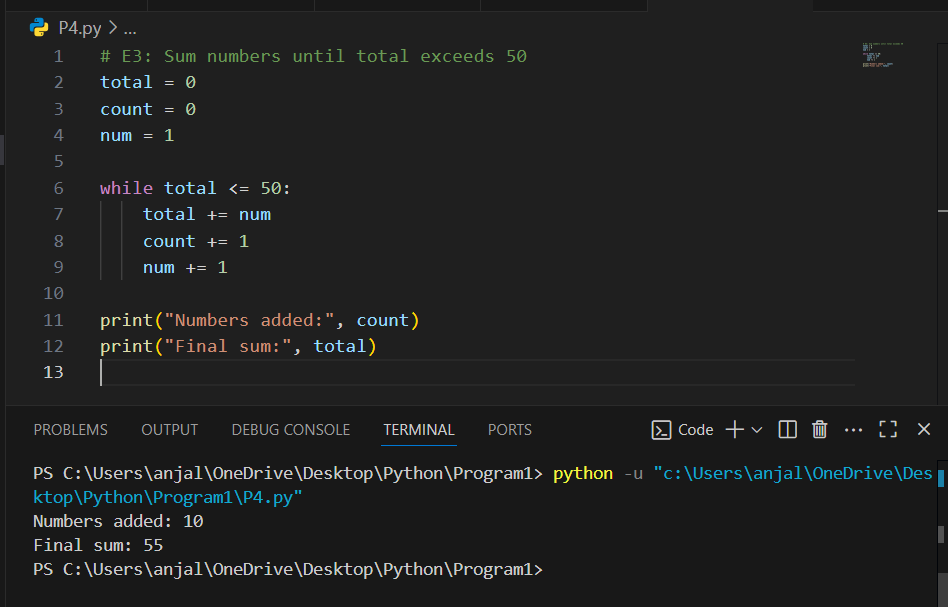
### E2) for loop

Task: Print squares of numbers 1 to 10.  
Deliverables: Script + output.  
Success criteria: Correct sequence printed.



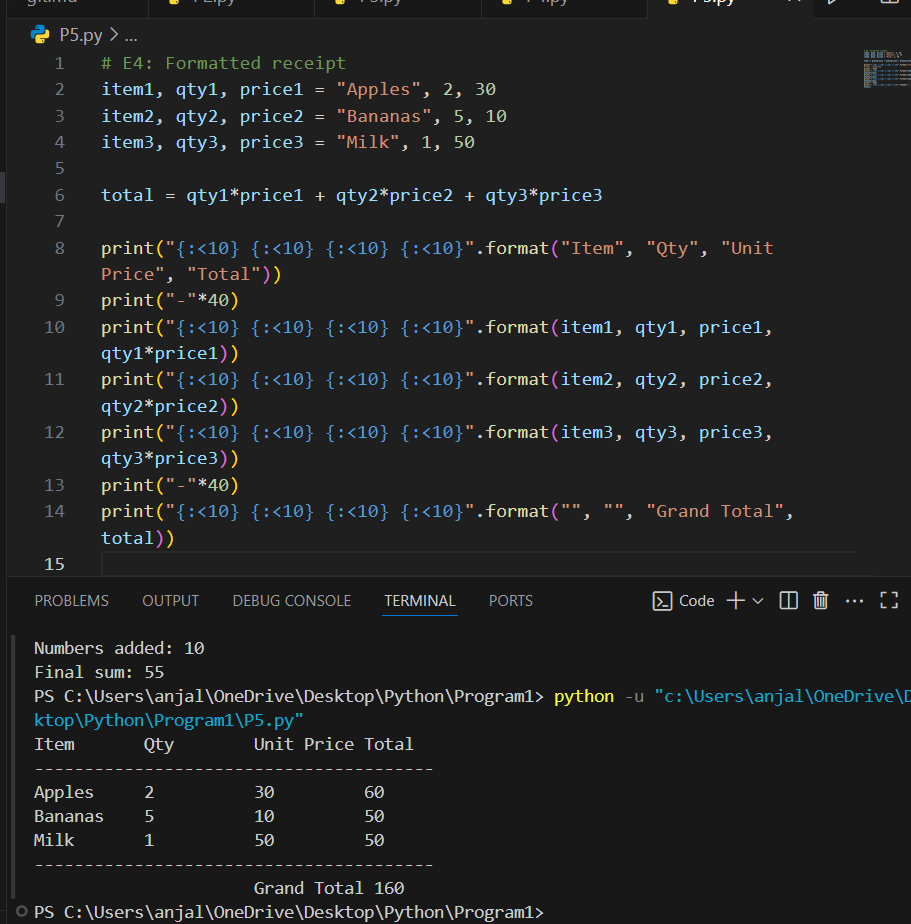
### E3) while loop

Task: Sum numbers until total exceeds 50; print how many numbers were added.  
Deliverables: Script + output.  
Success criteria: Shows count and sum.



### E4) print formatting

Task: Print formatted receipt with 3 items, quantities, unit prices, total.  
Deliverables: Script + output.  
Success criteria: Aligned columns.

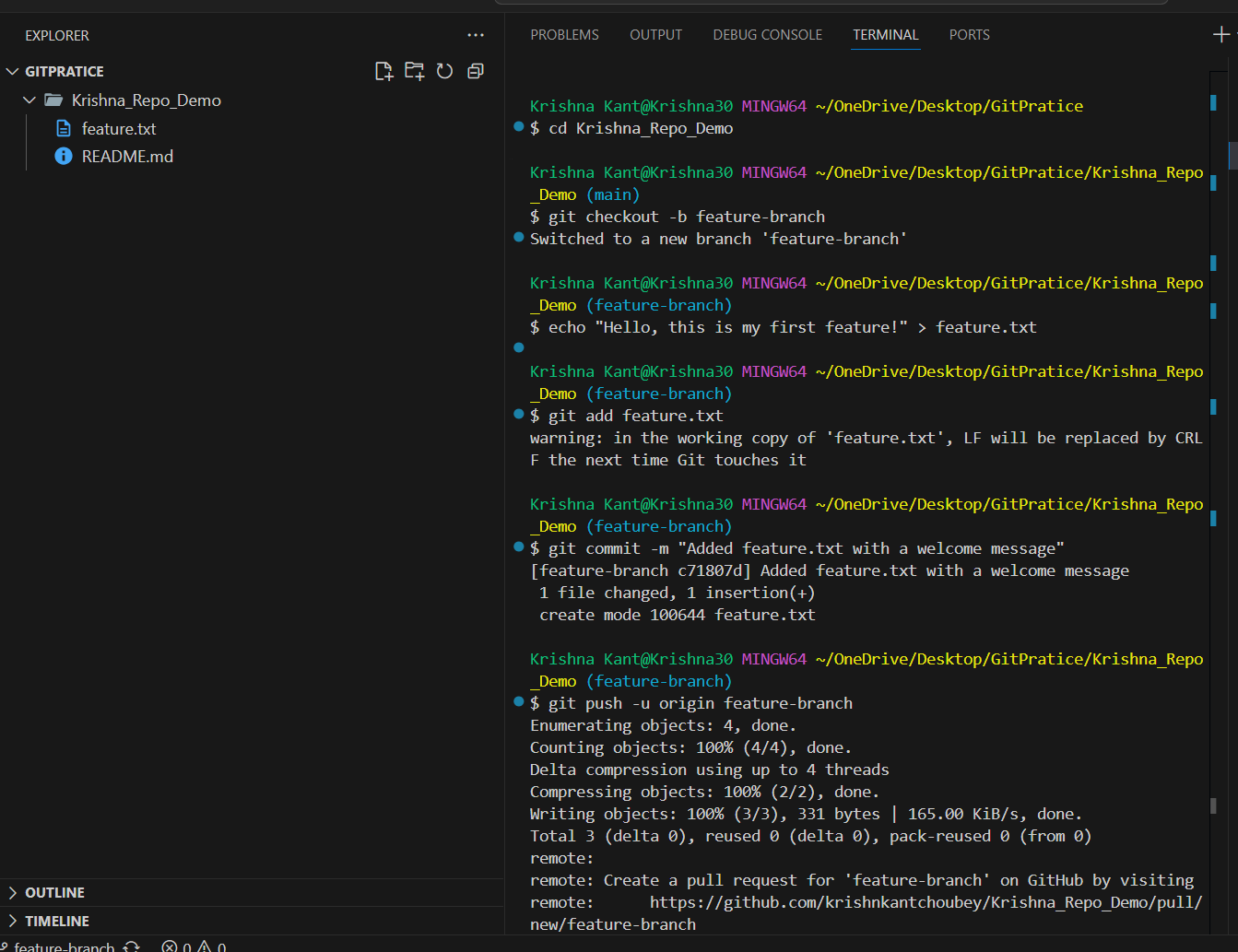


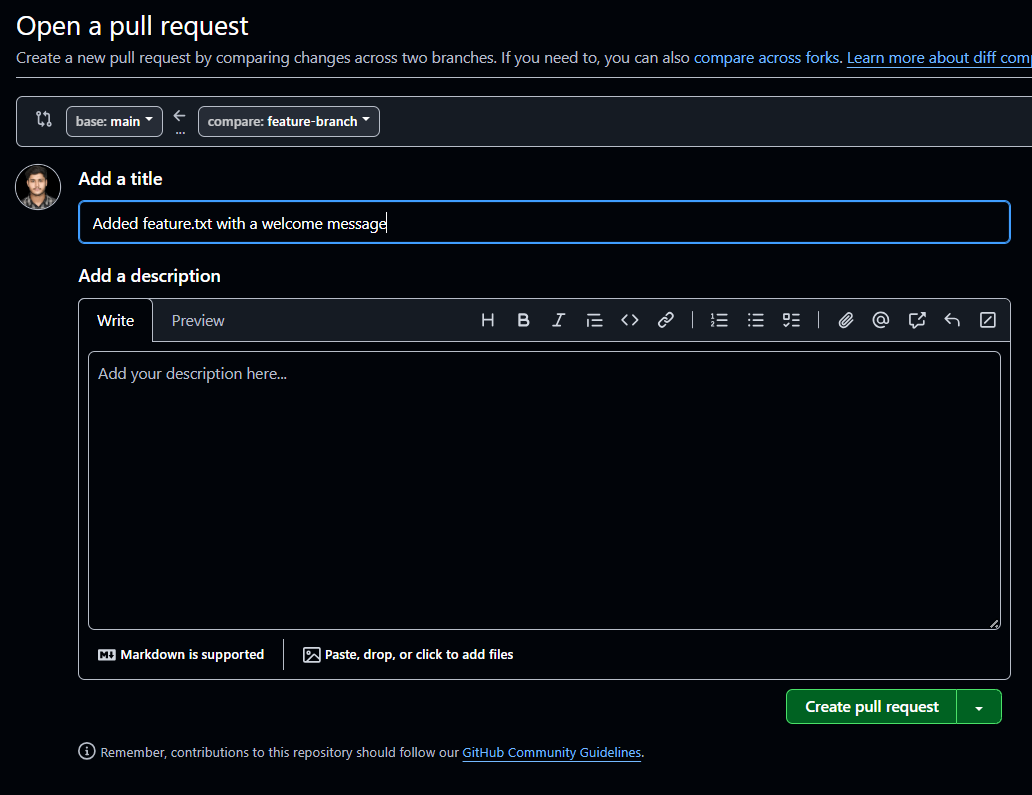
## F. Git & GitHub

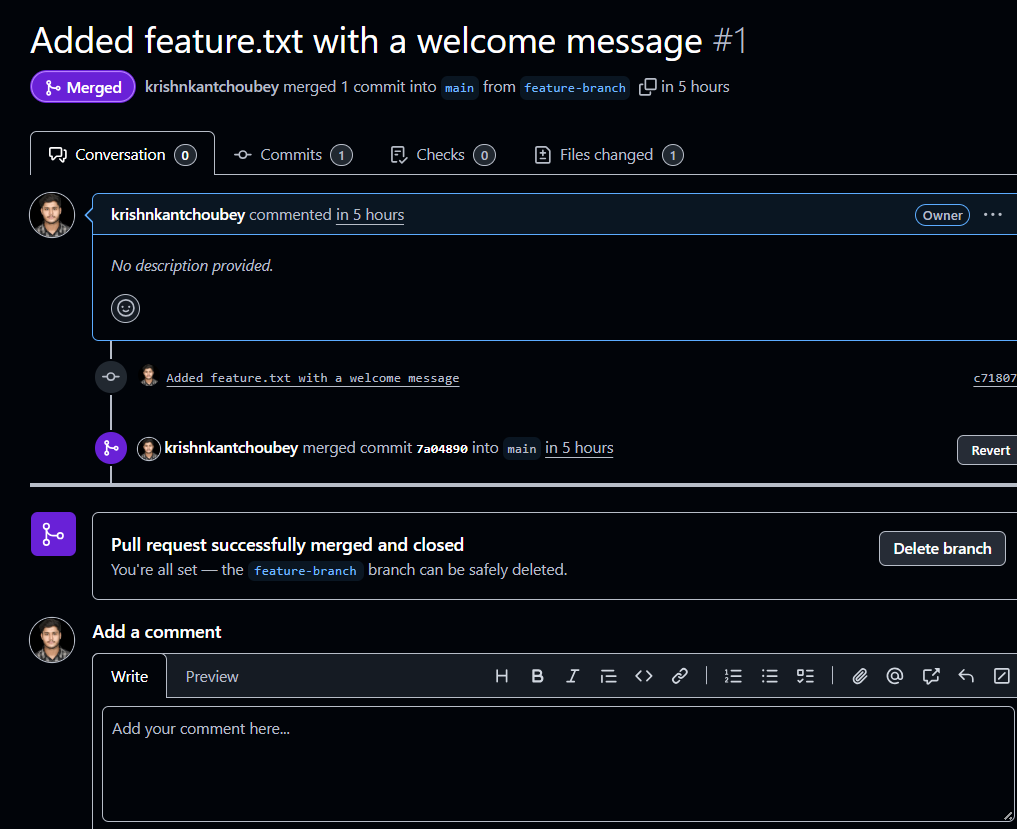
### F1) Repo,

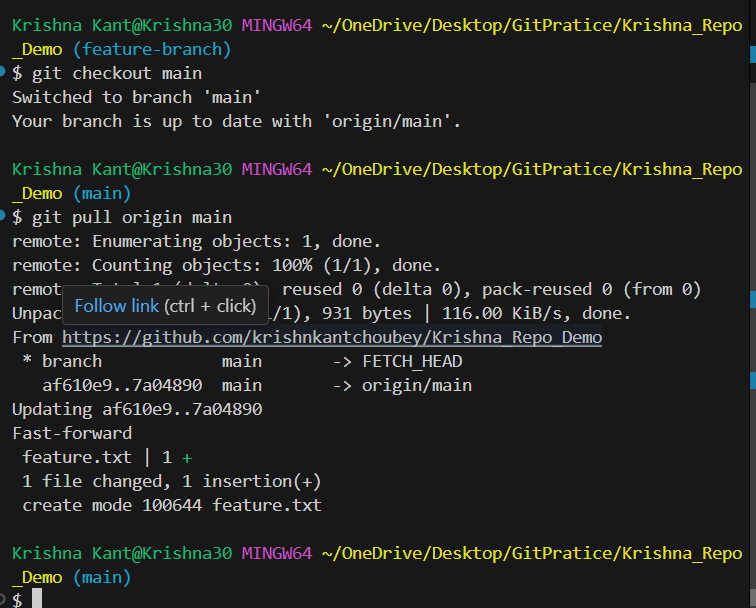
### branch, PR, merge

Task: Create GitHub repo, push to feature branch, raise PR, merge.  
Deliverables: Command history, GitHub PR screenshots, merged commit link.  
Success criteria: PR merged; branch history shows change.G. Maven









### G1) Build/deploy Address Book project

Task: Clone repo, run clean build, produce artifact, run locally.  
Deliverables: Build logs, artifact path, running proof.  
Success criteria: Build succeeds, artifact produced, app runs.

## H. SonarQube

### H1) QA report for Address Book

Task: Analyze Address Book project with SonarQube, capture metrics.  
Deliverables: Dashboard screenshots or report.  
Success criteria: Analysis completed; metrics visible.

## I. Jenkins

### I1) Install Jenkins on Ubuntu EC2

Task: Provision EC2, install Jenkins, ensure starts on boot, access UI.  
Deliverables: Commands, service status, UI screenshot.  
Success criteria: Jenkins reachable and setup completed.