Deploying a ML model using Streamlit, AWS ec2 (Ubuntu), and github:

1. Create a file with your final model .pkl file (ex: my_model.pkl)
2. Create your streamlit app file and add this to the folder (ex: my_app.py)
3. Create a requirements.txt file
    a. You will add all packages needed for running your model/streamlit app. You can find this by accessing your command prompt and typing: **pip list** this will print a list of all current packages installed.
    b. Format of requirements.txt example:

```
xgboost == 2.1.1
scikit-learn == 1.5.2
pandas == 2.2.3
numpy == 1.26.4
streamlit == 1.39.0
```

4. Create a github repository with these files (model.pkl, app.py, requirements.txt)
5. Open your AWS account and go to the ec2 dashboard
6. Click orange 'Launch Instance' button
7. Create your instance:
    a. Name and Tags: Name your server
    b. Application and OS Images: select the Ubuntu tile and leave everything as default
    c. Instance type: t2.micro (for free tier)
    d. Key pair (login): select your .pem or create one and then add it if needed
    e. Network settings: check all 3 boxes
        i. Allow SSH traffic from
        ii. Allow HTTPS traffic from the internet
        iii. Allow HTTP traffic from the internet
    f. Click Launch instance on right side of screen
8. Once your instance is launched click the **View all instances** button
9. If your new instance is not showing, refresh until it shows and shows Running
10. Click on your new 'Instance ID' and scroll down to select the **Security** tab
11. Under Security details click the link for Security groups
12. Select 'Edit inbound rules'
13. Click 'Add rule' button and add the following rule:
    a. Type: Custom TCP
    b. Port range: 8501 (this is the streamlit port)
    c. Source: Anywhere IPv4 (should be set as default)
    d. Match the 0.0.0.0/0
    e. 'Save rules'
14. Go back to your instances, select your instance, and then click the 'Connect' button leave everything as default and click 'Connect' again
15. You will now be in the EC2 Instance console, you can type 'clear' and enter to clear the screen for a cleaner workspace at anytime without deleting anything important

Type or Copy and Paste the following lines of code one by one ensuring that each line fully executes before moving on to the next.

```
sudo apt update
sudo apt-get update
sudo apt upgrade -y
sudo apt install git curl unzip tar make sudo vim wget -y
git clone "Your-repository"
```

(example: go to repository with files click the down arrow on the green ‘Code’ button and under the HTTPS url copy that link and that is your repository URL it should end with .git as follows https://github.com/mygithub/myrepository.git)

ls (this will list your repository name either copy it or type it in the following line as follows)
cd your_repository
ls (double check that it returns a list of all necessary files, it should match those in the repository)

sudo apt install python3-pip *(when asked ‘Do you want to continue? [Y/n] type y enter)*
sudo apt install python3-venv *(when asked ‘Do you want to continue? [Y/n] type y enter)*

```
python3 -m venv nameyourenvironment
source nameofenvironment/bin/activate
pip3 install -r requirements.txt
```

Now you have two options: Temporary deployment which ends/closes when EC2 instance is closed or Permanent deployment stays accessible and running after closing EC2 instance

Temp Deployment: good to try this first to ensure everything is working properly

```
python3 -m streamlit run app.py
```

(This will return some urls. Copy and paste the second ‘External URL’ into your browser to check that your app is functioning correctly. If you are satisfied and ready for permanent deployment move on to next step. This URL is what will allow you and others to access the app. So, save it somewhere.)

Permanent Deployment: so that you can show off your work

```
nohup python3 -m streamlit run app.py
```

Now, you can close your ec2 instance. You should have successfully completed deployment and your app is live via the URL of that instance with the port 8501.