NOM:……Prénom:…..…… Matricule:……..…

Examen Final (CAGL M2 IL – 26/01/2023)

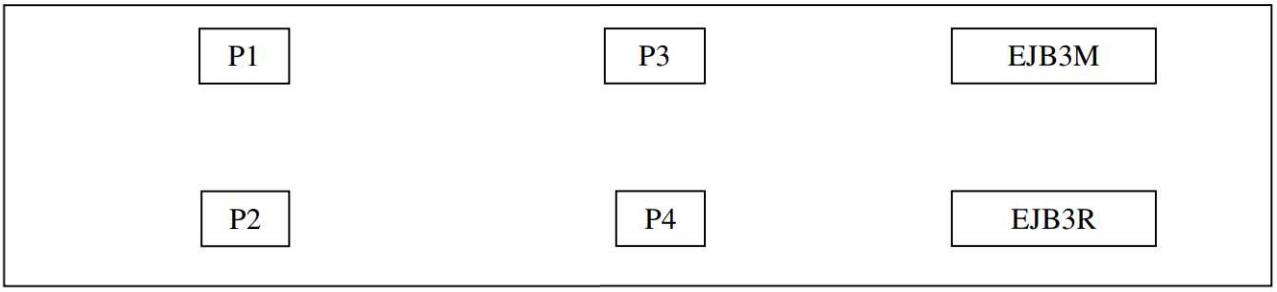
Exercice :

Nous considérons une application (AppS) composée de six programmes Java (P1, P2, P3, P4, EJB3M, EJB3R) distribués respectivement sur six sites (M1, M2, M3, M4, M5, M6). Les codes Java des programmes P1, P2, P3, P4, EJB3R et EJB3M sont donnés par (voir feuille 2) :

1. Trouver les erreurs dans les programmes donnés.

2. Donner l'architecture, le mode de communication et le fonctionnement de cette application (AppS)

Architecture et mode de communication



Fonctionnement (une réponse fausse élimine une réponse correcte)

Programme source	Message envoyé	Programme destination
P1		
P1		
P1		
P2		
P2		
P2		
P3		
P3		
P3		
P4		
P4		
P4		
P4		
EJB3R		
EJB3M		

3. Compléter le tableau ci-dessous

Élément	Type	Rôle
Architecture d'AppS		
Composant EJB3M		
Composant EJB3R		

4. Donner les résultats d'exécution des programmes P1, P2, P3, P4, EJB3R et EJB3M (*Ordre d'exécution: EJB3R, EJB3M, P4, P3, P2, P1*).

P1	***P2***	***P3***	***P4***
RBean	***IR***	***MBean***	***IM***

Questions :

1. Expliquer la différence entre la Java Bean et Entreprise Java Bean.

2. Expliquer la différence entre la Servlet et JSP.

3. Donner quatre méthodes de la classe EntityManager.

4. Donner quatre annotations utilisées dans un composant de type EJB3 Entité (sauf les annotations de relation).

5. Donner deux solutions permettant la description du mapping entre un Bean entité et une table dans la BD.

6. Citer les trois éléments qui permettent de créer un composant EJB3.

7. Les applications sont conçues généralement selon deux modèles. Donner ces modèles.

Bon courage.

```
// les imports
public class P1 {
public static void main(String[] args) {
try {
Socket sc = new Socket("localhost",2004);
ObjectOutputStream out = new ObjectOutputStream(sc.getOutputStream());
out.writeObject("USTHB");
ObjectInputStream in = new ObjectInputStream(sc.getInputStream());
String ch = (String) in.readObject();
System.out.println(ch);
DatagramSocket c1 = new DatagramSocket();
byte[] ef = new byte[20];ef=(ch).getBytes();
DatagramPacket p = new DatagramPacket(ef, ef.length, InetAddress.getByName("localhost"), 9876);
c1.send(p);
DatagramPacket q = new DatagramPacket(ef, ef.length);
c1.receive(q);
System.out.println(new String(q.getData()));
sc.close();c1.close();in.close();
}catch(Exception e) {System.out.println(e.toString());}}}
```

```
// les imports
public class P2 {
public static void main(String[] args) {
try {
DatagramSocket ss = new DatagramSocket(9876);
byte[] dr = new byte[20];
DatagramPacket p = new DatagramPacket(dr, dr.length);
ss.receive(p); String ch = new String(p.getData());
Registry r = LocateRegistry.getRegistry("localhost", 1099);
IP3 i = (IP3) r.lookup("py");
int j = i.me(15, 25);
System.out.println(j);
System.out.println(ch);
dr = "Ack".getBytes();
DatagramPacket q = new DatagramPacket(dr, dr.length, InetAddress.getByName("localhost"), p.getPort());
ss.send(q);
ss.close();
}catch(Exception e) {System.out.println(e.toString());}}}
```

```
// les imports
@Remote
public interface IP3
{public int me(int x, int y) throws RemoteException;}

public class P3 extends java.rmi.server.UnicastRemoteObject{
public P3() throws RemoteException {System.out.println("P3...");}
public static void main(String[] args) {
try {    P3 p3 = new P3();
Registry r = LocateRegistry.createRegistry(1099);
r.rebind("psy", p3);
}catch(Exception e) {System.out.println(e.toString());}}
public int me(int x, int y) throws RemoteException {System.out.println("P3...");return x+y;}}
```

```
// les imports
public class P4 {
public static void main(String[] args) {
try {
ServerSocket s = new ServerSocket(2004);
Socket c = s.accept();
ObjectInputStream in = new ObjectInputStream(c.getInputStream());
String ch = (String) in.readObject();
System.out.println(ch);
```

```
Properties props = new Properties();
props.put("java.naming.factory.url.pkgs","org.jboss.ejb.client.naming");
props.put("jboss.naming.client.ejb.context", true);
InitialContext context = new InitialContext(props);
IM x = (IM)context.lookup("ejb:/EJB3M/MBean!Pack.IM?stateful");
IR y = (IR)context.lookup("ejb:/EJB3R/RBean!Pack.IR");
x.m1(3);x.m2(3);
x.T0();
int sm = y.me(20);
System.out.println(sm);
ObjectOutputStream out = new ObjectOutputStream(c.getOutputStream());
out.writeObject("Univ "+ch+" "+Integer.toString(y.me(15)));
s.close();out.close();
}catch(Exception e) {System.out.println(e.toString());}}}
```

```
import javax.ejb.Remote;
@Remote
public interface IR {int me(int a);}

import javax.ejb.Stateless;
@Stateless
public class RBean implements IR {
public RBean() {}
public int me(int a) {
        System.out.println("Rbean...");
        int s=0;
        for (int i=1;i<=a/2; i++)
        {if ((a%i==0)&&(i%2==0)) s++;}
        return s;}}
```

```
import javax.ejb.Remote;
@Remote
public interface IM {
        void me(int a);
        void m1(int a);
        void m2(int b);
        void T0();}

import javax.ejb.Stateful;
@Stateful
public class MBean implements IM {
int A;   int B;
public MBean()          {A=0; B=0;}
public void m1(int a) {A=a;}
public void m2(int b) {B=b;}
public void T0() {
System.out.println("Mbean...");
int T=1;
for (int i=1; i<=B;i++) T=T*A; T=T*2;
System.out.println(T);}}
```

NOM:……Prénom:…… Matricule:……..…

Examen Final (CAGL M2 IL – 26/01/2023)

Exercice :

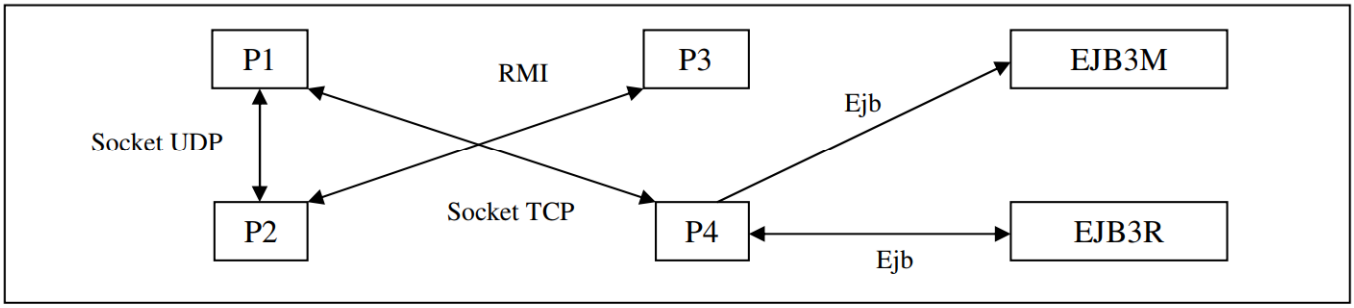
Nous considérons une application (AppS) composée de six programmes Java (P1, P2, P3, P4, EJB3M, EJB3R) distribués respectivement sur six sites (M1, M2, M3, M4, M5, M6). Les codes Java des programmes P1, P2, P3, P4, EJB3R et EJB3M sont donnés par (voir feuille 2) :

1. Trouver les erreurs dans les programmes donnés. (Chaque erreur trouvé 0.5 points – Total : 3.00 points)

Dans IP3 : supprimer @Remote, il manque extends Remote Dans P3 : il manque implements IP3, py au lieu de psy (ou inversement) Dans MBean : il manque l'implémentation de la méthode me (sinon on la supprime de l'interface IM). Dans les deux EJBs il manque le Package.

2. Donner l'architecture, le mode de communication et le fonctionnement de cette application (AppS)

Architecture et mode de communication (02 points, chaque erreur -0.5 points)



Fonctionnement (une réponse fausse élimine une réponse correcte) (chaque ligne correcte 0.25 point. L'ordre n'est pas important – Total : 4.25 points)

Programme source	Message envoyé	Programme destination
P1	USTHB	P4
P1	Univ USTHB 0	P2
P1		
P2	15	P3
P2	25	P3
P2	Ackv USTHB 0 (Ou bien Ack)	P1
P3	40	P2
P3		
P3		
P4	3	EJB3M
P4	3	EJB3M
P4	20	EJB3R
P4	15	EJB3R
EJB3R	3	P4
EJB3M		

EJB3R	0	P4
P4	Univ USTHB 0	P1

3. Compléter le tableau ci-dessous (0.25 point * 6 – Total 1.50 points)

Élément	Type	Rôle
Architecture d'AppS	Distribuée	Calcule arithmétique ou toute autre réponse logique
Composant EJB3M	Session avec état	Calculer le double de A power B
Composant EJB3R	Session sans état	Calculer le nombre des diviseurs pairs d'un entier

4. Donner les résultats d'exécution des programmes P1, P2, P3, P4, EJB3R et EJB3M (*Ordre d'exécution: EJB3R, EJB3M, P4, P3, P2, P1*). Chaque réponse correcte (sauf IR et IM qui sont toujours vide) 0.5 point (Total 3.00 points)

P1	***P2***	***P3***	***P4***
Univ USTHB 0 Ackv USTHB 0 (Ou bien Ack)	40 Univ USTHB 0	P3... P3...	USTHB 3
RBean	***IR***	***MBean***	***IM***
Rbean... Rbean...		Mbean... 54	

Questions : (chaque question 1.00 points – Total 7.00 points)

1. Expliquer la différence entre la Java Bean et Entreprise Java Bean.

Voir support de cours

2. Expliquer la différence entre la Servlet et JSP.

Voir support de cours

3. Donner quatre méthodes de la classe EntityManager.

Voir support de cours			

4. Donner quatre annotations utilisées dans un composant de type EJB3 Entité (sauf les annotations de relation).

Voir support de cours			

5. Donner deux solutions permettant la description du mapping entre un Bean entité et une table dans la BD.

Voir support de cours	

6. Citer les trois éléments qui permettent de créer un composant EJB3.

Voir support de cours		

7. Les applications sont conçues généralement selon deux modèles. Donner ces modèles.

Voir support de cours	

Bon courage.