Page No.
Date

Assignment - II

Title :- Socket programming.

Problem statement :-
Enhance the system with the help of socket programming, use client server architecture Develop chat server.

Objective :-
- To learn basic & advanced techniques of socket based client server programming
- To understand java.net package of J2SE APIs which provides the low level communication details.
- To understand TCP/UDP client server application developement.

Outcome :-
- Design & implement socket programming in Java.
- Implement TCP/UDP client server application developement.

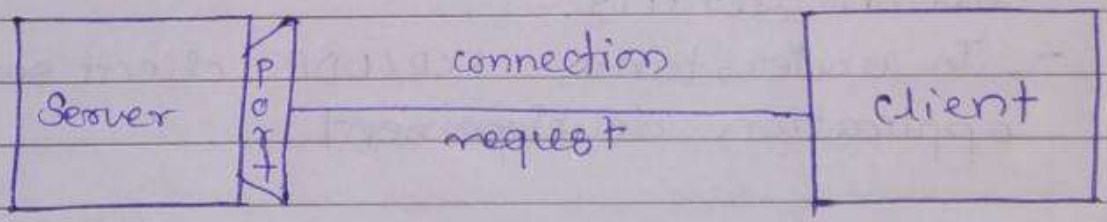
S/W & H/W requirements :-
Java SE developement kit, Fedora OS 64 bit, programming tools.

Page No.
Date / /

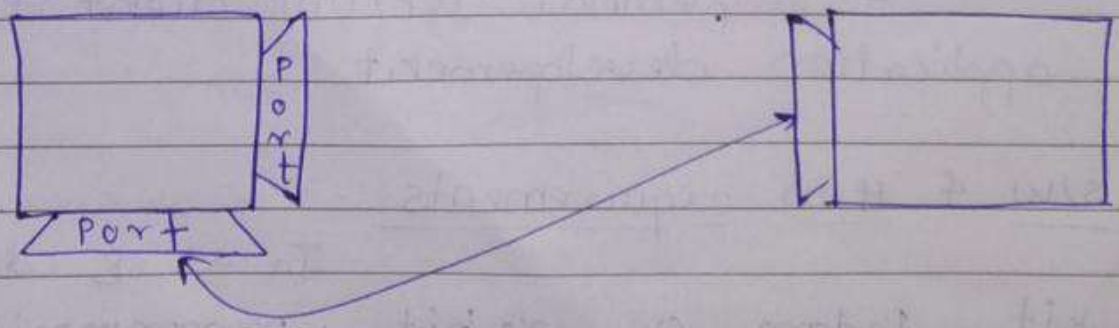
Concept related theory :-

Java creates a network application using sockets. program running on client mic makes request to platform / program running on server, involves networking services provided by transport layer. Transport layer has 2 protocols (TCP, UDP). These are port to map incoming data to particular process on computer.

A socket is a software endpoint that establishes bidirectional communication bet^n a server program & one or more client program.



client making a connection request to the server.



Session established with temparary ports used for 2-way communication.

Page No.
Date

The socket associates the server program with a specific hardware port on machine where it runs to any client program anywhere in the network with a socket associated with that same port can communicate with server program.

Steps :-

1) Open the server socket;
   ServerSocket = new ServerSocket(port);
2) Wait for the client request
   Socket client = Server.accept();
3) create I/o streams for communicating to the client

DataInputStream is = new DataInputStream(client.getInputStream());
DataOutput Stream os = new DataOutputStream(client.getOutputStream());

4) Perform communication with servers from client
   Recieve from client
   string line = is.readLine();

5) Send data to the server :-
   os.writeBytes("Hello \n");

close the socket when done.
client.close();

Page No.
Date

Applications :-

① Client-Server :-

client programming :-

To connect to other machine we need a socket connection.

command :-

Socket socket = new Socket("127.0.0.1", 5000);

"127.0.0.1" → IP address of localhost

5000 → TCP port.

- Streams are used to comminicate over socket connection.
- Socket connection closed explicitly once the communication is done.
- accept() method used to connect a client to server.

Test-Cases :-

I/P	O/P	Expected O/P	Result
2. Chat to Admin	Server: Welcome client: SomeIssue Server: Resolved client: Over	same	Success
Sign Up as Employee Manager (name: "admin")	options for Employee Manager (respective options).	same	Success

Page No.
Date

Conclusion :-

We are able to implement chat server using socket programming in Java.

```
//Server.java

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


public class Server
{
   //initialize socket and input stream
   private Socket          socket   = null;
   private ServerSocket    server   = null;
   private DataInputStream  input   = null;
   private ObjectInputStream inObj = null;
   private DataOutputStream out     = null;
   private ObjectOutputStream outObj = null;
   private DataInputStream in       = null;
  ArrayList<String> al = new ArrayList<String>();
      ArrayList ale = new ArrayList();
      ArrayList prj = new ArrayList();
   // constructor with port
   public Server(int port)
   {

      // starts server and waits for a connection
      try
      {



         server = new ServerSocket(port);
         System.out.println(" -- Server started");

         System.out.println(" -- Waiting for a client ...");


         socket = server.accept();
         System.out.println(" -- Client accepted");

         // takes input from the client socket
         in = new DataInputStream(
            new BufferedInputStream(socket.getInputStream()));

         inObj = new ObjectInputStream(socket.getInputStream());

         input  = new DataInputStream(System.in);

         // sends output to the socket
         out    = new DataOutputStream(socket.getOutputStream());

         outObj = new ObjectOutputStream(socket.getOutputStream());
```

```
int ch,ch2;
Employee e;
//Project p;
Scanner sc = new Scanner(System.in);
String nm,nm2;
Combine c;
boolean flag ;
try{
   do{
      ch = in.read();

      switch(ch)
      {
         case 1 :
            flag = false;
            nm = in.readUTF();
            for(int i=0;i<ale.size();i++)
            {
               e = (Employee)ale.get(i);
               //System.out.println("@   " + e.giveEmp());
               if(nm.equals(e.name))
               {
                  flag= true;
                  outObj.writeObject(e);
                  break;
               }
            }
            if(flag)
            {
               System.out.println(" -- Client :  " + nm + " , Just Logged In");

               do{
                  ch2 = in.read();

                  switch(ch2)
                  {

                     case 2 :
                        System.out.println("----------------------");
                        String line = "";

                        // reads message from client until "Over" is sent
                        try
                           {
                                 line = "Welcome to Admin service ";
                                 System.out.println("Server  :   " + line);
                                 out.writeUTF(line);
                           }
                            catch(IOException i)
                           {
                              System.out.println(i);
                           }
                        while(!line.equals("Over"))
                        {
```

```
                        try
                        {
                           line = in.readUTF();
                           System.out.println("Client  :   " +line);
                           al.add(line);
                           if(!line.equals("Over"))
                           {
                              System.out.print("Server    :   ");
                              line = input.readLine();
                              out.writeUTF(line);
                           }

                        }
                        catch(IOException i)
                        {
                           System.out.println(i);
                        }
                  }
                  break;

               case 3 :
                  nm2 = in.readUTF();
                  for(int i = 0;i<prj.size();i++)
                  {
                     c = (Combine)prj.get(i);
                     outObj.writeObject(c);
                  }
                  outObj.writeObject(null);
                  break;
               default :
                  if(ch2!= 0 && ch2 !=1)
                  {
                     System.out.println("Invaild Input from client");
                  }
            }
         }while(ch2 !=0);
      }
      else
      {
         e = null;
         outObj.writeObject(e);
         System.out.println(" -- Login failure for " + nm);
      }

      break;
   case 2 :
      e = (Employee)inObj.readObject();
      ale.add(e);
      System.out.println(" -- Client  :  " +e.getName() + " Signed Up Successfully ");
      System.out.println(" -- Client Details : \n");
      System.out.println(e.giveEmp());
      break;

   case 3 :
```

```
do{
   ch2 = in.readInt();

   switch(ch2)
   {
      case 1:

         Project p;
         p = (Project)inObj.readObject();
         nm = in.readUTF();
         c = new Combine(p,nm);
         prj.add(c);
         out.writeUTF("Project Created Successfully !");


         break;
      case 2:
         flag = false;
         nm = in.readUTF();
         for(int i=0;i<ale.size();i++)
         {
            e = (Employee)ale.get(i);
            if(nm.equals(e.getName()))
            {
               ale.remove(i);
               flag =true;
               out.writeUTF("Employee Removed !");
               break;
            }
         }
         if(!flag)
         {
            out.writeUTF("Employee Not Removed");
         }
         break;
      case 3:
         flag = false;
         nm = in.readUTF();

         for(int i=0;i<prj.size();i++)
         {
            c = (Combine)prj.get(i);

            if(nm.equals(c.getName()))
            {
               flag =true;
               prj.remove(i);
               out.writeUTF("Project Deleted !");
               break;
            }
         }
         if(!flag)
         {
```

```
                        out.writeUTF("Project Not Deleted !!");
                    }
                    break;

                case 5 :
                    ch2=0;
                    break;
                }

            }while(ch2!=0);


            break;

        default:
            if(ch != 0)
            {
                System.out.println(" -- Invalid Input from Client !");
            }
        }

      }while(ch != 0);
    }
    catch(IOException i)
    {
      System.out.println(i);
    }
    catch(ClassNotFoundException cnf)
    {
      System.out.println(cnf);
    }


    System.out.println("Closing connection");

    // close connection
    socket.close();
    in.close();
    input.close();
    out.close();
  }
  catch(IOException i)
  {
    System.out.println(i);
  }
 }

 public static void main(String args[])
 {
  Server server = new Server(8000);
 }
}
```

```
//Client.java


import java.net.*;
import java.io.*;
import java.util.*;
import java.util.logging.Level;
import java.util.logging.Logger;

class Client
{
   // initialize socket and input output streams
   private Socket socket           = null;
   private BufferedReader  input   = null;
   private DataOutputStream out    = null;
   private DataInputStream in      = null;
   private ObjectOutputStream outObj = null;
   private ObjectInputStream inObj = null;


   // constructor to put ip address and port
   public Client(String address, int port)
   {
      // establish a connection
      try
      {
         socket = new Socket(address, port);
         System.out.println("Connected");

         // takes input from terminal
         input  = new BufferedReader(new InputStreamReader(System.in));

         // sends output to the socket
         out    = new DataOutputStream(socket.getOutputStream());

         outObj = new ObjectOutputStream(socket.getOutputStream());

         in = new DataInputStream(
               new BufferedInputStream(socket.getInputStream()));
         inObj = new ObjectInputStream(socket.getInputStream());
      }
      catch(UnknownHostException u)
      {
         System.out.println(u);
      }
      catch(IOException i)
      {
         System.out.println(i);
      }


      int ch,ch2;
```

```
Employee e;
Combine c;
String nm,msg;
Employee response;
Scanner sc = new Scanner(System.in);
try{
   do{
      System.out.println("\n\n\t1.Login \n\t2.SignUp \n\t3.Admin \n\t0.Exit");
      System.out.print("\n\tEnter Your Choice   :   ");
      ch = sc.nextInt();
      out.write(ch);
      switch(ch)
      {
         case 1 :
            //Login
            System.out.print("\n\n\tEnter Name  :   ");
            nm = sc.nextLine();
            nm = sc.nextLine();
            out.writeUTF(nm);
            response = (Employee)inObj.readObject();

            if(response != null)
            {
               System.out.println("Login Successful !");

               do{
                  System.out.println("\n\n\t1.Profile \n\t2.Chat to Server \n\t3.All Projects \n\t0.LogOut");
                  System.out.print("Enter Your Choice   :   ");
                  ch2 = sc.nextInt();
                  out.write(ch2);
                  switch(ch2)
                  {
                     case 1 :
                        System.out.println(response.giveEmp());
                        break;
                     case 2 :
                        String line = "";
                        System.out.println("\n\t-------------------------");
                         while(!line.equals("Over"))
                         {
                               try
                               {
                                  line = in.readUTF();
                                  System.out.println("Server  :   " + line);
                                  if(!line.equals("Over"))
                                  {
                                     System.out.print("Client  :   ");
                                     line = input.readLine();
                                     out.writeUTF(line);
                                  }

                               }
                               catch(IOException i)
                               {
```

```
                        System.out.println(i);
                    }
                }
                break;
            case 3 :
                int prg;
                out.writeUTF(response.getName());
                do{
                  c =(Combine) inObj.readObject();
                  if(c != null)
                  {
                    prg = c.getProgress();
                    c.showProject(prg);
                  }
                  }while(c != null);
                break;
            default :
                if(ch2 != 0)
                {
                    System.out.println("Invalid Choice !");
                }

            }

        }while(ch2!=0);
    }
    else
    {
        System.out.println("Login Failed !");
    }
    break;
case 2 :
    //SignUp
    e = new Employee();
    e.getEmp();
    outObj.writeObject(e);
    System.out.println("SignUp Successful !");
    break;

case 3 :
    System.out.println("\n\t------ Admin Section -------");
    System.out.print("\n\tEnter UserName  :   ");
    nm = sc.nextLine();
    nm = sc.nextLine();
    if(nm.equals("admin") || nm.equals("Admin") || nm.equals("ADMIN"))
    {
        do{
            System.out.println("\n\n\t1.Create Project \n\t2.Remove Employee \n\t3.Delete Project\n\t0.Exit
");
            System.out.print("\n\tEnter your choice   :   ");
            ch2 = sc.nextInt();
            out.writeInt(ch2);
            switch(ch2)
            {
```

```
                    case 1 :
                        Project p = new Project();
                        p.getProject();
                        outObj.writeObject(p);
                        System.out.print("\n\tEnter Employee Name :   ");
                        msg = sc.nextLine();
                        msg = sc.nextLine();
                        out.writeUTF(msg);
                        msg = in.readUTF();
                        System.out.println("\n\t" + msg);
                        break;
                    case 2 :
                        System.out.print("\n\tEnter Employee Name :   ");
                        msg = sc.nextLine();
                        msg = sc.nextLine();
                        out.writeUTF(msg);
                        msg = in.readUTF();
                        System.out.println("\n\t" + msg);
                        break;
                    case 3 :
                        int prg;
                        System.out.print("\n\tEnter Employee Name :   ");
                        msg = sc.nextLine();
                        msg = sc.nextLine();
                        out.writeUTF(msg);
                        msg = in.readUTF();
                        System.out.println("\n\t" + msg);
                        break;
                    default :
                        if(ch2 !=0)
                        {
                            System.out.println("\n\tInvalid Choice");
                        }
                    }

                }while(ch2!=0);
            }
            else{
                out.writeInt(5);
            }
            break;
          default:
            if(ch!=0)
            {
                System.out.println("Invalid Input");
            }
            break;
        }

    }while(ch != 0);
}
catch(IOException i)
{
    System.out.println(i);
```

```
        } catch (ClassNotFoundException ex) {
            Logger.getLogger(Client.class.getName()).log(Level.SEVERE, null, ex);
        }




        // close the connection
        try
        {
            input.close();
            out.close();
            socket.close();
            in.close();
        }
        catch(IOException i)
        {
            System.out.println(i);
        }
    }

   public static void main(String args[])
   {
 System.out.println("Client");
 Client c = new Client("localhost" , 8000);




   }
}


//Employee.java


import java.io.Serializable;
import java.util.Scanner;


public class Employee implements Serializable {
    String name;
    int exp;
    int id;
    String desig;
    String email;
    float sal;
    static int st_id = 0;
    Project prj;

    public Employee(){
        name="";
```

```
    exp=0;
    desig="";
    email="";
    sal = exp<=5 ? 30000 : 50000;
    id=st_id;
    st_id++;
    prj = null;
}

public Employee(String nm , int ex , String des ,String mail,  Project p){
   name = nm;
   exp=ex;
   desig = des;
   email = mail;
   sal = exp<=5 ? 30000 : 50000;
   id = st_id;
   st_id++;
   prj = null;
}

public Employee(Employee e)
{
    name = e.name;
    exp = e.exp;
    desig = e.desig;
    email = e.email;
    sal = exp<=5 ? 30000 : 50000;
    prj = e.prj;
    id = st_id;
    st_id++;
}

public void showProject(int prg)
{
        prj.showPrj(prg);

}

public Project givePrj()
{
    return prj;
}

public void setPrj(Project p)
{
    prj = p;
}

public String getName()
{
    return name;
}

public String getDesig()
```

```
    {
        return desig;
    }

    public String giveEmp()
    {
        String msg;
        msg = "------------------------------"  +
              "\n\tName        :   " + name +
              "\n\tExpierence   :   " + exp +
              "\n\tDesignation  :   " + desig +
              "\n\tContact      :   " + email +
              "\n\tSalary       :   Rs. " + sal   +
              "\n\n------------------------------";
        return msg;
    }

    public void setPrg(int prg)
    {
        //if(prj != null)
            prj.setProg(prg);
        //else
        //    System.out.println("No Projects Available for Employee  :   " + name);
    }

    public void getEmp()
    {
        Scanner sc = new Scanner(System.in);
        System.out.print("\n\n\tEnter Name          :   ");
        name = sc.nextLine();
        System.out.print("\n\tEnter Experience    :   ");
        exp = sc.nextInt();
        System.out.print("\n\tEnter Designation   :   ");
        desig = sc.nextLine();
        desig = sc.nextLine();
        System.out.print("\n\tEnter Email         :   ");
        email = sc.nextLine();
        sal = exp<=5 ? 30000 : 50000;

    }

    public void showEmp()
    {
        System.out.println("----------------------------");
        System.out.println("\tName          :   " + name);
        System.out.println("\tExpierence    :   " + exp);
        System.out.println("\tDesignation   :   " + desig);
        System.out.println("\tContact       :   " + email);
    }

    public static void main(String[] args) {

    }
}
```

```
//Project.java


import java.io.Serializable;
import java.util.ArrayList;
import java.util.Scanner;


public class Project implements Serializable{
   String prjTaskName;
   String prjTaskDetail;
   int d,m,y;
   int progress;

   public Project()
   {
      prjTaskName = "";
      prjTaskDetail = "";
      d=1;m=1;y=2020;
      progress = 0;

   }

   public Project(Project pr)
   {
      prjTaskName = pr.prjTaskName;
      prjTaskDetail = pr.prjTaskDetail;
      d = pr.d;
      m = pr.m;
      y = pr.y;
      progress = pr.progress;
   }

   public Project(Project pr , int p)
   {
      prjTaskName = pr.prjTaskName;
      prjTaskDetail = pr.prjTaskDetail;
      d = pr.d;
      m = pr.m;
      y = pr.y;
      progress = p;
   }

   public int getProg()
   {
      return progress;
   }

   public void setProg(int p)
   {
      progress = p;
   }
```

```
    public String prjName()
    {
        return prjTaskName;
    }

    public void showPrj(int prg)
    {
        System.out.println("\n\tProject Task Name   :   " + prjTaskName);
        System.out.println("\n\tProject Task Detail :   " + prjTaskDetail);
        System.out.println("\n\tProject Deadline    :   " + d + "/" +m + "/" +y);
    }

    public void getProject()
    {
        Scanner sc  = new Scanner(System.in);

        System.out.print("\n\tEnter Task Name  :   ");
        prjTaskName = sc.nextLine();
        System.out.print("\n\tEnter Project Task Details   :   ");
        prjTaskDetail = sc.nextLine();

        System.out.println("\n\n\tEnter Deadline    :   ");
        System.out.print("\n\t\tDay :   ");
        d = sc.nextInt();
        System.out.print("\n\t\tMonth :   ");
        m = sc.nextInt();
        System.out.print("\n\t\tYear  :   ");
        y = sc.nextInt();
    }

}


//Combine.java


import java.io.Serializable;

public class Combine implements Serializable{
    String name;
    int progress;
    Project p;
    public Combine(Project pr,String nm){
        p = new Project(pr);
        name = nm;
    }

    public void setProgress(int pr){
        progress = pr;
    }
    public int getProgress()
    {
        return progress;
```

```
    }


    public void updateProg(int progress)
    {
       p.setProg(progress);
    }

    public void showProject(int prg)
    {
       System.out.println("\n\n\tEmployee  :   " + name);
       System.out.println("\t--------------------------------");
       if(p == null)
       {
          System.out.println("\n\tNo Projects ");
       }
       else
       {
          p.showPrj(prg);
       }
    }


    public int getPrg()
    {
       return p.getProg();
    }

    public String getName()
    {
       return name;
    }
}
```

//Server output

```
run:
 -- Server started
 -- Waiting for a client ...
 -- Client accepted
 -- Client  :  rajat Signed Up Successfully
 -- Client Details :

------------------------------
 Name        :   rajat
 Expierence   :   7
 Designation  :   software engineer
 Contact      :   rajat@g.com
 Salary       :   Rs. 50000.0

------------------------------
 -- Client  :  mohit Signed Up Successfully
 -- Client Details :

------------------------------
 Name        :   mohit
 Expierence   :   4
 Designation  :   jr.software engineer
 Contact      :   mohit@g.com
 Salary       :   Rs. 30000.0

------------------------------
 -- Client  :  pritesh Signed Up Successfully
 -- Client Details :

------------------------------
 Name        :   pritesh
 Expierence   :   1
 Designation  :   jr.software engineer
 Contact      :   pritesh@g.com
 Salary       :   Rs. 30000.0

------------------------------
 -- Client :  rajat , Just Logged In
----------------------
Server  :   Welcome to Admin service
Client  :   I had issues
Server    :   I will resolve them
Client  :   Over
 -- Client :  rajat , Just Logged In
 -- Client :  mohit , Just Logged In
 -- Client :  mohit , Just Logged In
Closing connection
```

// Client output

```
run:
```

Client
Connected

1.Login
2.SignUp
3.Employee Manager
0.Exit

Enter Your Choice : 2

Enter Name : rajat

Enter Experience : 7

Enter Designation : software engineer

Enter Email : rajat@g.com
SignUp Successful !

1.Login
2.SignUp
3.Employee Manager
0.Exit

Enter Your Choice : 2

Enter Name : mohit

Enter Experience : 4

Enter Designation : jr.software engineer

Enter Email : mohit@g.com
SignUp Successful !

1.Login
2.SignUp
3.Employee Manager
0.Exit

Enter Your Choice : 2

Enter Name : pritesh

Enter Experience : 1

Enter Designation : jr.software engineer

```
 Enter Email        :   pritesh@g.com
SignUp Successful !


 1.Login
 2.SignUp
 3.Employee Manager
 0.Exit

 Enter Your Choice   :   1


 Enter Name  :   rajat
Login Successful !


 1.Profile
 2.Help
 3.Projects
 0.LogOut
Enter Your Choice   :   1
------------------------------
 Name        :   rajat
 Expierence   :   7
 Designation  :   software engineer
 Contact      :   rajat@g.com
 Salary       :   Rs. 50000.0

------------------------------


 1.Profile
 2.Help
 3.Projects
 0.LogOut
Enter Your Choice   :   2

 -------------------------
Server  :   Welcome to Admin service
Client  :   I had issues
Server  :   I will resolve them
Client  :   Over


 1.Profile
 2.Help
 3.Projects
 0.LogOut
Enter Your Choice   :   3


 1.Profile
 2.Help
 3.Projects
```

0.LogOut
Enter Your Choice : 0

1.Login
2.SignUp
3.Employee Manager
0.Exit

Enter Your Choice : 3

------ Employee Manager Section -------

Enter UserName : admin

1.Create Project
2.Remove Employee
3.Delete Project
0.Exit

Enter your choice : 1

Enter Task Name : web app

Enter Project Task Details : build a web app

Enter Deadline :

Day : 30

Month : 9

Year : 2020

Enter Employee Name : rajat

Project Created Successfully !

1.Create Project
2.Remove Employee
3.Delete Project
0.Exit

Enter your choice : 1

Enter Task Name : ML app

Enter Project Task Details : build a ML app

Enter Deadline :

Day : 30

Month : 10

Year : 2020

Enter Employee Name : mohit

Project Created Successfully !

1.Create Project
2.Remove Employee
3.Delete Project
0.Exit

Enter your choice : 2

Enter Employee Name : pritesh

Employee Removed !

1.Create Project
2.Remove Employee
3.Delete Project
0.Exit

Enter your choice : 0

1.Login
2.SignUp
3.Employee Manager
0.Exit

Enter Your Choice : 1

Enter Name : rajat
Login Successful !

1.Profile
2.Help
3.Projects
0.LogOut
Enter Your Choice : 3

Employee : rajat

Project Task Name : web app

Project Task Detail : build a web app

Project Deadline : 30/9/2020

1.Profile
2.Help
3.Projects
0.LogOut
Enter Your Choice : 0

1.Login
2.SignUp
3.Employee Manager
0.Exit

Enter Your Choice : 1

Enter Name : mohit
Login Successful !

1.Profile
2.Help
3.Projects
0.LogOut
Enter Your Choice : 3

Employee : mohit

Project Task Name : ML app

Project Task Detail : build a ML app

Project Deadline : 30/10/2020

1.Profile
2.Help
3.Projects
0.LogOut
Enter Your Choice : 0

1.Login
2.SignUp
3.Employee Manager
0.Exit

Enter Your Choice : 3

------ Employee Manager Section -------

Enter UserName : admin

1.Create Project
2.Remove Employee
3.Delete Project
0.Exit

Enter your choice : 3

Enter Employee Name : mohit

Project Deleted !

1.Create Project
2.Remove Employee
3.Delete Project
0.Exit

Enter your choice : 0

1.Login
2.SignUp
3.Employee Manager
0.Exit

Enter Your Choice : 1

Enter Name : mohit
Login Successful !

1.Profile
2.Help
3.Projects
0.LogOut
Enter Your Choice : 3

1.Profile
2.Help
3.Projects
0.LogOut
Enter Your Choice : 0

1.Login
2.SignUp
3.Employee Manager
0.Exit

Enter Your Choice : 0