DOCKER

#Installing Docker on Ubuntu - go to official website (Docker docs)
sudo snap install docker

#mkdir fapp2
#touch app.py

app.py

```
from flask import Flask
app=Flask(__name__)
@app.route('/')
def run():
    return "vicky punda"
app.run('0.0.0.0',port=5000)
```

touch Dockerfile

Dockerfile

```
FROM python:3.10
WORKDIR /app
COPY . /app
#RUN pip install -r requirements.txt
RUN pip install flask
EXPOSE 5000
CMD ["python3","app.py"]
```

touch requirements.txt

requirements.txt
Flask==2.0.1
Werkzeug==2.0.1

docker build -t fapp2 .

#Docker.sock permission denied
#cd .. do till you are in /
#do ls -a
#cd var
#cd run
#ls - there is docker.sock
#sudo chmod 666 docker.sock

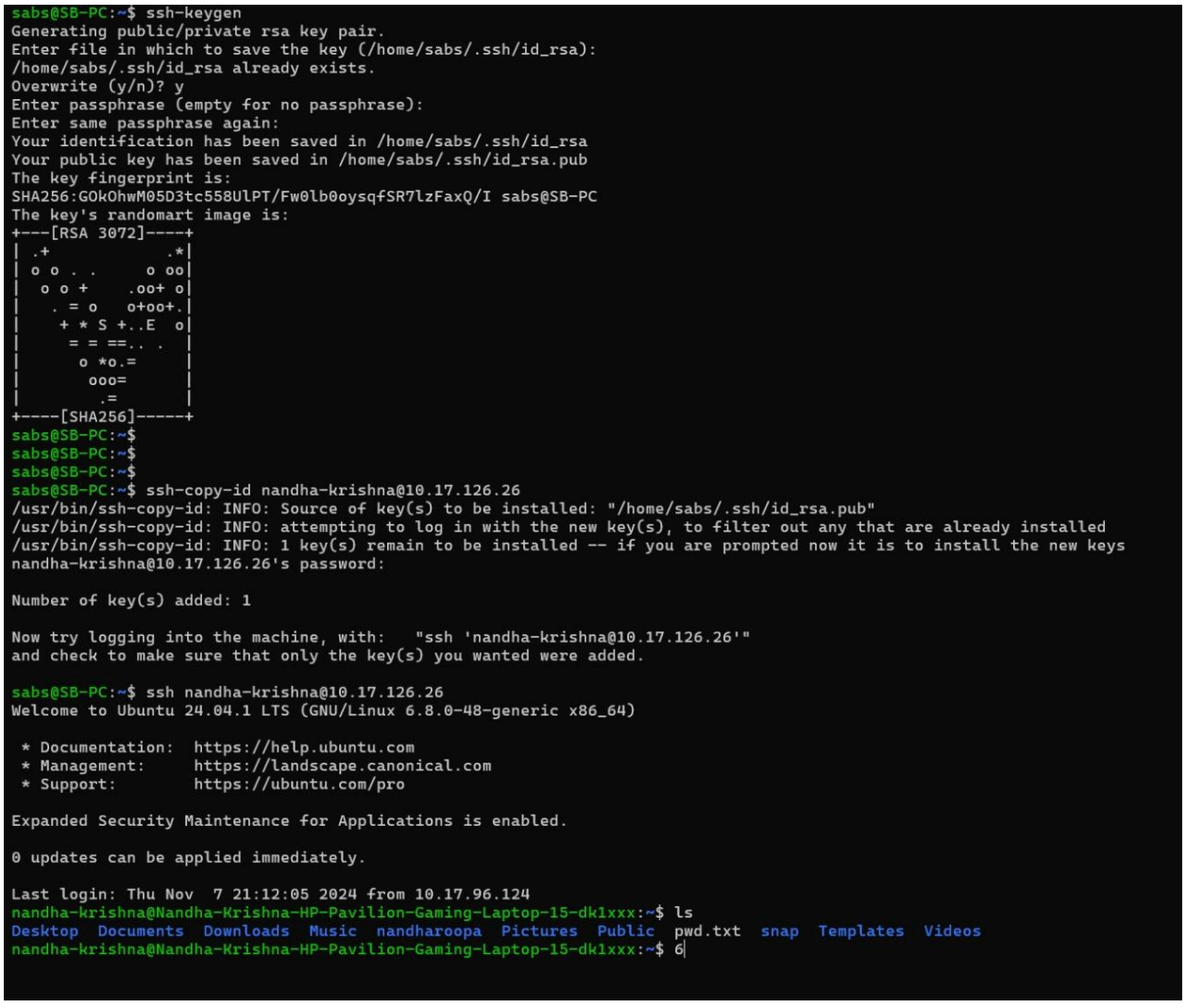
docker run -p 5000:5000 fapp2

SSH

ssh username@hostname_or_ip_address

If key pair not generated already

ssh-keygen
ssh-copy-id username@hostname_or_ip_address
ssh username@hostname_or_ip_address



Q) Implement shell script to transfer application data from local server to remote server using SSH (use scp command inside shell script)

Here is a simple shell script to transfer application data from a local server to a remote server using `scp` (secure copy) over SSH. This script will copy a specified directory or file from your local machine to a remote server, ensuring data is transferred securely.

```bash
#!/bin/bash

# Define variables
LOCAL_PATH="/path/to/local/application_data" # Replace with the path to your local data
REMOTE_USER="remote_username"                    # Replace with your remote server username
REMOTE_HOST="remote_server_ip_or_hostname"  # Replace with the remote server IP or hostname
REMOTE_PATH="/path/to/remote/destination"   # Replace with the path on the remote server where data will be copied

# Transfer data using scp
echo "Starting data transfer from $LOCAL_PATH to $REMOTE_USER@$REMOTE_HOST:$REMOTE_PATH"

scp -r "$LOCAL_PATH" "$REMOTE_USER@$REMOTE_HOST:$REMOTE_PATH"

fi
```

Explanation of Script:

1. **Variables**: Set the paths and credentials for the local and remote servers.
2. **`scp` Command**: `scp -r` copies files and directories recursively.
3. **Error Checking**: The `if` statement checks if the `scp` command succeeded. `0` indicates success, while other codes indicate errors.

Usage:

1. Make sure you have SSH access to the remote server (you may need to exchange SSH keys).
2. Save the script, e.g., `transfer_data.sh`.
3. Make the script executable:

   ```bash
 sudo chmod +x transfer_data.sh
```

```

4. Run the script:

```bash
./transfer_data.sh
```

This script provides a secure way to transfer application data with `scp`.

JENKINS

sudo apt-get install jenkins
sudo apt-get install fontconfig openjdk-17-jre (if Java doesnt exist)

sudo systemctl start jenkins
sudo systemctl status jenkins

Go to https://localhost:8080 - default Jenkins server

It will ask for the authentication key. Copy the path and use command

sudo cat “paste_the_path_here”

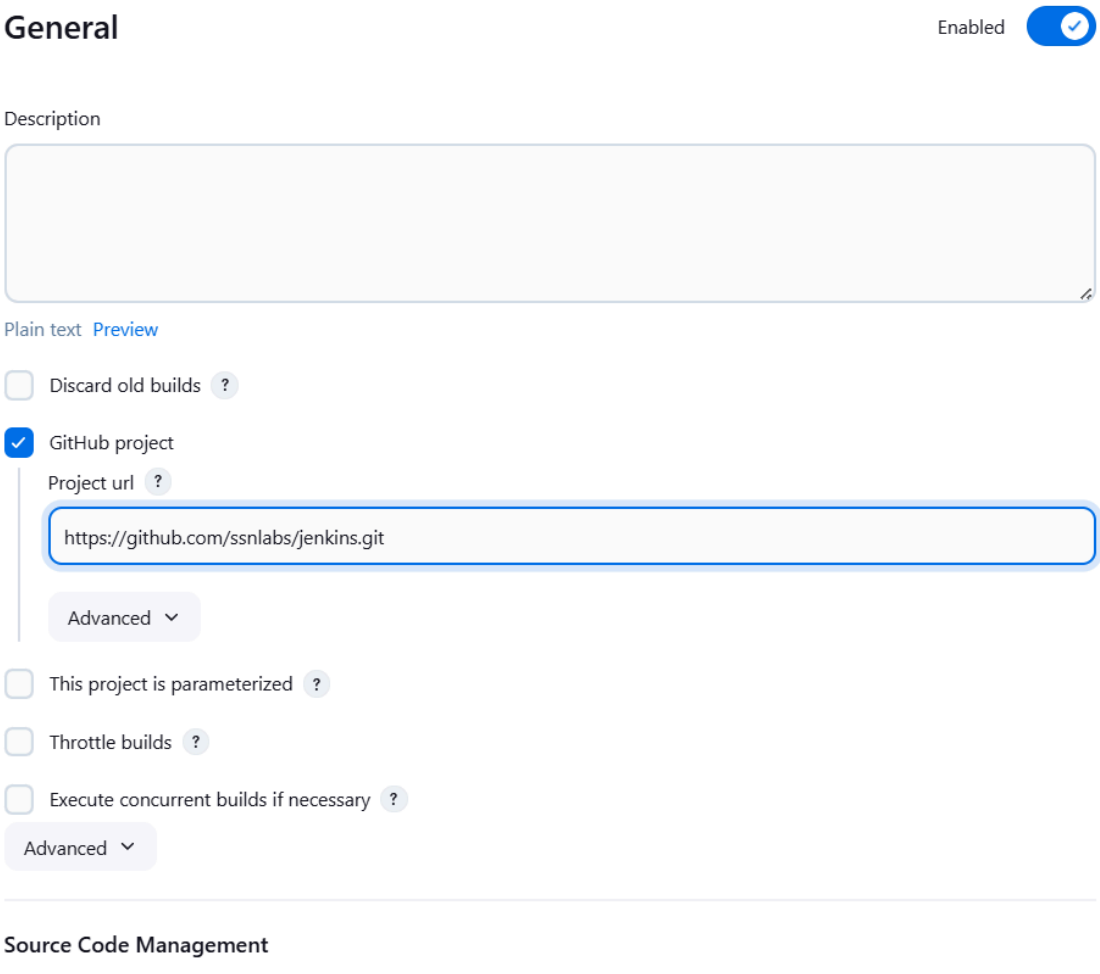
Paste the key from terminal to the Jenkins server

Create an account with username and password
```

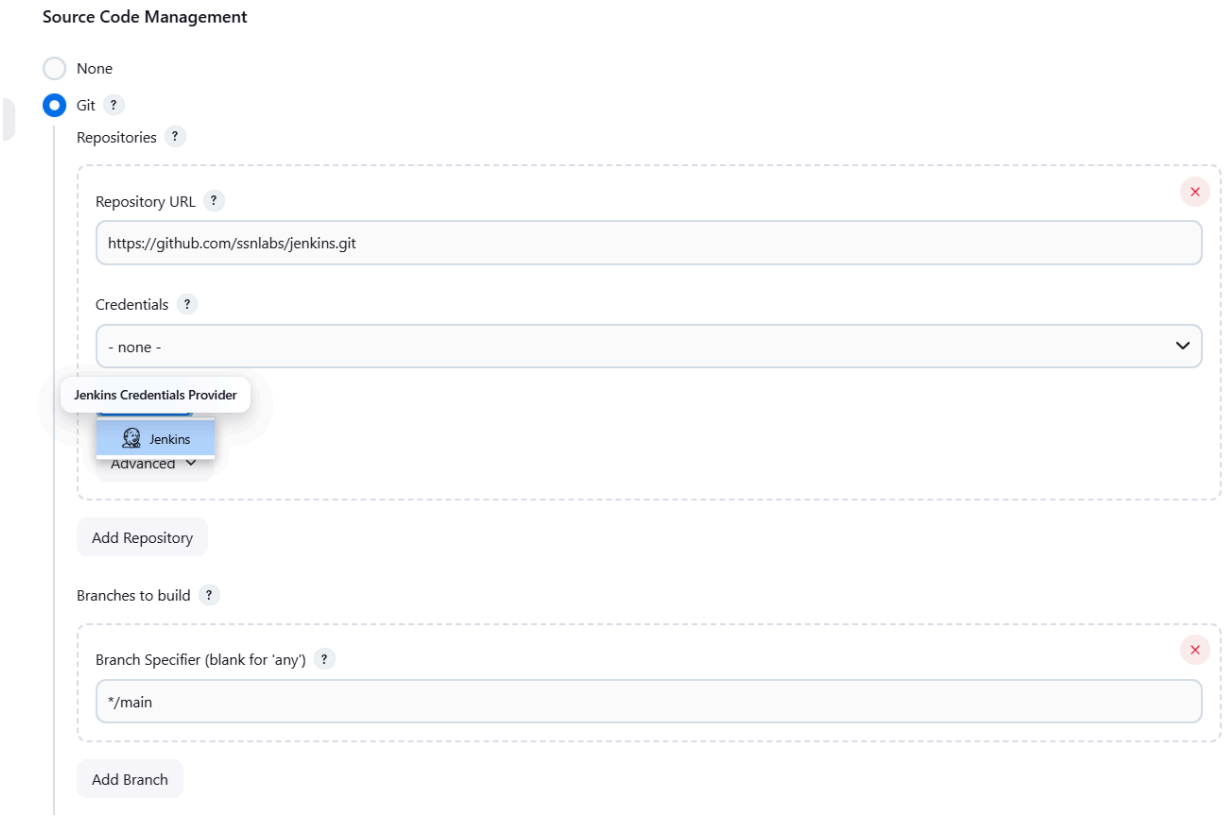
Create a repository with a Java file
For eg: https://github.com/ssnlabs/jenkins.git

Jenkins

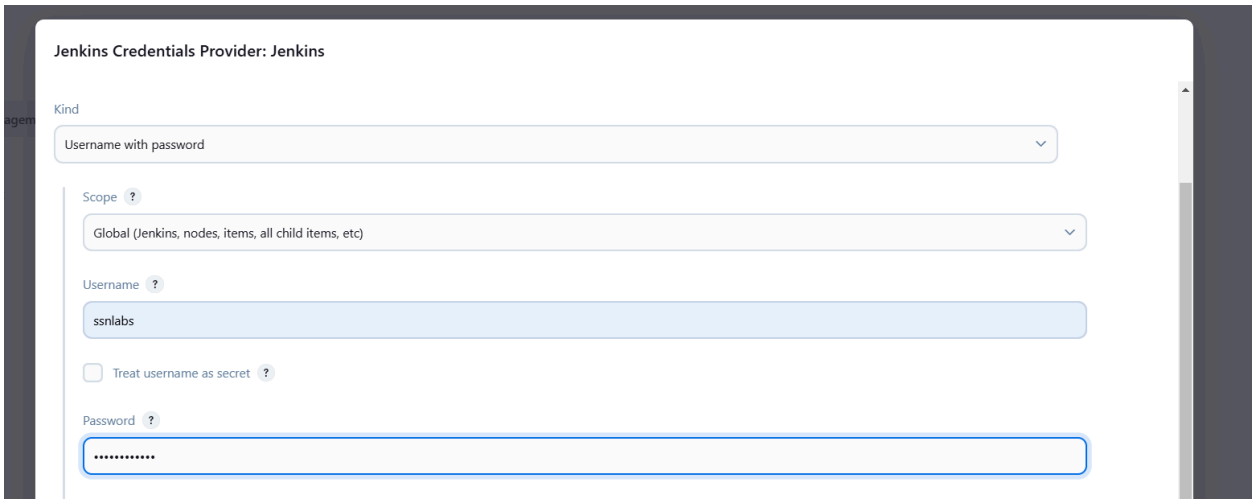
1. Click New Item
2. Give a name and select Freestyle Project
3. Select Github project and paste repository name



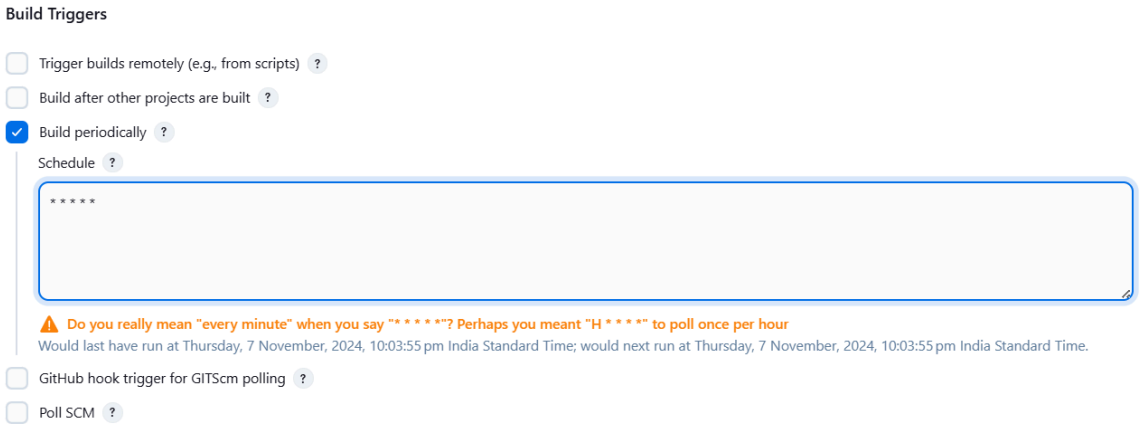
4. Select Git, Under Credentials, Click Add & Jenkins
   Change “Branch Specifier” to */main



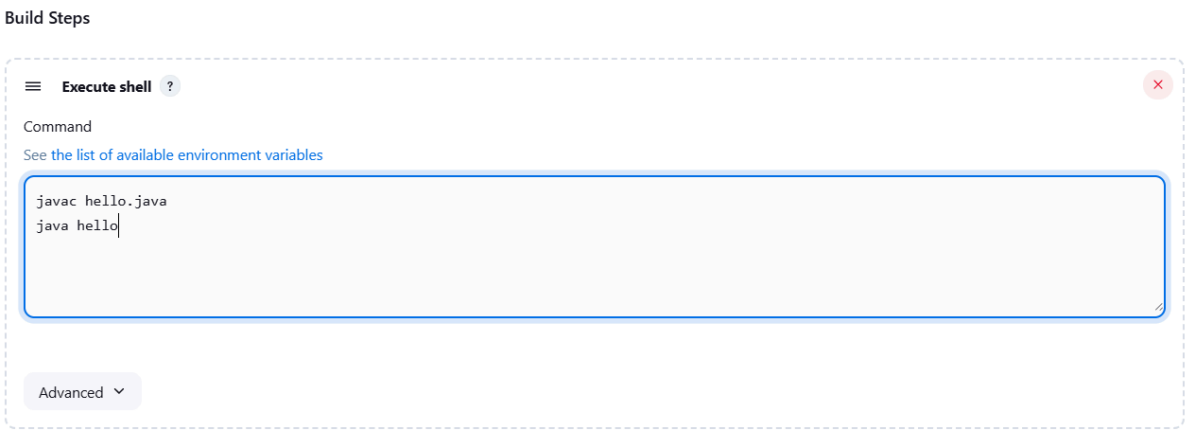
5. Enter your github username and password



6. Add * * * * * under Build Periodically -> Schedule (Cron Job - triggers a build every minute)



7. Under Build Step -> Select Execute Shell and enter the following



8. Give Save and Build Now in next page
9. Enjoy!

Pipeline script

```
pipeline {
    agent any

    stages {
    stage('Checkout') {
    steps {
        // Replace 'your-repository-url' with your repository URL
        git url: 'https://github.com/your-repository-url.git'
    }
    }

    stage('Build') {
    steps {
        // Commands to build the application
        // Example for a Node.js application
        sh 'npm install'
        sh 'npm run build'
    }
    }

    stage('Test') {
    steps {
        // Commands to run tests
        // Example for running Jest tests
        sh 'npm test'
    }
    post {
        always {
```

```
                // Archive test results (JUnit format for example)
                junit 'reports/junit/*.xml'
                }
        }
        }

        stage('Results') {
        steps {
                // Display test results in Jenkins
                echo 'Displaying test results'
        }
        }
        }

        post {
        always {
        // Clean up workspace if necessary
        cleanWs()
        }
        }
}
```