

Practical No. 1

Name: Sakshi Mahendra Jagtap

Div: A

Roll No:49

Title:- Implement DFS and BFS Algorithm. Use and Undirected Graph and develop a Recursive Algorithm for searching all the vertices of the graph or tree data structure.

Program :-

Breadth First Search(BFS):-

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

```
visited = [] # List to keep track of visited nodes.
```

```
queue = [] #Initialize a 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)
```

```
# Driver Code
```

```
bfs(visited, graph, 'A')
```

The screenshot shows a Jupyter Notebook interface. At the top, there's a toolbar with File, Edit, View, Insert, Cell, Kernel, Widgets, Help, Trusted, Python 3 (pykernel), and Logout. Below the toolbar is a code cell labeled In [16]. The code in the cell is:

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

visited = [] # List to keep track of visited nodes.
queue = [] #Initialize a 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)

# Driver Code
bfs(visited, graph, 'A')
```

Below the code cell, the output A B C D E F is displayed. At the bottom left, there's another In [] cell.

2. Depth-first Search:

```
# Using a Python dictionary to act as an adjacency list
graph = {
    'A' : ['B','C'],
    'B' : ['D', 'E'],
    'C' : ['F'],
    'D' : [],
    'E' : ['F'],
    'F' : []
}
visited = set() # Set to keep track of visited nodes of graph.

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

Driver Code

```
print("Following is the Path using Depth-First Search")
dfs(visited, graph, 'A')
```

The screenshot shows a Jupyter Notebook interface with the following details:

- Header:** jupyter Untitled Last Checkpoint: 10 minutes ago (unsaved changes), Trusted, Python 3 (ipykernel).
- Toolbar:** File, Edit, View, Insert, Cell, Kernel, Widgets, Help.
- Cell Content:** In [19]:
graph = {
 'A' : ['B','C'],
 'B' : ['D','E'],
 'C' : ['F'],
 'D' : [],
 'E' : ['F'],
 'F' : []
}
visited = set() # Set to keep track of visited nodes of graph.
def dfs(visited, graph, node): #function for dfs
 if node not in visited:
 print (node)
 visited.add(node)
 for neighbour in graph[node]:
 dfs(visited, graph, neighbour)
Driver Code
print("Following is the Path using Depth-First Search")
dfs(visited, graph, 'A')

Following is the Path using Depth-First Search
A
B
D
E
F
C
- Input Cell:** In []: (empty)

Practical No. 2

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

Program :-
A* Algorithm

```
from collections import deque
```

```
class Graph:
```

```
    # example of adjacency list (or rather map)
    # adjacency_list = {
    # 'A': [('B', 1), ('C', 3), ('D', 7)],
    # 'B': [('D', 5)],
    # 'C': [('D', 12)]
    # }
```

```
def __init__(self, adjacency_list):
    self.adjacency_list = adjacency_list
```

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

```
# heuristic function with equal values for all nodes
```

```
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 is a list of nodes which have been visited, but who's
neighbors
    # haven't all been inspected, starts off with the start node
    # closed_list is a list of nodes which have been visited
    # and who's neighbors have been inspected
    open_list = set([start_node])
    closed_list = set([])

    # g contains current distances from start_node to all other nodes
    # the default value (if it's not found in the map) is +infinity
    g = { }

    g[start_node] = 0

    # parents contains an adjacency map of all nodes
    parents = { }
    parents[start_node] = start_node

while len(open_list) > 0:
    n = None

    # find a node with the lowest value of f() - evaluation function
    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 the current node is the stop_node
    # then we begin reconstructin the path from it to the start_node
    if n == stop_node:
        reconst_path = []

        while parents[n] != n:
            reconst_path.append(n)
            n = parents[n]

        reconst_path.append(start_node)
        return reconst_path

    # the node v is now closed
    closed_list.add(v)

    # update open_list and parents
    for neighbor in self.edges[v]:
        if neighbor not in closed_list:
            if neighbor not in open_list:
                open_list.append(neighbor)
                parents[neighbor] = v
            else:
                if g[v] + self.h(v) < g[neighbor] + self.h(neighbor):
                    parents[neighbor] = v
                    g[neighbor] = g[v] + self.h(v)

```

```

        reconst_path.append(start_node)

        reconst_path.reverse()

        print('Path found: {}'.format(reconst_path))
        return reconst_path

    # for all neighbors of the current node do
    for (m, weight) in self.get_neighbors(n):
        # if the current node isn't in both open_list and closed_list
        # add it to open_list and note n as its parent
        if m not in open_list and m not in closed_list:
            open_list.add(m)
            parents[m] = n
            g[m] = g[n] + weight

        # otherwise, check if its quicker to first visit n, then m
        # and if it is, update parent data and g data
        # and if the node was in the closed_list, move it to open_list
        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)

    # remove n from the open_list, and add it to closed_list
    # because all of his neighbors were inspected
    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')
```

The screenshot shows a Jupyter Notebook interface with the following details:

- Header:** localhost:8888/notebooks/Untitled8.ipynb?kernel_name=python3
- Title Bar:** Jupyter Untitled8 Last Checkpoint: 5 minutes ago (autosaved)
- Toolbar:** File, Edit, View, Insert, Cell, Kernel, Widgets, Help, Run, Stop, Kernel, Code dropdown, and a small icon.
- Code Cell:** Contains the Python code for the A* algorithm and the adjacency list definition.
- Output Cell:** Shows the path found and the output of the command.
- Status Bar:** Trusted, Python 3, and a user profile icon.

```
for (m, weight) in self.get_neighbors(n):
    # if the current node isn't in both open_list and closed_list
    # add it to open_list and note n as its parent
    if m not in open_list and m not in closed_list:
        open_list.add(m)
        parents[m] = n
        g[m] = g[n] + weight

    # otherwise, check if it's quicker to first visit n, then m
    # and if it is, update parent data and g data
    # and if the node was in the closed_list, move it to open_list
    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)

    # remove n from the open_list, and add it to closed_list
    # because all of his neighbors were inspected
    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')
```

Path found: ['A', 'B', 'D']
Out[11]: ['A', 'B', 'D']

Practical No. 3

Title:- Implement Greedy search algorithm for selection sort

Program :-

Greedy search Algorithm for selection sort program

```
# Selection sort in Python
# time complexity O(n*n)
#sorting by finding min_index
def selectionSort(array, size):

    for ind in range(size):
        min_index = ind

        for j in range(ind + 1, size):
            # select the minimum element in every iteration
            if array[j] < array[min_index]:
                min_index = j
            # swapping the elements to sort the array
        (array[ind], array[min_index]) = (array[min_index], array[ind])

arr = [-2, 45, 0, 11, -9, 88, -97, -202, 747]
size = len(arr)
selectionSort(arr, size)
print('The array after sorting in Ascending Order by selection sort is:')

print(arr)
```

localhost:8888/notebooks/Untitled9.ipynb?kernel_name=python3

Jupyter Untitled9 Last Checkpoint 3 minutes ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

```
In [1]: # Selection sort in Python  
# time complexity O(n*n)  
#sorting by finding min_index  
def selectionSort(array, size):  
    for ind in range(size):  
        min_index = ind  
        for j in range(ind + 1, size):  
            # select the minimum element in every iteration  
            if array[j] < array[min_index]:  
                min_index = j  
        # swapping the elements to sort the array  
        (array[ind], array[min_index]) = (array[min_index], array[ind])  
  
arr = [-2, 45, 0, 11, -9, 88, -97, -202, 747]  
size = len(arr)  
selectionSort(arr, size)  
print('The array after sorting in Ascending Order by selection sort is:')
```

The array after sorting in Ascending Order by selection sort is:
[-202, -97, -9, -2, 0, 11, 45, 88, 747]

In []:

Practical No. 4

Title:- Implement a solution for a constraint satisfaction problem using branch and bound and backtracking for n-queens problem or a graph coloring problem

Program :-

n-queens problem

```
# Python program to solve N Queen  
# Problem using backtracking
```

```
global N
```

```
N = 4
```

```
def printSolution(board):  
    for i in range(N):  
        for j in range(N):  
            print (board[i][j],end=' ')  
    print()
```

```
# A utility function to check if a queen can  
# be placed on board[row][col]. Note that this  
# function is called when "col" queens are  
# already placed in columns from 0 to col -1.
```

```
# So we need to check only left side for  
# attacking queens
```

```
def isSafe(board, row, col):
```

```
    # Check this row on left side  
    for i in range(col):  
        if board[row][i] == 1:  
            return False
```

```
    # Check upper diagonal on left side
```

```

for i, j in zip(range(row, -1, -1), range(col, -1, -1)):
    if board[i][j] == 1:
        return False

# Check lower diagonal on left side
for i, j in zip(range(row, N, 1), range(col, -1, -1)):
    if board[i][j] == 1:
        return False

return True

def solveNQUtil(board, col):
    # base case: If all queens are placed
    # then return true
    if col >= N:
        return True

    # Consider this column and try placing
    # this queen in all rows one by one
    for i in range(N):

        if isSafe(board, i, col):
            # Place this queen in board[i][col]
            board[i][col] = 1

            # recur to place rest of the queens
            if solveNQUtil(board, col + 1) == True:
                return True

            # If placing queen in board[i][col]
            # doesn't lead to a solution, then
            # queen from board[i][col]
            board[i][col] = 0

    # if the queen can not be placed in any row in
    # this column col then return false
    return False

# This function solves the N Queen problem using

```

```

# Backtracking. It mainly uses solveNQUtil() to
# solve the problem. It returns false if queens
# cannot be placed, otherwise return true and
# placement of queens in the form of 1s.
# note that there may be more than one
# solutions, this function prints one of the
# feasible solutions.
def solveNQ():

    board = [ [0, 0, 0, 0],
              [0, 0, 0, 0],
              [0, 0, 0, 0],
              [0, 0, 0, 0]
            ]

    if solveNQUtil(board, 0) == False:
        print ("Solution does not exist")
        return False

    printSolution(board)
    return True

# driver program to test above function
solveNQ()

```

The screenshot shows a Jupyter Notebook interface with the following details:

- Header:** localhost:8888/notebooks/Untitled9.ipynb?kernel_name=python3
- Toolbar:** Includes back, forward, search, and other standard browser-like buttons.
- Header Bar:** Shows "Jupyter Untitled9 Last Checkpoint: 11 minutes ago (unsaved changes)" and a "Logout" button.
- Menu Bar:** File, Edit, View, Insert, Cell, Kernel, Widgets, Help.
- Cell Area:**
 - A code cell containing the provided Python code for the N-Queens problem.
 - An output cell below it showing the resulting 4x4 board configuration:
- Output:**

```

0 0 1 0
1 0 0 0
0 0 0 1
0 1 0 0

```

out[5]: True

Practical No. 5

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

Program :-

Chatbot Program

```
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) % 105
```

```
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('TE-Chatbot', '2022') # change it as you need
remind_name()
guess_age()
count()
```

```
test()  
end()
```

The screenshot shows a Jupyter Notebook interface running on a Windows desktop. The notebook title is "Untitled10 - Jupyter Notebook". The code cell contains Python code for a chatbot, which includes printing a dotted separator, prompting for name and age, and testing programming knowledge. The output cell displays the chatbot's welcome message, the user's name ("Tejas"), age ("33"), and a list of four reasons for using methods.

```
print('.....')  
print('.....')  
input()  
  
greet('TE-chatbot', '2022') # change it as you need  
remind_name()  
guess_age()  
count()  
test()  
end()  
  
Hello! My name is TE-Chatbot.  
I was created in 2022.  
Please, remind me your name.  
Tejas  
What a great name you have, Tejas!  
Let me guess your age.  
Enter remainders of dividing your age by 3, 5 and 7.  
21  
3  
5  
Your age is 33; that's a good time to start programming!  
Now I will prove to you that I can count to any number you want.  
2  
0 !  
1 !  
2 !  
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.
```

Practical No. 6

Title:- Implement of Expert system Help Desk Management System

Program :-

Help Desk Management System

Define a dictionary of common problems and their solutions

```
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"
```

```
}
```

Define a function to handle user requests

```
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."
```

Main loop to prompt user for input

```
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. Printer not working
Check that it's turned on and connected to the network
What's the problem? Type 'exit' to quit. Can't log in
Make sure you're using the correct username and password
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. Internet connection not working
Restart your modem or router
What's the problem? Type 'exit' to quit. Email not sending
I'm sorry, I don't know how to help with that problem.
What's the problem? Type 'exit' to quit. exit
Goodbye!
What's the problem? Type 'exit' to quit. █
```

PRACTICAL NO : 7

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

STEPS TO STUDY AMAZON EC2 :

How to Use AWS EC2

Step 1 – Sign-in to AWS account and open IAM console by using the following link

<https://console.aws.amazon.com/iam/>.

Step 2 – In the navigation Panel, create/view groups and follow the instructions.

Step 3 – Create IAM user. Choose users in the navigation pane. Then create new users and add users to the groups.

Step 4 – Create a Virtual Private Cloud using the following instructions.

Open the Amazon VPC console by using the following link –<https://console.aws.amazon.com/vpc/>

Select VPC from the navigation panel. Then select the same region in which we have created key-pair.

Select start VPC wizard on VPC dashboard.

Select VPC configuration page and make sure that VPC with single subnet is selected. Then choose Select.

VPC with a single public subnet page will open. Enter the VPC name in the name field and leave other configurations as default.

Select create VPC, then select Ok.

Step 5 – Create WebServerSG security groups and add rules using the following instructions.

On the VPC console, select Security groups in the navigation panel.

Select create security group and fill the required details like group name, name tag, etc.

Select your VPC ID from the menu. Then select yes, create button.

Now a group is created. Select the edit option in the inbound rules tab to create rules.

Step 6 – Launch EC2 instance into VPC using the following instructions.

Open EC2 console by using the following link – <https://console.aws.amazon.com/ec2/>

Select launch instance option in the dashboard.

A new page will open. Choose Instance Type and provide the configuration. Then select Next: Configure Instance Details.

A new page will open. Select VPC from the network list. Select subnet from the subnet list and leave the other settings as default.

Click Next until the Tag Instances page appears.

Step 7 – On the Tag Instances page, provide a tag with a name to the instances. Select Next: Configure Security Group.

Step 8 – On the Configure Security Group page, choose the Select an existing security group option. Select the WebServerSG group that we created previously, and then choose Review and Launch.

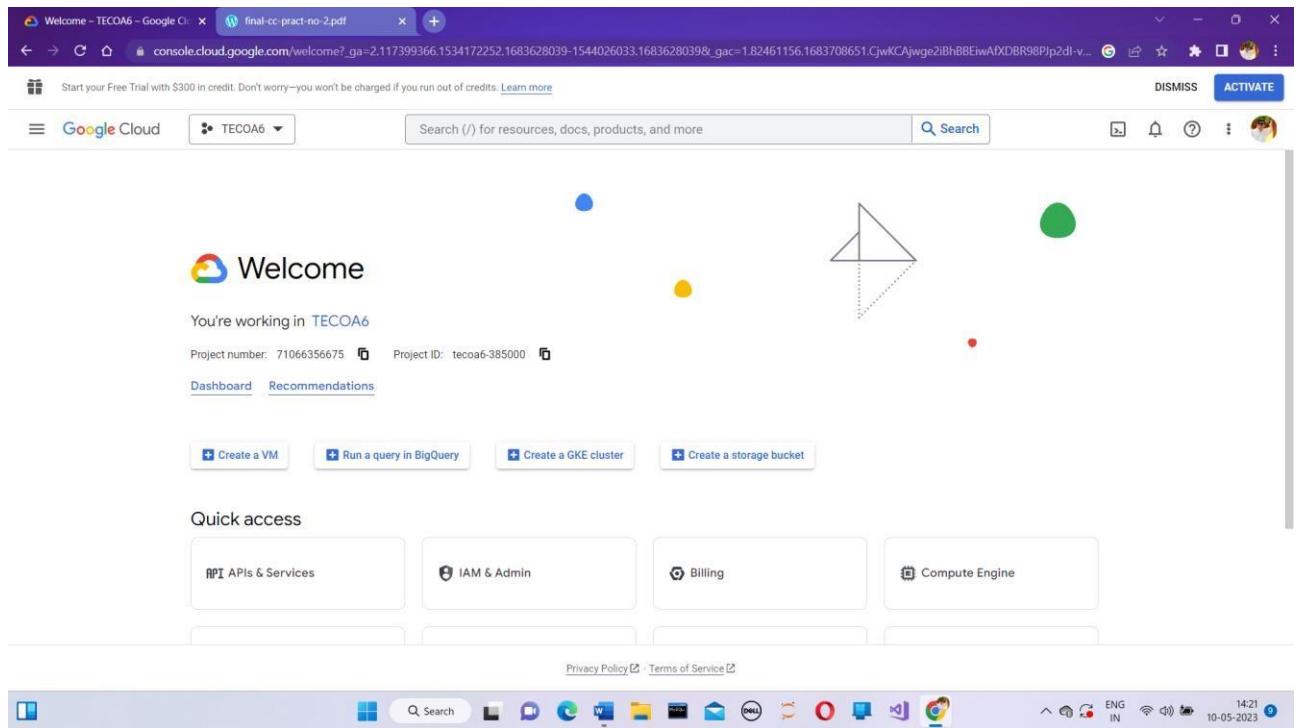
Step 9 – Check Instance details on Review Instance Launch page then click the Launch button.

Step 10 – A pop up dialog box will open. Select an existing key pair or create a new key pair. Then select the acknowledgement check box and click the Launch Instances button.

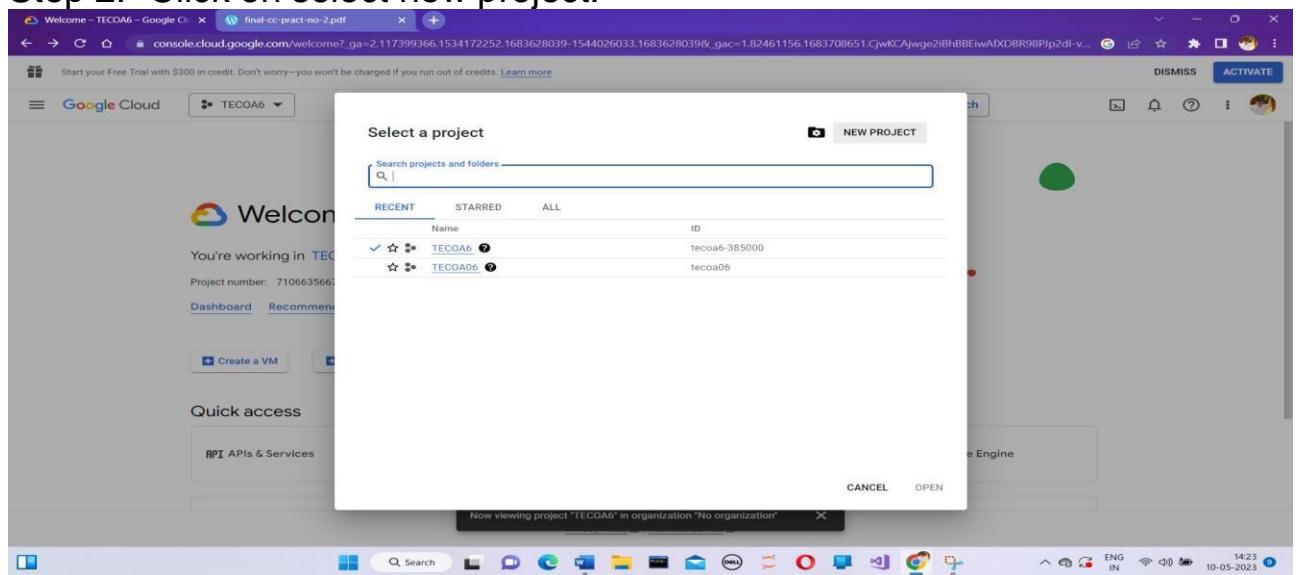
Practical No. 8

Title:- Installation and configure Google app engine

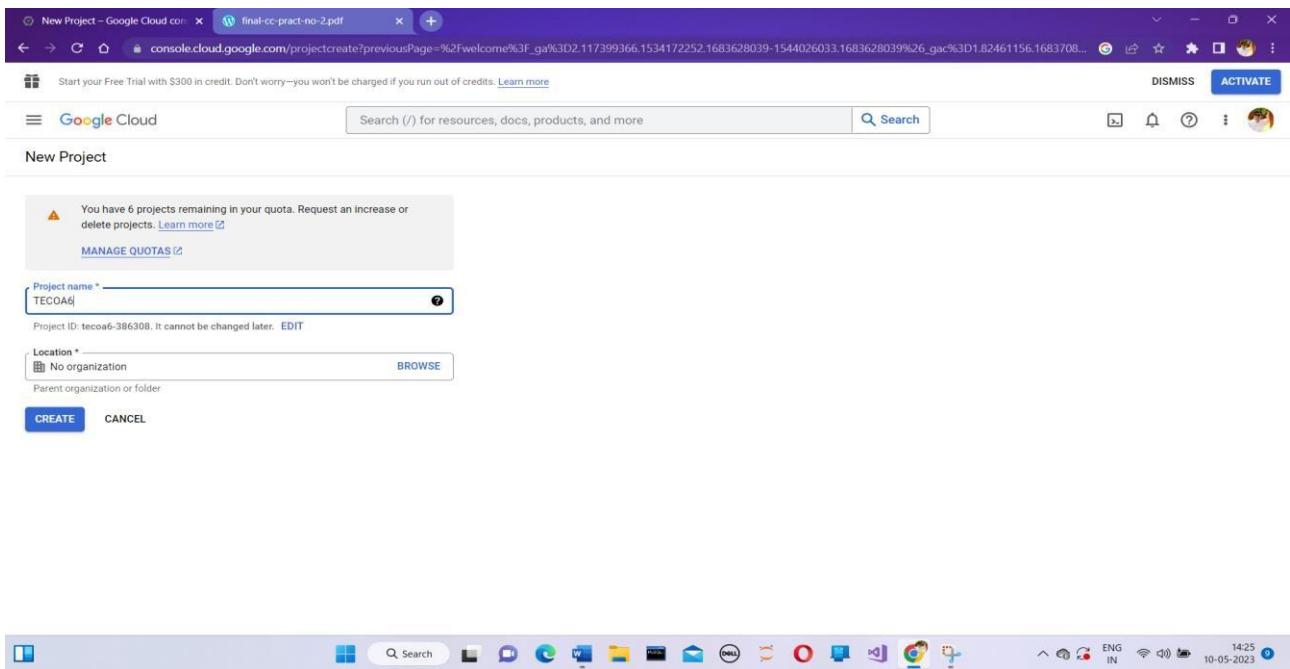
Step 1:- Search Google Cloud Platform in a any search engine& Click on Console.



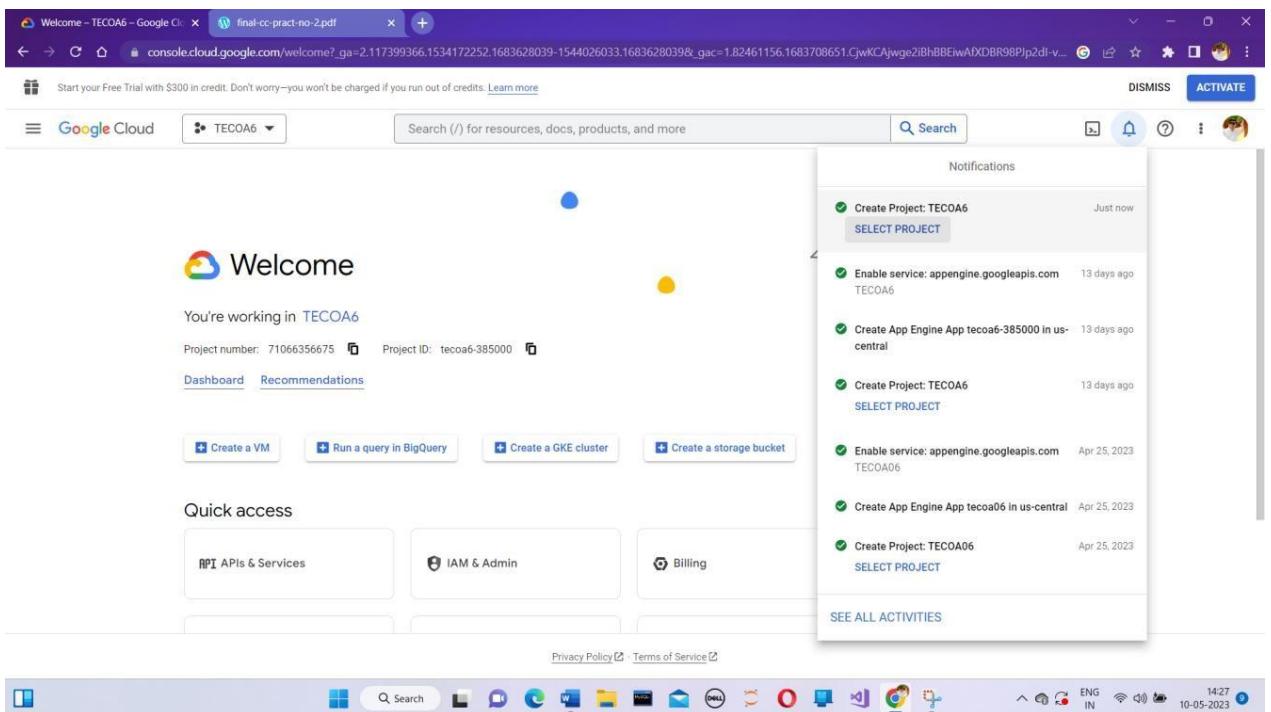
Step 2:- Click on select new project.



Step3:- Give Project name and click on create.



Step4:- Click on select project



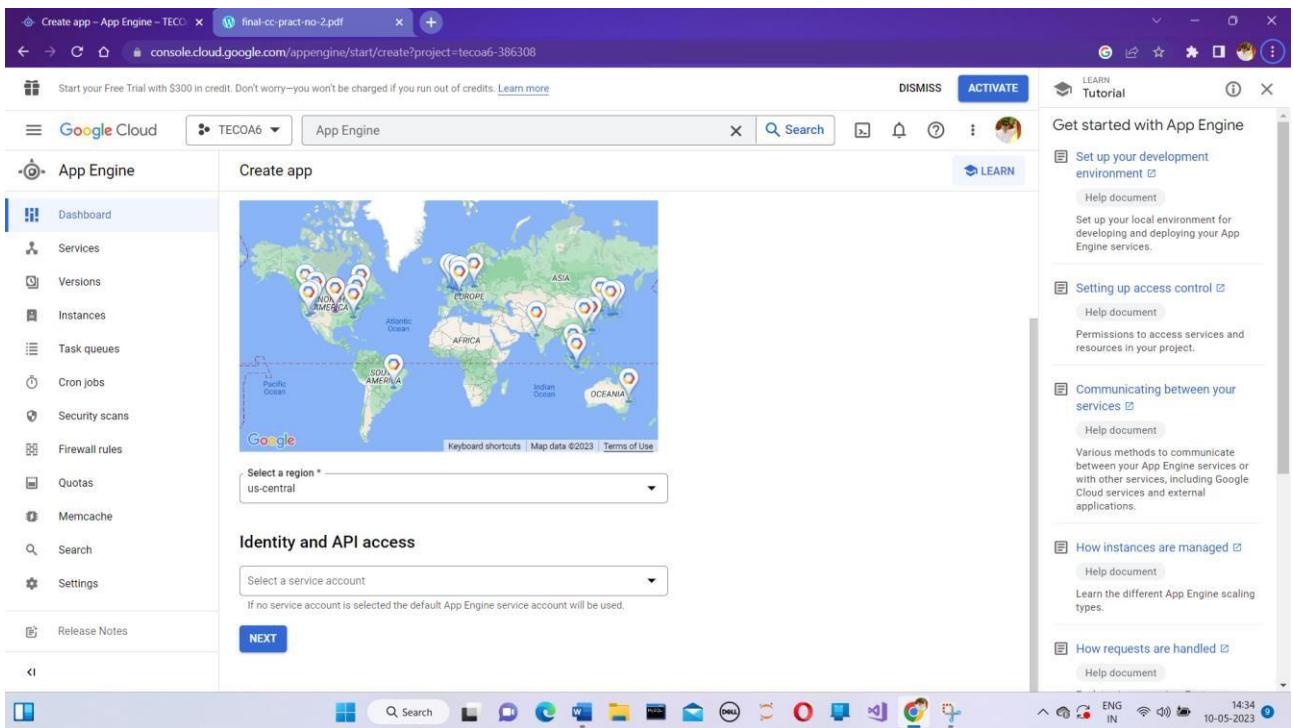
Step5:- In a search bar type search App Engine

The screenshot shows the Google Cloud Platform dashboard for project TECOA6. A search bar at the top contains the text 'App Engine'. The left sidebar has a 'DASHBOARD' tab selected. The main content area displays 'PRODUCTS & PAGES' related to App Engine, including 'Application Settings' and 'App Engine Managed app platform'. Below this is a section for 'DOCUMENTATION & TUTORIALS' with links like 'Run a Python web application on Google Compute Engine and Cloud SQL' and 'Create a client-server application on Compute Engine'. On the right side, there's a sidebar titled 'Google Cloud Platform status' showing 'Multiple Google Cloud services in the europe-west9-a region are impacted'.

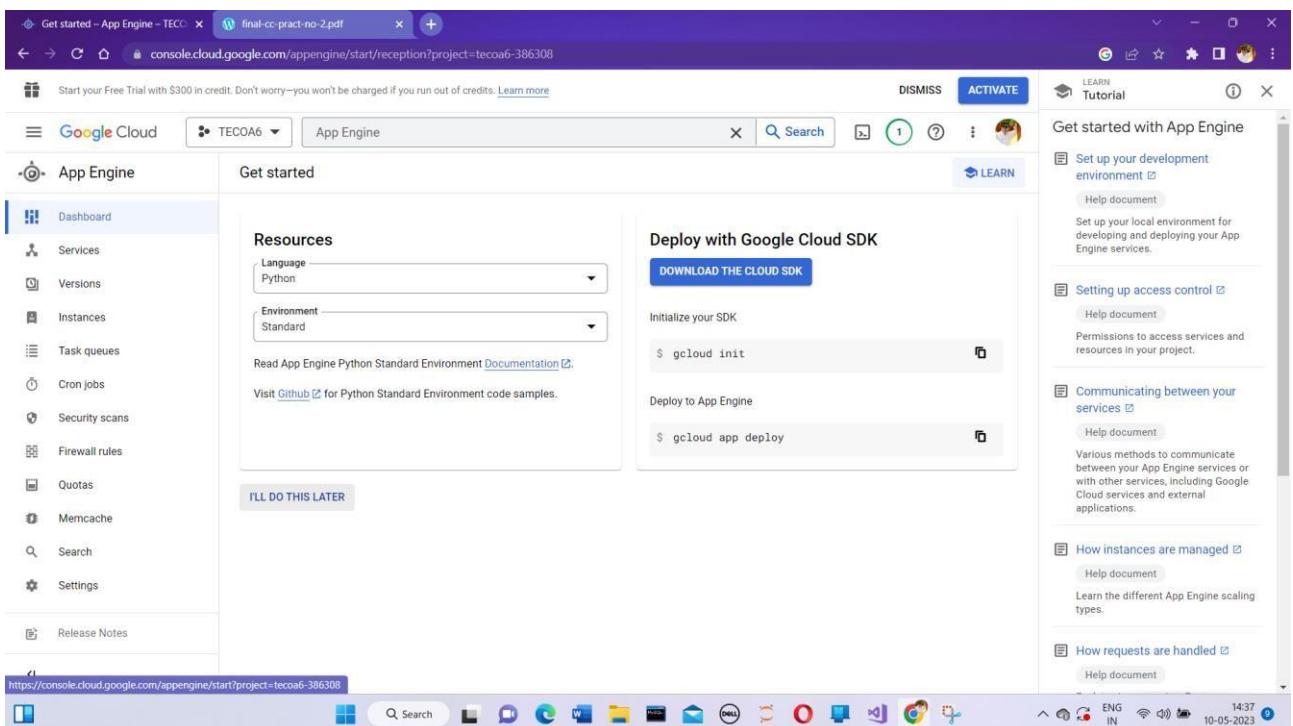
Step 6:-Click on App Engine and Following screen will appear& Click on Create Application.

The screenshot shows the 'App Engine - App Engine - TECOA6' start page. 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 main content area features a 'Welcome to App Engine' message: 'Build scalable apps in any language on Google's infrastructure' with a prominent 'CREATE APPLICATION' button. To the right, there's a 'LEARN Tutorial' sidebar with sections for 'Get started with App Engine', including 'Set up your development environment', 'Setting up access control', 'Communicating between your services', 'How instances are managed', and 'How requests are handled'.

Step 7:- Click on next



Step 8:- Scroll down and click on I'll do this later.



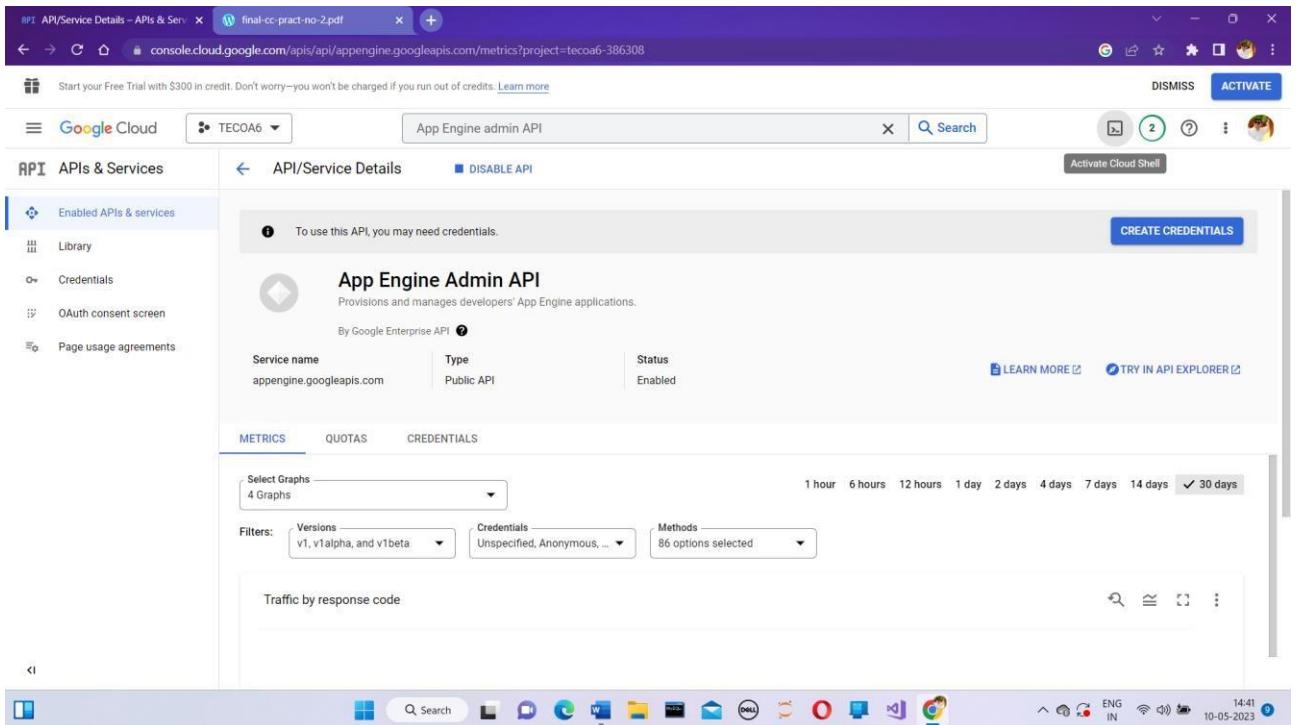
Step 9:- In search bar type App Engine Admin API.

The screenshot shows the Google Cloud Platform console with the search bar containing 'App Engine Admin API'. The results page displays several items under 'DOCUMENTATION & TUTORIALS' and 'MARKETPLACE'. On the right side, there is a sidebar titled 'Get started with App Engine' containing links to various documentation sections like 'Set up your development environment', 'Setting up access control', and 'Communicating between your services'. The status bar at the bottom indicates the URL as <https://console.cloud.google.com/marketplace/product/google/appengine.googleapis.com?q=search&referrer=search&project=tecoa6-386308>.

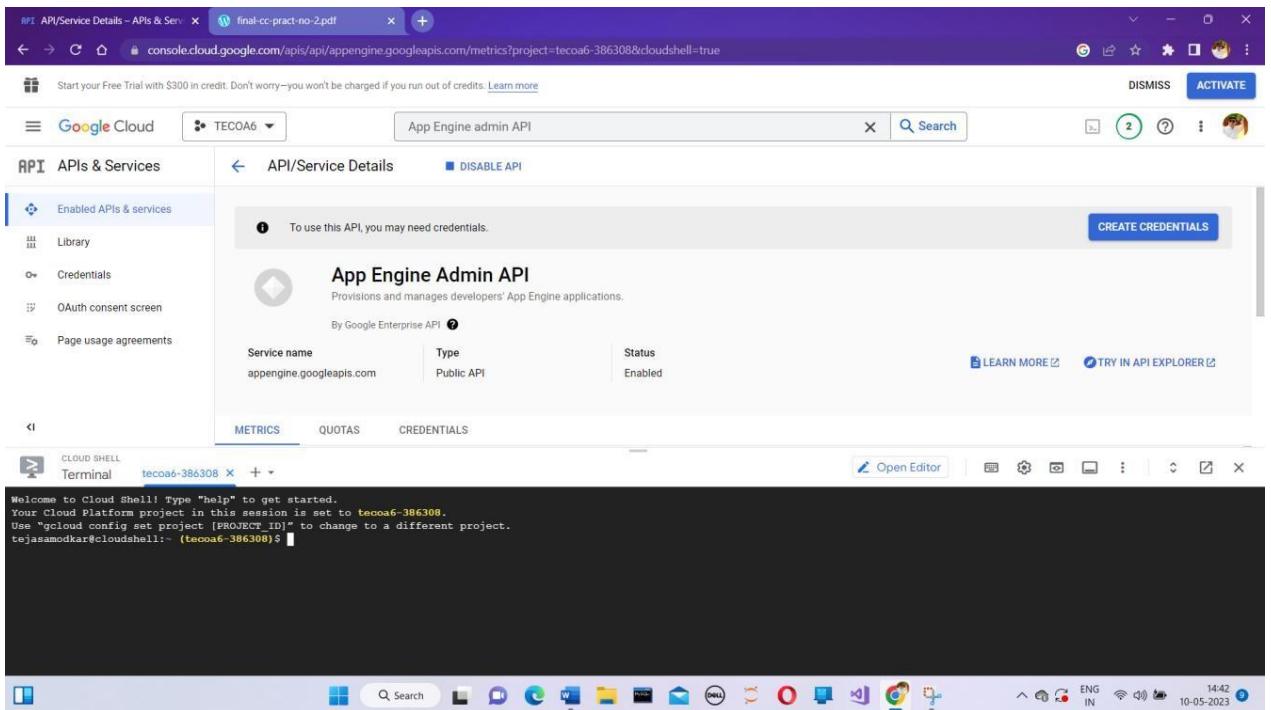
Step 10:- Click Enables

The screenshot shows the Google Cloud Marketplace product details page for the 'App Engine Admin API'. The page includes a large image of a smartphone, a brief description, and two main buttons: 'ENABLE' and 'TRY THIS API'. Below these buttons is a link 'Click to enable this API'. At the bottom of the page, there are tabs for 'OVERVIEW', 'DOCUMENTATION', and 'RELATED PRODUCTS'. The 'OVERVIEW' tab is selected. The 'Additional details' section provides information such as Type: SaaS & APIs, Last updated: 7/22/22, Category: Compute, Google Enterprise APIs, and Service name: appengine.googleapis.com. The status bar at the bottom indicates the URL as <https://console.cloud.google.com/marketplace/product/google/appengine.googleapis.com?q=search&referrer=search&project=tecoa6-386308>.

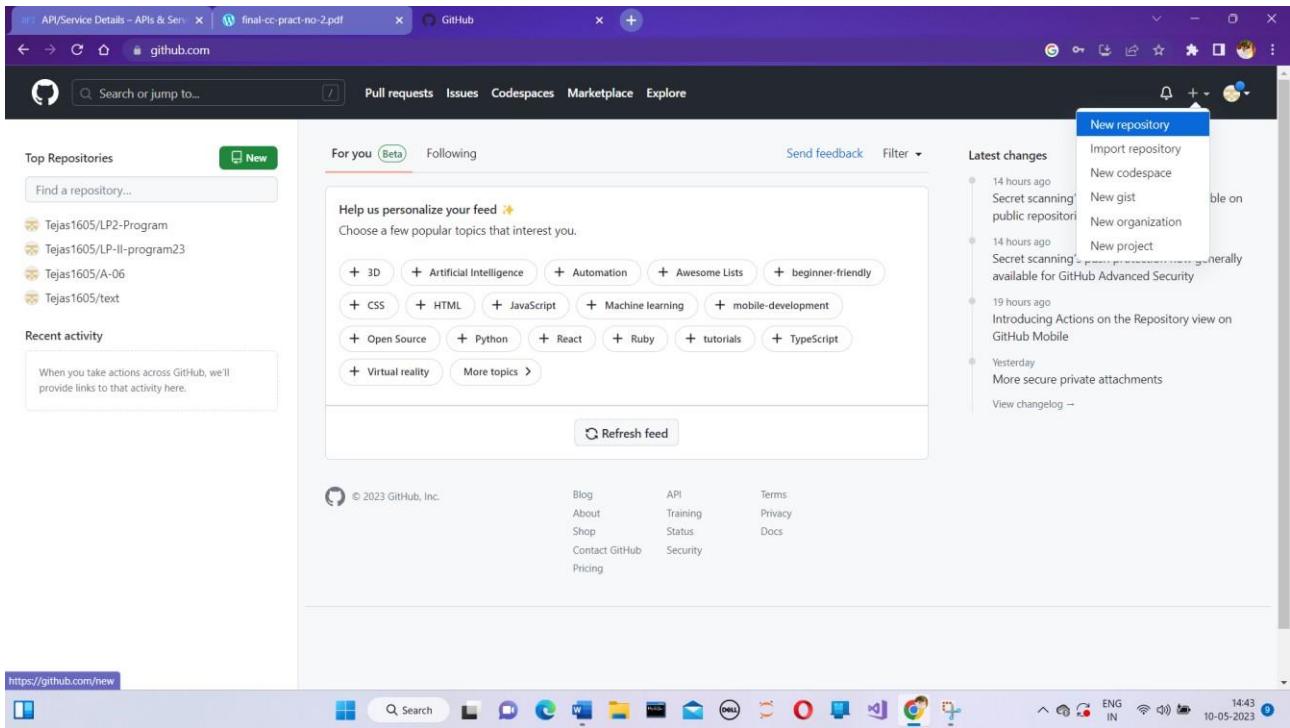
Step 11:- Click Activate Cloud Shell:-



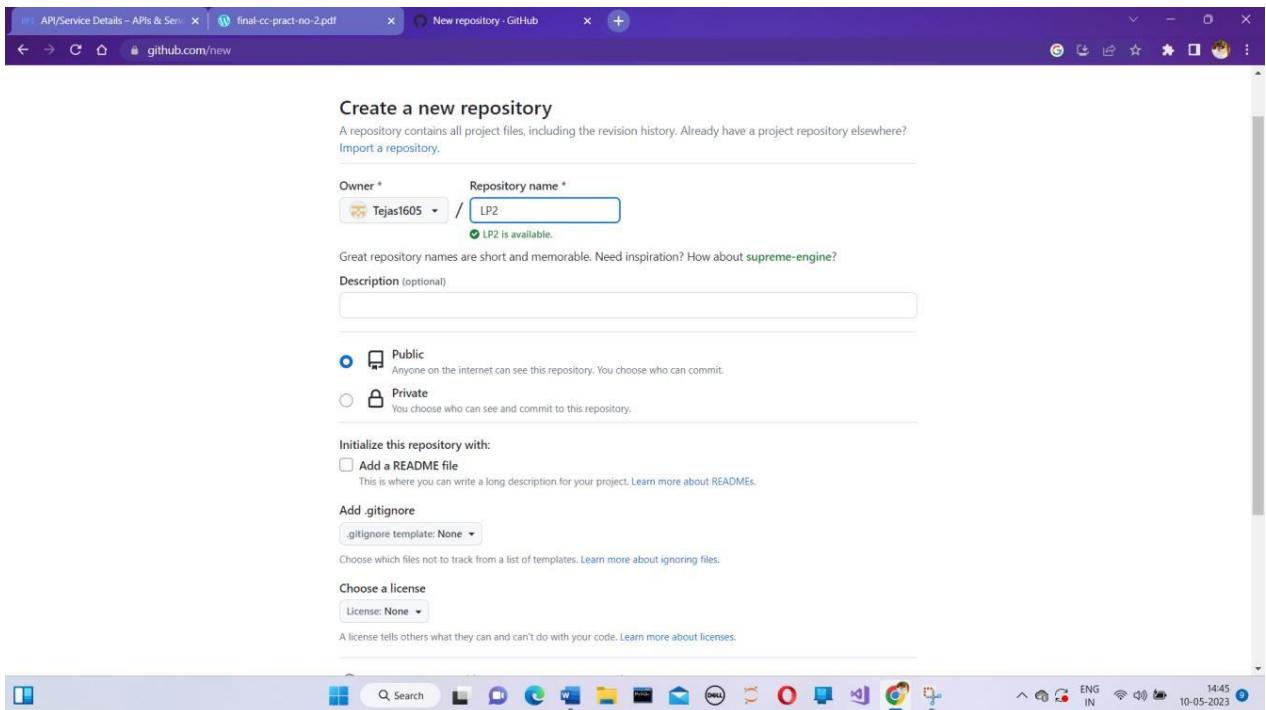
Step 12 :- Following screen will appear:-



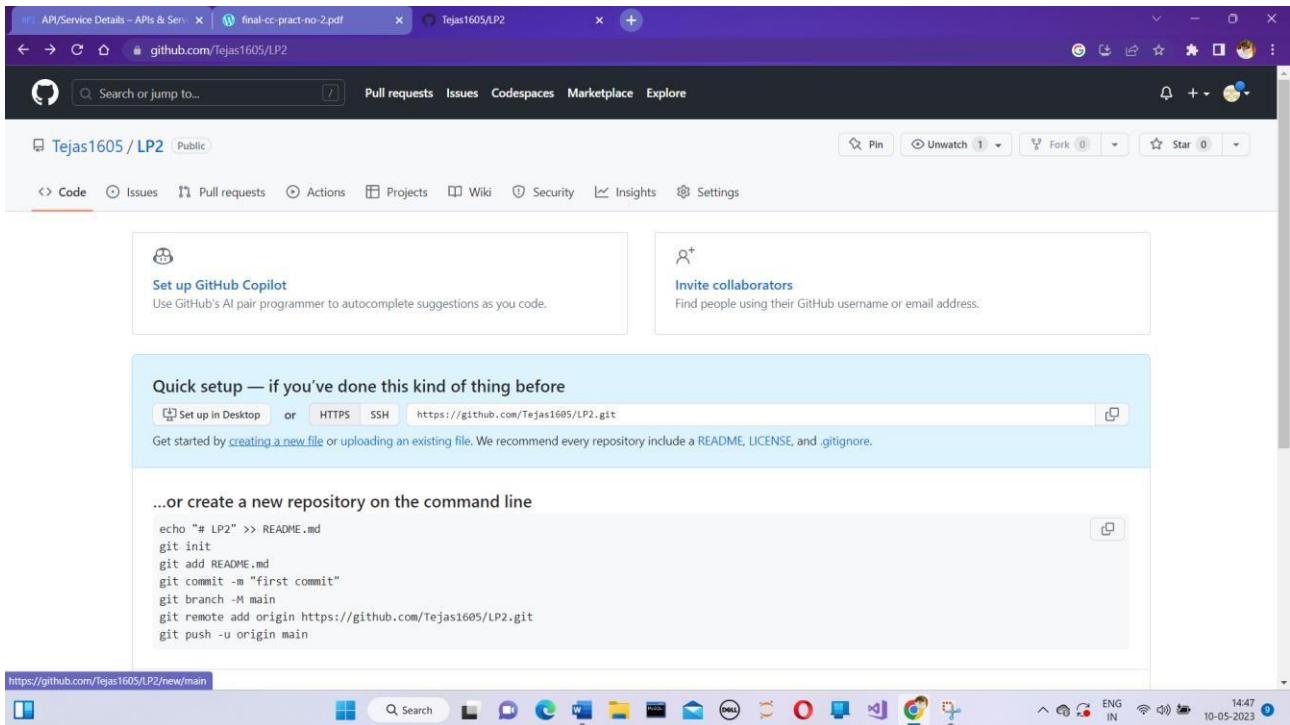
Step 13:- Login into your GitHub account and click on new repository.



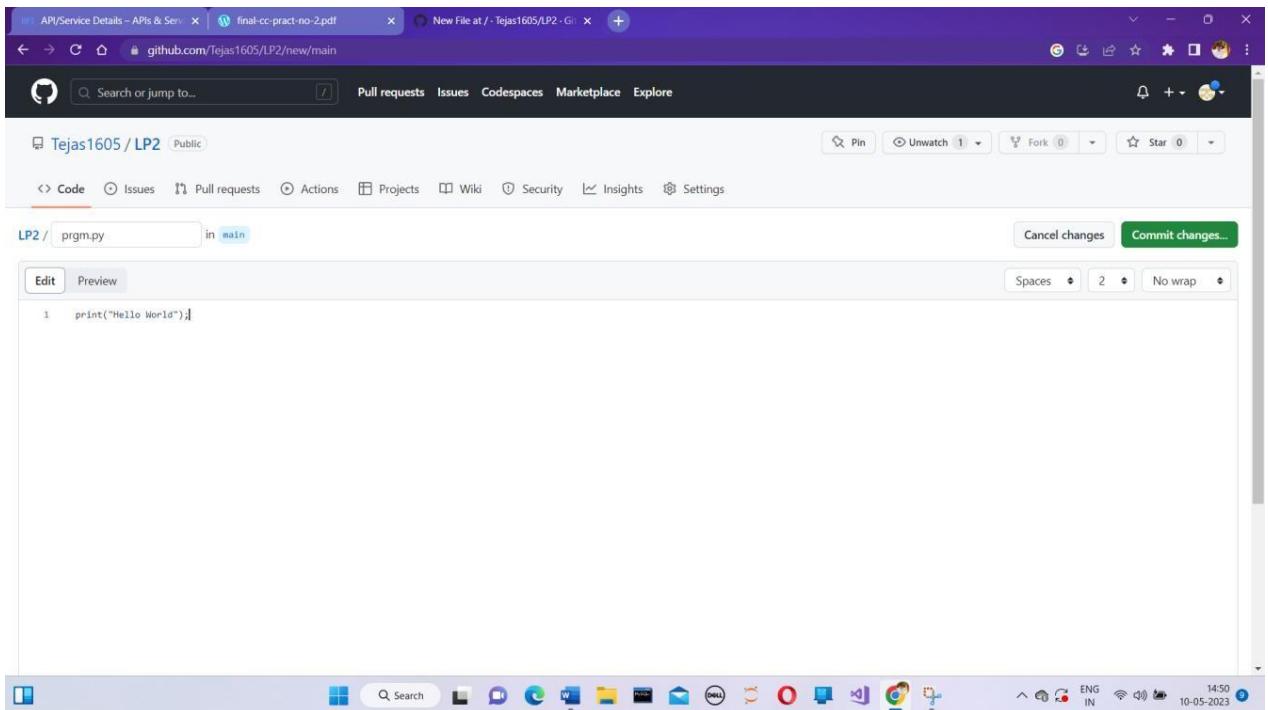
Step 14 :- Give name to your repository and click create.



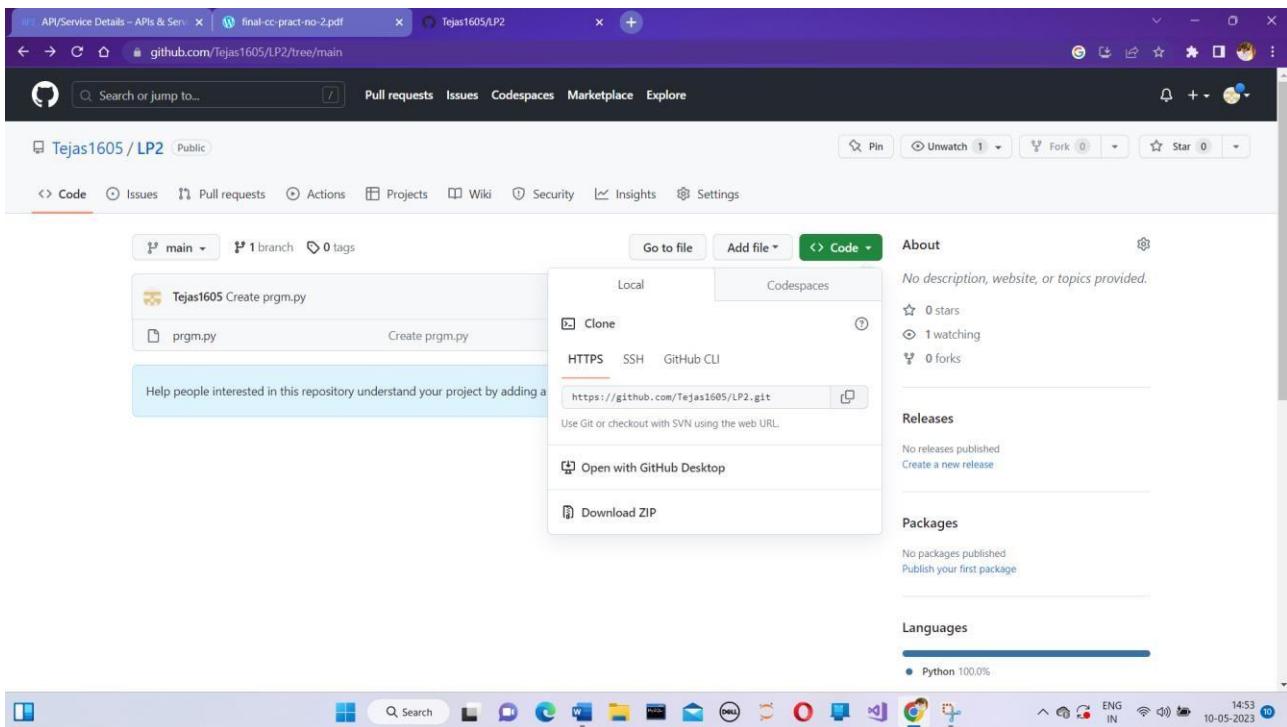
Step 15:- Click on creating new file



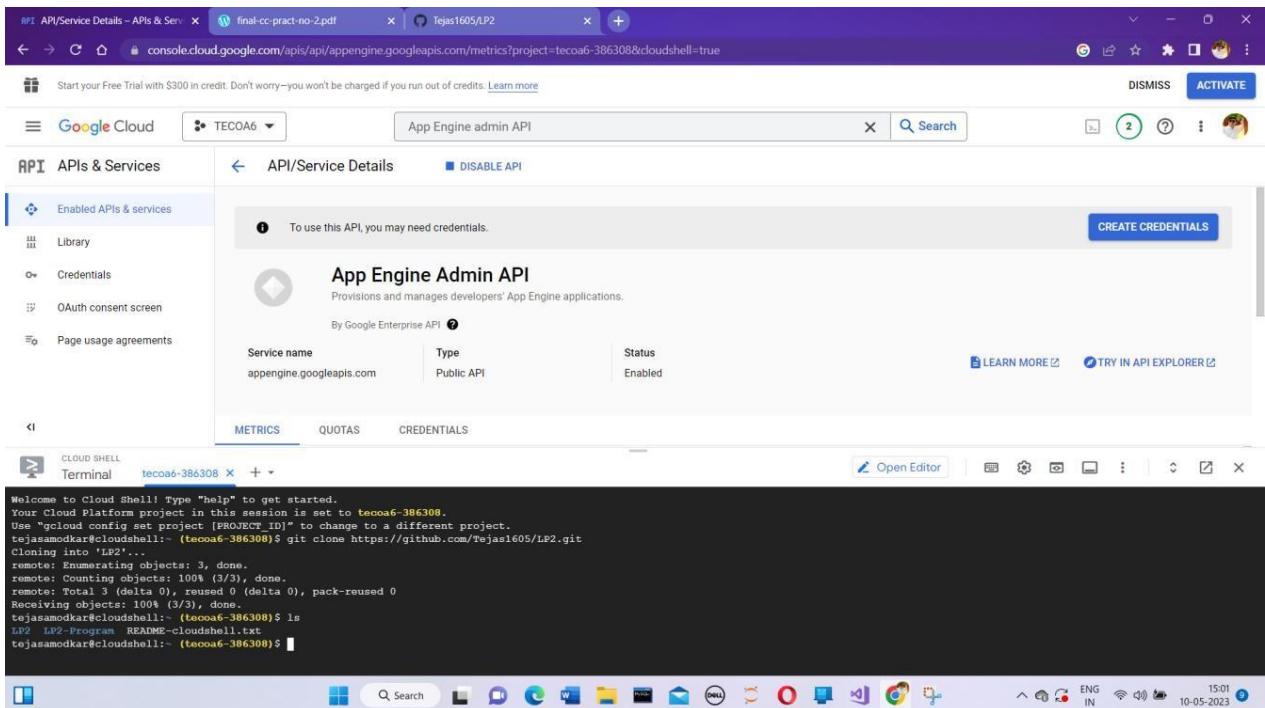
Step 16 :- Give name to the python file & type your code.



Step 17:- Click on code and copy URL.



Step 18:- Go to cloud and type – git clone and paste url.



Step 19 :- Type ls

The screenshot shows the Google Cloud Platform interface. In the top navigation bar, there are tabs for 'API Service Details - APIs & Services', 'final-cc-pract-no-2.pdf', and 'Tejas1605/LP2'. Below the navigation bar, there is a banner with a message about a free trial and a 'DISMISS' button. The main content area is titled 'API/Service Details' for the 'App Engine Admin API'. It shows the service name as 'appengine.googleapis.com', type as 'Public API', and status as 'Enabled'. There are 'CREATE CREDENTIALS' and 'TRY IN API EXPLORER' buttons. At the bottom, there are tabs for 'METRICS', 'QUOTAS', and 'CREDENTIALS'. Below these tabs, a terminal window is open with the command 'ls' entered, showing the directory structure of the repository. The terminal window has tabs for 'CLOUD SHELL' and 'Terminal' with the identifier 'tecoa6-386308'. The system tray at the bottom right shows the date as 10-05-2023.

```
Welcome to Cloud Shell! Type "help" to get started.  
Your Cloud Platform project in this session is set to tecoa6-386308.  
Use "gcloud config set project [PROJECT_ID]" to change to a different project.  
tejasamodkar@cloudshell:~(tecoa6-386308)$ git clone https://github.com/Tejas1605/LP2.git  
Cloning into 'LP2'...  
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.  
tejasamodkar@cloudshell:~(tecoa6-386308)$ ls  
LP2 LP2-Program README-cloudshell.txt  
tejasamodkar@cloudshell:~(tecoa6-386308)$
```

Step 20 :- Enter cd-repository name.

The screenshot shows the Google Cloud Platform interface, identical to the previous one but with a different terminal command. The terminal window now displays the command 'cd' followed by the repository name. The rest of the interface, including the API details and system tray, remains the same.

```
Use "gcloud config set project [PROJECT_ID]" to change to a different project.  
tejasamodkar@cloudshell:~(tecoa6-386308)$ git clone https://github.com/Tejas1605/LP2.git  
Cloning into 'LP2'...  
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.  
tejasamodkar@cloudshell:~(tecoa6-386308)$ ls  
LP2 LP2-Program README-cloudshell.txt  
tejasamodkar@cloudshell:~(tecoa6-386308)$ cd LP2  
tejasamodkar@cloudshell:~/LP2 (tecoa6-386308)$ ls  
prgm.py  
tejasamodkar@cloudshell:~/LP2 (tecoa6-386308)$
```

Step 21:- Type ls and to run python code type python-program name.

The screenshot shows the Google Cloud API Service Details page for the App Engine Admin API. The left sidebar shows 'Enabled APIs & services' with 'appengine.googleapis.com' selected. The main panel displays the 'App Engine Admin API' details, including its description ('Provisions and manages developers' App Engine applications.'), provider ('By Google Enterprise API'), service name ('appengine.googleapis.com'), type ('Public API'), and status ('Enabled'). Below this are tabs for 'METRICS', 'QUOTAS', and 'CREDENTIALS'. At the bottom, there's a terminal window showing command-line output related to cloning objects and running a Python script named 'prgm.py' which prints 'Hello World'.

```
Cloning into 'LP2'...
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.
tejasamodkar@cloudshell:~/LP2$ ls
LP2 LP2-program README-cloudshell.txt
tejasamodkar@cloudshell:~/LP2$ cd LP2
tejasamodkar@cloudshell:~/LP2$ ./prgm.py
Hello World
tejasamodkar@cloudshell:~/LP2$
```

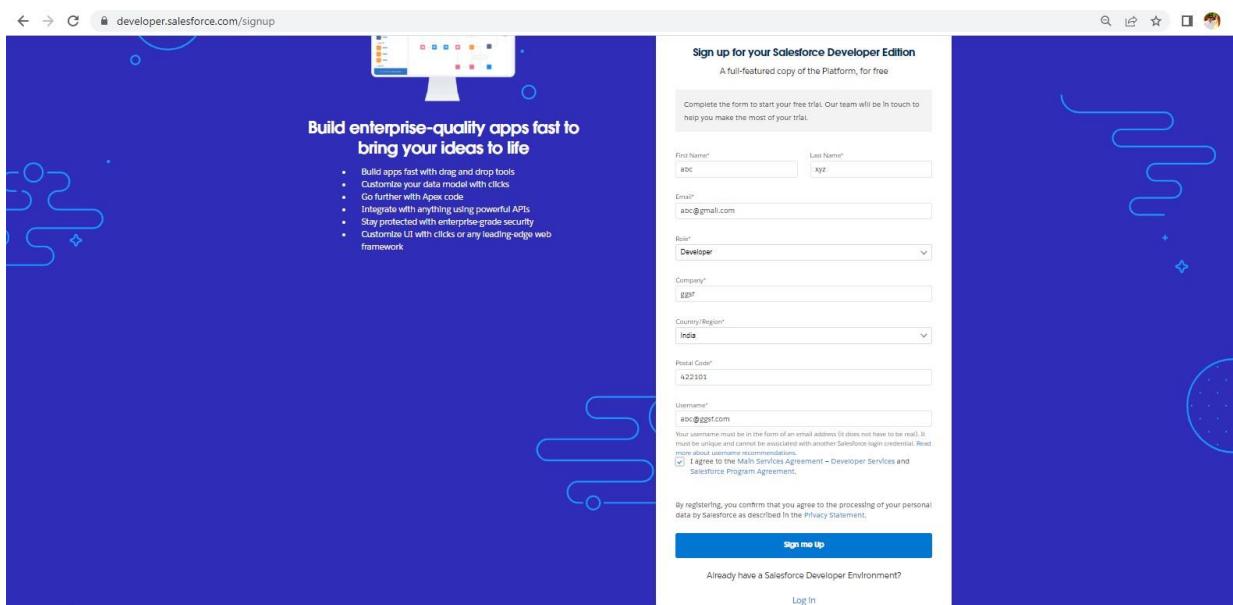
PRACTICAL NO : 9

Title : Creating an Application in SalesForce.com using Apex programming Language

STEPS TO CREATE AN APPLICATION :

Step No 1: Create new org:

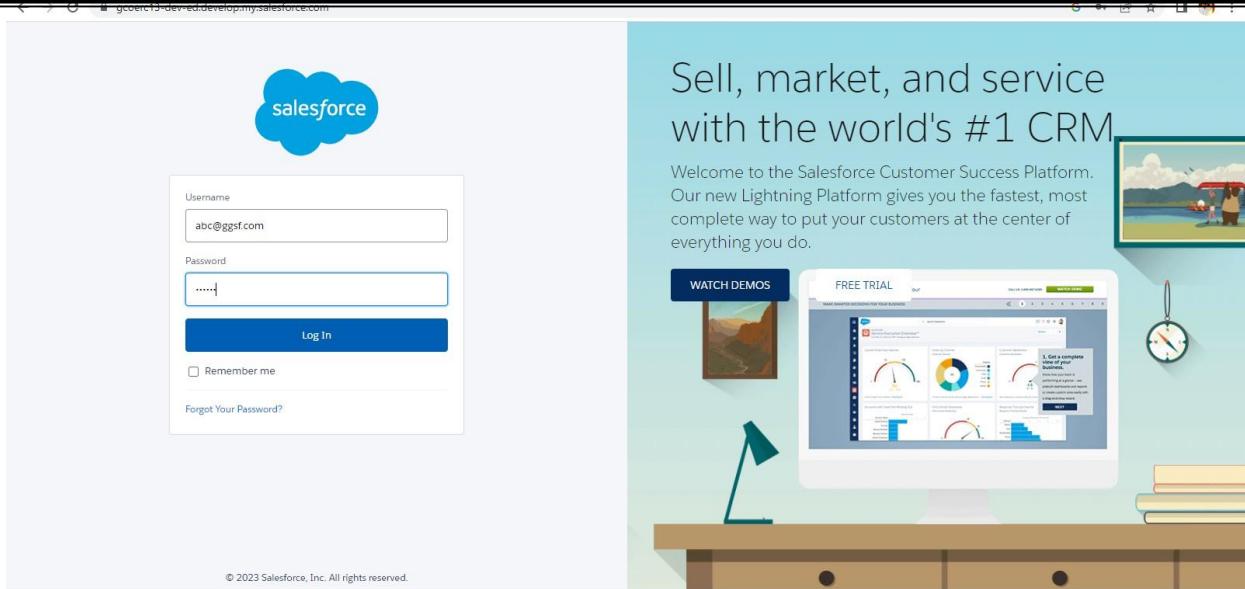
<https://developer.salesforce.com/signup>



Step No 2: After signup, loging using following URL

<https://login.salesforce.com/>

Step No 3: Login Page (Enter your credential to login)



1. Open Developer Console

2. File ---> New ---> Select Apex Class

Type below mentioned code

```
public class firstClass1 {
    public static void Addition()
    {
        Integer a = 4;
        Integer b = 5;
        Integer c = a + b;
        Integer d = 4 + 5;
        Integer e = a + 5;
        System.debug('Add =' + c);
        System.debug('Add =' + d);
        System.debug('Add =' + e);
    }
    public static void Subtraction()
    {

        Integer a = 4;
        Integer b = 5;
        Integer c1 = a - b;
        Integer d1 = b - a;
        Integer e1 = 4 - 5;
        Integer f1 = a - 5;

        System.debug('Sub =' + c1);
        System.debug('Sub =' + d1);
        System.debug('Sub =' + e1);
        System.debug('Sub =' + f1);
    }
    public static void Multi()
    {
        Integer a = 4;
        Integer b = 5;
```

```
Integer c = a + b;
Integer d = 4 * 5;
Integer e = a * 5;

System.debug(c);
System.debug(d);
System.debug(e);

Integer f = -4;
Integer g = a * f;
System.debug(g);

}

public static void Div()
{
    Integer a = 4;
    Integer b = 5;
    Integer c = a / b;
    Integer d = 4 / 5;
    Integer e = a / 5;

    System.debug(c);
    System.debug(d);
    System.debug(e);
}
```

```
Developer Console - Google Chrome
gcoerc13-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage
File Edit Debug Test Workspace Help < >
firstClass1.apxc Log executeAnonymous @3/9/2023, 11:35:51 AM Log executeAnonymous @3/9/2023, 11:37:23 AM
Code Coverage: None API Version: 57 Go To
1 public class firstClass1 {
2     public static void Addition()
3     {
4         Integer a = 4;
5         Integer b = 5;
6         Integer c = a + b;
7         Integer d = 4 + 5;
8         Integer e = a + 5;
9         System.debug('Add = ' + c);
10        System.debug('Add =' + d);
11        System.debug('Add =' + e);
12    }
13    public static void Subtraction()
14    {
15
16        Integer a = 4;
17        Integer b = 5;
18        Integer c1 = a - b;
19        Integer d1 = b - a;
20        Integer e1 = 4 - 5;
21        Integer f1 = a - 5;
22
23        System.debug('Sub =' + c1);
24        System.debug('Sub =' + d1);
25        System.debug('Sub =' + e1);
26        System.debug('Sub =' + f1);
27    }
28 }
```

The screenshot shows the Salesforce Developer Console interface. The title bar says "Developer Console - Google Chrome" and the address bar shows "gcoerc13-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage". The menu bar includes File, Edit, Debug, Test, Workspace, Help, and a Go To button. A tab bar at the top has "firstClass1.apxc" selected, along with two log tabs: "Log executeAnonymous @3/9/2023, 11:35:51 AM" and "Log executeAnonymous @3/9/2023, 11:37:23 AM". Below the tabs is a dropdown for "Code Coverage: None" and "API Version: 57". The main area contains the following Apex code:

```
28     }
29     public static void Multi()
30     {
31         Integer a = 4;
32         Integer b = 5;
33         Integer c = a * b;
34         Integer d = 4 * 5;
35         Integer e = a * 5;
36
37         System.debug(c);
38         System.debug(d);
39         System.debug(e);
40
41         Integer f = -4;
42         Integer g = a * f;
43         System.debug(g);
44     }
45     public static void Div()
46     {
47         Integer a = 4;
48         Integer b = 5;
49         Integer c = a / b;
50         Integer d = 4 / 5;
51         Integer e = a / 5;
52
53         System.debug(c);
54         System.debug(d);
55         System.debug(e);
56     }
```

3. Click on Debug ---> Open Execute Acronymous Window

4. Type below code(Apex Code)

```
firstClass1.Addition();
firstClass1.Subtaction();
firstClass1.Multi();
firstClass1.Div();
```

5. Click on Open log then Execute code

Enter Apex Code

```

1 firstClass1.Addition();
2 firstClass1.Subtraction();
3 firstClass1.Multi();
4 FirstClass1.Div();|

```

Open Log **Execute** Execute Highlighted

6. Click on Debug only (You will get output)

Developer Console - Google Chrome

gcoerc13-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage

File ▾ Edit ▾ Debug ▾ Test ▾ Workspace ▾ Help ▾ < >

Log executeAnonymous @3/9/2023, 11:37:23 AM | Log executeAnonymous @3/9/2023, 11:49:29 AM | **Log executeAnonymous @3/9/2023, 11:50:57 AM**

Execution Log		
Timestamp	Event	Details
11:50:57:002	USER_INFO	[EXTERNAL]0052w00000Flbbu!tejas@gcoerc.com (GMT+05:30) India Standard Time (Asia/Kolkata) GMT+05:30
11:50:57:002	EXECUTION_ST...	
11:50:57:002	CODE_UNIT_ST...	[EXTERNAL]execute_anonymous_apex
11:50:57:002	HEAP_ALLOCATE	[79]Bytes:3
11:50:57:002	HEAP_ALLOCATE	[84]Bytes:152
11:50:57:002	HEAP_ALLOCATE	[399]Bytes:408
11:50:57:002	HEAP_ALLOCATE	[412]Bytes:408
11:50:57:002	HEAP_ALLOCATE	[520]Bytes:48
11:50:57:002	HEAP_ALLOCATE	[139]Bytes:6
11:50:57:002	HEAP_ALLOCATE	[EXTERNAL]Bytes:7
11:50:57:003	STATEMENT_EX...	[1]
11:50:57:003	STATEMENT_EX...	[1]
11:50:57:003	HEAP_ALLOCATE	[52]Bytes:5
11:50:57:003	HEAP_ALLOCATE	[58]Bytes:5
11:50:57:003	HEAP_ALLOCATE	[66]Bytes:7
11:50:57:003	SYSTEM_MODE...	false
11:50:57:003	HEAP_ALLOCATE	[1]Bytes:5
11:50:57:004	HEAP_ALLOCATE	[1]Bytes:10
11:50:57:004	HEAP_ALLOCATE	[1]Bytes:40
11:50:57:004	METHOD_ENTRY	[1]01p2w00000dJUSg firstClass1.firstClass1()
11:50:57:004	STATEMENT_EX...	[1]
11:50:57:004	STATEMENT_EX...	[1]
11:50:57:004	METHOD_EXIT	[1]firstClass1
11:50:57:004	METHOD_ENTRY	[1]01p2w00000dJUSg firstClass1.Addition()
11:50:57:004	STATEMENT_EX...	[3]
11:50:57:004	STATEMENT_EX...	[4]
11:50:57:004	VARIABLE_SCO...	[4] a Integer false false
11:50:57:004	HEAP_ALLOCATE	[4]Bytes:4
11:50:57:004	VARIABLE_ASSI...	[4] a 4

This Frame Executable Debug Only Filter Click here to filter the log

Logs, Tests, and Problems

Output :

The screenshot shows the Salesforce Developer Console interface. The title bar reads "Developer Console - Google Chrome" and the URL is "gcoerc13-dev-ed.develop.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage". The menu bar includes File, Edit, Debug, Test, Workspace, Help, and a set of navigation icons. Below the menu is a toolbar with three tabs: "pxc" (selected), "Log executeAnonymous @3/9/2023, 11:35:51 AM", "Log executeAnonymous @3/9/2023, 11:37:23 AM", and "Log executeAnonymous @3/9/2023, 11:49:29 AM" (highlighted in orange). The main area is titled "Execution Log" and contains a table with three columns: Timestamp, Event, and Details. The table lists 20 log entries from 11:49:29:008 to 11:49:29:013. The details column shows various DEBUG statements related to variable assignments and calculations. At the bottom of the log table, there are checkboxes for "This Frame", "Executable", "Debug Only" (which is checked), and "Filter", followed by a link "Click here to filter the log". The status bar at the bottom of the console window displays "Logs, Tests, and Problems".

Timestamp	Event	Details
11:49:29:008	USER_DEBUG	[9] DEBUG Add = 9
11:49:29:010	USER_DEBUG	[10] DEBUG Add =9
11:49:29:010	USER_DEBUG	[11] DEBUG Add =9
11:49:29:011	USER_DEBUG	[24] DEBUG Sub =-1
11:49:29:011	USER_DEBUG	[25] DEBUG Sub =1
11:49:29:011	USER_DEBUG	[26] DEBUG Sub =-1
11:49:29:011	USER_DEBUG	[27] DEBUG Sub =-1
11:49:29:012	USER_DEBUG	[37] DEBUG 20
11:49:29:012	USER_DEBUG	[38] DEBUG 20
11:49:29:012	USER_DEBUG	[39] DEBUG 20
11:49:29:013	USER_DEBUG	[43] DEBUG -16
11:49:29:013	USER_DEBUG	[53] DEBUG 0
11:49:29:013	USER_DEBUG	[54] DEBUG 0
11:49:29:013	USER_DEBUG	[55] DEBUG 0

PRACTICAL NO : 10

Title : Design and develop custom Application (Mini Project) using Salesforce Cloud.

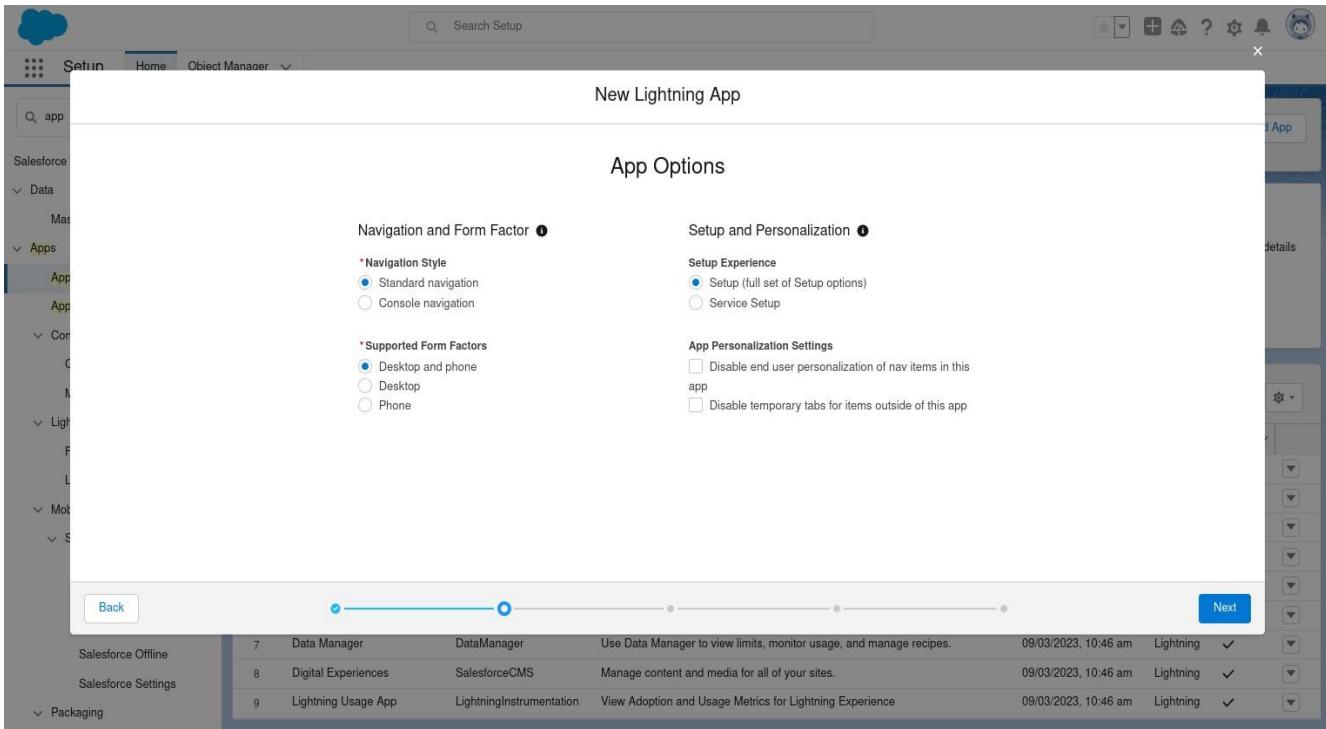
STEPS TO CREATE AN APPLICATION(mini project) :

1. Open salesforce in Lightning Experience.
2. Open App Launcher • View all • Quick Search • App Manager • Click on New Lightning App.

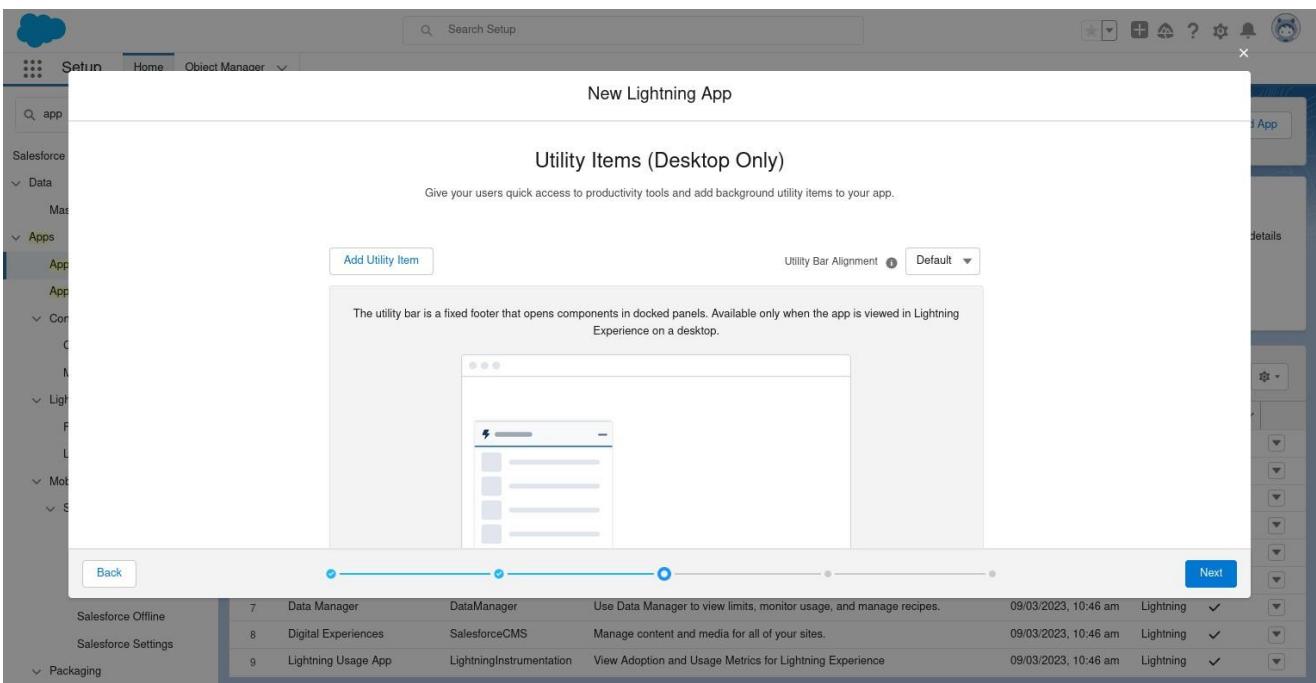
App Name ↑	Developer Name	Description	Last Modified Date	App ...	Vis...
All Tabs	AllTabSet	Build CRM Analytics dashboards and apps	09/03/2023, 10:46 am	Classic	✓
Analytics Studio	Insights	Build CRM Analytics dashboards and apps	09/03/2023, 10:46 am	Classic	✓
App Launcher	AppLauncher	App Launcher tabs	09/03/2023, 10:46 am	Classic	✓
Bolt Solutions	LightningBolt	Discover and manage business solutions designed for your industry.	09/03/2023, 10:48 am	Lightning	✓
Community	Community	Salesforce CRM Communities	09/03/2023, 10:46 am	Classic	✓
Content	Content	Salesforce CRM Content	09/03/2023, 10:46 am	Classic	✓
Data Manager	DataManager	Use Data Manager to view limits, monitor usage, and manage recipes.	09/03/2023, 10:46 am	Lightning	✓
Digital Experiences	SalesforceCMS	Manage content and media for all of your sites.	09/03/2023, 10:46 am	Lightning	✓
Lightning Usage App	LightningInstrumentation	View Adoption and Usage Metrics for Lightning Experience	09/03/2023, 10:46 am	Lightning	✓

3. fill the mention information.

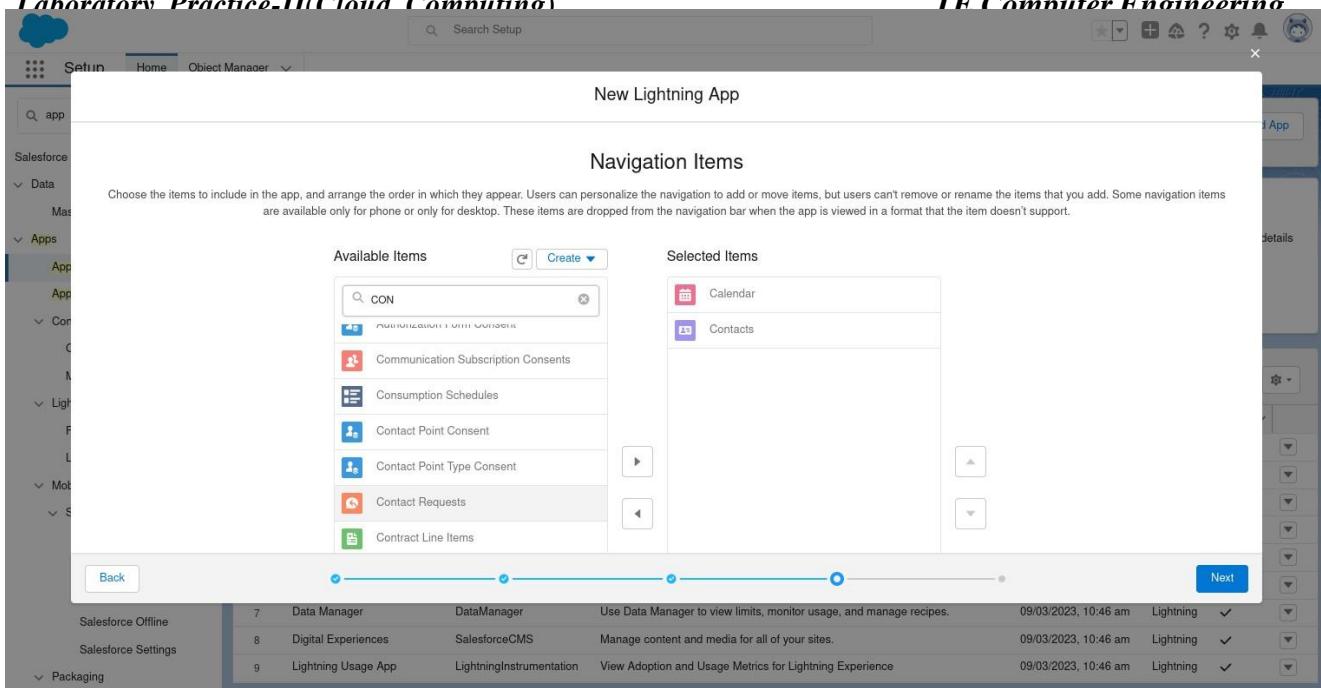
4.Next ▪ App option(optional).



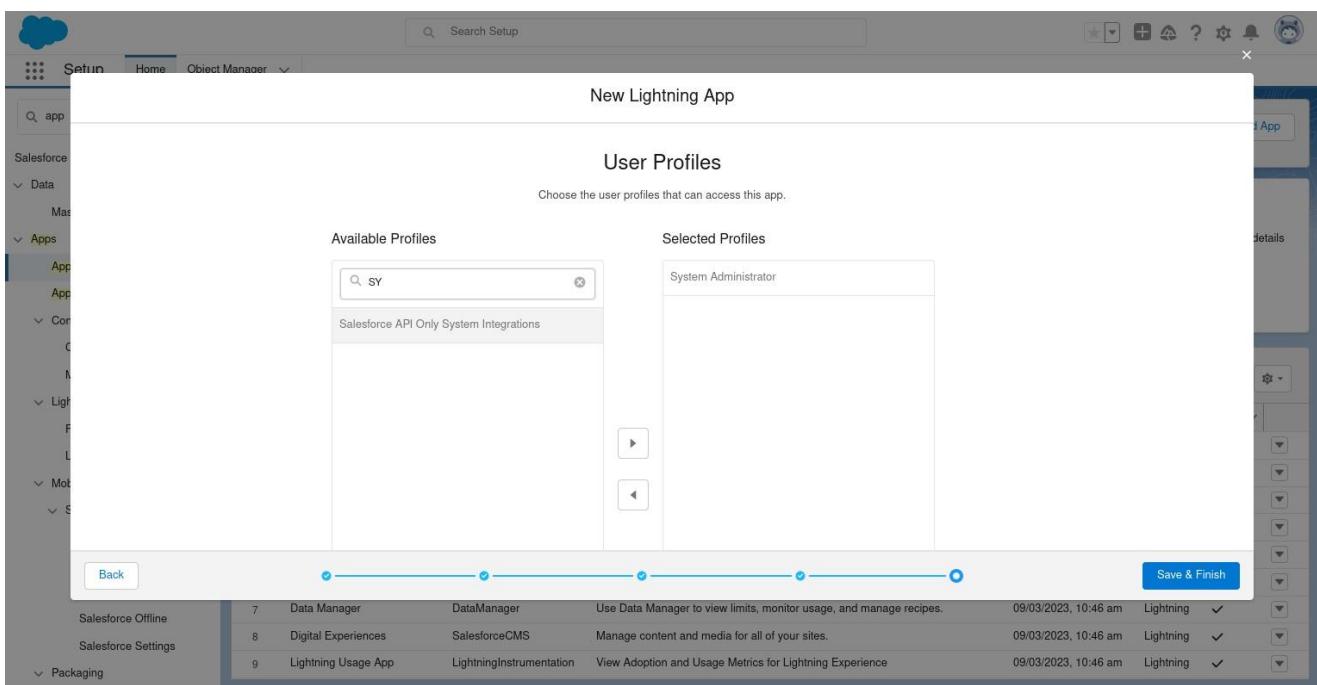
5.Next ▪ Add Utility (optional).



6.Next ▪ Navigation Item(you can add item).



7. Next ▪ User Profile(use the user profilethat can access this app) ▪ Save and Finish.



8. You will get Lightning Experience App Manager Window.

The screenshot shows the Salesforce App Manager window. On the left, there's a sidebar with a search bar for 'app'. Under 'Salesforce Mobile App', there's a section for 'Data' with 'Mass Transfer Approval Requests'. Under 'Apps', 'App Manager' is selected, showing a list of installed apps. A toggle switch for 'Enable App Cloning' is set to 'Disabled'. The main area displays a table with 24 items, sorted by App Name, filtered by All appmenuitems - TabSet Type. The columns are: App Name, Developer Name, Description, Last Modified Date, App Type, and Visibility. The table includes rows for Marketing, Platform, Queue Management, Sales, and various Salesforce and TECOA6 custom applications.

App Name	Developer Name	Description	Last Modified Date	App Type	Vis...
10 Marketing	Marketing	Best-in-class on-demand marketing automation	09/03/2023, 10:46 am	Classic	✓
11 Platform	Platform	The fundamental Lightning Platform	09/03/2023, 10:46 am	Classic	✓
12 Queue Management	QueueManagement	Create and manage queues for your business.	09/03/2023, 10:46 am	Lightning	✓
13 Sales	Sales	The world's most popular sales force automation (SFA) solution	09/03/2023, 10:46 am	Classic	✓
14 Sales	LightningSales	Manage your sales process with accounts, leads, opportunities, and more	09/03/2023, 10:49 am	Lightning	✓
15 Sales Console	LightningSalesConsole	(Lightning Experience) Lets sales reps work with multiple records on one screen	09/03/2023, 10:46 am	Lightning	✓
16 Salesforce Chatter	Chatter	The Salesforce Chatter social network, including profiles and feeds	09/03/2023, 10:46 am	Classic	✓
17 Salesforce Scheduler Setup	LightningScheduler	Set up personalized appointment scheduling.	09/03/2023, 10:48 am	Lightning	✓
18 Service	Service	Manage customer service with accounts, contacts, cases, and more	09/03/2023, 10:46 am	Classic	✓
19 Service Console	LightningService	(Lightning Experience) Lets support agents work with multiple records across c...	09/03/2023, 10:46 am	Lightning	✓
20 Site.com	Sites	Build pixel-perfect, data-rich websites using the drag-and-drop Site.com applica...	09/03/2023, 10:46 am	Classic	✓
21 Subscription Management	RevenueCloudConsole	Get started automating your revenue processes	09/03/2023, 10:46 am	Lightning	✓
22 TECOA6	TECOA6		13/04/2023, 11:43 am	Lightning	✓
23 TECOA60	TECOA60		13/04/2023, 11:50 am	Lightning	✓
24 TECOA61	TECOA61		13/04/2023, 12:22 pm	Lightning	✓

9. Search Your App by its name in App Launcher. just by entering some character, you can see app name with its logo.

The screenshot shows the Salesforce App Launcher window. On the left, there's a sidebar with a search bar for 'app'. Under 'Salesforce Mobile App', there's a section for 'Data' with 'Mass Transfer Approval Requests'. Under 'Apps', 'App Manager' is selected, showing a list of installed apps. The main area displays a grid of installed apps. A search bar at the top right says 'Search apps or items...'. The grid includes icons and names for Marketing, Platform, Queue Management, Sales, and various Salesforce and TECOA6 custom applications. At the bottom, there's a table for 'All Items' showing the same list of apps from the App Manager.

App Name	Developer Name	Description	Last Modified Date	App Type	Vis...
22 TECOA6	TECOA6		13/04/2023, 11:43 am	Lightning	✓
23 TECOA60	TECOA60		13/04/2023, 11:50 am	Lightning	✓
24 TECOA61	TECOA61		13/04/2023, 12:22 pm	Lightning	✓

10. Click on App you can see your application appearance in window Output.

This screenshot shows a contacts management interface. At the top, there's a navigation bar with icons for cloud, TECOA6, Contacts, and Calendar. A search bar is at the top right. Below the navigation is a header for 'Recently Viewed' with a dropdown arrow and a plus sign. It also includes buttons for New, Import, Add to Campaign, and Send List Email. A search bar and filter icons are below the header. The main area displays a message: '0 items • Updated a few seconds ago' followed by 'You haven't viewed any Contacts recently. Try switching list views.' There are columns for Name, Account Name, Account Site, Phone, Email, and Contact Owner Alias.

This screenshot shows a calendar interface. At the top, there's a navigation bar with icons for cloud, TECOA6, Contacts, and Calendar. A search bar is at the top right. Below the navigation is a header for 'Calendar' with the date '9 April 2023–15 April 2023'. It includes buttons for back, forward, today, and new event. To the right is a monthly calendar for April 2023, showing days from Sunday to Saturday. The month starts on Friday, April 28, 2023. The weekly calendar grid shows time slots from 7am to 2pm for each day. A red horizontal bar is visible on Thursday, April 13, between 12pm and 1pm. On the right side, there are sections for 'My Calendars' (with a 'My Events' button) and 'Other Calendars'.

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 fileon/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

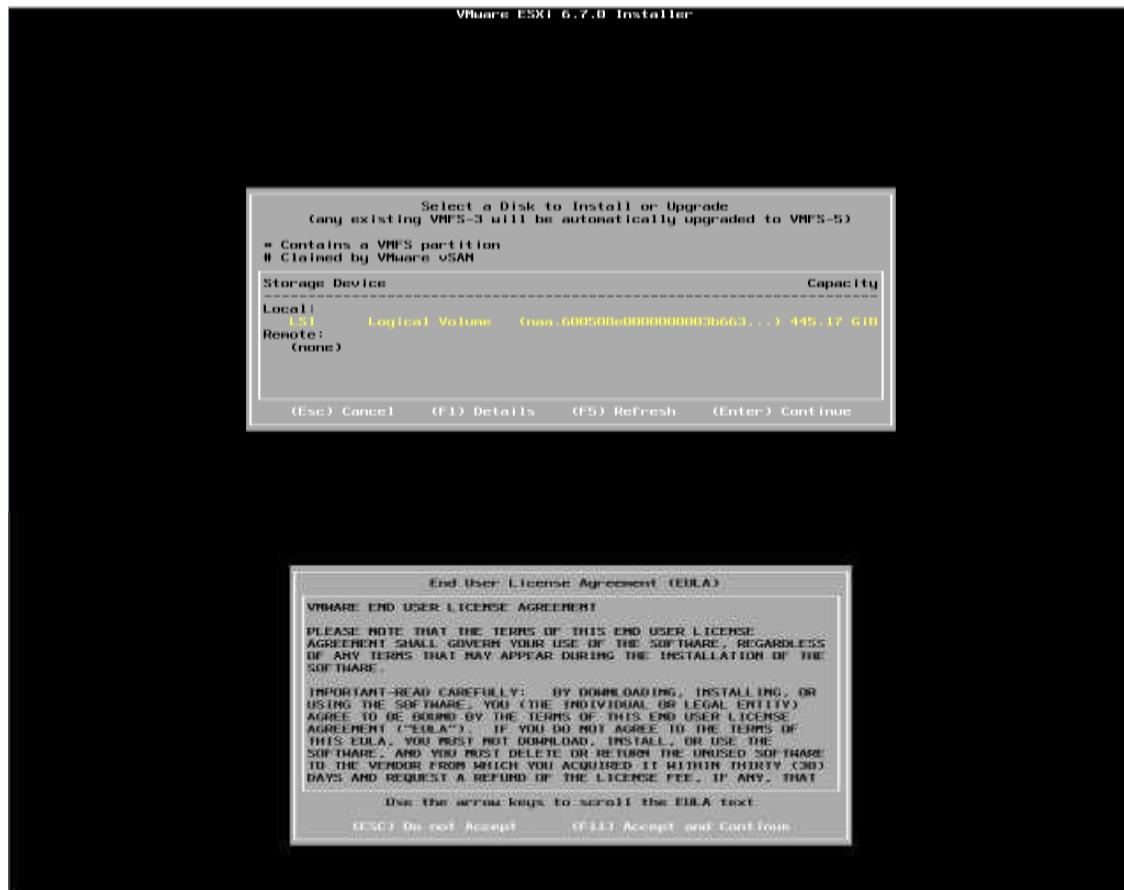
- Host/NodeESXi installation:-
- **ESXiHardwareRequirements:-**
- ESXi6.7 requires a host machine with at least two CPU cores.
- ESXi6.7 supports 64-bit x86 processors
- ESXi6.7 requires the NX/XD bit to be enabled for the CPU in the BIOS.
- ESXi6.7 requires a minimum of 4GB of physical RAM. It is recommended 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.

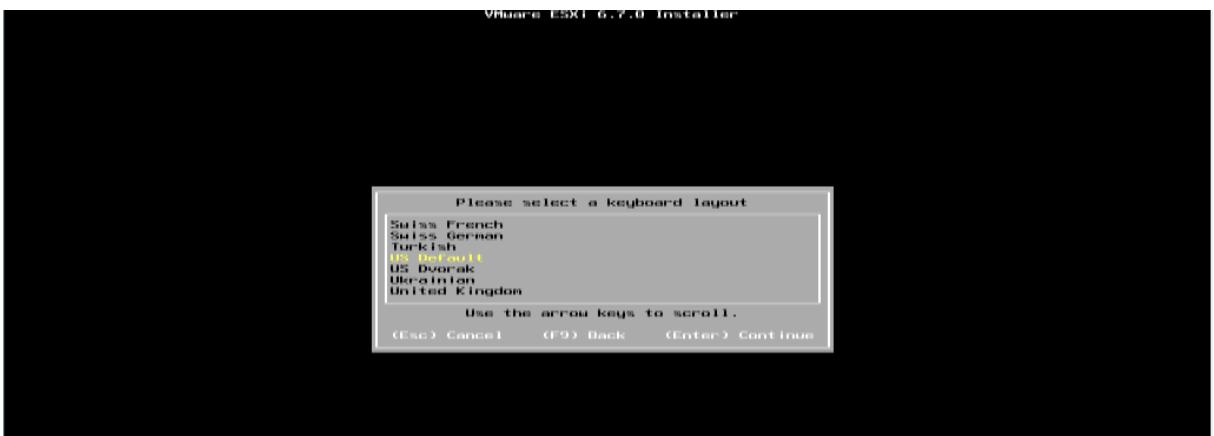
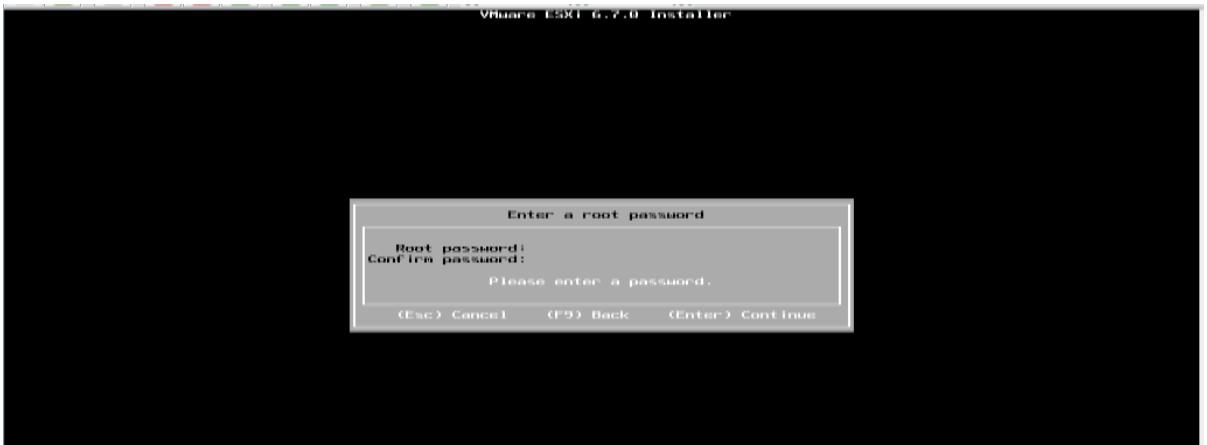
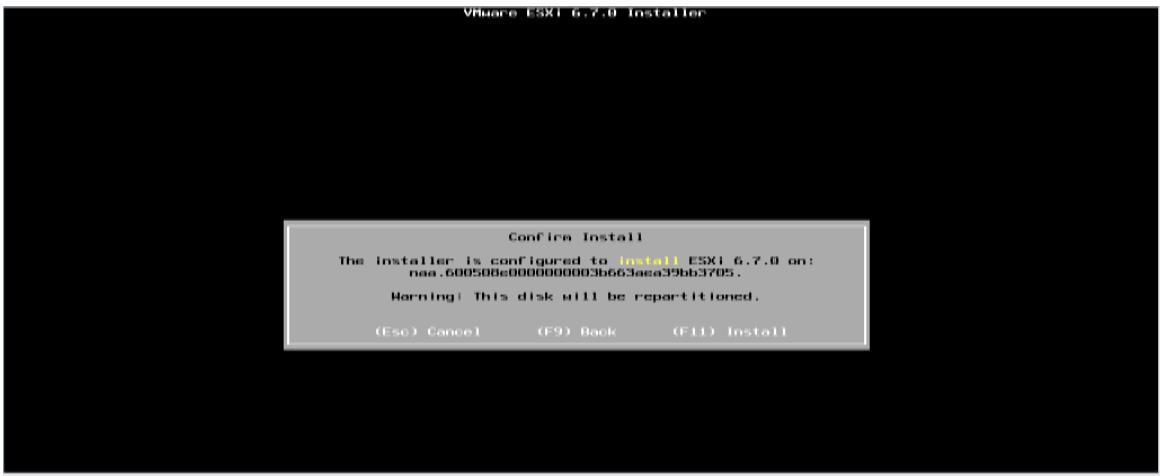
ForSerialATA(SATA), a disk connected through supported SAS controller or supported on board SATA controllers. SATA disks are considered remote not local. These disks are not used as a scratch partition by default because they are seen as remote.



ESXiInstaller:

Accept Agreement:

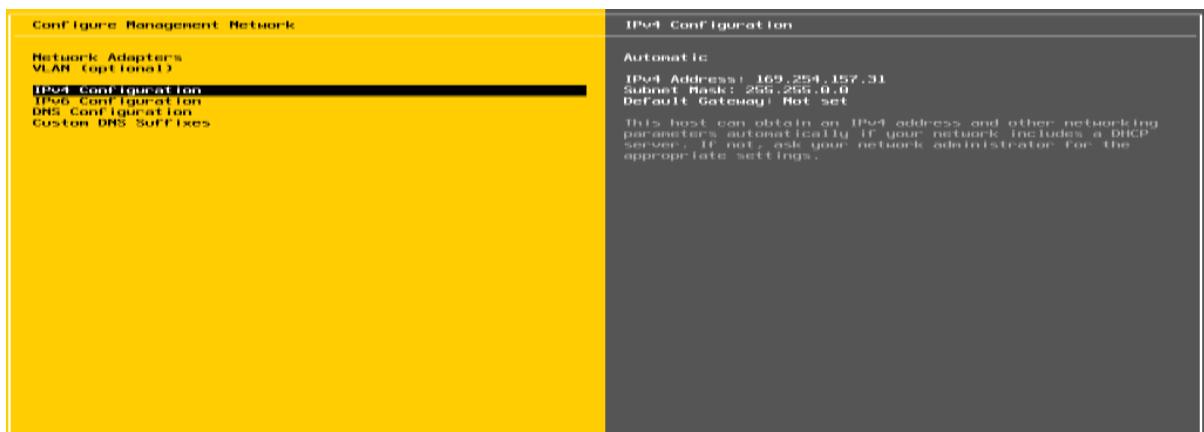


Select storage :**Select Keyboard Layout :****Set NodeESXi Root Password :**

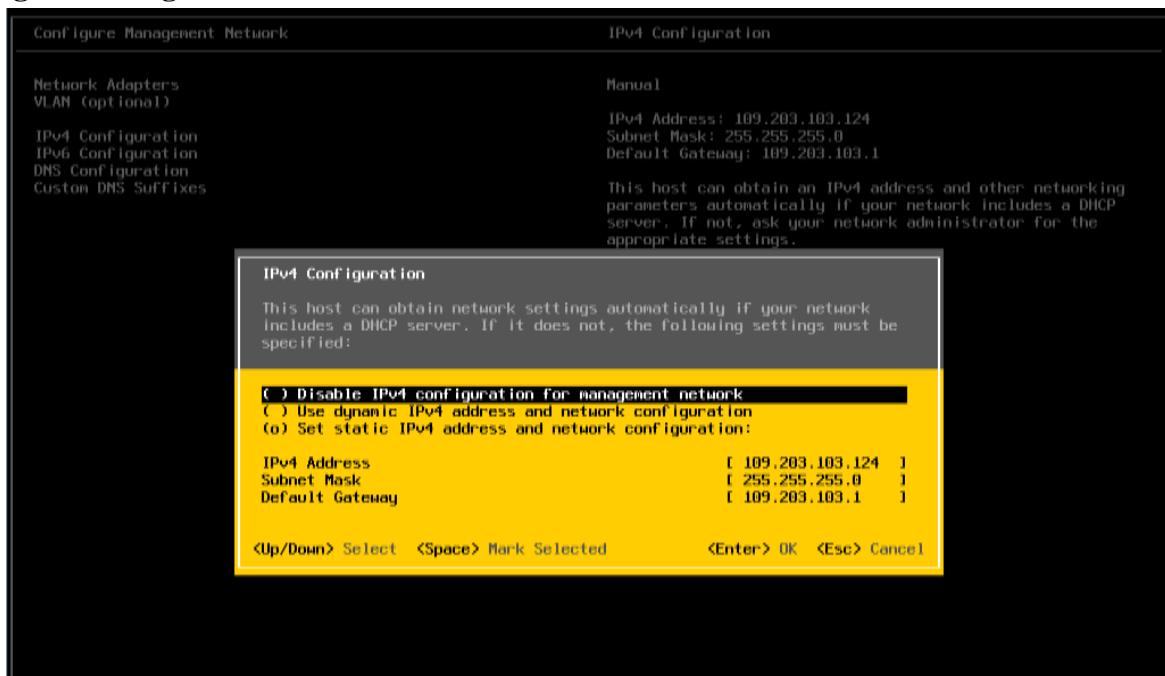
Installation complete (Reboot)CLII interface to configuration



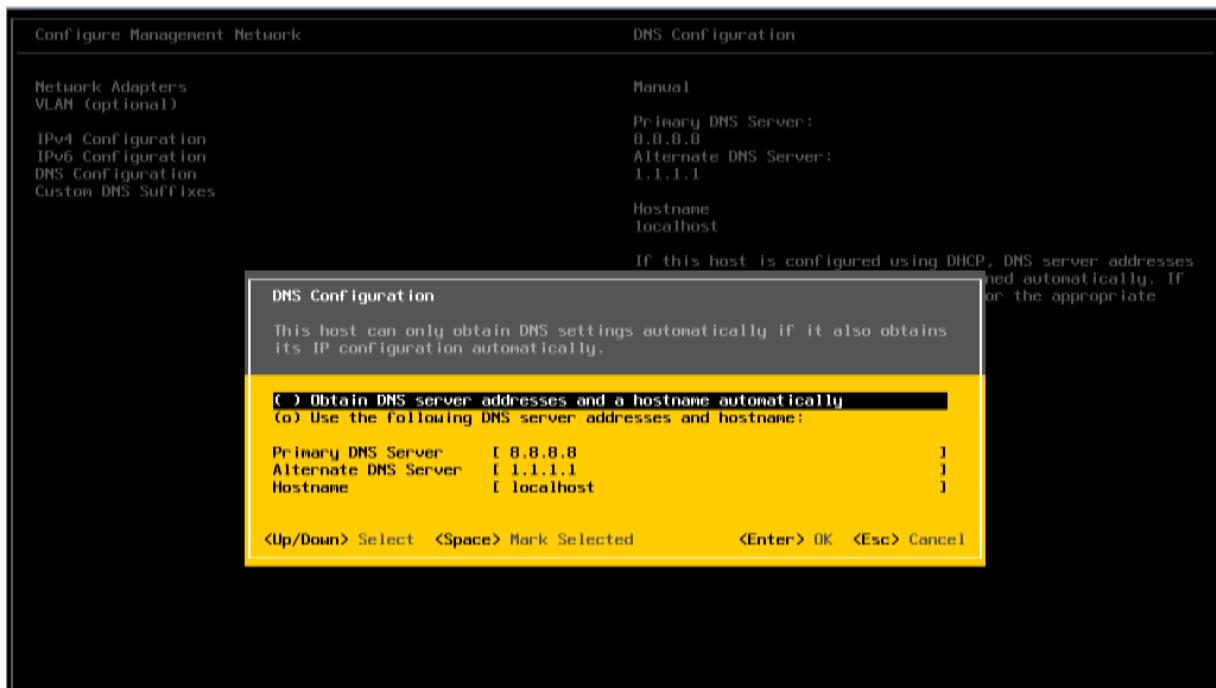
CLI Interface to Configuration:



Configure Management Network

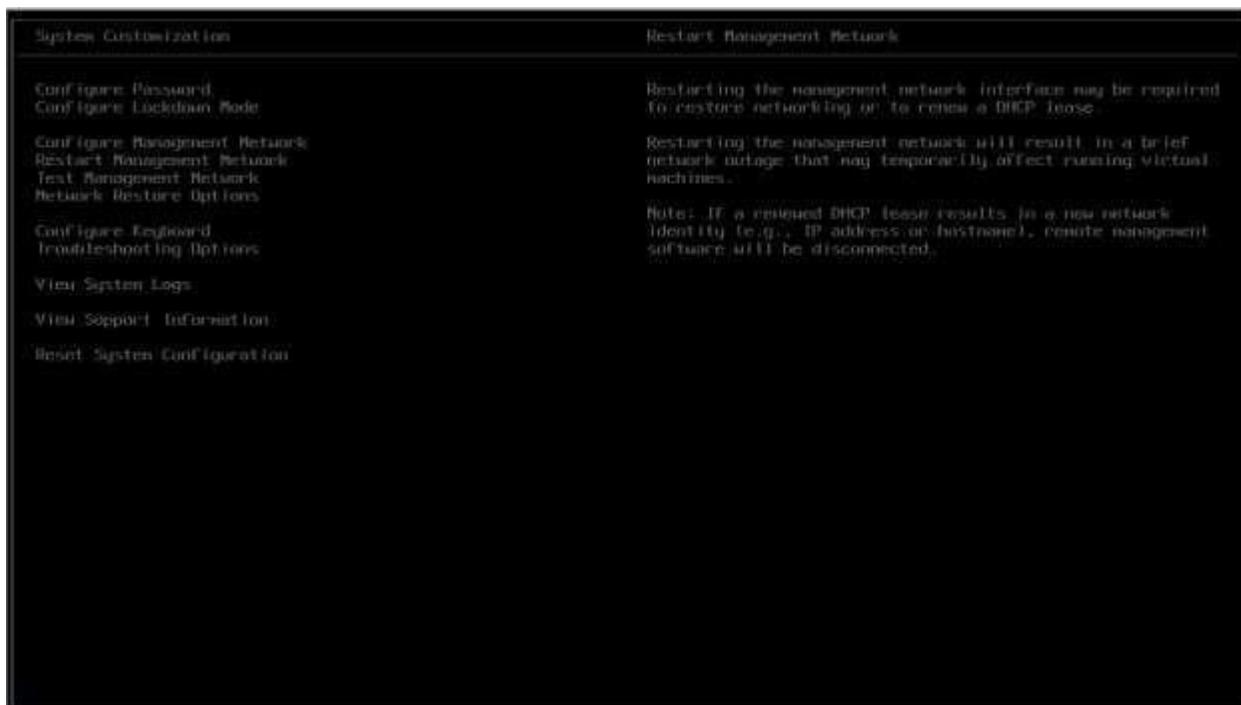


Set IPV4



Set DNServer :

Restart Management Network



GUIAccess :**ClusterSetup**

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

VCenter Access:

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

Create Datacenter:

The screenshot shows the vSphere Client interface. In the left sidebar, under 'Actions - AVOCOE', the 'New Datacenter...' option is selected. A modal window titled 'New Datacenter' is open, showing the 'Name' field set to 'AVOCOE' and the 'Location' field set to 'vsphere.avcoe.com'. Below the modal, a table of recent tasks shows two completed tasks: 'Remove datacenter' and 'Create datacenter'.

The screenshot shows the vSphere Client interface. In the left sidebar, under 'Actions - AVOCOE', the 'New Cluster...' option is selected. A modal window titled 'New Cluster | AVOCOE' is open, showing the 'Name' field set to 'AVOCOE-CLUSTER' and the 'Location' field set to 'AVOCOE'. Under 'vSphere HA', the 'Turn ON' checkbox is checked. Under 'EVC', the 'Disable' dropdown is set to 'Disabled'. Below the modal, a table of recent tasks shows three completed tasks: 'Create datacenter', 'Remove datacenter', and 'Create datacenter'.

Create cluster :

Assign cluster name :

The screenshot shows the vSphere Client interface. In the left sidebar, under 'Actions - AVOCOE', the 'New Cluster...' option is selected. A modal window titled 'New Cluster | AVOCOE' is open, showing the 'Name' field set to 'AVOCOE-CLUSTER' and the 'Location' field set to 'AVOCOE'. Under 'vSphere HA', the 'Turn ON' checkbox is checked. Under 'EVC', the 'Disable' dropdown is set to 'Disabled'. Below the modal, a table of recent tasks shows three completed tasks: 'Create datacenter', 'Remove datacenter', and 'Create datacenter'.

Add host

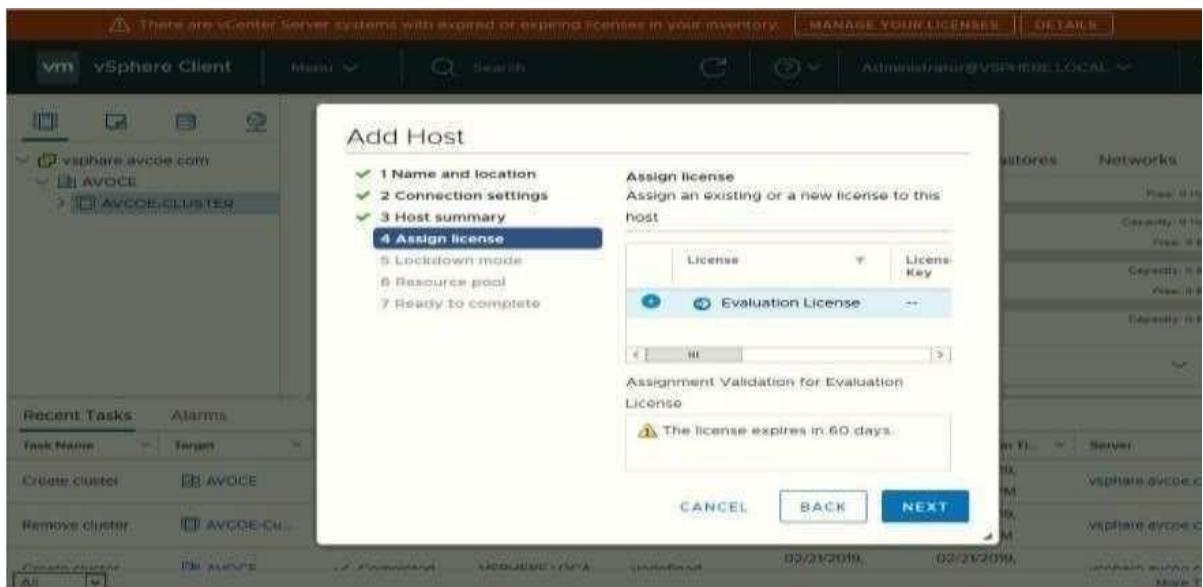
The screenshot shows the vSphere Client interface. In the center, there is a cluster summary for 'AVCOE-CLUSTER'. On the left, a context menu is open over the cluster name, listing options like 'Add Host...', 'New Virtual Machine...', 'New Resource Pool...', 'Deploy OVF Template...', 'New vApp...', 'Storage', 'Host Profiles', 'Edit Default VM Compatibility...', 'Assign License...', 'Settings', 'Rename...', 'Tags & Custom Attributes', and 'Delete'. At the bottom of this menu, there is a 'vSAN' section with a 'Configure...' button. The top of the screen has a banner stating 'There are vCenter Server systems with expired or expiring licenses in your inventory.' with buttons for 'MANAGE YOUR LICENSES' and 'DETAILS'.

Add host IP :

This screenshot shows the 'Add Host' wizard, Step 1: Name and location. The host name field contains '172.14.5.244' and the location dropdown is set to 'AVCOE-CLUSTER'. The wizard also lists steps 2 through 7: Connection settings, Host summary, Assign license, Lockdown mode, Resource pool, and Ready to complete.

Enter host cred

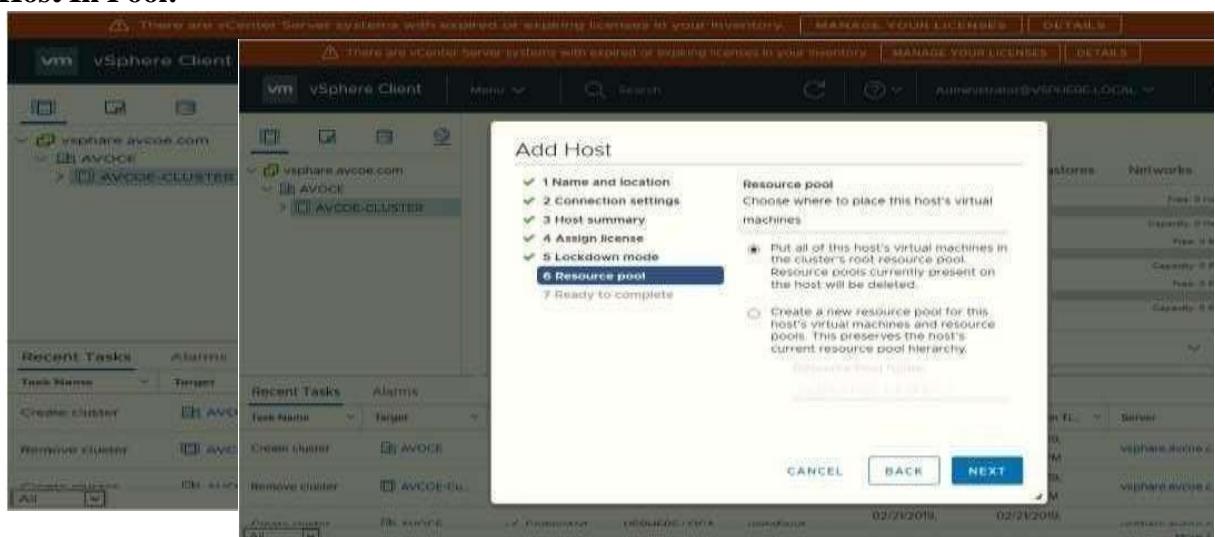
This screenshot shows the 'Add Host' wizard, Step 2: Connection settings. It displays the 'Host summary' table for the host '172.14.5.244', which includes information such as Name, Vendor, Model, Version, and Virtual Machines. The host is identified as an HP Z420 Workstation running VMware ESXi 6.7.0 build-10302608. The wizard also lists steps 1 through 7.



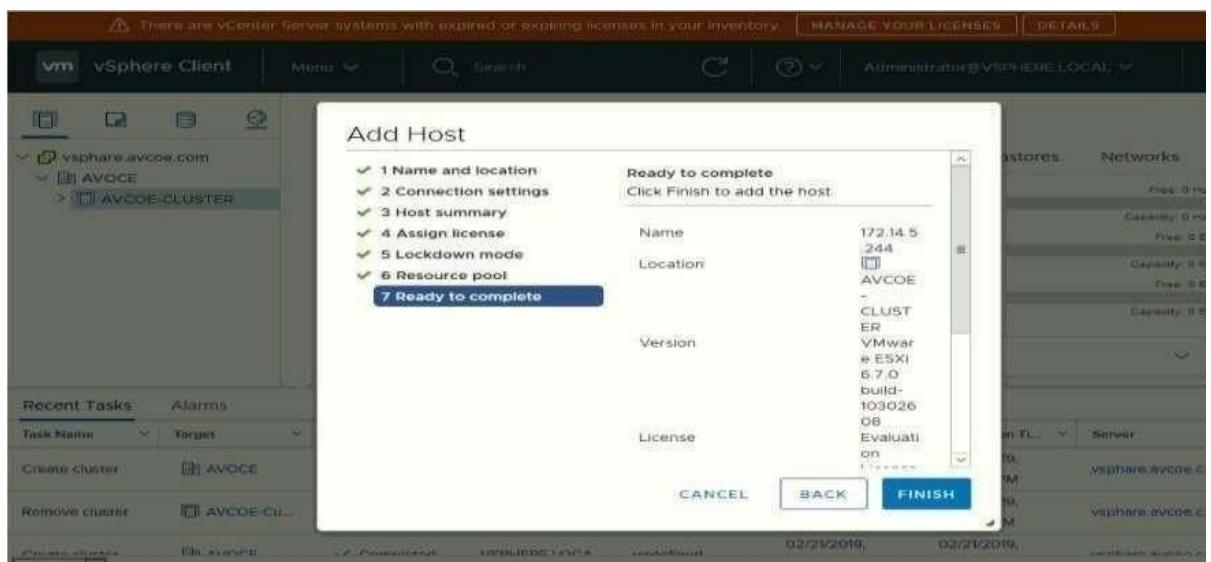
Hot summary :

Lock Down mode:

Add Host In Pool:



Finish:



Host View and View Config:

Cluster View and Configuration:

The screenshot displays two views of the vSphere Client interface. The top half shows the 'Host View' for the host '172.14.5.245'. The bottom half shows the 'Cluster View' for the cluster 'AVCOE-CLUSTER'. Both views include tabs for Summary, Monitor, Configure, Permissions, VMs, Datastores, and Networks. The 'Recent Tasks' and 'Alarms' sections are also visible.

Host View (Top):

- Summary:**
 - Hypervisor: VMware ESXi, 6.7.0, 10302608
 - Model: HP Z420 Workstation
 - Processor Type: Intel(R) Xeon(R) CPU E5-1607 v2 @ 3.00GHz
 - Logical Processors: 4
 - NICs: 1
 - Virtual Machines: 0
 - State: Connected
 - Uptime: 0 second
- Datastore:** CPU Free: 11.87 GHz Used: 0 Hz Capacity: 11.87 GHz
- Memory:** Free: 15.93 GB Used: 0 B Capacity: 15.93 GB
- Storage:** Free: 922.58 GB Used: 1.42 GB Capacity: 924.08 GB

Cluster View (Bottom):

- Summary:**
 - Total Processors: 8
 - Total vMotion Migrations: 0
- Datastore:** CPU Free: 21.39 GHz Used: 2.58 GHz Capacity: 23.94 GHz
- Memory:** Free: 29.0 GB Used: 2.96 GB Capacity: 31.85 GB
- Storage:** Free: 1.8 TB Used: 234 GB Capacity: 1.8 TB

Conclusion: Like this we have configure VSphere Private Cloud