Y20 - RELAY – LEARNATHON

(30th, 31st March & 1st April 2022)

Day 1 - 30th MAR 2022 (Complete ONLINE)

06:00AM - **Review 1 + Group Work**

Check Points for Review – 1

- Checking the status of the project completion till now.
- Front-end implementation [usage of bootstrap, static files, …]
- Models concept implemented [Yes/No]
- Hosting a static website using S3 and Cloud Front.

07:30AM – **Surprize Quiz -1**

08:30AM – **Session on Django and Postgresql to build backend connectivity - by Dr.V.Murali Mohan**

10:30AM - Group work commences

01:30PM - Coding test 1 Practice (https://icode.ccc.training)

03:00PM - Group work commences

04:30PM – **Surprize Quiz -2**

05:00PM - Review 2 + Group Work

Check Points for Review – 2

- Creations of Models
- Key Relationships [Primary key, Foreign key, etc]
- CRUD Operations on RDS [Using Cloud9 on AWS]
- RDS with MySQL workbench to establish

07:00PM – **Session on AWS Hosting & AWS Services - by RM Balajee**

09:30PM - Coding test 2 Practice (https://icode.ccc.training)

11:00PM - Group work commences

11:30PM – **Surprize Quiz -3**

01:30AM - Closure of Group Work for the day.

Day 2 - 31st MARCH 2022 (HYBRID)

09:15AM - Coding Test 3 Practice (https://icode.ccc.training)

11:00AM - Review 3 + Group work
Check Points for Review – 3

- Implementation of Sessions, Roles & permissions
- AWS Cognito to integrate with login/Registration API's
- AWS VPC Peer Connections

12:30PM – **Surprize Quiz -4**

01:30PM - **Session on Web Security - by Dr. G. Rama koteswara Rao**

03:30PM - Group Work

07:30PM - Coding Test 4 Practice (https://icode.ccc.training)

09:30PM – **Surprize Quiz -5**

10:30PM - preparation for Coding Exam

01:30AM - Closure of Preparation for coding Exam.

Day 3 - 1st APRIL 2022 (ON-CAMPUS)

09:15AM - Coding Test 5 Practice (https://icode.ccc.training)

11:00AM - Coding Grand Test (https://icode.ccc.training)

01:30PM - Group work for TS-SDP2

04:00PM - Final Review of Project + AWS Mini Projects
Check Points for Review – 4

- Percentage of project completion
- Web penetration testing in deployment
- Deployment in cloud along with database
- SDP2 Project hosting on AWS EC2 and RDS with [ASG + ELB].

Expectations with Respect to AWS Cloud

Static Web Hosting

1. Hosting Static Website with S3 bucket and Cloud Front

Dynamic Web Hosting

1. **Cloud RDS -** Connecting AWS RDS (MySQL) with workbentch and do CRUD Operations
2. **Cloud IDE -** Using Cloud9 AWS Cloud IDE for doing some Python Code to do CRUD Operations on Cloud RDS
3. **Cloud Authentication -** Implement Cognito service for authentication
4. **Cloud Environment -** Create your own VPC to launch AWS Services
5. **Cloud Full Stack Hosting -** Launch EC2 Service in created VPC to host Python Django based application which can interact with the above mentioned cloud RDS (MySQL)

Appendix

Django based hosting

Support Link 1: https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/create-deploy-python-django.html#python-django-setup-venv

Support Link 2: https://realpython.com/django-setup/

Commands in Terminal (Linux Based)

```
sudo yum update

sudo yum install python3

pip3 list

pip3 install virtualenv

pip3 install awsebcli

virtualenv virt

source ~/virt/bin/activate

// for deactivation – go to the folder where virtual file is created and then give “deactivate”

// to remove a directory “rm -r <directory_name>”

pip3 install django==2.1

//to verify

pip freeze

django-admin startproject proj1
```

//the above command will create the directory as below,

~/proj1

```
  |-- proj1
  |   |-- __init__.py
  |   |-- settings.py
  |   |-- urls.py
  |   `-- wsgi.py
  `-- manage.py
```

cd proj1

python manage.py startapp app1

it will create a following directory structure

```
proj1/
|
├── app1/
|   |
|   ├── migrations/
|   |   └── __init__.py
|   |
|   ├── __init__.py
|   ├── admin.py
|   ├── apps.py
|   ├── models.py
|   ├── tests.py
|   └── views.py
|
├── proj1/
|   ├── __init__.py
|   ├── asgi.py
|   ├── settings.py
|   ├── urls.py
|   └── wsgi.py
```

|

└── manage.py

go to settings.py in second “proj1” and add “app1” in the installed apps

go to settings.py in second “proj1” and add “ALLOWED_HOSTS=['35.154.168.20']” this is a public ip of the instance

Come out to first “proj1” folder and type

sudo nano runserver.py

//create the file “runserver.py” in parallel to manage.py and past the code below to change the port from 8000 to 8080

```
#!/bin/bash
exec ./manage.py runserver 0.0.0.0:8080
```

sudo chmod +x runserver.py

./runserver.py

// no need of this command “python manage.py runserver” due to above one

put “public_ip_address:8080” in browser

Now break the server and go to urls.py in your second proj1 directory and have these codes

Code

```
from django.contrib import admin
from django.urls import path, include
urlpatterns = [
    path('admin/', admin.site.urls),
    path('', include('app.urls')),
]
```

Install “filezilla client” and connect your EC2

Now go to “app1” directory and past and replace the docs (urls.py, views.py, test.py, models.py, forms.py, apps.py, admin.py)

Inside app directory, create a folder called “templates” and past the below files (display.html, index.html, page2.html, upload.html)

Now in putty terminal, go to “apps.py” under app1 directory and replace the text “app” to “app1”.

Now go to “views.py” in app1 directory using terminal and open it.

Change this line “con = sql.connect(host='localhost', user='root', password='root', database='mydb');” with your data of RDS.

Now go to “settings.py” under second proj1 folder and add replace these code,

Code

```
DATABASES = {
   'default': {
      'ENGINE': 'django.db.backends.mysql',
      #'ENGINE': 'django.db.backends.sqlite3',
      #'NAME': BASE_DIR / 'db.sqlite3',
      'HOST': 'localhost',
      'USER': 'root',
      'PASSWORD': 'root',
      'NAME': 'mydb',
   }
}
```

In the above code, provide your RDS details

Then go to and edit the “__init__.py” file in your project origin dir(the same as settings.py)

Add code:

```
import pymysql
pymysql.install_as_MySQLdb()
```

Now, go to your terminal and install the following

```
pip3 install mysql-client
pip3 install mysql-connector-python
pip3 install pymysql
```

now run application as “./runserver.py”

you can able to communicate with the database now by adding “/upload” and show it by adding “/display” in the url.