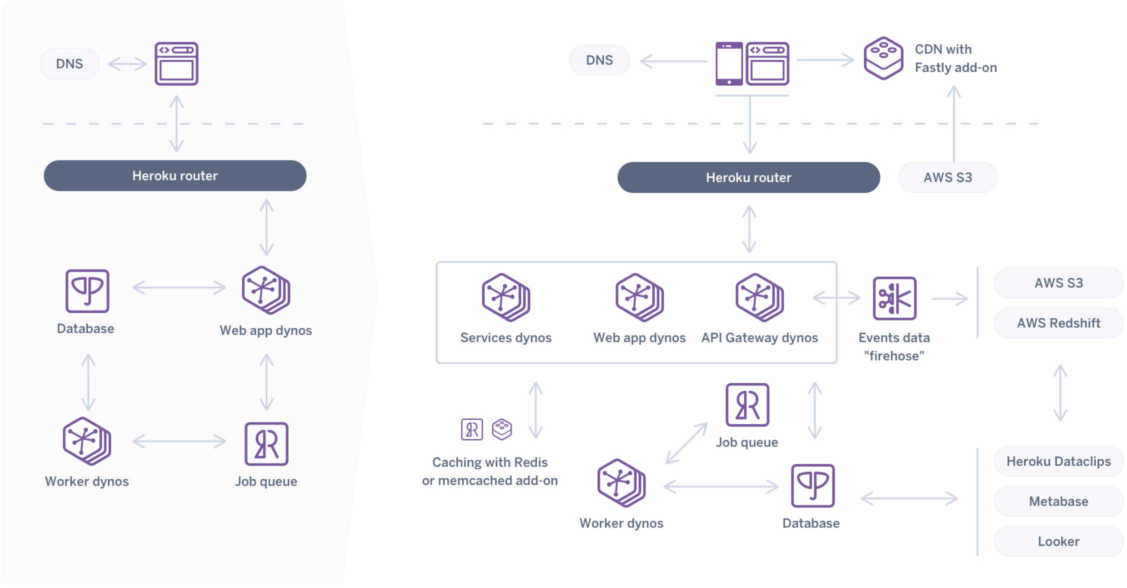
Group Project Milestone 4

Q1: How is your project architecture related to the theory taught in the lecture?

In our group project, we use Heroku base on PaaS architecture to build the Line Bot. According to the lecture Paas is a pre-defined “ready-to-use” environment comprised of already deployed and configured IT resources. Heroku is a cloud-based platform-as-a-service which based on a managed container system for building, running and managing modern apps.

The following chart is Heroku structure. Nigix in Heroku is used as the reverse proxy server. And Heroku’s HTTP cache adopts varnish. Heroku’s dynamic routing processing layer, which solves the problem of routing addressing, is a distributed routing pool for dynamic HTTP requests. And we deploy apps in dynamic grid layer. It also includes database and memory Cache in the architecture.



Heroku provides the following services:

(1) Development to build:

We submit source code by Heroku git and Heroku provides version management: we can submit code, create new version, modify deployment configuration, roll back old version, etc.

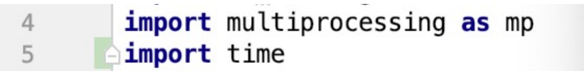
(2) Deploy to run:

Heroku starts a separate container for each project deployment, solves request routing and load balancing, and provides process management. We can expand and shrink capacity, view logs, monitor status, etc.

Q2: Can you demonstrate, with some screen cap, how to increase capacity of your chat bot service?

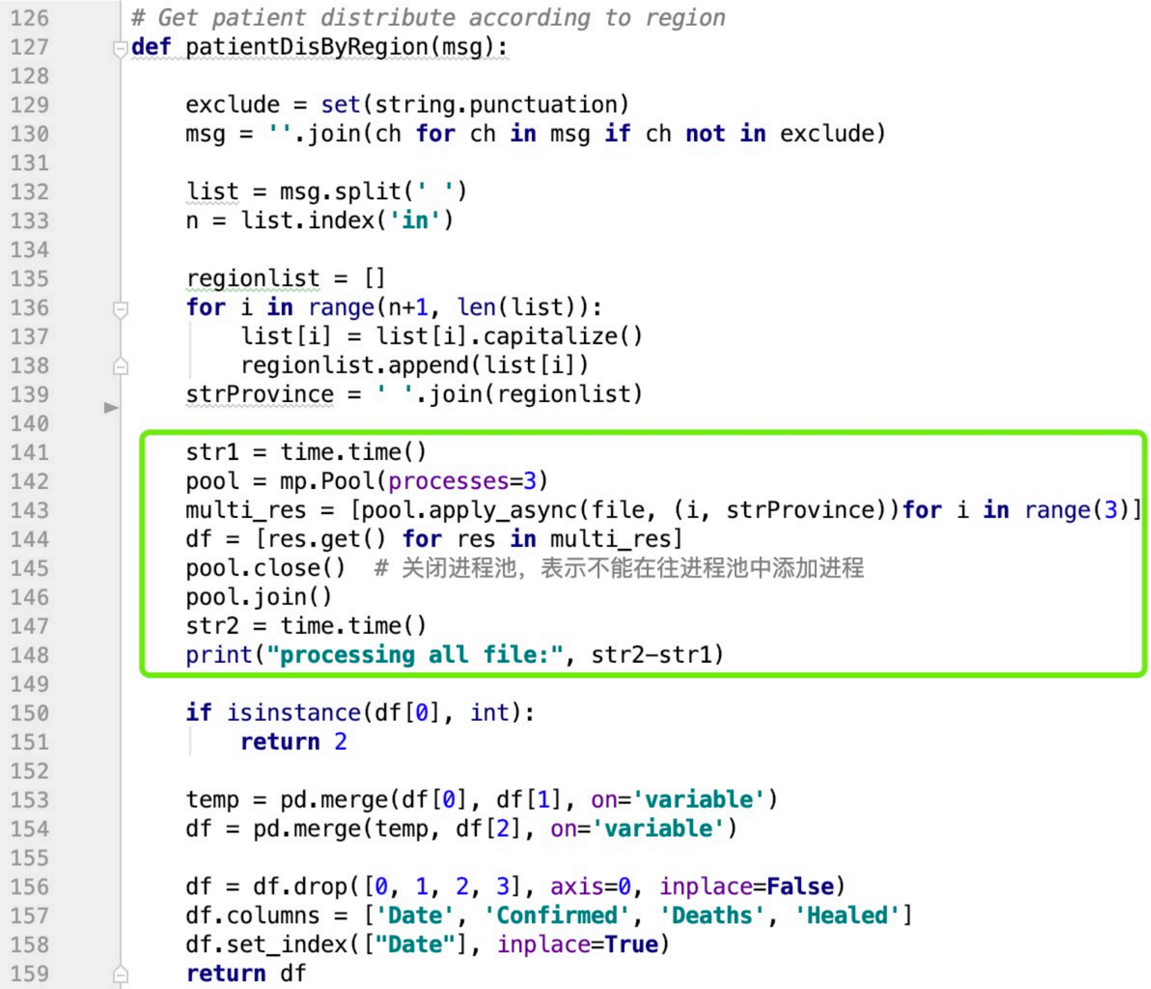
We add multi process processing in the code and speed up the operation speed when querying the distribution of pneumonia patient and increase capacity of our bot service. The code is as follows:

1. Import multiprocessing and time class



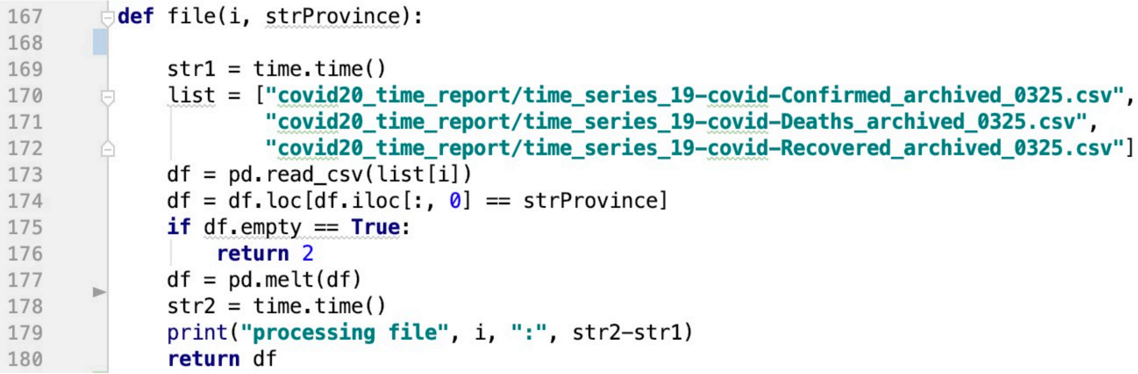
```
import multiprocessing as mp
import time
```

2. Here is function to query patient distribution. In the green box, we define a process pool first. Since we have three csv files to process, we define the number of CPU cores as 3. Then, we use apply_async() asynchronous method to make each process run a csv file processing function. Finally, use res.get() method to get all the result of file processing.



```python
# Get patient distribute according to region
def patientDisByRegion(msg):

    exclude = set(string.punctuation)
    msg = ''.join(ch for ch in msg if ch not in exclude)

    list = msg.split(' ')
    n = list.index('in')

    regionlist = []
    for i in range(n+1, len(list)):
        list[i] = list[i].capitalize()
        regionlist.append(list[i])
    strProvince = ' '.join(regionlist)

    str1 = time.time()
    pool = mp.Pool(processes=3)
    multi_res = [pool.apply_async(file, (i, strProvince))for i in range(3)]
    df = [res.get() for res in multi_res]
    pool.close()  # 关闭进程池，表示不能在往进程池中添加进程
    pool.join()
    str2 = time.time()
    print("processing all file:", str2-str1)

    if isinstance(df[0], int):
        return 2

    temp = pd.merge(df[0], df[1], on='variable')
    df = pd.merge(temp, df[2], on='variable')

    df = df.drop([0, 1, 2, 3], axis=0, inplace=False)
    df.columns = ['Date', 'Confirmed', 'Deaths', 'Healed']
    df.set_index(["Date"], inplace=True)
    return df
```

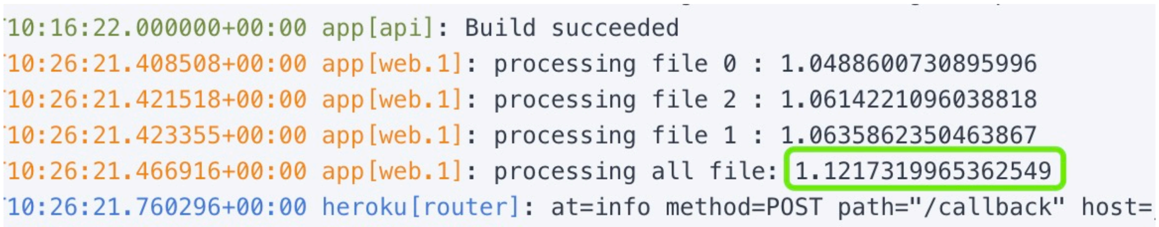
3. Here is function for file processing.



```python
def file(i, strProvince):

    str1 = time.time()
    list = ["covid20_time_report/time_series_19-covid-Confirmed_archived_0325.csv",
            "covid20_time_report/time_series_19-covid-Deaths_archived_0325.csv",
            "covid20_time_report/time_series_19-covid-Recovered_archived_0325.csv"]
    df = pd.read_csv(list[i])
    df = df.loc[df.iloc[:, 0] == strProvince]
    if df.empty == True:
        return 2
    df = pd.melt(df)
    str2 = time.time()
    print("processing file", i, ":", str2-str1)
    return df
```

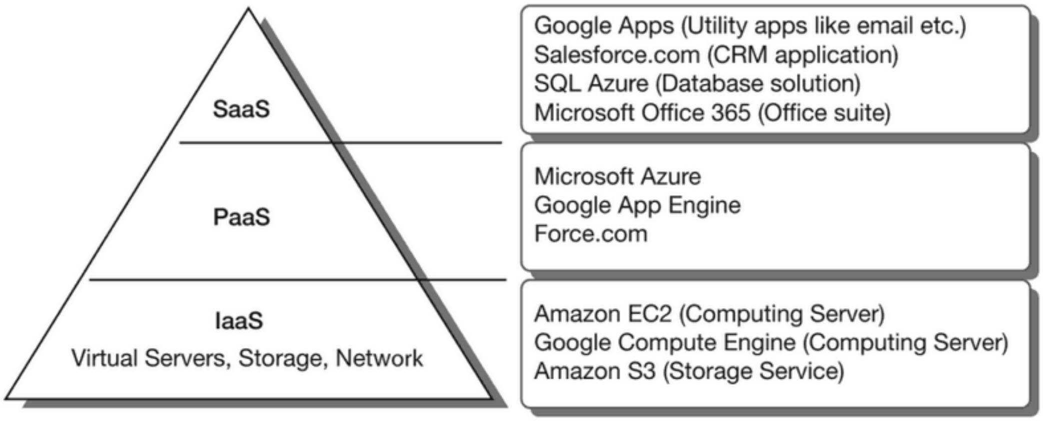
4. The following figure is view log of Heroku. As you can see from figure, three file processing methods took about 3 seconds in total. When using parallel

processes, it runs three methods which took about 1 second in total. It improves the efficiency of query and increases the capacity of service.



```
10:16:22.000000+00:00 app[api]: Build succeeded
10:26:21.408508+00:00 app[web.1]: processing file 0 : 1.0488600730895996
10:26:21.421518+00:00 app[web.1]: processing file 2 : 1.0614221096038818
10:26:21.423355+00:00 app[web.1]: processing file 1 : 1.0635862350463867
10:26:21.466916+00:00 app[web.1]: processing all file: 1.1217319965362549
10:26:21.760296+00:00 heroku[router]: at=info method=POST path="/callback" host=
```

Q3: Can you identify if you bot is one of the example of PaaS, IaaS, SaaS? Explain your answer.


SaaS
PaaS
IaaS
Virtual Servers, Storage, Network
Google Apps (Utility apps like email etc.)
Salesforce.com (CRM application)
SQL Azure (Database solution)
Microsoft Office 365 (Office suite)
Microsoft Azure
Google App Engine
Force.com
Amazon EC2 (Computing Server)
Google Compute Engine (Computing Server)
Amazon S3 (Storage Service)

Graph: The layered cloud service model

(1) IaaS, PaaS, and SaaS represent the three major models of cloud service.

IaaS (Infrastructure as a Service): IaaS is fully self-service for accessing and monitoring computers, networking, storage, and other services. An IaaS provider offers a pool of computation resources, in the form of physical machines or virtual machines, as well as other resources.

PaaS (Platform as a Service): it delivers development environment as a service, which typically includes the operating system, programming language execution environment, database, web server, etc.

SaaS (Software as a Service): SaaS utilizes the internet to deliver applications, which are managed by a third-party vendor, to its users.

(2) We use Baidu Map to call API which also is considered as a kind of SaaS. Baidu Maps is a software program based on the JavaScript and it is also a desktop and mobile web mapping service, offering satellite imagery, street maps. In this lab it enables to give a specific geographic location of hospitals to help some patients who infected coronavirus.

(3) In addition, Heroku is a cloud platform-as-a-service (PaaS) supporting several programming languages, based on managed container system for building, running, and managing modern apps. In this chat bot lab, we build LINE bot and communicate with it by using python and Heroku. We install Heroku CLI and push the code to Heroku Repo then hook our LINE bot with Heroku Service. For users, it is very convenient to get some useful information about COVID-19 or make a quick action of preventing infected.

(4) Finally, Redis and MySQL database applied in our robot also become the application of PaaS as it delivers development environment as a service, which typically includes the operating system, programming language execution environment, database, web server, etc.