Practical No.1

Remote Process Communication

A] Aim: Develop a program for a multi-client chat server.

Concept: Develop a multi-client chat server application where clients chat with each other concurrently. The messages sent by clients are communicated to the server and then the server, on behalf of the source client,communicates the messages to the appropriate destination client.

Source Code:

ChatClient.java //1st file

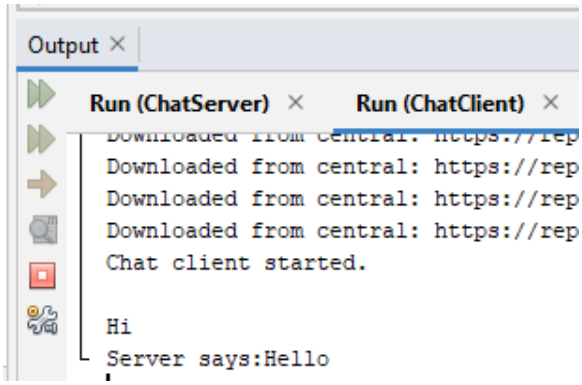
```
package com.mycompany.chatclient;
import java.io.*;
import java.net.*;
public class ChatClient {
  Socket soc;
  BufferedReader br,br1;
  PrintWriter out;
  String str;
  public ChatClient()
  {
 try{
soc=new
Socket(InetAddress.getLocalHost(),9999);
br=new BufferedReader(new
InputStreamReader(System.in));
 out=new
PrintWriter(soc.getOutputStream(),true);
System.out.println("Chat client started.");
while(true){
                str=br.readLine();
                out.println(str);
                new ChatServer();
        }  }
    catch(Exception e){
    }
  }
  class ChatServer extends Thread
  {
    String str1;
    ChatServer()
    {
      try{
br1=new BufferedReader(new
InputStreamReader(soc.getInputStream()));
start();
       }
       catch(Exception e)
       {}
    }
    public void run(){
       try{
        str1=br1.readLine();
  System.out.println("Server says:"+str1);
       }
       catch(Exception e){}
    }
  }
  public static void main(String[] args)
  {
    new ChatClient();
  }}
```

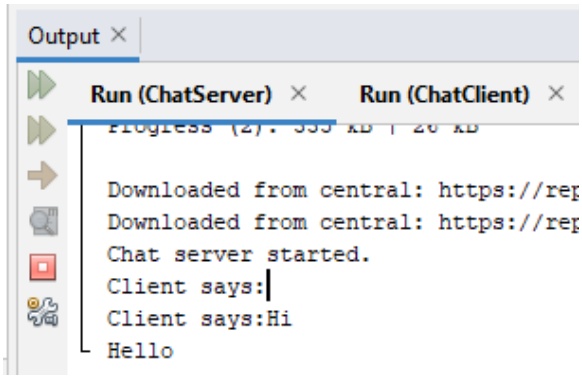
ChatServer.java //2nd file

```
package com.mycompany.chatclient;
import java.io.*;
import java.net.*;
public class ChatServer extends Thread{
   ServerSocket ss;
   Socket soc;
   BufferedReader br,br1;
   PrintWriter out;
   String str;
   public ChatServer()
   {
   try{
   ss=new ServerSocket(9999);
   soc=ss.accept();
   br=new BufferedReader(new
InputStreamReader(soc.getInputStream()));
```

```
out=new
PrintWriter(soc.getOutputStream(),true);
System.out.println("Chat server started.");
          start();
          new ChatServer1();
    }
    catch(Exception e)
    {}    }
    public void run(){
       try{
    while(true)
     {
        str=br.readLine();
        System.out.println("Client says:"+str);
           }       }
        catch(Exception e)
        {}    }
   class ChatServer1{
       String str1;
       ChatServer1()
       {
        try{
br1=new BufferedReader(new
InputStreamReader(System.in));
out=new
PrintWriter(soc.getOutputStream(),true);
while(true)
       {
              str1=br1.readLine();
              out.println(str1);
       }            }
              catch(Exception e)
              {}}}
        public static void main(String[] args)
        {
           new ChatServer();
        }}
```

OUTPUT:





B] Aim: Implement a server to find whether an entered string a palindrome using socket

Source Code:

PalinClient.java //1st File

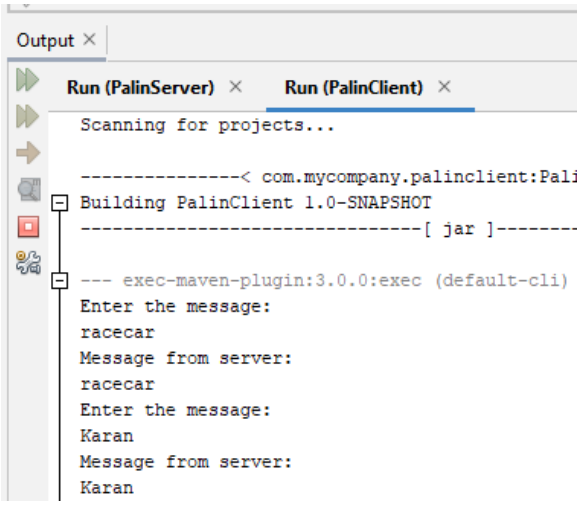
```
package com.mycompany.palinclient;

import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.PrintWriter;
import java.net.InetAddress;
import java.net.Socket;

public class PalinClient {

   Socket soc;
   BufferedReader br,br1;
   PrintWriter out;
   String str;
   public PalinClient()
   {
      try{
 soc=new Socket("127.0.0.1",8765);
br=new BufferedReader(new
InputStreamReader(System.in));
br1=new BufferedReader(new
InputStreamReader(soc.getInputStream()));
 out=new PrintWriter(soc.getOutputStream());
 while(true){
System.out.println("Enter the message: ");
            str=br.readLine();
            out.println(str);
            out.flush();
    System.out.println("Message from server: ");
            str=br1.readLine();
            System.out.println(str);
            if(str.equals("q"))
               break; }
            soc.close();
          }
      catch(Exception e){
      }
   }
   public static void main(String[] args)
   {
      new PalinClient();
   }
}
```

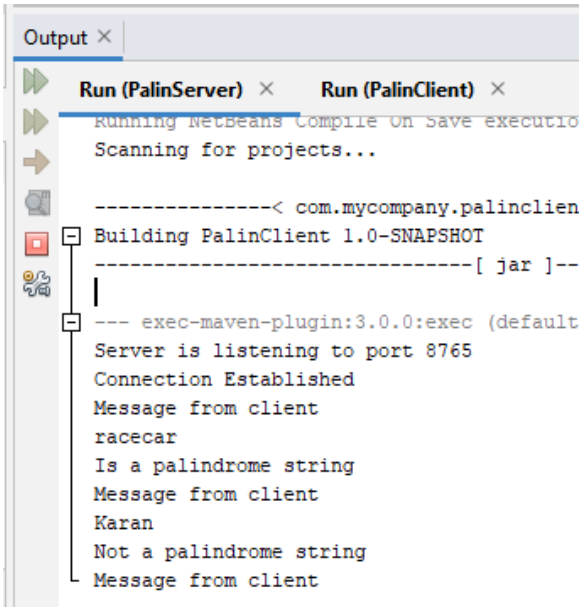
PalinServer.java //2nd File

```
package com.mycompany.palinclient;

import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.PrintWriter;
import java.net.ServerSocket;
import java.net.Socket;


public class PalinServer {
    ServerSocket ss;
    Socket soc;
    BufferedReader br,br1;
    PrintWriter out;
    String str;
    public PalinServer()
    {
    try{
 ss=new ServerSocket(8765);
System.out.println("Server is listening to   port
8765");
  soc=ss.accept();
System.out.println("Connection Established");
br=new BufferedReader(new
InputStreamReader(System.in));
  br1=new BufferedReader(new
InputStreamReader(soc.getInputStream()));
out=new PrintWriter(soc.getOutputStream());
while(true)
        {
 System.out.println("Message from client");
            str=br1.readLine();
            int k=str.length();
```

```
            System.out.println(str);
            int left=0, right=k-1, flag=1;
            while(left<=right)
            {
               if(str.charAt(left)!=(str.charAt(right)))
               {
                  flag=0;
                  break;
               }
               else
               {
                  left++; right--;
               }
            }
            if(flag==0)
     System.out.println("Not a palindrome string");
            else
      System.out.println("Is a palindrome string");
            out.println(str);
            out.flush();
            if(str.equals("q"))
               break;
         }
      }
      catch(Exception e)
      {}
     }
      public static void main(String[] args)
            {
               new PalinServer();
            }
}
```

OUTPUT:





C] Aim: Implement a program to return the reverse of the string using socket

Source Code:

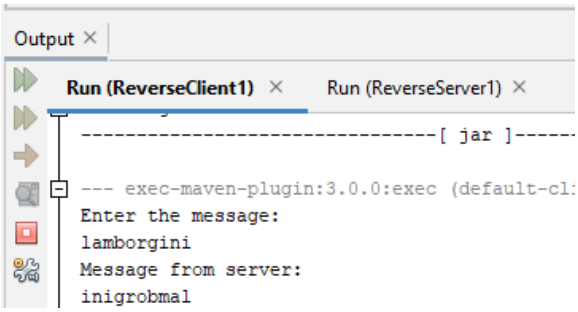
ReverseClient1.java

```
package com.mycompany.reverseclient1;
import java.net.*;
import java.io.*;
public class ReverseClient1 {
    Socket soc;
    BufferedReader br,br1;
    PrintWriter out;
    String str;
    public ReverseClient1(){
    try{
 soc=new Socket("127.0.0.1",8765);
br=new BufferedReader(new
InputStreamReader(System.in));
br1=new BufferedReader(new
InputStreamReader(soc.getInputStream()));
out=new PrintWriter(soc.getOutputStream());
while(true){
 System.out.println("Enter the message: ");
            str=br.readLine();
            out.println(str);
            out.flush();
 System.out.println("Message from server: ");
            str=br1.readLine();
            System.out.println(str);
            if(str.equals("q"))
               break;}
            soc.close();}
        catch (Exception e){}}
    public static void main(String[] args) {
        new ReverseClient1();}}
```

ReverseServer1.java

```
package com.mycompany.reverseclient1;
import java.net.*;
import java.io.*;
public class ReverseServer1 {
    ServerSocket ss;
    Socket soc;
    BufferedReader br,br1;
    PrintWriter out;
    String str;
    public ReverseServer1(){
    try{
    ss=new ServerSocket(8765);
System.out.println("Server is listening to port
8765");
soc=ss.accept();
System.out.println("Connection established!!");
br=new BufferedReader(new
InputStreamReader(System.in));
 br1=new BufferedReader(new
InputStreamReader(soc.getInputStream()));
out=new PrintWriter(soc.getOutputStream());
while(true){
 System.out.println("Message from client");
            str=br1.readLine();
            int k=str.length();
            System.out.println(str);
            String reverse="";
            for(int i=k-1;i>=0;i--)
            { reverse=reverse+str.charAt(i);}
System.out.println("Reverse of the string is:
"+reverse);
            out.println(reverse);
            out.flush();
            if(str.equals("q"))
               break;}
            soc.close();}
        catch (Exception e){}}
    public static void main(String[] args) {
        new ReverseServer1(); }}
```

Output:



D] Implement a program to check whether entered number is even or odd using socket.

Source Code:

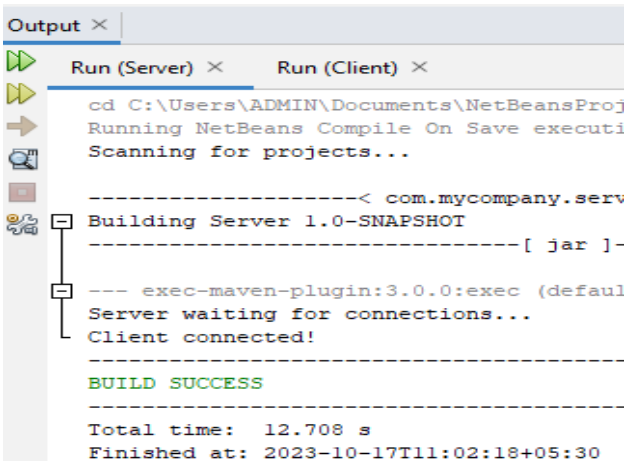
EvenOddClient.java

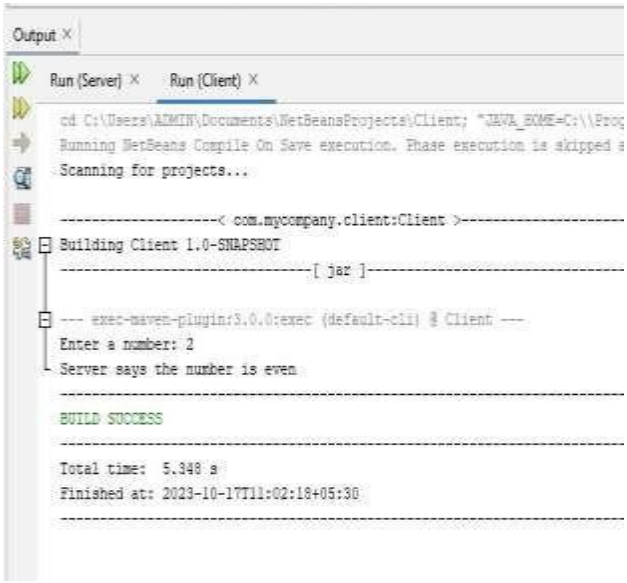
```
package evenoddclient;
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.PrintWriter;
import java.net.Socket;
public class EvenOddClient {
   Socket soc;
   BufferedReader br,br1;
   PrintWriter out;
    int num;
    String str;
 public EvenOddClient() {
  try {
  soc=new Socket("127.0.0.1",8765);
br=new BufferedReader(new
InputStreamReader(System.in));
br1=new BufferedReader(new
InputStreamReader(soc.getInputStream()));
out=new      PrintWriter(soc.getOutputStream());
while(true){
System.out.println("Enter a Number");
              str=br.readLine();
              num=Integer.parseInt(str);
              out.println(num);
              out.flush();
System.out.println("Message from server: ");
num=Integer.parseInt(br1.readLine());
System.out.println(num);
        soc.close();    } }
       catch(Exception e) { } }
    public static void main(String[] args) {
       new EvenOddClient(); } }
```

EvenOddServer.java

```
package evenoddserver;
import java.net.*;
import java.io.*;
public class EvenOddServer {
   ServerSocket ss;
   Socket soc;
   BufferedReader br, br1;
   PrintWriter out;
   int num;
   String str;
   public EvenOddServer() {
   try {
    ss=new ServerSocket(8765);
System.out.println("Server is listening to port
8765");
  soc=ss.accept();
System.out.println("Connection established!!");
br=new BufferedReader(new
InputStreamReader(System.in));
 br1=new BufferedReader(new
InputStreamReader(soc.getInputStream()));
out=new PrintWriter(soc.getOutputStream());
while(true){
System.out.println("Enter a Number: ");
              str=br.readLine();
              num=Integer.parseInt(str);
              System.out.println(num);
              out.flush();
              if(num%2==0){
out.println("Number is Even");  }
              else {
 out.println("Number is odd");   }
              out.println();  }  }
       catch(Exception e) {}  }
    public static void main(String[] args) {
       new EvenOddServer();} }
```

OUTPUT:


Output
Run (Server)
Run (Client)
cd C:\Users\ADMIN\Documents\NetBeansProj
Running NetBeans Compile On Save executi
Scanning for projects...
----------< com.mycompany.serv
Building Server 1.0-SNAPSHOT
--------[jar]-
--- exec-maven-plugin:3.0.0:exec (defaul
Server waiting for connections...
Client connected!
BUILD SUCCESS
Total time: 12.708 s
Finished at: 2023-10-17T11:02:18+05:30


Output
Run (Server)
Run (Client)
Scanning for projects...
----------< com.mycompany.client:Client >----------
Building Client 1.0-SNAPSHOT
--------[jar]----------
--- exec-maven-plugin:3.0.0:exec (default-cli) @ Client ---
Enter a number: 2
Server says the number is even
BUILD SUCCESS
Total time: 5.348 s
Finished at: 2023-10-17T11:02:18+05:30

Practical No:2

Remote Procedure Call

Concept: A remote procedure call is an inter process communication technique that is used for client-server-based applications. A client has a request message that the RPC translates and sends to the server. This request may be a procedure or a function call to a remote server. When the server receives the request, it sends the required response back to the client. The client is blocked while the server is processing the call and only resumed execution after the server is finished.

A] Aim:Implement a Server calculator Add(), Mul(), Sub(), Div() using datagram

Source Code:

RPCCalClient.java

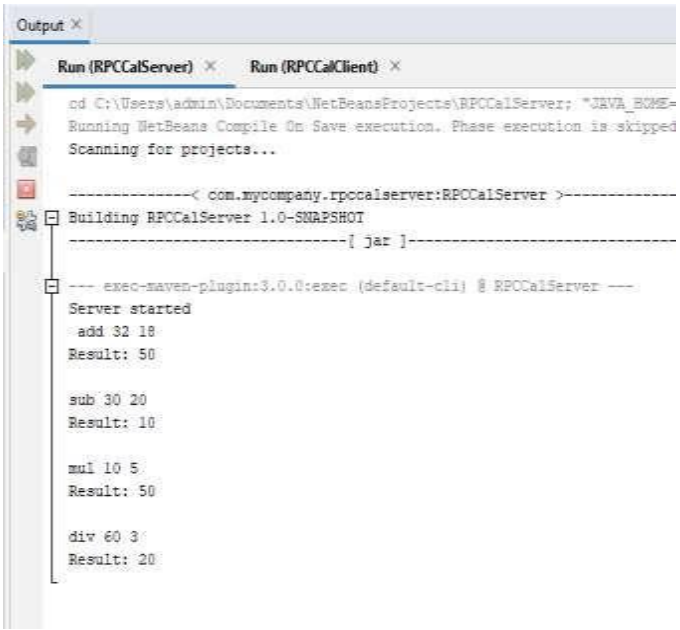
```
package com.mycompany.rpccalclient;
import java.net.*;
import java.io.*;
public class RPCCalClient {
   RPCCalClient(){
   try{
      InetAddress
ia=InetAddress.getLocalHost();
      DatagramSocket ds=new
DatagramSocket();
      DatagramSocket ds1=new
DatagramSocket(1300);
      System.out.println("\nRPC Client.\n");
      System.out.println("Enter method name
and parameter like add 3 4:");
      while(true){
      BufferedReader br=new
BufferedReader(new
InputStreamReader(System.in));
      String str=br.readLine();
      byte b[]=str.getBytes();
      DatagramPacket dp=new
DatagramPacket(b,b.length,ia,1200);
      ds.send(dp);
      dp=new DatagramPacket(b,b.length);
      ds1.receive(dp);
      String s=new
String(dp.getData(),0,dp.getLength());
      System.out.println("\nResult="+s+"\n");
} }
   catch(Exception e){}}
   public static void main(String[] args) {
      new RPCCalClient();}}
```

RPCCalServer.java

```
package com.mycompany.rpccalserver;

import java.net.*;
import java.io.*;
import java.util.*;

public class RPCCalServer {
   DatagramPacket dp;
   DatagramSocket ds;
   String str,methodName,result;
   int val1,val2;
   RPCCalServer(){
   try{
      ds=new DatagramSocket(1200);
      byte b[]=new byte[4096];
      System.out.println("Server started");
      while(true){
      dp=new DatagramPacket(b,b.length);
      ds.receive(dp);
      str=new
String(dp.getData(),0,dp.getLength());
      if(str.equalsIgnoreCase("q")){
         System.exit(1);}
      else{
         StringTokenizer st=new
StringTokenizer(str," ");
         int i=0;
         while(st.hasMoreElements()){
         String token=st.nextToken();
         methodName=token;
         val1=Integer.parseInt(st.nextToken());
```

```
            val2=Integer.parseInt(st.nextToken());
            } }
        System.out.println(str);
        if(methodName.equalsIgnoreCase("add")){
            result=""+add(val1,val2);}
        else
if(methodName.equalsIgnoreCase("sub")){
            result=""+sub(val1,val2);}
        else
if(methodName.equalsIgnoreCase("mul")){
            result=""+mul(val1,val2);}
        else
if(methodName.equalsIgnoreCase("div")){
            result=""+div(val1,val2);}
        else{
            System.out.println("Enter a valid
operation");}
            byte b1[]=result.getBytes();
            DatagramSocket ds1=new
DatagramSocket();
            DatagramPacket dp1=new
DatagramPacket(b1,b1.length,InetAddress.getLo
calHost(),1300);
            System.out.println("Result:
"+result+"\n");
            ds1.send(dp1);}     }
    catch(Exception e){}     }
        public int add(int val1,int val2){
        return val1+val2;}
        public int sub(int val1,int val2){
        return val1-val2;}
        public int mul(int val1,int val2){
        return val1*val2;}
        public int div(int val1,int val2){
        return val1/val2;}
    public static void main(String[] args) {
        new RPCCalServer();    }}
```

OUTPUT:



B] Aim:Implement a Date Time Server containing date() and time() using datagram

Source Code:

DateTimeClient.java

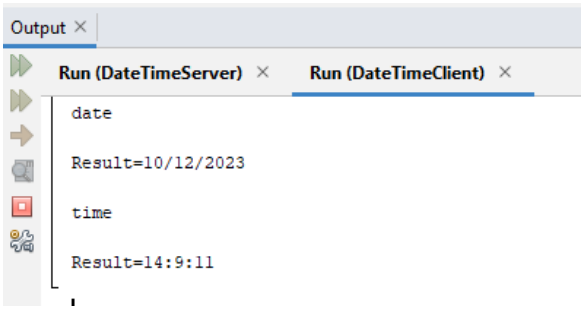
```
package com.mycompany.datetimeclient;
import java.net.*;
import java.io.*;
public class DateTimeClient {
   DateTimeClient(){
      try{
         InetAddress
ia=InetAddress.getLocalHost();
         DatagramSocket ds=new
DatagramSocket();
DatagramSocket ds1=new
DatagramSocket(1300);
System.out.println("\nDate Time Client\n");
 byte b1[]=new byte[1000];  while(true)
   {
BufferedReader br=new BufferedReader(new
InputStreamReader(System.in));
            String str=br.readLine();
            byte b[]=str.getBytes();
            DatagramPacket dp=new
DatagramPacket(b,b.length,ia,1200);
            ds.send(dp);
            dp=new
DatagramPacket(b1,b1.length);
            ds1.receive(dp);
            String s=new
String(dp.getData(),0,dp.getLength());
System.out.println("\nResult="+s+"\n");
         }}
      catch(Exception e){}}
   public static void main(String[] args) {
      new DateTimeClient();}}
```

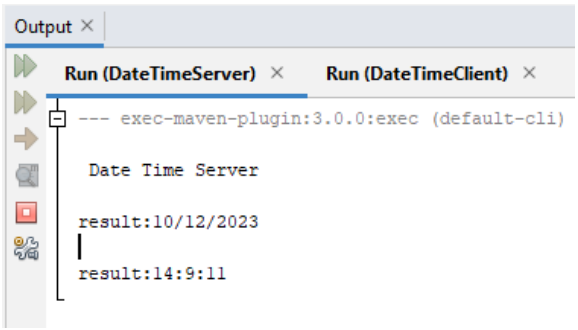
DateTimeServer.java

```
package com.mycompany.datetimeserver;
import java.net.*;
import java.io.*;
import java.util.*;
import java.text.SimpleDateFormat;
public class DateTimeServer {
   DatagramPacket dp;
   DatagramSocket ds;
   String str,methodName,result;
   DateTimeServer()    {
   try{
      ds=new DatagramSocket(1200);
      byte b[]=new byte[4096];
  System.out.println("\n Date Time Server \n");
  while(true)
      {
         dp=new DatagramPacket(b,b.length);
         ds.receive(dp);
         str=new
String(dp.getData(),0,dp.getLength());
         if(str.equalsIgnoreCase("q"))
            System.exit(1);
         else
         {
            StringTokenizer st=new
StringTokenizer(str," ");
            int i=0;
            while(st.hasMoreTokens())
 {
      String token=st.nextToken();
      methodName=token;}}
      Calendar c=Calendar.getInstance();
SimpleDateFormat dateFormat=new
SimpleDateFormat("MM/dd/yyyy");
Date d=c.getTime();

 InetAddress ia=InetAddress.getLocalHost();

  if(methodName.equalsIgnoreCase("date"))
     result=""+dateFormat.format(d);
  else if(methodName.equalsIgnoreCase("time"))
  {

result=""+d.getHours()+":"+d.getMinutes()+":"+
d.getSeconds();
   }
```

```
            byte b1[]=result.getBytes();
            DatagramSocket ds1=new
DatagramSocket();
DatagramPacket dp1=new
DatagramPacket(b1,b1.length,ia,1300);
System.out.println("result:"+result+"\n");
            ds1.send(dp1);
            }}
    catch(Exception e){}}
    public static void main(String[] args) {
        new DateTimeServer();}}
```

OUTPUT:





Practical No:3

Remote Method Invocation

Concept: The Remote Method invocation is an API that provide a mechanism to create distributed application in java. The client invoke method via an interface. These methods are implement on the server side.

A] Aim: Implement a Server calculator containing ADD(), MUL(),SUB(),DIV() Using RMI

CalciClient.java

```
package com.mycompany.calciclient;
import java.net.MalformedURLException;
import java.rmi.Naming;
import java.rmi.NotBoundException;
import java.rmi.RemoteException;
import java.util.Scanner;
public class CalciClient {
   public static void main(String[] args) throws
NotBoundException, MalformedURLException,
RemoteException {
       Scanner sc=new Scanner(System.in);
       try{
 CalciInterface c=
(CalciInterface)Naming.lookup("rmi://localhost:
1099/CalciInterface");
System.out.println("Client is connected to
server.");
 System.out.println("Please enter your choice:
\n1. add\n2. sub\n3. mul\n4. div\n");
          int choice=sc.nextInt();
          int x,y;
          switch(choice){
          case 1:
          System.out.println("Enter x and y: ");
                x=sc.nextInt();
                y=sc.nextInt();
                System.out.println(c.add(x,y));
                break;
              case 2:
 System.out.println("Enter x and y: ");
    x=sc.nextInt();
    y=sc.nextInt();
    System.out.println(c.sub(x,y));
    break;
case 3:
System.out.println("Enter x and y: ");
    x=sc.nextInt();
    y=sc.nextInt();
  System.out.println(c.mul(x,y));
  break;
case 4
 System.out.println("Enter x and y: ");
    x=sc.nextInt();
    y=sc.nextInt();
    System.out.println(c.div(x,y));
    break;    }}
       catch(Exception e){} }}
```

Calciserver.java

```
package com.mycompany.calciclient;
import  java.rmi.NotBoundException;
import java.rmi.Remote;
import java.rmi.RemoteException;
import java.rmi.registry.Registry;
public class CalciServer {
   public static  void main(String[] args) throws
RemoteException, NotBoundException {
      Registry
r=java.rmi.registry.LocateRegistry.createRegistr
y(1099);
      r.rebind("CalciInterface", (Remote) new
CalciRMI());
      System.out.println("server is running");}}
```

CalciRMI.java

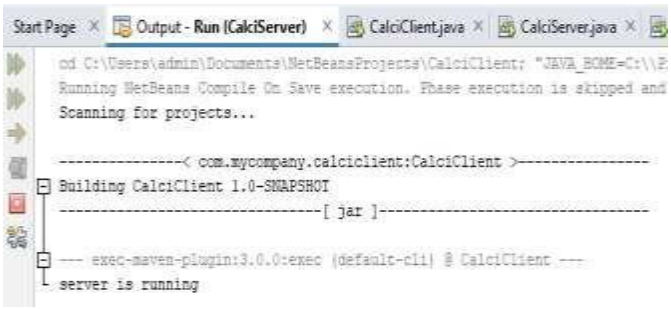
```
package com.mycompany.calciclient;
import java.rmi.RemoteException;
import java.rmi.server.UnicastRemoteObject;
```

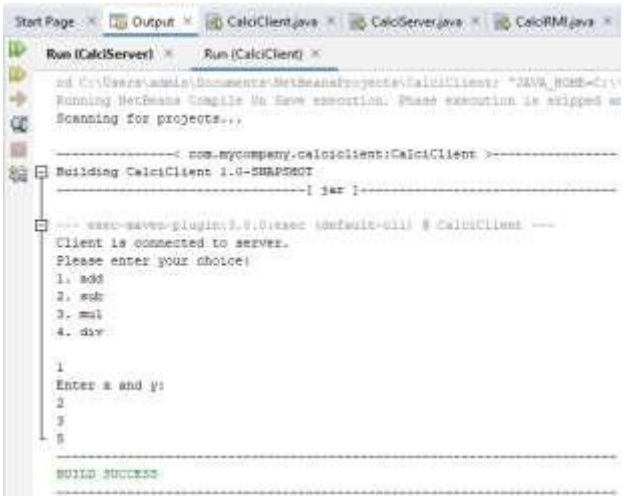
```
public class CalciRMI extends
UnicastRemoteObject implements
CalciInterface {
     public CalciRMI() throws
RemoteException{
          int a,b;}
  public int add (int a,int b)throws
RemoteException{
          return a+b;}
 public int sub (int a,int b)throws
RemoteException{
          return a-b;}
     public int mul (int a,int b)throws
RemoteException{
          return a*b;}
 public int div (int a,int b)throws
RemoteException{
           return a/b;}
public static void main(String[] args){
}}
```

CalciInterface.java

```
package com.mycompany.calciclient;
import java.rmi.Remote;
import java.rmi.RemoteException;
public interface CalciInterface extends
Remote{public int add (int x,int y)throws
RemoteException;
public int sub(int x,int y) throws
RemoteException;
public int mul(int x,int y) throws
RemoteException;
public int div(int x,int y) throws
RemoteException;        }
```

OUTPUT:





B] Aim: Retrieve time and date function from server to client. This program should display server date and time.

DatetimeClient.java

```
package com.mycompany.datetimeclient;
import java.net.MalformedURLException;
import java.rmi.*;
public class DateTimeClient {
   public static void main(String args[]) {
      try{
DateTimeInterface intf=(DateTimeInterface)
Naming.lookup("rmi://localhost:1099/DateServe
r");
System.out.println("The date on the server is:
"+intf.getDate());  }
      catch(MalformedURLException |
NotBoundException | RemoteException e) {
         }  } }
```

Datetimeserver.java

```
package com.mycompany.datetimeclient;
import java.net.MalformedURLException;
import java.rmi.*;
import java.rmi.registry.Registry;
public class DateTimeServer {
   public static void main(String args[]) {
      try{
Registry r= java.rmi.registry.LocateRegistry.
createRegistry(1099);
DateTimeRMI di=new DateTimeRMI();
Naming.rebind("DateServer", (Remote) di);
System.out.println("Datetime Server is ready");
}
      catch(MalformedURLException |
RemoteException e){}  }}
```

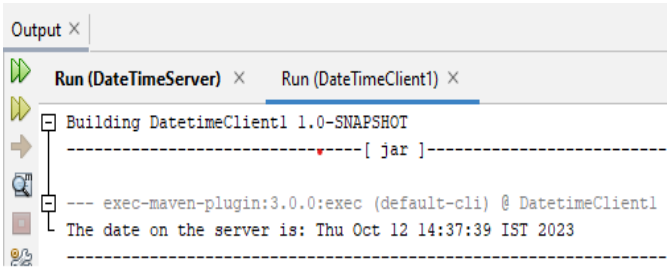
Datetimeinterface.java

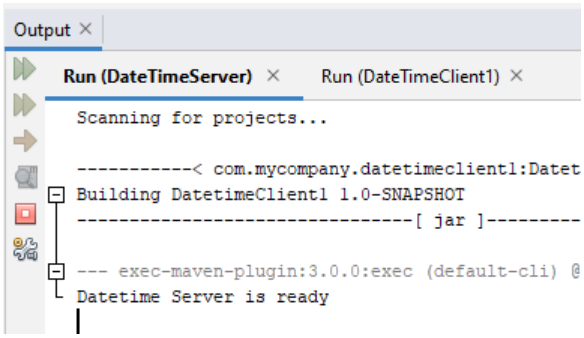
```
package com.mycompany.datetimeclient;
import java.rmi.*;
public interface DateTimeInterface extends
Remote{
   String getDate() throws RemoteException;}
```

DatetimeRMI.java

```
package com.mycompany.datetimeclient;
import java.rmi.*;
import java.rmi.server.*;
import java.util.*;
public class DateTimeRMI extends
UnicastRemoteObject implements
DateTimeInterface {
   public DateTimeRMI() throws
RemoteException {}
      public String getDate() {
         Date d=new Date();
         return(d.toString());    }}
```

OUTPUT:





C] Aim: Equation solver. The client should provide an equation to the server through an interface. The server will solve the expression given by the client. Such as $(a-b)^2 = a^2 - 2ab + b^2$, $(a+b)^2 = a^2 + 2ab + b^2$, $(a+b)^3 = a^3+3a^2b+3ab^2+b^3$ etc

EqSolverClient .java

```
package com.mycompany.eqsolverclient;
import java.rmi.*;
import java.io.*;
public class EqSolverClient {
   public static void main(String[] args) {
      try {
         int num1=0, num2=0, res=0,choice;
         EqSolverInterface
object=(EqSolverInterface)Naming.lookup("hell
o");
BufferedReader br=new BufferedReader(new
InputStreamReader(System.in));
         System.out.println("Equations:-");
         System.out.println("1. (a-b)2");
         System.out.println("2. (a+b)2");
         System.out.println("3. (a-b)3");
         System.out.println("4. (a+b)3");
         System.out.println("5. Exit");
               while (true)
         {     System.out.println("Choose the
equation: ");
          choice=Integer.parseInt(br.readLine());
            if(choice<=4)
            {
            System.out.println("Enter the values
of a and b");
            num1=Integer.parseInt(br.readLine());
    num2=Integer.parseInt(br.readLine());
            }
            switch(choice)
            {     case 1:{
        res=object.solveEq1(num1,num2);
            System.out.println("Result is: "+res);
               break;}
            case 2:
            {res=object.solveEq2(num1,num2);
               System.out.println("Result is:
"+res);
             break;}
              case 3:
            {res=object.solveEq3(num1,num2);
               System.out.println("Result is:
"+res);          break;}      case 4:
            {res=object.solveEq4(num1,num2);
               System.out.println("Result is:
"+res);
              break;} case 5:
            {     System.exit(0);
               break;}
            default:
            { System.out.println("Invalid
option");       break;    }
          }       }     }
      catch(Exception e) {}  }}
```

EqSolverServer.java

```
package com.mycompany.eqsolverclient;
import java.rmi.*;
import java.rmi.registry.Registry;
public class EqSolverServer {
   public static void main(String[] args) throws
RemoteException, NotBoundException {
      try{
      Registry r=
java.rmi.registry.LocateRegistry.createRegistry(
1099);
      EquationSolverRMI obj=new
EquationSolverRMI();
      r.rebind("hello", obj);
      System.out.println("Equation Solver Server
is ready");     }
      catch(Exception e) {}   } }
```

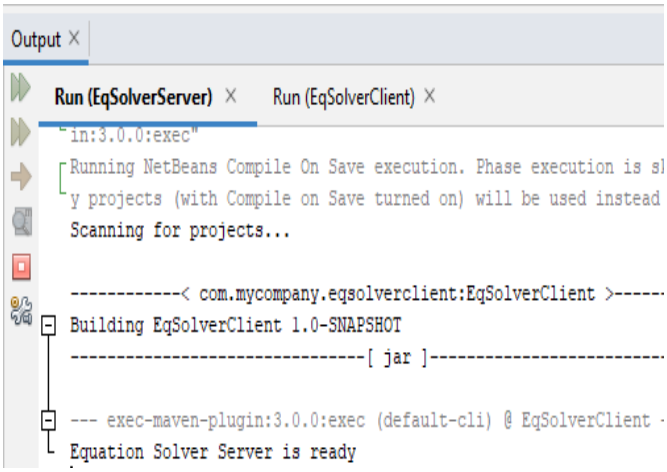
EqSolverInterface .java

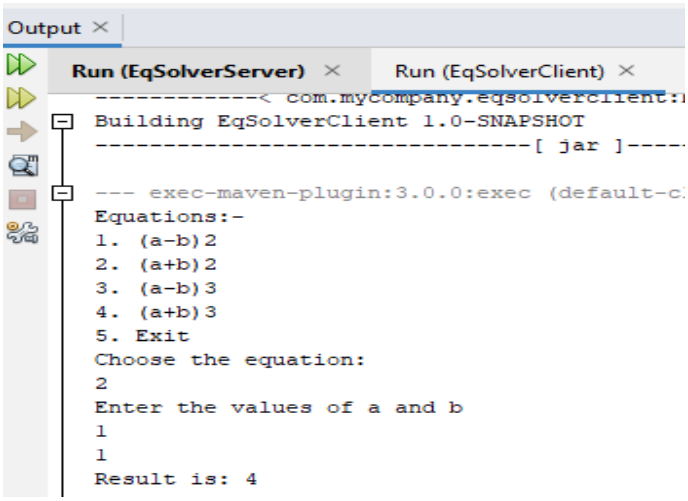
```
package com.mycompany.eqsolverclient;
import java.rmi.*;
import java.rmi.Remote;
import java.rmi.RemoteException;
public interface EqSolverInterface extends
Remote{
   public int solveEq1(int a, int b) throws
RemoteException;
   public int solveEq2(int a, int b) throws
RemoteException;
   public int solveEq3(int a, int b) throws
RemoteException;
   public int solveEq4(int a, int b) throws
RemoteException; }
```

EquationSolverRMI.java

```
package com.mycompany.eqsolverclient;
import java.rmi.*;
import java.rmi.server.*;
import java.util.*;
public class EquationSolverRMI extends
UnicastRemoteObject implements
EqSolverInterface {
   public EquationSolverRMI() throws
RemoteException {}
      public int solveEq1(int a,int b) throws
RemoteException {
         int ans=(a*a)-(2*a*b)+(b*b);
         return ans;     }
      public int solveEq2(int a,int b) throws
RemoteException {
         int ans=(a*a)+(2*a*b)+(b*b);
         return ans;     }
      public int solveEq3(int a,int b) throws
RemoteException {
         int ans=(a*a*a)-(3*a*a*b)+(3*a*b*b)-
(b*b*b);      return ans;     }
      public int solveEq4(int a,int b) throws
RemoteException {          int
ans=(a*a*a)+(3*a*a*b)+(3*a*b*b)+(b*b*b);
         return ans;     }  }
```

OUTPUT:





PRACTICAL NO.4

Remote Object Communication

Concept: Pass remote objects from the server to the client. The client will receive the stub object (through remote interfaces) and saves it in an object variable with the same type as the remote interface. Then the client can access the actual object on the server through the variable.

A]. Aim: Using MySQL create College database. Create table Book and retrieve the Book information from Library database using Remote Object Communication concept.

DBClient.java

```
package com.mycompany.dbclient;
import java.rmi.*;
import java.io.*;

public class DBClient
{
   public static void main(String[] args)
   {
       String db="", sql="", ch="", ch1="",
res="";
      try
      {
 BufferedReader br= new
BufferedReader(new
InputStreamReader(System.in));
          while(true)
          {
 System.out.println("Retrieve College
Information.");
             db="college";
System.out.println("Select an option");
System.out.println("a) Retrieve Student
Information.");
 System.out.println("b) Retreive Books
Information.");
System.out.println("Enter your choice: ");
                ch1=br.readLine();
                if(ch1.equals("a"))
                {
                   sql="select * from student";
                }
                else if(ch1.equals("b"))
                {
                   sql="select * from book";
                }
                else
                {
System.out.println("Please select a valid
option.");
                  System.exit(0);
                }
 DBIntf
id=(DBIntf)Naming.lookup("rmi://localhost:
1099/DBConn");
             res=id.getData(sql,db);
             System.out.println(res);
      }
   }
      catch (Exception e)
      {
         e.printStackTrace();
      }
}
}
```

DBServer.java

```
package com.mycompany.dbclient;
import java.rmi.*;
import java.rmi.registry.Registry;
public class DBServer {
   public static void main(String[] args) {
      try {
Registry r=
java.rmi.registry.LocateRegistry.
createRegistry(1099);
 DBCollege di=new DBCollege();
Naming.rebind("DBConn",(Remote) di);
System.out.println("Server Registered."); }
      catch(Exception e) {
         e.printStackTrace();} } }
```

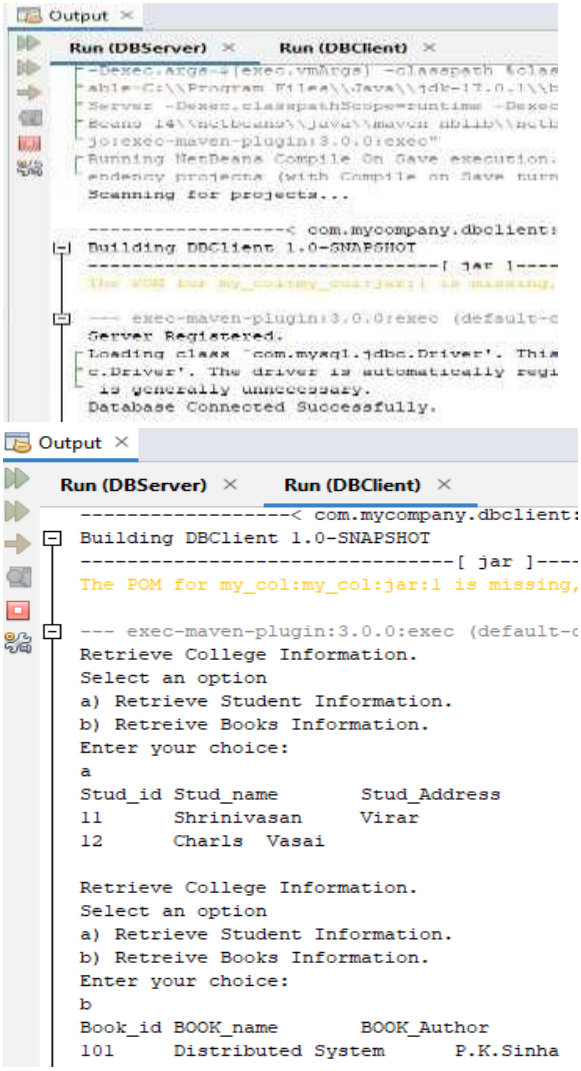
DBIntf.java

```
package com.mycompany.dbclient;
import java.rmi.*;
public interface DBIntf extends Remote
{
   public String getData(String s, String db)
throws RemoteException;
}
```

DBCollege.java

```
package com.mycompany.dbclient;
import java.rmi.*;
import java.rmi.server.*;
import java.sql.*;
public class DBCollege extends
UnicastRemoteObject implements DBIntf{
   String str="", str1="";
   public DBCollege() throws
RemoteException {}
   public String getData(String sql, String
dsn) throws RemoteException {
       String
URL="jdbc:mysql://localhost/"+dsn;
//dsn=data source name
       try {
Class.forName("com.mysql.jdbc.Driver");
Connection con=DriverManager.
getConnection(URL,"root","");
System.out.println("Database Connected
Successfully.");
          Statement s=con.createStatement();
          ResultSet rs=s.executeQuery(sql);
ResultSetMetaData rsmd=rs.getMetaData();
    str1="";
    str="";
for(int i=1;i<=rsmd.getColumnCount();i++)
{
str1=str1+rsmd.getColumnName(i)+"\t";  }
          System.out.println();
          while(rs.next()) {
for(int i=1;i<=rsmd.getColumnCount();i++)
{
          str=str+rs.getString(i)+"\t"; }
       str=str+"\n"; } }
catch(Exception e) {
   e.printStackTrace();  }
return(str1+"\n"+str); }}
```

OUTPUT:



B]. Aim: Using MySQL create Employee database. Create table employee and retrieve the employee information from the Employee database using Remote Object Communication concept.

EmployeeClient.java

```
package com.mycompany.employeeclient;
import java.rmi.*;
import java.io.*;
public class EmployeeClient {
   public static void main(String[] args) {
String db="", sql="", ch="", ch1="", res="";
      try {
BufferedReader br= new BufferedReader(new
InputStreamReader(System.in));
         while(true) {
            System.out.println("1. Press E to retrieve Employee information");
            System.out.println("2. Press x to exit the system.");
            ch=br.readLine();
            if(ch.equals("E")||ch.equals("e")) {
               db="emp";
               sql="select * from employee";
}
            else if(ch.equals("x")) {
               System.exit(0);  }
            else {
 System.out.println("Please select a valid option."); }
EmployeeIntf id=(EmployeeIntf)Naming.
lookup("rmi://localhost:1099/EmpDBConn");
            res=id.getInfo(sql,db);
            System.out.println(res); } }
      catch (Exception e) {
            e.printStackTrace(); } }}
```

EmployeeServer.java

```
package com.mycompany.employeeclient;
import java.rmi.*;
import java.rmi.registry.Registry;
public class EmployeeServer {
   public static void main(String[] args) {
      try {
Registry r= java.rmi.registry.LocateRegistry.
createRegistry(1099);
EmployeeDBRMI di=new EmployeeDBRMI();
Naming.rebind("EmpDBConn",(Remote) di);
System.out.println("Server Registered.");
}
      catch(Exception e) { } }}
```

EmployeeIntf.java

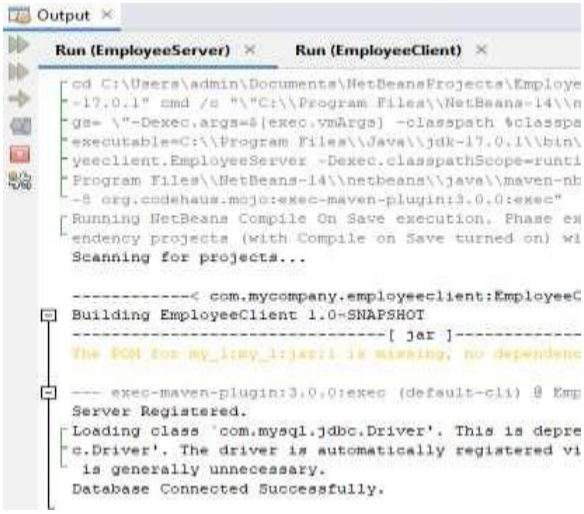
```
package com.mycompany.employeeclient;
import java.rmi.*;
public interface EmployeeIntf extends Remote
{
   public String getInfo(String s, String db)
throws RemoteException;   }
```

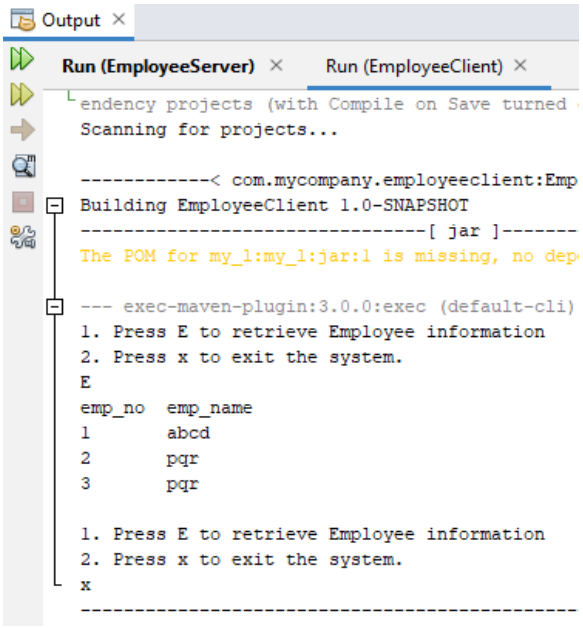
EmployeeDBRMI.java

```
package com.mycompany.employeeclient;
import java.rmi.*;
import java.rmi.server.*;
import java.sql.*;

public class EmployeeDBRMI extends
UnicastRemoteObject implements
EmployeeIntf{
   String str="", str1="";
```

```
public EmployeeDBRMI() throws RemoteException {}
public String getInfo(String sql, String dsn) throws RemoteException {
String URL="jdbc:mysql://localhost/"+dsn;
try {
Class.forName("com.mysql.jdbc.Driver");
Connection con=DriverManager.getConnection(URL,"root","");
System.out.println("Database Connected Successfully.");
Statement s=con.createStatement();
ResultSet rs=s.executeQuery(sql);
ResultSetMetaData rsmd=rs.getMetaData();
str1="";
str="";
for(int i=1;i<=rsmd.getColumnCount();i++)
{
str1=str1+rsmd.getColumnName(i)+"\t"; }
System.out.println();
while(rs.next()) {
for(int i=1;i<=rsmd.getColumnCount();i++)
{
str=str+rs.getString(i)+"\t"; }
str=str+"\n"; } }
catch(Exception e) {
e.printStackTrace();}
return(str1+"\n"+str); }}
```

OUTPUT:





PRACTICAL NO.5

Mutual Exclusion

Concept: Token ring algorithm solves the mutual exclusion existing in the process communication.

Aim: Implementation of mutual exclusion using Token Ring Technique.

TokenRing.java

```
import java.net.*;
import java.io.*;
public class TokenRing {
   public static DatagramSocket ds;
   public static DatagramPacket dp;
   public static void main(String[] args) throws Exception {
      try {
 ds=new DatagramSocket(1000); }
catch(Exception e) {e.printStackTrace();}
while(true) {
         byte buff[]=new byte[1024];
         ds.receive(dp=new DatagramPacket(buff, buff.length));
  String str=new String(dp.getData(),0,dp.getLength());
System.out.println("Message from "+str);
}}}
```

TokenRingClient1.java

```
import java.net.*;
import java.io.*;
public class TokenRingClient1 {
   public static DatagramSocket ds;
   public static DatagramPacket dp;
   public static BufferedReader br;
   public static void main(String[] args) throws Exception {
      boolean hasToken;
      try {
         ds=new DatagramSocket(100); }
      catch(Exceptione) {e.printStackTrace();}
   hasToken=true;
   while(true) {
   if(hasToken==true) {
System.out.println("Do you want to enter data? (yes/no): ");
 br=new BufferedReader(new InputStreamReader(System.in));
 String ans=br.readLine();
    if(ans.equalsIgnoreCase("yes")) {
System.out.println("Ready to send.");
System.out.println("Sending...");
System.out.println("Enter the data: ");
br=new BufferedReader(new InputStreamReader(System.in));
 String str="Client-1===> "+br.readLine();
      byte buff[]=new byte[1024];
      buff=str.getBytes();
 ds.send(new DatagramPacket (buff, buff.length,InetAddress.getLocalHost(), 1000));
System.out.println("Now sending..."); }
 else if(ans.equalsIgnoreCase("no")) {
      System.out.println("I am busy.");
            //Sending message to client 2
            String msg="Token";
            byte buff1[]=new byte[1024];
            buff1=msg.getBytes();
            ds.send(new DatagramPacket(buff1, buff1.length, InetAddress.getLocalHost(),200));
            hasToken=false;
   //receiving message from Client 2
            byte buff2[]=new byte[1024];
            ds.receive(dp=new DatagramPacket(buff2, buff2.length));
        String clientmsg=new String(dp.getData(),0,dp.getLength());
```

```
System.out.println("The data is: "+clientmsg);
    if(clientmsg.equals("Token"))
       hasToken=true;
System.out.println("I am leaving busy state.");
   }  }
        else {
System.out.println("Enter in receive mode.");
            byte buff[]=new byte[1024];
            ds.receive(dp=new DatagramPacket(buff, buff.length));
            String clientmsg1=new String(dp.getData(),0,dp.getLength());
            System.out.println("The data is: "+clientmsg1);
            if(clientmsg1.equals("Token")) {
               hasToken=true; } } } } }
```

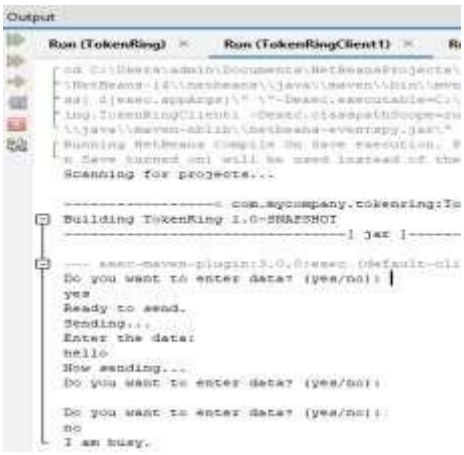
TokenRingClient2.java

```
import java.net.*;
import java.io.*;
public class TokenRingClient2 {
   public static DatagramSocket ds;
   public static DatagramPacket dp;
   public static BufferedReader br;
   public static void main(String[] args) throws Exception {
      boolean hasToken;
      try {
         ds=new DatagramSocket(200);  }
      catch(Exception e) {e.printStackTrace();}
      hasToken=false;
      while(true) {
         if(hasToken==true) {
System.out.println("Do you want to enter data? (yes/no): ");
            br=new BufferedReader(new InputStreamReader(System.in));
            String ans=br.readLine();
            if(ans.equalsIgnoreCase("yes")) {
System.out.println("Ready to send.");
System.out.println("Sending...");
System.out.println("Enter the data: ");
br=new BufferedReader(new InputStreamReader(System.in));
String str="Client-2===> "+br.readLine();
               byte buff1[]=new byte[1024];
               buff1=str.getBytes();
ds.send(new DatagramPacket(buff1, buff1.length,
InetAddress.getLocalHost(),1000));
System.out.println("Data sent."); }
            else {
  //Sending message to client 1
               String clientmsg="Token";
               byte buff2[]=new byte[1024];
               buff2=clientmsg.getBytes();
               ds.send(new DatagramPacket(buff2, buff2.length,
InetAddress.getLocalHost(),100));
               hasToken=false;  }  }
         else {
            try {
               byte buff[]=new byte[1024];
System.out.println("Enter in receive mode.");
               ds.receive(dp=new DatagramPacket(buff, buff.length));
               String clientmsg1=new String(dp.getData(),0,dp.getLength());
System.out.println("The data is: "+clientmsg1);
  if(clientmsg1.equals("Token"))
        hasToken=true;  }
catch(Exception e){e.printStackTrace();}}}}}
```

.

Output:


Output
Run (TokenRing)
Run (TokenRingClient1)
Building TokenRing 1.0-SNAPSHOT
Message from Client-1===> hello
Message from Client-2===> hi


Output
Run (TokenRing)
Run (TokenRingClient1)
Building TokenRing 1.0-SNAPSHOT
Do you want to enter data? (yes/no):
yes
Ready to send.
Sending...
Enter the data:
hello
Now sending...
Do you want to enter data? (yes/no):
Do you want to enter data? (yes/no):
no
I am busy.


Output
Run (TokenRing)
Run (TokenRingClient1)
Run (TokenRingClient2)
Building TokenRing 1.0-SNAPSHOT
Enter in receive mode.
The data is: Token
Do you want to enter data? (yes/no):
yes
Ready to send.
Sending...
Enter the data:
hi
Data sent.
Do you want to enter data? (yes/no):

Practical No.6

Implementation of Cloud Computing Services

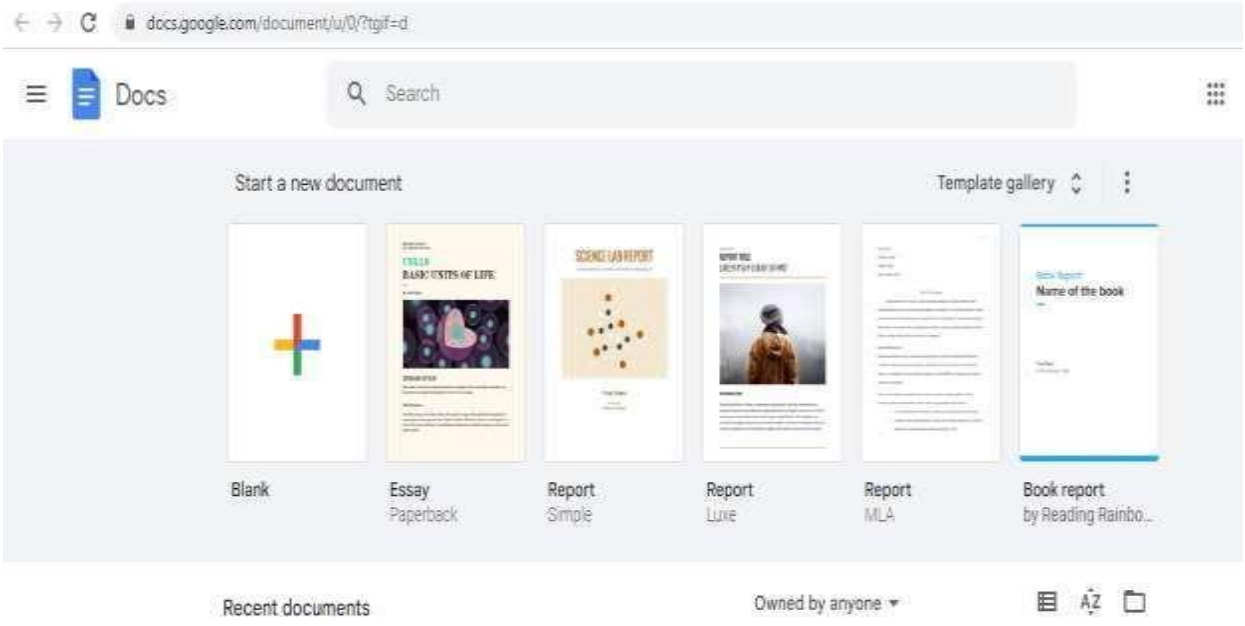
Aim: To Implement a concept of Storage as a Service using Google Docs

Concept: Storage as a service (STaaS) is a business model in which a company leases or rents its storage infrastructure to another company or individuals to store data. Small companies and individuals often find this to be a convenient methodology for managing backups, and providing cost savings in personnel, hardware and physical space.

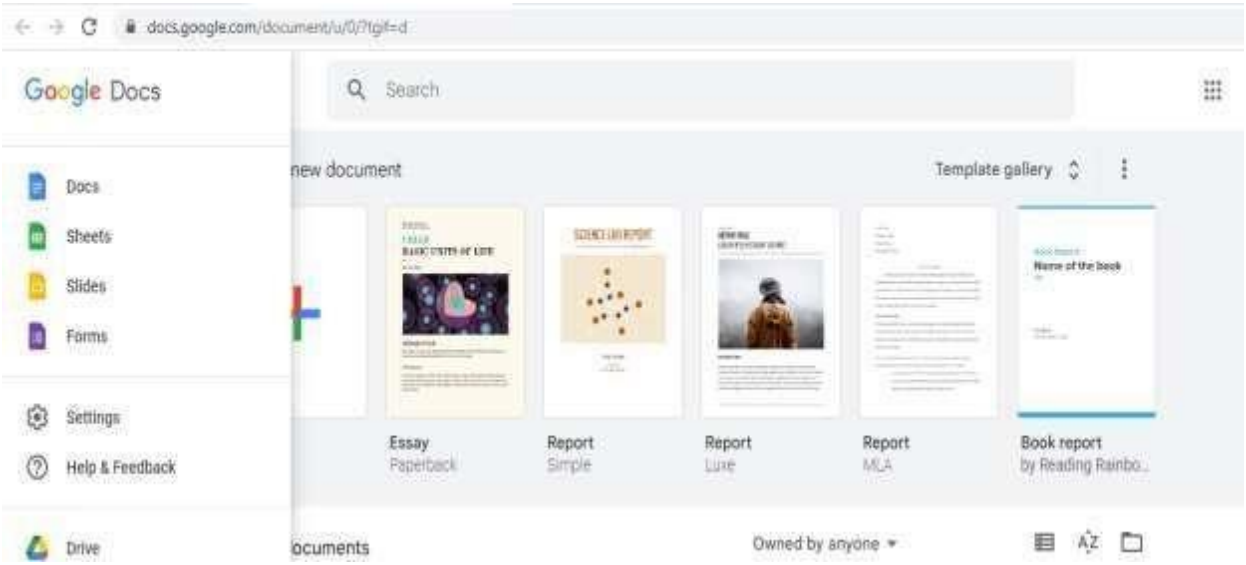
A company providing STaaS may be called a storage service provider (SSP). Storage as a service can also be referred to as hosted storage as a Service is a business model in which a large company rents space in their storage infrastructure to a smaller company or individual. In the enterprise, STaaS vendors are targeting secondary storage applications by promoting SaaS as a convenient way to manage backups. The key advantage to STaaS in the enterprise is in cost savings --in personnel, in hardware and in physical storage space.

Procedure:

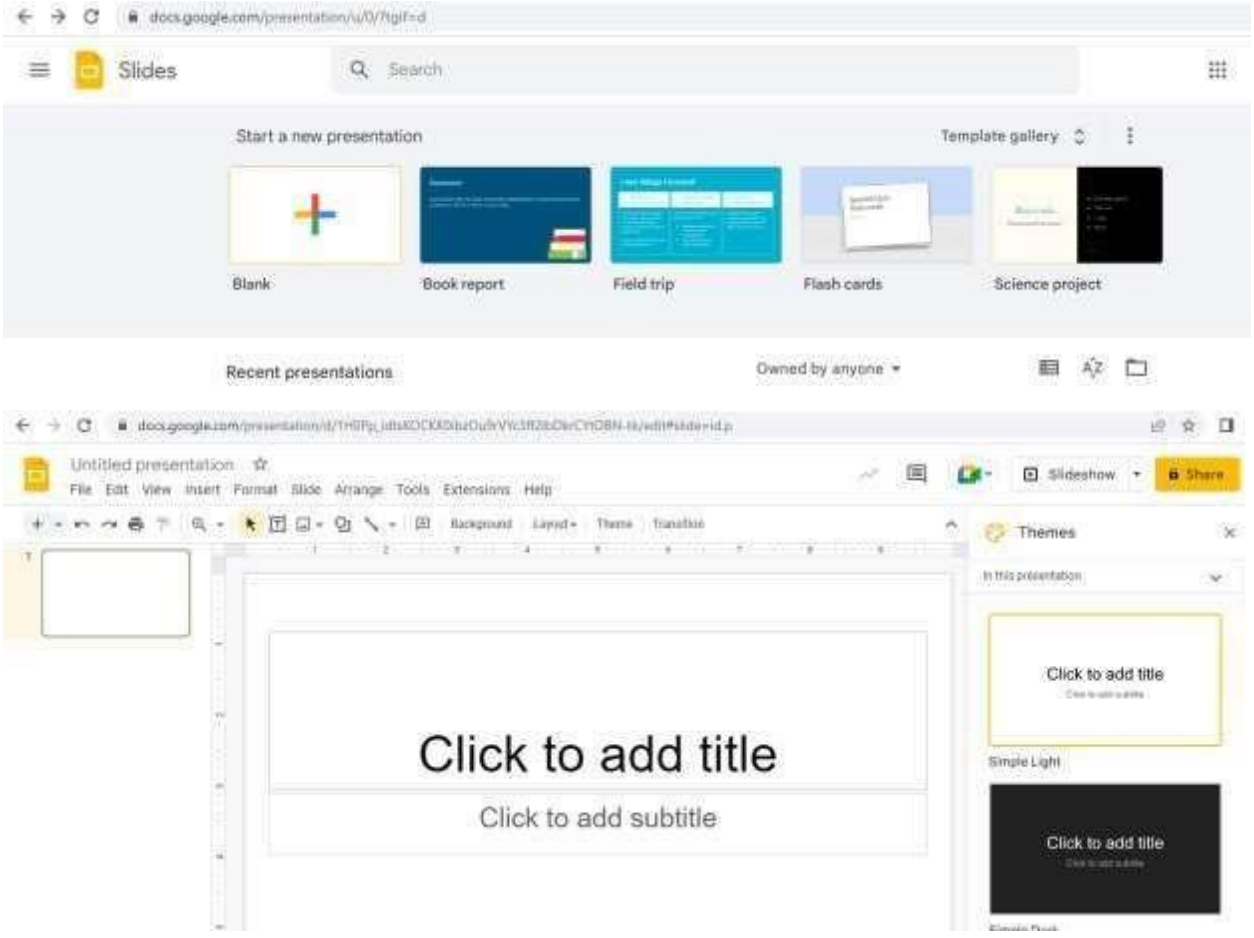
Step 1: Login to Gmail Account and go to Google Docs

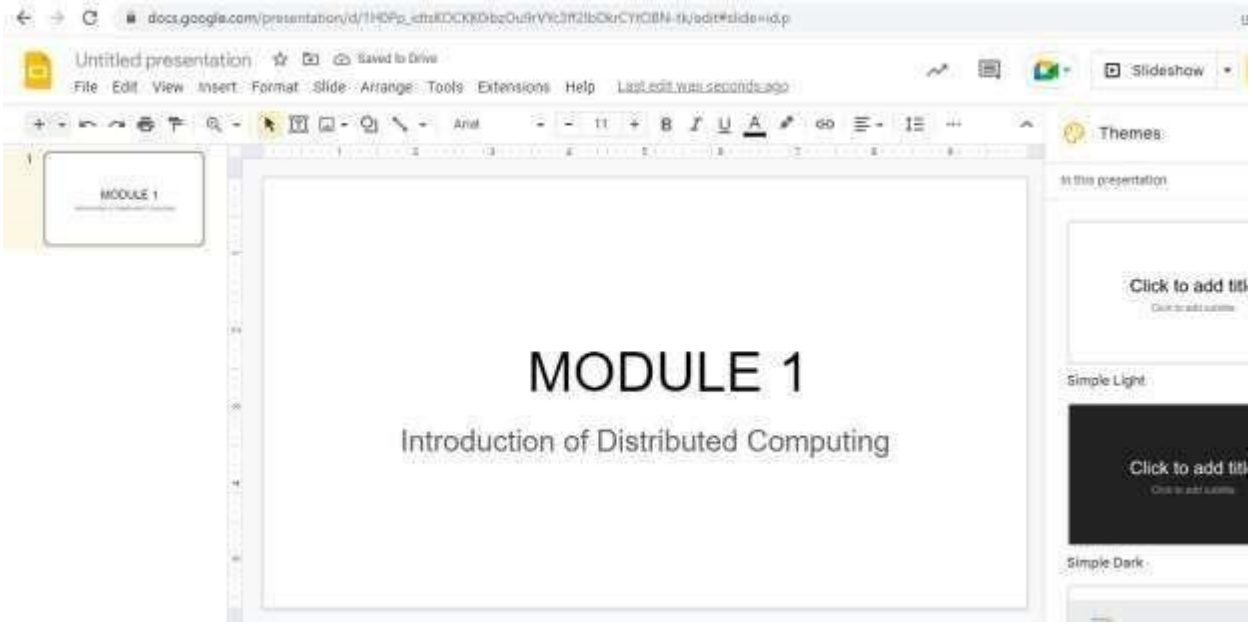


Step 2: Click on three lines which are to the left top corner



Step 3: Edit one Slide Online with Google Slides





Conclusion:

Google Docs provide an efficient way for storage of data. It fits well in Storage as a service (STaaS). It has varied options to create documents, presentations and also spreadsheets. It saves documents automatically after a few seconds and can be shared anywhere on the Internet at the click of a button.

Practical No.7

Implementation of Identity Management using Cloud Computing concept

Aim: To implement concept of Identity Management in cloud computing

Concept: Identity management (ID management) is the organizational process for identifying, authenticating and authorizing individuals or groups of people to have access to applications, systems or networks by associating user rights and restrictions with established identities. The managed identities can also refer to software processes that need access to organizational systems.

Identity management includes authenticating users and determining whether they're allowed access to particular systems. ID management works hand-in-hand with identity access management systems. Identity management is focused on authentication, while access management is aimed at authorization.

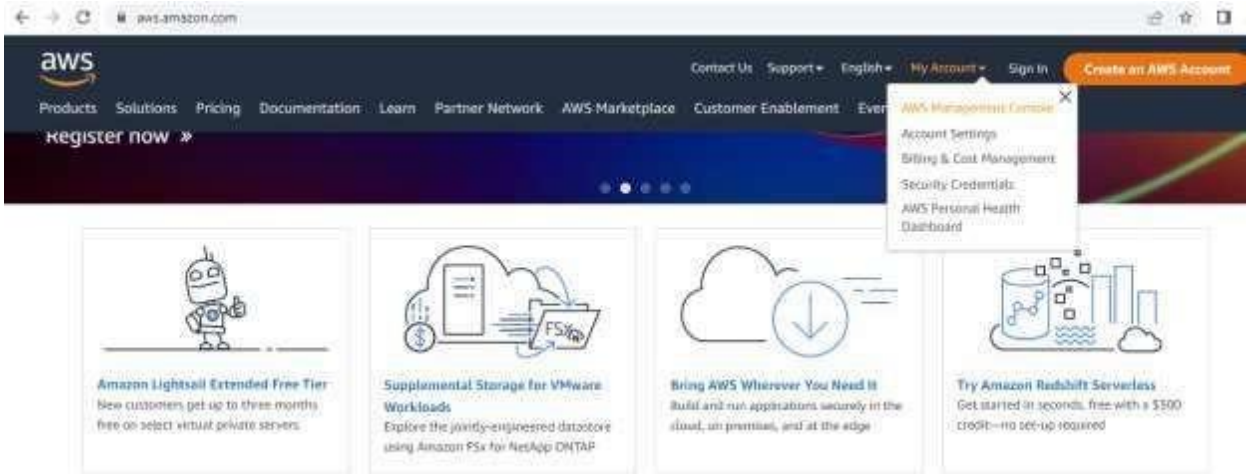
ID management determines whether a user has access to systems, but also sets the level of access and permissions a user has on a particular system. For instance, a user may be authorized to access a system but be restricted from

The main goal of identity management is to ensure that only authenticated users are granted access to the specific applications, systems or IT environments for which they are authorized. This includes control over user provisioning and the process of onboarding new users such as employees, partners, clients and other stakeholders. Identity management also includes control over the process of authorizing system or network permissions for existing users and the off boarding of users who are no longer authorized to access organization systems.

Procedure:

Step1: Open the following link https://aws.amazon.com/

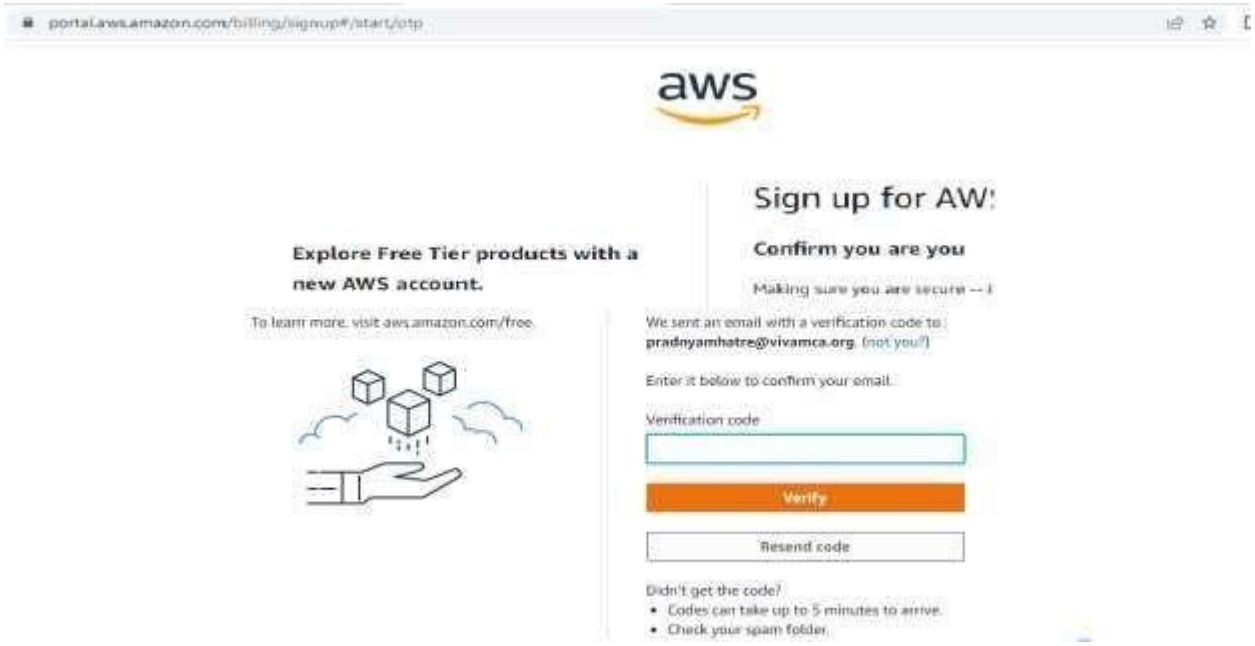
Step2: Go to my Account-> AWS management console



Step3: click on Create new user AWS account

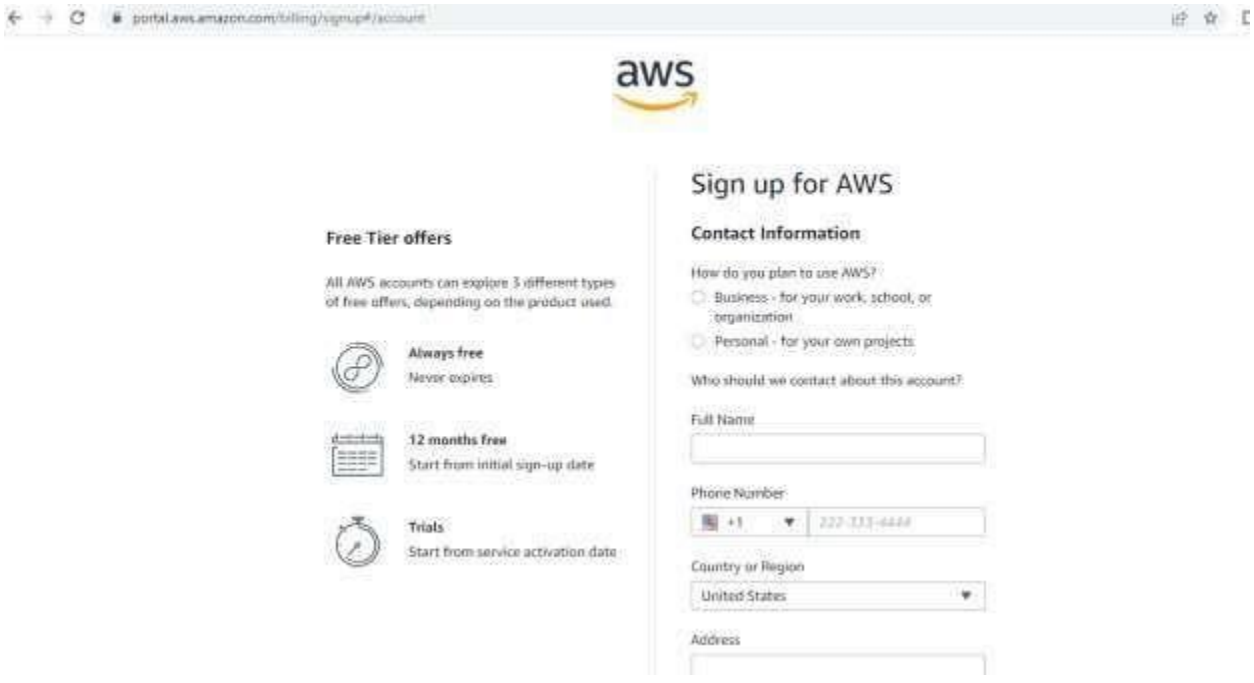
Step4: Fill all the details and click on Verify email address

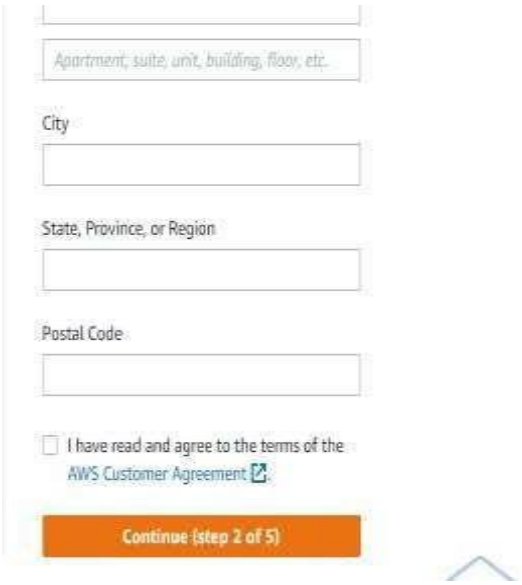
Step 5: Add verification code and click on verify.



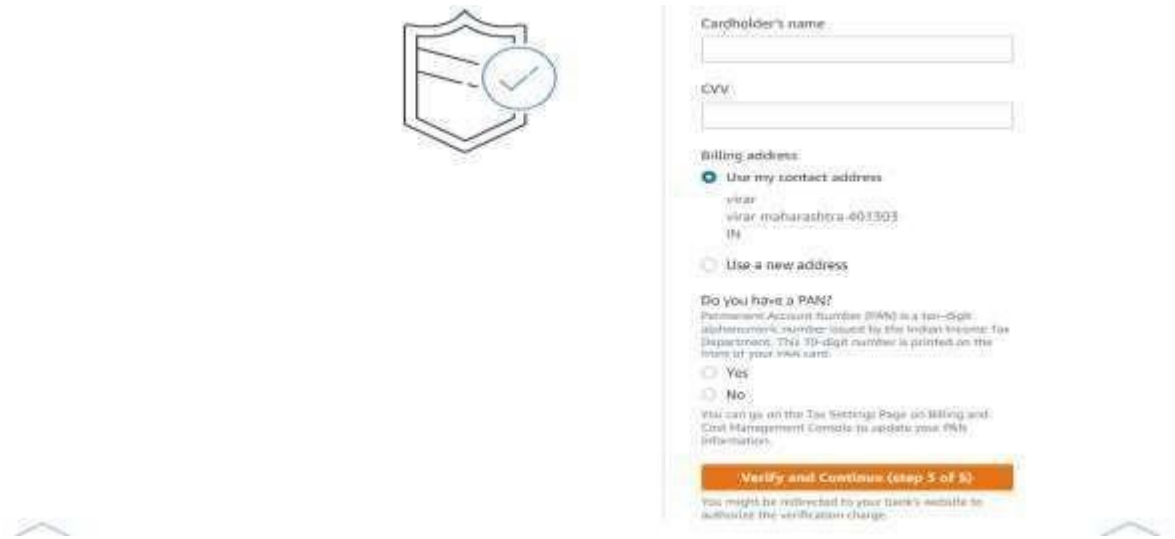
Step 6: Create your password and click on continue.

Step 7: Fill Contact information and click on continue.



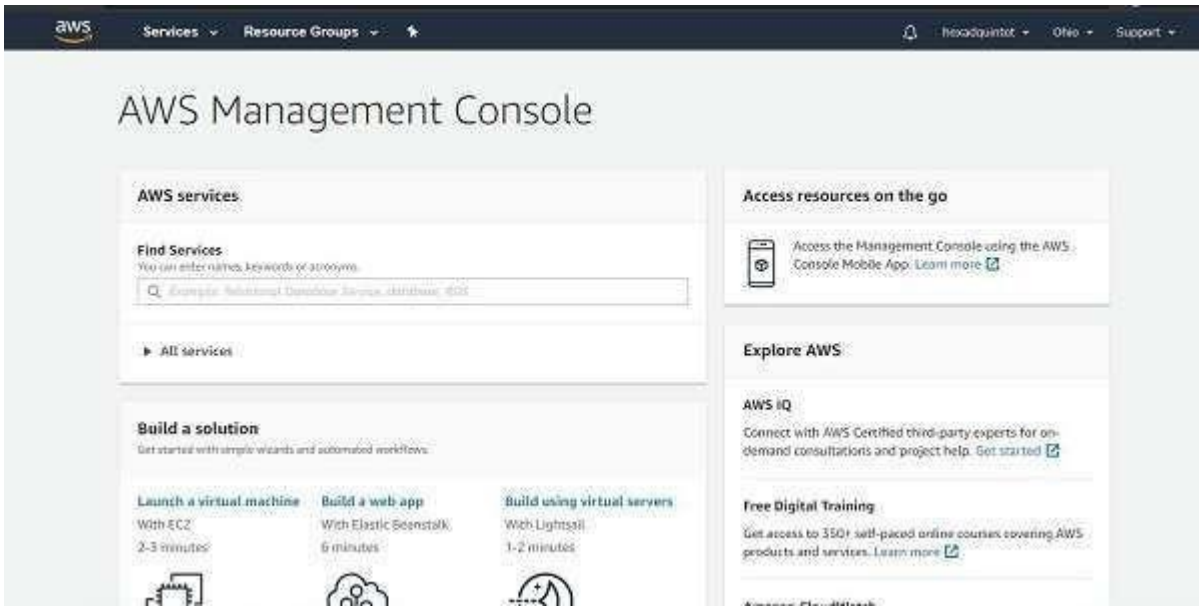


Step 8: Now AWS will ask for credit card and debit card details. You have to close the browser

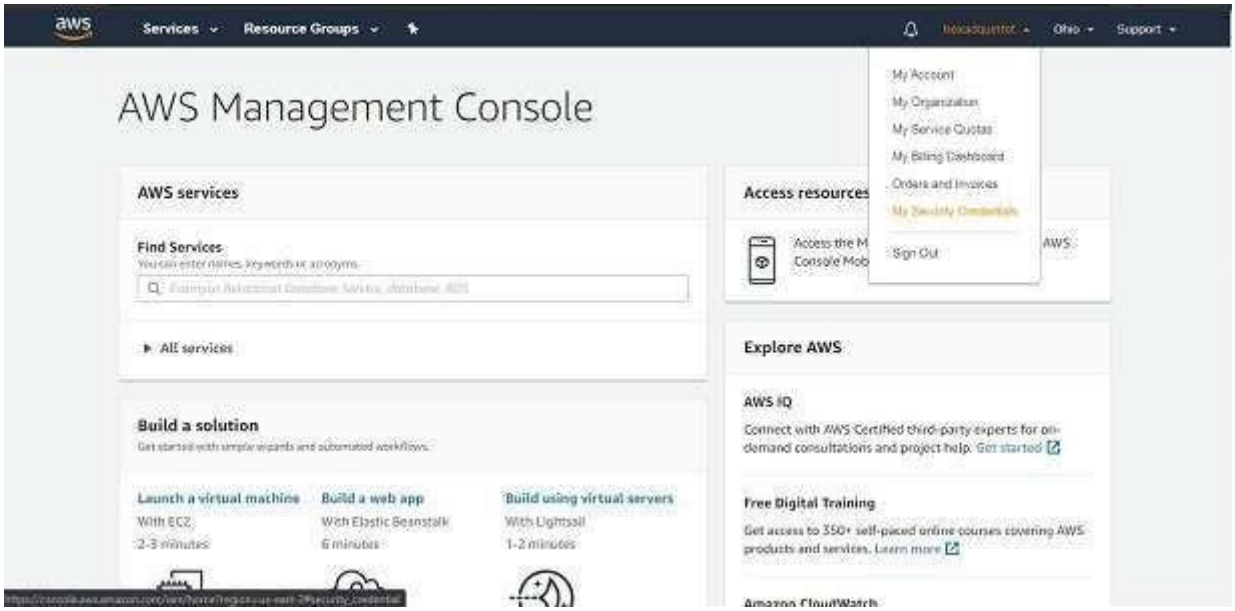


Step 9: Go to my Account->AWS Management console

Step 10: you will get the following screen

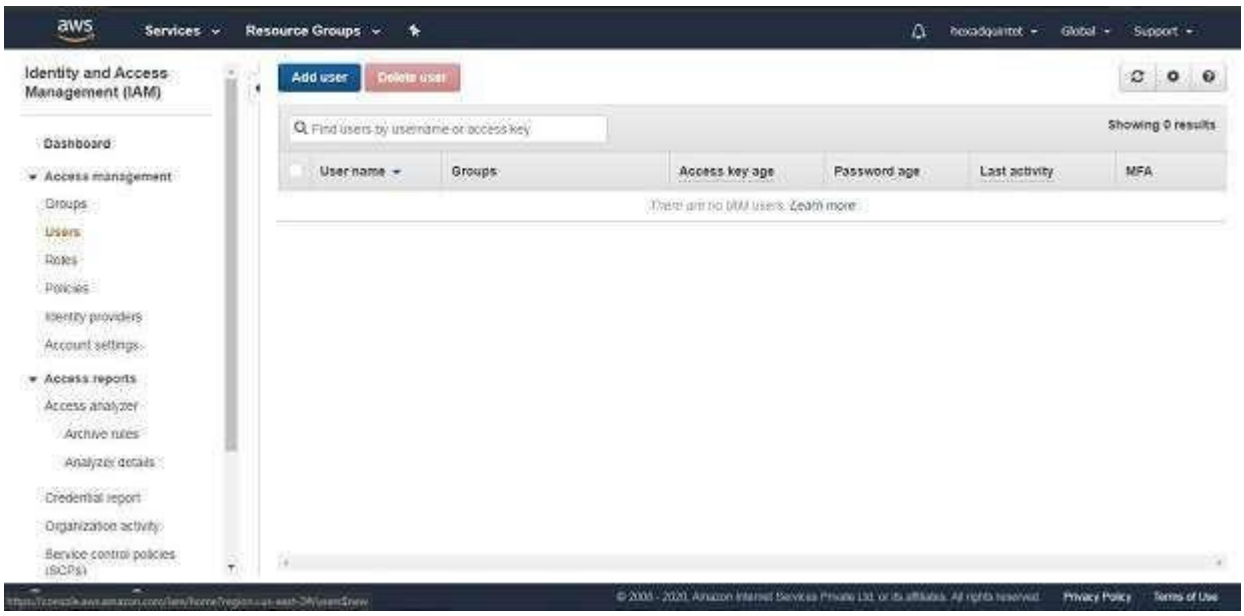


Step 11: Go to My Security credential

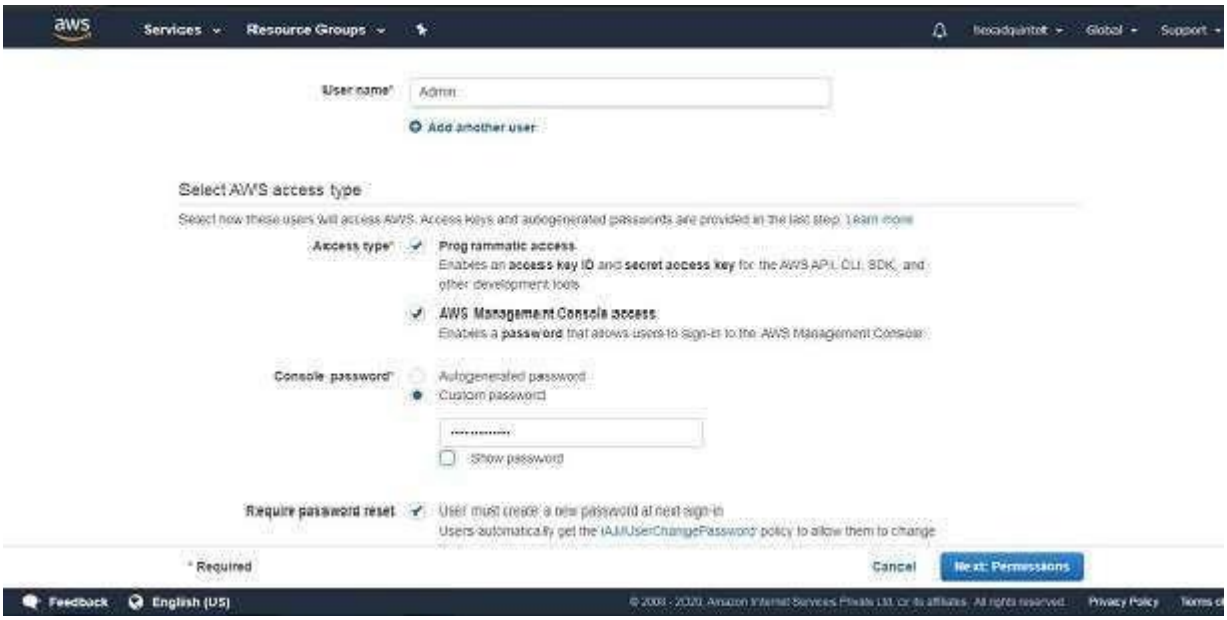


Step 12: now click on user

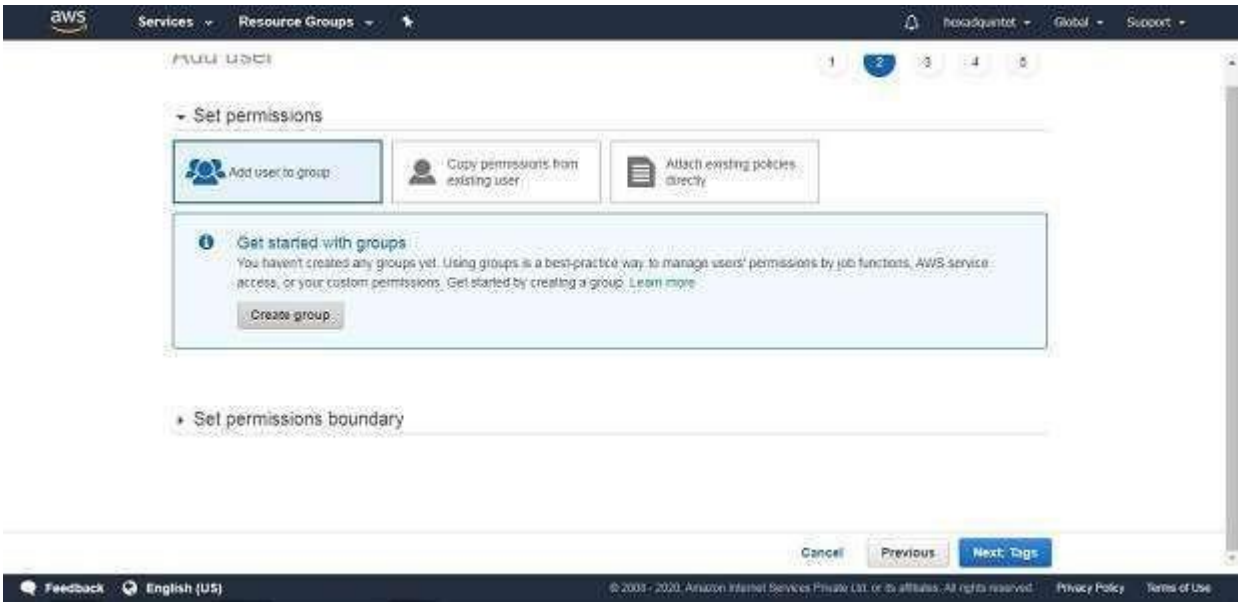
Step 13: Click on add user



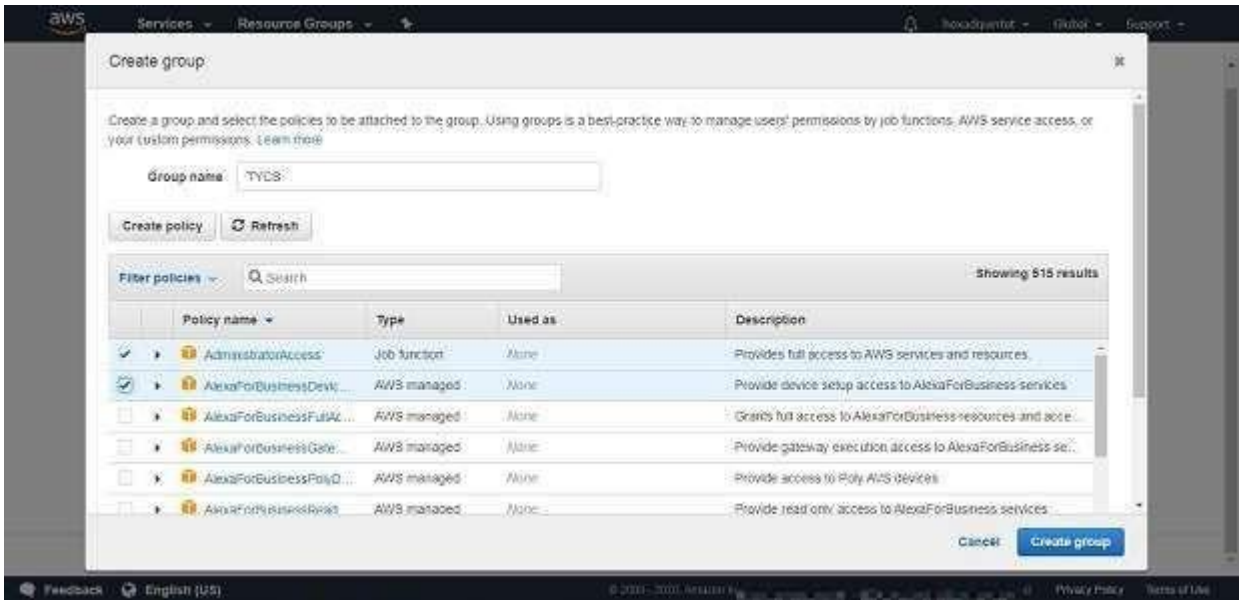
Step 14: Provide the user name and check the check box in front of programmatic access and AWS Management console Access and enter the password for new user Click on custom password and click on next permission



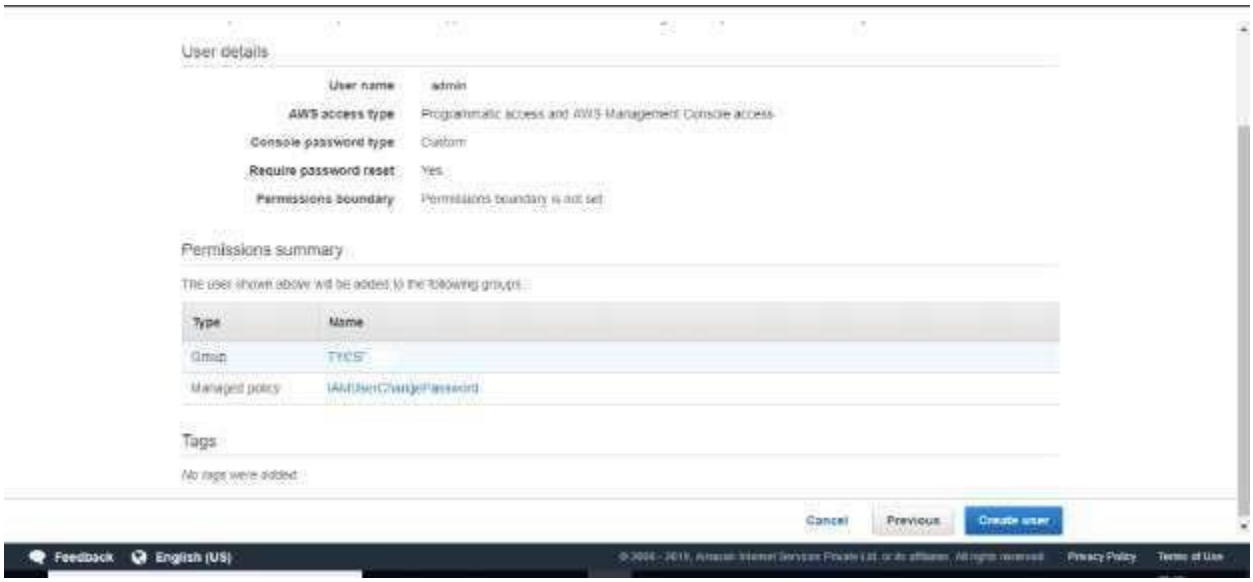
Step 15: click on create Group



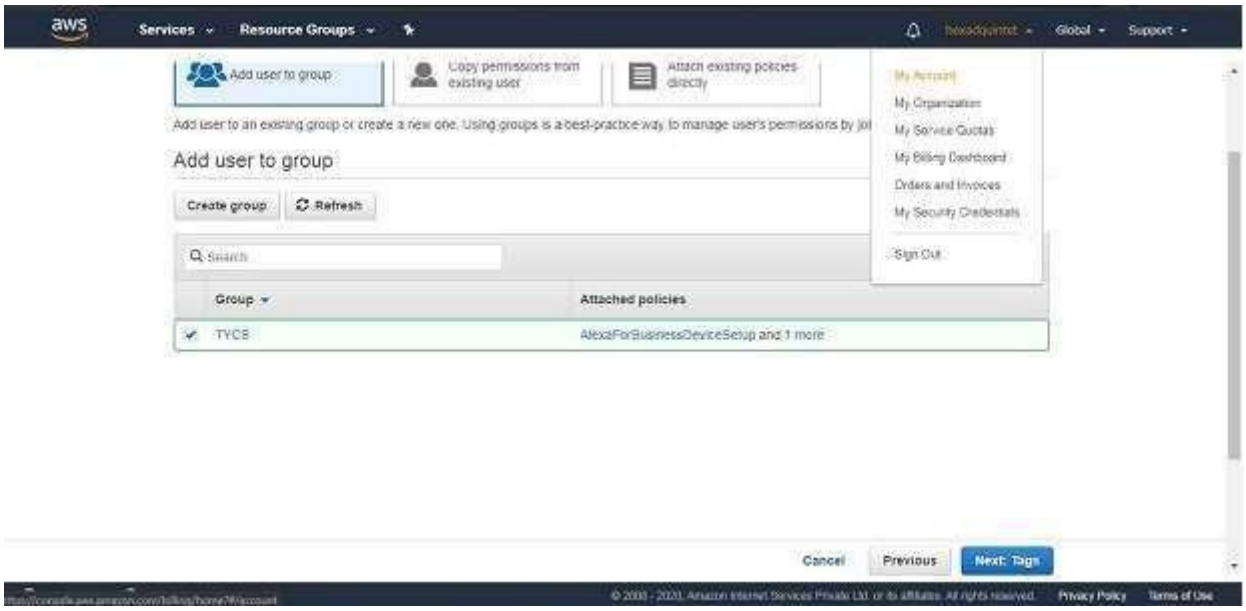
Step16: Fill the information and click on Create Group



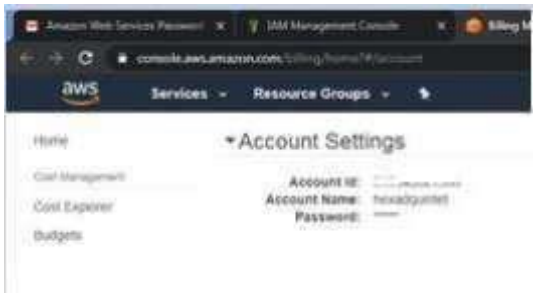
Step17: click on next tag leave blank, again click on next review leave as it is and click on create user



Step 18: click on close

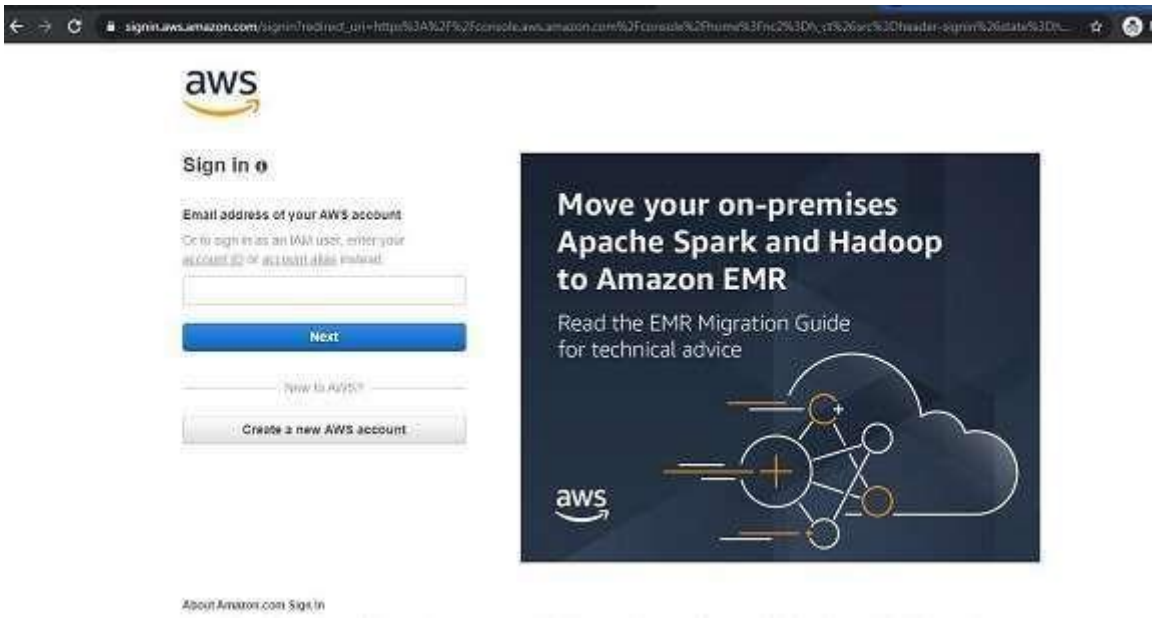


And COPY Account ID



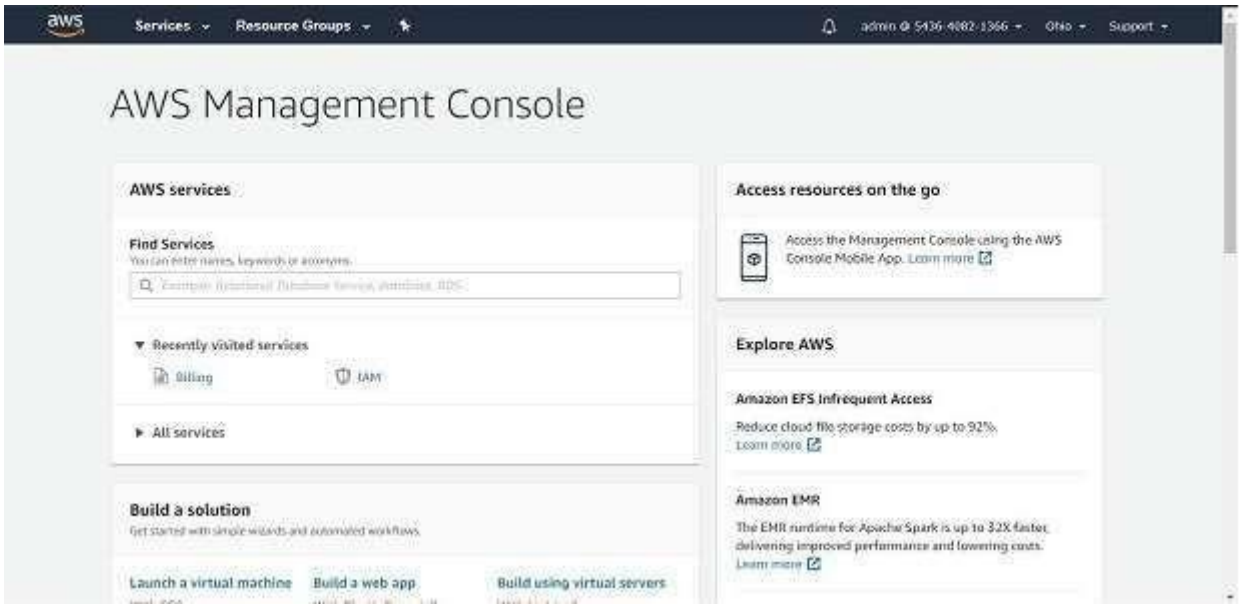
Now logout the admin account and try to login as user (newly created) .

Step 19: again Go to my Account->AWS Management console



Step 20: Click on next Provide the Account ID username and password and click on sign in It will ask you to change the password which is been set by administrator

Yow will redirect to home screen



Conclusion: Hence we have studied the concept and implementation of identity management using amazon aws.

Practical No. 9

App Development using Cloud Computing

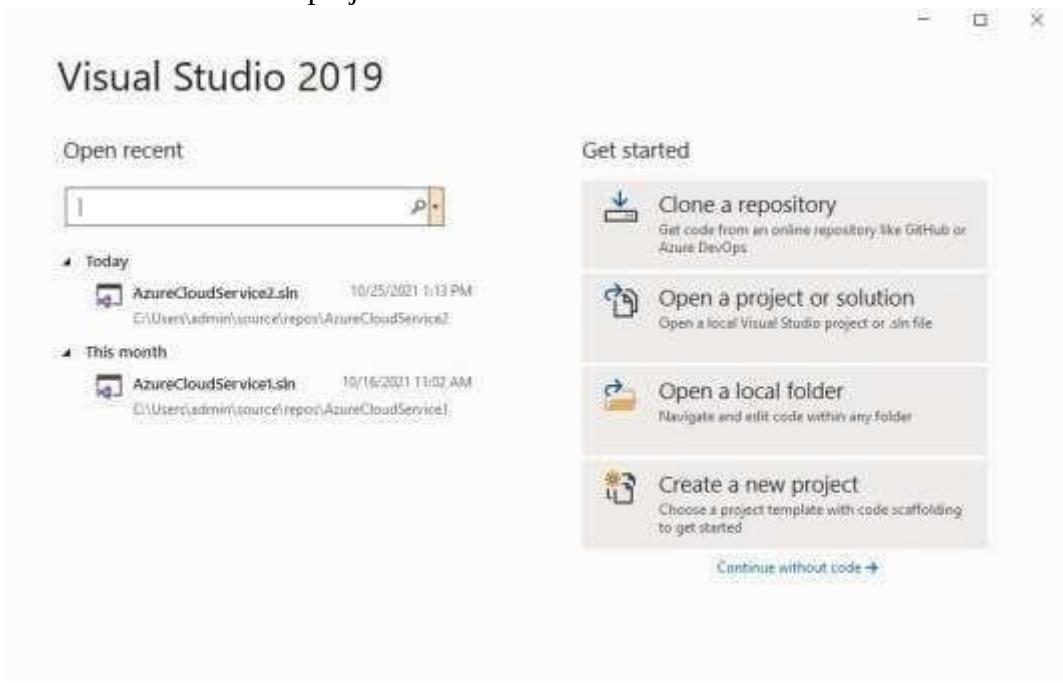
Aim: Develop application for Microsoft Azure

Step 1:

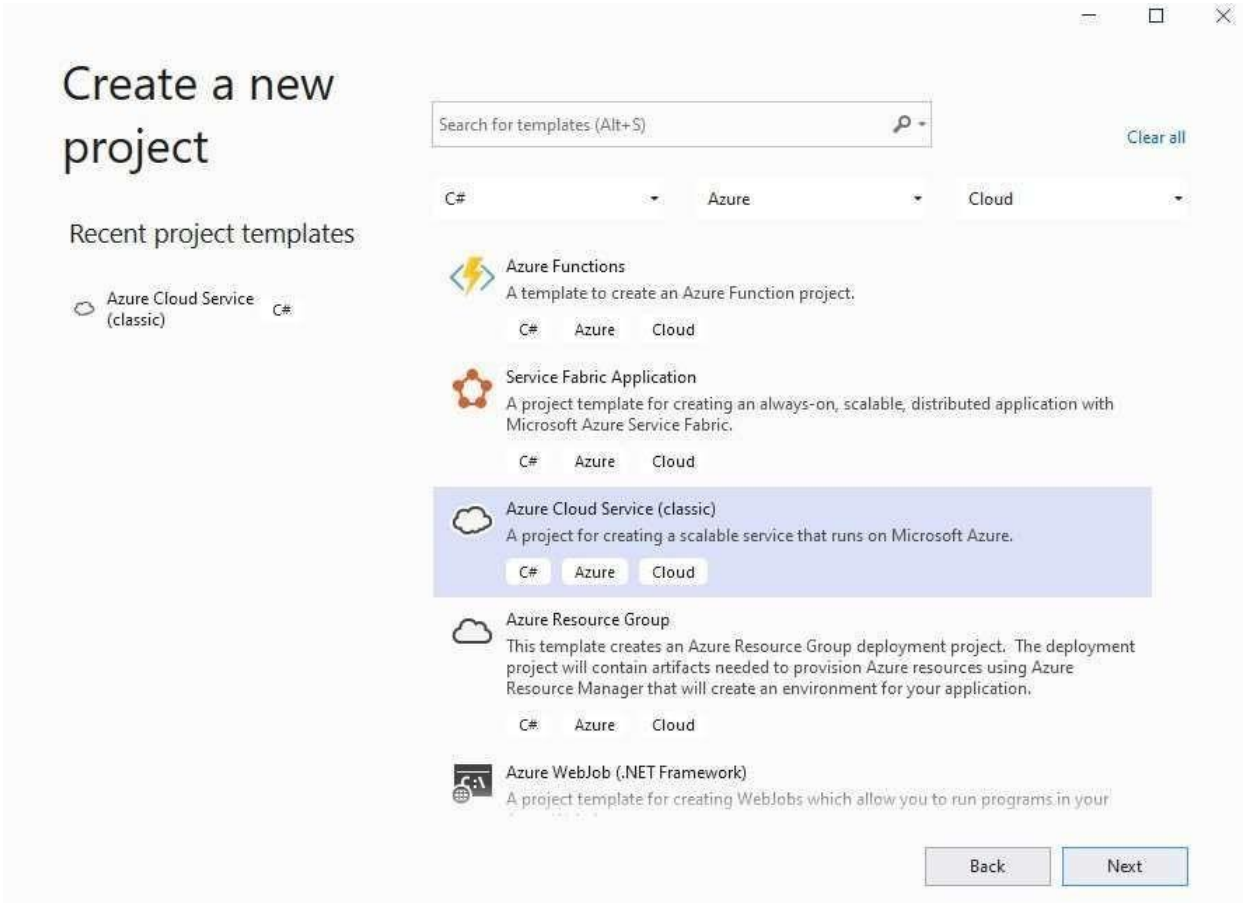
To develop an application for Windows Azure on Visual Studio 2019, install the Visual Studio 2019.

Step2:

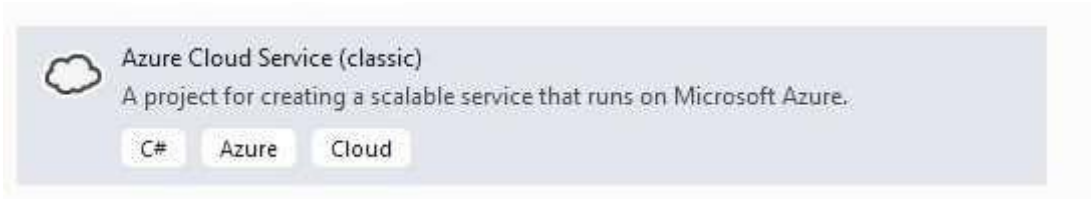
- Open Visual Studio 2019
- Click on create new project



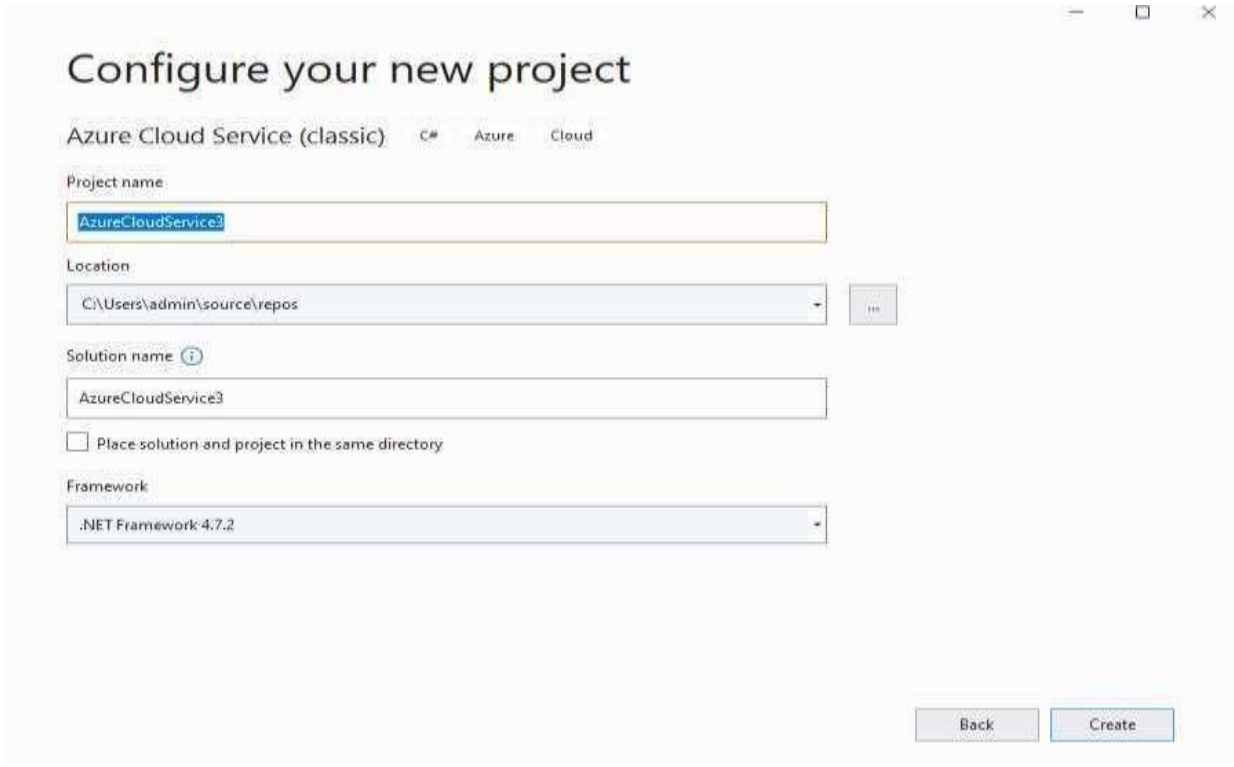
- After clicking on Create new project, a **New Project Window** will open



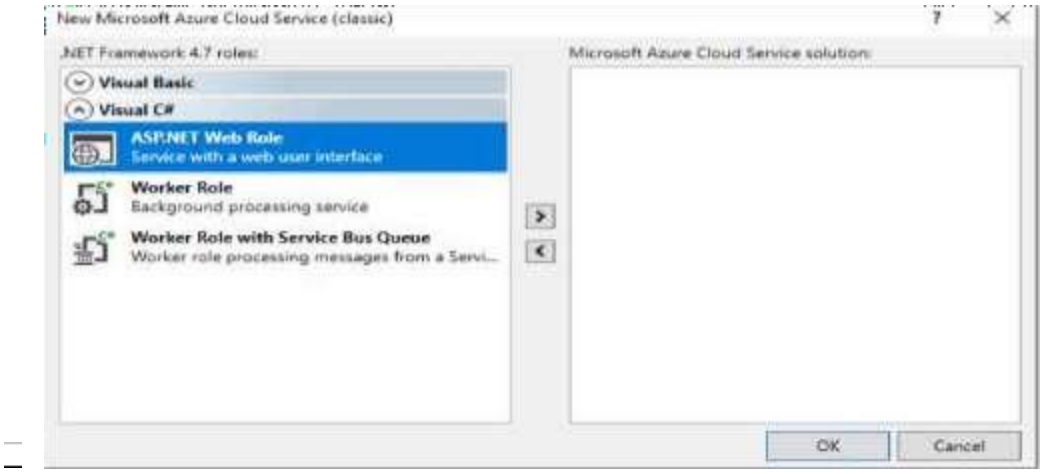
- In above window, choose **Language** as C#, **Platform** as Azure, **Project** asCloud
- Then select Azure Cloud Service(classic) option



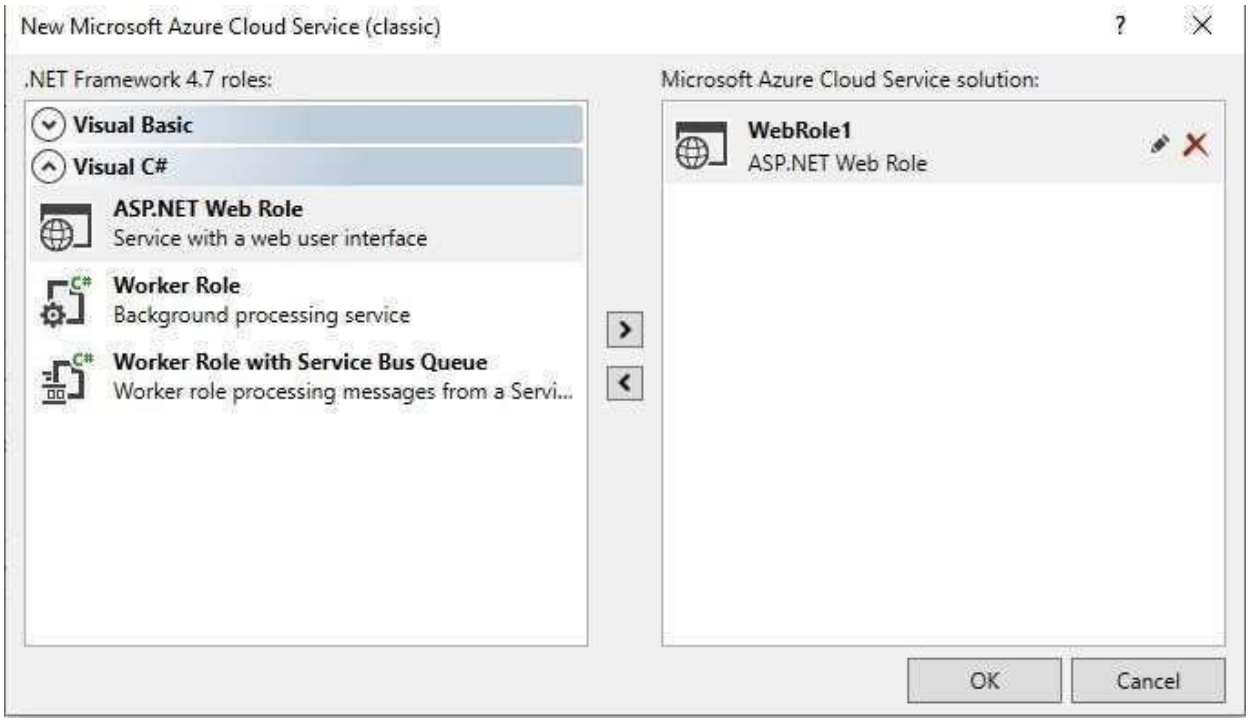
- Click Next
- Configure project window will appear.



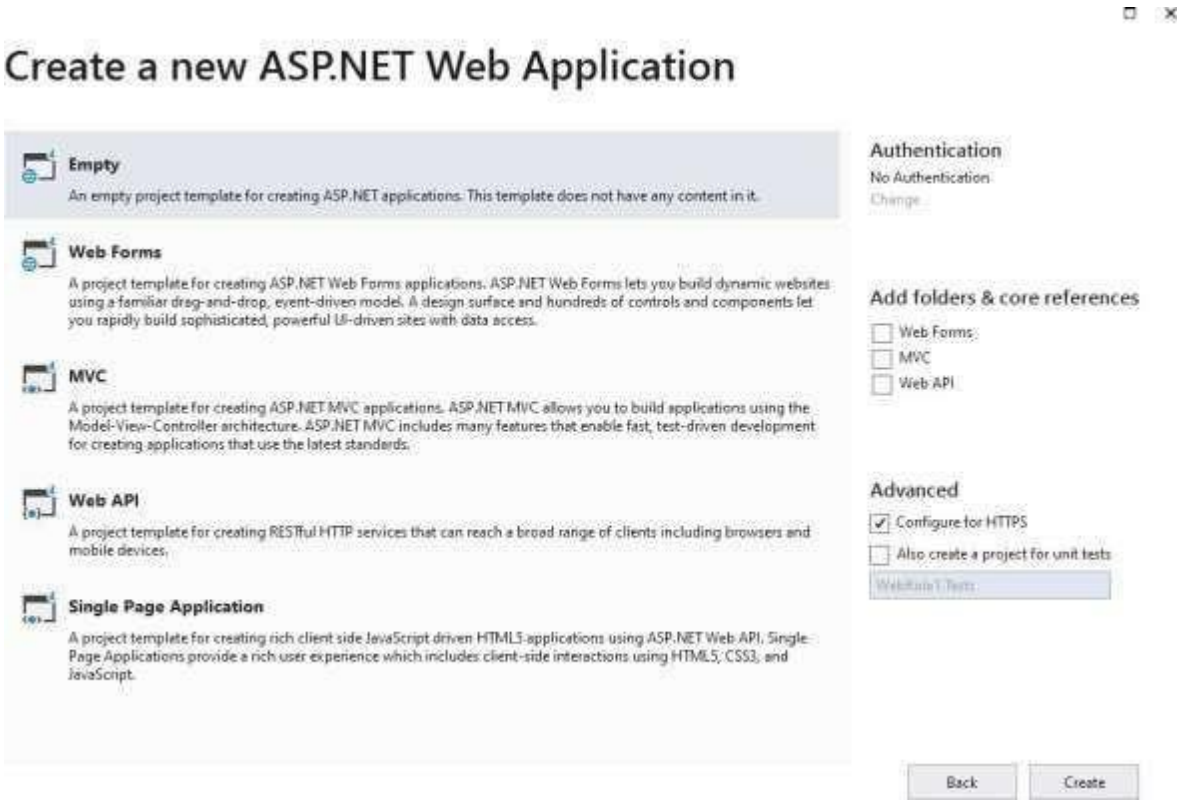
- Enter Project name(AzureCloudService3), then click on Create button.
- New Microsoft Azure Cloud Service(Classic) window will appear as below. Select Visual C# -> ASP.net Web Role.



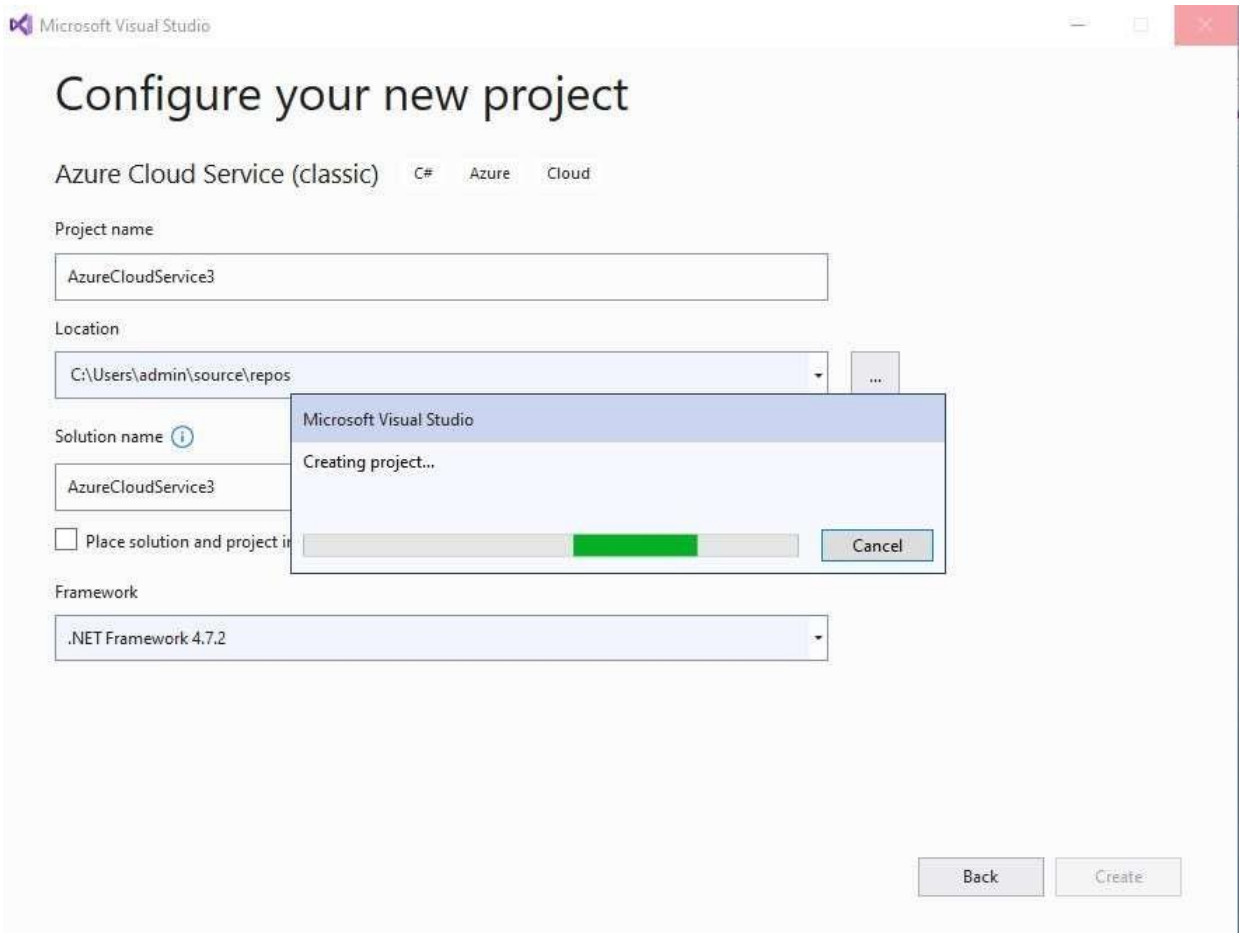
- Click on **> button for add role to solution.**
- Below window will appear.



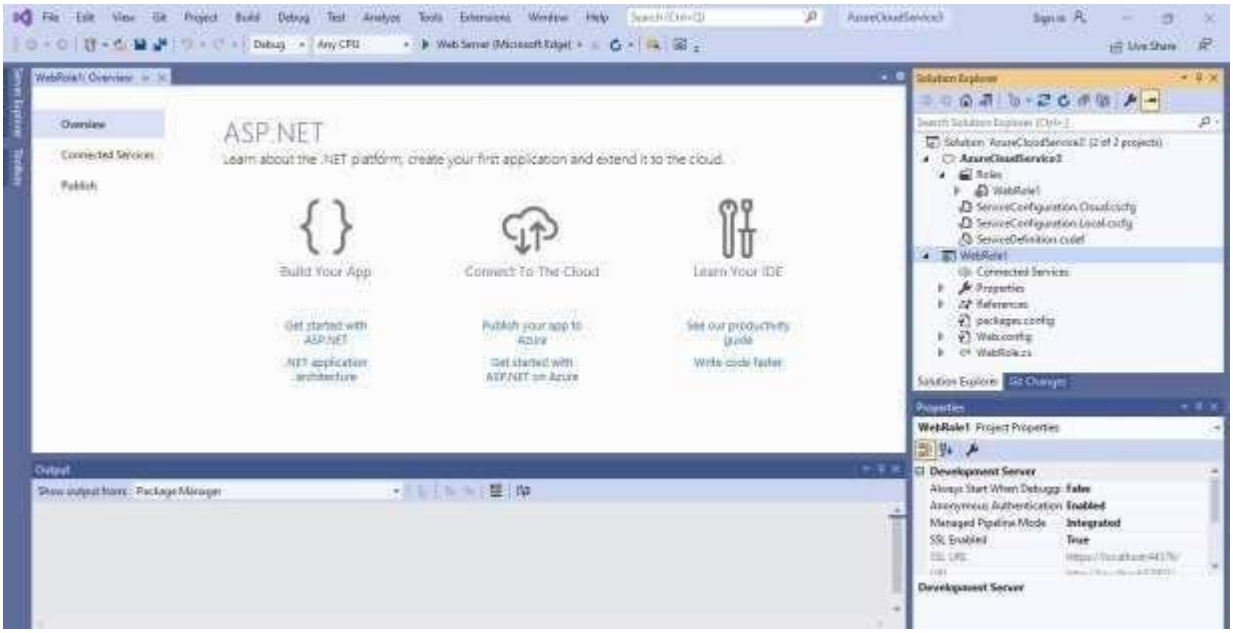
- Click on OK button.
- Below window will appear.



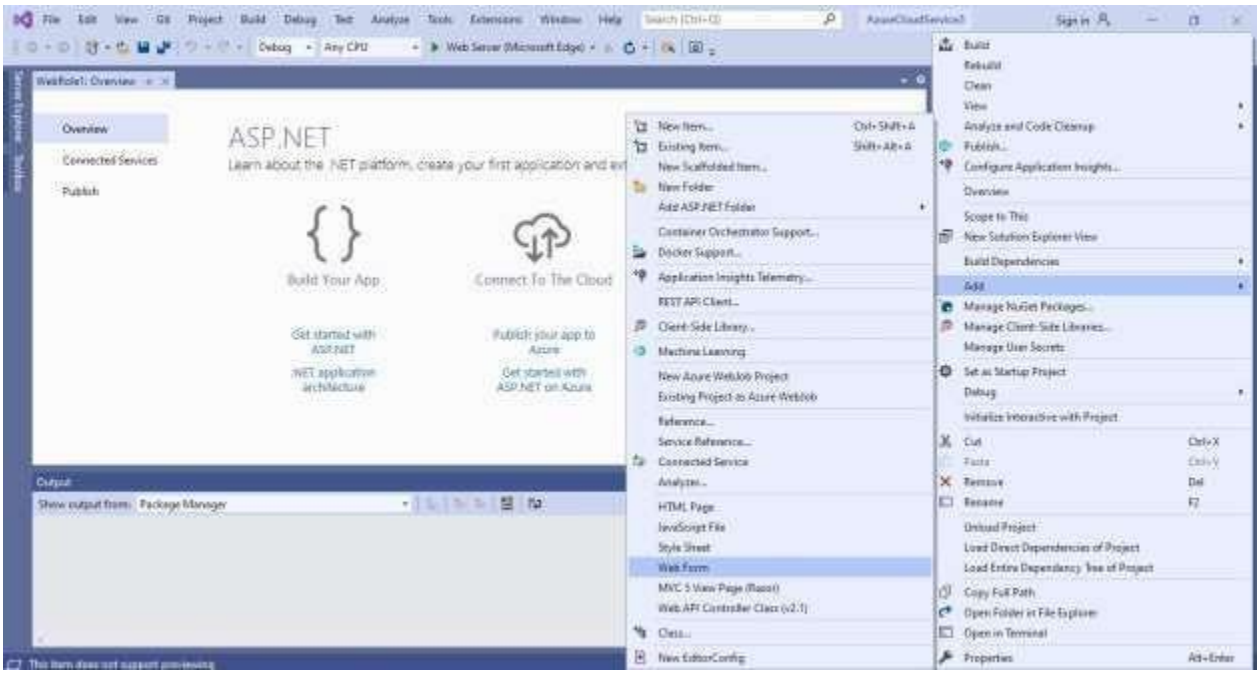
- Select Empty Option for creating empty project template and then click on Create Button. Below window will appear.



Right Click on WebRole1 in Solution Explorer Window.

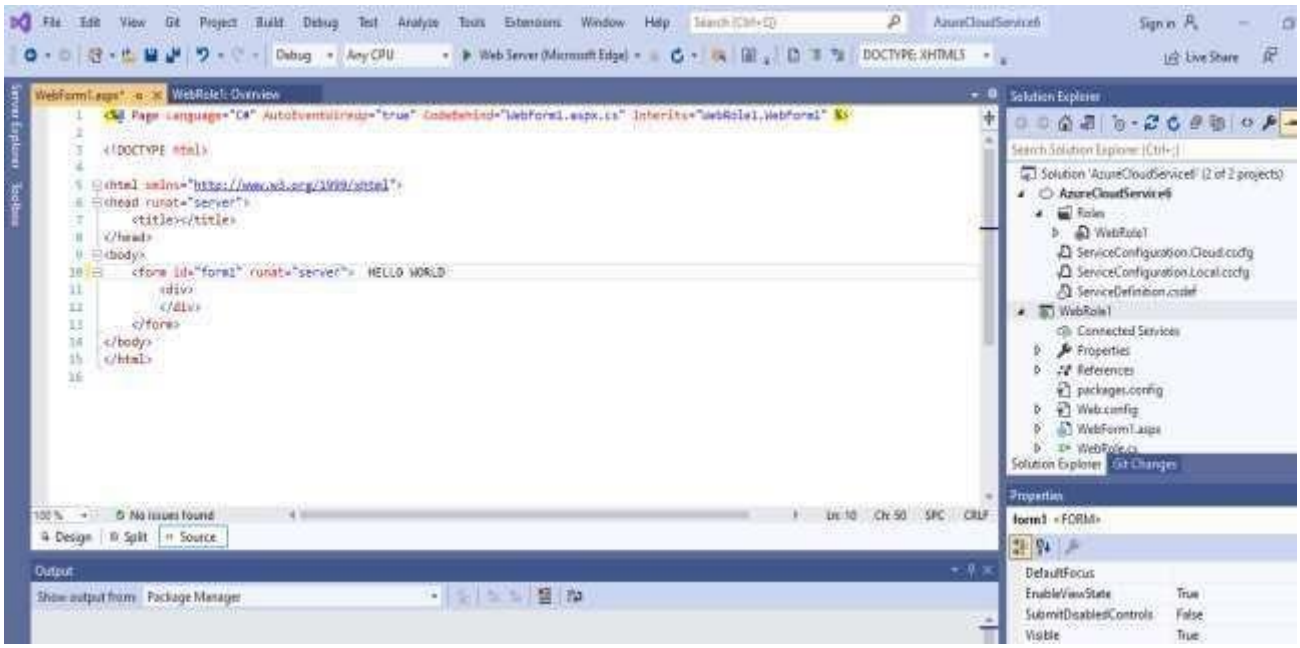


Then click on **Add Button → Select Web Form**



Give Name to Web Form

Click on OK Button. Below window will appear.



Then click on Execute Project.

Following is the output: