Note: In write-ups, you have to write only steps but at the of offline submission take printout of screenshots and attached as an output.

Assignment 3 (b):

Create Docker Container Environment (NVIDEIA Docker or any other).

Theory:

Docker Installation

Step1: View the cat /etc./*release* file to ensure your Linux os is compatible with latest Docker version

```
root1@sujata:~$ sudo cat /etc/*release*
[sudo] password for root1:
DISTRIB_ID=Ubuntu
DISTRIB_RELEASE=18.04
DISTRIB_CODENAME=bionic
DISTRIB_DESCRIPTION="Ubuntu 18.04.1 LTS"
NAME="Ubuntu"
VERSION="18.04.1 LTS (Bionic Beaver)"
ID=ubuntu
ID_LIKE=debian
PRETTY_NAME="Ubuntu 18.04.1 LTS"
VERSION_ID="18.04"
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
VERSION_CODENAME=bionic
UBUNTU_CODENAME=bionic
```

To install Docker Engine - Community, you need the 64-bit version of one of these Ubuntu versions:

- Disco 19.04
- Cosmic 18.10
- Bionic 18.04 (LTS)
- Xenial 16.04 (LTS)

Docker Engine - Community is supported on x86_64 (or amd64), armhf, arm64, s390x (IBM Z), and ppc64le (IBM Power) architectures.

```
root1@sujata:~$ lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 18.04.1 LTS
Release:        18.04
Codename:       bionic
```

$sudo apt-get update

```
root1@sujata:~$ sudo apt-get update
Ign:2 http://pkg.jenkins.io/debian-stable binary/ InRelease
Hit:1 https://packages.cloud.google.com/apt kubernetes-xenial InRelease
Hit:3 http://pkg.jenkins.io/debian-stable binary/ Release
Hit:5 http://in.archive.ubuntu.com/ubuntu bionic InRelease
Hit:6 http://in.archive.ubuntu.com/ubuntu bionic-updates InRelease
Hit:7 http://security.ubuntu.com/ubuntu bionic-security InRelease
Hit:8 http://in.archive.ubuntu.com/ubuntu bionic-backports InRelease
Reading package lists... Done
```

Step2: Uninstall old versions

Older versions of Docker were called Docker, docker.io, or docker-engine. If these are installed, uninstall them:

$ sudo apt-get remove docker docker-engine docker.io containerd runc

```
root1@sujata:~$ sudo apt-get remove docker docker-engine docker.io containerd runc
Reading package lists... Done
Building dependency tree
Reading state information... Done
Package 'docker' is not installed, so not removed
Package 'docker-engine' is not installed, so not removed
The following packages were automatically installed and are no longer required:
  bridge-utils cgroupfs-mount pigz ubuntu-fan
Use 'sudo apt autoremove' to remove them.
The following packages will be REMOVED:
  containerd docker.io runc
0 upgraded, 0 newly installed, 3 to remove and 462 not upgraded.
After this operation, 256 MB disk space will be freed.
Do you want to continue? [Y/n] y
(Reading database ... 139814 files and directories currently installed.)
Removing docker.io (18.09.7-0ubuntu1~18.04.4) ...
'/usr/share/docker.io/contrib/nuke-graph-directory.sh' -> '/var/lib/docker/nuke-graph-directory.sh'
Removing containerd (1.2.6-0ubuntu1~18.04.2) ...
Removing runc (1.0.0~rc7+git20190403.029124da-0ubuntu1~18.04.2) ...
Processing triggers for man-db (2.8.3-2) ...
```

Step3: Install using the convenience script

Docker provides convenience scripts at get.docker.com and test.docker.com for installing edge and testing versions of Docker Engine - Community into development environments quickly and non-interactively. The source code for the scripts is in the docker-install repository.

Using these scripts is not recommended for production environments, and you should understand the potential risks before you use them:

● The scripts require root or sudo privileges to run. Therefore, you should carefully examine and audit the scripts before running them.

● The scripts attempt to detect your Linux distribution and version and configure your package management system for you. In addition, the scripts do not allow

you to customize any installation parameters. This may lead to an unsupported configuration, either from Docker's point of view or from your own organization's guidelines and standards.

● The scripts install all dependencies and recommendations of the package manager without asking for confirmation. This may install a large number of packages, depending on the current configuration of your host machine.

- The script does not provide options to specify which version of Docker to install, and installs the latest version that is released in the "edge" channel. ● Do not use the convenience script if Docker has already been installed on the host machine using another mechanism.

Warning:

Always examine scripts downloaded from the internet before running them locally.

$ curl -fsSL https://get.docker.com -o get-docker.sh

$ sudo sh get-docker.sh

```
root1@sujata:~$ curl -fsSL https://get.docker.com -o get-docker.sh
root1@sujata:~$ sudo sh get-docker.sh
# Executing docker install script, commit: f45d7c11389849ff46a6b4d94e0dd1ffebca32c1
+ sh -c apt-get update -qq >/dev/null
+ sh -c DEBIAN_FRONTEND=noninteractive apt-get install -y -qq apt-transport-https ca-certificates curl >/dev/null
+ sh -c curl -fsSL "https://download.docker.com/linux/ubuntu/gpg" | apt-key add -qq - >/dev/null
Warning: apt-key output should not be parsed (stdout is not a terminal)
+ sh -c echo "deb [arch=amd64] https://download.docker.com/linux/ubuntu bionic stable" > /etc/apt/sources.list.d/docker.list
+ sh -c apt-get update -qq >/dev/null
+ [ -n  ]
+ sh -c apt-get install -y -qq --no-install-recommends docker-ce >/dev/null
+ sh -c docker version
Client: Docker Engine - Community
 Version:           19.03.5
 API version:       1.40
 Go version:        go1.12.12
 Git commit:        633a0ea838
 Built:             Wed Nov 13 07:29:52 2019
 OS/Arch:           linux/amd64
 Experimental:      false

Server: Docker Engine - Community
 Engine:
  Version:          19.03.5
  API version:      1.40 (minimum version 1.12)
  Go version:       go1.12.12
  Git commit:       633a0ea838
  Built:            Wed Nov 13 07:28:22 2019
  OS/Arch:          linux/amd64
  Experimental:     false
 containerd:
  Version:          1.2.10
  GitCommit:        b34a5c8af56e510852c35414db4c1f4fa6172339
 runc:
  Version:          1.0.0-rc8+dev
  GitCommit:        3e425f80a8c931f88e6d94a8c831b9d5aa481657
 docker-init:
  Version:          0.18.0
  GitCommit:        fec3683
If you would like to use Docker as a non-root user, you should now consider
adding your user to the "docker" group with something like:

  sudo usermod -aG docker your-user

Remember that you will have to log out and back in for this to take effect!

WARNING: Adding a user to the "docker" group will grant the ability to run
         containers which can be used to obtain root privileges on the
         docker host.
         Refer to https://docs.docker.com/engine/security/security/#docker-daemon-attack-surface
         for more information.
```

Step4: now check the version of docker installed.

#docker version or #docker -v

```
root1@sujata:~$ docker version
Client: Docker Engine - Community
 Version:           19.03.5
 API version:       1.40
 Go version:        go1.12.12
 Git commit:        633a0ea838
 Built:             Wed Nov 13 07:29:52 2019
 OS/Arch:           linux/amd64
 Experimental:      false
Got permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Get http://%2Fvar%2Frun%2Fdocker.sock/v1.40/version: dial unix
 /var/run/docker.sock: connect: permission denied
```

$sudo chmod 666 /var/run/docker.sock

```
root1@sujata:~$ sudo chmod 666 /var/run/docker.sock
root1@sujata:~$ docker version
Client: Docker Engine - Community
 Version:           19.03.5
 API version:       1.40
 Go version:        go1.12.12
 Git commit:        633a0ea838
 Built:             Wed Nov 13 07:29:52 2019
 OS/Arch:           linux/amd64
 Experimental:      false

Server: Docker Engine - Community
 Engine:
  Version:          19.03.5
  API version:      1.40 (minimum version 1.12)
  Go version:       go1.12.12
  Git commit:       633a0ea838
  Built:            Wed Nov 13 07:28:22 2019
  OS/Arch:          linux/amd64
  Experimental:     false
 containerd:
  Version:          1.2.10
  GitCommit:        b34a5c8af56e510852c35414db4c1f4fa6172339
 runc:
  Version:          1.0.0-rc8+dev
  GitCommit:        3e425f80a8c931f88e6d94a8c831b9d5aa481657
 docker-init:
  Version:          0.18.0
  GitCommit:        fec3683
```

Step5: try to launch a docker image for testing purpose, you can find the images in docker public repository at https://hub.docker.com

 # docker run docker/whalesay cowsay hello-sujata

some error occur

Error response from daemon: Get https://registry-1.docker.io/v2/

```
root1@sujata:~$ docker run docker/whalesay cowsay hello_sujata
docker: Got permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Post http://%2Fvar%2Frun%2Fdocker.sock/v1.40/containers/create: dial unix /var/run/docker.sock: connect: permission denied.
See 'docker run --help'.
```

$ sudo nano /etc/resolv.conf

#add these lines on top and above one for home router…

nameserver 8.8.8.8

nameserver 8.8.4.4

```
File  Edit  View  Search  Terminal  Help
  GNU nano 2.9.3                                              /etc/resolv.conf

# This file is managed by man:systemd-resolved(8). Do not edit.
#
# This is a dynamic resolv.conf file for connecting local clients to the
# internal DNS stub resolver of systemd-resolved. This file lists all
# configured search domains.
#
# Run "systemd-resolve --status" to see details about the uplink DNS servers
# currently in use.
#
# Third party programs must not access this file directly, but only through the
# symlink at /etc/resolv.conf. To manage man:resolv.conf(5) in a different way,
# replace this symlink by a static file or a different symlink.
#
# See man:systemd-resolved.service(8) for details about the supported modes of
# operation for /etc/resolv.conf.


nameserver 8.8.8.8
nameserver 8.8.4.4
nameserver 127.0.0.53
```

ubuntu@ubuntuserver:~$ sudo systemctl daemon-reload #THIS IS RESCUE COMMAND…

ubuntu@ubuntuserver:~$ sudo systemctl restart docker

```
root1@sujata:~$ sudo systemctl daemon-reload
root1@sujata:~$ sudo systemctl restart docker
```

ubuntu@ubuntuserver:~$ sudo systemctl status docker

```
root1@sujata:~$ sudo systemctl status docker
● docker.service - Docker Application Container Engine
   Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
   Active: active (running) since Sat 2020-02-08 21:33:50 IST; 24min ago
     Docs: https://docs.docker.com
 Main PID: 19594 (dockerd)
    Tasks: 16
   CGroup: /system.slice/docker.service
           └─19594 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock

Feb 08 21:34:45 sujata dockerd[19594]: time="2020-02-08T21:34:45.773886495+05:30" level=info msg="Attempting next endpoint for pull after error: Get https://registry-1.
Feb 08 21:34:45 sujata dockerd[19594]: time="2020-02-08T21:34:45.774077804+05:30" level=error msg="Handler for POST /v1.40/images/create returned error: Get https://reg
Feb 08 21:36:48 sujata dockerd[19594]: time="2020-02-08T21:36:48.858533862+05:30" level=info msg="Attempting next endpoint for pull after error: Get https://registry-1.
Feb 08 21:36:48 sujata dockerd[19594]: time="2020-02-08T21:36:48.858710011+05:30" level=error msg="Handler for POST /v1.40/images/create returned error: Get https://reg
Feb 08 21:44:31 sujata dockerd[19594]: time="2020-02-08T21:44:31.128028340+05:30" level=warning msg="Image docker.io/docker/whalesay:latest uses outdated schema1 manife
Feb 08 21:44:52 sujata dockerd[19594]: time="2020-02-08T21:44:52.597456615+05:30" level=error msg="Download failed, retrying: net/http: TLS handshake timeout"
Feb 08 21:45:46 sujata dockerd[19594]: time="2020-02-08T21:45:46.295246294+05:30" level=error msg="Download failed, retrying: net/http: TLS handshake timeout"
Feb 08 21:46:20 sujata dockerd[19594]: time="2020-02-08T21:46:20.263562527+05:30" level=error msg="Download failed, retrying: net/http: TLS handshake timeout"
Feb 08 21:52:54 sujata dockerd[19594]: time="2020-02-08T21:52:54.643734553+05:30" level=info msg="ignoring event" module=libcontainerd namespace=moby topic=/tasks/delet
Feb 08 21:52:54 sujata dockerd[19594]: time="2020-02-08T21:52:54.782837546+05:30" level=warning msg="ed9a8faf64cc35cacd773458e1a9979c0338ea75fb8b280d149a32eea198f29e cl
lines 1-19/19 (END)
```

#docker run docker/whalesay cowsay hello-sujata

Sometimes your machine can be behind a proxy network for which you will have to configure the proxy environment in the docker.

Steps to configure proxy in docker:

p /etc/systemd/system/docker.service.d #nano /etc/systemd/system/docker.service.d/http-proxy.conf

Environment="HTTP_PROXY=http://USER:PASSWD@SERVER:PORT/"

Environment="HTTPS_PROXY=http://USER:PASSWD@SERVER:PORT/"

#systemctl daemon-reload

#systemctl restart docker

Basic Docker Command

Step1: To Run the Container:

$sudo docker run <image-name> / <image-id>

```
root1@sujata:~/dockerfile-tutorial$ sudo docker images
REPOSITORY                                TAG                 IMAGE ID            CREATED             SIZE
raitnginxserver                           latest              1ac8603be6c7        21 hours ago        152MB
mynginxserver1                            latest              447e0693112e        23 hours ago        64.2MB
<none>                                    <none>              1166e35685f8        24 hours ago        247MB
sujata18061977/raitmodified_ubuntu        latest              6d5e32844787        38 hours ago        189MB
k8s.gcr.io/kube-proxy                     v1.17.2             cba2a99699bd        3 weeks ago         116MB
k8s.gcr.io/kube-apiserver                 v1.17.2             41ef50a5f06a        3 weeks ago         171MB
k8s.gcr.io/kube-controller-manager        v1.17.2             da5fd66c4068        3 weeks ago         161MB
k8s.gcr.io/kube-scheduler                 v1.17.2             f52d4c527ef2        3 weeks ago         94.4MB
ubuntu                                    latest              ccc6e87d482b        3 weeks ago         64.2MB
k8s.gcr.io/coredns                        1.6.5               70f311871ae1        3 months ago        41.6MB
k8s.gcr.io/etcd                           3.4.3-0             303ce5db0e90        3 months ago        288MB
quay.io/coreos/flannel                    v0.11.0-amd64       ff281650a721        12 months ago       52.6MB
k8s.gcr.io/pause                          3.1                 da86e6ba6ca1        2 years ago         742kB
docker/whalesay                           latest              6b362a9f73eb        4 years ago         247MB
root1@sujata:~/dockerfile-tutorial$ sudo docker run docker/whalesay
```

To Check the status of Docker Container:

$sudo docker ps

```
root1@sujata:~/dockerfile-tutorial$ sudo docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
```

To List all Container are Running / Exited:

$sudo docker ps -a

```
root1@sujata:~/dockerfile-tutorial$ sudo docker ps -a
CONTAINER ID        IMAGE                     COMMAND                  CREATED             STATUS                      PORTS               NAMES
34c53b419270        docker/whalesay           "/bin/bash"              45 seconds ago      Exited (0) 44 seconds ago                       vigilant_clarke
f4d051c48a74        raitnginxserver           "nginx -g 'daemon of…"   21 hours ago        Exited (0) 21 hours ago                         flamboyant_burnell
08f8f6add9fb        447e0693112e              "/bin/sh -c 'apt-get…"   21 hours ago        Exited (100) 21 hours ago                       sharp_booth
5ed051cd386d        447e0693112e              "/bin/sh -c 'apt ins…"   22 hours ago        Exited (100) 22 hours ago                       stupefied_lehmann
79ed3a4c5c8d        447e0693112e              "/bin/sh -c 'apt ins…"   22 hours ago        Exited (100) 22 hours ago                       quizzical_edison
3c4dbd17de2a        447e0693112e              "/bin/sh -c 'apt ins…"   22 hours ago        Exited (100) 22 hours ago                       determined_engelba
rt
3e9d1f18ac47        447e0693112e              "/bin/sh -c 'apt ins…"   23 hours ago        Exited (100) 23 hours ago                       lucid_cerf
b56d5bf2e4fb        447e0693112e              "/bin/sh -c 'apt ins…"   23 hours ago        Exited (100) 23 hours ago                       hungry_ride
030ee1976b5e        447e0693112e              "/bin/sh -c 'apt ins…"   23 hours ago        Exited (100) 23 hours ago                       romantic_chandrase
khar
18edc88f96fe        447e0693112e              "/bin/sh -c 'apt ins…"   23 hours ago        Exited (100) 23 hours ago                       sharp_meitner
e8cb0780bc0d        447e0693112e              "/bin/sh -c 'apt ins…"   23 hours ago        Exited (100) 23 hours ago                       cocky_chaplygin
b78a1d471f4e        447e0693112e              "/bin/sh -c 'apt ins…"   23 hours ago        Exited (100) 23 hours ago                       funny_greider
99f59fc1bd22        447e0693112e              "/bin/sh -c 'apt ins…"   23 hours ago        Exited (100) 23 hours ago                       gallant_matsumoto
395286abc3f8        447e0693112e              "/bin/sh -c 'apt ins…"   23 hours ago        Exited (100) 23 hours ago                       determined_kirch
33de9c293991        447e0693112e              "/bin/sh -c 'apt ins…"   23 hours ago        Exited (100) 23 hours ago                       nervous_goldberg
106dde3b0c0f        447e0693112e              "/bin/sh -c 'apt ins…"   23 hours ago        Exited (100) 23 hours ago                       brave_hamilton
7a8e44049421        447e0693112e              "/bin/sh -c 'apt ins…"   23 hours ago        Exited (100) 23 hours ago                       boring_clarke
```

To Start the Container and check the status:

sudo docker start flamboyant_burnell

sudo docker ps -a

```
root1@sujata:~/dockerfile-tutorial$ sudo docker start flamboyant_burnell
flamboyant_burnell
root1@sujata:~/dockerfile-tutorial$ sudo docker ps -a
CONTAINER ID        IMAGE                  COMMAND                  CREATED             STATUS                      PORTS                    NAMES
34c53b419270        docker/whalesay        "/bin/bash"              2 minutes ago       Exited (0) 2 minutes ago                             vigilant_clarke
f4d051c48a74        raitnginxserver        "nginx -g 'daemon of…"   21 hours ago        Up 7 seconds                0.0.0.0:8888->80/tcp     flamboyant_burn
ell
08f8f6add9fb        447e0693112e           "/bin/sh -c 'apt-get…"   21 hours ago        Exited (100) 21 hours ago                            sharp_booth
5ed051cd386d        447e0693112e           "/bin/sh -c 'apt ins…"   22 hours ago        Exited (100) 22 hours ago                            stupefied_lehma
nn
79ed3a4c5c8d        447e0693112e           "/bin/sh -c 'apt ins…"   22 hours ago        Exited (100) 22 hours ago                            quizzical_ediso
n
3c4dbd17de2a        447e0693112e           "/bin/sh -c 'apt ins…"   22 hours ago        Exited (100) 22 hours ago                            determined_enge
lbart
3e9d1f18ac47        447e0693112e           "/bin/sh -c 'apt ins…"   23 hours ago        Exited (100) 23 hours ago                            lucid_cerf
b56d5bf2e4fb        447e0693112e           "/bin/sh -c 'apt ins…"   23 hours ago        Exited (100) 23 hours ago                            hungry_ride
030ee1976b5e        447e0693112e           "/bin/sh -c 'apt ins…"   23 hours ago        Exited (100) 23 hours ago                            romantic_chandr
asekhar
18edc88f96fe        447e0693112e           "/bin/sh -c 'apt ins…"   23 hours ago        Exited (100) 23 hours ago                            sharp_meitner
e8cb0780bc0d        447e0693112e           "/bin/sh -c 'apt ins…"   23 hours ago        Exited (100) 23 hours ago                            cocky_chaplygin
b78a1d471f4e        447e0693112e           "/bin/sh -c 'apt ins…"   23 hours ago        Exited (100) 23 hours ago                            funny_greider
99f59fc1bd22        447e0693112e           "/bin/sh -c 'apt ins…"   23 hours ago        Exited (100) 23 hours ago                            gallant_matsumo
to
```

To pull any new images from Docker Hub:

sudo docker pull httpd

sodo docker images

```
root1@sujata:~$ sudo docker pull httpd
Using default tag: latest
latest: Pulling from library/httpd
bc51dd8edc1b: Pull complete
dca5bc65e18f: Pull complete
ccac3445152a: Pull complete
8515f2015fbc: Pull complete
e35494488b8c: Pull complete
Digest: sha256:b783a610e75380aa152dd855a18368ea2f3becb5129d0541e2ec8b662cbd8afb
Status: Downloaded newer image for httpd:latest
docker.io/library/httpd:latest
```

```
root1@sujata:~$ sudo docker images
REPOSITORY                                TAG                 IMAGE ID            CREATED             SIZE
raitnginxserver                           latest              1ac8603be6c7        22 hours ago        152MB
mynginxserver1                            latest              447e0693112e        24 hours ago        64.2MB
<none>                                    <none>              1166e35685f8        25 hours ago        247MB
sujata18061977/raitmodified_ubuntu        latest              6d5e32844787        39 hours ago        189MB
httpd                                     latest              c562eeace183        8 days ago          165MB
k8s.gcr.io/kube-proxy                     v1.17.2             cba2a99699bd        3 weeks ago         116MB
k8s.gcr.io/kube-apiserver                 v1.17.2             41ef50a5f06a        3 weeks ago         171MB
k8s.gcr.io/kube-controller-manager        v1.17.2             da5fd66c4068        3 weeks ago         161MB
k8s.gcr.io/kube-scheduler                 v1.17.2             f52d4c527ef2        3 weeks ago         94.4MB
ubuntu                                    latest              ccc6e87d482b        3 weeks ago         64.2MB
k8s.gcr.io/coredns                        1.6.5               70f311871ae1        3 months ago        41.6MB
k8s.gcr.io/etcd                           3.4.3-0             303ce5db0e90        3 months ago        288MB
quay.io/coreos/flannel                    v0.11.0-amd64       ff281650a721        12 months ago       52.6MB
k8s.gcr.io/pause                          3.1                 da86e6ba6ca1        2 years ago         742kB
docker/whalesay                           latest              6b362a9f73eb        4 years ago         247MB
```

Conclusion: Thus, Studied and Implemented Basic Docker Commands.

Assignment 2 (c):

Create an Angular application which will do following actions: Register User, Login User, Show User Data on Profile Component.

Theory:

What is Angular?

Angular is a typescript based free and open source web application framework developed by Google. Angular 10+ is a JavaScript framework which is used to create single page application. The Angular applications are created with the help of HTML and Typescript.

Angular CLI- command line interface tool that you use to initialize, develop and maintain angular application directly from command prompt or terminal.

Features:

High speed web application
Dynamic development
Full stack development

Pre-requisite for angular installation:

node.js and **npm** should be installed on your machine

How to install angular?

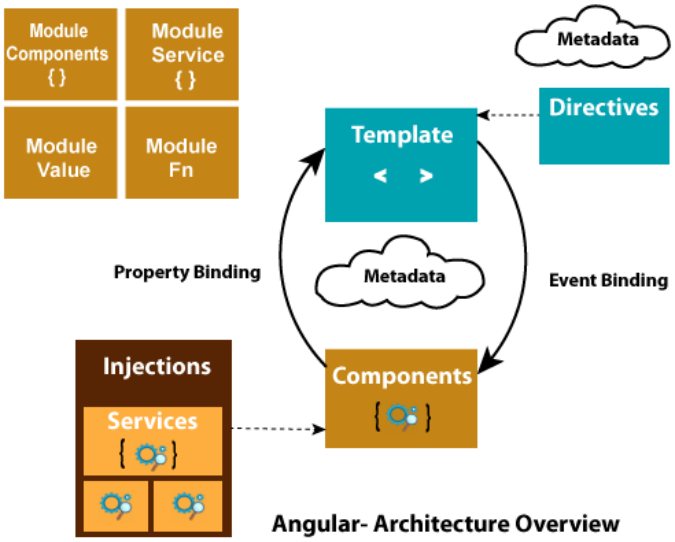
- **Step 1:** check your machine for node.js and npm
  node –v
  npm –v
- **Step 2:** Install Angular CLI
  npm install -g @angular/cli
- **Step 3:** check version of Angular
  ng version
- **Step 4:** Create workspace folder
  ng new Proj_name
  Would you like to add Angular routing? **Y**
  Which stylesheet format would you like to use? **CSS**
- **Step 5:** change directory to the proj_name folder and issue command
  ng serve -o

Angular Project Structure

The folder structure of Angular project is:

- **node_modules:** contains folders of packages which are installed
- **src folder:** this is the place where we need to put all our application source code
- **app folder:** When we want to create any component, service or module, we need to create it within this app folder.
- **assets folder:** you can store static assets like images, icons etc.
- **environment folder:** used to set up different environments.
- **favicon.ico:** It is the icon file that displays on the browser
- **index.html:** Starting point of our application.
- **main.ts** file
- **polyfills.ts:** used for browser-related configuration
- **angular. json file:** It contains the configuration of your angular project
- **test.ts** and **karma.config.js:** used for testing purpose
- **Package.json:** mandatory for every npm project

Angular Architecture



Components

Components and services both are simply classes with decorators that mark their types and provide metadata which guide Angular to do things.

Angular Modules

Angular 10 NgModules are different from other JavaScript modules. Every Angular 10 app has a root module known as AppModule. It provides the bootstrap mechanism that launches the application. Generally, every Angular 10 app contains many functional modules.

Angular Data Binding

Data binding defines the communication between components and its view.

Property binding: one way binding in which we can set the properties of the element to the user interface page.

Event binding: flow of data from view component.

Directives and Pipes

Directives is a technique in Angular that adds additional behavior to the elements in the Angular applications.

Pipes are used to transform the data.

The pipes are written using pipe operator which is denoted by |.

Angular Services and Dependency Injections (DI)

An Angular service is plain Typescript class having one or more methods along with @Injectable decorator.

The Services in angular are injected into the application using the dependency injection mechanism.

Dependency injection is a technique in which an object receives other objects that it depends on.

Conclusion:

Hence we have studied Angular and successfully implemented angular application.

Assignment 3 (b)

Create four API using Node.JS, ExpressJS and MongoDB for CURD Operations on assignment 2.C.

Theory:

REST HTTP Method APIs

- API (Application Programming Interface) are a set of functions and procedures that allow for the creation of applications that access data and features of other applications.
- REST (Representational State Transfer) is a set of rules that developers follow while creating API.
- REST uses various representation to represent a resource like text, JSON, XML but JSON is the most popular one.
- REST APIs enable you to develop all kinds of web applications having all possible CRUD (create, retrieve, update, delete) operations.
- The request consists of:
- **End Point:** An **endpoint** contains a *Uniform Resource Identifier (URI)* indicating where and how to find the resource on the Internet.
- **Method:** GET, POST, DELETE, PUT
- **Headers:** used to provide authentication and other useful information to client and server.
- **Data:** The DATA contains information which the client wants to send to the server. Preferred to send data in JSON format.

HTTP methods

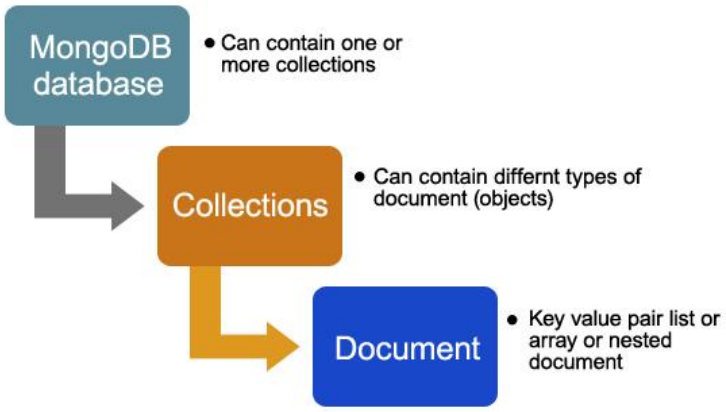
- GET − This is used to provide a read only access to a resource.
- PUT − This is used to update a new resource.
- DELETE − This is used to remove a resource.
- POST − This is used to create a new resource.

What is MongoDB?

MongoDB is an open-source database management system (DBMS) that uses a document-oriented database model. MongoDB is a NoSQL Database. MongoDB stores data in JSON-like documents, which makes the database very flexible and scalable.

MongoDB is a document-oriented database model. Each MongoDB database contains collections and which in turn contains documents. Each document can

be different and depends on the varying number of fields. The model of each document will be different in size and content from each other. The data model features allow you to store arrays and complex structured in a hierarchical relationship.



- **Collection:** Its group of MongoDB documents. This can be thought similar to a table in RDBMS like Oracle, MySQL. This collection doesn't enforce any structure. Hence schema-less MongoDB is so popular.
- **Document:** Document is referred to as a record in MongoDB collection.

MongoDB CURD Operation

CRUD operations create, read, update, and delete documents.

1. Create Operations

Create or insert operations add new documents to a collection. If the collection does not currently exist, insert operations will create the collection.

MongoDB provides the following methods to insert documents into a collection:

- db.collection.insertOne()
- db.collection.insertMany()

2. Read Operation

Read operations retrieve documents from a collection; i.e. query a collection for documents. MongoDB provides the following methods to read documents from a collection:

- db.collection.find()

3. Update Operation

MongoDB Update method is used to update the document from the collection.

We have used a $set operator at the time of updating the document.
Using the update method, we can update a single as well as multiple documents in one statement.

- db.collection.updateOne()
- db.collection.updateMany()
- db.collection.replaceOne()

4. Delete Operations

Delete operations remove documents from a collection. MongoDB provides the following methods to delete documents of a collection:

- db.collection.deleteOne()
- db.collection.deleteMany()

Conclusion:

Hence we have studied Node.js, ExpressJS and MongoDB and successfully created API.

Assignment 4 (b)

Deploy/Host Your web application on AWS VPC or AWS Elastic Beanstalk.

What is Cloud Computing?

Cloud computing is a term referred to storing and accessing data over the internet. It doesn't store any data on the hard disk of your personal computer. In cloud computing, you can access data from a remote server.

What is AWS?

The full form of AWS is Amazon Web Services. It is a platform that offers flexible, reliable, scalable, easy-to-use and, cost-effective cloud computing solutions.

AWS is a comprehensive, easy to use computing platform offered Amazon. The platform is developed with a combination of infrastructure as a service (IaaS), platform as a service (PaaS) and packaged software as a service (SaaS) offerings.

It is a secure cloud services platform, offering compute power, database storage, content delivery and other functionality to help businesses scale and grow.

In simple words AWS allows you to do the following things- Running web and application servers in the cloud to host dynamic websites.

History of AWS

- 2002- AWS services launched
- 2006- Launched its cloud products
- 2012- Holds first customer event
- 2015- Reveals revenues achieved of $4.6 billion
- 2016- Surpassed $10 billon revenue target
- 2016- Release snowball and snowmobile
- 2019- Offers nearly 100 cloud services
- 2021- AWS comprises over 200 products and services

Important AWS Services

Amazon Web Services offers a wide range of different business purpose global cloud-based products. The products include storage, databases, analytics, networking, mobile, development tools, enterprise applications, with a pay-as-you-go pricing model.

Applications of AWS services

Amazon Web services are widely used for various computing purposes like:

- Web site hosting
- Application hosting/SaaS hosting
- Media Sharing (Image/ Video)
- Mobile and Social Applications
- Content delivery and Media Distribution
- Storage, backup, and disaster recovery
- Development and test environments
- Academic Computing
- Search Engines
- Social Networking

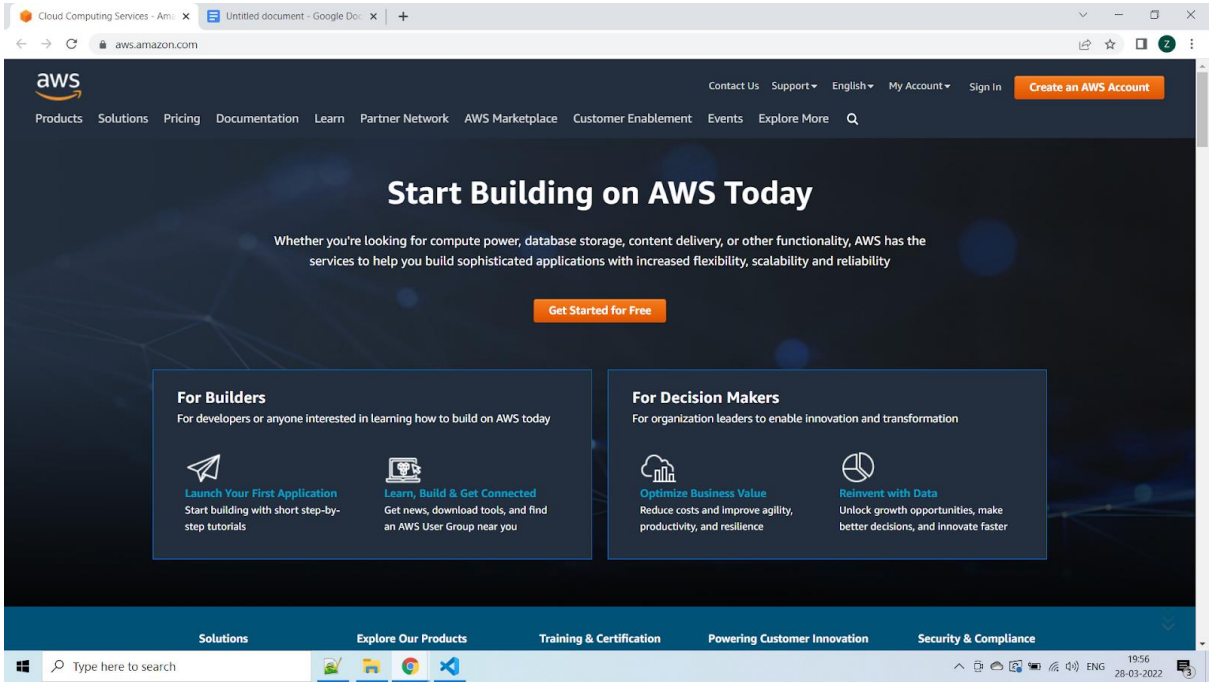
Creating an AWS Account is the first step you need to take in order to learn **Amazon Web Services.**

<u>Steps to follow are as follows:</u>

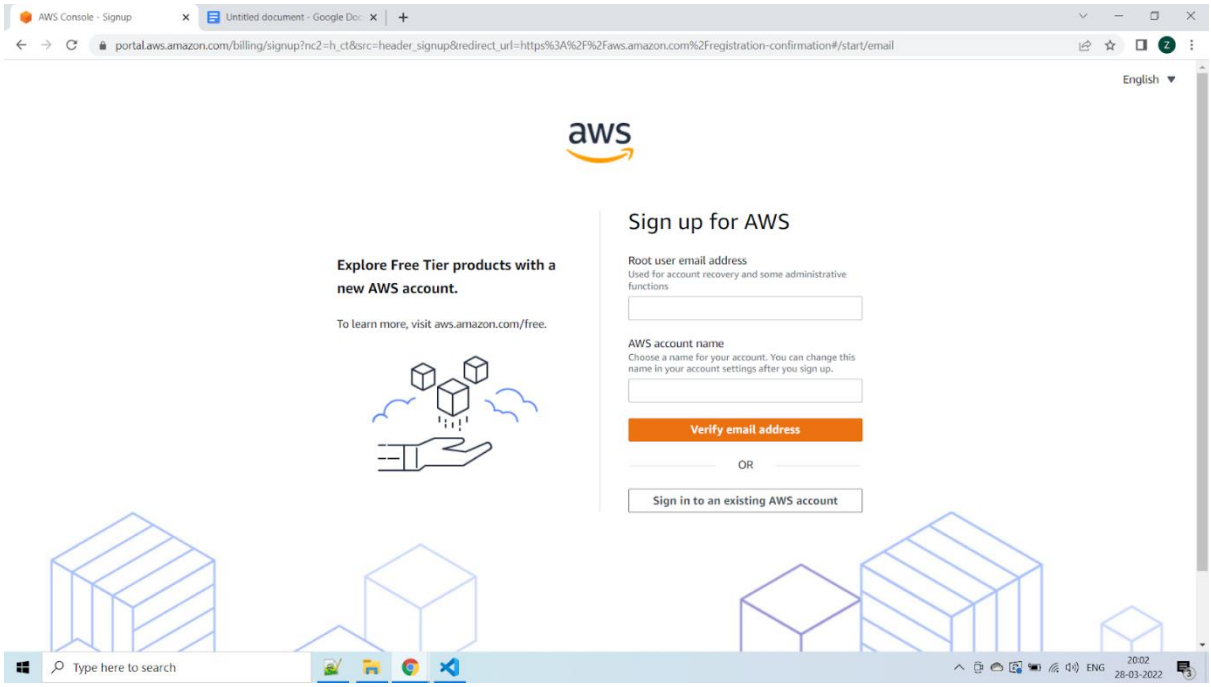
Step 1 – Visiting the Signup Page

Go to https://aws.amazon.com

You should see something like below:

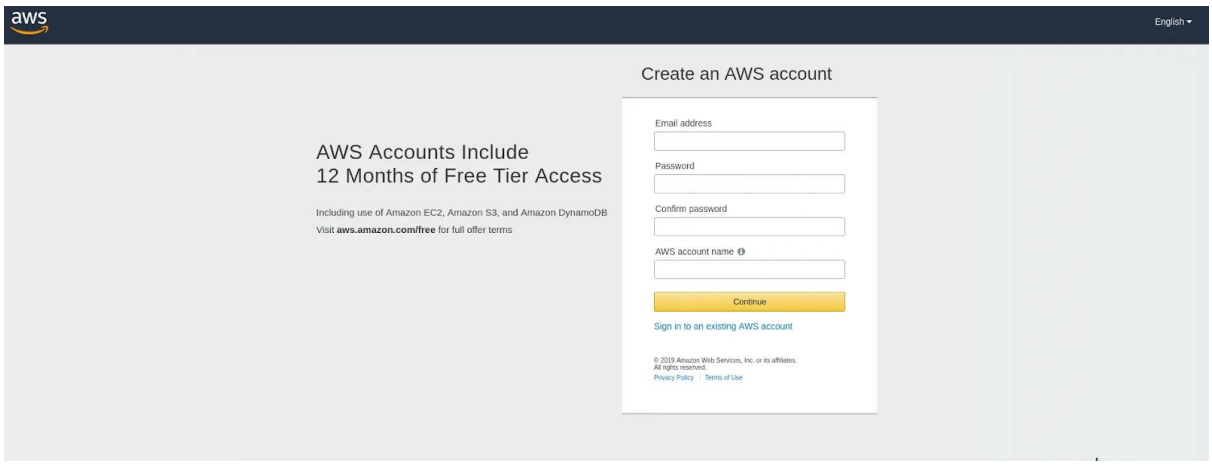


In order to continue, click the **Complete Sign Up** button in the middle of the screen or on the top right corner of the screen. You will see the below screen.



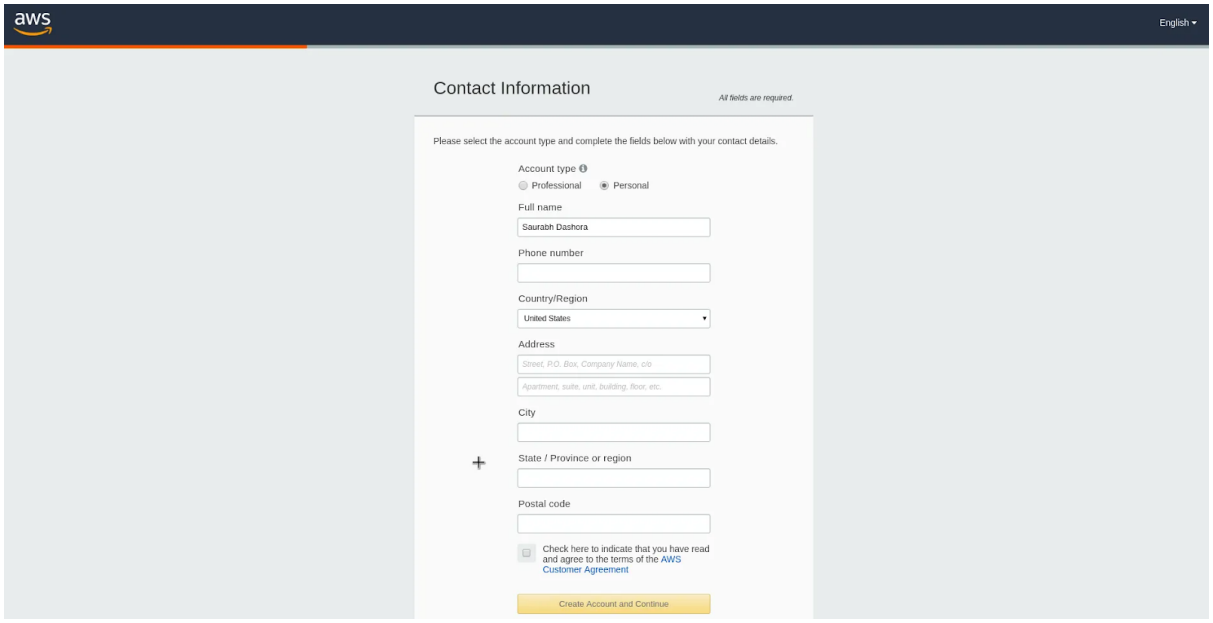
Step 2 – Entering User Details

After you have chosen to **Create a new AWS account**, you will see the below screen asking for few details.



You can fill up the details as per your requirements and click **Continue**.

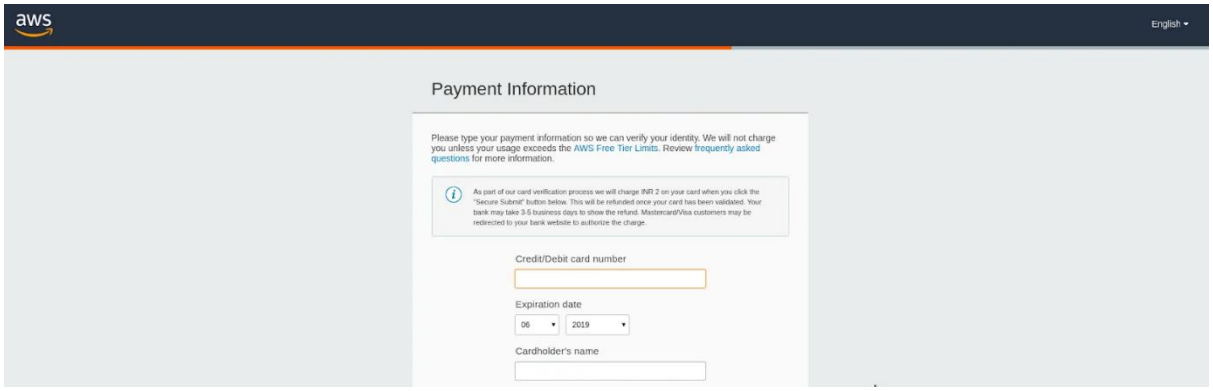
Next you will be asked to fill up your contact details such contact number, country, address and so on. You should fill them up properly because your contact number is important for further steps.



After filling up the details, click on the **Create Account and Continue** button at the bottom of the form.

Step 3 – Filling up the Credit Card details

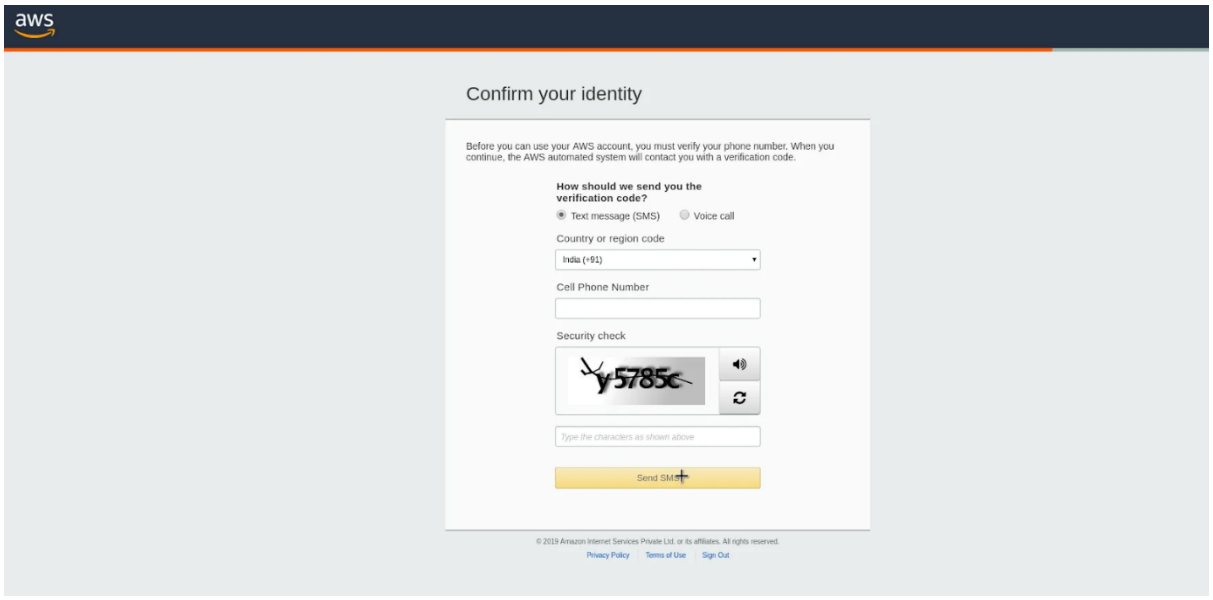
For **Creating an AWS Account**, you need to enter your **Credit Card** details.



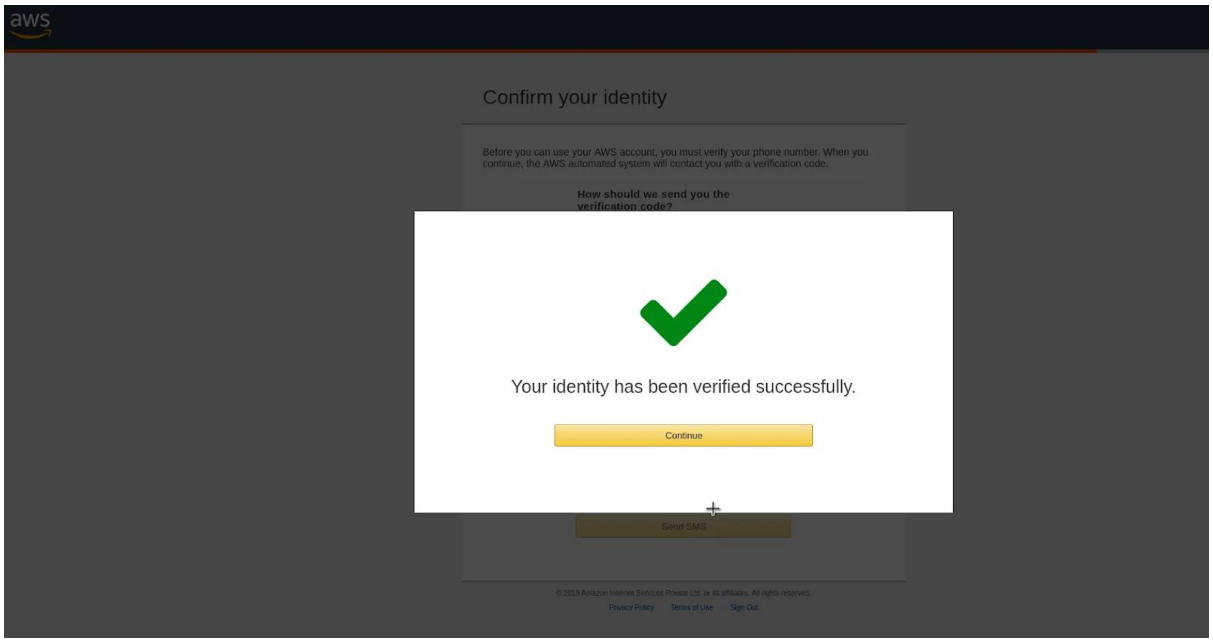
After entering the details, click on **Secure Submit** button. It might take a while to process the request depending on your bank/credit card company servers.

Step 4 – Identity Confirmation

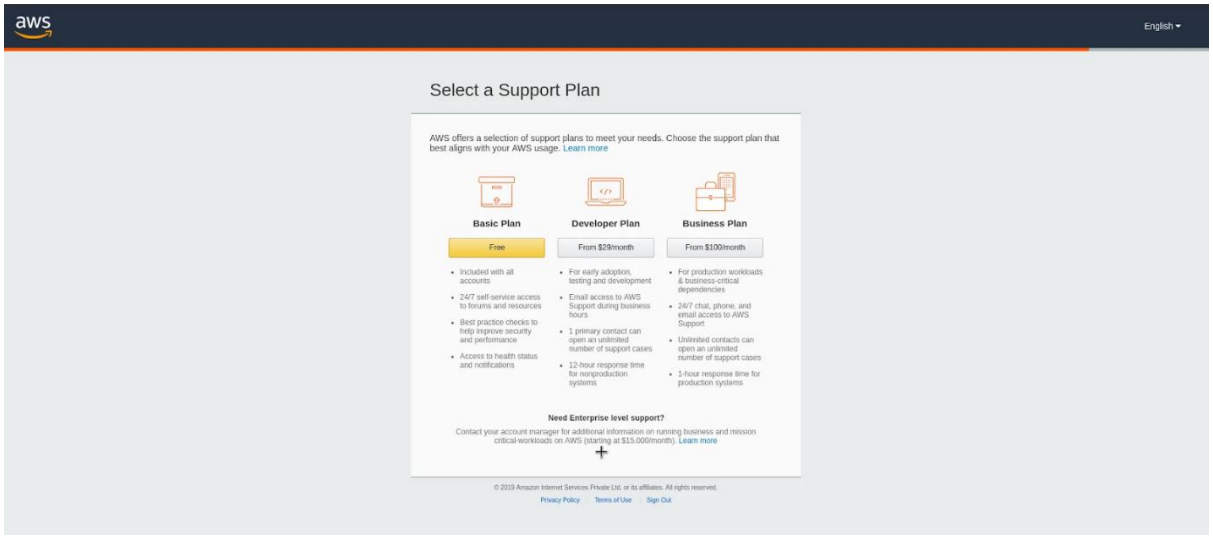
Once the credit card details are confirmed, you will need to complete the **Identity Confirmation** step. You will see the below screen:



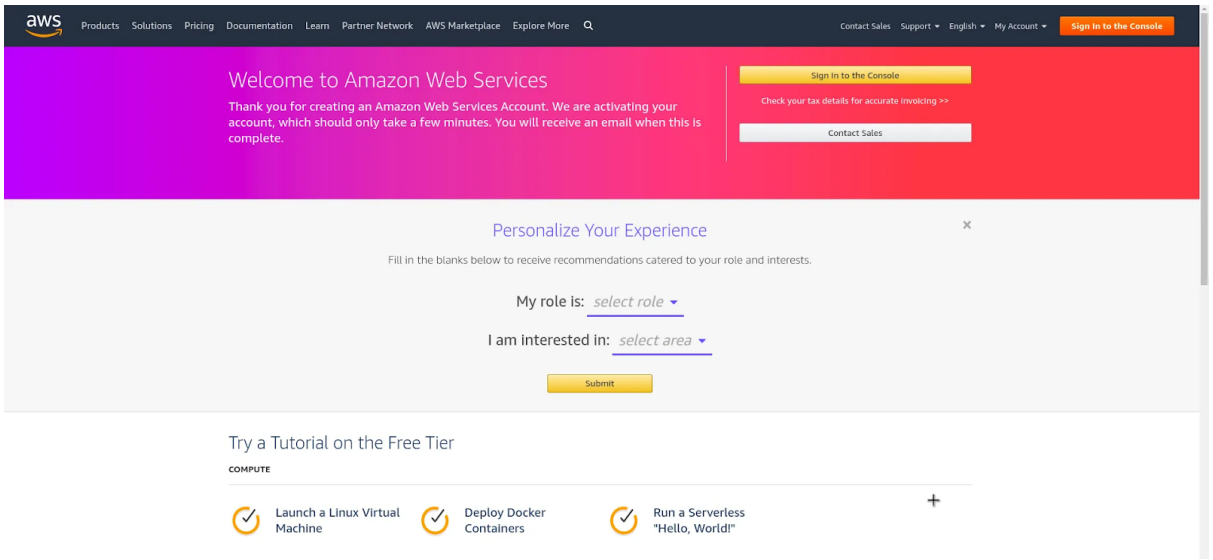
Once you have verified successfully, you should see a screen like below:



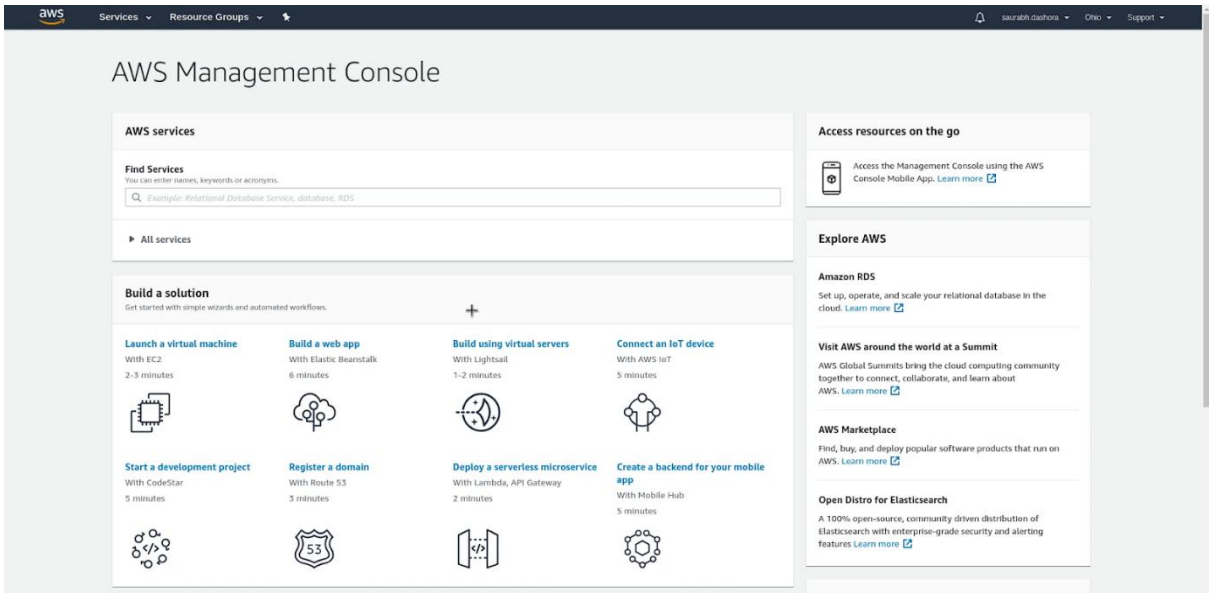
Step 5 – Selecting a Support Plan



Go for **Basic Plan**. It is Free of cost and great for learning purposes.
The other plans are **Developer Plan** and a **Business Plan**. But both of them are paid options.
Once you select your plan, you will see the below **Welcome** screen. From here on, you can Sign in to your **AWS Console**.



Finally, after logging in, you should be able to see the **AWS Management Console** as below:



If you have reached this far, you have successfully finished **Creating an AWS Account**.

Deployment Steps:

Step 1: Launch a Windows Server Amazon EC2 instance.
Step 2: Configure your source content to deploy to the Windows Server Amazon EC2 instance.
Step 3: Upload your "hello, world!" ...
Step 4: Deploy your Hello World application.
Step 5: Update and redeploy your "hello, world!" ...
Step 6: Clean up your "hello, world!"

Conclusion:
Hence we have successfully deployed web on AWS Elastic Beanstalk.