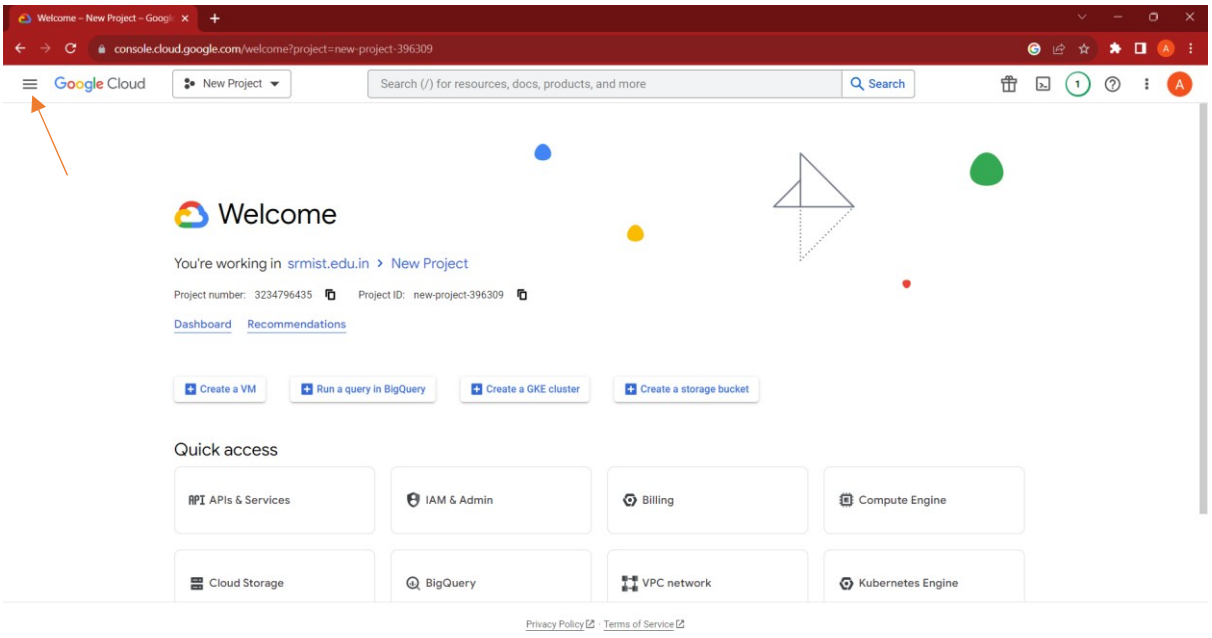
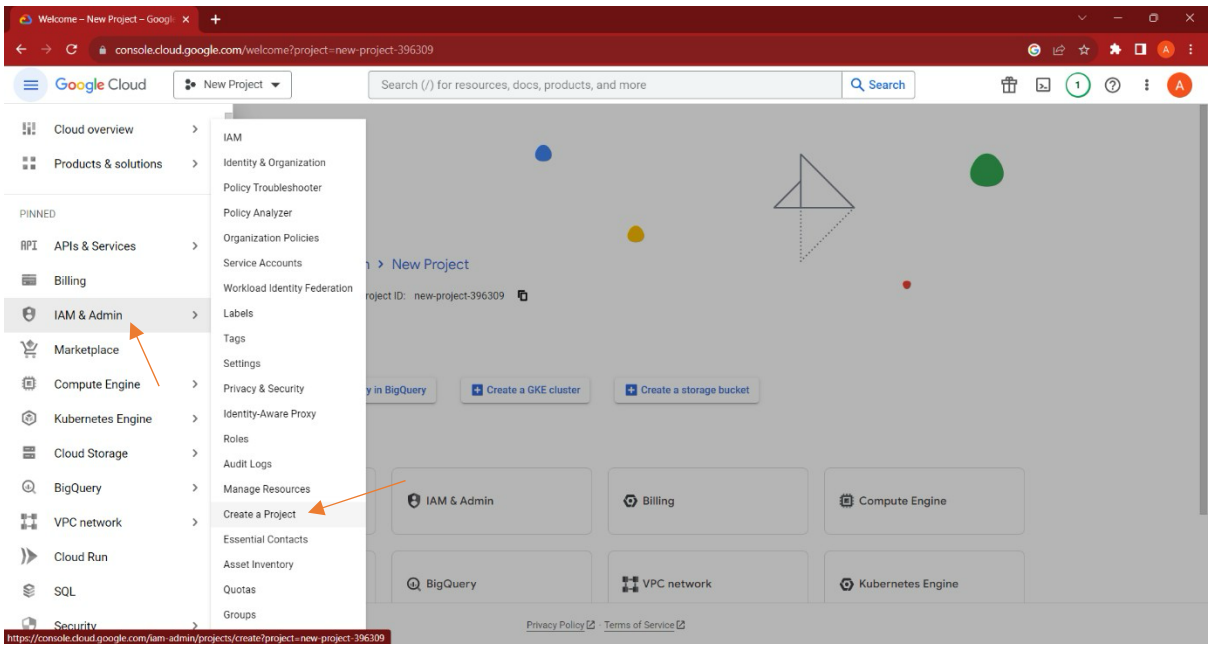
EX NO. 1
CREATE GOOGLE ACCOUNT WITH PROJECT

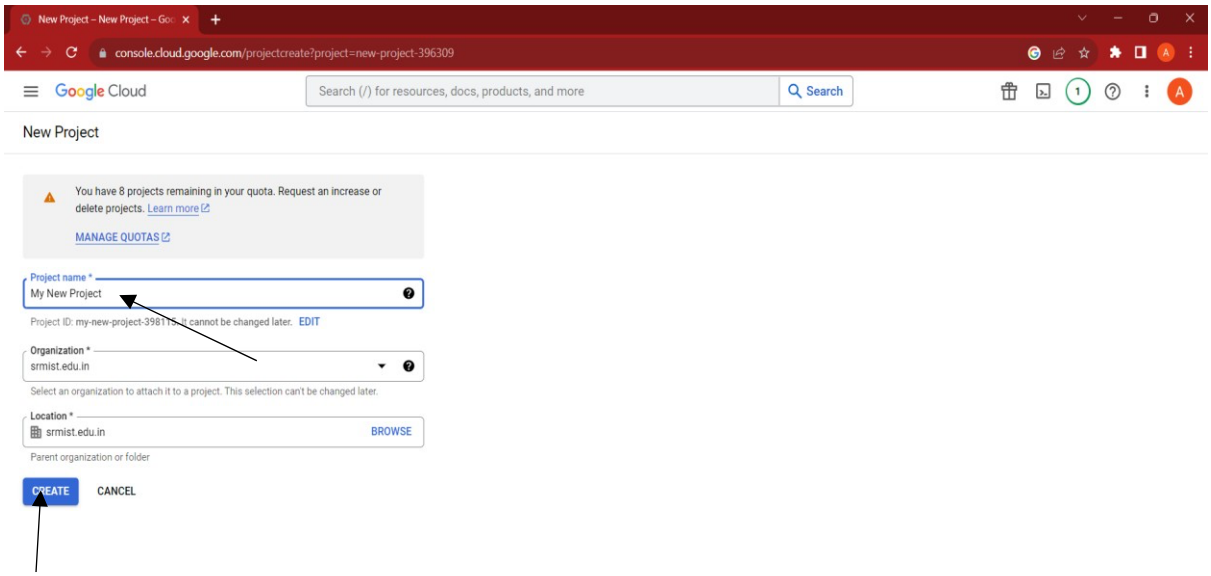
1. Browse ‘Google Console’ on Google Chrome and make an account. If you have already, an existing account then Login in your account.
2. Click on top left corner three-line toggle.



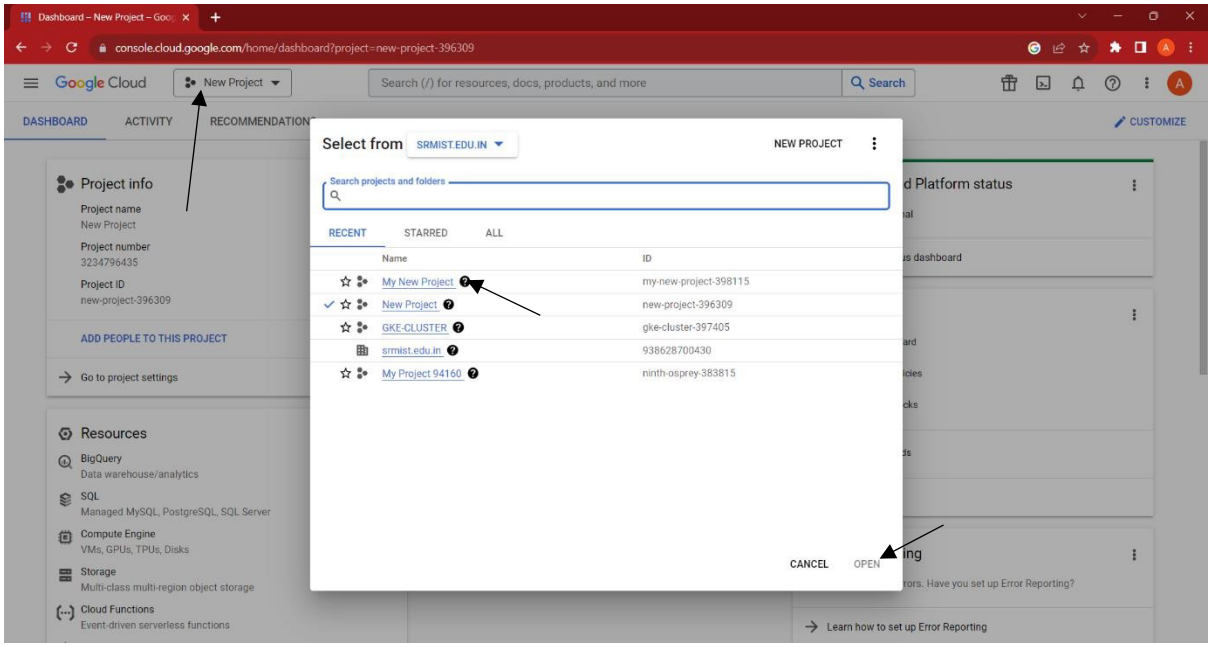
3. Click on ‘IAM & Admin’ section and select ‘Create Project’ option.



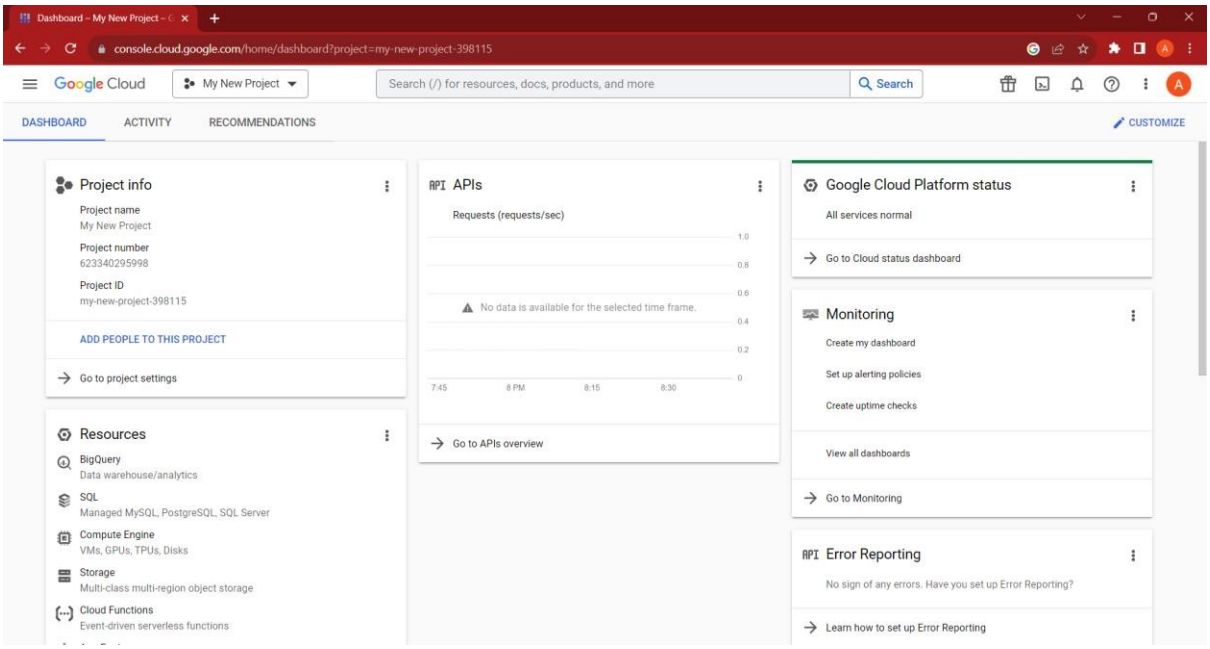
4. Mention the name of your project, organization and location. After mentioning all details click on ‘Create’ button.
5. Your Project has created.



6. Now, to check your project click on Project section on top left corner just aside of ‘Google Cloud’ logo. After selecting the project, click on open.



7. Hence, your google account and first project on Google Cloud Console is created.



EX NO. 2

PRINT THE DATA IN OPENMP PARALLEL

```
#include <stdio.h>
#include <omp.h>

int main() {
   #pragma omp parallel
   {
      int thread_id = omp_get_thread_num();
      printf("Hello from thread %d\n", thread_id);
   }

   return 0;
}
```

Output

```
Hello from thread 0
Hello from thread 1
Hello from thread 2
Hello from thread 3
```

EX NO. 3

SENDING AND RECEIVING DATA WITH MPI

```
#include <stdio.h>
#include <mpi.h>

int main(int argc, char** argv) {
   int rank, size;
   int send_data, recv_data;

   MPI_Init(&argc, &argv);
   MPI_Comm_rank(MPI_COMM_WORLD, &rank);
   MPI_Comm_size(MPI_COMM_WORLD, &size);

   if (size != 2) {
      printf("This example requires exactly 2 MPI processes.\n");
      MPI_Finalize();
      return 1;
   }

   if (rank == 0) {
      // Process 0 sends data to process 1
      send_data = 42;
      MPI_Send(&send_data, 1, MPI_INT, 1, 0, MPI_COMM_WORLD);
      printf("Process %d sent data: %d\n", rank, send_data);
   } else if (rank == 1) {
      // Process 1 receives data from process 0
      MPI_Recv(&recv_data, 1, MPI_INT, 0, 0, MPI_COMM_WORLD,
MPI_STATUS_IGNORE);
      printf("Process %d received data: %d\n", rank, recv_data);
   }

   MPI_Finalize();
   return 0;
}
```

Compilation Command

```
mpicc -o mpi_send_recv mpi_send_recv.c
mpirun -n 2 ./mpi_send_recv
```

Output

```
Process 0 sent data: 42
Process 1 received data: 42
```

EX NO. 4

CALCULATE MULTIPLICATIVE INVERSE MATRICES

```
import numpy as np

x = np.array([[1,2],[3,4]])
y = np.linalg.inv(x)

print(x,'\n')
print(y,'\n')
print(np.dot(x,y))
```

Output:

```
In [2]: import numpy as np

        x = np.array([[1,2],[3,4]])
        y = np.linalg.inv(x)

        print(x,'\n')
        print(y,'\n')
        print(np.dot(x,y))
```

```
[[1 2]
 [3 4]]

[[-2.   1. ]
 [ 1.5 -0.5]]

[[1.0000000e+00 0.0000000e+00]
 [8.8817842e-16 1.0000000e+00]]
```

EX NO. 5

CONVERT LIST TO ARRAY IN PYTHON

```
import numpy as np
x = [1,2,3,4]
a = np.asarray(x)

print(a)
print(type(x))
print(type(a))
```

Output:

```
In [11]: import numpy as np
         x = [1,2,3,4]
         a = np.asarray(x)

         print(a)
         print(type(x))
         print(type(a))

         [1 2 3 4]
         <class 'list'>
         <class 'numpy.ndarray'>
```

EX NO. 6

NDARRAY FROM LIST TO TUPLES

```
import numpy as np

x = [(1,2,3),(4,5)]
a = np.asarray(x)

print(x,type(x))
print(a,type(a))
```

Output:

```
In [15]: import numpy as np

         x = [(1,2,3),(4,5)]
         a = np.asarray(x)

         print(x,type(x))
         print(a,type(a))

         [(1, 2, 3), (4, 5)] <class 'list'>
         [(1, 2, 3) (4, 5)] <class 'numpy.ndarray'>
```

EX NO. 7

PRINT THE "HELLO WORLD" USING PYTHON

```
"""PRINT THE
"HELLO WORLD"
USING PYTHON """

print("Hello World!")
```

Output:

EX NO. 8

FIND THE DATA TYPE VALUE WITH ARRAY

```
import numpy as np

stu_class = np.dtype([('name','S20'),('age','i1'),('marks','f4')])

print(stu_class)
```

Output:

```
In [4]: import numpy as np

        stu_class = np.dtype([('name','S20'),('age','i1'),('marks','f4')])

        print(stu_class)

        [('name', 'S20'), ('age', 'i1'), ('marks', '<f4')]
```

EX NO. 9

FIND THE VALUE OF MATH LIBRARY

```
import numpy.matlib
import numpy as np

x = np.matlib.eye(n=3, M=4, k=0, dtype=float)

print(x)
```

Output:

```
In [19]: import numpy.matlib
         import numpy as np

         x = np.matlib.eye(n=3, M=4, k=0, dtype=float)

         print(x)

         [[1. 0. 0. 0.]
          [0. 1. 0. 0.]
          [0. 0. 1. 0.]]
```

EX NO. 10

FIND THE VALUE OF MATH LIBRARY MATRIX

```
import numpy.matlib
import numpy as np

j = np.array([[1,2,3],[4,5,6],[7,8,9]])

x = np.asmatrix(j)

print(j,type(j),'\n')
print(x,type(x))
```

Output:

```
In [21]: import numpy.matlib
         import numpy as np

         j = np.array([[1,2,3],[4,5,6],[7,8,9]])

         x = np.asmatrix(j)

         print(j,type(j),'\n')
         print(x,type(x))
```

```
[[1 2 3]
 [4 5 6]
 [7 8 9]] <class 'numpy.ndarray'>

[[1 2 3]
 [4 5 6]
 [7 8 9]] <class 'numpy.matrix'>
```

EX NO. 11

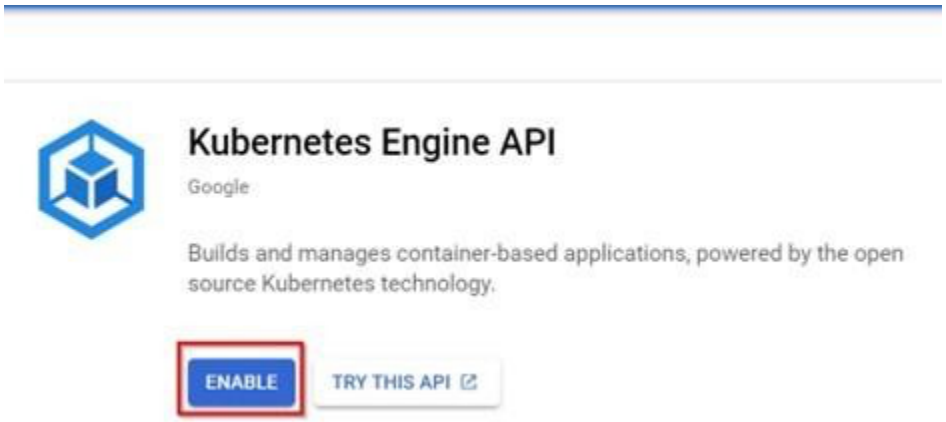
CREATE GKE CLUSTER USING CONSOLE

Setting Up Environment:

1. Navigate to https://console.cloud.google.com/ and sign in with your Google Account Credentials.
2. Select your existing project or if you do not have one then click on create a new project.

3. After selecting the project, in the Cloud Console, navigate to the **Navigation menu** > **Kubernetes Engine**.
Note: Make sure you have selected the correct project.

4. Click on **Enable** to enable the API.



Steps to Create GKE Cluster Using Console:

Steps to Create Autopilot Cluster:

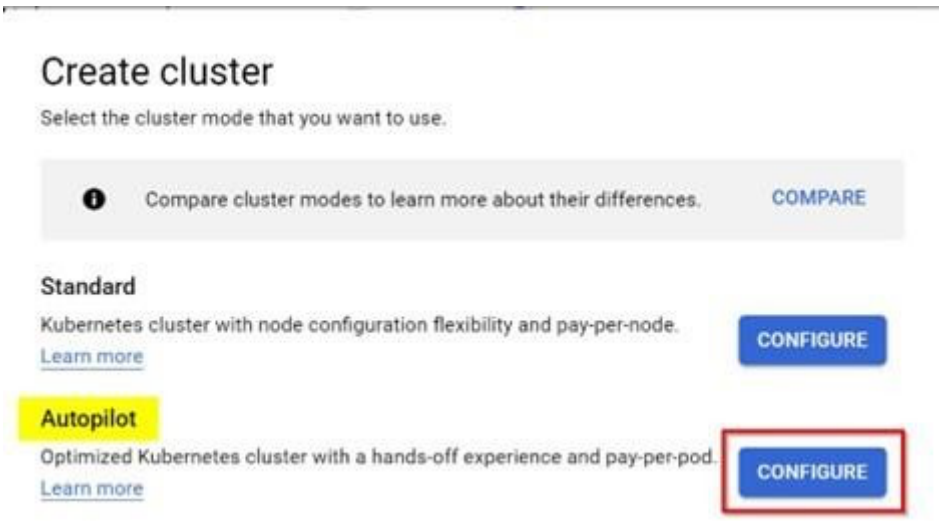
1. Log in to the Google Cloud Console (console.cloud.google.com).

2. Select your project from the drop-down menu at the top of the screen.

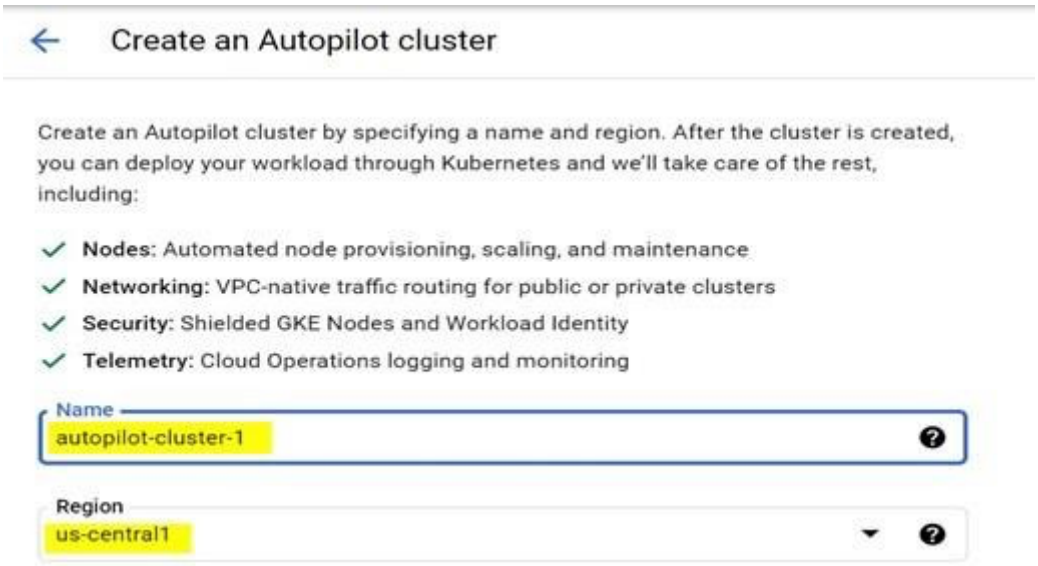
3. Click on the Navigation menu (☰) in the upper-left corner and select “Kubernetes Engine” under “Compute”.

4. Click “Create cluster” to start the cluster creation process.

5. Under the “Cluster mode” section, select “Autopilot”.



6. Configure the cluster settings, such as the cluster name, location, and node pool size.



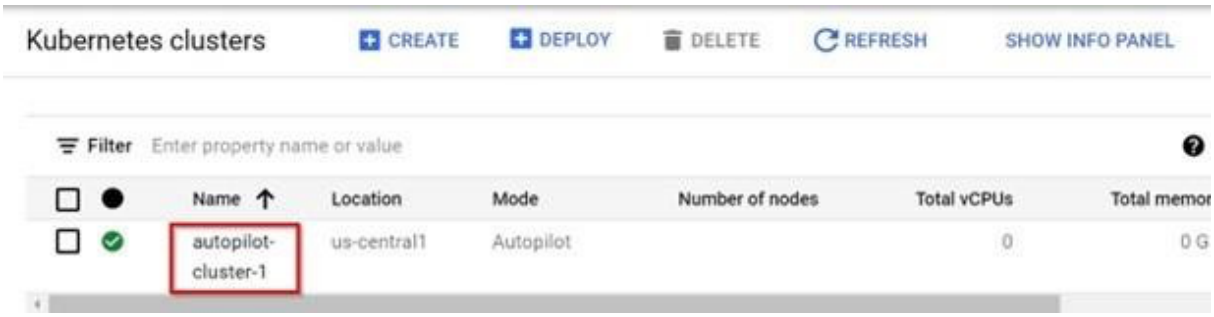
7. Choose the desired Kubernetes version for the cluster.

8. Customize additional settings like node configuration, networking, and security options as required.

Networking defines how applications in the cluster communicate with each other and how clients can reach them.

Note: By default, you can configure access from public networks to your cluster's workloads. Routes are not created automatically. Private clusters assign internal addresses to Pods and nodes, and workloads are completely isolated from public networks.

9. Leave the default settings for Advanced Options as that is an optional field. Click "**Create**" to start the cluster creation process



After a few minutes, your Autopilot GKE cluster will be ready for use. Autopilot manages the cluster resources for you, so you don't need to worry about node sizing, scaling, or upgrades. You can then deploy your applications to the cluster using Kubernetes commands or using the Google Cloud Console.

EX NO. 12

CONVERT LIST TO NDARRAY

```
import numpy as np

lst = [[1,2,4],[4,5,6],[6,7,8]]

ndarray = np.asarray(lst)

print(lst,type(lst),'\n')
print(ndarray,type(ndarray))
```

Output:

```
In [2]: import numpy as np

        lst = [[1,2,4],[4,5,6],[6,7,8]]

        ndarray = np.asarray(lst)

        print(lst,type(lst),'\n')
        print(ndarray,type(ndarray))

        [[1, 2, 4], [4, 5, 6], [6, 7, 8]] <class 'list'>

        [[1 2 4]
         [4 5 6]
         [6 7 8]] <class 'numpy.ndarray'>
```

EX NO. 13

CONVERT ARRAY TO LIST

```
import numpy as np

arr = np.array([[1,2,3],[4,5,6]])

con_lst = list(arr)

print(arr,type(arr),'\n')
print(con_lst,type(con_lst))
```

Output:

```
In [3]: import numpy as np

        arr = np.array([[1,2,3],[4,5,6]])

        con_lst = list(arr)

        print(arr,type(arr),'\n')
        print(con_lst,type(con_lst))
```

```
[[1 2 3]
 [4 5 6]] <class 'numpy.ndarray'>

[array([1, 2, 3]), array([4, 5, 6])] <class 'list'>
```

EX NO. 14

ARRAY LIST TO NUMPY

import numpy as np

lst = [1,2,3,4,5]

arr = np.array(lst)

print(arr,type(arr))

Output:

```
In [4]: import numpy as np

        lst = [1,2,3,4,5]

        arr = np.array(lst)

        print(arr,type(arr))
```

```
[1 2 3 4 5] <class 'numpy.ndarray'>
```

EX NO. 15

APPLY ARITHMETIC OPERATORS NUMPY FUNCTIONS ON TWO ARRAYS

```
import numpy as np

x = np.array([1,2,3,4,5])
y = np.array([6,7,8,9,10])

print('\n Addition:')
print(np.add(x,y))

print('\n Substraction:')
print(np.subtract(x,y))

print('\n Multiplication:')
print(np.multiply(x,y))

print('\n Division:')
print(np.divide(x,y))

print('\n Exponential:')
print(np.power(x,y))
```

Output:

```
In [6]: import numpy as np

        x = np.array([1,2,3,4,5])
        y = np.array([6,7,8,9,10])

        print('\n Addition:')
        print(np.add(x,y))

        print('\n Substraction:')
        print(np.subtract(x,y))

        print('\n Multiplication:')
        print(np.multiply(x,y))

        print('\n Division:')
        print(np.divide(x,y))

        print('\n Exponential:')
        print(np.power(x,y))
```

```
 Addition:
[ 7  9 11 13 15]

 Substraction:
[-5 -5 -5 -5 -5]

 Multiplication:
[ 6 14 24 36 50]

 Division:
[0.16666667 0.28571429 0.375      0.44444444 0.5       ]

 Exponential:
[      1     128    6561  262144 9765625]
```

EX NO. 16

FIND THE SQUARE ROOT VALUE

```
import math

a = int(input('Value of a:'))
b = int(input('Value of b:'))
c = int(input('Value of c:'))

print('Square root of a is',math.sqrt(a))
print('Square root of b is',math.sqrt(b))
print('Square root of c is',math.sqrt(c))
```

Output:

```
In [3]: import math

        a = int(input('Value of a:'))
        b = int(input('Value of b:'))
        c = int(input('Value of c:'))

        print('Square root of a is',math.sqrt(a))
        print('Square root of b is',math.sqrt(b))
        print('Square root of c is',math.sqrt(c))

        Value of a:25
        Value of b:36
        Value of c:81
        Square root of a is 5.0
        Square root of b is 6.0
        Square root of c is 9.0
```

EX NO. 17

FIND THE AREA OF A CIRCLE

```
from math import pi

r = float(input('Enter the value of radius:'))

area = pi * r**2

print('Area of the circle is',area)
```

Output:

```
In [4]: from math import pi

        r = float(input('Enter the value of radius:'))

        area = pi * r**2

        print('Area of the circle is',area)

        Enter the value of radius:2.3
        Area of the circle is 16.619025137490002
```

EX NO. 18

FIND THE QUADRATIC EQUATION.

```
import cmath
a = int(input('Enter the value of a:'))
b = int(input('Enter the value of b:'))
c = int(input('Enter the value of c:'))

d = (b**2 - 4*a*c)

sol1 = (-b - cmath.sqrt(d))/(2*a)
sol2 = (-b + cmath.sqrt(d))/(2*a)

print('Two Solutions of quadratic equation is {0} and {1}'.format(sol1,sol2))
```

Output:

```
In [5]: import cmath
        a = int(input('Enter the value of a:'))
        b = int(input('Enter the value of b:'))
        c = int(input('Enter the value of c:'))

        d = (b**2 - 4*a*c)

        sol1 = (-b - cmath.sqrt(d))/(2*a)
        sol2 = (-b + cmath.sqrt(d))/(2*a)

        print('Two Solutions of quadratic equation is {0} and {1}'.format(sol1,sol2))

        Enter the value of a:3
        Enter the value of b:4
        Enter the value of c:5
        Two Solutions of quadratic equation is (-0.6666666666666666-1.1055415967851332j) and (-0.6666666666666666+1.1055415967851332j)
```

EX NO. 19

PRINT THE VALUE OF SWAP TWO VARIABLES.

```
x = int(input('Value of x after swaping: '))
y = int(input('Value of y after swaping: '))

temp = x
x = y
y = temp

print('Value of x after swaping:',x)
print('Value of y after swaping:',y)
```

Output:

```
In [8]: x = int(input('Value of x after swaping: '))
        y = int(input('Value of y after swaping: '))

        temp = x
        x = y
        y = temp

        print('Value of x after swaping:',x)
        print('Value of y after swaping:',y)

        Value of x after swaping: 4
        Value of y after swaping: 3
        Value of x after swaping: 3
        Value of y after swaping: 4
```

EX NO. 20

PRINT ALL OPERATORS IN PYTHON

```
# 1 Addition
x = float(input('Enter value of x: '))
y = float(input('Enter value of y: '))
add = x+y
print('Addition of {0} and {1} is {2}'.format(x,y,add))
```

Output:

```
In [9]: # Addition
        x = float(input('Enter value of x: '))
        y = float(input('Enter value of y: '))
        add = x+y
        print('Addition of {0} and {1} is {2}'.format(x,y,add))

        Enter value of x: 4
        Enter value of y: 5
        Addition of 4.0 and 5.0 is 9.0
```

```
# 2 Substraction
x = float(input('Enter value of x: '))
y = float(input('Enter value of y: '))
subs = x-y
print('Addition of {0} and {1} is {2}'.format(x,y,subs))
```

Output:

```
In [10]: # 2 Substraction
         x = float(input('Enter value of x: '))
         y = float(input('Enter value of y: '))
         subs = x-y
         print('Addition of {0} and {1} is {2}'.format(x,y,subs))

         Enter value of x: 4
         Enter value of y: 3
         Addition of 4.0 and 3.0 is 1.0
```

3 Multiplication

```
x = float(input('Enter value of x: '))
y = float(input('Enter value of y: '))
multi = x*y
print('Addition of {0} and {1} is {2}'.format(x,y,multi))
```

Output:

```
In [11]: # 3 Multiplication
         x = float(input('Enter value of x: '))
         y = float(input('Enter value of y: '))
         multi = x*y
         print('Addition of {0} and {1} is {2}'.format(x,y,multi))

         Enter value of x: 3
         Enter value of y: 4
         Addition of 3.0 and 4.0 is 12.0
```

```
# 4 Division
x = float(input('Enter value of x: '))
y = float(input('Enter value of y: '))
divide = x/y
print('Addition of {0} and {1} is {2}'.format(x,y,divide))
```

Output:

```
In [12]: # 4 Division
         x = float(input('Enter value of x: '))
         y = float(input('Enter value of y: '))
         divide = x/y
         print('Addition of {0} and {1} is {2}'.format(x,y,divide))

         Enter value of x: 6
         Enter value of y: 3
         Addition of 6.0 and 3.0 is 2.0
```

```
# 5 Modulus
x = float(input('Enter value of x: '))
y = float(input('Enter value of y: '))
modulus = x%y
print('Addition of {0} and {1} is {2}'.format(x,y,modulus))
```

Output:

```
In [13]: # 5 Modulus
         x = float(input('Enter value of x: '))
         y = float(input('Enter value of y: '))
         modulus = x%y
         print('Addition of {0} and {1} is {2}'.format(x,y,modulus))

         Enter value of x: 5
         Enter value of y: 3
         Addition of 5.0 and 3.0 is 2.0
```

# 6 Floor Division
x = float(input('Enter value of x: '))
y = float(input('Enter value of y: '))
floor_div = x//y
print('Addition of {0} and {1} is {2}'.format(x,y,floor_div))

Output:

```
In [14]: # 6 Floor Division
         x = float(input('Enter value of x: '))
         y = float(input('Enter value of y: '))
         floor_div = x//y
         print('Addition of {0} and {1} is {2}'.format(x,y,floor_div))

         Enter value of x: 6
         Enter value of y: 4
         Addition of 6.0 and 4.0 is 1.0
```

# 7 Exponential
x = float(input('Enter value of x: '))
y = float(input('Enter value of y: '))
expo = x**y
print('Addition of {0} and {1} is {2}'.format(x,y,expo))

Output:

```
In [15]: # 7 Exponential
         x = float(input('Enter value of x: '))
         y = float(input('Enter value of y: '))
         expo = x**y
         print('Addition of {0} and {1} is {2}'.format(x,y,expo))

         Enter value of x: 3
         Enter value of y: 2
         Addition of 3.0 and 2.0 is 9.0
```

EX NO. 21

PRINT THE VALUE DIVIDE IN TWO NUMBERS.

```
# divide two numbers and print the output
print(5/4)
print(7/10)
print(-16/2)
print(0/100)

# with user input
x = int(input('Enter the value of x: '))
y = int(input('Enter the value of y: '))
print(x/y)
```

Output:

```
In [18]: print(5/4)
         print(7/10)
         print(-16/2)
         print(0/100)

         x = int(input('Enter the value of x: '))
         y = int(input('Enter the value of y: '))
         print(x/y)

         1.25
         0.7
         -8.0
         0.0
         Enter the value of x: 678
         Enter the value of y: 34
         19.941176470588236
```

EX NO. 22

GENERATE RANDOM NUMBERS BETWEEN 0 AND 100

```
import random

my_list = []

for i in range(0,100):
   my_list.append(random.randint(0,10))
print(my_list)
```

Output:

```
In [19]: import random

         my_list = []

         for i in range(0,100):
             my_list.append(random.randint(0,10))
         print(my_list)

         [3, 5, 7, 10, 8, 1, 6, 6, 6, 6, 7, 4, 8, 7, 9, 4, 10, 3, 1, 3, 7, 5, 10, 6, 1, 9, 3, 8, 3, 2, 0, 3, 6, 10, 7, 8, 1, 2, 10, 2,
         7, 3, 1, 9, 1, 10, 1, 4, 1, 6, 2, 0, 1, 2, 2, 10, 4, 10, 1, 7, 1, 8, 7, 7, 0, 0, 3, 9, 2, 5, 2, 4, 5, 7, 3, 9, 3, 7, 3, 0, 5,
         3, 0, 9, 0, 8, 3, 8, 0, 3, 8, 6, 3, 8, 0, 4, 2, 0, 2, 6]
```

EX NO. 23

CONVERT IN PYTHON KILOMETRES TO MILES.

```
# take user input
kilo = float(input("Enter the distance in kilometers: "))

#conversion factor
d = 0.621317

# Calculate miles
miles = kilo*d

# print final output which convert kilo into miles
print('{0} kilometers is equal to {1} miles'.format(kilo,miles))
```

Output:

```
In [20]: kilo = float(input("Enter the distance in kilometers: "))

         #conversion factor
         d = 0.621317

         # Calculate miles
         miles = kilo*d

         print('{0} kilometers is equal to {1} miles'.format(kilo,miles))

         Enter the distance in kilometers: 23
         23.0 kilometers is equal to 14.290291 miles
```

EX NO. 24

PRINT THE SAME STRING WITH DIFFERENT VARIABLES NAME

my_var = 'Hello World'
MY_VAR = 'Hello World'
my_VAR = 'Hello World'
MY_var = 'Hello World'
_my_var_ = 'Hello World'
_MY_VAR_ = 'Hello World'
My_VaR = 'Hello World'

Output:

```
In [2]: print(my_var)
        print(MY_VAR)
        print(my_VAR)
        print(MY_var)
        print(_my_var_)
        print(_MY_VAR_)
        print(My_VaR)

        Hello World
        Hello World
        Hello World
        Hello World
        Hello World
        Hello World
        Hello World
```

EX NO. 25

FIND OUT THE GREATER AND SMALLER VALUE USING PYTHON CONDITIONAL STATEMENT

```
x = float(input('Enter the value of x:'))
y = float(input('Enter the value of y:'))
if x > y:
print(x,'is greater than',y)
elif x == y:
print(x,'is equal to',y)
else:
print(x,'is smaller than',y)
```

Output:

```
Enter the value of x:45
Enter the value of y:23
45.0 is greater than 23.0
```