

Name:

Division:

Roll No.:

Practical Number: 1

Title: Implement depth-first search algorithm and Breadth First Search algorithm, Use an undirected graph, and develop a recursive algorithm for searching all the vertices of a graph or tree data structure.

DFS:

Code:-

```
graph = {
    '5' : ['3', '7'],
    '3' : ['2', '4'],
    '7' : ['8'],
    '2' : [],
    '4' : ['8'],
    '8' : []
}

visited = set()

def dfs(visited, graph, node):
    if node not in visited:
        print (node)
        visited.add(node)
        for neighbour in graph[node]:
            dfs(visited, graph, neighbour)

print("Following is the Depth-First Search:")
dfs(visited, graph, '5')
```

Output:-

```
Following is the Depth-First Search:
5
3
2
4
8
7
```

BFS:

Code:-

```
graph = {
    'A' : ['B', 'C', 'D'],
    'B' : ['E'],
    'C' : ['D', 'F'],
    'D' : [],
    'E' : [],
    'F' : []
}
visited = []
queue = []

def bfs(visited, graph, node):
    visited.append(node)
    queue.append(node)
    while queue:
        s = queue.pop(0)
        print (s, end = " ")

        for neighbour in graph[s]:
            if neighbour not in visited:
                visited.append(neighbour)
                queue.append(neighbour)

print("Following is the Path using Breadth-First Search:")
bfs(visited, graph, 'A')
```

Output:-

```
Following is the Path using Breadth-First Search:
A B C D E F
```

Practical Number 2:

Title: Implement A star Algorithm for any game search problem.

Code:-

```
from collections import deque

class Graph:
    def __init__(self, adjacency_list):
        self.adjacency_list = adjacency_list

    def get_neighbors(self, v):
        return self.adjacency_list[v]

    def h(self, n):
        H = {
            'A': 1,
            'B': 1,
            'C': 1,
            'D': 1
        }
        return H[n]

    def a_star_algorithm(self, start_node, stop_node):
        open_list = set([start_node])
        closed_list = set([])
        g = {}
        g[start_node] = 0
        parents = {}
        parents[start_node] = start_node

        while len(open_list) > 0:
            n = None

            for v in open_list:
                if n == None or g[v] + self.h(v) < g[n] + self.h(n):
                    n = v;

            if n == None:
                print("Path does not exist!")
                return None

            if n == stop_node:
                reconst_path = []
                while parents[n] != n:
                    reconst_path.append(n)
                    n = parents[n]
                reconst_path.append(start_node)
                reconst_path.reverse()
                print("Path found: {}".format(reconst_path))
                return reconst_path

            for (m, weight) in self.get_neighbors(n):
                if m not in open_list and m not in closed_list:
                    open_list.add(m)
                    parents[m] = n
                    g[m] = g[n] + weight

                else:
                    if g[m] > g[n] + weight:
                        g[m] = g[n] + weight
                        parents[m] = n

                        if m in closed_list:
                            closed_list.remove(m)
                            open_list.add(m)

            open_list.remove(n)
            closed_list.add(n)

            print('Path does not exist!')
            return None
        adjacency_list = {
            'A': [('B', 1), ('C', 3), ('D', 7)],
            'B': [('D', 5)],
            'C': [('D', 12)]
        }
        graph1 = Graph(adjacency_list)
        graph1.a_star_algorithm('A', 'D')
```

Output:-

```
Path found: ['A', 'B', 'D']
```

```
['A', 'B', 'D']
```

Practical Number: 3

Title: Implement Greedy search algorithm for Selection Sort

Code:-

```
x = []
n = int(input("Enter how many elements you want for sorting"))
for i in range(n):
    print("Enter list elements")
    a = int(input(""))
    x.append(a)
print("unsorted Elements list is: ", x)

for i in range(0,len(x)-1):
    for j in range(i+1,len(x)):
        if x[i]>x[j]:
            c=x[i]
            x[i]=x[j]
            x[j]=c
print("Sorted Elements List is: ",x)
```

Output:-

```
Enter how many elements you want for sorting 5
Enter list elements
5
Enter list elements
3
Enter list elements
1
Enter list elements
4
Enter list elements
2
unsorted Elements list is: [5, 3, 1, 4, 2]
Sorted Elements List is: [1, 2, 3, 4, 5]
```

Practical Number: 4

Title: Implement a solution for a Constraint Satisfaction Problem using Branch and Bound and Backtracking for an n-queens problem.

Code:-

```
n = int(input("Enter the value for n:\n"))
board = [[0 for i in range(n)] for i in range(n)]

def check_column(board, row, column):
    for i in range(row, -1, -1):
        if board[i][column] == 1:
            return False
    return True

def check_diagonal(board, row, column):
    for i, j, in zip(range(row, -1, -1), range(column, -1, -1)):
        if board[i][j] == 1:
            return False
    for i, j, in zip(range(row, -1, -1), range(column, n)):
        if board[i][j] == 1:
            return False
    return True

def nqn(board, row):
    if row == n:
        return True
    for i in range(n):
        if (check_column(board, row, i) == True and check_diagonal(board, row, i) == True):
            board[row][i] = 1
            if nqn(board, row + 1):
                return True
            board[row][i] = 0
    return False

nqn(board, 0)
for row in board:
    print(row)
```

Output:-

```
Enter the value for n
4
[0, 1, 0, 0]
[0, 0, 0, 1]
[1, 0, 0, 0]
[0, 0, 1, 0]
```

Practical Number: 5

Title: Develop an elementary chatbot for any suitable customer interaction application.

Code:-

```
def greet(bot_name, birth_year):
    print("Hello! My name is {0}.".format(bot_name))
    print("I was created in {0}.".format(birth_year))

def remind_name():
    print('Please, remind me your name.')
    name = input()
    print("What a great name you have, {0}!".format(name))

def guess_age():
    print('Let me guess your age.')
    print('Enter remainders of dividing your age by 3, 5 and 7.')

    rem3 = int(input())
    rem5 = int(input())
    rem7 = int(input())
    age = (rem3 * 70 + rem5 * 21 + rem7 * 15) % 185

    print("Your age is {0}; that's a good time to start programming!".format(age))

def count():
    print('Now I will prove to you that I can count to any number you want.')
    num = int(input())

    counter = 0
    while counter <= num:
        print("{0} !".format(counter))
        counter += 1

def test():
    print("Let's test your programming knowledge.")
    print("Why do we use methods?")
    print("1. To repeat a statement multiple times.")
    print("2. To decompose a program into several small subroutines.")
    print("3. To determine the execution time of a program.")
    print("4. To interrupt the execution of a program.")

    answer = 2
    guess = int(input())
    while guess != answer:
        print("Please, try again.")
        guess = int(input())

    print('Completed, have a nice day!')
    print('.....')
    print('.....')
    print('.....')

def end():
    print('Congratulations, have a nice day!')
    print('.....')
    print('.....')
    print('.....')
    input()

greet('Chatbot', '2023')
remind_name()
guess_age()
count()
test()
end()
```

Output:-

```
Hello! My name is Chatbot.  
I was created in 2023.  
Please, remind me your name.  
ABCD  
What a great name you have, ABCD!  
Let me guess your age.  
Enter remainders of dividing your age by 3, 5 and 7.  
2  
0  
6  
Your age is 20; that's a good time to start programming!  
Now I will prove to you that I can count to any number you want.  
5  
0 !  
1 !  
2 !  
3 !  
4 !  
5 !  
Let's test your programming knowledge.  
Why do we use methods?  
1. To repeat a statement multiple times.  
2. To decompose a program into several small subroutines.  
3. To determine the execution time of a program.  
4. To interrupt the execution of a program.  
2  
Completed, have a nice day!  
.....  
.....  
.....  
Congratulations, have a nice day!  
.....  
.....  
.....
```

Practical Number: 6

Title: Implement any one Expert System (Help desks management)

Code:-

```
problem_dict = {
    "Printer not working": "Check that it's turned on and connected to the network",
    "Can't log in": "Make sure you're using the correct username and password",
    "Software not installing": "Check that your computer meets the system requirements",
    "Internet connection not working": "Restart your modem or router",
    "Email not sending": "Check that you're using the correct email server settings"
}

def handle_request(user_input):
    if user_input.lower() == "exit":
        return "Goodbye!"
    elif user_input in problem_dict:
        return problem_dict[user_input]
    else:
        return "I'm sorry, I don't know how to help with that problem."

while True:
    user_input = input("What's the problem? Type 'exit' to quit. ")
    response = handle_request(user_input)
    print(response)
```

Output:-

```
What's the problem? Type 'exit' to quit. Software not installing
Check that your computer meets the system requirements
What's the problem? Type 'exit' to quit. Printer not working
I'm sorry, I don't know how to help with that problem.
What's the problem? Type 'exit' to quit. exit
Goodbye!
```

Practical No: 7

Practical Title: Case study on Amazon EC2 and learn about Amazon EC2 web services.

Objectives:

- To learn Amazon EC2 web services
- To study on Amazon EC2 and learn about Amazon EC2 web services.

Hardware Requirements :

- Pentium IV with latest configuration

Software Requirements :

- Ubuntu 20.04

Theory:

An EC2 instance is nothing but a virtual server in Amazon [Web services](#) terminology. It stands for Elastic Compute Cloud. It is a web service where an AWS subscriber can request and provision a compute server in AWS cloud.

An on-demand EC2 instance is an offering from AWS where the subscriber/user can rent the virtual server per hour and use it to deploy his/her own applications.

The instance will be charged per hour with different rates based on the type of the instance chosen. AWS provides multiple instance types for the respective business needs of the user.

Thus, you can rent an instance based on your own CPU and memory requirements and use it as long as you want. You can terminate the instance when it's no more used and save on costs. This is the most striking advantage of an on-demand instance- you can drastically save on your CAPEX.

Let us see in detail how to launch an on-demand EC2 instance in AWS Cloud. Login and access to AWS services

Step 1) In this step,

- Login to your AWS account and go to the AWS Services tab at the top left corner.
- Here, you will see all of the AWS Services categorized as per their area viz. Compute, Storage, Database, etc. For creating an EC2 instance, we have to choose Compute & EC2 as in the next step.

The screenshot shows the AWS Services menu with a red arrow pointing to the 'Compute' section under 'All AWS Services'. The 'Compute' section is highlighted with a red box and contains the following options: Storage & Content Delivery, Database, Networking, Developer Tools, Management Tools, and Security & Identity.

- Open all the services and click on EC2 under Compute services. This will launch the dashboard of EC2.

Here is the EC2 dashboard. Here you will get all the information in gist about the AWS EC2 resources running.

The screenshot shows the EC2 Dashboard with a red arrow pointing to the 'Resources' section. The 'Resources' section is highlighted with a red box and displays the following statistics:

3 Running Instances	4 Elastic IPs
0 Dedicated Hosts	17 Snapshots
12 Volumes	0 Load Balancers
22 Key Pairs	28 Security Groups
0 Placement Groups	

Step 2) On the top right corner of the EC2 dashboard, choose the AWS Region in which you want to provision the EC2 server.

Here we are selecting N. Virginia. AWS provides 10 Regions all over the globe

Step 3) In this step

- Once your desired Region is selected, come back to the EC2 Dashboard.
- Click on 'Launch Instance' button in the section of Create Instance (as shown below).

The screenshot shows the AWS EC2 Dashboard. In the top navigation bar, the region is set to 'N. Virginia'. A red arrow points down to the dropdown menu where 'US East (N. Virginia)' is highlighted. The 'Resources' section displays statistics for running instances, dedicated hosts, volumes, key pairs, and placement groups. The 'Create Instance' section features a prominent 'Launch Instance' button, which is also highlighted with a red box.

- Instance creation wizard page will open as soon as you click 'Launch Instance'. Choose AMI

Step 1) In this step we will do,

- You will be asked to choose an AMI of your choice. (An AMI is an Amazon Machine Image. It is a template basically of an Operating System platform which you can use as a base to create your instance). Once you launch an EC2 instance from your preferred AMI, the instance will automatically be booted with the desired OS. (We will see more about AMIs in the coming part of the tutorial).
- Here we are choosing the default Amazon Linux (64 bit) AMI.

The screenshot shows the 'Choose an Amazon Machine Image (AMI)' step in the instance creation wizard. Step 1 is highlighted with a red circle labeled 1. The 'Amazon Linux AMI 2015.09.1 (HVM), SSD Volume Type - ami-60b6c60a' is selected and highlighted with a red box labeled 2. The 'Select' button is also highlighted with a red box labeled 2. The page displays information about the selected AMI, including its description and supported architectures.

Choose EC2 Instance Types

Step 1) In the next step, you have to choose the type of instance you require based on your business needs.

1. We will choose t2.micro instance type, which is a 1vCPU and 1GB memory server offered by AWS.
2. Click on "Configure Instance Details" for further configurations

The screenshot shows the AWS EC2 wizard at Step 2: Choose an Instance Type. A red box highlights the 't2.micro' instance type in the list, with a red number '1' indicating it is selected. Another red box highlights the 'Next: Configure Instance Details' button at the bottom right, with a red number '2' indicating the next step.

- In the next step of the wizard, enter details like no. of instances you want to launch at a time.
- Here we are launching one instance. Configure Instance

Step 1) No. of instances- you can provision up to 20 instances at a time. Here we are launching one instance.

The screenshot shows the AWS EC2 wizard at Step 3: Configure Instance Details. A red arrow points to the 'Number of instances' input field, which contains the value '1'. Below the input field is a 'Launch into Auto Scaling Group' checkbox. The purchasing options section is visible at the bottom.

Step 2) Under Purchasing Options, keep the option of 'Request Spot Instances' unchecked as of now. (This is done when we wish to launch Spot instances instead of on-demand ones. We will come back to Spot instances in the later part of the tutorial).

The screenshot shows the AWS EC2 wizard at Step 3: Configure Instance Details. A red arrow points to the 'Request Spot instances' checkbox, which is currently unchecked. The purchasing options section is visible at the bottom.

Step 3) Next, we have to configure some basic networking details for our EC2 server.

- You have to decide here, in which VPC (Virtual Private Cloud) you want to launch your instance and under which subnets inside your VPC. It is better to determine and plan this prior to launching the instance. Your AWS architecture set-up should include IP ranges for your subnets etc. pre-planned for better management. (We will see how to create a new VPC in Networking section of the tutorial).

- Subnetting should also be pre-planned. E.g.: If it's a web server you should place it in the public subnet and if it's a DB server, you should place it in a private subnet all inside your VPC.

Below,

- Network section will give a list of VPCs available in our platform.
- Select an already existing VPC
- You can also create a new VPC

Here I have selected an already existing VPC where I want to launch my instance.

Step 3: Configure Instance Details

Configure the instance to suit your requirements. You can launch multiple instances from the same AMI, request Spot instances to take advantage of the role to the instance, and more.

Number of instances	1	Launch into Auto Scaling Group
Purchasing option	<input type="checkbox"/> Request Spot instances	
Network	vpc-d5194fb0 (192.168.0.0/16) Prachi_Test - VPC	
Subnet	subnet-b3e3d0ea(192.168.2.0/24) Prachi_Test-Public Subnet 3 us-east-1a	
Auto-assign Public IP	vpc-d5194fb0 (192.168.0.0/16) Prachi_Test - VPC	
IAM role	None	

Create new VPC

Create new subnet

Create new IAM role

Step 4) In this step,

- A VPC consists of subnets, which are IP ranges that are separated for restricting access.
 - Below,
- Under Subnets, you can choose the subnet where you want to place your instance.
 - I have chosen an already existing public subnet.
 - You can also create a new subnet in this step.

Step 3: Configure Instance Details

Configure the instance to suit your requirements. You can launch multiple instances from the same AMI, request Spot instances to take advantage of the role to the instance, and more.

Number of instances	1	Launch into Auto Scaling Group
Purchasing option	<input type="checkbox"/> Request Spot instances	
Network	vpc-d5194fb0 (192.168.0.0/16) Prachi_Test - VPC	
Subnet	subnet-b3e3d0ea(192.168.2.0/24) Prachi_Test-Public Subnet 3 us-east-1a	
Auto-assign Public IP	vpc-d5194fb0 (192.168.0.0/16) Prachi_Test - VPC	
IAM role	None	

Create new VPC

Create new subnet

Create new IAM role

- Once your instance is launched in a public subnet, AWS will assign a dynamic public IP to it from their pool of IPs.

Step 5) In this step,

- You can choose if you want AWS to assign it an IP automatically, or you want to do it manually later. You can enable/ disable 'Auto assign Public IP' feature here likewise.
- Here we are going to assign this instance a static IP called as EIP (Elastic IP) later. So we keep this feature disabled as of now.

Step 3: Configure Instance Details

Configure the instance to suit your requirements. You can launch multiple instances from the same AMI, request Spot Instances to take advantage of the lower price, assign an access management role to the instance, and more.

Number of instances	1	Launch into Auto Scaling Group
Purchasing option	<input type="checkbox"/> Request Spot instances	
Network	vpc-d5194fb0 (192.168.0.0/16) Prachi_Test - VPC	<input type="button"/> Create new VPC
Subnet	subnet-b3e3d0ea(192.168.2.0/24) Prachi_Test-Subnet	<input type="button"/> Create new subnet 251 IP Addresses available
Auto-assign Public IP	<input type="button"/> Use subnet setting (Disable) <input checked="" type="button"/> Use subnet setting (Enable)	<input type="button"/> Create new IAM role
IAM role	None	<input type="button"/> Create new IAM role
Shutdown behavior	Stop	

Step 3: Configure Instance Details

Configure the instance to suit your requirements. You can launch multiple instances from the same AMI, request Spot Instances to take advantage of the lower price, assign an access management role to the instance, and more.

IAM role	None	<input type="button"/> Create new IAM role
Shutdown behavior	Stop	
Enable termination protection	<input checked="" type="checkbox"/> Protect against accidental termination	
Monitoring	<input type="checkbox"/> Enable CloudWatch detailed monitoring Additional charges apply.	
Tenancy	<input type="button"/> Shared - Run a shared hardware instance <input checked="" type="button"/> Shared - Run a shared hardware instance <input type="button"/> Dedicated - Run a Dedicated instance <input type="button"/> Dedicated host - Launch this instance on a Dedicated host	
Network interfaces		

Step 3: Configure Instance Details

Configure the instance to suit your requirements. You can launch multiple instances from the same AMI, request Spot Instances to take advantage of the lower price, assign an access management role to the instance, and more.

Number of instances	1	Launch into Auto Scaling Group
Purchasing option	<input type="checkbox"/> Request Spot instances	
Network	vpc-d5194fb0 (192.168.0.0/16) Prachi_Test - VPC	<input type="button"/> Create new VPC
Subnet	subnet-b3e3d0ea(192.168.2.0/24) Prachi_Test-Subnet	<input type="button"/> Create new subnet 251 IP Addresses available
Auto-assign Public IP	<input type="button"/> Use subnet setting (Disable)	
IAM role	None	<input type="button"/> Create new IAM role
Shutdown behavior	Stop	
Enable termination protection	<input checked="" type="checkbox"/> Protect against accidental termination	
Monitoring	<input type="checkbox"/> Enable CloudWatch detailed monitoring Additional charges apply.	
Tenancy	<input type="button"/> Shared - Run a shared hardware instance	
<input type="button"/> Cancel <input type="button"/> Previous <input type="button"/> Review and Launch <input type="button"/> Next: Add Storage		

Launch Status

✓ Your instances are now launching

The following instance launches have been initiated: i-4c2c3cff [Hide launch log](#)

Creating security groups	Successful (sg-62d7d21b)
Authorizing inbound rules	Successful
Initiating launches	Successful
Applying tags	Successful
Launch initiation complete	

ⓘ Get notified of estimated charges

Create billing alerts to get an email notification when estimated charges on your AWS bill exceed an am

The screenshot shows the AWS EC2 Instances page. On the left is a navigation sidebar with links like EC2 Dashboard, Events, Tags, Reports, Limits, Instances, Spot Requests, Reserved Instances, Scheduled Instances, Commands, Dedicated Hosts, Images, AMIs, Bundle Tasks, Elastic Block Store, Volumes, Snapshots, Networks & Security, Security Groups, Elastic IPs, Placement Groups, and Key Pairs. The main area has tabs for Launch Instance, Connect, and Actions. Below is a search bar and a table with one row. The table columns include Name, Instance ID, Instance Type, Availability Zone, Instance State, Status Checks, and Alarm Status. The instance listed is 'Dev_Web_Server_01' with ID 'i-4c2c3cff', type 't2.micro', and state 'running'. A red box highlights the 'running' status. At the bottom of the table is a summary bar with 'Instance: i-4c2c3cff (Dev_Web_Server_01) Private IP: 192.168.2.167' and four tabs: Description, Status Checks, Monitoring, and Tags. The 'Description' tab is selected.

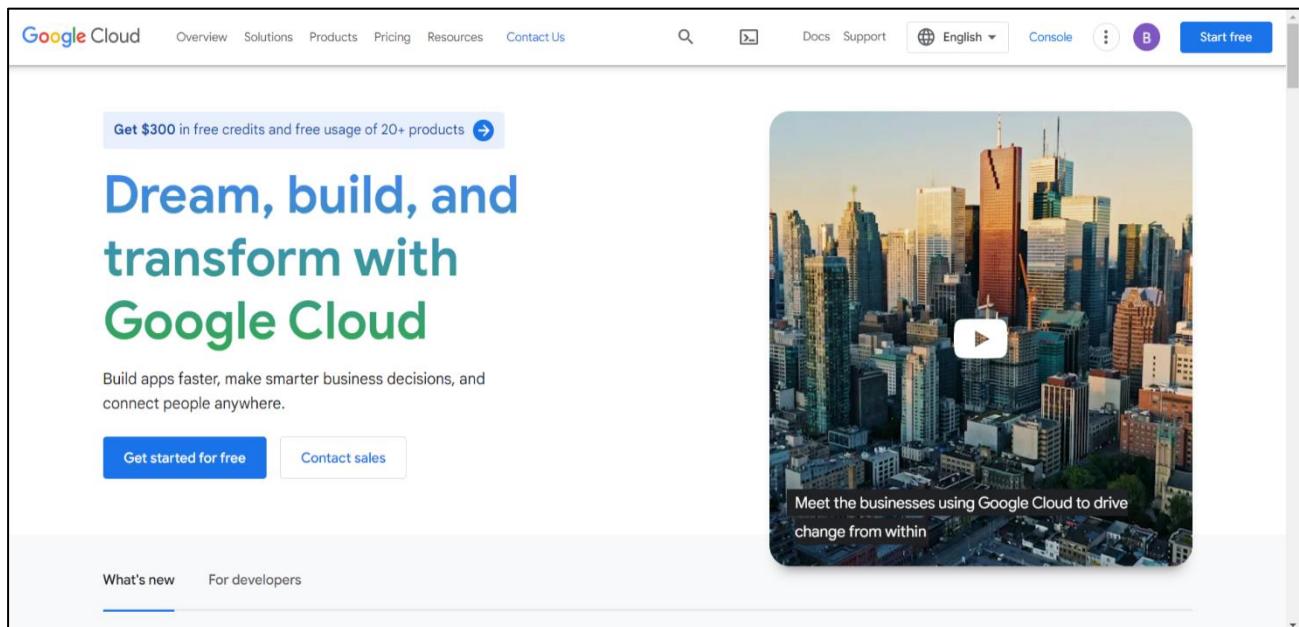
Conclusion:

Thus, we saw in detail how to create an on-demand EC2 instance in this tutorial. Because it is an on-demand server, you can keep it running when in use and 'Stop' it when it's unused to save on your costs

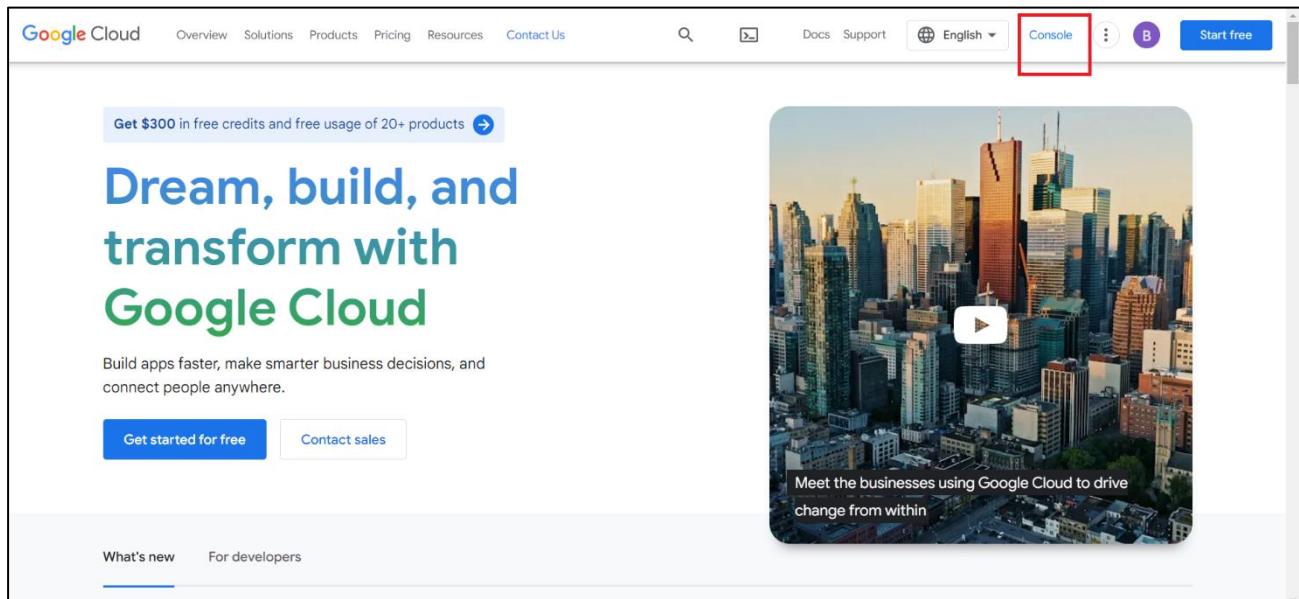
Practical Number: 8

Title: Installation and Configure Google App Engine.

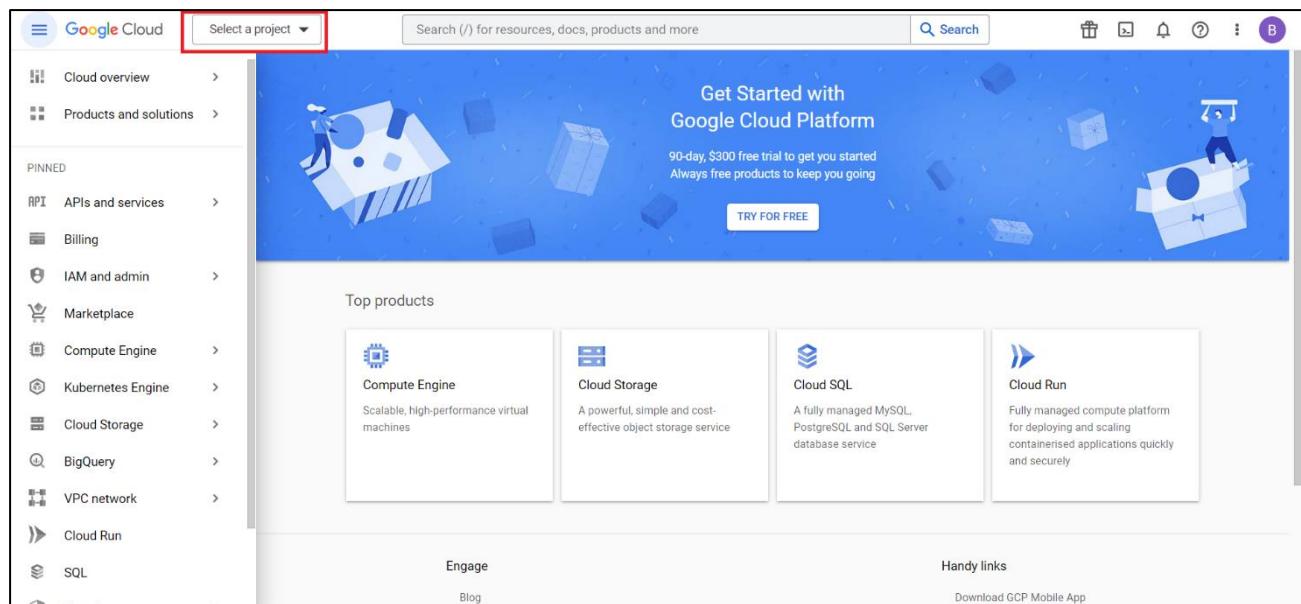
- Go to the Google Cloud Platform(GCP) Website and Sign in to your Google account: <https://cloud.google.com>



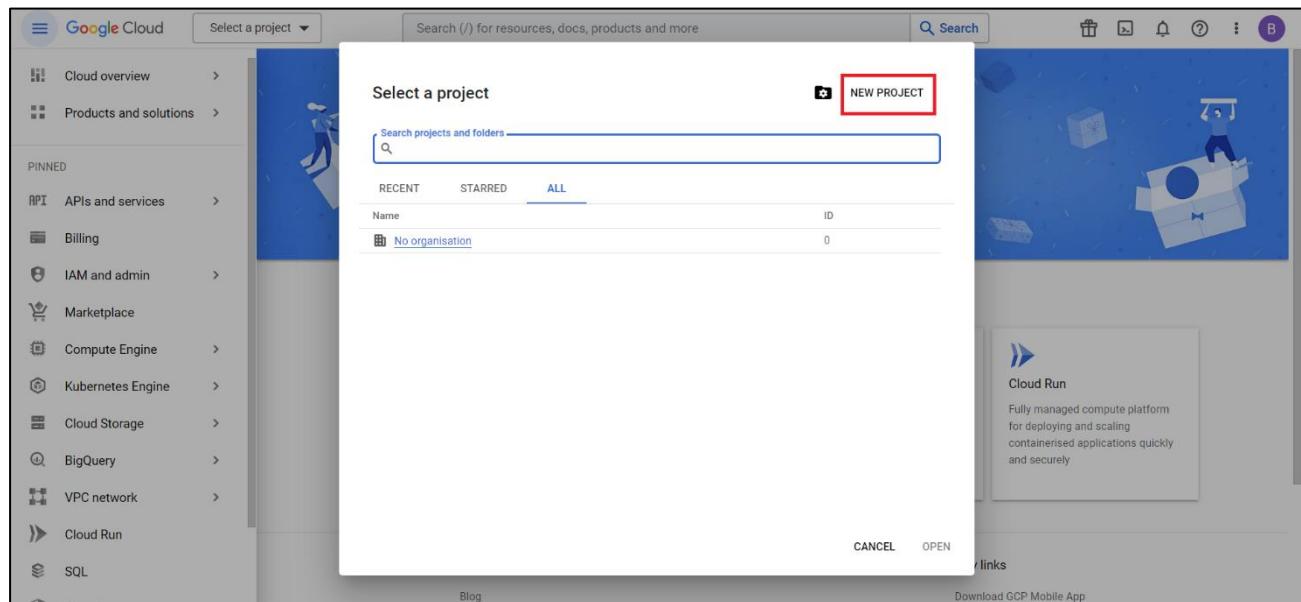
- Click on “Console”:



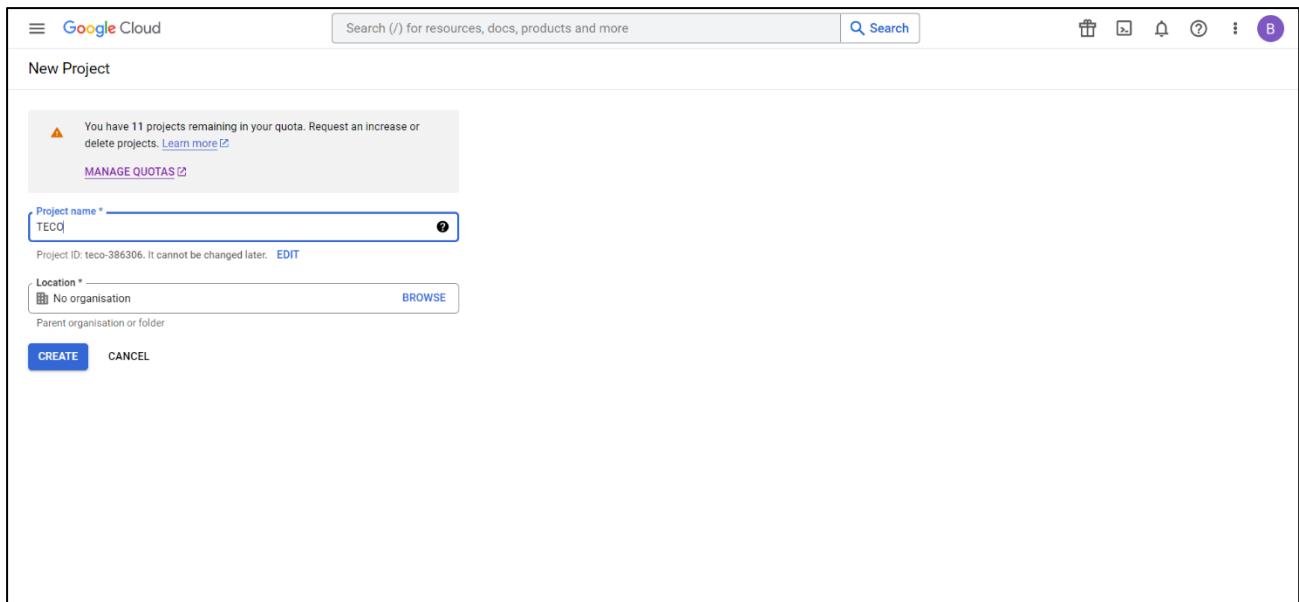
- Click on “Select a Project”:



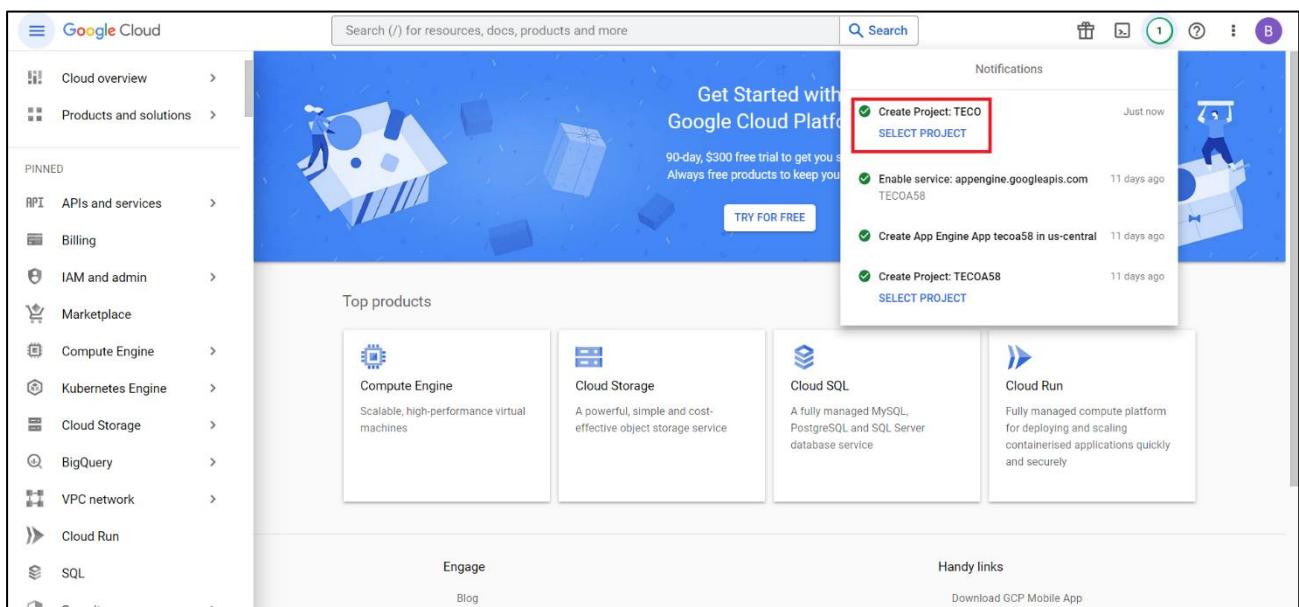
- Click on “NEW PROJECT”:



➤ **Enter Project Name :**



➤ **Click on “Select Project” from the notifications:**



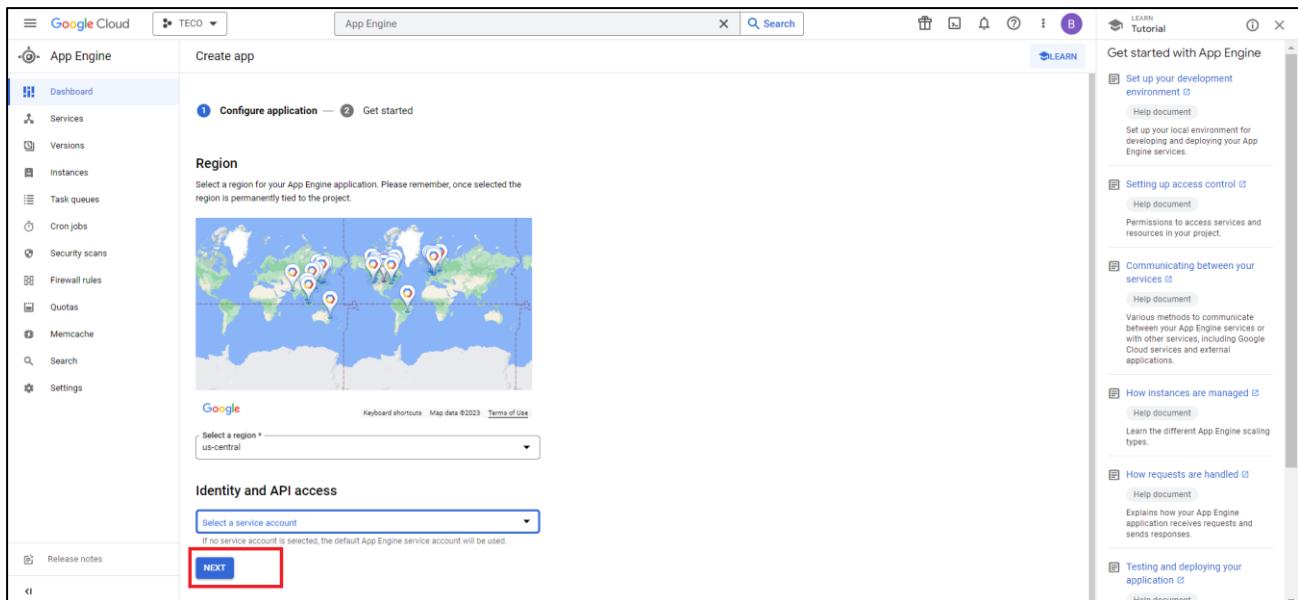
- Search “App Engine” and select it from the list:

The screenshot shows the Google Cloud Platform dashboard for project 'TECO'. The search bar at the top right contains the query 'App Engine'. Below the search bar, the results are displayed under the heading 'PRODUCTS & PAGES'. The first result, 'App Engine Managed app platform', is highlighted with a red box. Other results include 'Compute Engine' (VMs, GPUs, TPUs, disks), 'Engines' (Gen App Builder), and various documentation links like 'Run a Python web application on Google Compute Engine and Cloud SQL' and 'Create a client-server application on Compute Engine'.

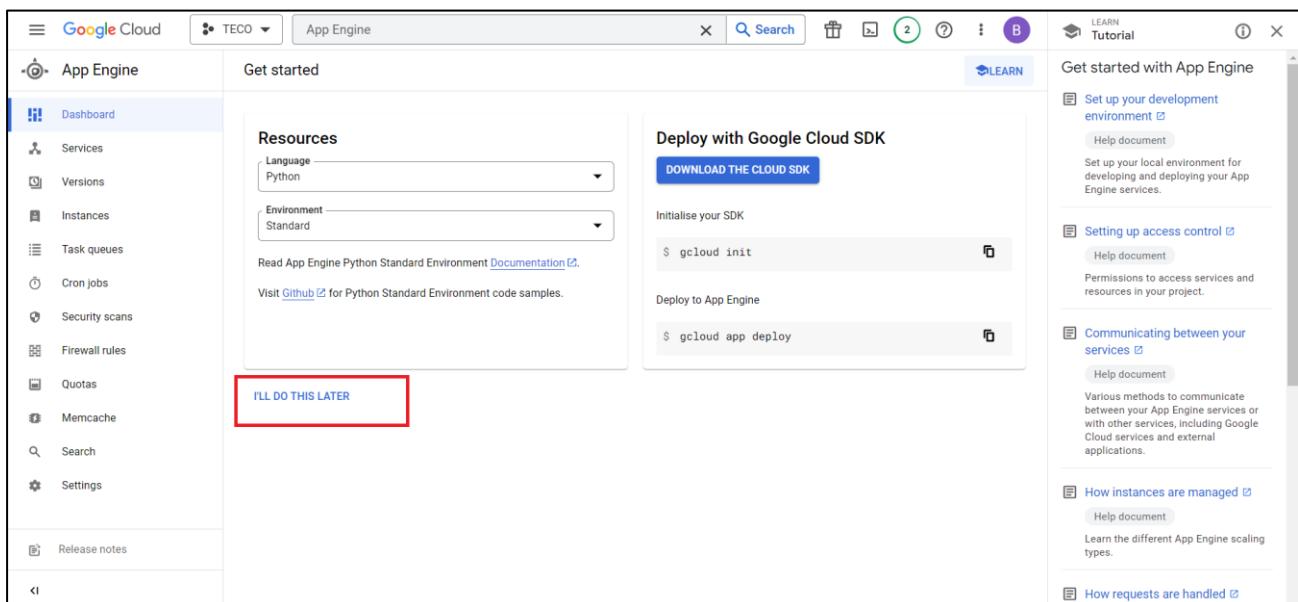
- Click on “CREATE APPLICATION”:

The screenshot shows the 'App Engine' dashboard for project 'TECO'. The left sidebar lists various management options: Dashboard, Services, Versions, Instances, Task queues, Cron jobs, Security scans, Firewall rules, Quotas, Memcache, Search, Settings, and Release notes. The 'Dashboard' item is currently selected. The main content area is titled 'Welcome to App Engine' with the sub-instruction 'Build scalable apps in any language on Google's infrastructure'. A prominent blue 'CREATE APPLICATION' button is centered in this area and is highlighted with a red box. To the right of the main content, there is a 'LEARN' section with several expandable help documents: 'Get started with App Engine', 'Setting up your development environment', 'Setting up access control', 'Communicating between your services', 'How instances are managed', and 'How requests are handled'.

➤ Click on “Next”:



➤ Click on “I’LL DO THIS LATER”:



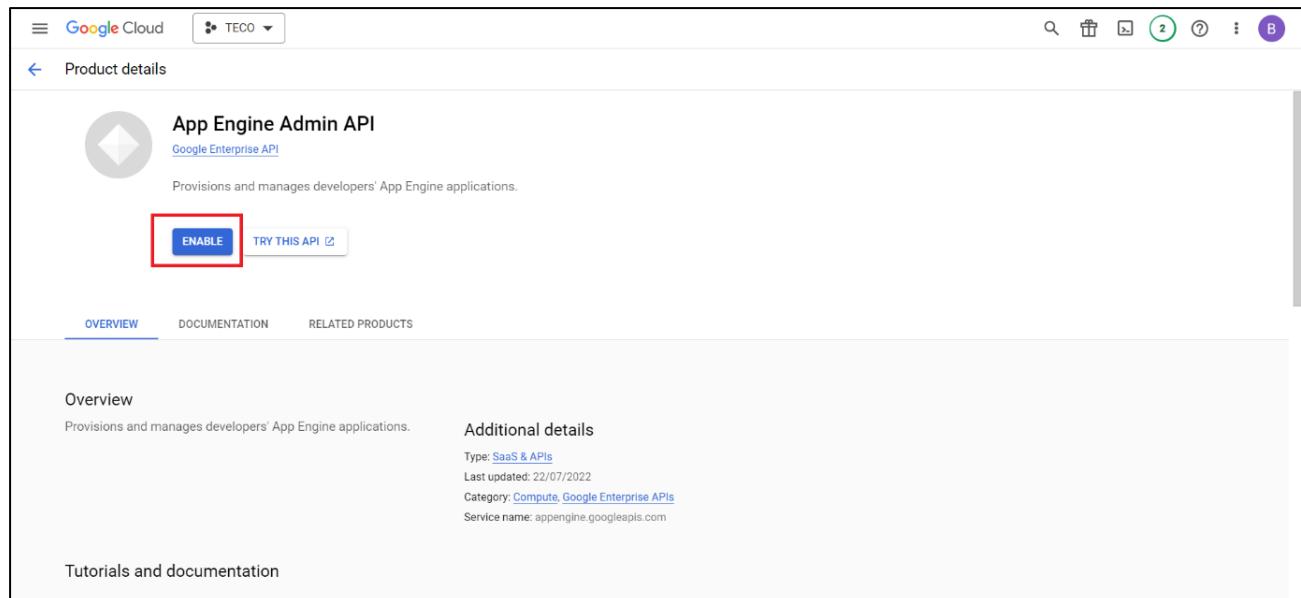
- Click on “GET STARTED”:

The screenshot shows the Google Cloud App Engine dashboard. On the left, there's a sidebar with various options like Dashboard, Services, Versions, Instances, Task queues, Cron jobs, Security scans, Firewall rules, Quotas, Memcache, Search, Settings, and Release notes. The main area has a title 'Welcome to App Engine' and a message: 'Your App Engine application has been created. Let us help you deploy your application by pointing you towards the relevant resources based on your programming language.' Below this is a blue 'GET STARTED' button, which is highlighted with a red box. To the right, there's a sidebar titled 'LEARN Tutorial' with sections for 'Get started with App Engine', 'Setting up access control', 'Communicating between your services', 'How instances are managed', and 'How requests are handled'. Each section includes a 'Help document' link.

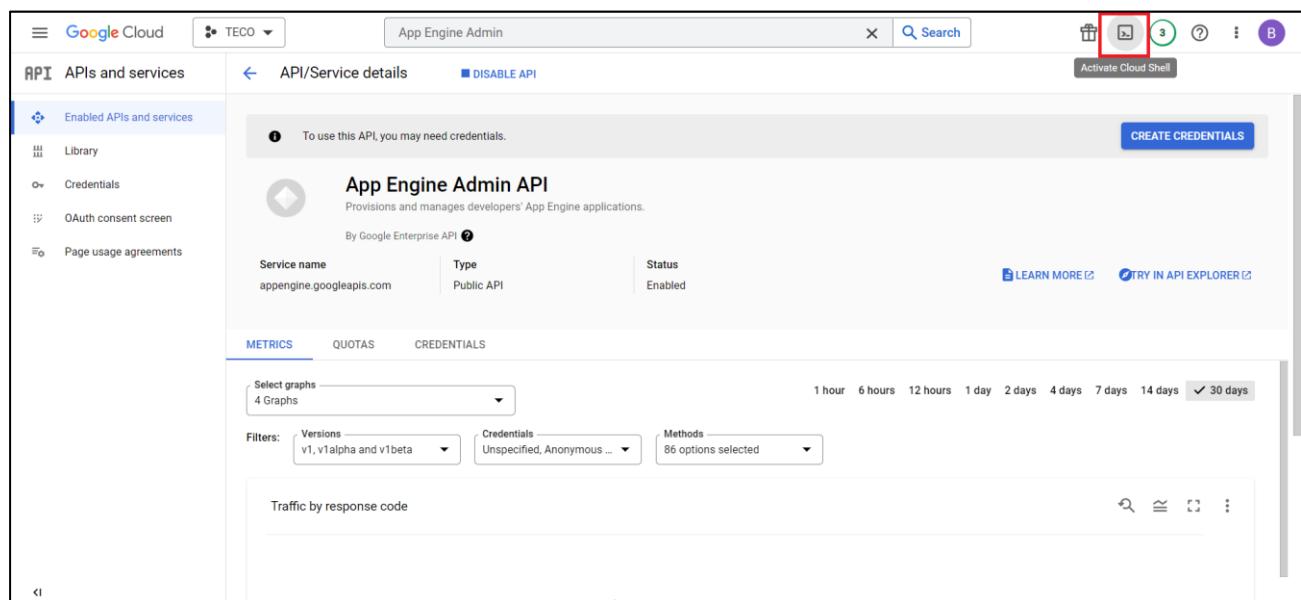
- Search “App Engine Admin API” and select it from the list:

The screenshot shows the Google Cloud App Engine Admin documentation page. The left sidebar is identical to the previous screenshot. The main content area has a 'DOCUMENTATION AND TUTORIALS' section with links to Roles that grant access to App Engine, Google App Engine Admin API documentation, and Controlling Access in the Admin API - App Engine. Below this is a 'MARKETPLACE' section with a list of APIs. The 'App Engine Admin API' is highlighted with a red box. Other items in the list include 'AI Platform' and 'App Engine'.

- Click on “Enable”:

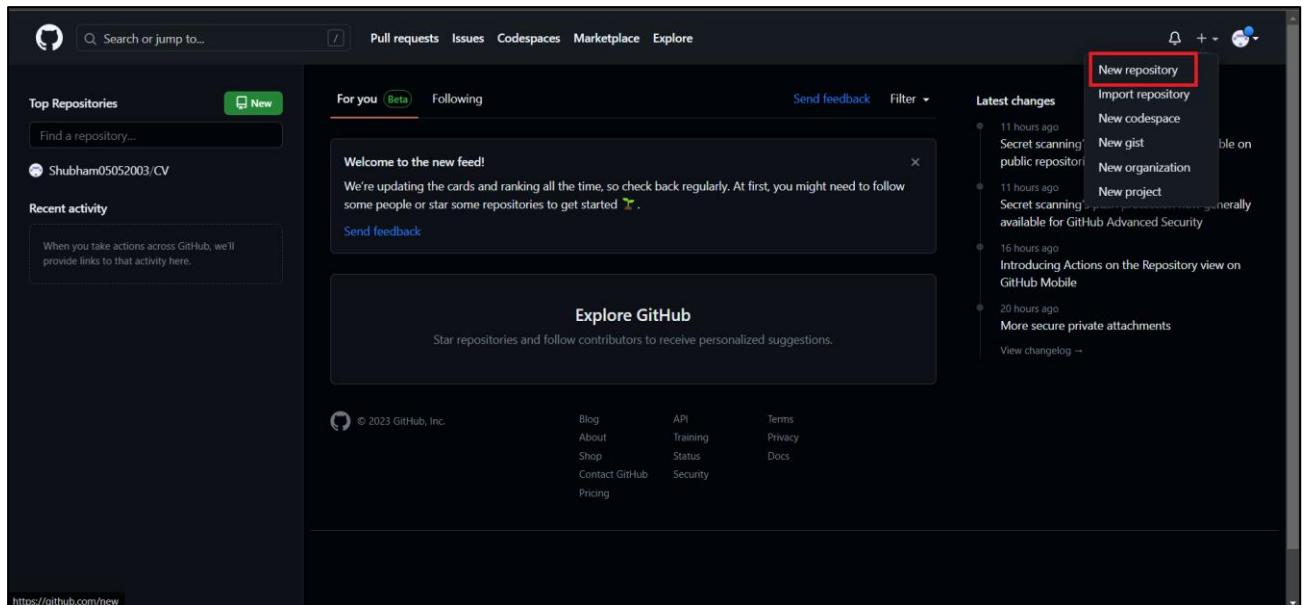


- Click on “Cloud Shell Icon” on the top right corner of the window:

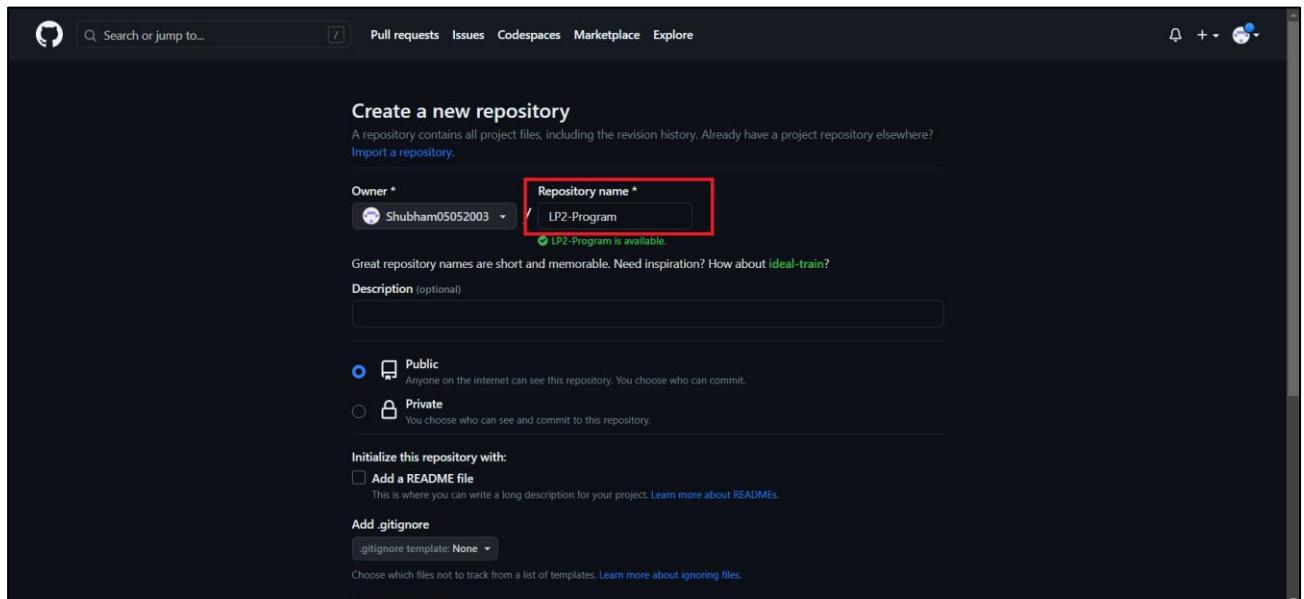


- Log in to GitHub (<https://github.com>) :

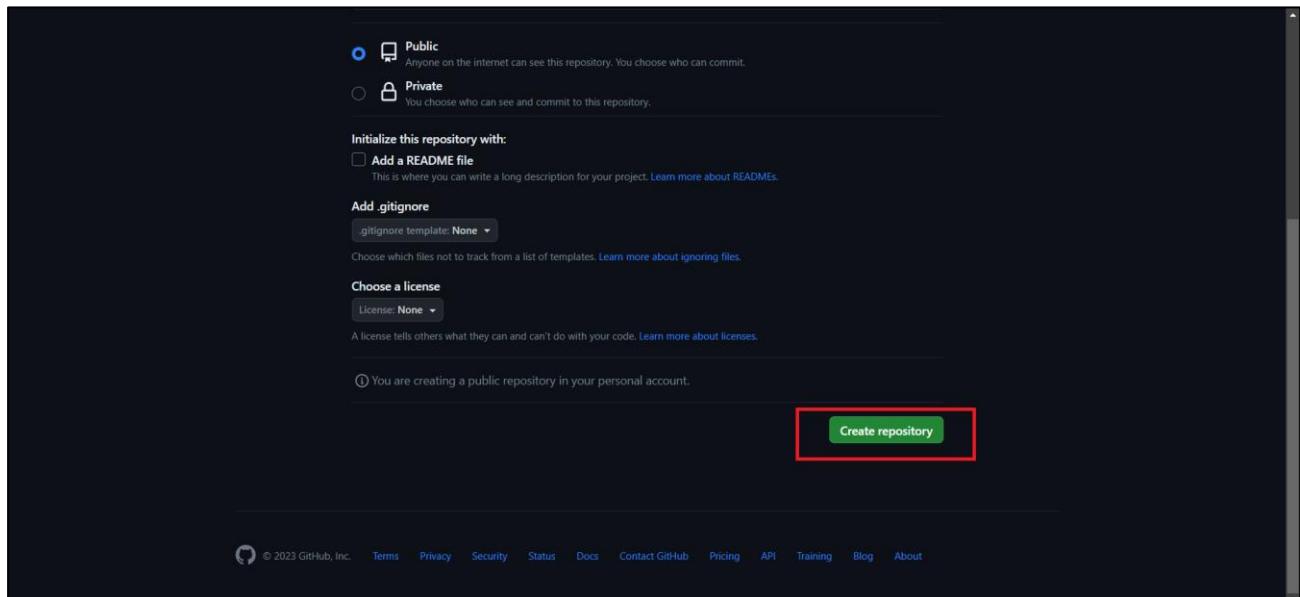
- Click on ‘+’ icon at top right corner and select “New Repository”:



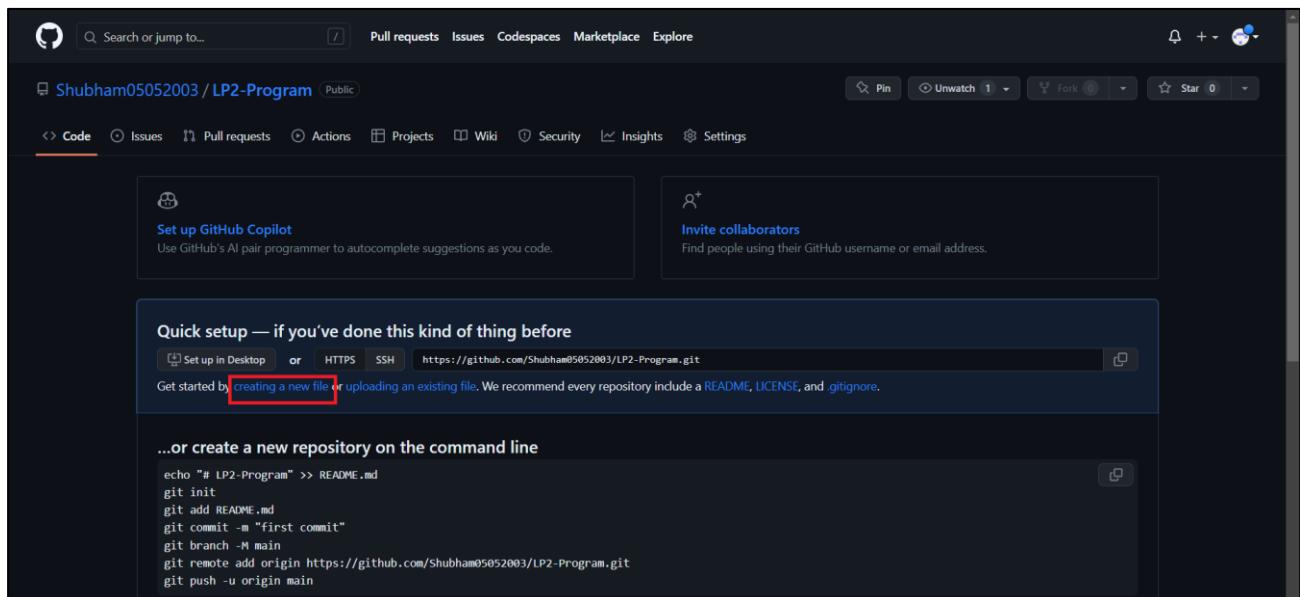
- Give a name to the repository:



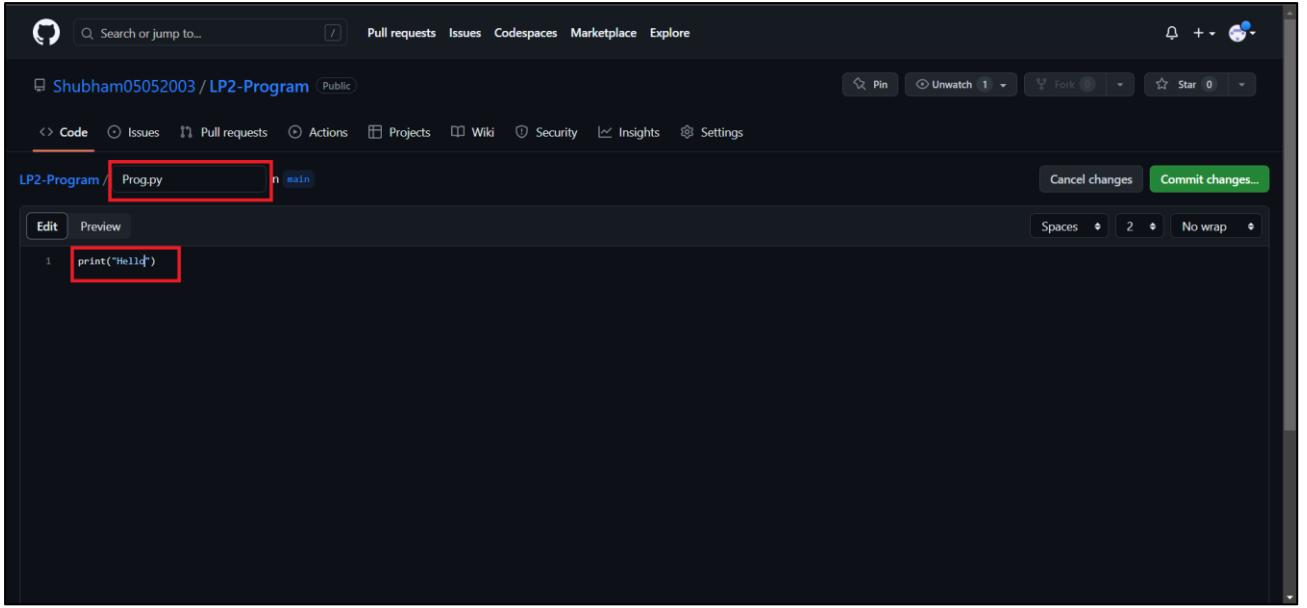
- Click on “Create Repository”:



- Click on “creating a new file”:



- Give a file name i.e “Prog.py” and write Python code in it i.e print(“Hello”):

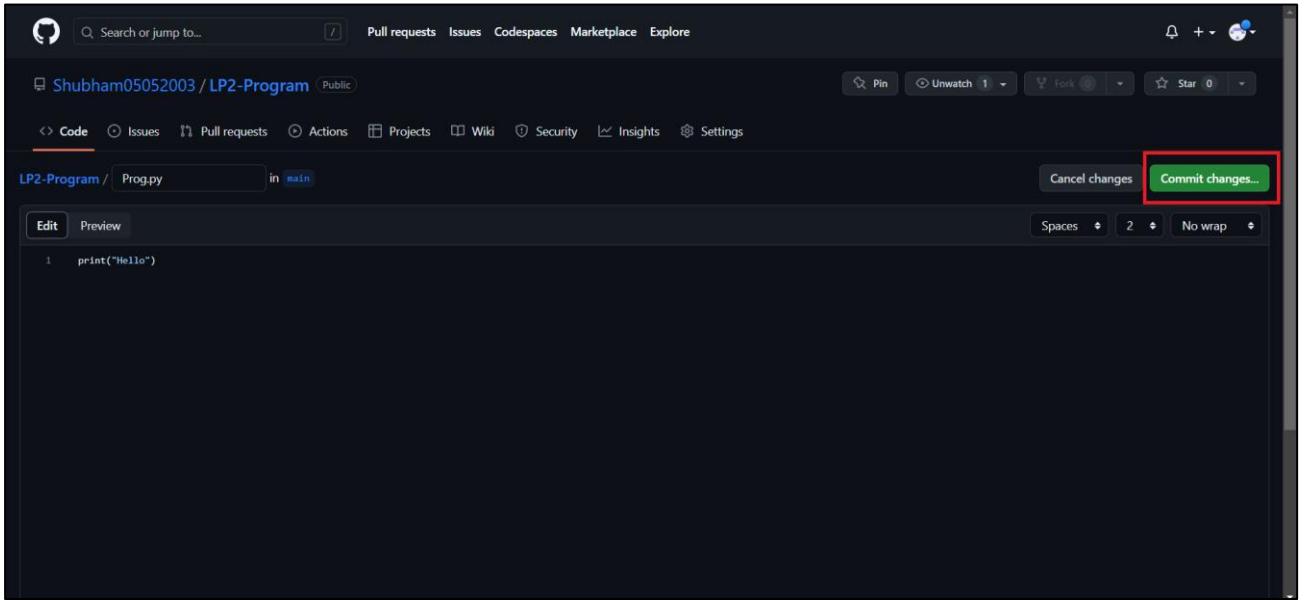


A screenshot of the GitHub Code editor interface. The repository is "Shubham05052003 / LP2-Program". The "Code" tab is selected. A file named "Prog.py" is open, showing the following content:

```
1 print("Hello")
```

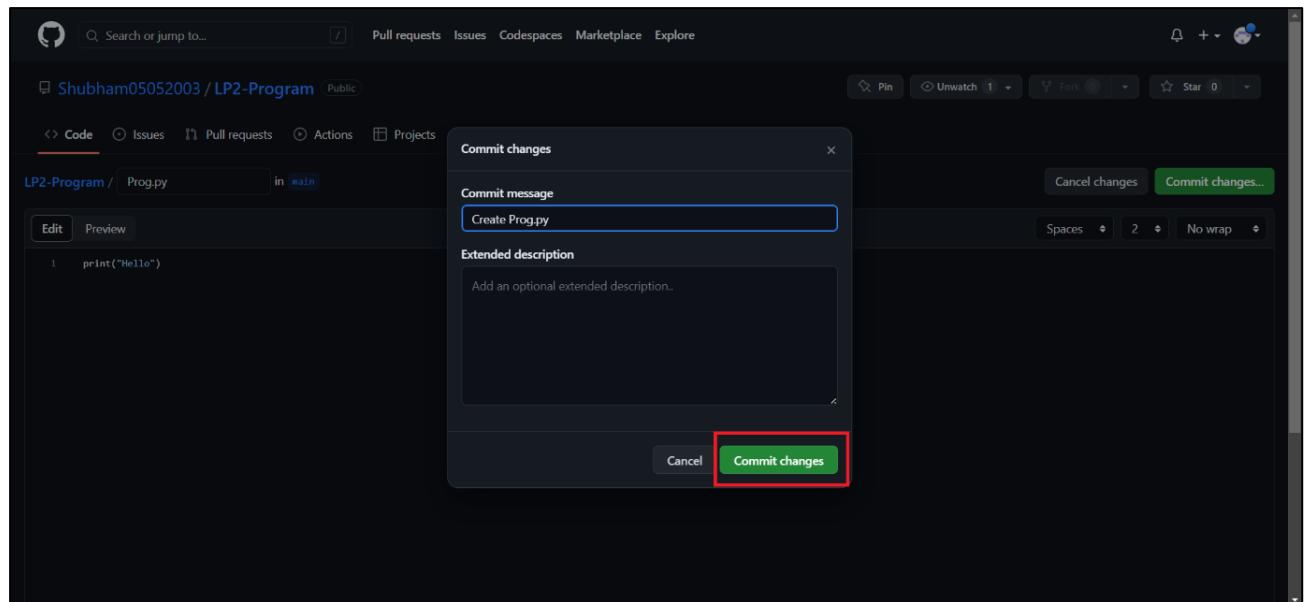
The line "print("Hello")" is highlighted with a red box. The status bar at the bottom right shows "Spaces 2 No wrap".

- Click on “Commit changes”:

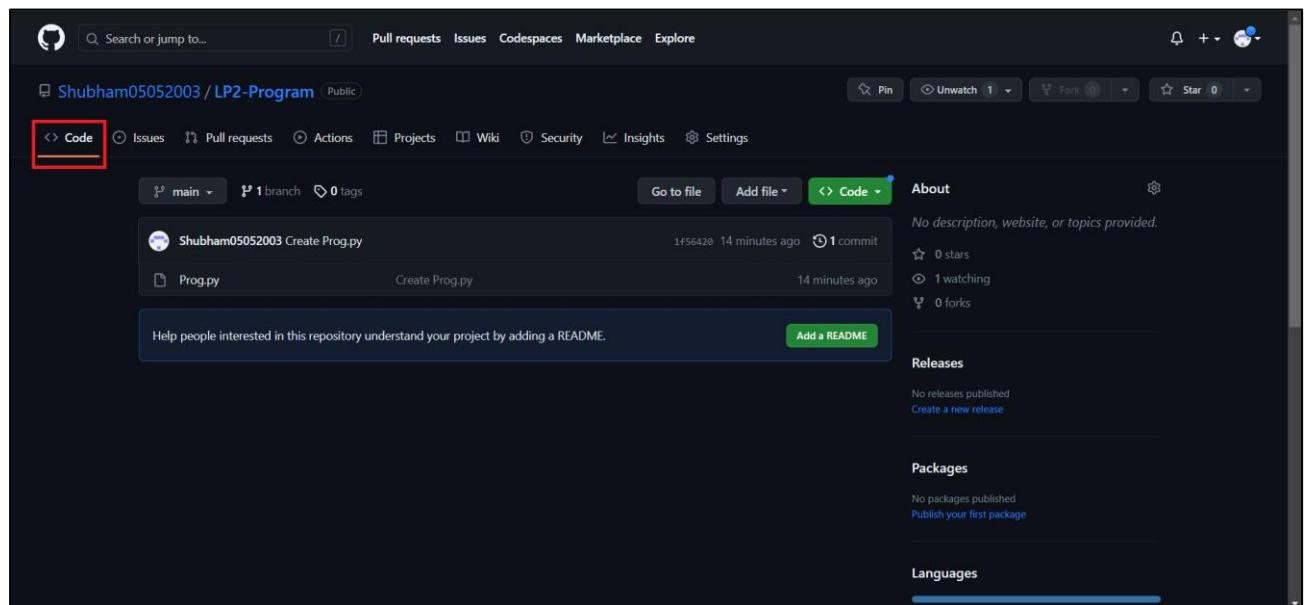


A screenshot of the GitHub Code editor interface, identical to the previous one but with a red box highlighting the "Commit changes..." button in the top right corner of the editor toolbar.

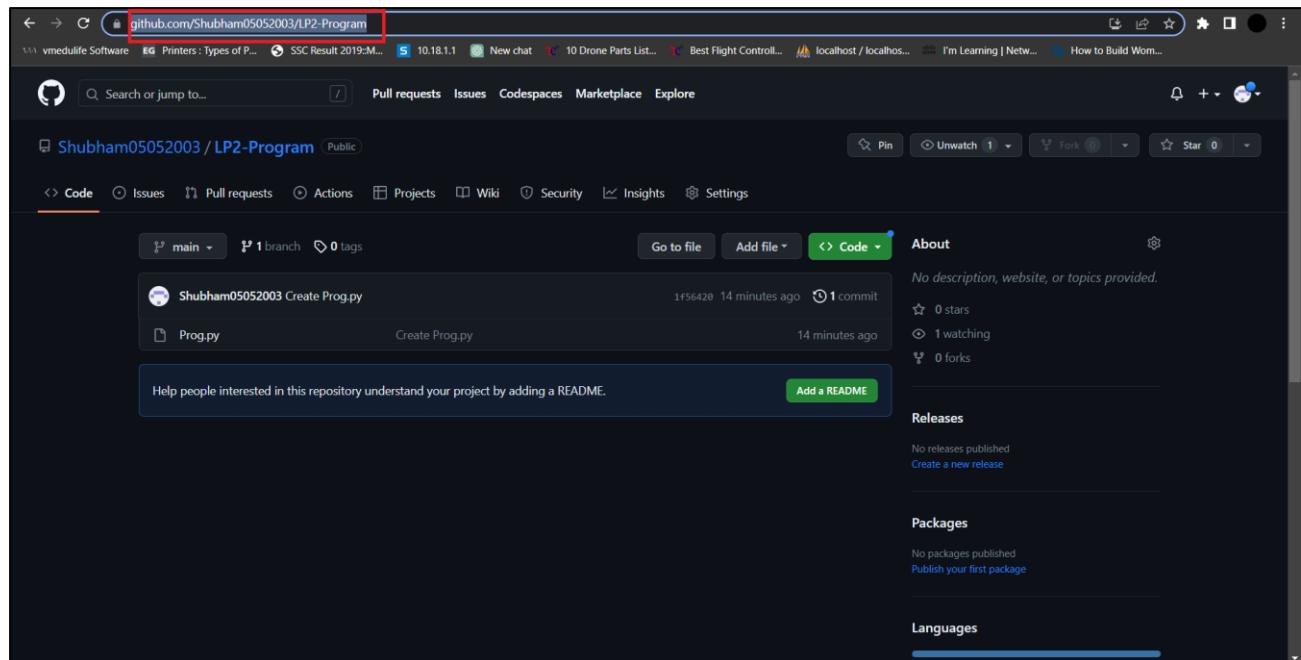
- Again, click on "Commit changes":



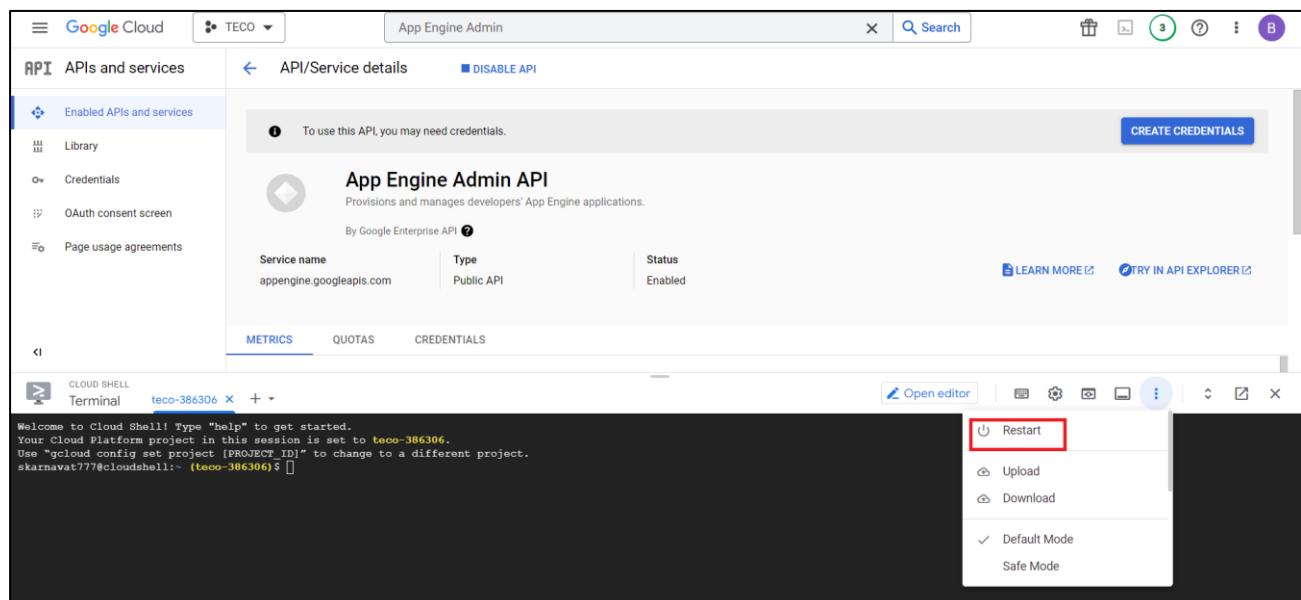
- Go to the code section at the top left corner:



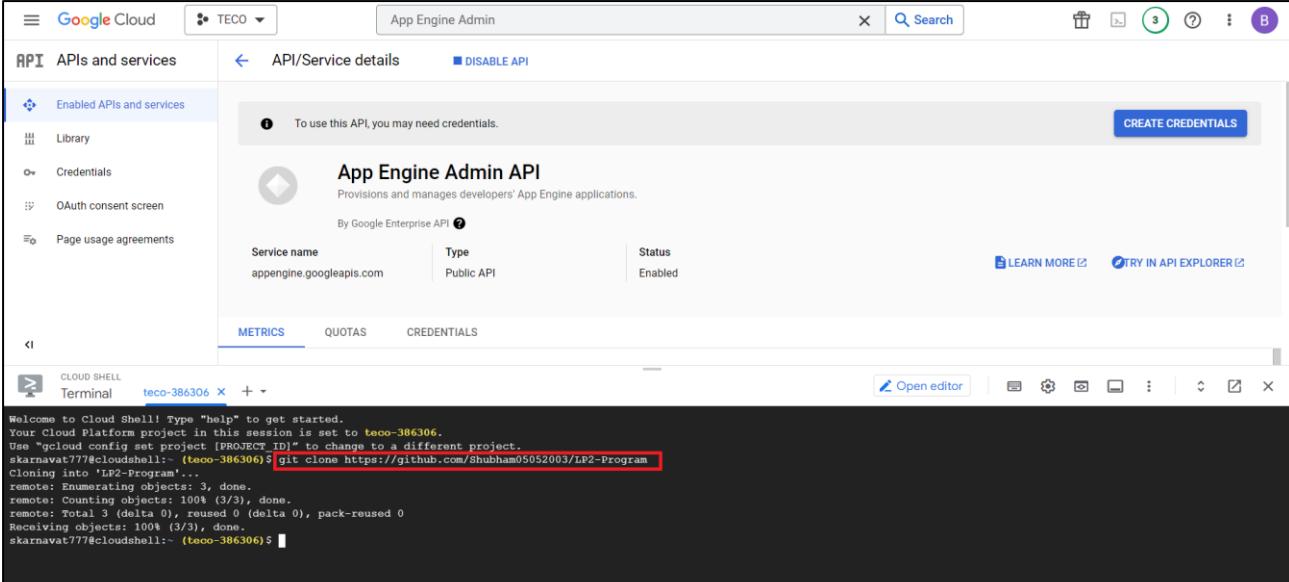
- Copy the URL from the address bar:



- Again, go to the Google Cloud Platform and if any problem occurs in a shell then “Restart” it:



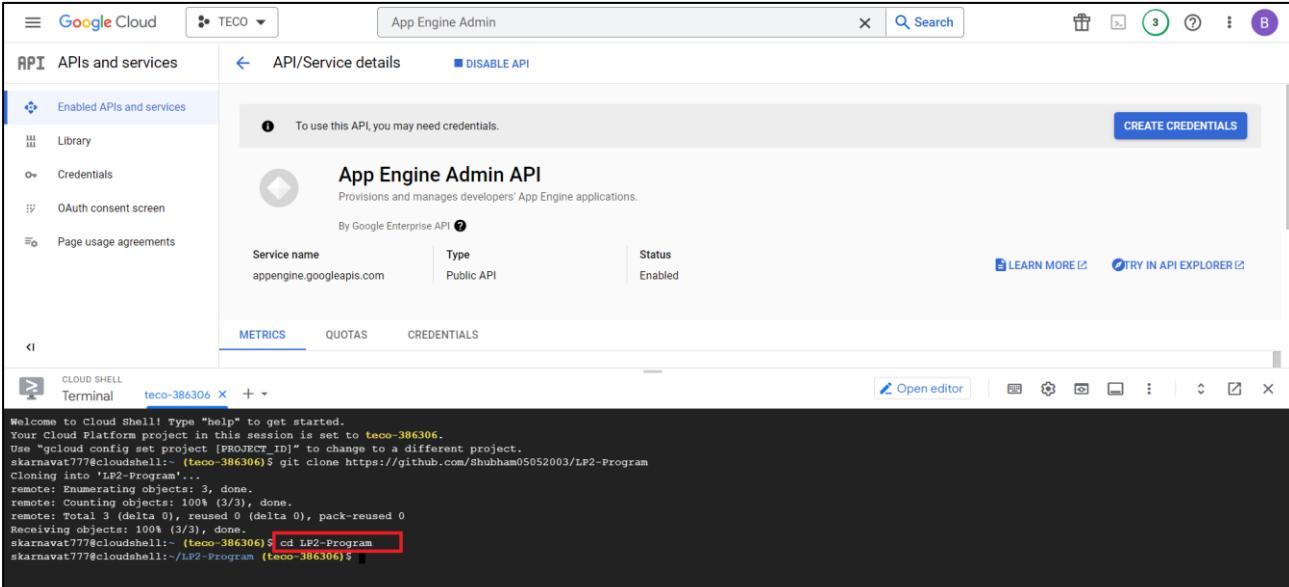
- Type the “git clone” command and paste the URL:



The screenshot shows the Google Cloud Platform API Admin interface. On the left, under 'Enabled APIs and services', the 'App Engine Admin API' is listed. It has a status of 'Enabled', is a 'Public API', and its service name is 'appengine.googleapis.com'. Below this, there are tabs for 'METRICS', 'QUOTAS', and 'CREDENTIALS'. At the bottom, a terminal window titled 'TECO' shows the command 'git clone https://github.com/Shubham05052003/LP2-Program' being run.

```
Welcome to Cloud Shell! Type "help" to get started.
Your Cloud Platform project in this session is set to teco-386306.
Use "gcloud config set project [PROJECT_ID]" to change to a different project.
skarnavat777@cloudshell:~ (teco-386306)$ git clone https://github.com/Shubham05052003/LP2-Program
Cloning into 'LP2-Program'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.
skarnavat777@cloudshell:~ (teco-386306)$
```

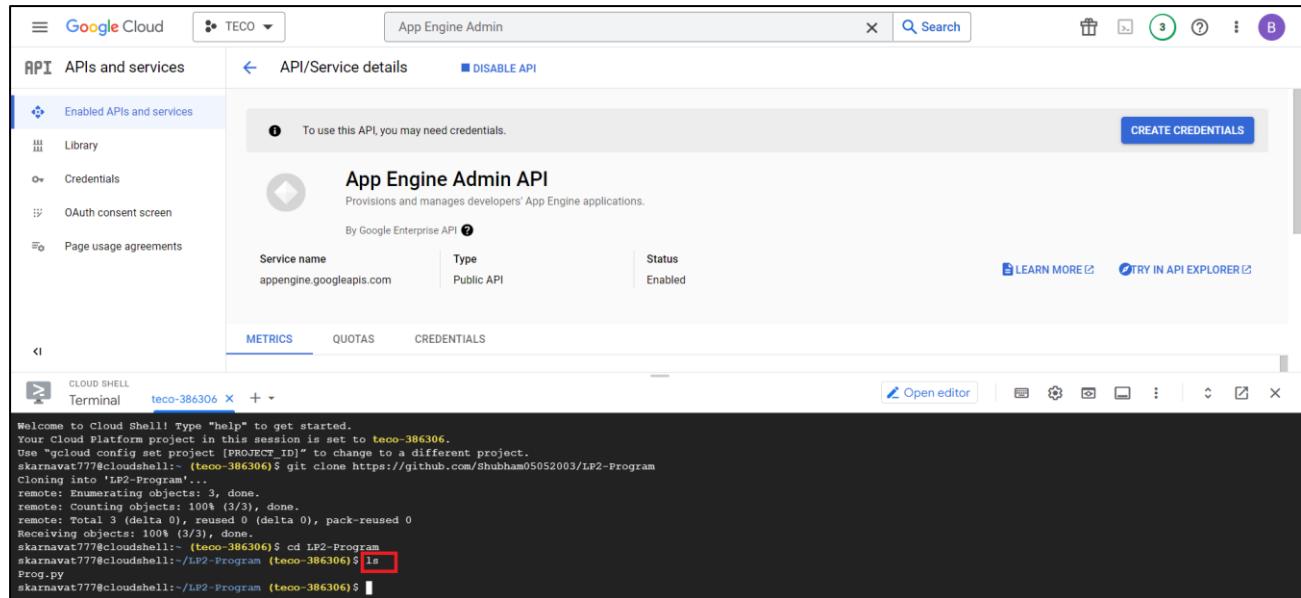
- Change the directory with the help of the “cd” command and enter the repository name i.e “LP2-Program”:



The screenshot shows the Google Cloud Platform API Admin interface, identical to the previous one. The terminal window now shows the command 'cd LP2-Program' being run, indicating the user has changed into the directory of the cloned repository.

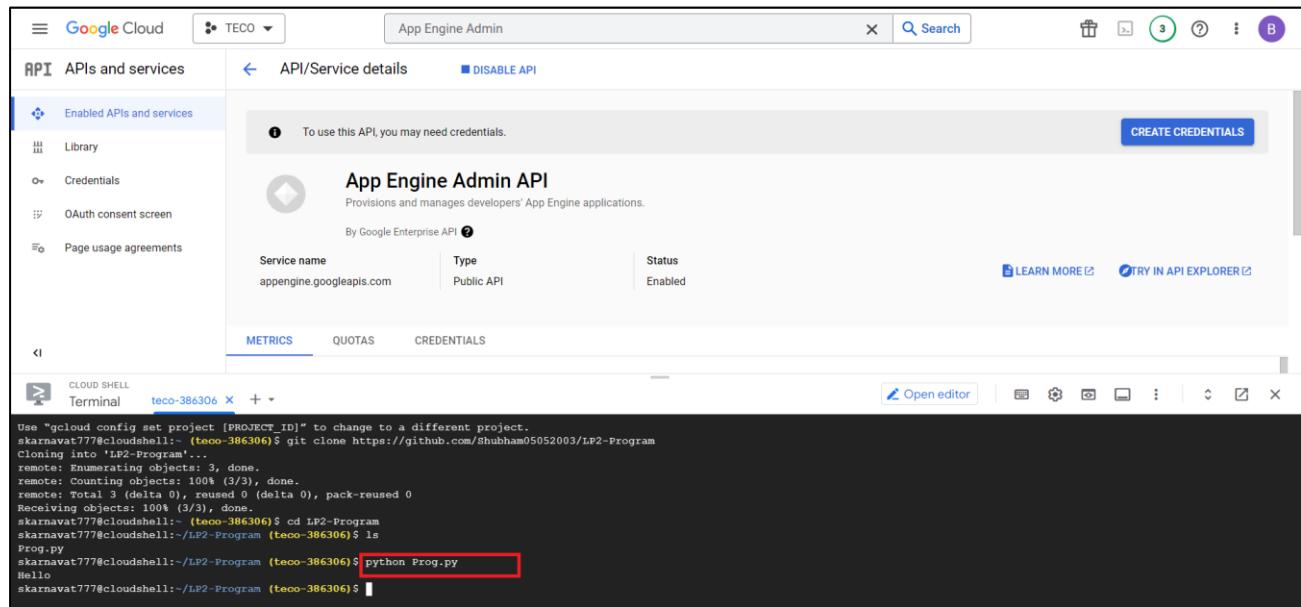
```
Welcome to Cloud Shell! Type "help" to get started.
Your Cloud Platform project in this session is set to teco-386306.
Use "gcloud config set project [PROJECT_ID]" to change to a different project.
skarnavat777@cloudshell:~ (teco-386306)$ git clone https://github.com/Shubham05052003/LP2-Program
Cloning into 'LP2-Program'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.
skarnavat777@cloudshell:~ (teco-386306)$ cd LP2-Program
skarnavat777@cloudshell:~/LP2-Program (teco-386306)$
```

- Enter the command “ls” to get the file name:



Welcome to Cloud Shell! Type "Help" to get started.
Your Cloud Platform project in this session is set to **teco-386306**.
Use "gcloud config set project [PROJECT_ID]" to change to a different project.
skarnavat777@cloudshell:~ (teco-386306)\$ git clone https://github.com/Shubham05052003/LP2-Program
Cloning into 'LP2-Program'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.
skarnavat777@cloudshell:~ (teco-386306)\$ cd LP2-Program
skarnavat777@cloudshell:~/LP2-Program (teco-386306)\$ ls
Prog.py
skarnavat777@cloudshell:~/LP2-Program (teco-386306)\$

- Enter the command “python” and the “file name” to get output i.e python Prog.py:

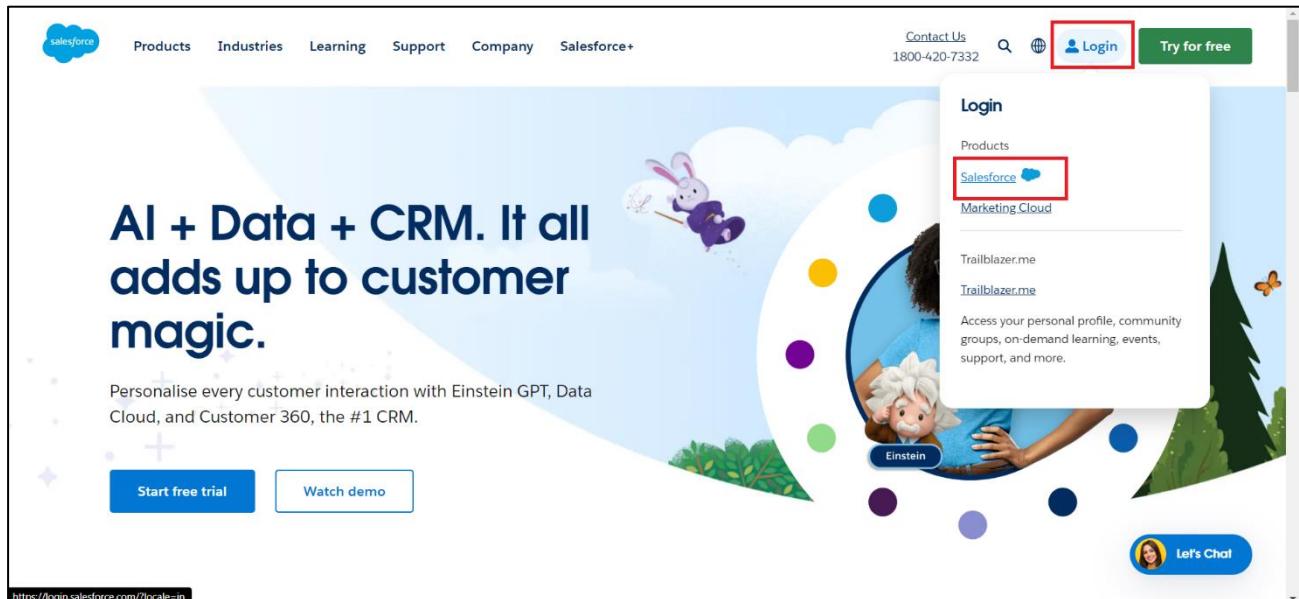


Use "gcloud config set project [PROJECT_ID]" to change to a different project.
skarnavat777@cloudshell:~ (teco-386306)\$ git clone https://github.com/Shubham05052003/LP2-Program
Cloning into 'LP2-Program'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.
skarnavat777@cloudshell:~ (teco-386306)\$ cd LP2-Program
skarnavat777@cloudshell:~/LP2-Program (teco-386306)\$ ls
Prog.py
skarnavat777@cloudshell:~/LP2-Program (teco-386306)\$ Python Prog.py
Hello
skarnavat777@cloudshell:~/LP2-Program (teco-386306)\$

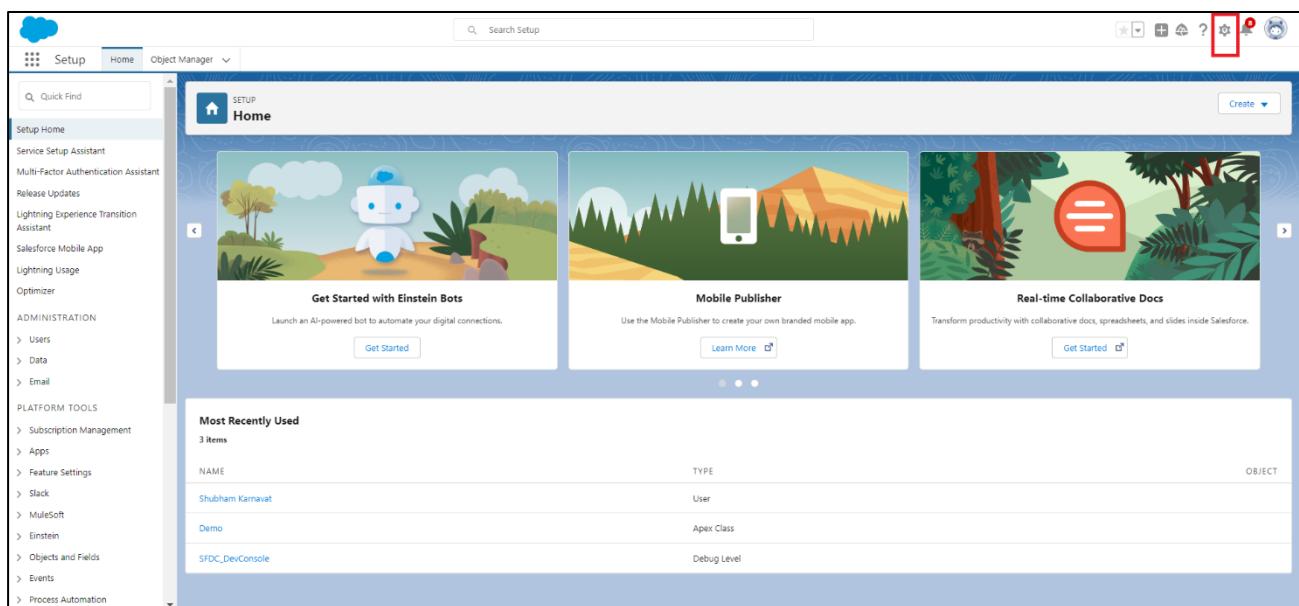
Practical Number :9

Title: Creating an application in SalesForce.com using Apex programming language

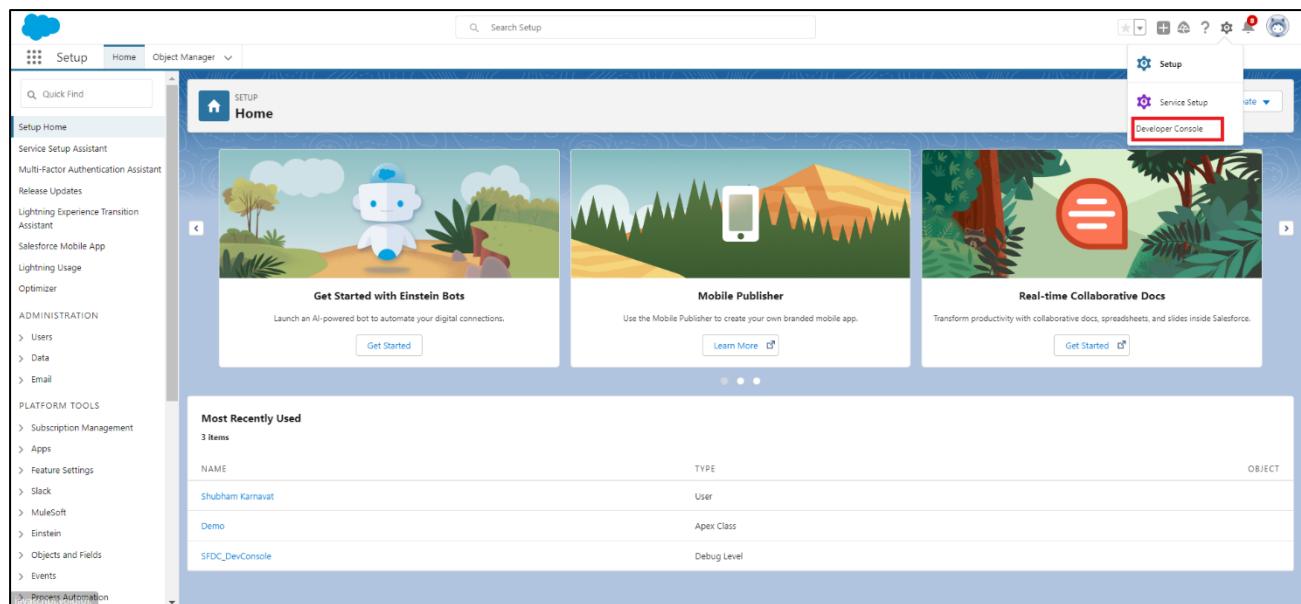
- Go to SalesForces.com (<https://www.salesforce.com/in/>) & Login using the credentials:



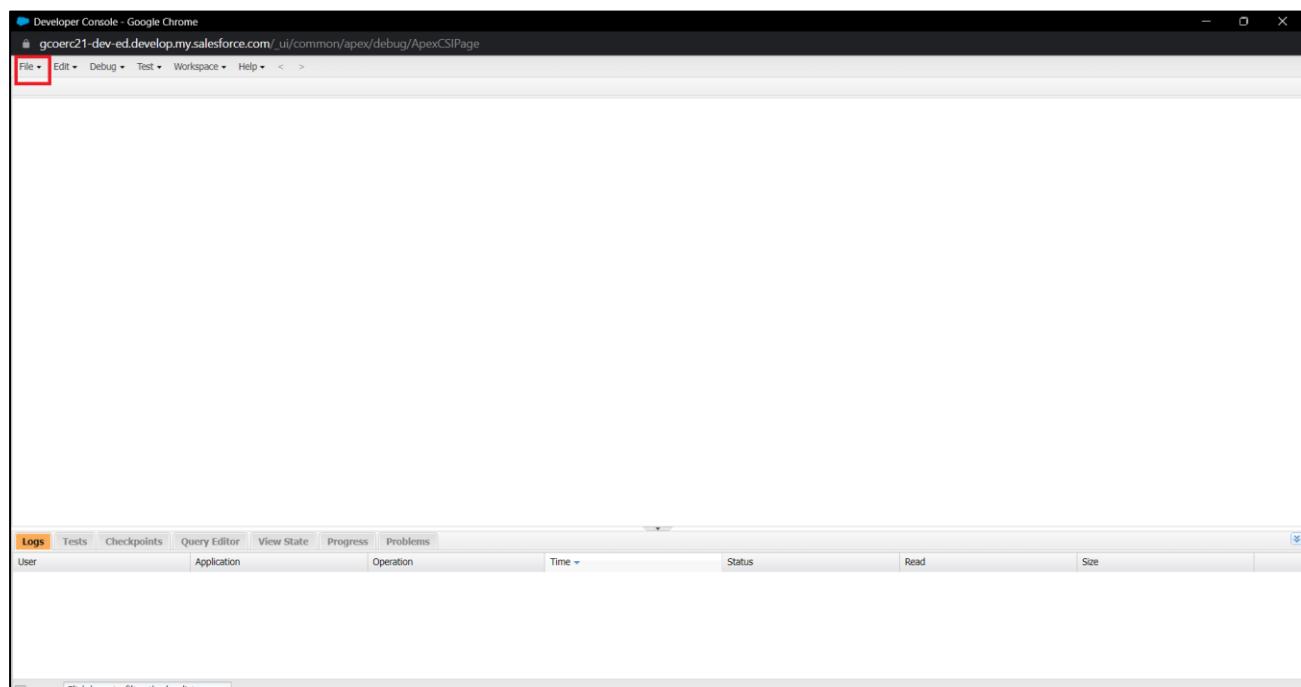
- Click on “Setting Icon” at the top right corner:



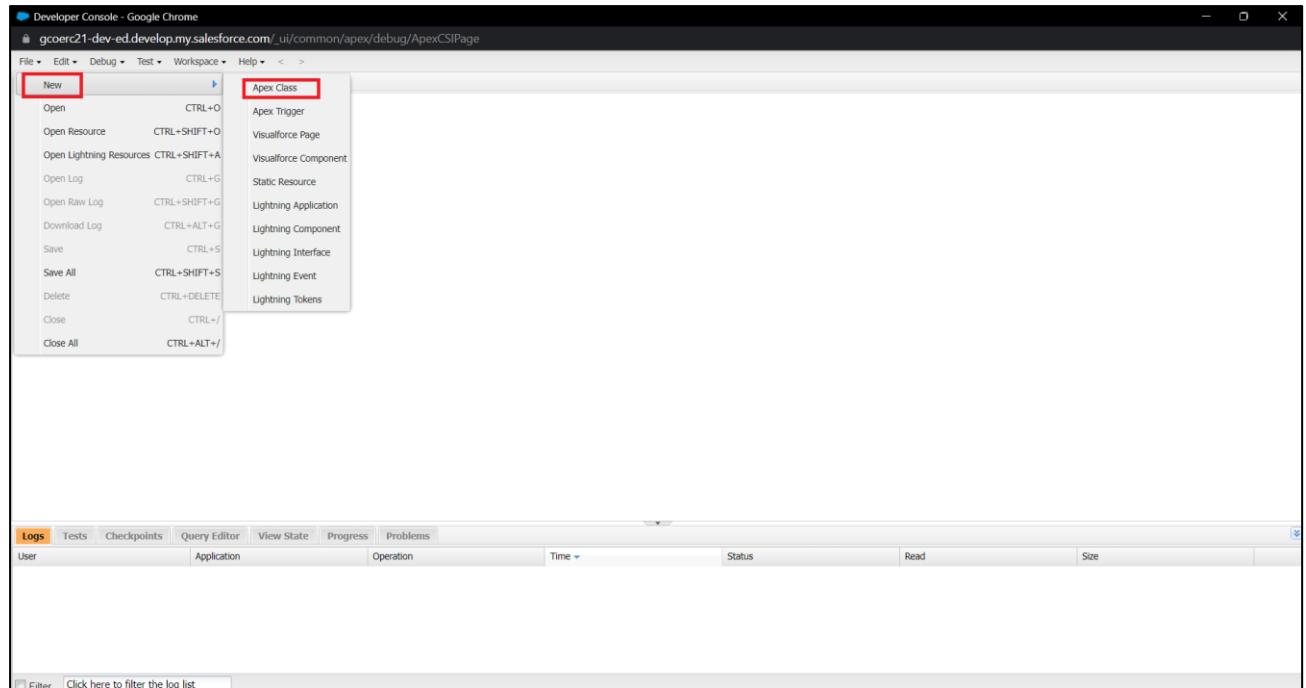
- Select “Developer Console” from the list:



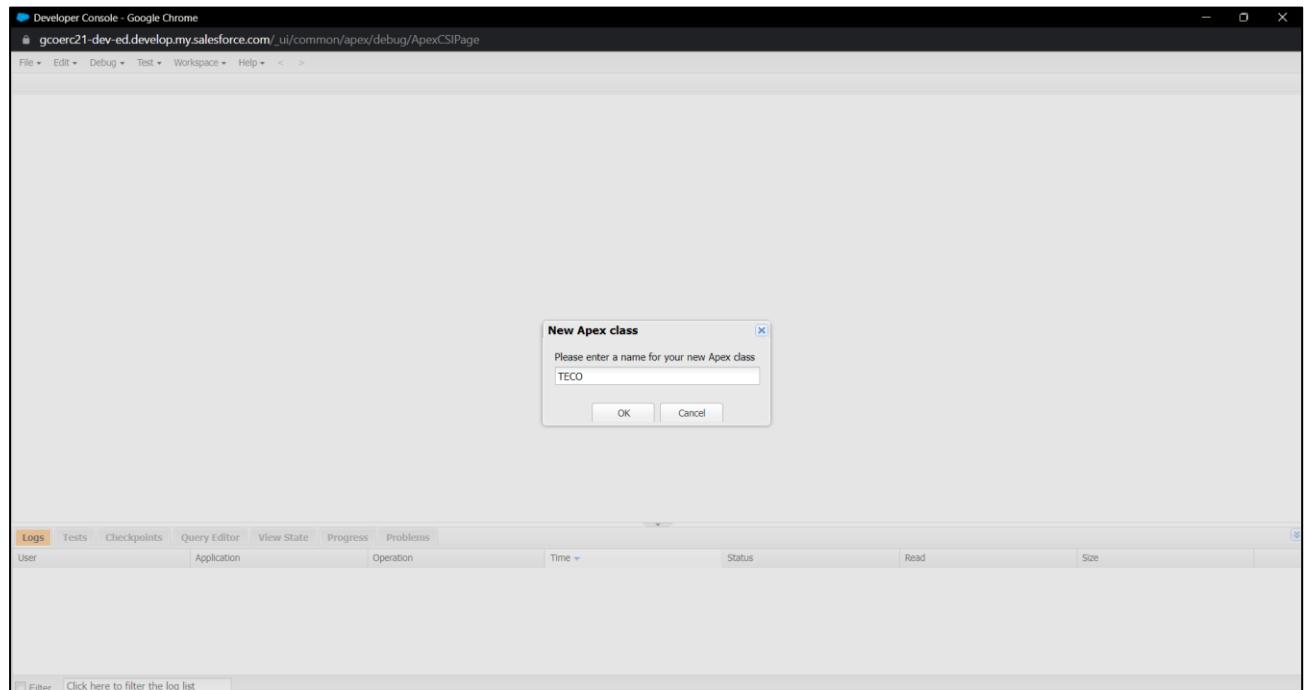
- Go to “File”:



➤ Select New => Apex Class:



➤ Enter the name of the Apex class:



➤ Write code in the console:

The screenshot shows the Salesforce Developer Console in Google Chrome. The URL is `gcoerc21-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage`. The page title is "Developer Console - Google Chrome". The main content area displays the Apex code for the TECO class:

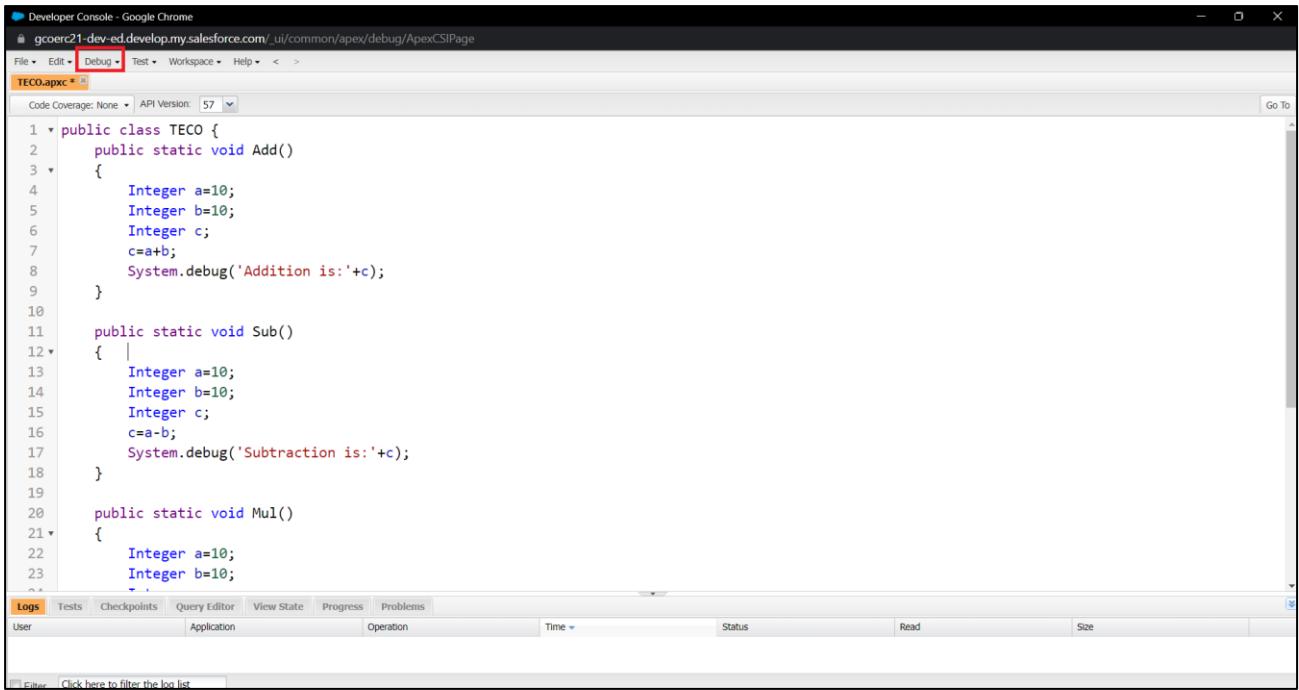
```
1 * public class TECO {
2     public static void Add()
3     {
4         Integer a=10;
5         Integer b=10;
6         Integer c;
7         c=a+b;
8         System.debug('Addition is:' +c);
9     }
10    public static void Sub()
11    {
12        Integer a=10;
13        Integer b=10;
14        Integer c;
15        c=a-b;
16        System.debug('Subtraction is:' +c);
17    }
18    public static void Mul()
19    {
20        Integer a=10;
21        Integer b=10;
22        Integer c;
23        c=a*b;
24        System.debug('Multiplication is:' +c);
25    }
26    public static void Div()
27    {
28        Integer a=10;
29        Integer b=10;
30        Integer c;
31        c=a/b;
32        System.debug('Division is:' +c);
33    }
34}
```

The code includes several debug statements using `System.debug()`. The developer console interface at the bottom shows tabs for Logs, Tests, Checkpoints, Query Editor, View State, Progress, and Problems. The Logs tab is selected.

➤ Save the code before executing:

The screenshot shows the Salesforce Developer Console in Google Chrome. The URL is `gcoerc21-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage`. The main content area displays the Apex code for the TECO class, identical to the previous screenshot. However, the "File" menu is open, and the "Save" option is highlighted with a red box. The keyboard shortcut `CTRL+S` is displayed next to it. The developer console interface at the bottom shows tabs for Logs, Tests, Checkpoints, Query Editor, View State, Progress, and Problems. The Logs tab is selected.

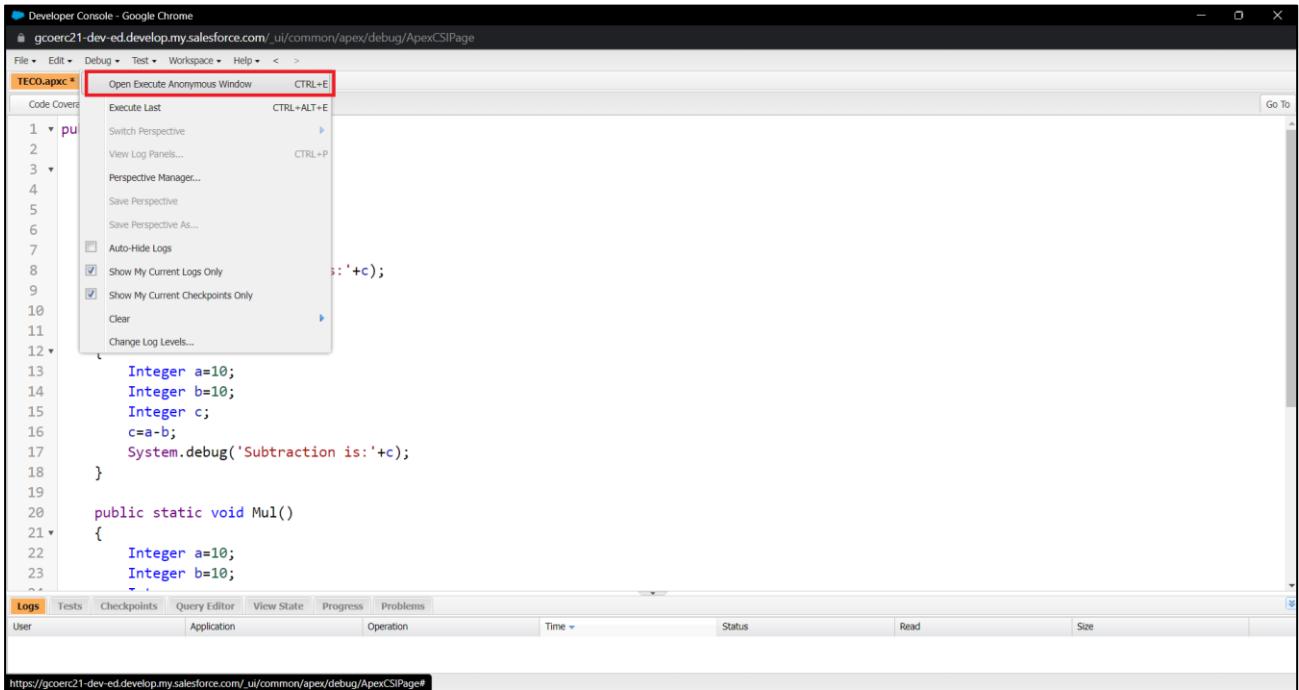
➤ Click on “Debug”:



The screenshot shows the Salesforce Developer Console interface in Google Chrome. The URL is https://gcoerc21-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage. The 'Logs' tab is active. The 'Debug' menu item in the top navigation bar is highlighted with a red box. The code editor window contains a class named TECO.apxc with three methods: Add(), Sub(), and Mul(). Each method contains logic to add, subtract, or multiply two integers and output the result via System.debug.

```
1 public class TECO {
2     public static void Add()
3     {
4         Integer a=10;
5         Integer b=10;
6         Integer c;
7         c=a+b;
8         System.debug('Addition is:'+c);
9     }
10
11    public static void Sub()
12    {
13        Integer a=10;
14        Integer b=10;
15        Integer c;
16        c=a-b;
17        System.debug('Subtraction is:'+c);
18    }
19
20    public static void Mul()
21    {
22        Integer a=10;
23        Integer b=10;
24    }
25 }
```

➤ Select “Open Execute Anonymous Window” from the list:



The screenshot shows the Salesforce Developer Console interface in Google Chrome. The URL is https://gcoerc21-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage. The 'Logs' tab is active. The 'Debug' menu is open, and the 'Open Execute Anonymous Window' option is highlighted with a red box. This option is part of a larger dropdown menu that includes other items like 'Execute Last' and 'Perspective Manager...'. The code editor window contains the same TECO.apxc class as the previous screenshot.

```
1 public class TECO {
2     public static void Add()
3     {
4         Integer a=10;
5         Integer b=10;
6         Integer c;
7         c=a+b;
8         System.debug('Addition is:'+c);
9     }
10
11    public static void Sub()
12    {
13        Integer a=10;
14        Integer b=10;
15        Integer c;
16        c=a-b;
17        System.debug('Subtraction is:'+c);
18    }
19
20    public static void Mul()
21    {
22        Integer a=10;
23        Integer b=10;
24    }
25 }
```

- Call the function with the help of the class name:

The screenshot shows the Salesforce Developer Console interface. On the left, there is a code editor window titled "TECO.apxc" containing the following Apex code:

```

1 public class TECO {
2     public static void Add()
3     {
4         Integer a=10;
5         Integer b=10;
6         Integer c;
7         c=a+b;
8         System.debug('Addition is:'+c);
9     }
10    public static void Sub()
11    {
12        Integer a=10;
13        Integer b=10;
14        Integer c;
15        c=a-b;
16        System.debug('Subtraction is:'+c);
17    }
18    public static void Mul()
19    {
20        Integer a=10;
21        Integer b=10;
22    }

```

In the center, a modal dialog box titled "Enter Apex Code" displays the following code:

```

1 TECO.Add();
2 TECO.Sub();
3 TECO.Mul();
4 TECO.Div();

```

At the bottom of the dialog box are three buttons: "Open Log", "Execute", and "Execute Highlighted". The "Execute" button is highlighted with a red box.

- Tick on “Open log” and click on “Execute”:

This screenshot is similar to the previous one, showing the Developer Console and the "Enter Apex Code" dialog box. The difference is that the "Open Log" checkbox at the bottom of the dialog box is now checked (indicated by a blue checkmark). The "Execute" button remains highlighted with a red box.

- Log Window will appear:

Developer Console - Google Chrome
gcoerc21-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCsIPage

File ▾ Edit ▾ Debug ▾ Test ▾ Workspace ▾ Help ▾ < >
TECO.apxc Log executeAnonymous @5/10/2023, 2:30:11 PM

Execution Log

Timestamp	Event	Details
14:30:11:001	USER_INFO	[EXTERNAL][00551000006aQ2P gcoerc58@gcoerc.com (GMT+05:30) India Standard Time (Asia/Kolkata) GMT+05:30]
14:30:11:001	EXECUTION_ST...	
14:30:11:001	CODE_UNIT_ST...	[EXTERNAL]execute_anonymous_apex
14:30:11:001	HEAP_ALLOCATE	[79]Bytes:3
14:30:11:001	HEAP_ALLOCATE	[84]Bytes:152
14:30:11:001	HEAP_ALLOCATE	[399]Bytes:408
14:30:11:001	HEAP_ALLOCATE	[412]Bytes:408
14:30:11:001	HEAP_ALLOCATE	[520]Bytes:48
14:30:11:001	HEAP_ALLOCATE	[139]Bytes:6
14:30:11:001	HEAP_ALLOCATE	[EXTERNAL]Bytes:7
14:30:11:001	STATEMENT_EX...	[1]
14:30:11:001	STATEMENT_EX...	[1]
14:30:11:001	HEAP_ALLOCATE	[52]Bytes:5
14:30:11:001	HEAP_ALLOCATE	[58]Bytes:5
14:30:11:001	HEAP_ALLOCATE	[66]Bytes:7
14:30:11:001	SYSTEM_MODE...	false
14:30:11:001	HEAP_ALLOCATE	[1]Bytes:5
14:30:11:013	HEAP_ALLOCATE	[1]Bytes:10
14:30:11:013	HEAP_ALLOCATE	[1]Bytes:10
14:30:11:013	METHOD_ENTRY	[1]@01p5000000Pk1JN[TECO.TECO()]
14:30:11:013	STATEMENT_EX...	[1]
14:30:11:013	STATEMENT_EX...	[1]
14:30:11:013	METHOD_EXIT	[1]@TECO
14:30:11:013	METHOD_ENTRY	[1]@01p5000000Pk1JN[TECO.Add()]
14:30:11:013	STATEMENT_EX...	[4]
14:30:11:013	STATEMENT_EX...	[5]

This Frame Executable Debug Only Filter [Click here to filter the log](#)

Logs Tests Checkpoints Query Editor View State Progress Problems

User	Application	Operation	Time	Status	Read	Size
Shubham Karnavat	Unknown	/services/data/v57.0/tooling/executeAn...	5/10/2023, 2:30:11 PM	Success		7.39 KB

Filter [Click here to filter the log list](#)

- Tick on “Debug Only”:

The screenshot shows the Google Chrome Developer Console with the title bar "Developer Console - Google Chrome" and the URL "gcoerc21-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage". The tab "TECO.apxc" is selected. Below the tabs, the title "Log executeAnonymous @5/10/2023, 2:30:11 PM" is displayed. The main content area is titled "Execution Log" and contains a table of log entries. The first few entries are highlighted with a light blue background. The columns are "Timestamp", "Event", and "Details". The log entries show various system events such as USER_INFO, EXECUTION_STARTED, and HEAP_ALLOCATE. The last few entries are standard log entries from the Apex code. At the bottom, there are checkboxes for "This Frame", "Executable", and "Debug Only" (which is checked), and a "Filter" button. The "Logs" tab is currently selected. A red box highlights the "Debug Only" checkbox.

Timestamp	Event	Details
14:30:11:001	USER_INFO	[EXTERNAL][0055 000006xQ2P gcoerc58@gcoerc.com (GMT+05:30) India Standard Time (Asia/Kolkata) GMT+05:30]
14:30:11:001	EXECUTION_STARTED	
14:30:11:001	CODE_UNIT_STARTED	[EXTERNAL]execute_anonymous_apex
14:30:11:001	HEAP_ALLOCATE	[79]Bytes:3
14:30:11:001	HEAP_ALLOCATE	[84]Bytes:152
14:30:11:001	HEAP_ALLOCATE	[399]Bytes:408
14:30:11:001	HEAP_ALLOCATE	[412]Bytes:408
14:30:11:001	HEAP_ALLOCATE	[520]Bytes:48
14:30:11:001	HEAP_ALLOCATE	[139]Bytes:6
14:30:11:001	HEAP_ALLOCATE	[EXTERNAL]Bytes:7
14:30:11:001	STATEMENT_EXECUTE	[1]
14:30:11:001	STATEMENT_EXECUTE	[1]
14:30:11:001	HEAP_ALLOCATE	[52]Bytes:5
14:30:11:001	HEAP_ALLOCATE	[58]Bytes:5
14:30:11:001	HEAP_ALLOCATE	[66]Bytes:7
14:30:11:001	SYSTEM_MODE	false
14:30:11:001	HEAP_ALLOCATE	[1]Bytes:5
14:30:11:013	HEAP_ALLOCATE	[1]Bytes:10
14:30:11:013	HEAP_ALLOCATE	[1]Bytes:10
14:30:11:013	METHOD_ENTRY	[1]@0 p5000000Pk1IN[TECO.TECO()]
14:30:11:013	STATEMENT_EXECUTE	[1]
14:30:11:013	STATEMENT_EXECUTE	[1]
14:30:11:013	METHOD_EXIT	[1]TECO
14:30:11:013	METHOD_ENTRY	[1]@0 p5000000Pk1IN[TECO.Add()]
14:30:11:013	STATEMENT_EXECUTE	[4]
14:30:11:013	STATEMENT_EXECUTE	[5]

This Frame Executable Debug Only Filter [Click here to filter the log](#)

[Logs](#) Tests Checkpoints Query Editor View State Progress Problems

User	Application	Operation	Time	Status	Read	Size
Shubham Karnavat	Unknown	/services/data/v57.0/tooling/executeAn...	5/10/2023, 2:30:11 PM	Success		7.39 KB

➤ Final Output:

The screenshot shows the Salesforce Developer Console interface in Google Chrome. The URL is `gcoer21-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage`. The top navigation bar includes File, Edit, Debug, Test, Workspace, Help, and tabs for TECO.apxic and Log executeAnonymous @5/10/2023, 2:30:11 PM.

Execution Log

Timestamp	Event	Details
14:30:11:013	USER_DEBUG	[9]]DEBUG Addition is:20
14:30:11:013	USER_DEBUG	[18]]DEBUG Subtraction is:0
14:30:11:014	USER_DEBUG	[27]]DEBUG Multiplication is:100
14:30:11:014	USER_DEBUG	[36]]DEBUG Division is:1

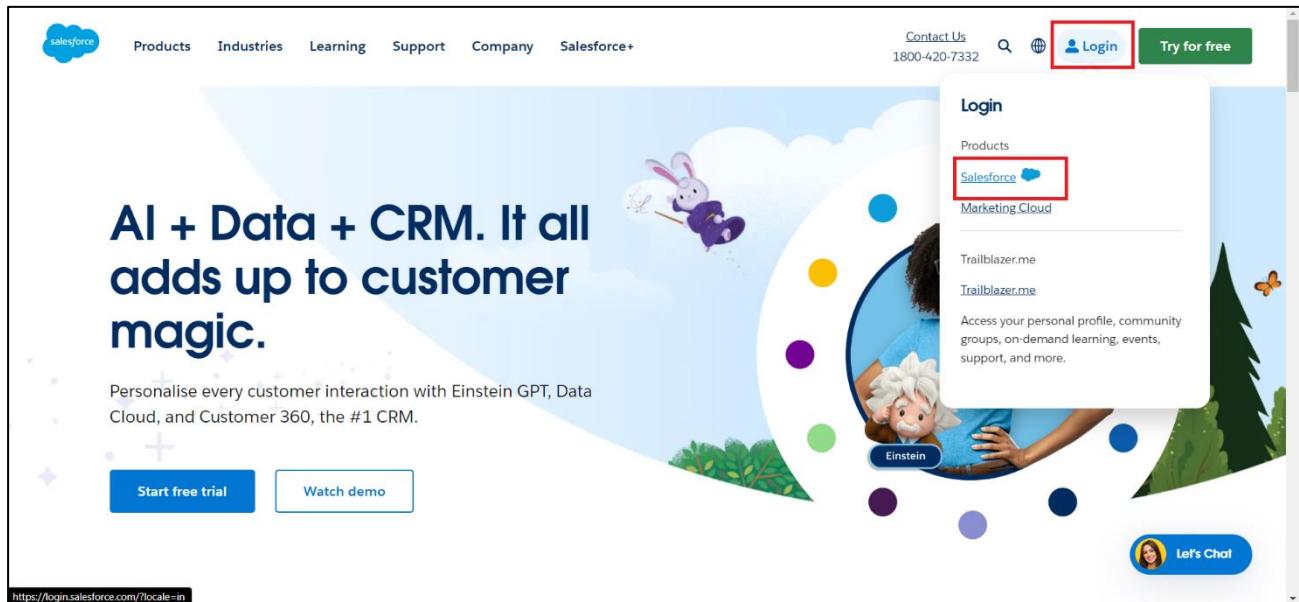
Logs

User	Application	Operation	Time	Status	Read	Size
Shubham Karnavat	Unknown	/services/data/v57.0/tooling/executeAn...	5/10/2023, 2:30:11 PM	Success		7.39 KB

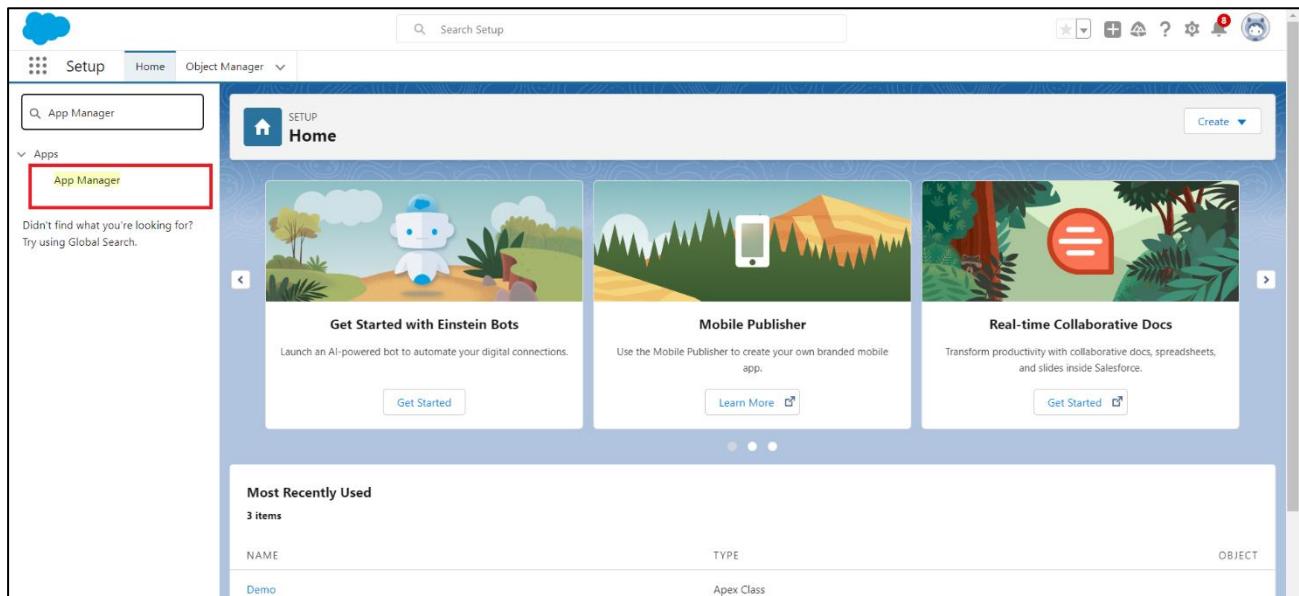
Practical Number :10

Title: Design and develop a custom application using the SalesForce cloud.

- Go to SalesForces.com (<https://www.salesforce.com/in/>) & Login using the credentials:



- Search “App Manager” in the search box and select it from the list:



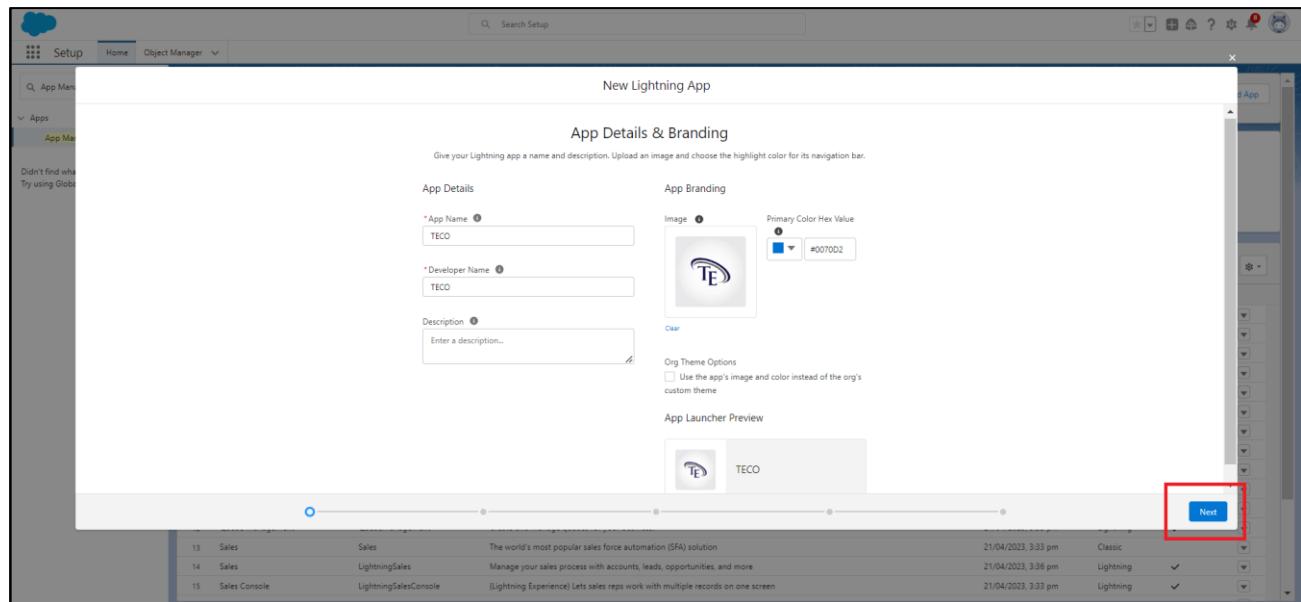
- Click on “New Lightning App”:

The screenshot shows the Salesforce Setup interface with the 'App Manager' selected. The main title is 'Lightning Experience App Manager'. Below it, a section titled 'Clone Apps(Beta)' is displayed. A note says: 'Quickly create new Lightning apps by cloning existing apps. To use the beta feature, indicate that you've read all legal requirements and agree to participate by toggling Enable App Cloning. See additional details and terms in the Winter '23 release notes.' A toggle switch labeled 'Enable App Cloning' is set to 'Disabled'. Below this is a table listing 21 items, sorted by App Name. The columns are: App Name, Developer Name, Description, Last Modified Date, App Type, and Visiblity. The table includes rows for All Tabs, Analytics Studio, App Launcher, Bolt Solutions, Community, Content, and Data Manager.

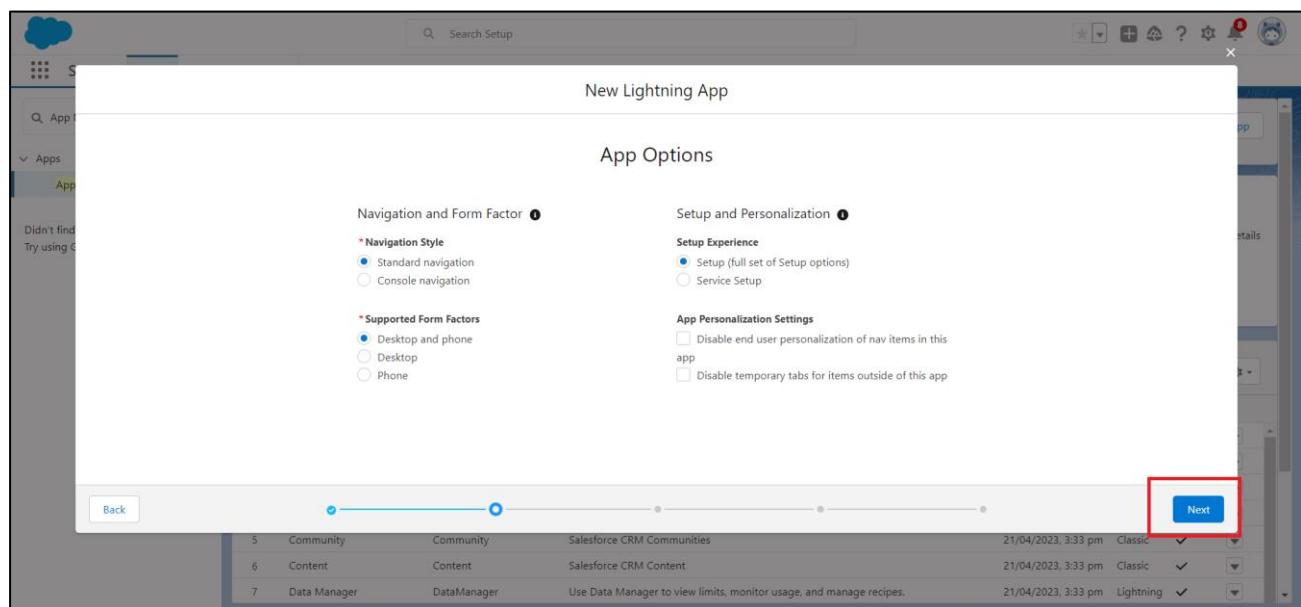
- Enter the App name and Developer name:

This screenshot shows the 'New Lightning App' configuration screen. The title bar says 'New Lightning App'. The main area is titled 'App Details & Branding'. It contains two sections: 'App Details' and 'App Branding'. In 'App Details', there are fields for 'App Name' (set to 'TECO') and 'Developer Name' (also set to 'TECO'). There is also a 'Description' field with the placeholder 'Enter a description...'. In 'App Branding', there is a 'Primary Color Hex Value' field set to '#007002'. Below these sections are 'Org Theme Options' and an 'App Launcher Preview' section. At the bottom, there is a 'Next' button. The background shows the 'App Manager' interface with a list of existing apps like Sales, Sales Console, and Salesforce Chatter.

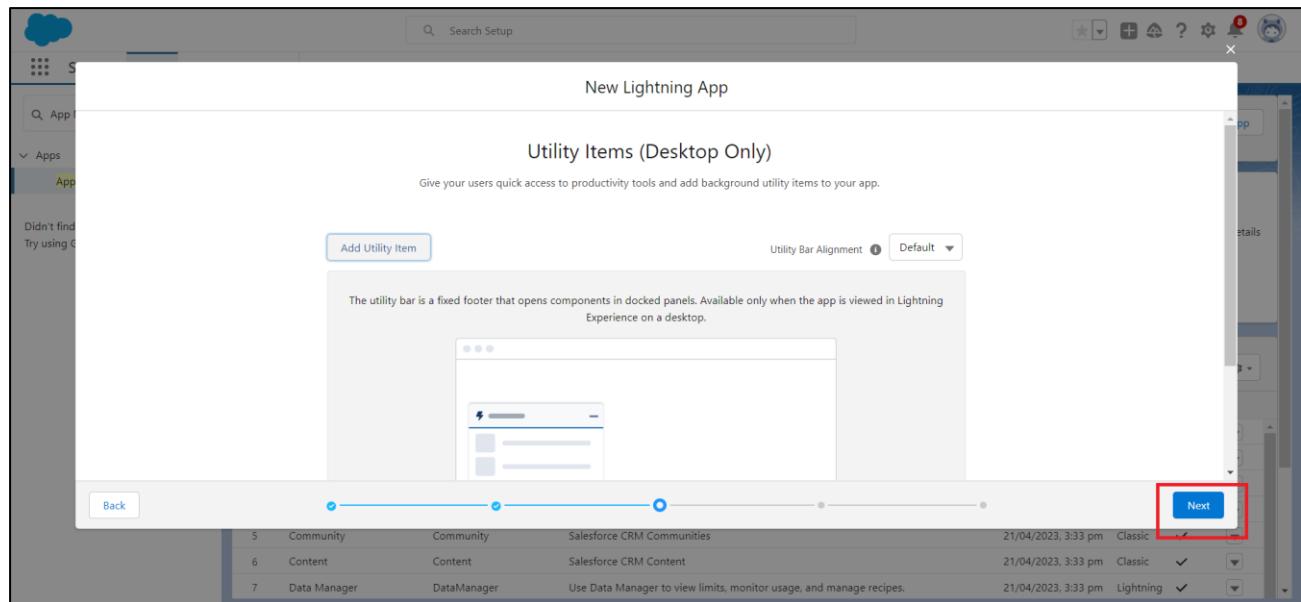
➤ Click on “Next”:



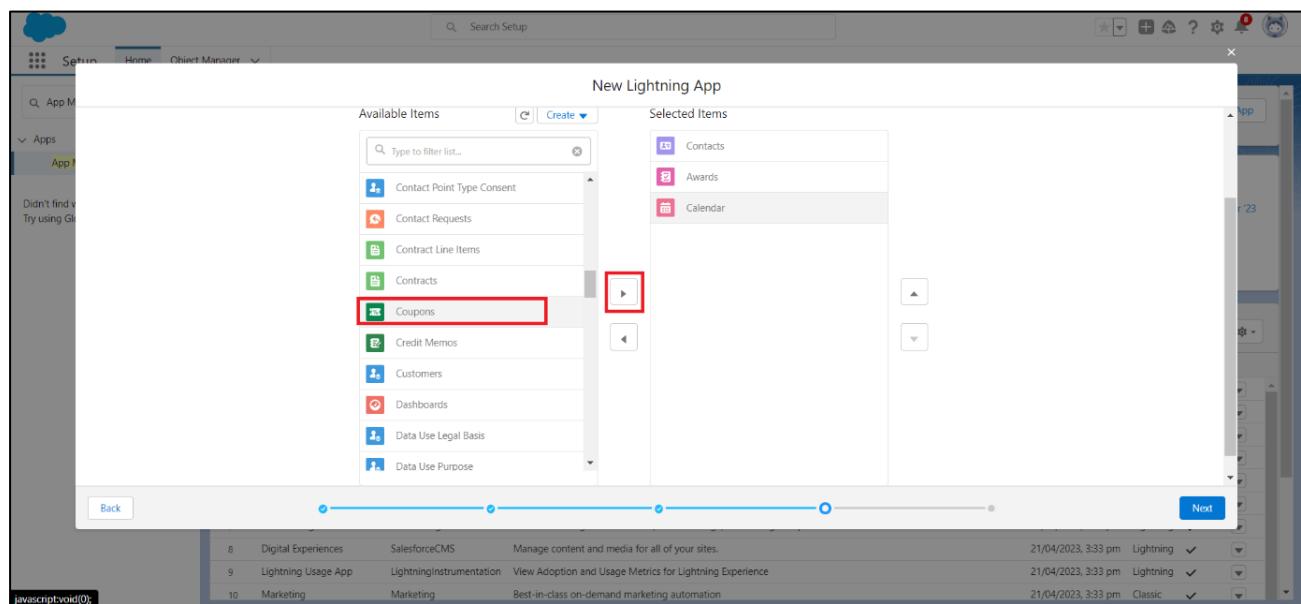
➤ Click on “Next”:



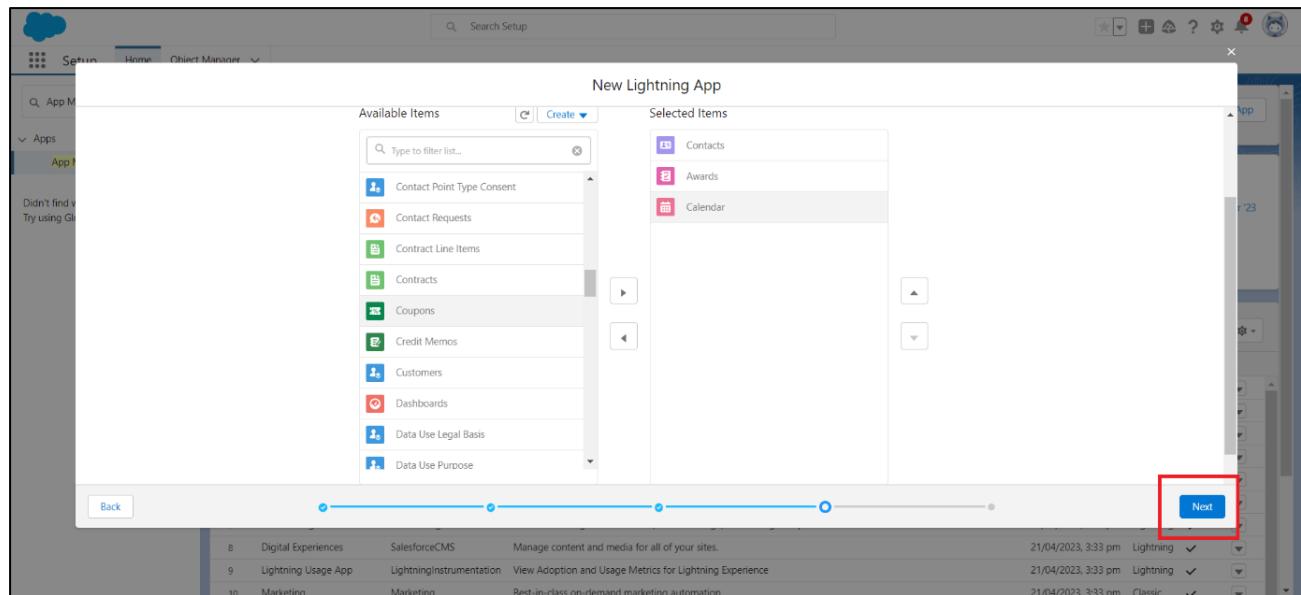
➤ Click on “Next”:



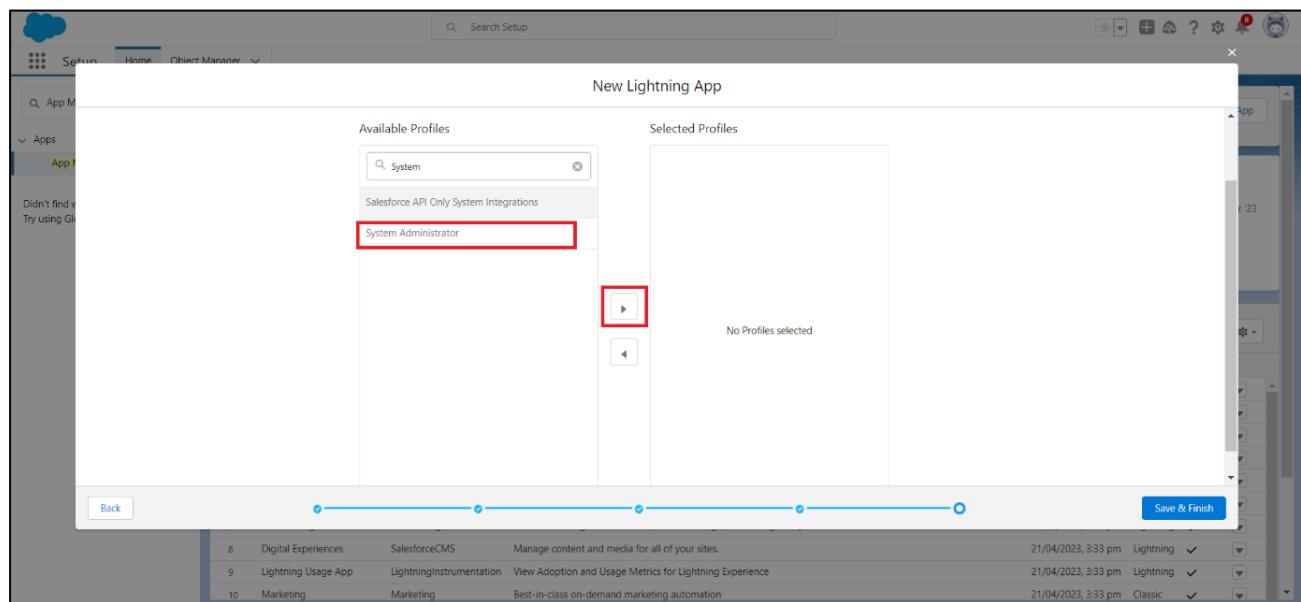
➤ Select items from the given item list:



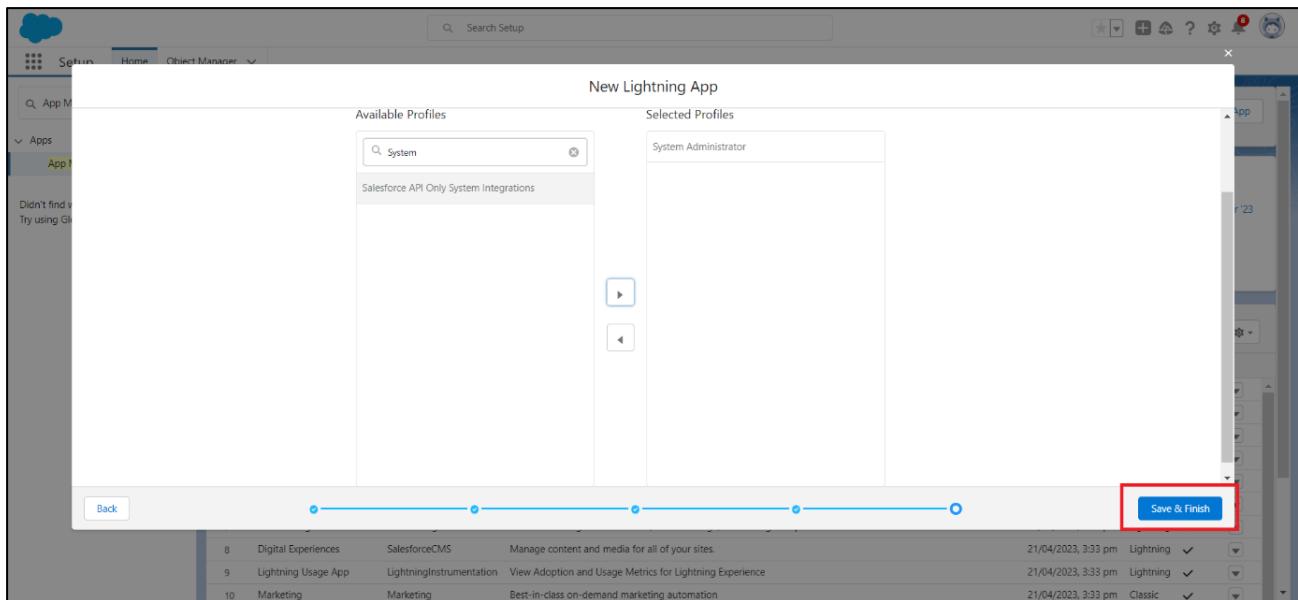
- Click on “Next”:



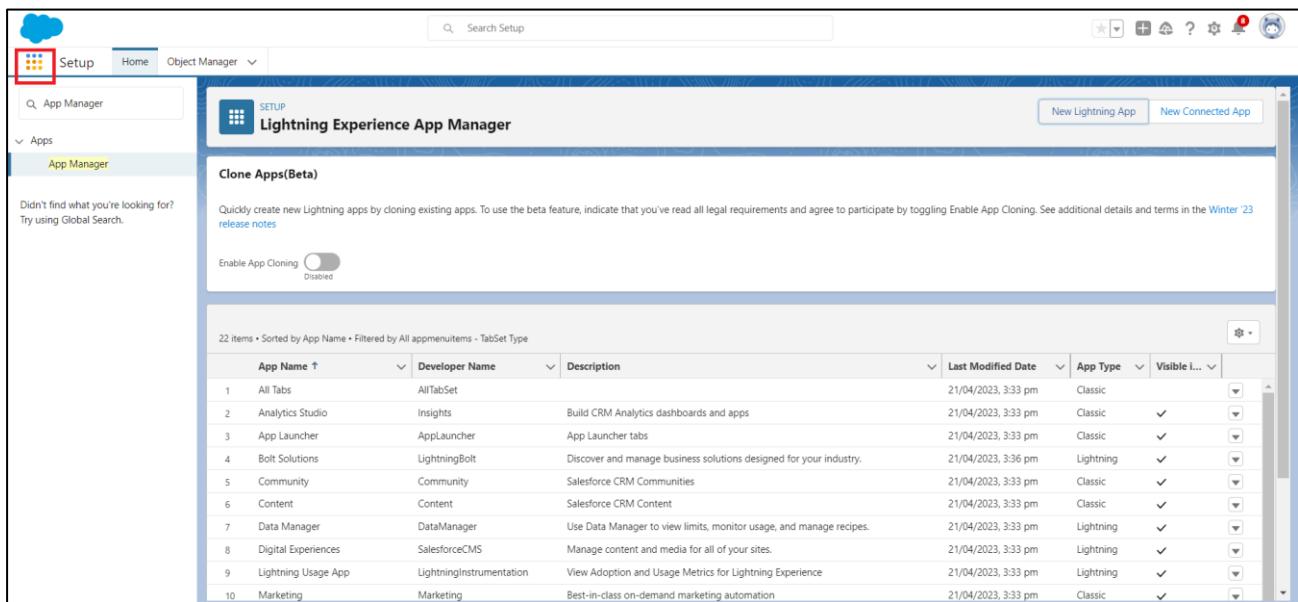
- Search “System Administrator” and select it from the list:



- Click on “Save & Finish”:



- Click on “App launcher Icon” at the top left side of the window:



- Click on “View All”:

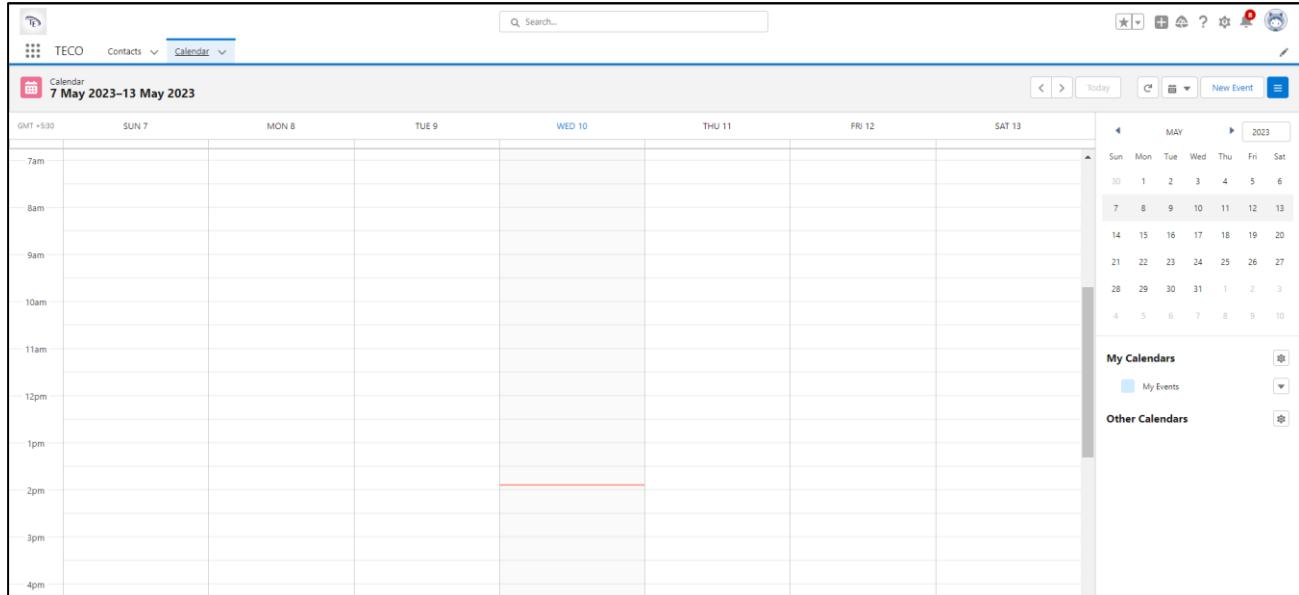
The screenshot shows the Salesforce Setup interface with the "Lightning Experience App Manager" page selected. On the left, there's a sidebar with "Apps" and various app icons like Service, Marketing, Community, etc. Below the sidebar, a button labeled "View All" is highlighted with a red box. The main area displays a table of apps with columns for App Name, Developer Name, Description, Last Modified Date, App Type, and Visible. The table lists 10 apps, including "All Tabs", "Analytics Studio", "App Launcher", "Bolt Solutions", "Community", "Content", "Data Manager", "Digital Experiences", "Lightning Usage App", and "Marketing".

- Select the app from the list of apps :

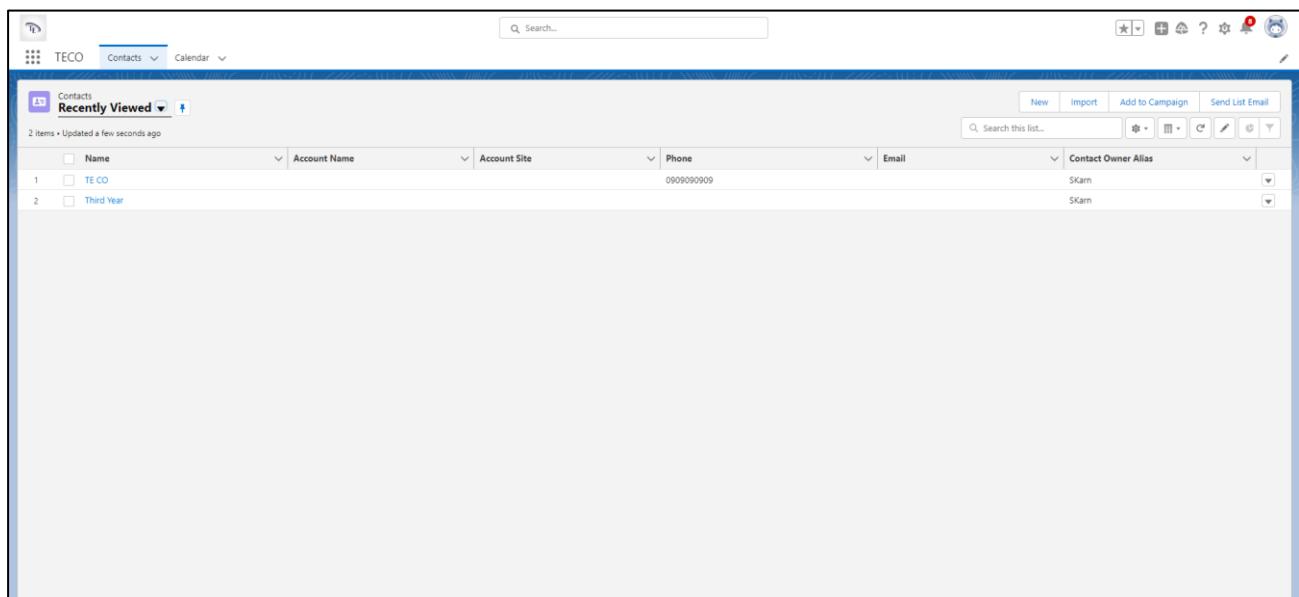
The screenshot shows the Salesforce Setup interface with the "App Launcher" page selected. The left sidebar shows "All Apps" under "Apps". In the main area, there's a grid of app icons. One specific app, "TECO", is highlighted with a red box. The grid also includes other apps like Service, Marketing, Community, Salesforce Chatter, Content, Sales Console, Digital Experiences, Lightning Usage App, Sales, and Bolt Solutions.

Explore the features of the app:

➤ **Calendar:**



➤ **Contacts:**



Practical No : 11

Practical Title: Setup your own cloud for Software as a Service (SaaS) over the existing LAN in your laboratory. In this assignment you have to write your own code for cloud controller using open-source technologies to implement with HDFS. Implement the basic operations may be like to divide the file in segments/blocks and upload/ download file on/from cloud in encrypted form.

Objectives:

- To set your own cloud for SaaS over existing LAN
- To implement the basic operations may be like to divide the file in segments/blocks

Hardware Requirements :

- Pentium IV with latest configuration

Software Requirements :

- Ubuntu 20.04, VMwareESXi cloud

Theory:

Here we are installing VMwareESXi cloud

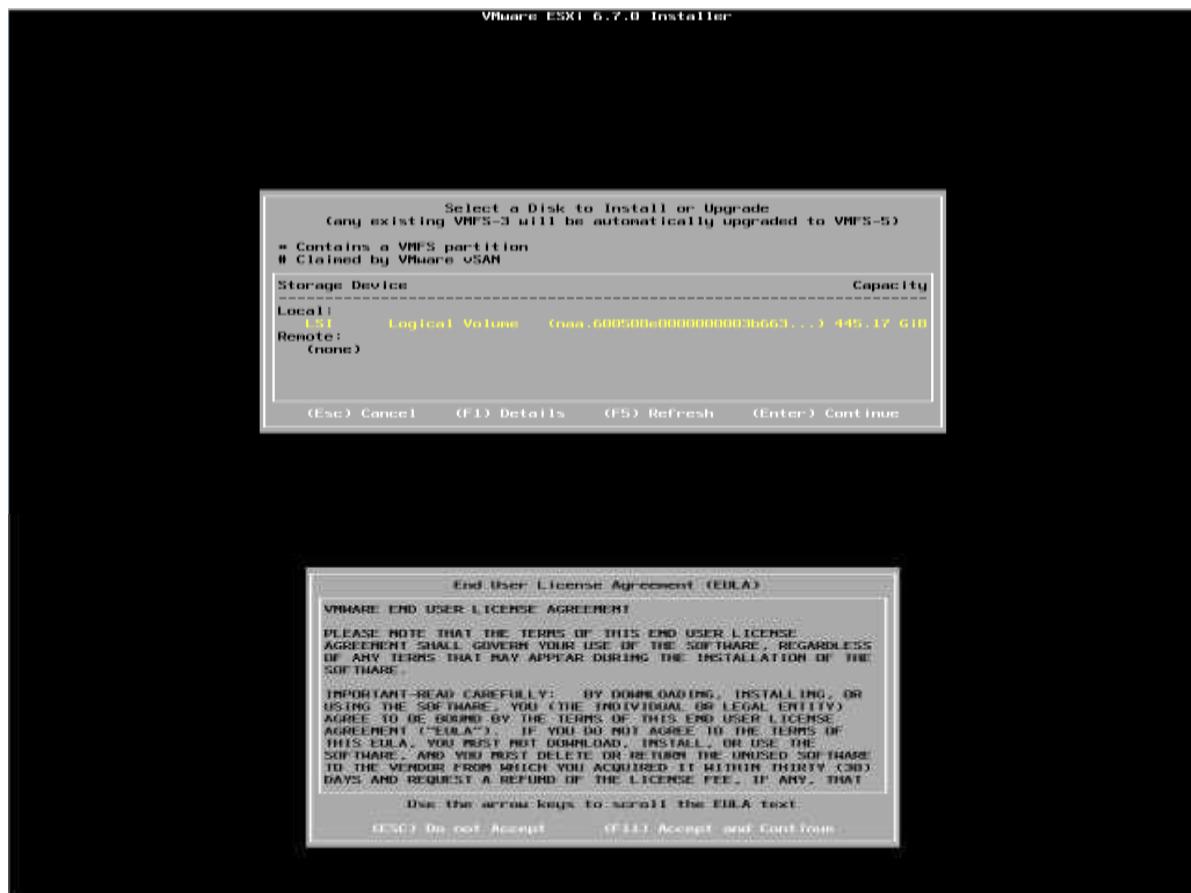
- Host/NodeESXi installation:-
- **ESXiHardwareRequirements:-**
 - ESXi6.7 requires a host machine with at least two CPU cores.
 - ESXi6.7 supports 64-bit x86 processors
 - ESXi6.7 requires the NX/XD bit to be enabled for the CPU in the BIOS.
 - ESXi6.7 requires a minimum of 4 GB of physical RAM. It is recommended to provide at least 8 GB of RAM to run virtual machines in typical production environments.
 - To support 64-bit virtual machines, support for hardware virtualization (Intel VT-x or AMD RVI) must be enabled on x64 CPUs.
 - One or more Gigabit or faster Ethernet controllers. For a list of supported network adapter models.
 - SCSI disk or a local, non-network, RAID LUN with unpartitioned space for the virtual machines.

ForSerialATA(SATA), a disk connected through supported SAS controller or supported on board SATA controllers. SATA disks are considered remote not local. These disks are not used as a scratch partition by default because they are seen as remote.

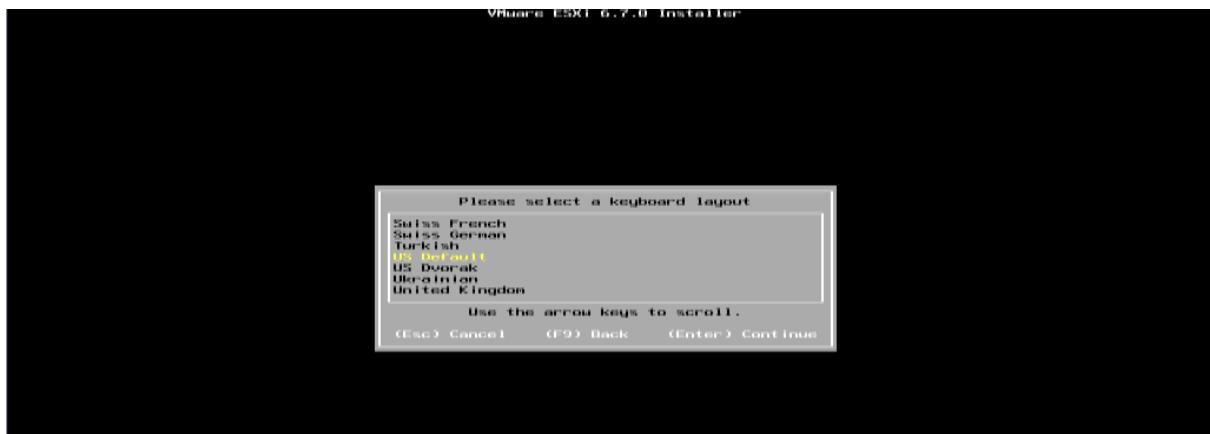


ESXiInstaller:

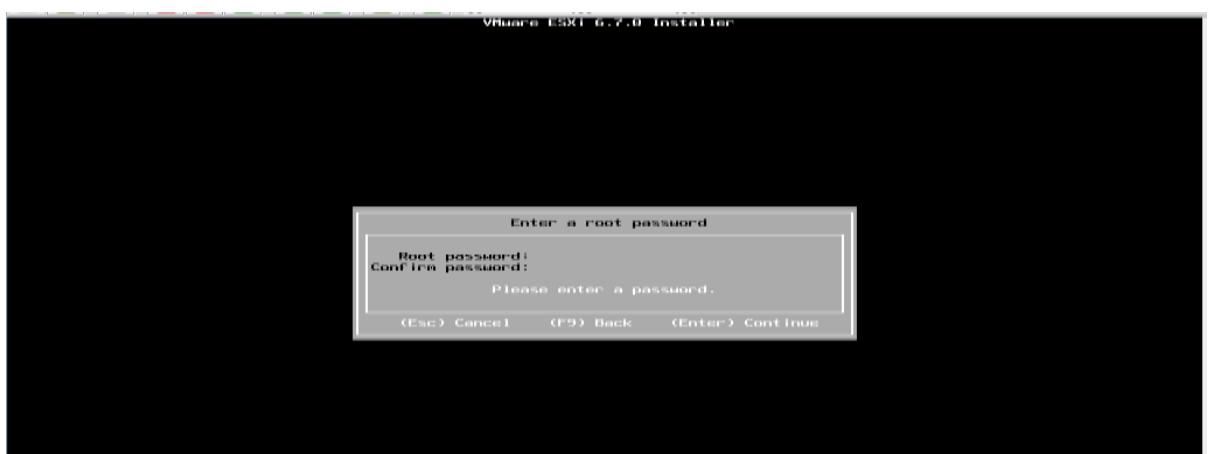
Accept Agreement:



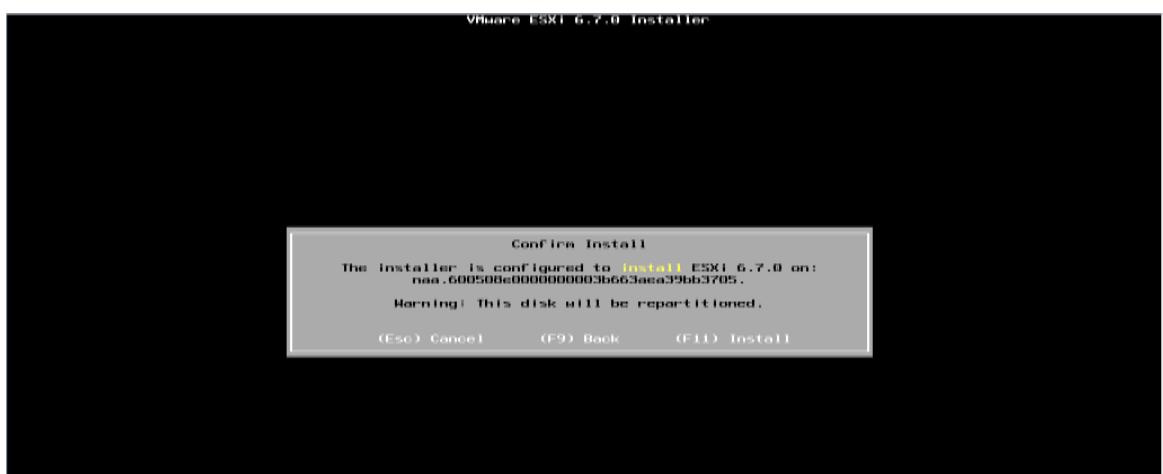
Select storage :



Select Keyboard Layout :



Set NodeESXi Root Password :



Installation complete (Reboot)CLII interface to configuration



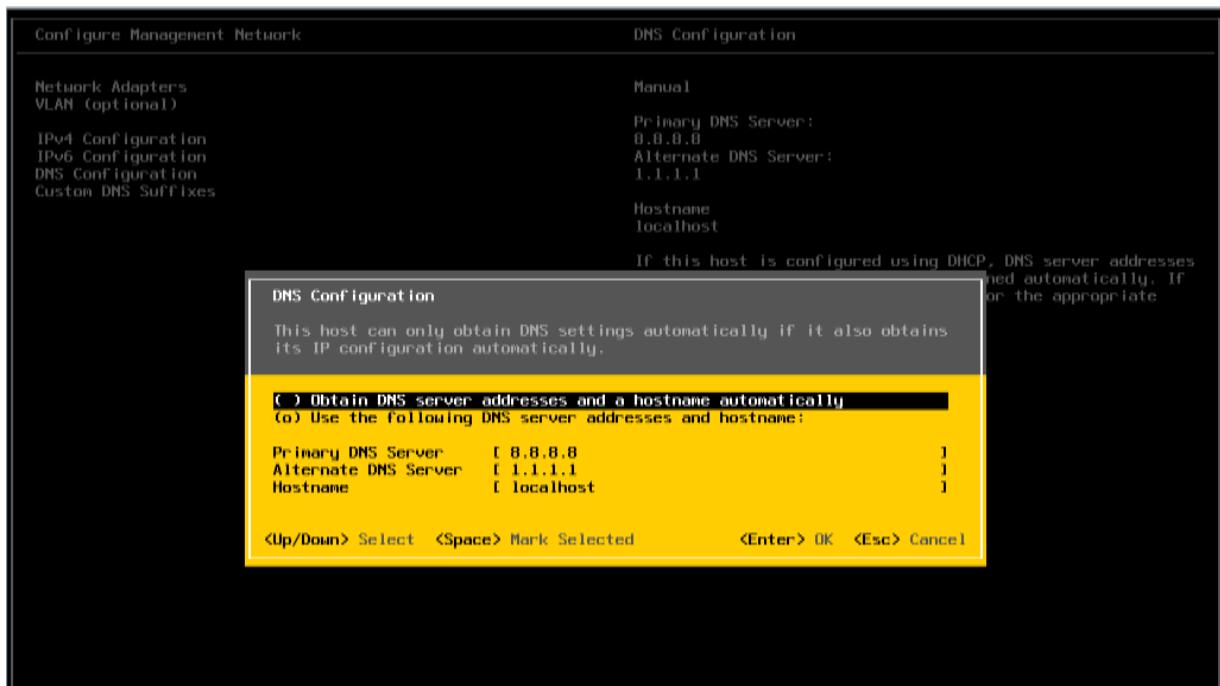
CLI Interface to Configuration:



Configure Management Network

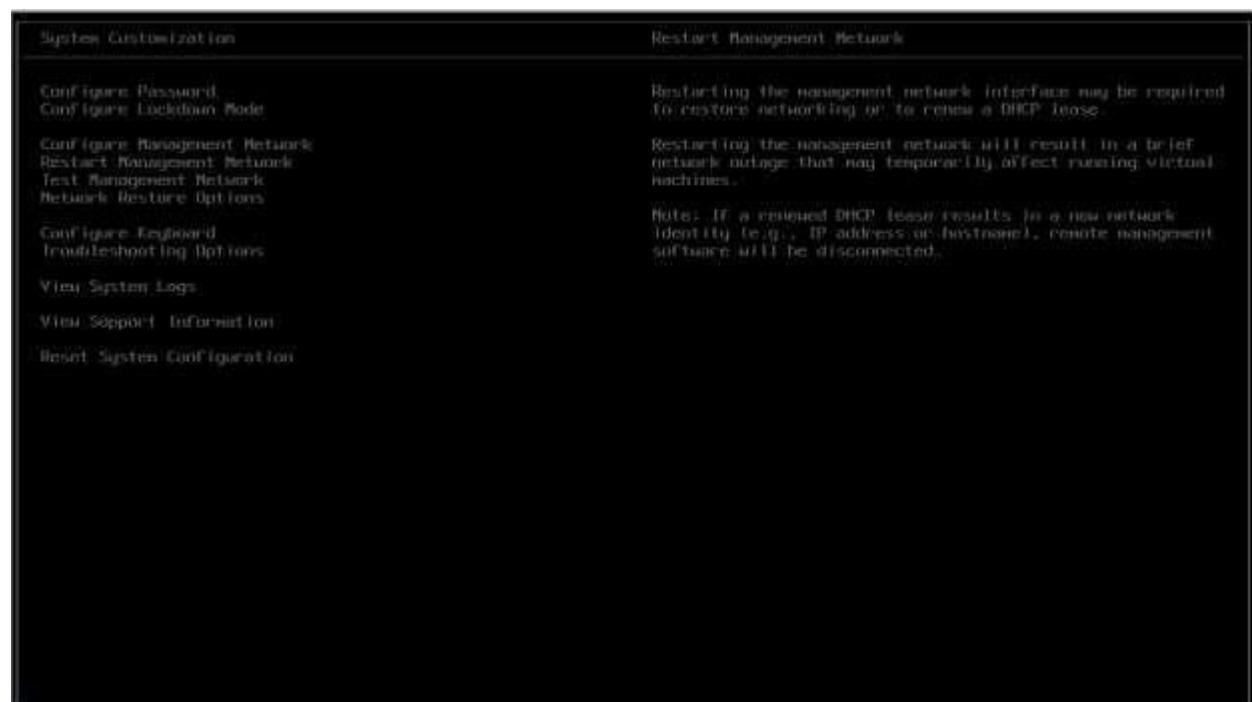


Set IPV4



Set DNServer :

Restart Management Network



GUIAccess :



ClusterSetup

- **CreatingDatacenter**
- **CreatingCluster**
- **Adding Hosts incluster**
- **Resourcesafteraddingcluster.**
- **DRS**
- **Failover**

VCenter Access:

A screenshot of the vSphere Client interface. The title bar says "vSphere Client" and the URL is "https://172.14.5.79/ui/#/extensionId=vSphere.core.inventory". The main pane shows a summary of resources: Virtual Machines: 0, Hosts: 0. It includes sections for CPU, Memory, and Storage usage statistics. Below this is a "Recent Tasks" table:

Task Name	Target	Status	Initiator	Queued For	Start Time	Completion T...	Served
Remove datacenter	EL AVCOE	✓ Completed	VSPHERE.LOCAL	undefined	02/21/2019, 2:50:42 PM	02/21/2019, 2:50:42 PM	vSphere.avcoe.C...
Create datacenter	vSphere.av...	✓ Completed	VSPHERE.LOCAL	undefined	02/21/2019, 2:49:46 PM	02/21/2019, 2:49:46 PM	vSphere.avcoe.C...

Create Datacenter:

The screenshot shows the vSphere Client interface. In the top navigation bar, it says "vSphere Client" and "vsphere.avcoe.com". The main pane displays a "New Datacenter" dialog box. The "Name" field contains "AVOCE" and the "LOCATION" dropdown is set to "vsphere.avcoe.com". Below the dialog, the inventory tree shows "vsphere.avcoe.com" with a "Datacenters" folder expanded, containing the newly created "AVOCE" datacenter.

The screenshot shows the vSphere Client interface. In the top navigation bar, it says "vSphere Client" and "vsphere.avcoe.com". The main pane displays a "New Cluster" dialog box. The "Name" field contains "AVOCE-TESTCLUSTER" and the "LOCATION" dropdown is set to "vsphere.avcoe.com". Below the dialog, the inventory tree shows "vsphere.avcoe.com" with a "Clusters" folder expanded, containing the newly created "AVOCE-TESTCLUSTER" cluster.

Create cluster : Assign

cluster name :

The screenshot shows the vSphere Client interface. In the top navigation bar, it says "vSphere Client" and "vsphere.avcoe.com". The main pane displays a "New Cluster" dialog box for "AVOCE-TESTCLUSTER". The "Name" field is "AVOCE-TESTCLUSTER" and the "LOCATION" dropdown is "vsphere.avcoe.com". Under the "Hosts" section, two hosts are selected: "AVOCE-01" and "AVOCE-02". Both hosts are listed in the "Selected" column with status "Assigned". The "LOCATION" dropdown is set to "vsphere.avcoe.com".

Add host ::

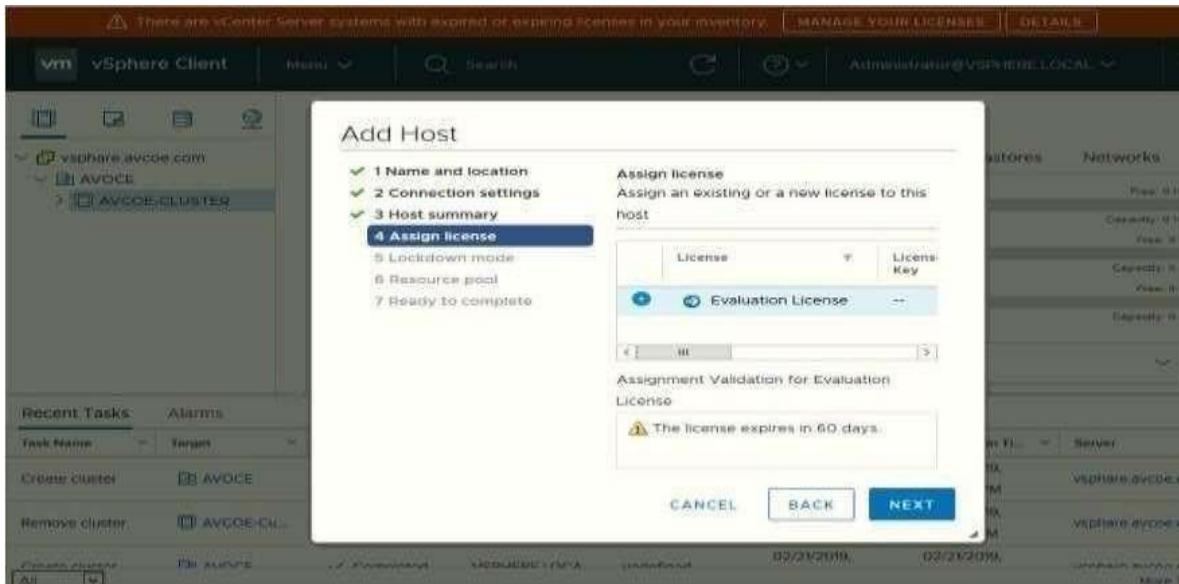
The screenshot shows the vSphere Client interface. In the left sidebar, under the 'AVCOE' folder, there is a 'vSphere.avcoe' entry. The main pane displays the 'AVCOE-CLUSTER' summary. On the left, a context menu is open over the 'vSphere.avcoe' entry, with 'Add Host...' highlighted. The top bar has a warning message: 'There are vCenter Server systems with expired or expiring licenses in your inventory.' Buttons for 'MANAGE YOUR LICENSES' and 'DETAILS' are also present.

Add host IP :

This screenshot shows the 'Add Host' wizard, step 1: Name and location. The title bar says 'Add Host'. The left pane lists steps: 1. Name and location, 2. Connection settings, 3. Host summary, 4. Assign license, 5. Lockdown mode, 6. Resource pool, 7. Ready to complete. The right pane shows a form with 'Host name or IP address' set to '172.16.5.244' and 'Location' set to 'AVCOE-CLUSTER'. A progress bar at the bottom indicates '0% / 100%'.

Enter host credential :

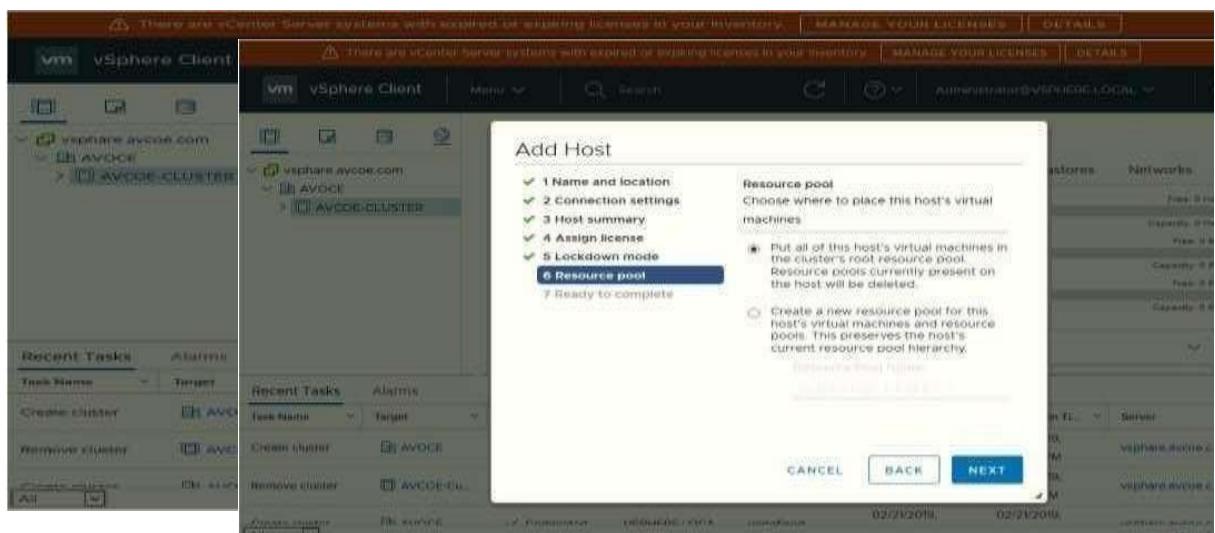
This screenshot shows the 'Add Host' wizard, step 2: Connection settings. The title bar says 'Add Host'. The left pane lists steps: 1. Name and location, 2. Connection settings, 3. Host summary, 4. Assign license, 5. Lockdown mode, 6. Resource pool, 7. Ready to complete. The right pane shows a 'Host summary' section with details: Name '172.16.5.244', Vendor 'Hewlett-Packard', Model 'HP Z420 Workstation', Version 'VMware ESXi 6.7.0 build-10302608'. Below it is a 'Virtual Machines' section. A progress bar at the bottom indicates '100% / 100%'.



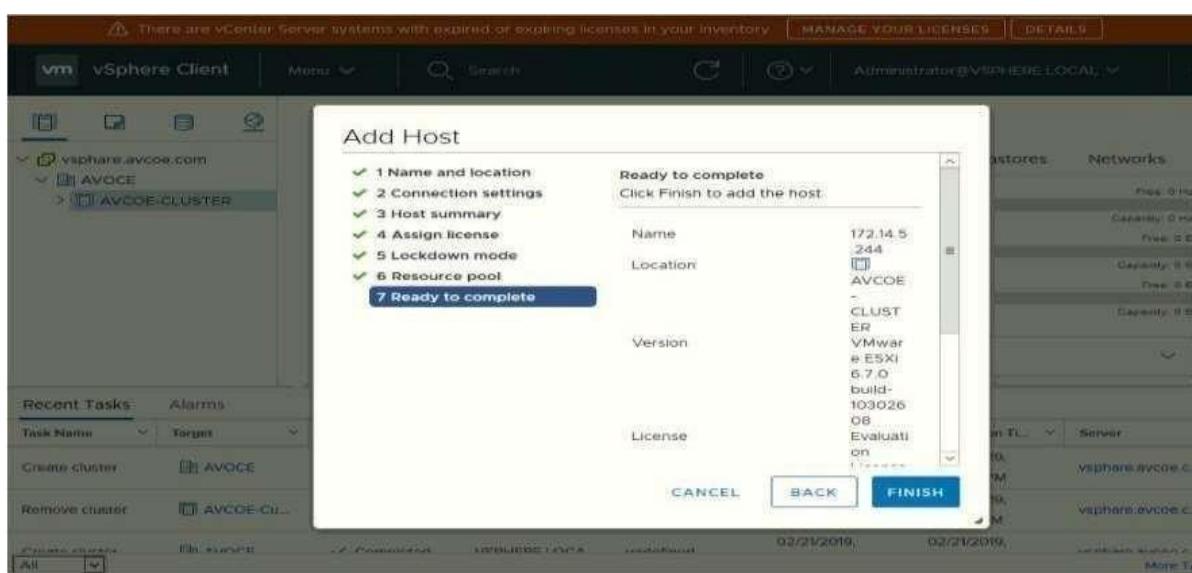
Hot summary :

Lock Down mode:

Add Host In Pool:



Finish:



Host View and View Config:

Cluster View and Configuration:

The screenshot shows the vSphere Client interface for host 172.14.5.245. The left sidebar shows the navigation tree with 'vSphere Client' selected. The main pane displays the host's summary information, including its name (172.14.5.245), hypervisor type (VMware ESXi 6.7.0), processor (Intel(R) Xeon(R) CPU E5-1607 v2 @ 3.00GHz), and memory usage (Used: 0.8 GB, Free: 11.87 GHz). Below this, there are sections for logical processors, NICs, virtual machines, state, and uptime. At the bottom, a table lists recent tasks such as 'Configuring vSphere HA' and 'Add host'.

The screenshot shows the vSphere Client interface for the cluster AVCOE-CLUSTER. The left sidebar shows the navigation tree with 'vSphere Client' selected. The main pane displays the cluster's summary information, including total processors (8), total vMotion migrations (0), and resource statistics for CPU, memory, and storage. Below this, there are tabs for 'Related Objects' and 'vSphere DRS'. At the bottom, a table lists recent tasks such as 'Configuring vSphere HA' and 'Add host'.

Conclusion: Like this we have configure VSphere Private Cloud