Aim: Study of Platform as a Service

Theory:

Prepare a detailed study of Platform as a Service

1. What is PaaS ?

Platform as a service (PaaS) is a complete development and deployment environment in the cloud, with resources that enable you to deliver everything from simple cloud-based apps to sophisticated, cloud-enabled enterprise applications. You purchase the resources you need from a cloud service provider on a pay-as-you-go basis and access them over a secure Internet connection.Like IaaS, PaaS includes infrastructure—servers, storage and networking—but also middleware, development tools, business intelligence (BI) services, database management systems and more.

PaaS is designed to support the complete web application lifecycle: building, testing, deploying, managing and updating. PaaS allows you to avoid the expense and complexity of buying and managing software licenses, the underlying application infrastructure and middleware, container orchestrators such as Kubernetes or the development tools and other resources.

You manage the applications and services you develop and the cloud service provider typically manages everything else.

2. How to use PaaS (Customer) ?

Customers get many products to use from different PaaS providers. Few of the Key Features of PaaS that customers can use it as will be:

- Resources can be scaled up and down based on the requirements
- Multiple users can access the same application
- Allows for developing, testing and hosting apps in the same environment
- Web services and databases are integrated
- Teams can collaborate easily
- Provides a platform with tools to test, develop and host applications in the same environment
- Enables organizations to focus on development without having to worry about underlying infrastructure
- Providers manage security, operating systems, server software and backups
- Facilitates collaborative work even if teams work remotely

3. How to provide PaaS (Cloud Service Provider) ?

Platform-as-a-Service (PaaS) gives users with the platform and the environment for them to develop, manage, and run applications over the Internet. PaaS takes away or lessens the complexities of building, maintaining, and enhancing the cloud infrastructure that developers need in order to develop and launch an app. In other words, PaaS makes higher-level programming easy for web and software developers, and ultimately for businesses.

There are different kinds of PaaS providers. The below table shows some popular PaaS providers and services that are provided by them -

Providers	**Services**
Google App Engine (GAE)	App Identity, URL Fetch, Cloud storage client library, Logservice
Salesforce.com	Faster implementation, Rapid scalability, CRM Services, Sales cloud, Mobile connectivity, Chatter.
Microsoft Azure	Compute, security, IoT, Data Storage.
AppFog	Justcloud.com, SkyDrive, GoogleDocs
Openshift	RedHat, Microsoft Azure.
Cloud Foundry from VMware	Data, Messaging, and other services.

4. Advantages and Limitation of PaaS.

Advantages of PaaS

- By delivering infrastructure as a service, PaaS offers the same advantages as IaaS. But its additional features—middleware, development tools and other business tools—give you more advantages:
- Cut coding time. PaaS development tools can cut the time it takes to code new apps with pre-coded application components built into the platform, such as workflow, directory services, security features, search and so on.
- Add development capabilities without adding staff.
- Develop for multiple platforms—including mobile—more easily.
- Use sophisticated tools affordably. A pay-as-you-go model makes it possible for individuals or organisations to use sophisticated development software and business intelligence and analytics tools that they could not afford to purchase outright.

- Support geographically distributed development teams.
- Efficiently manage the application lifecycle.

Disadvantages of PaaS

- Dependency on Vendor : It is a great advantage that a certain part of work is done by the provider without you having to make an effort. On the other hand, your business will still be governed by the provider’s functional capabilities, speed and reliability. At the very least, you should perform your own data backup, for your peace of mind.

- Compatibility of Existing Infrastructure :A new platform is a new environment where legacy solutions are supposed to continue to work. Undoubtedly, some difficulties and contradictions may arise when two systems come into contact.

- Security Risks : As a rule, PaaS software is available in a public environment where multiple end users have access to the same basic resources. For some apps that contain sensitive data or have strict compliance requirements, this is not a good option since the data can be hacked.

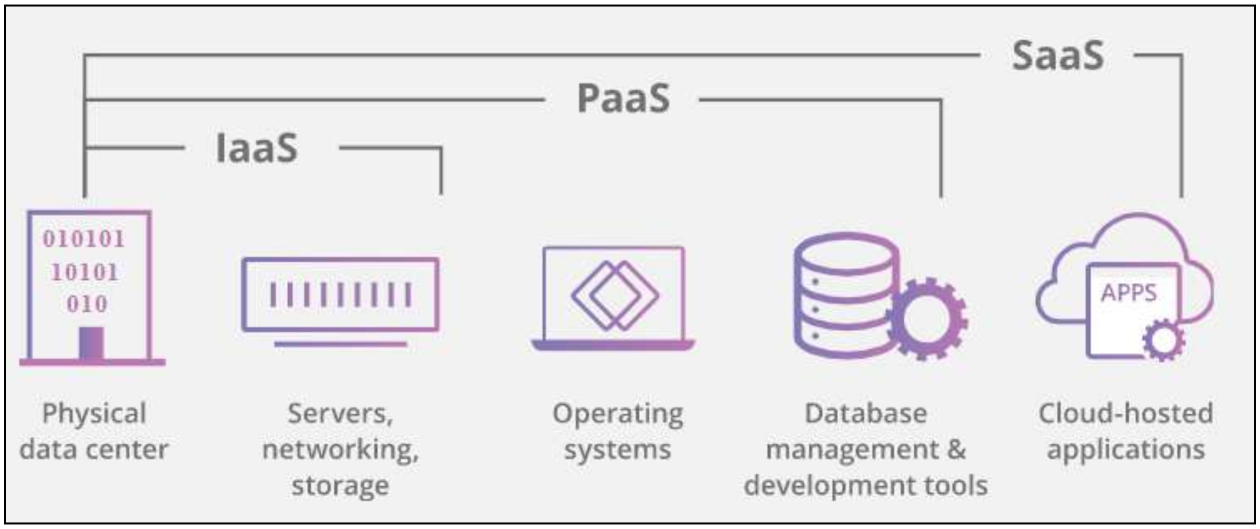
5. Study security issues in PaaS.

PaaS allows companies to build, run and ultimately manage Web applications without the infrastructure that is normally required.

Since PaaS is based on the notion of using shared resources (such as hardware, network, and security provisions), security concerns are usually focused on mission-critical information that hackers can obtain during a data breach. If the PaaS tenants have Administrator/’root’, or shell access to the servers running their instances, additional security issues could arise if hackers are able to gain unauthorized access and change configurations. Additionally, security controls and self-service entitlements offered by the PaaS platform could pose a problem if not properly configured. Providers should be able to provide clear policies, guidelines, and adhere to industry accepted best practices.

- Types of encryption used?
- Data independence and availability? (Can you move your virtual machines and all of their data to another provider? Who has access to it? What happens if a cloud instance migrates to another country?)
- Disaster recovery/business continuity protocols?
- Default application configurations
- SSL protocol and implementation flaws
- Insecure permissions on cloud data

6. Technologies used to provide PaaS.


SaaS
PaaS
IaaS
010101
10101
010
APPS
Physical data center
Servers, networking, storage
Operating systems
Database management & development tools
Cloud-hosted applications

PaaS includes multiple underlying cloud infrastructure components, including servers, networking equipment, operating systems, storage services, middleware, and databases.

Infrastructure : PaaS is the next layer up from IaaS in the cloud computing service model, and everything included in IaaS is also included in PaaS. A PaaS provider either manages servers, storage, and physical data centers, or purchases them from an IaaS provider.

Development tools : PaaS vendors offer a variety of tools that are necessary for software development, including a source code editor, a debugger, a compiler, and other essential tools. These tools may be offered together as a framework. The specific tools offered will depend on the vendor, but PaaS offerings should include everything a developer needs to build their application.

Middleware : Platforms offered as a service usually include middleware, so that developers don't have to build it themselves. Middleware is software that sits in between user-facing applications and the machine's operating system; for example, middleware is what allows software to access input from the keyboard and mouse. Middleware is necessary for running an application, but end users don't interact with it.

Operating systems : A PaaS vendor will provide and maintain the operating system that developers work on and the application runs on.

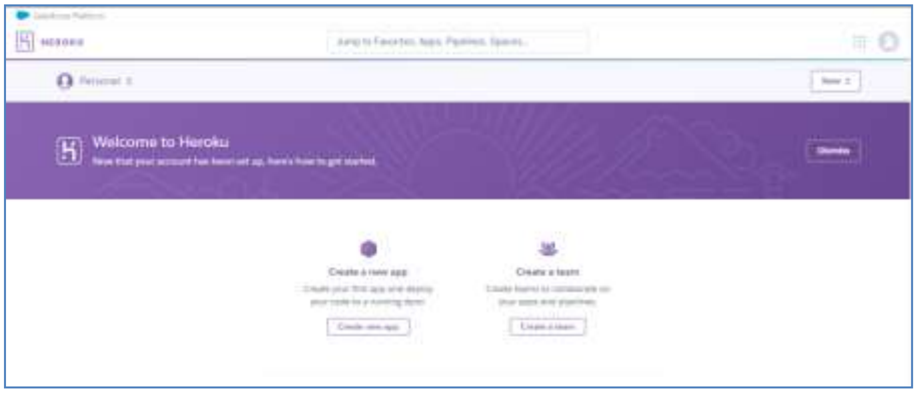
Databases : PaaS providers administer and maintain databases. They will usually provide developers with a database management system as well.

Activity

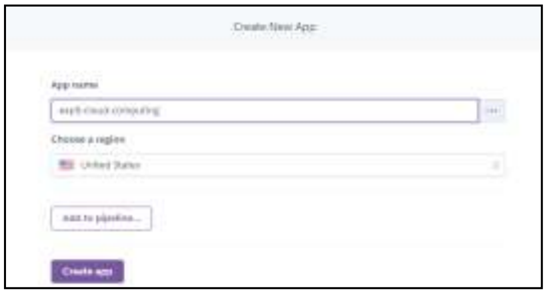
Use any suitable cloud service, providing a platform as a service. Deploy an application. Access it from a remote machine.

We will use Heroku as PaaS:

1. Login to Heroku and click create app

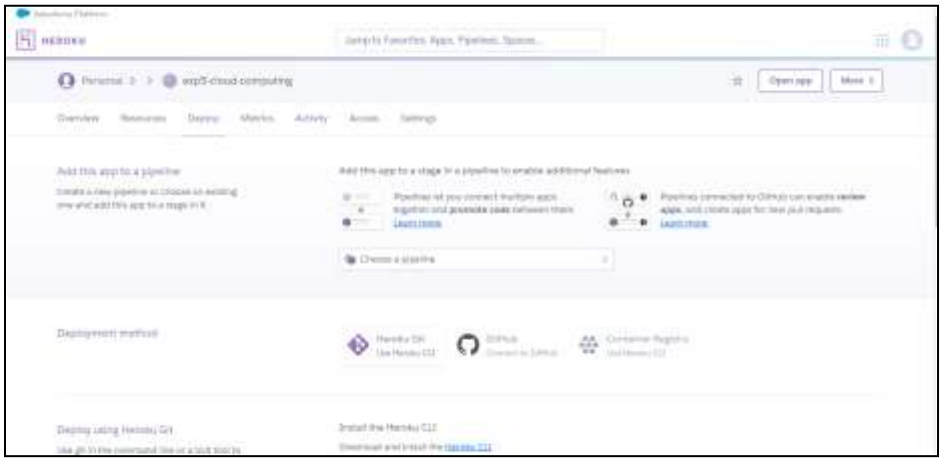

HEROKU
Jump to Favorites, Apps, Pipelines, Spaces...
Personal
New
Welcome to Heroku
Now that your account has been set up, here's how to get started.
Create a new app
Create new app
Create a team
Create a team

2. Enter App details

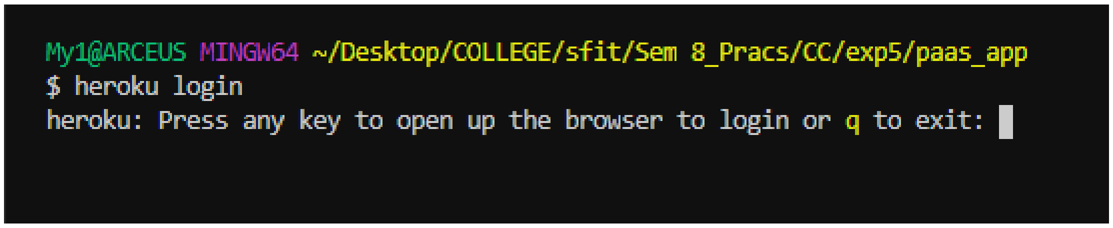


App name
exp5-cloud-computing
Choose a region
United States

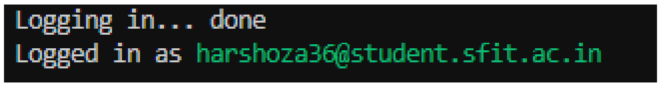
3. Finally we get app dashboard, which we can invoke using heroku cli.



4. Using heroku login in heroku cli to connect with the application.



```
My1@ARCEUS MINGW64 ~/Desktop/COLLEGE/sfit/Sem 8_Pracs/CC/exp5/paas_app
$ heroku login
heroku: Press any key to open up the browser to login or q to exit:
```

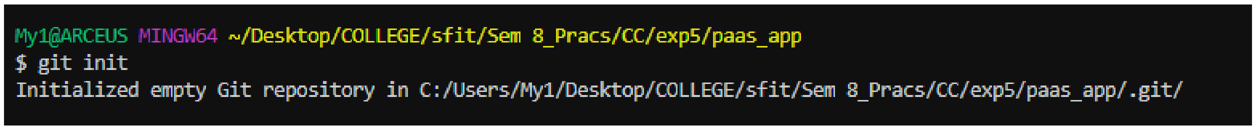


```
Logging in... done
Logged in as harshoza36@student.sfit.ac.in
```

5. Now we prepare to deploy our Python application using Git

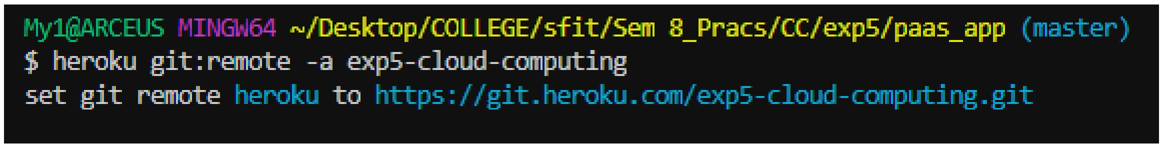
6. We need git cli downloaded and signed up with github account.

7. Now once git cli is setup we can use git with heroku cli.



```
My1@ARCEUS MINGW64 ~/Desktop/COLLEGE/sfit/Sem 8_Pracs/CC/exp5/paas_app
$ git init
Initialized empty Git repository in C:/Users/My1/Desktop/COLLEGE/sfit/Sem 8_Pracs/CC/exp5/paas_app/.git/
```

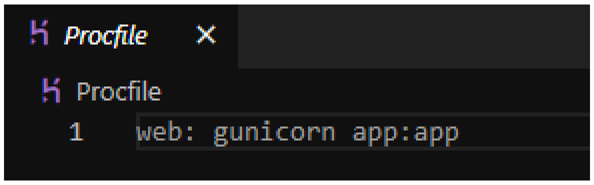
8. Now we use git remote with heroku and connect our app



```
My1@ARCEUS MINGW64 ~/Desktop/COLLEGE/sfit/Sem 8_Pracs/CC/exp5/paas_app (master)
$ heroku git:remote -a exp5-cloud-computing
set git remote heroku to https://git.heroku.com/exp5-cloud-computing.git
```

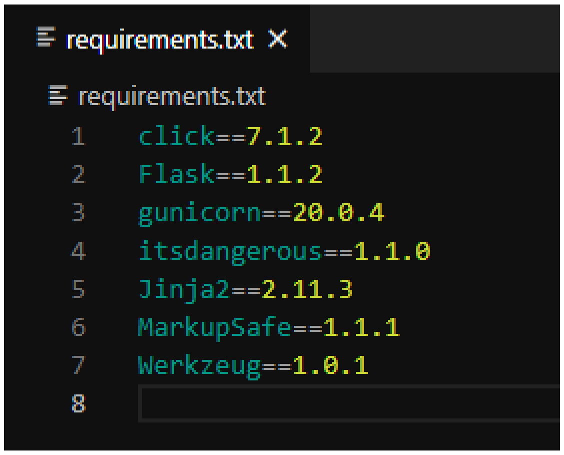
9. Now heroku is connected with our app. We need to add few files so heroku understands How to run app and what are the modules python requires :

- We create Procfile(case-sensitive) which helps heroku to find the app to run and specify the interface



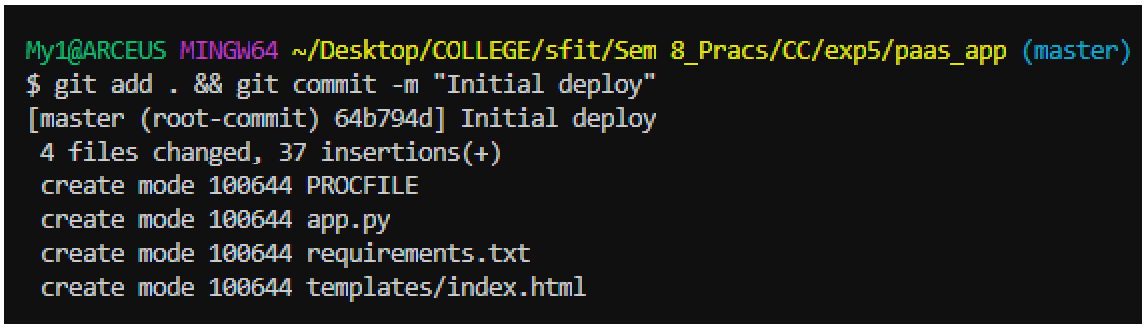
```
web: gunicorn app:app
```

- We create requirements.txt for python modules



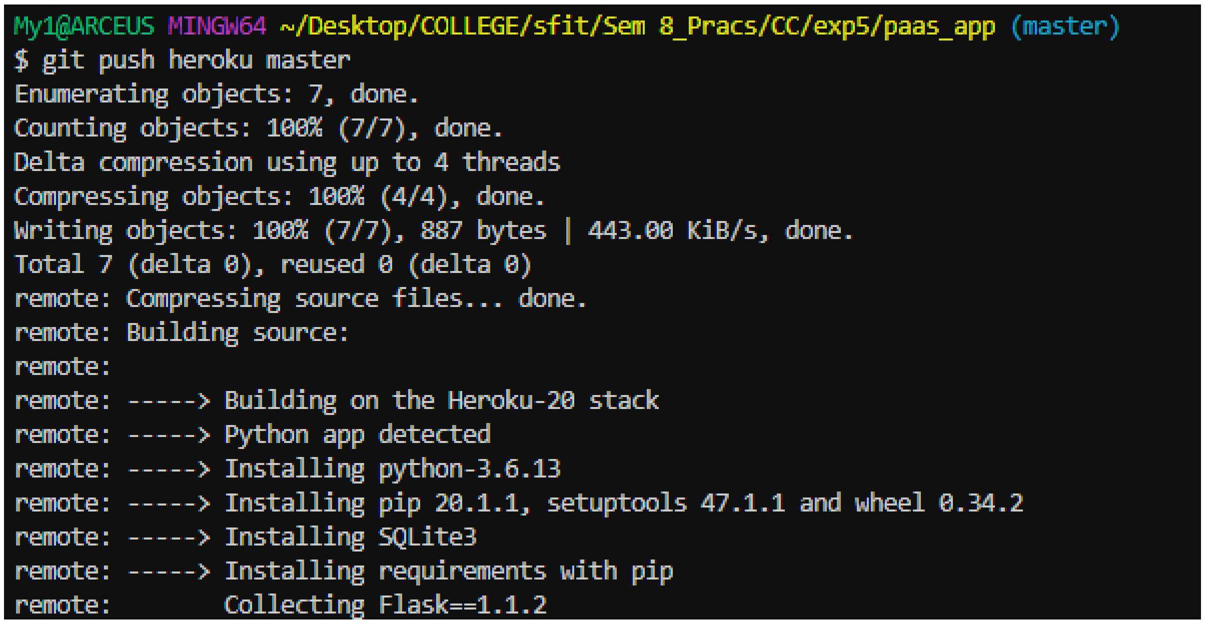
```
requirements.txt ×
requirements.txt
click==7.1.2
Flask==1.1.2
gunicorn==20.0.4
itsdangerous==1.1.0
Jinja2==2.11.3
MarkupSafe==1.1.1
Werkzeug==1.0.1
```

10. Now we use git add and git commit to Commit the code to github

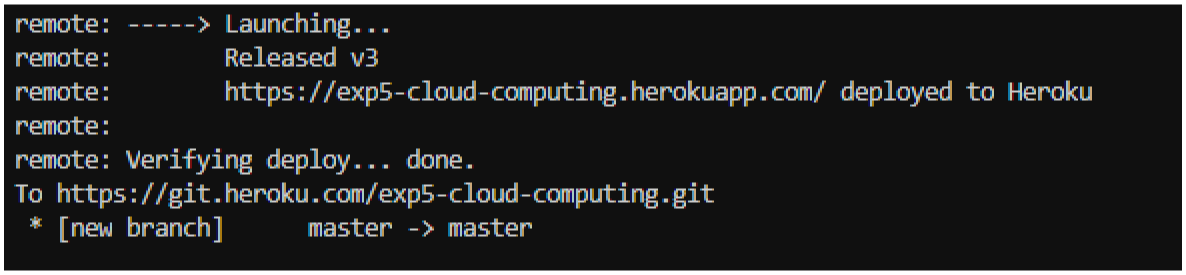


```
My1@ARCEUS MINGW64 ~/Desktop/COLLEGE/sfit/Sem 8_Pracs/CC/exp5/paas_app (master)
$ git add . && git commit -m "Initial deploy"
[master (root-commit) 64b794d] Initial deploy
 4 files changed, 37 insertions(+)
 create mode 100644 PROCFILE
 create mode 100644 app.py
 create mode 100644 requirements.txt
 create mode 100644 templates/index.html
```

11. Now we use git commit heroku master to deploy the application

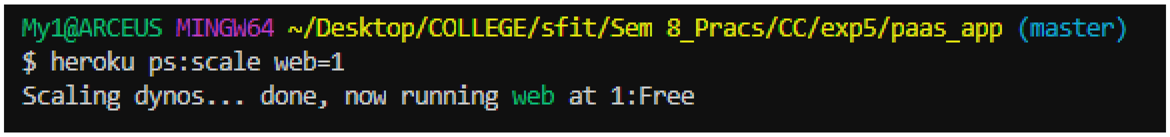


```
My1@ARCEUS MINGW64 ~/Desktop/COLLEGE/sfit/Sem 8_Pracs/CC/exp5/paas_app (master)
$ git push heroku master
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 4 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (7/7), 887 bytes | 443.00 KiB/s, done.
Total 7 (delta 0), reused 0 (delta 0)
remote: Compressing source files... done.
remote: Building source:
remote:
remote: -----> Building on the Heroku-20 stack
remote: -----> Python app detected
remote: -----> Installing python-3.6.13
remote: -----> Installing pip 20.1.1, setuptools 47.1.1 and wheel 0.34.2
remote: -----> Installing SQLite3
remote: -----> Installing requirements with pip
remote:        Collecting Flask==1.1.2
```



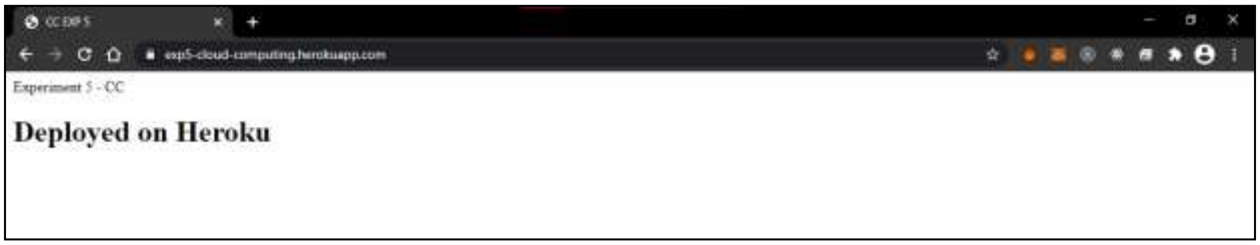
```
remote: -----> Launching...
remote:        Released v3
remote:        https://exp5-cloud-computing.herokuapp.com/ deployed to Heroku
remote:
remote: Verifying deploy... done.
To https://git.heroku.com/exp5-cloud-computing.git
 * [new branch]      master -> master
```

12. Now we can go to the website mentioned and see our website. But before running the website, we need to add dynos (heroku credits) to run it. Hence we use the command below.



```
My1@ARCEUS MINGW64 ~/Desktop/COLLEGE/sfit/Sem 8_Pracs/CC/exp5/paas_app (master)
$ heroku ps:scale web=1
Scaling dynos... done, now running web at 1:Free
```

13. On to the website URL : https://exp5-cloud-computing.herokuapp.com/

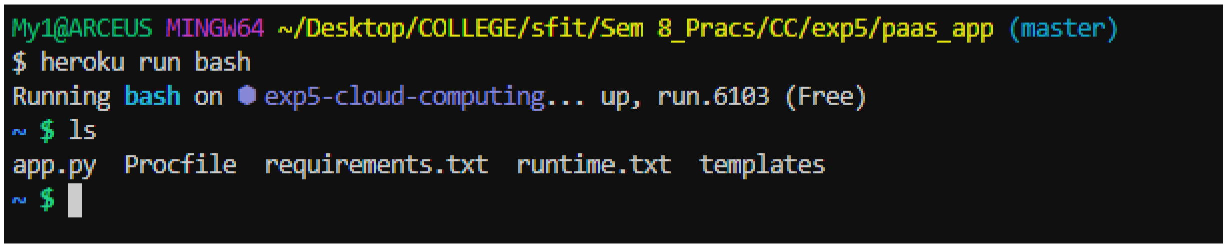


Experiment 5 - CC

Deployed on Heroku

14. To check for errors or any other information we use "heroku logs --tail".

15. Now to remotely access the application on heroku server we use the command below.



```
My1@ARCEUS MINGW64 ~/Desktop/COLLEGE/sfit/Sem 8_Pracs/CC/exp5/paas_app (master)
$ heroku run bash
Running bash on ⬢ exp5-cloud-computing... up, run.6103 (Free)
~ $ ls
app.py  Procfile  requirements.txt  runtime.txt  templates
~ $
```

Hence the app is deployed and we can access the app in the remote heroku machine. If you have multiple apps, and say you are in a different app folder connected with a different git repository you can use *heroku run bash -a APPNAME* to connect with your other application.

CONCLUSION :

- From this experiment, we learned about Platform as a service (PaaS). We understood what PaaS is , with its advantages and disadvantages, security concerns, how to use it, who provides it.
- Some benefits of PaaS are it can be used for development on multiple platforms and also by decreasing the staff required for it
- Using PaaS we can have multiple development tools which can help cut coding time for professionals for quick deployment.
- There are also some business tools which can be used with development tools that benefits an organization.
- Finally we Deployed an app to Heroku which is a PaaS and saw its procedure.

REFERENCES :

1. https://azure.microsoft.com/en-in/overview/what-is-paas/

2. https://en.wikipedia.org/wiki/Platform_as_a_service

3. https://www.sam-solutions.com/blog/advantages-and-disadvantages-of-paas-practical-overview/

4. https://www.arrow.com/ecs/na/channeladvisor/channel-advisor-articles/saas-paas-and-iaas-what-are-all-the-risks/

5. https://searchcloudcomputing.techtarget.com/tip/Top-threats-in-a-PaaS-cloud-service-and-how-to-avoid-them

6. https://stackify.com/top-paas-providers/

7. https://www.cloudflare.com/learning/serverless/glossary/platform-as-a-service-paas/

8. https://www.ibm.com/services/cloud/platform-as-a-service

9. https://www.quora.com/What-are-the-main-characteristics-of-PaaS