DEPLOY FLASK (OR) PYTHON WEB APPLICATION USING GIT,GITHUB,JENKINS,TERRAFORM,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 version3.4 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*

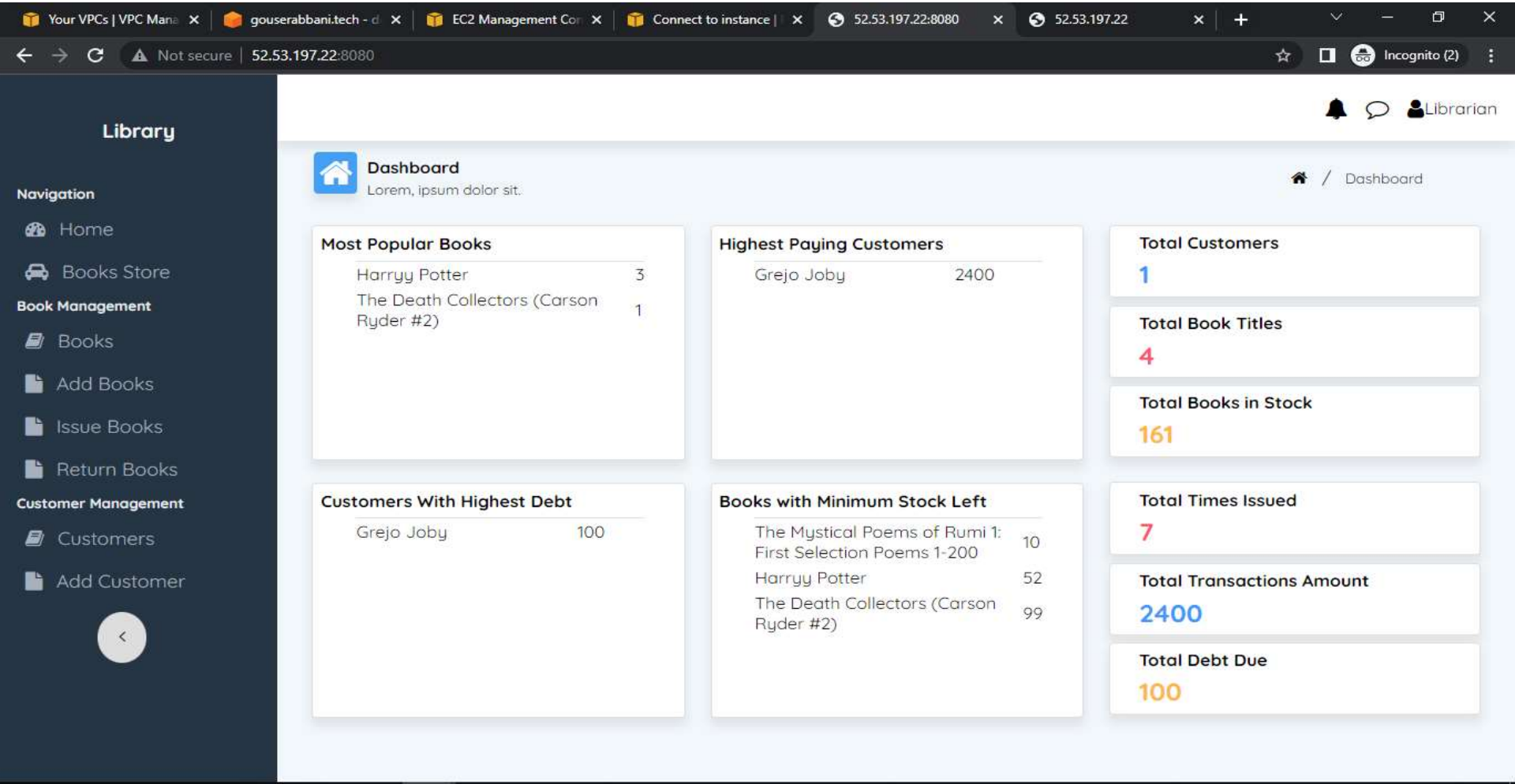
- Go to very bottom of the file and paste this below text and save the file.

- *app.run(host='0.0.0.0', port=8080, debug=True)*

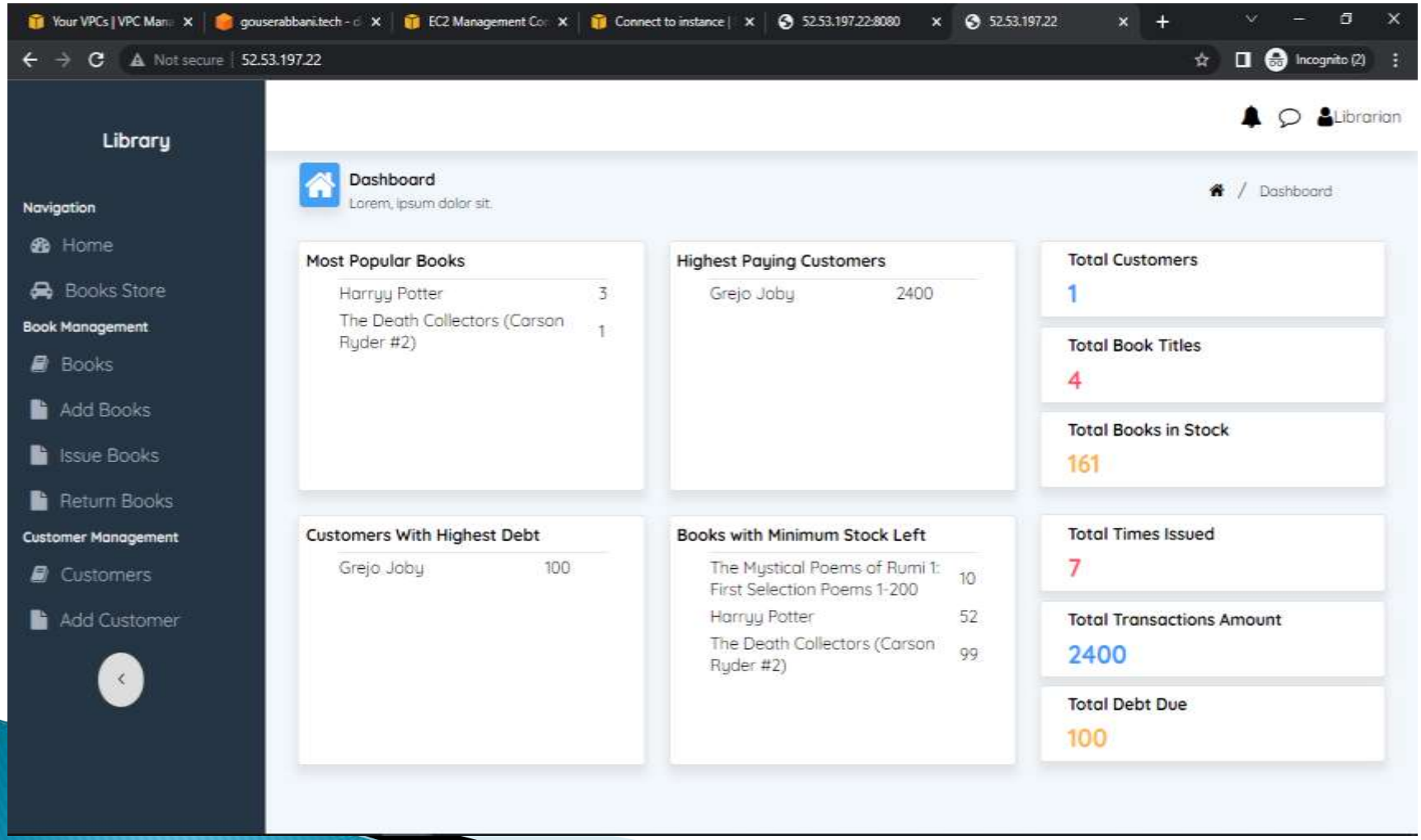
- Now, again run the Flask server by using below command

- *python3 app.py*

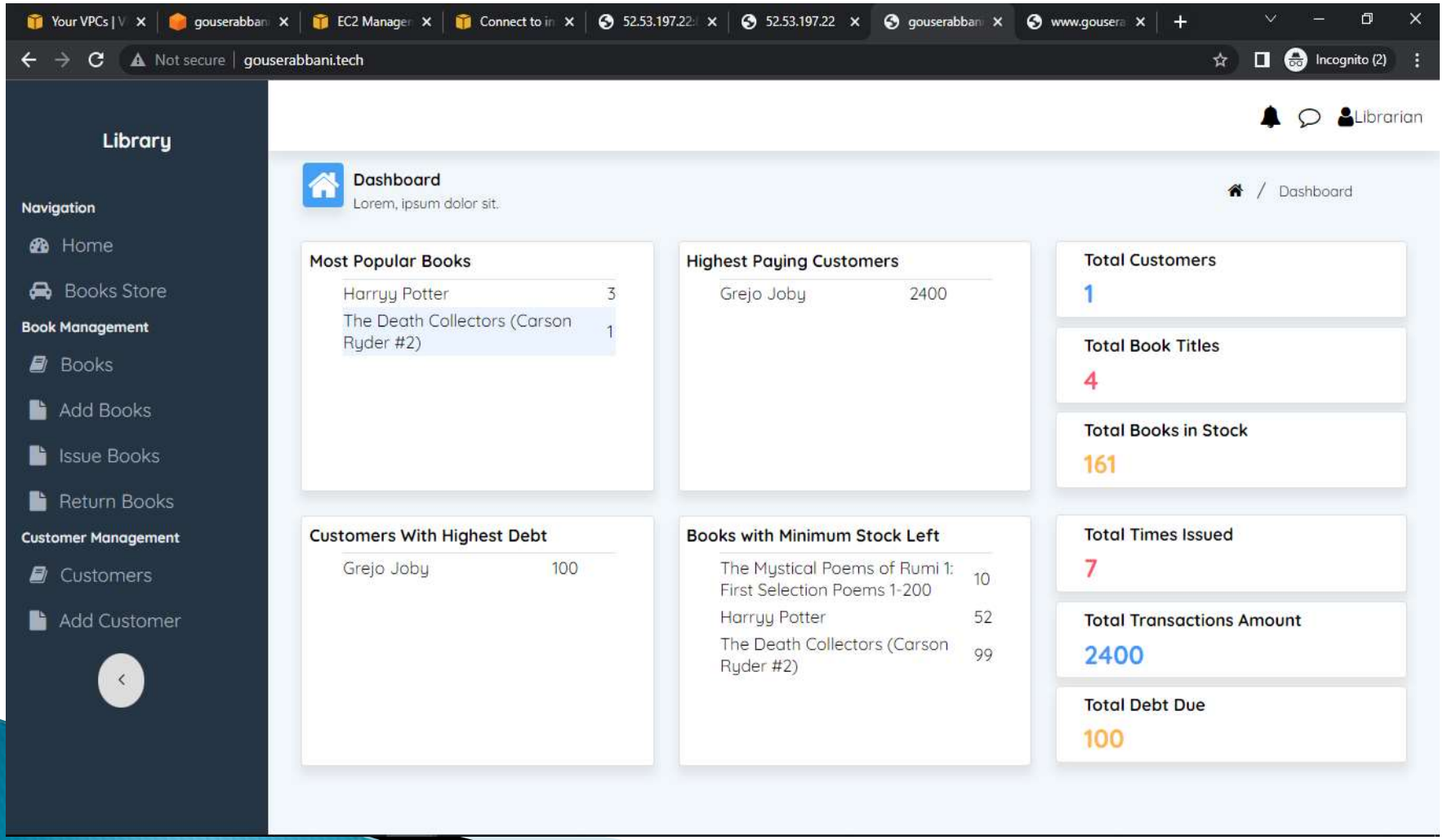
- Now , copy EC2 instance public IP and give port number and search in web browser.
- *IP:8080*
- We will get output like this



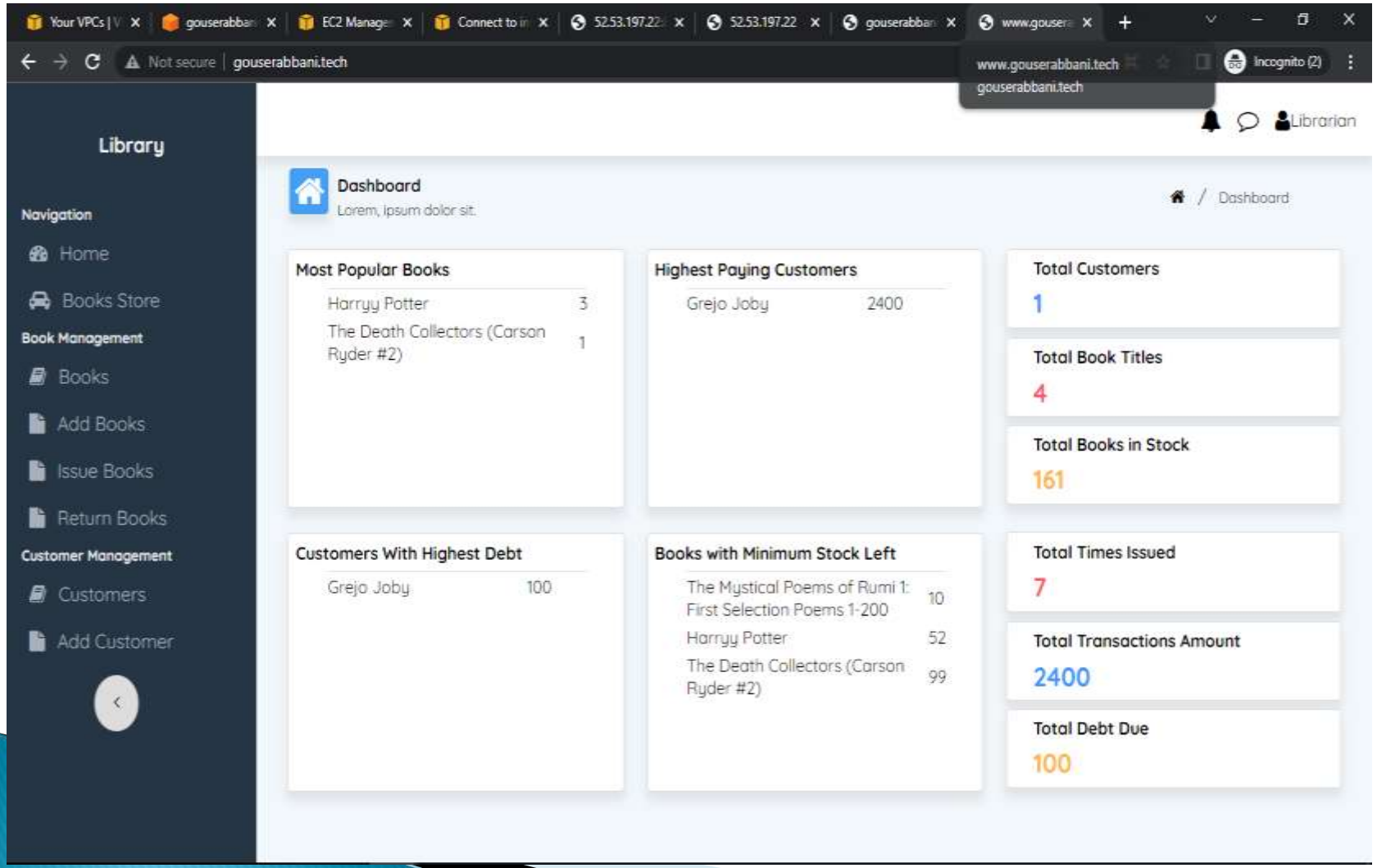
- After reverse proxy of nginx server I browse the server with ip4 address only



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



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



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

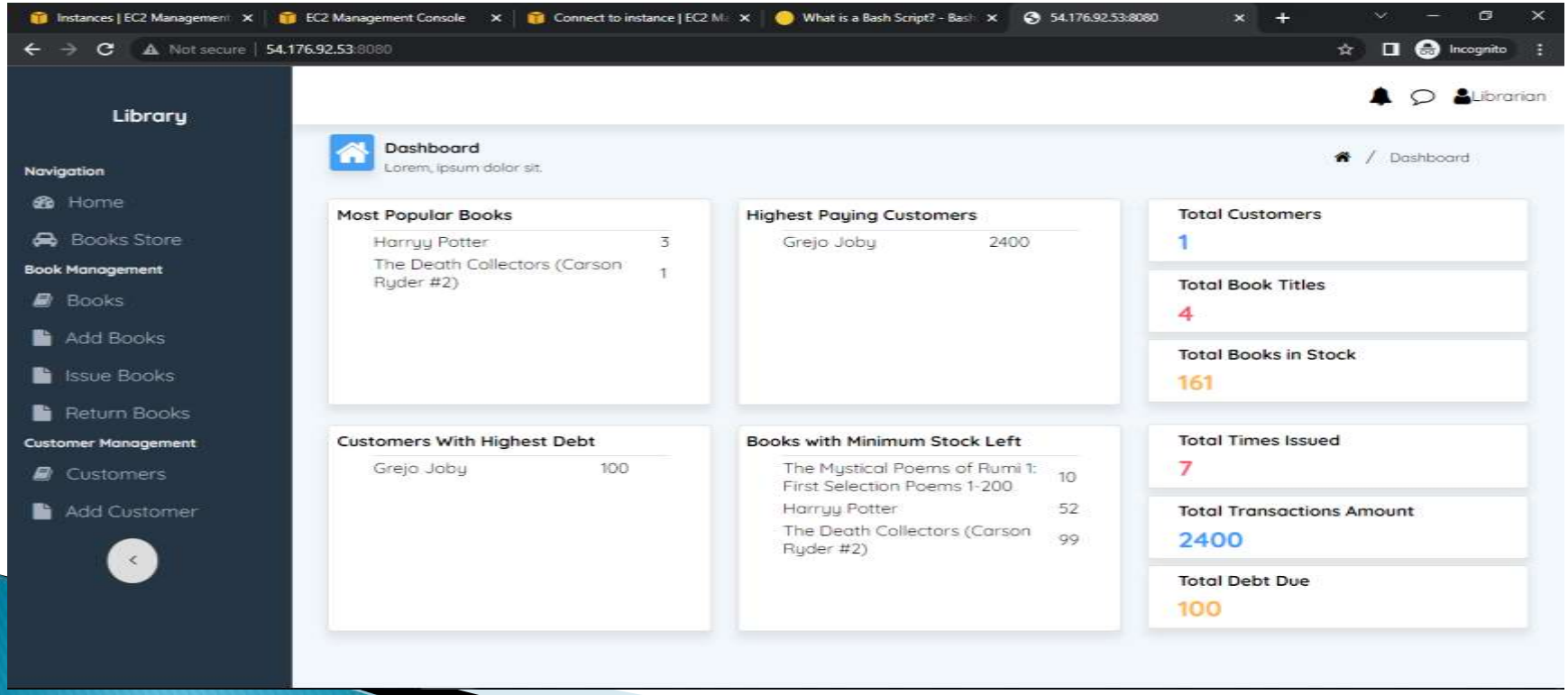
  - **UBUNTU**
    - #!/bin/bash
    - sudo apt update
    - sudo apt-get full-upgrade -y
    - sudo apt-get install python3-pip
    - git clone https://github.com/GOUSERABBANI44/flask-library-app.git

- 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 &

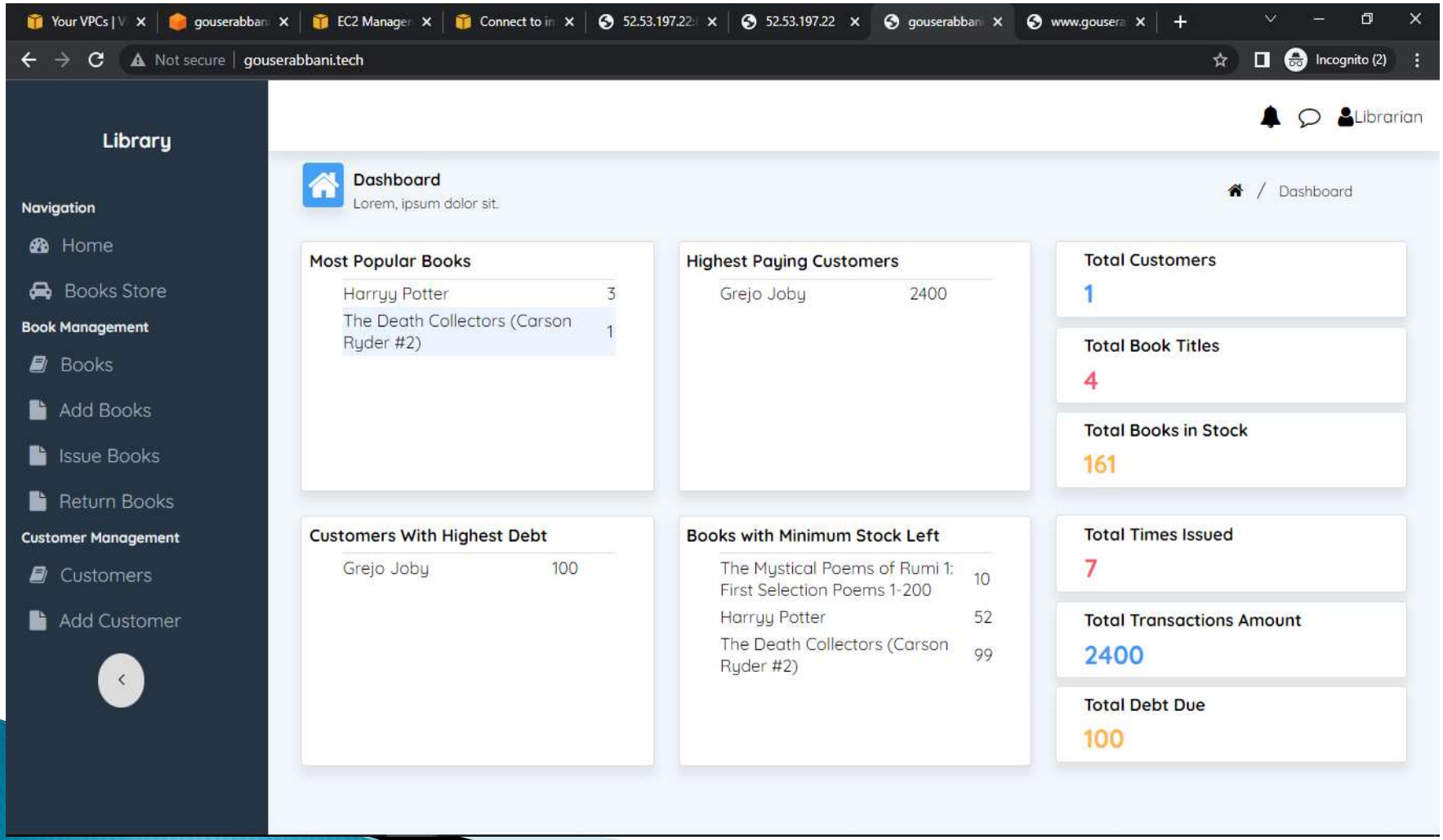
- **AMAZONLINUX2**
  - #!/bin/bash
  - sudo yum -y update
  - sudo yum -y install git
  - git clone https://github.com/GOUSERABBANI44/flask-library-app.git
  - sudo yum -y python3-pip
  - cd flask-library-app
  - pip3 install -r requirements.txt
  - nohup python3 -u ./app.py &

- Now launch the instance by using the SSH command with Git-bash or CMD

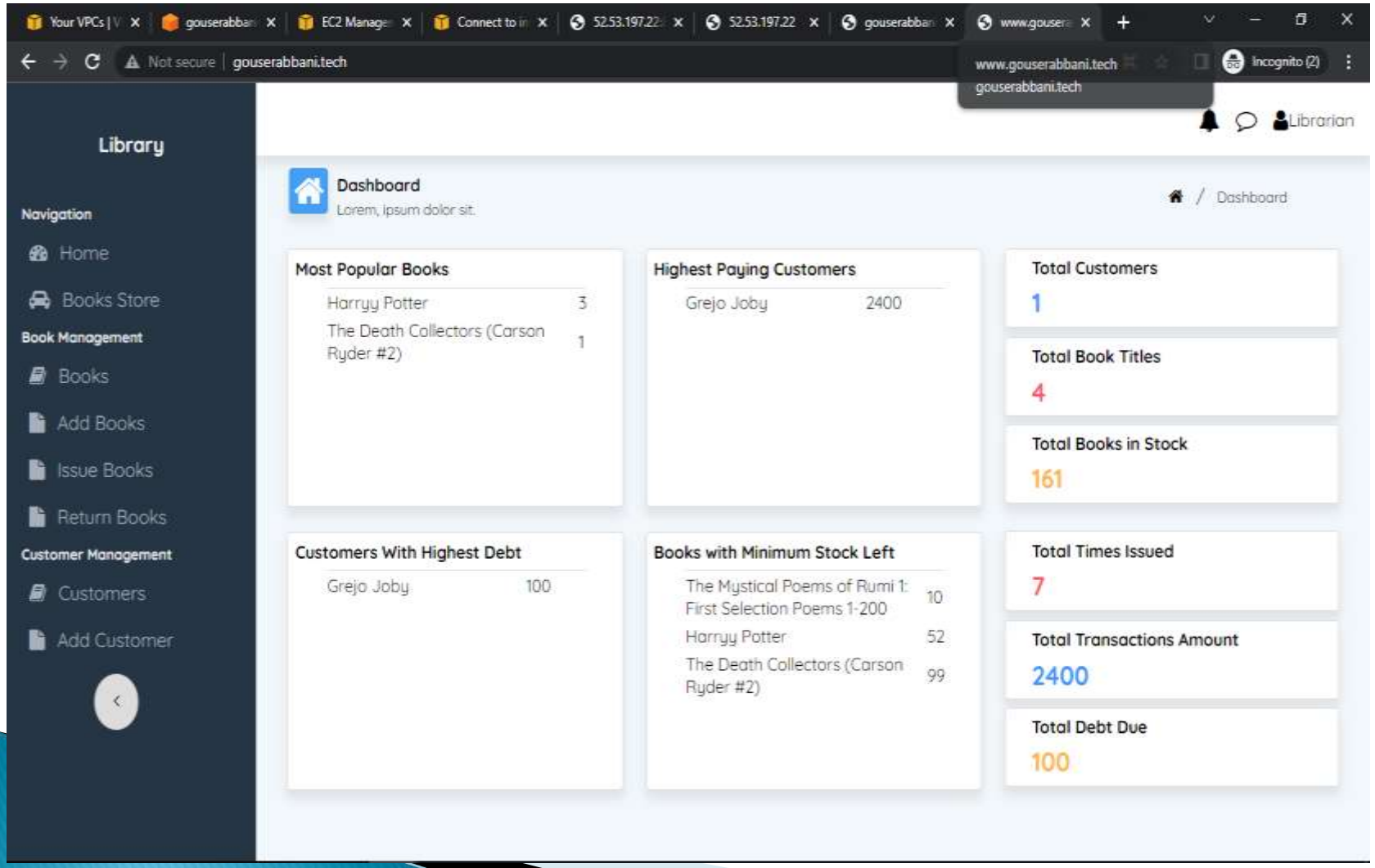
- After launching the instance just go with public ip of launched instance copy it and browse it along with port 8080.



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



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



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:

- pip3 install -r requirements.txt
- screen -m -d python3 app.py

  (or)
- nohup python3 -u ./app.py &

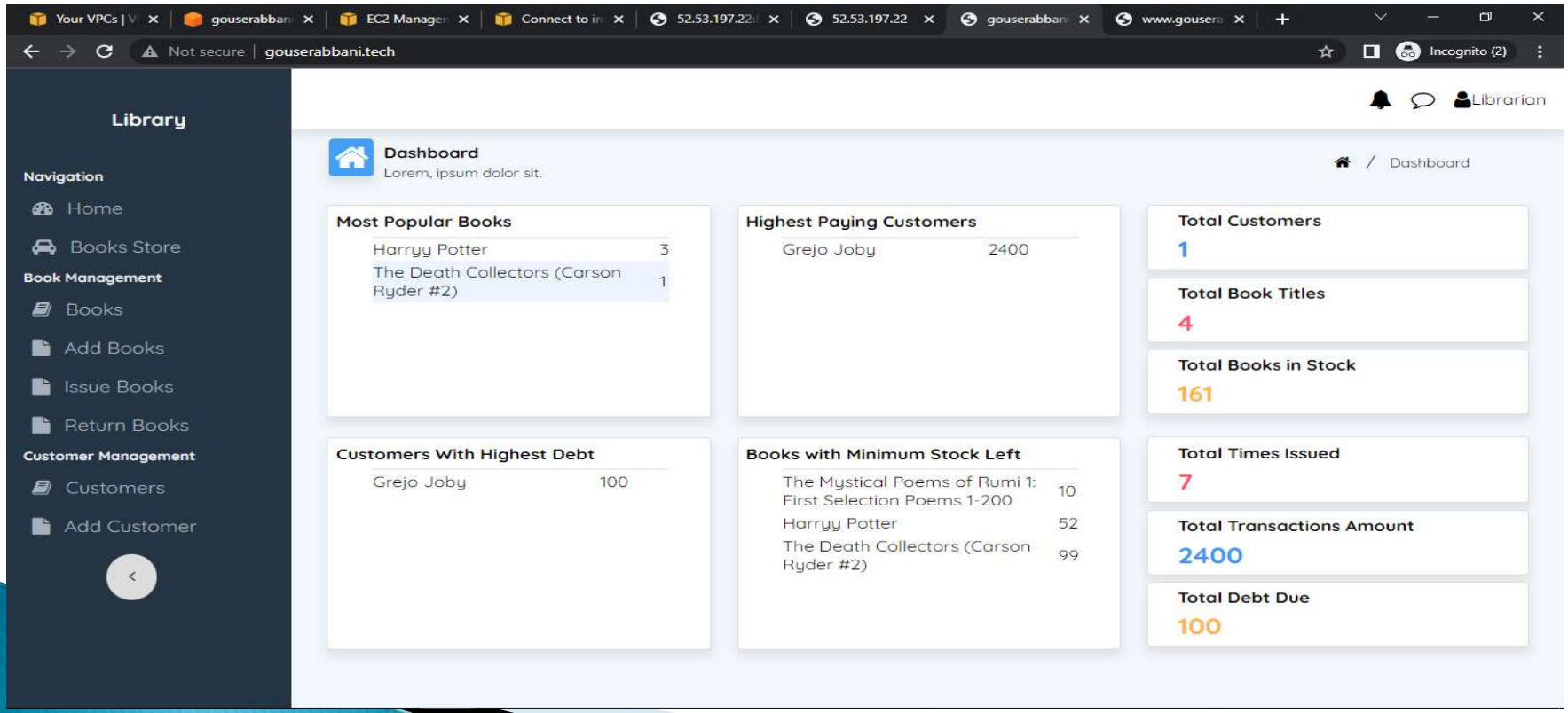
AMAZONLINUX2:

- pip3 install -r requirements.txt
- screen -m -d python3 app.py

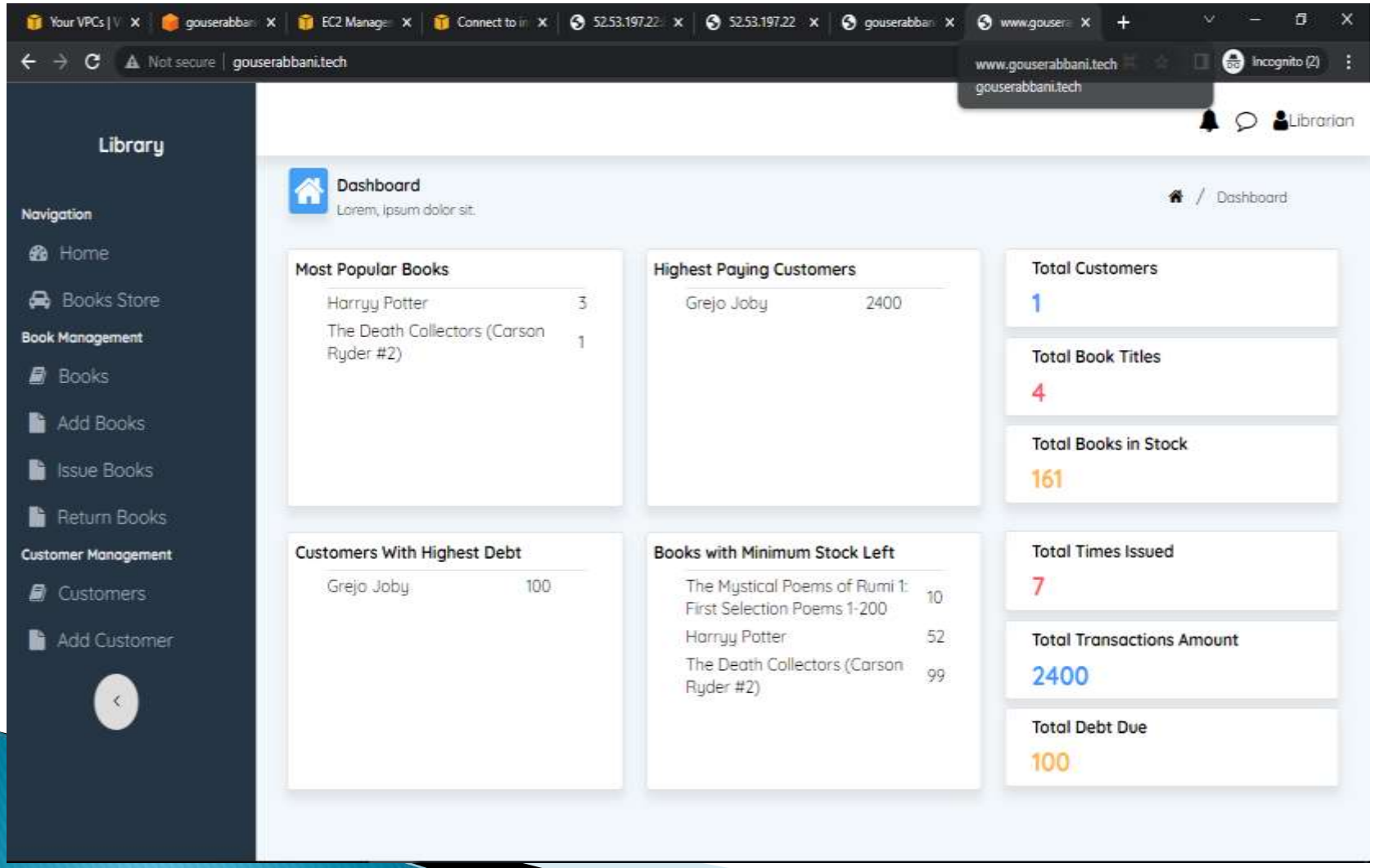
  (or)
- nohup python3 -u ./app.py &

- 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



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



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"
  }
}
```

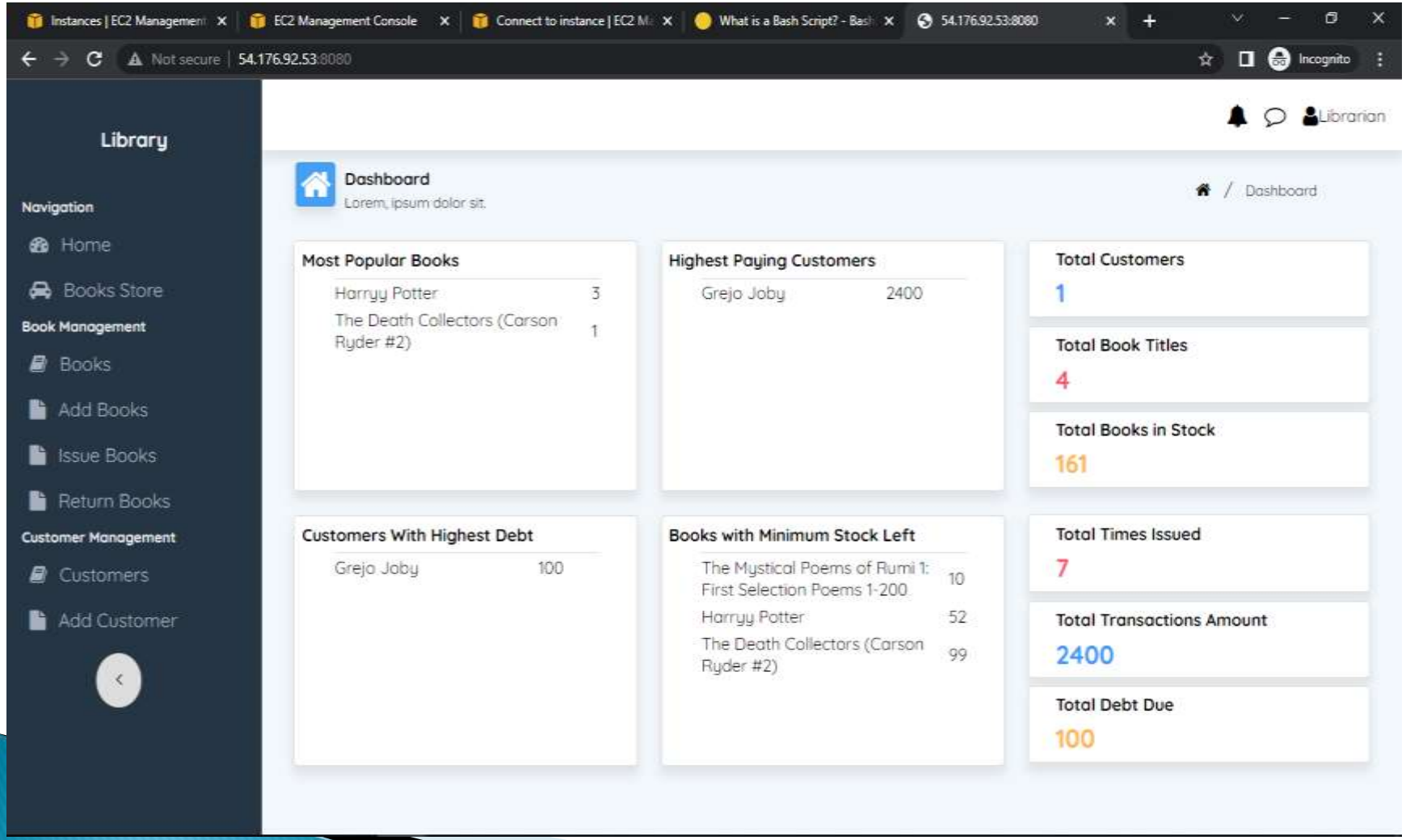
- Now create bash file for bash scripting by using vi command and the bash script For UBUNTU server

  - #!/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 &

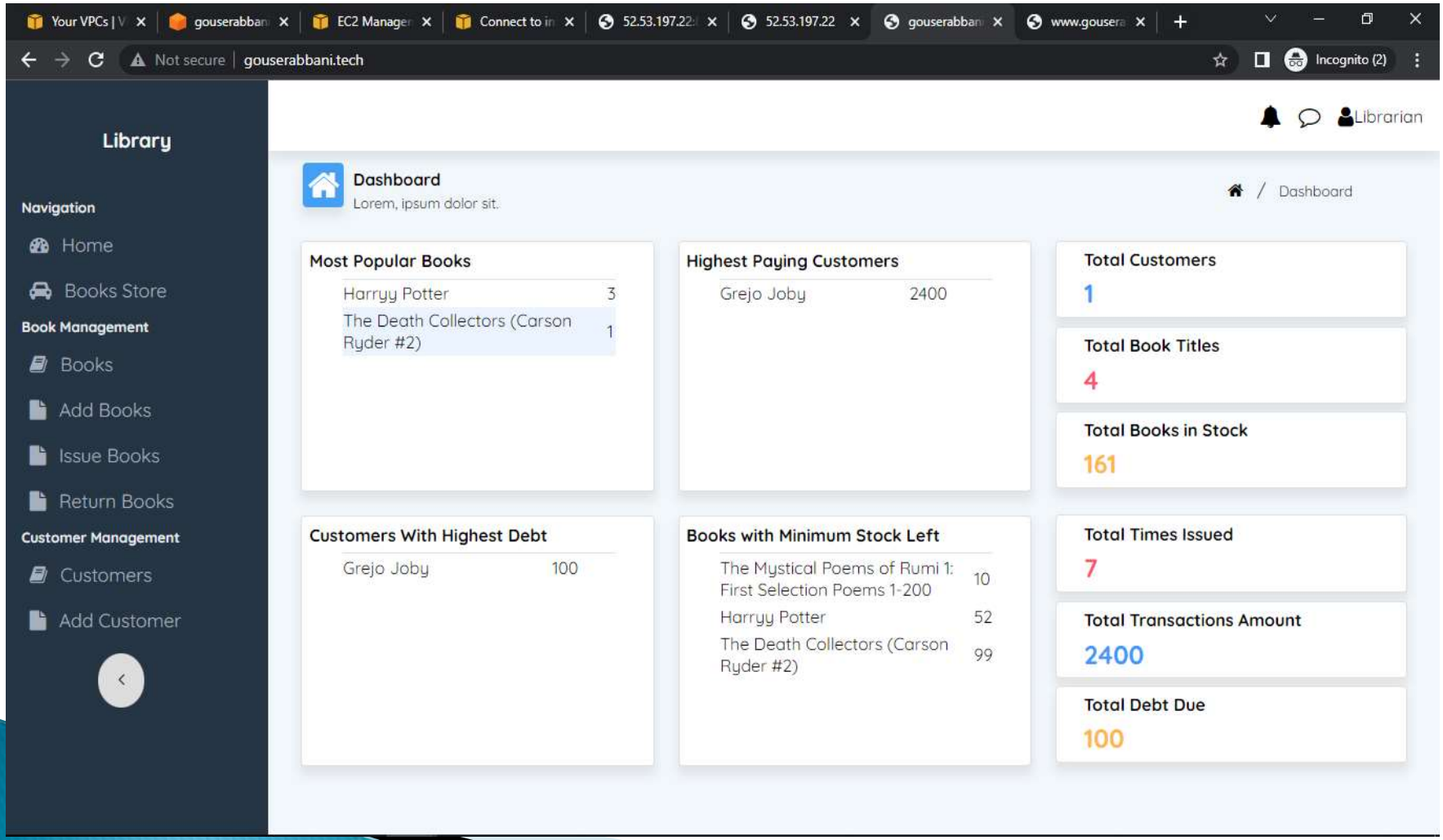
- Now create bash file for bash scripting by using vi command and the bash script For AMAZON-LINUX2 server

  - #!/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 &

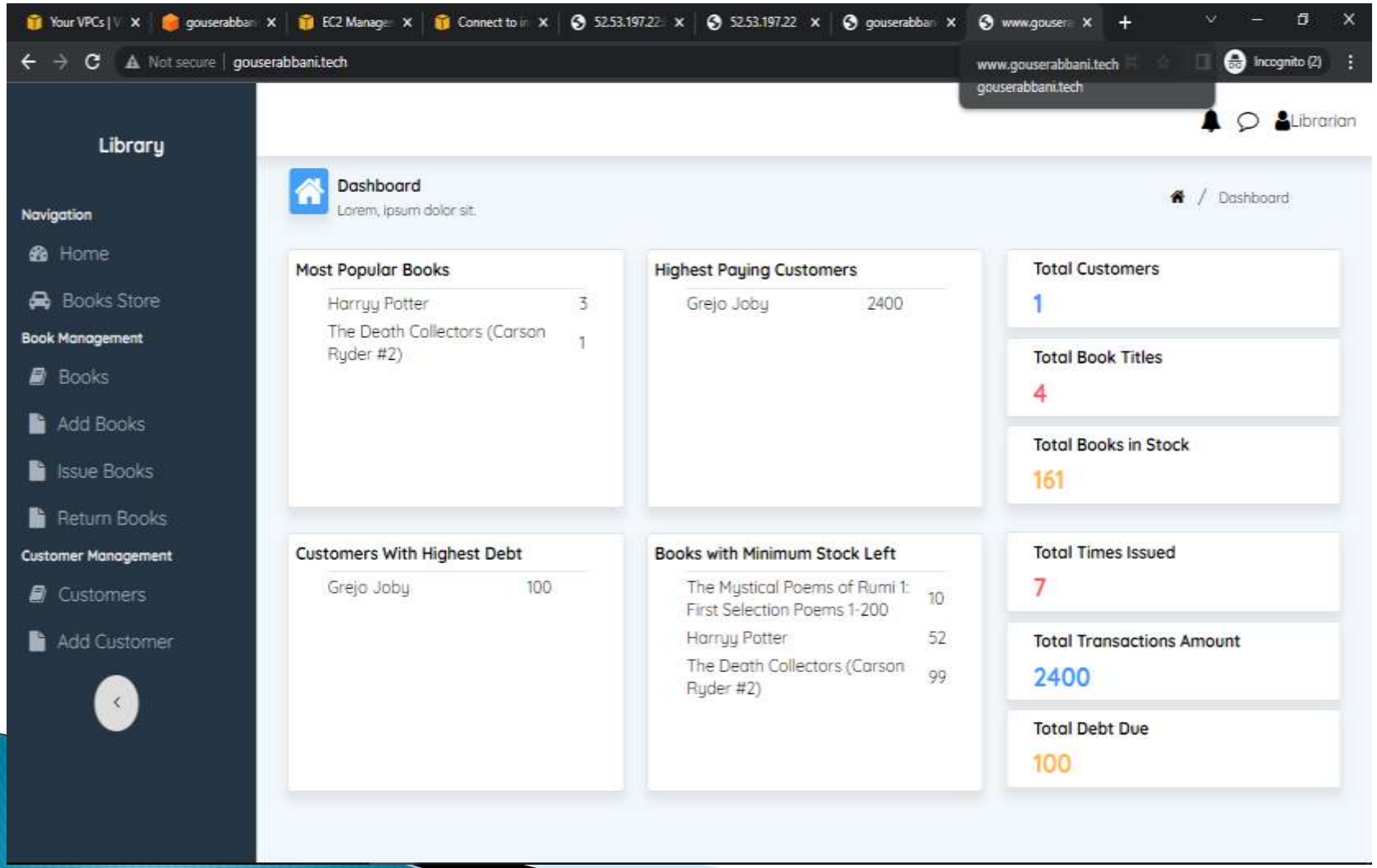
- Now go to copy public ip of created server (webapplication) and browse it along with port number –9000.



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



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



THANK YOU