UNIVERSITY OF PETROLEUM AND ENERGY STUDIES (UPES)

Linux Practical Report

Submitted by:

Name: Aadarshi Bisht

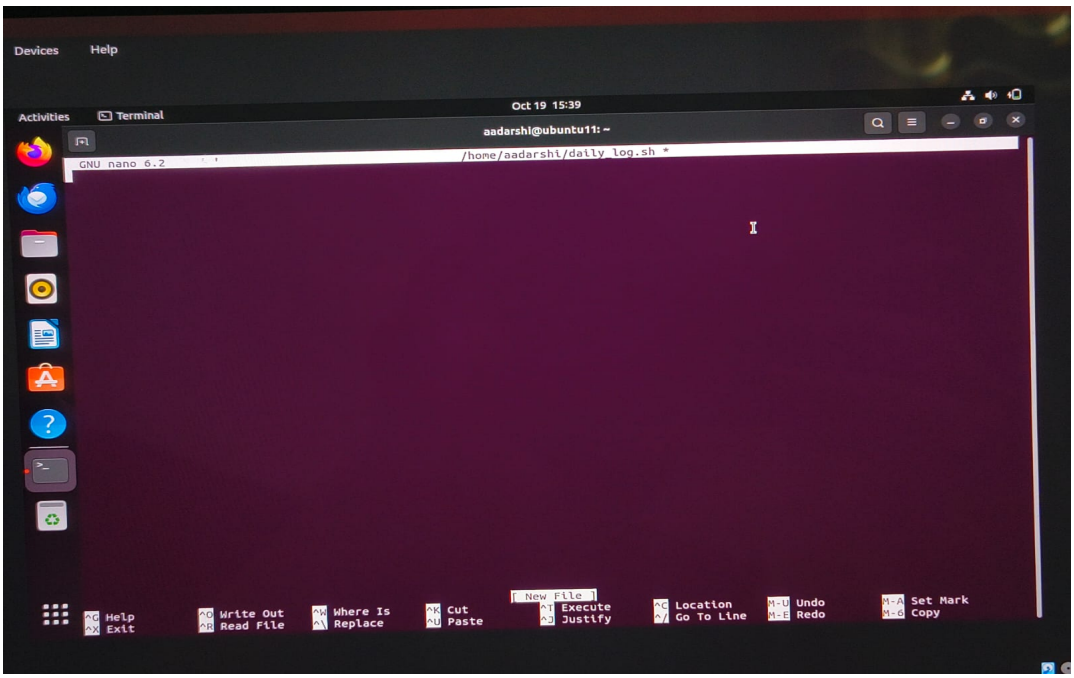
SAP ID: 590029330

Batch: 77-78

Subject: Linux

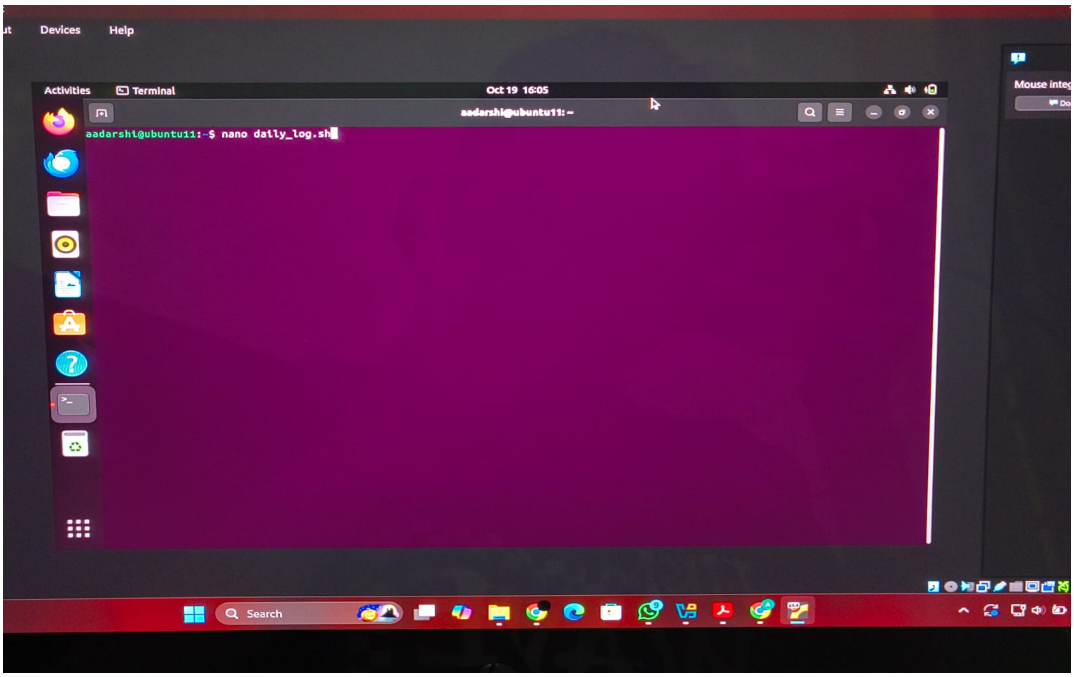
School of Computer Science

Step 1: Create the daily_log.sh script file using nano editor.



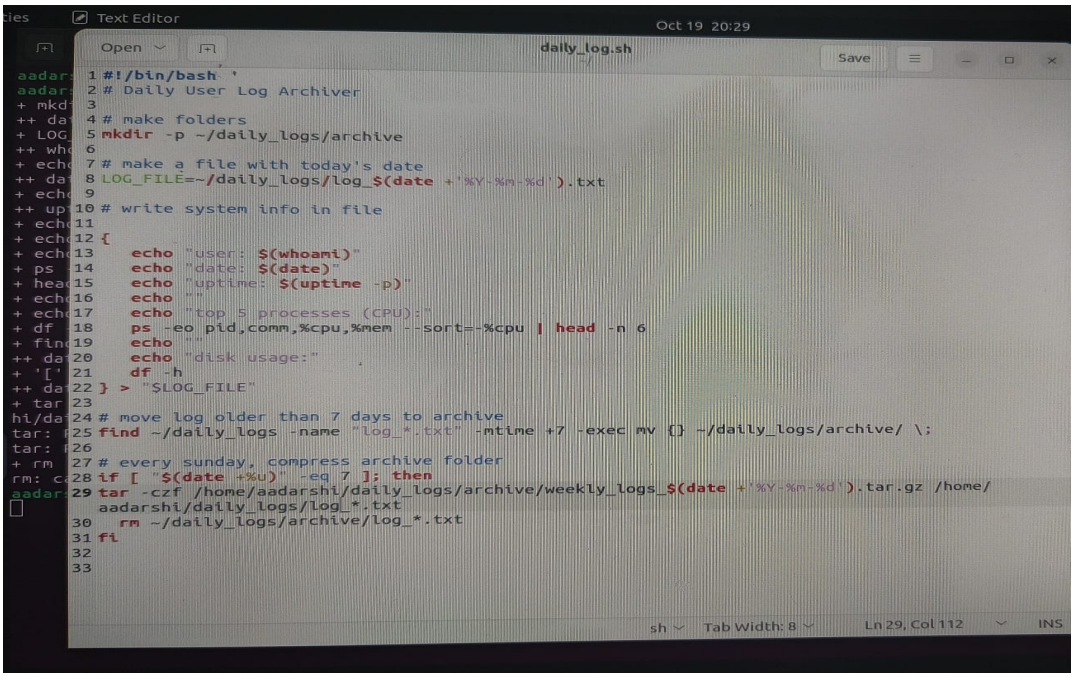
In this step, we create the `daily_log.sh` file using the nano text editor. This script will help automate daily system logging tasks.

Step 2: Write commands to capture system information and save it to a log file.



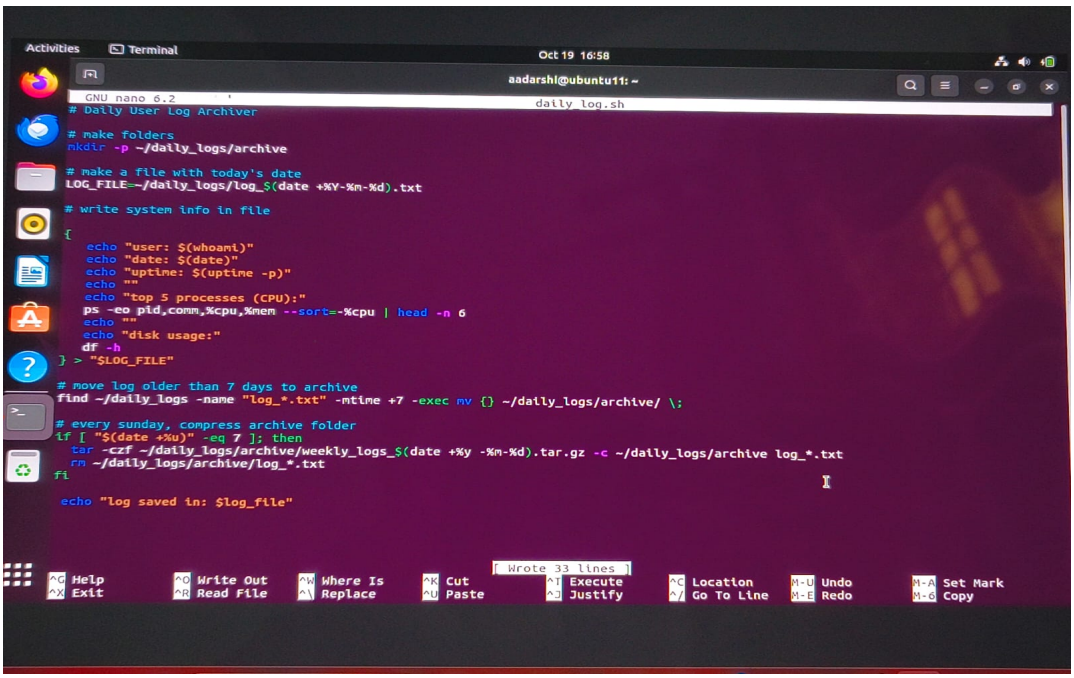
We add commands to gather information such as uptime, disk usage, and running processes. This helps track system performance.

Step 3: Add file creation date and user details using `whoami` and `date`.



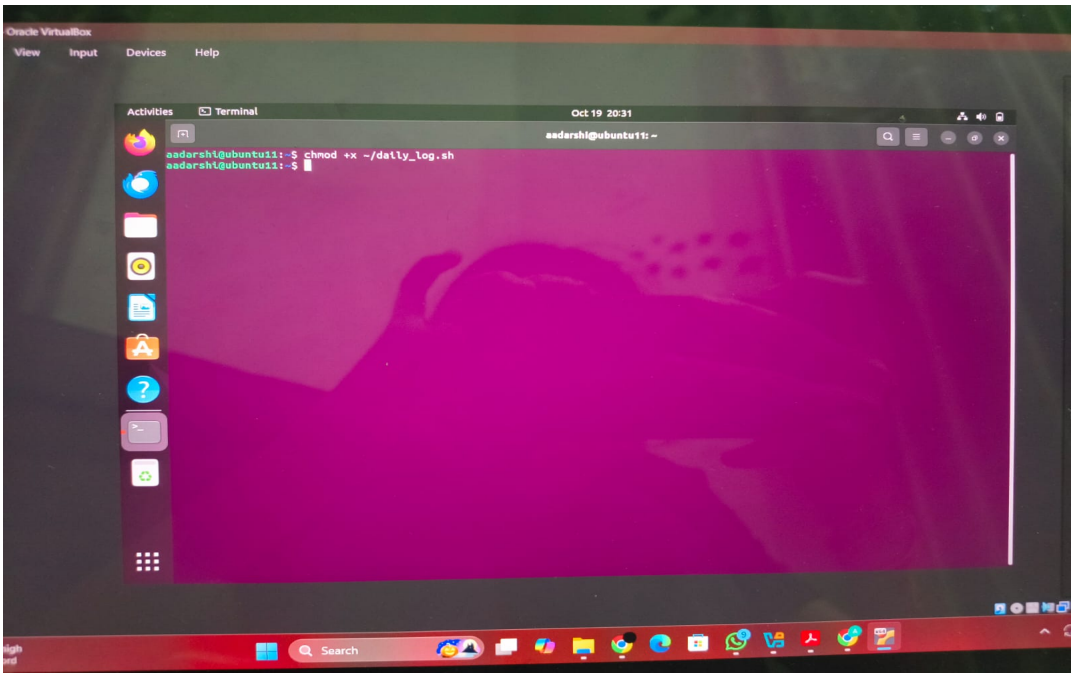
The script adds user details and date automatically to make logs unique and time-stamped.

Step 4: Include top 5 CPU processes and disk usage information.



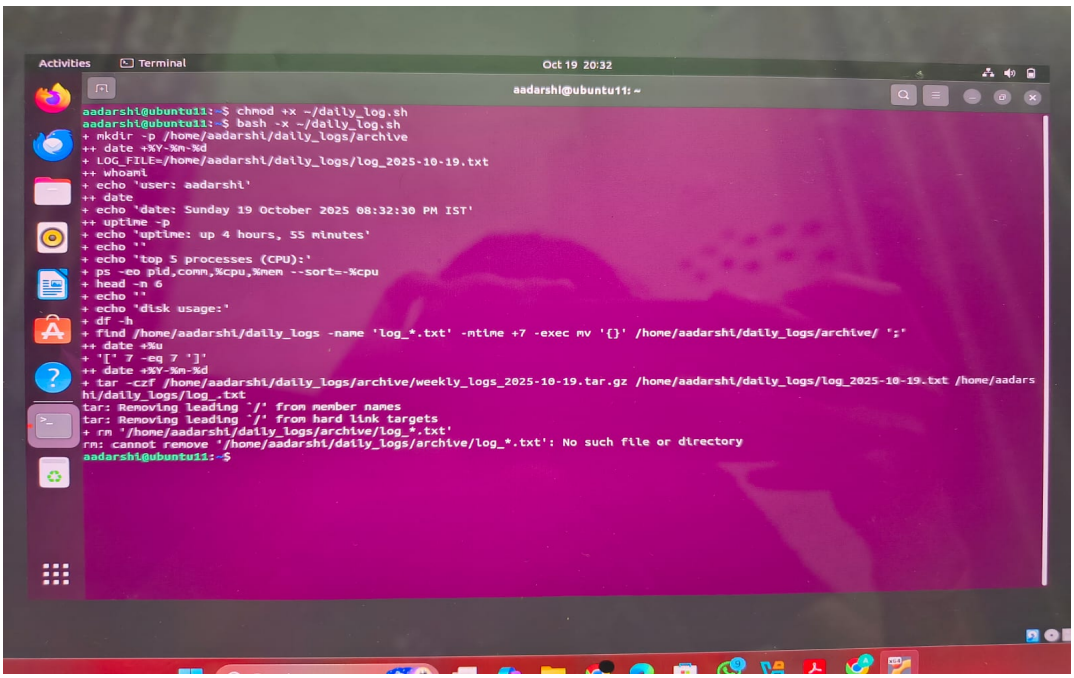
By capturing top CPU processes and disk usage, the log provides valuable system monitoring data.

Step 5: Set file permissions using chmod to make script executable.



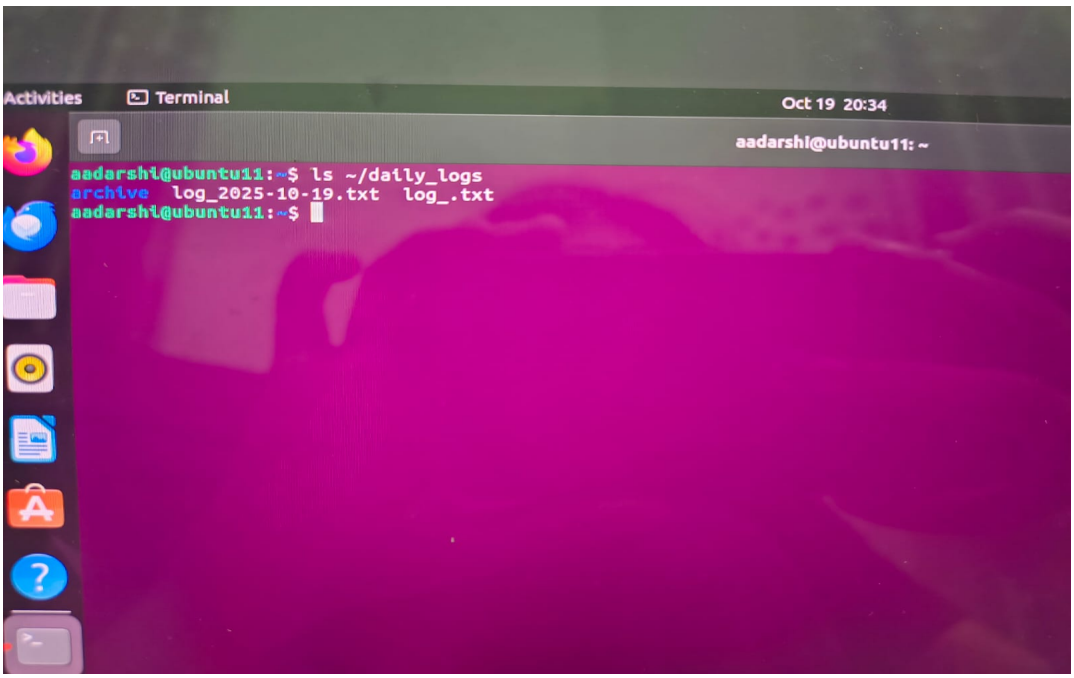
We change file permissions using `chmod +x` to make the script executable.

Step 6: Execute the script and check the generated log file output.



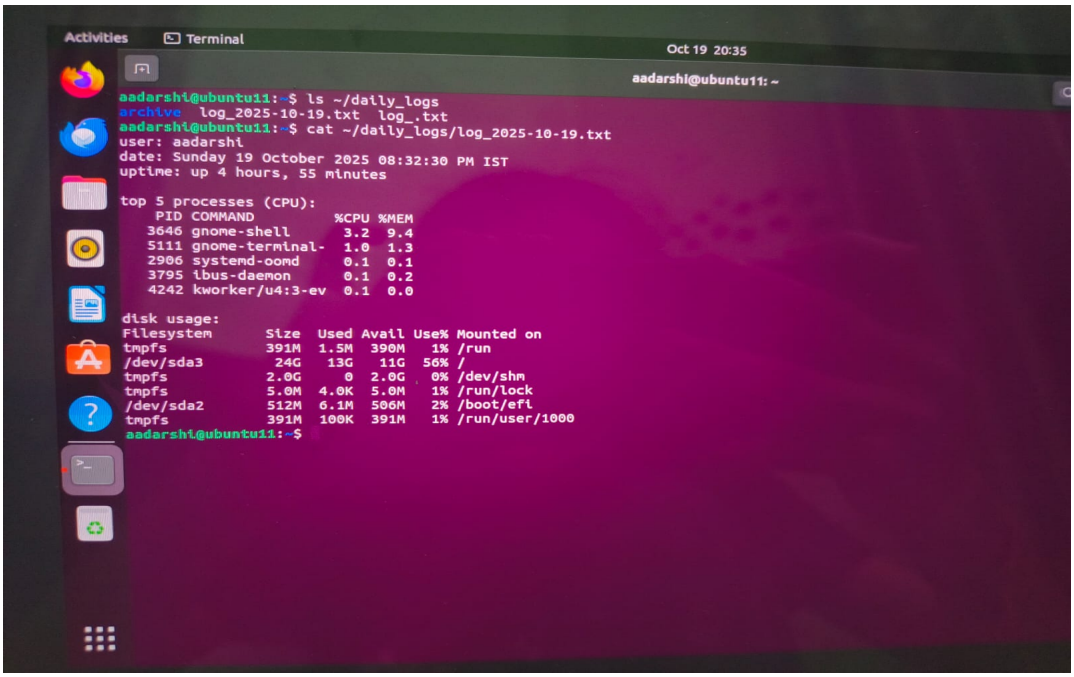
The script is executed, and a log file is generated inside the `/daily_logs/` folder containing system details.

Step 7: Archive old logs using the find and tar commands.



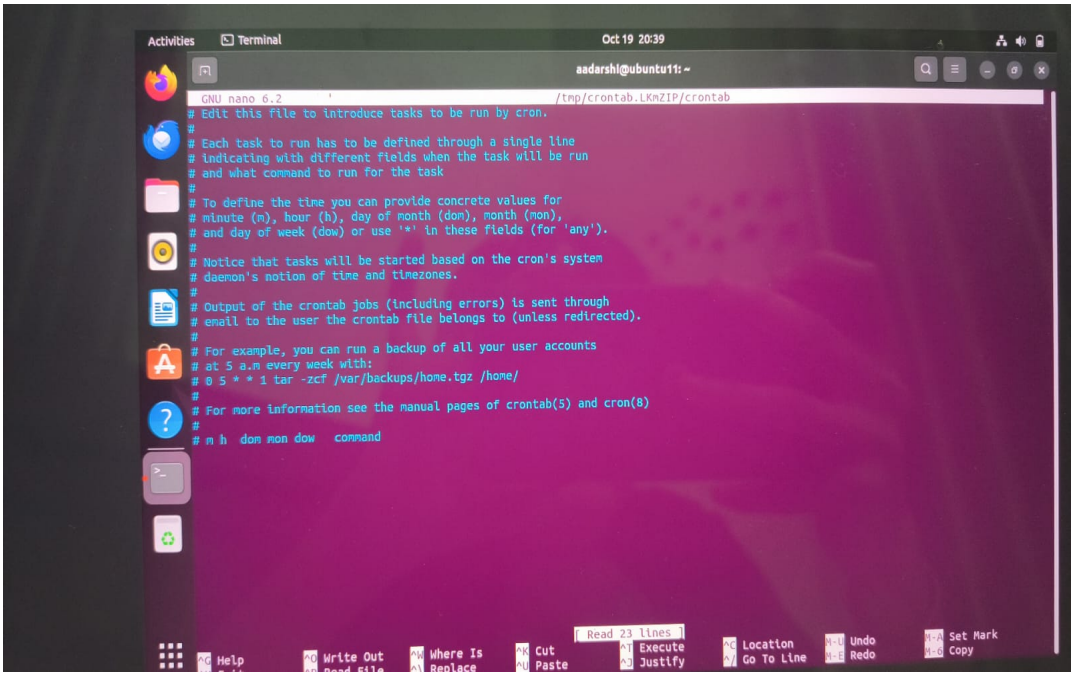
Old log files (older than 7 days) are archived into a weekly compressed file using tar.

Step 8: Schedule the script execution using cron job with `crontab -e`.



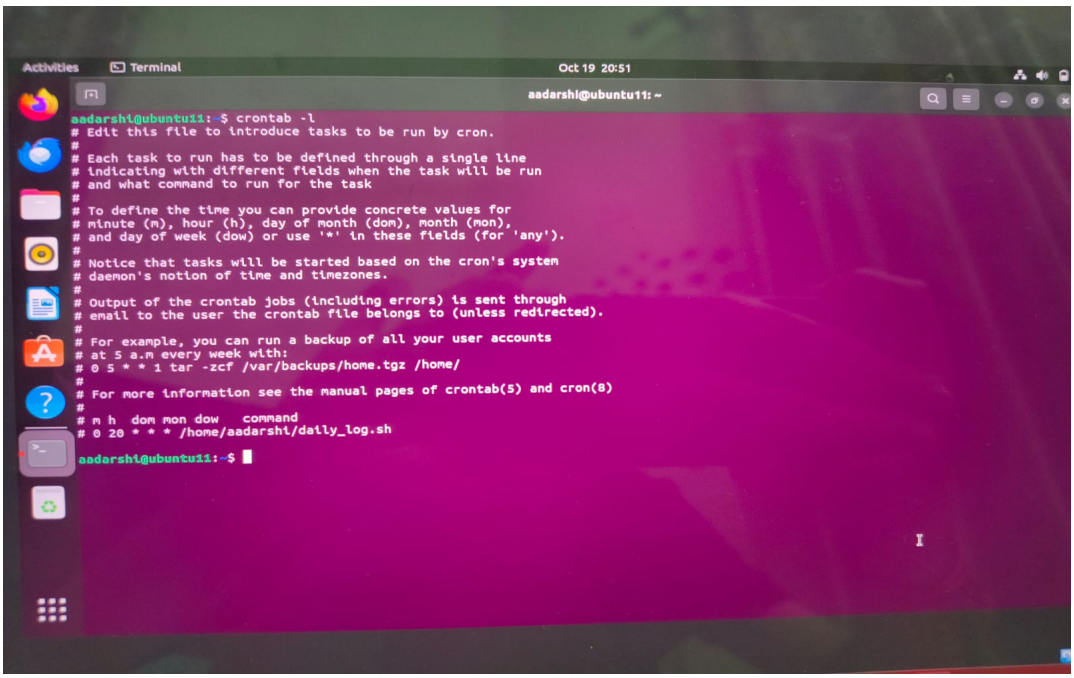
The cron scheduler is used to automate script execution daily at 8:00 PM using `crontab -e`.

Step 9: Verify the cron job with `crontab -l` command.



Verification of cron jobs ensures that the task is correctly scheduled and active.

Step 10: Check the final log and archive results.



The final results are checked, confirming proper log generation and archiving.