

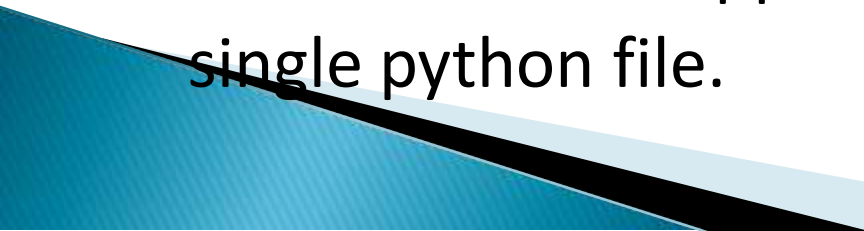
DEPLOY FLASK (OR)
PYTHON WEB
APPLICATION USING
GIT,GITHUB,JENKINS,TE
RRAFORM,ROUTE53 IN
AWS

SHAIK.GOUSERABBANI

METHOD-1

DEPLOY FLASK (OR) PYTHON WEB APPLICATION MANUALLY BY USING AWS RESOURCES

WHAT IS FLASK:

- ✓ Flask is a small and lightweight python web application framework that provides useful tools and feature that make creating web application in python easier.
 - ✓ It gives developers flexibility and it is a more accessible framework for new developer since you can build a web application quickly using only a single python file.
- 

WHAT IS PYTHON:

- ▶ It is a computer programming language often used to build websites and software automate tasks, and conduct data analysis.

WHAT IS PIP:

- ▶ PIP is a package manager python packages, or modules.

NOTE: if you have python version 3.4 or later PIP is include by default.

PRE-REQUISITES:

- ▶ AWS account
- ▶ IAM user
- ▶ Terminal
- ▶ Basic understand pf Python/Flask

SERVICES AND TOOLS USED:

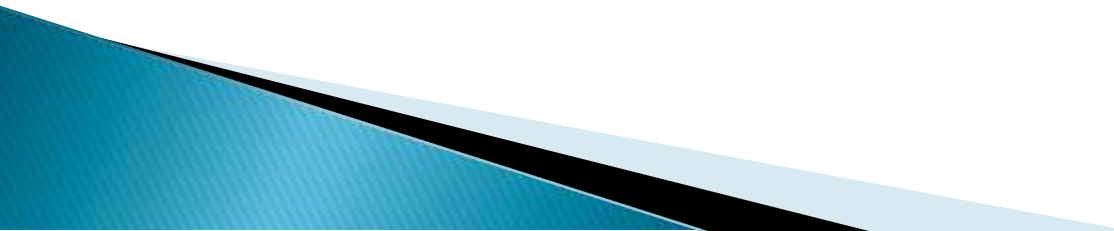
AWS SERVICE:

- ▶ IAM
 - ▶ VPC
 - ▶ EC2
 - ▶ ROUTE5
- 

DEVOPS TOOLS:

- ▶ Github
- ▶ Jenkins
- ▶ Terraform

STEP BY STEP PROCEDURE:

- ▶ Create and Login AWS Root Account.
 - ▶ Create a VPC along with subnets, route tables, internet gateway, elastic IP(if required),NACL(optional).
- 

- ▶ Create security group with respective ports
 - SSH = 22
 - HTTP = 80
 - HTTPS = 443
 - TCP = 8080,5000,7000
- ▶ Create EC2 instance launch with SSH
- ▶ Amazon-Linux2 (or) Ubuntu (or) RHEL
- ▶ Update ubuntu machine.
- ▶ *sudo apt update*
- ▶ Full upgrade the machine.
- ▶ *sudo apt-get full-upgrade -y*
- ▶ Install required packages or tools related for deployment project.
- ▶ *sudo apt-get install python3-pip*

- ▶ Install git and clone the project source code from Github
- ▶ *sudo git clone*
<https://github.com/GOUSERABBANI44/flask-library-app.git>
- ▶ Now, go to the source code directory
- ▶ *cd flask-library-app/*
- ▶ Now, install requirements packages
- ▶ *pip3 install -r requirements.txt*
- ▶ Run Flask server
- ▶ *python3 app.py (or) nohup python3 -u ./app.py &*

- ▶ Here, after running `python app.py` it will generate localhost IP address. We can't access web app with that IP address. Then we want to edit the file `app.py` with some details.
- ▶ *`sudo vi app.py`*
- ▶ Go to very bottom of the file and paste this below text and save the file.
- ▶ *`app.run(host='0.0.0.0', port=8080, debug=True)`*
- ▶ Now, again run the Flask server by using below command
- ▶ *`python3 app.py`*

- ▶ Now , copy EC2 instance public IP and give port number and search in web browser.
- ▶ *IP:8080*
- ▶ We will get output like this

The screenshot shows a web browser window with the address bar displaying '52.53.197.22:8080'. The browser tabs include 'Your VPCs', 'VPC Mana', 'gouserabbani.tech - d', 'EC2 Management Cor', 'Connect to instance', and two instances of '52.53.197.22'. The page is titled 'Library' and features a dark sidebar with navigation links: 'Home', 'Books Store', 'Book Management' (Books, Add Books, Issue Books, Return Books), and 'Customer Management' (Customers, Add Customer). The main content area is titled 'Dashboard' and contains several data cards:

- Most Popular Books**

Harry Potter	3
The Death Collectors (Carson Ryder #2)	1
- Highest Paying Customers**

Grejo Joby	2400
------------	------
- Customers With Highest Debt**

Grejo Joby	100
------------	-----
- Books with Minimum Stock Left**

The Mystical Poems of Rumi 1: First Selection Poems 1-200	10
Harry Potter	52
The Death Collectors (Carson Ryder #2)	99
- Total Customers**: 1
- Total Book Titles**: 4
- Total Books in Stock**: 161
- Total Times Issued**: 7
- Total Transactions Amount**: 2400
- Total Debt Due**: 100

- ▶ After reverse proxy of nginx server I browse the server with ip4 address only

The screenshot shows a web browser window with multiple tabs. The active tab is at the IP address 52.53.197.22, displaying a library management dashboard. The browser's address bar shows 'Not secure | 52.53.197.22'. The dashboard has a dark sidebar on the left with the title 'Library' and a navigation menu. The main content area is light blue and contains several data cards. The top right of the dashboard shows a user profile for 'Librarian'.

Library

Navigation

- Home
- Books Store

Book Management

- Books
- Add Books
- Issue Books
- Return Books

Customer Management

- Customers
- Add Customer

Dashboard

Dashboard

Most Popular Books

Harry Potter	3
The Death Collectors (Carson Ryder #2)	1

Highest Paying Customers

Grejo Joby	2400
------------	------

Customers With Highest Debt

Grejo Joby	100
------------	-----

Books with Minimum Stock Left

The Mystical Poems of Rumi 1: First Selection Poems 1-200	10
Harry Potter	52
The Death Collectors (Carson Ryder #2)	99

Total Customers

1

Total Book Titles

4

Total Books in Stock

161

Total Times Issued

7

Total Transactions Amount

2400

Total Debt Due

100

- ▶ Finally, By using router 53 service I browse the web application with my domain name service(DNS)– gouserabbani.tech

The screenshot shows a web browser window with the address bar displaying `gouserabbani.tech`. The browser tabs include "Your VPCs", "gouserabbani", "EC2 Manager", "Connect to in", and several IP addresses. The browser's address bar shows "Not secure" and "gouserabbani.tech". The application is running in Incognito mode.

The application interface has a dark sidebar on the left with the title "Library". The sidebar contains the following navigation links:

- Navigation
 - Home
 - Books Store
- Book Management
 - Books
 - Add Books
 - Issue Books
 - Return Books
- Customer Management
 - Customers
 - Add Customer

The main content area is titled "Dashboard" and displays the following data cards:

- Most Popular Books**

Book Title	Count
Harry Potter	3
The Death Collectors (Carson Ryder #2)	1
- Highest Paying Customers**

Customer Name	Amount
Grejo Joby	2400
- Customers With Highest Debt**

Customer Name	Debt Amount
Grejo Joby	100
- Books with Minimum Stock Left**

Book Title	Stock Left
The Mystical Poems of Rumi 1: First Selection Poems 1-200	10
Harry Potter	52
The Death Collectors (Carson Ryder #2)	99

On the right side of the dashboard, there are six summary cards:

- Total Customers: 1
- Total Book Titles: 4
- Total Books in Stock: 161
- Total Times Issued: 7
- Total Transactions Amount: 2400
- Total Debt Due: 100

- ▶ Finally, By using router 53 service I browse the web application with my domain name service(DNS)–www.gouserabbani.tech

The screenshot shows a web browser window with multiple tabs. The active tab is 'www.gouserabbani.tech'. The address bar shows 'gouserabbani.tech'. The page is a library management dashboard. On the left is a dark sidebar with a 'Library' header and a 'Navigation' section. The main content area is titled 'Dashboard' and contains several data cards.

Navigation:

- Home
- Books Store
- Book Management**
 - Books
 - Add Books
 - Issue Books
 - Return Books
- Customer Management**
 - Customers
 - Add Customer

Dashboard Data:

- Most Popular Books:**

Harry Potter	3
The Death Collectors (Carson Ryder #2)	1
- Highest Paying Customers:**

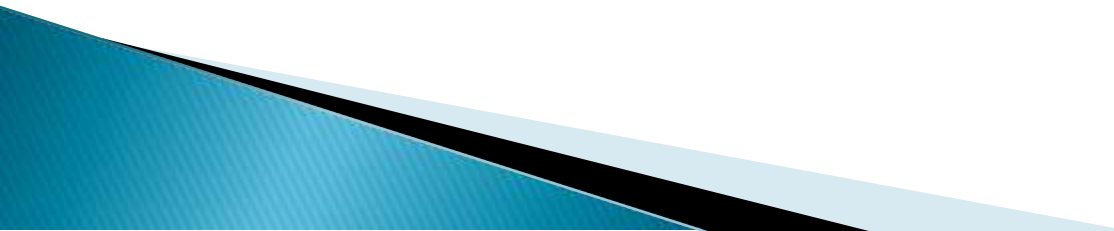
Grejo Joby	2400
------------	------
- Customers With Highest Debt:**

Grejo Joby	100
------------	-----
- Books with Minimum Stock Left:**

The Mystical Poems of Rumi 1: First Selection Poems 1-200	10
Harry Potter	52
The Death Collectors (Carson Ryder #2)	99
- Total Customers:** 1
- Total Book Titles:** 4
- Total Books in Stock:** 161
- Total Times Issued:** 7
- Total Transactions Amount:** 2400
- Total Debt Due:** 100

Method-2

DEPLOYING PYTHON WEB APPLICATION USING USERDATA/BASHSCRIPT

- ▶ First login to the AWS account with respective credentials and go to the EC2 services.
 - ▶ Create a VPC along with subnets, route tables, internet gateway, elastic IP(if required),NACL(optional).
- 

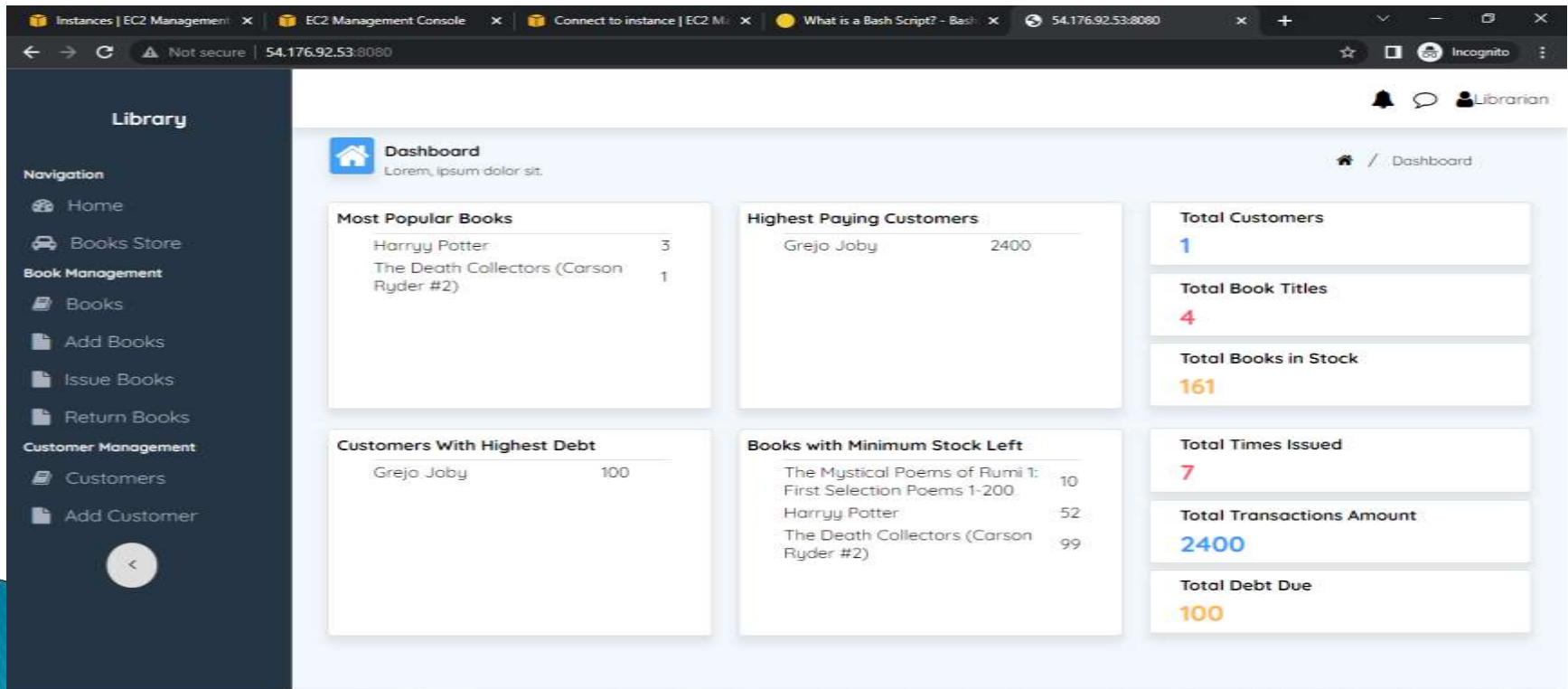
- ▶ Now create and launch EC2 instance by selecting ubuntu or amazonlinux2 versions with respective ports, they are SSH(22), HTTP(80), HTTPS(443), and Custom (8080).
- ▶ Now create the bash script at user data field with respective commands, those are
 - UBUNTU
 - `#!/bin/bash`
 - `sudo apt update`
 - `sudo apt-get full-upgrade -y`
 - `sudo apt-get install python3-pip`
 - `git clone https://github.com/GOUSERABBANI44/flask-library-app.git`

- `cd /`
- `sudo mv flask-library-app /home/ubuntu/`
- `cd /home/ubuntu`
- `cd flask-library-app`
- `pip3 install -r requirements.txt`
- `nohup python3 -u ./app.py &`

◦ AMAZONLINUX2

- `#!/bin/bash`
- `sudo yum -y update`
- `sudo yum -y install git`
- `git clone https://github.com/GOUSERABBANI44/flask-library-app.git`
- `sudo yum -y python3-pip`
- `cd flask-library-app`
- `pip3 install -r requirements.txt`
- `nohup python3 -u ./app.py &`

- ▶ Now launch the instance by using the SSH command with Git-bash or CMD
- ▶ After launching the instance just go with public ip of launched instance copy it and browse it along with port 8080.



- ▶ Finally, By using router 53 service I browse the web application with my domain name service(DNS)– gouserabbani.tech

The screenshot shows a web browser window with the address bar displaying `gouserabbani.tech`. The browser tabs include "Your VPCs", "gouserabbani", "EC2 Manager", "Connect to in", and several IP addresses. The browser's address bar shows "Not secure" and "gouserabbani.tech".

The web application is titled "Library" and has a dark sidebar with the following navigation links:

- Navigation
 - Home
 - Books Store
- Book Management
 - Books
 - Add Books
 - Issue Books
 - Return Books
- Customer Management
 - Customers
 - Add Customer

The main content area is titled "Dashboard" and displays the following data cards:

- Most Popular Books**

Book Title	Count
Harry Potter	3
The Death Collectors (Carson Ryder #2)	1
- Highest Paying Customers**

Customer Name	Amount
Grejo Joby	2400
- Total Customers**

1
- Total Book Titles**

4
- Total Books in Stock**

161
- Customers With Highest Debt**

Customer Name	Debt Amount
Grejo Joby	100
- Books with Minimum Stock Left**

Book Title	Stock Left
The Mystical Poems of Rumi 1: First Selection Poems 1-200	10
Harry Potter	52
The Death Collectors (Carson Ryder #2)	99
- Total Times Issued**

7
- Total Transactions Amount**

2400
- Total Debt Due**

100

- ▶ Finally, By using router 53 service I browse the web application with my domain name service(DNS)–www.gouserabbani.tech

The screenshot shows a web browser window with the address bar displaying www.gouserabbani.tech. The page is a library management dashboard. On the left is a dark sidebar with the title 'Library' and a 'Navigation' section. The main content area is titled 'Dashboard' and contains several data cards.

Navigation:

- Home
- Books Store
- Book Management**
 - Books
 - Add Books
 - Issue Books
 - Return Books
- Customer Management**
 - Customers
 - Add Customer

Dashboard Data:

- Most Popular Books:**

Harry Potter	3
The Death Collectors (Carson Ryder #2)	1
- Highest Paying Customers:**

Grejo Joby	2400
------------	------
- Customers With Highest Debt:**

Grejo Joby	100
------------	-----
- Books with Minimum Stock Left:**

The Mystical Poems of Rumi 1: First Selection Poems 1-200	10
Harry Potter	52
The Death Collectors (Carson Ryder #2)	99
- Total Customers:** 1
- Total Book Titles:** 4
- Total Books in Stock:** 161
- Total Times Issued:** 7
- Total Transactions Amount:** 2400
- Total Debt Due:** 100


Method-3

DEPLOYING PYTHON WEB APPLICATION USING GIT,GITHUB AND JENKINS

- ▶ First login to the AWS account with respective credentials and go to the EC2 services.
- ▶ Create a VPC along with subnets, route tables, internet gateway, elastic IP(if required),NACL(optional).
- ▶ Now create and launch EC2 instance by selecting ubuntu or amazonlinux2 versions with respective ports, they are SSH(22), HTTP(80), HTTPS(443), and Custom (8080), (9000).

- ▶ Now launch the instance by using the SSH command with Git-bash or putty.
- ▶ For ubuntu linux server use this commands as, This is the Debian package repository of Jenkins to automate installation and upgrade. To use this repository, first add the key to your system
- ▶ `curl -fsSL https://pkg.jenkins.io/debian-stable/jenkins.io.key | sudo tee \`
`/usr/share/keyrings/jenkins-keyring.asc > /dev/null`
- ▶ `echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \`
`https://pkg.jenkins.io/debian-stable binary/ | sudo`
`tee \`
`/etc/apt/sources.list.d/jenkins.list > /dev/null`

- ▶ `sudo apt-get update`
- ▶ `sudo apt-get install fontconfig openjdk-11-jre`
- ▶ `sudo apt-get install jenkins`
- ▶ Now install Jenkins by using the below commands
For amazon-linux2.
- ▶ `sudo wget -O /etc/yum.repos.d/jenkins.repo`
<https://pkg.jenkins.io/redhat-stable/jenkins.repo> `sudo rpm --import`
<https://pkg.jenkins.io/redhat-stable/jenkins.io.key>
- ▶ `yum install fontconfig java-11-openjdk yum install jenkins`

- ▶ Next launch the Jenkins with browsing public ip along with port number 8080
 - ▶ Now create the job for cloning and clone the python web application repository at the git field where placed under the job and mention branch name also and build the job.
 - ▶ Next create build job in this copy the clone while creating build job and write the bash script for python web application deployment in the under build trigger field at Execute-shell option and build the job.
- 

UBUNTU:

- `cd /var/lib/jenkins/workspace/clone(repository cloned job name.)`
- `sudo apt-get install python3-pip -y`
- `pip3 install -r requirements.txt`
- `nohup python3 -u ./app.py &`

AMAZONLINUX2:

- `cd /var/lib/jenkins/workspace/clone(repository cloned job name.)`
- `sudo yum -y install python3-pip -y`
- `pip3 install -r requirements.txt`
- `nohup python3 -u ./app.py &`

- ▶ finally just go with public ip of launched instance copy it and browse it along with port 8080.
- ▶ Finally, By using router 53 service I browse the web application with my domain name service(DNS)–
gouserabbani.tech

The screenshot shows a web browser with multiple tabs open, including 'Your VPCs', 'gouserabbani', 'EC2 Manager', 'Connect to in', and two instances of '52.53.197.22'. The active tab is 'gouserabbani' showing the URL 'gouserabbani.tech'. The browser's address bar indicates 'Not secure' and 'Incognito (2)'. The dashboard is titled 'Library' and features a sidebar with the following navigation links: Home, Books Store, Book Management (Books, Add Books, Issue Books, Return Books), and Customer Management (Customers, Add Customer). The main dashboard area is titled 'Dashboard' and contains several data cards:

- Most Popular Books:**

Harry Potter	3
The Death Collectors (Carson Ryder #2)	1
- Highest Paying Customers:**

Grejo Joby	2400
------------	------
- Customers With Highest Debt:**

Grejo Joby	100
------------	-----
- Books with Minimum Stock Left:**

The Mystical Poems of Rumi 1: First Selection Poems 1-200	10
Harry Potter	52
The Death Collectors (Carson Ryder #2)	99
- Total Customers:** 1
- Total Book Titles:** 4
- Total Books in Stock:** 161
- Total Times Issued:** 7
- Total Transactions Amount:** 2400
- Total Debt Due:** 100

- ▶ Finally, By using router 53 service I browse the web application with my domain name service(DNS)–www.gouserabbani.tech

The screenshot shows a web browser window with the address bar displaying www.gouserabbani.tech. The page is a library management dashboard. On the left is a dark sidebar with the title 'Library' and a 'Navigation' section. The main content area is titled 'Dashboard' and contains several data cards.

Navigation:

- Home
- Books Store
- Book Management**
 - Books
 - Add Books
 - Issue Books
 - Return Books
- Customer Management**
 - Customers
 - Add Customer

Dashboard Data:

- Most Popular Books:**

Harry Potter	3
The Death Collectors (Carson Ryder #2)	1
- Highest Paying Customers:**

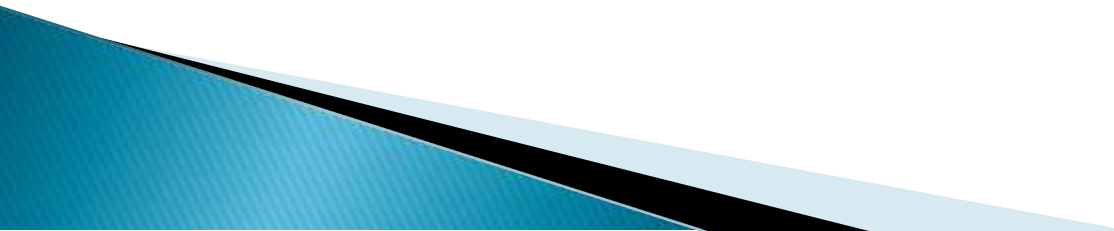
Grejo Joby	2400
------------	------
- Customers With Highest Debt:**

Grejo Joby	100
------------	-----
- Books with Minimum Stock Left:**

The Mystical Poems of Rumi 1: First Selection Poems 1-200	10
Harry Potter	52
The Death Collectors (Carson Ryder #2)	99
- Total Customers:** 1
- Total Book Titles:** 4
- Total Books in Stock:** 161
- Total Times Issued:** 7
- Total Transactions Amount:** 2400
- Total Debt Due:** 100

Method-4

DEPLOYING PYTHON WEB APPLICATION USING GIT,GITHUB AND TERRAFORM

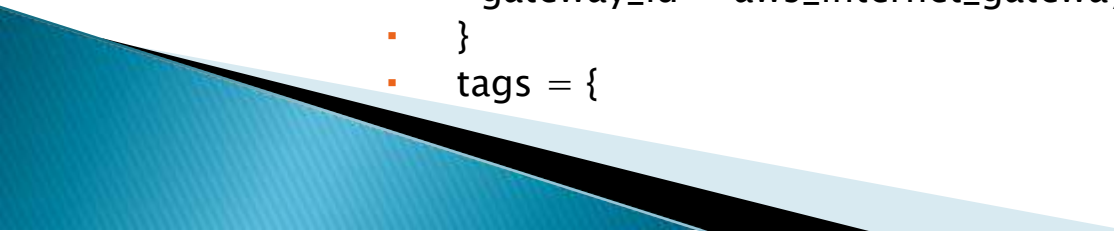
- ▶ First login to the AWS account with respective credentials and go to the EC2 services.
 - ▶ Create a VPC along with subnets, route tables, internet gateway, elastic IP(if required),NACL(optional).
- 

- ▶ Now create and launch EC2 instance by selecting ubuntu or amazonlinux2 versions with respective ports, they are SSH(22), HTTP(80), HTTPS(443), and Custom (8080).
- ▶ Now launch the instance by using the SSH command with Git-bash or CMD
- ▶ Now install the terraform by using below commands.
 - `wget -O- https://apt.releases.hashicorp.com/gpg | gpg --dearmor | sudo tee /usr/share/keyrings/hashicorp-archive-keyring.gpg`
 - `echo "deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg] https://apt.releases.hashicorp.com $(lsb_release -cs) main" | sudo tee /etc/apt/sources.list.d/hashicorp.list`
 - `sudo apt update && sudo apt install terraform`

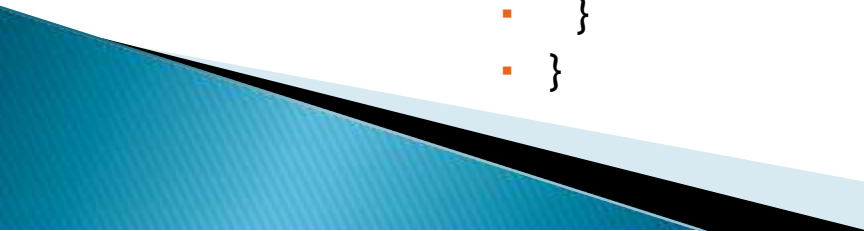
- ▶ Now create the .tf file by using vi mode and write script to entair vpc and EC2-instance as

- provider "aws" {
- region = "us-west-1"
- access_key = "AKIA352V5INCGY2RX2GX"
- secret_key =
"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"
- }
- resource "aws_vpc" "nvpc" {
- cidr_block = "10.0.0.0/16"
- instance_tenancy = "default"
- tags = {
- Name = "demovpc"
- }
- }

- #create igw
- resource "aws_internet_gateway" "ig" {
- vpc_id = aws_vpc.nvpc.id
- tags = {
- Name = "myigw"
- }
- }
-
- #create subnets
- resource "aws_subnet" "pub-sub" {
- vpc_id = aws_vpc.nvpc.id
- cidr_block = "10.0.0.0/20"
- availability_zone = "us-west-1a"
- map_public_ip_on_launch = "true"
- tags = {
- Name = "public"
- }
- }
-
- #create public-route-table
- resource "aws_route_table" "pub-route" {
- vpc_id = aws_vpc.nvpc.id
- route {
- cidr_block = "0.0.0.0/0"
- gateway_id = aws_internet_gateway.ig.id
- }
- tags = {



```
▪ Name = "pubroute"
▪ }
▪ }
▪
▪ #create subnet-association
▪ resource "aws_route_table_association" "a" {
▪   subnet_id      = aws_subnet.pub-sub.id
▪   route_table_id = aws_route_table.pub-route.id
▪ }
▪
▪ #create instance
▪ resource "aws_instance" "web" {
▪   ami              = "ami-0a1a70369f0fce06a"
▪   instance_type    = "t2.micro"
▪   key_name         = "ansible"
▪   subnet_id        = aws_subnet.pub-sub.id
▪   vpc_security_group_ids = ["${aws_security_group.sg.id}"]
▪   user_data         = file("mynn.sh")
▪   tags = {
▪     Name = "web-application"
▪   }
▪ }
```



```
▪ #create security group
▪ resource "aws_security_group" "sg" {
▪   vpc_id = aws_vpc.nvpc.id
▪
▪   ingress {
▪     from_port = 22
▪     to_port   = 22
▪     protocol  = "tcp"
▪     cidr_blocks = ["0.0.0.0/0"]
▪   }
▪
▪   ingress {
▪     from_port = 9000
▪     to_port   = 9000
▪     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"]
▪   }
▪   tags = {
▪     Name = "secure"
▪   }
▪ }
```

► Now create bash file for bash scripting by using vi command and the bash script For UBUNTU server

- `#!/bin/bash`
- `sudo apt update`
- `sudo apt-get full-upgrade -y`
- `git clone https://github.com/GOUSERABBANI44/flask-library-app.git`
- `sudo apt-get install python3-pip -y`
- `cd flask-library-app`
- `pip3 install -r requirements.txt`
- `nohup python3 -u ./app.py &`

- ▶ Now create bash file for bash scripting by using vi command and the bash script For AMAZON-LINUX2 server

- `#!/bin/bash`
- `sudo yum -y update`
- `git clone https://github.com/GOUSERABBANI44/flask-library-app.git`
- `sudo yum -y install python3-pip -y`
- `cd flask-library-app`
- `pip3 install -r requirements.txt`
- `nohup python3 -u ./app.py &`

- ▶ Now go to copy public ip of created server (webapplication) and browse it along with port number –9000.

The screenshot shows a web browser window with the address bar displaying '54.176.92.53:8080'. The browser tabs include 'Instances | EC2 Management', 'EC2 Management Console', 'Connect to instance | EC2 M...', and 'What is a Bash Script? - Bash...'. The browser's address bar shows 'Not secure | 54.176.92.53:8080'. The web application is titled 'Library' and features a dark sidebar with navigation links: 'Home', 'Books Store', 'Book Management' (with sub-links 'Books', 'Add Books', 'Issue Books', 'Return Books'), and 'Customer Management' (with sub-links 'Customers', 'Add Customer'). The main content area is titled 'Dashboard' and displays several widgets:

- Most Popular Books:**

Harry Potter	3
The Death Collectors (Carson Ryder #2)	1
- Highest Paying Customers:**

Grejo Joby	2400
------------	------
- Customers With Highest Debt:**

Grejo Joby	100
------------	-----
- Books with Minimum Stock Left:**

The Mystical Poems of Rumi 1: First Selection Poems 1-200	10
Harry Potter	52
The Death Collectors (Carson Ryder #2)	99
- Total Customers:** 1
- Total Book Titles:** 4
- Total Books in Stock:** 161
- Total Times Issued:** 7
- Total Transactions Amount:** 2400
- Total Debt Due:** 100

- ▶ Finally, By using router 53 service I browse the web application with my domain name service(DNS)– gouserabbani.tech

The screenshot shows a web browser window with the address bar displaying `gouserabbani.tech`. The browser tabs include "Your VPCs", "gouserabbani", "EC2 Manager", "Connect to in", and several IP addresses. The browser's address bar shows "Not secure" and "gouserabbani.tech".

The web application is titled "Library" and has a dark sidebar with the following navigation links:

- Navigation
 - Home
 - Books Store
- Book Management
 - Books
 - Add Books
 - Issue Books
 - Return Books
- Customer Management
 - Customers
 - Add Customer

The main content area is titled "Dashboard" and displays the following data cards:

- Most Popular Books**

Book Title	Count
Harry Potter	3
The Death Collectors (Carson Ryder #2)	1
- Highest Paying Customers**

Customer Name	Amount
Grejo Joby	2400
- Total Customers**

1
- Total Book Titles**

4
- Total Books in Stock**

161
- Customers With Highest Debt**

Customer Name	Debt Amount
Grejo Joby	100
- Books with Minimum Stock Left**

Book Title	Stock Left
The Mystical Poems of Rumi 1: First Selection Poems 1-200	10
Harry Potter	52
The Death Collectors (Carson Ryder #2)	99
- Total Times Issued**

7
- Total Transactions Amount**

2400
- Total Debt Due**

100

- ▶ Finally, By using router 53 service I browse the web application with my domain name service(DNS)–www.gouserabbani.tech

The screenshot shows a web browser window with the address bar displaying www.gouserabbani.tech. The browser tabs include "Your VPCs", "gouserabbani", "EC2 Manage", "Connect to in", and several instances of "52.53.197.22" and "gouserabbani". The browser's address bar shows "Not secure | gouserabbani.tech".

The web application is a library management dashboard. The sidebar on the left is titled "Library" and contains the following navigation links:

- Navigation
 - Home
 - Books Store
- Book Management
 - Books
 - Add Books
 - Issue Books
 - Return Books
- Customer Management
 - Customers
 - Add Customer

The main content area is titled "Dashboard" and displays the following data cards:

- Most Popular Books**

Harry Potter	3
The Death Collectors (Carson Ryder #2)	1
- Highest Paying Customers**

Grejo Joby	2400
------------	------
- Total Customers**

1
- Total Book Titles**

4
- Total Books in Stock**

161
- Customers With Highest Debt**

Grejo Joby	100
------------	-----
- Books with Minimum Stock Left**

The Mystical Poems of Rumi 1: First Selection Poems 1-200	10
Harry Potter	52
The Death Collectors (Carson Ryder #2)	99
- Total Times Issued**

7
- Total Transactions Amount**

2400
- Total Debt Due**

100

THANK

YOU

