MYUSERAPP.JAVA

```
package ed.jpa;


import java.util.Scanner;

public class MyuserApp {

   private MyuserDB mydb;
   public MyuserApp() {
      mydb = new MyuserDB();
   }
   public static void main(String[] args) {
      MyuserApp client = new MyuserApp();

      System.out.println("Welcome to the database");
      while (true) {
         System.out.println("1. Check current record\n" +
                     "2. Create a record\n" +
                     "3. Update a user\n" +
                     "4. Delete a record\n" +
                     "5. Exit");
         Scanner myObj = new Scanner(System.in);
         int choice = Integer.parseInt(myObj.nextLine());

         if (choice == 1) {
            Scanner myObj2 = new Scanner(System.in);
            System.out.println("Enter User ID");
            String userID = myObj2.nextLine();

            MyuserDTO user = client.getRecord(userID);
            System.out.println( "Name: " + user.getName() + "\n" +
                        "Password: " + user.getPassword() + "\n" +
                        "Email: " + user.getEmail() + "\n" +
                        "Phone: " + user.getPhone() + "\n" +
                        "Address: " + user.getAddress() + "\n" +
                        "SecQn: " + user.getSecQn() + "\n" +
                        "SecAns: " + user.getSecAns() + "\n"
                        );

         } else if (choice == 2) {
            Scanner s = new Scanner(System.in);

            System.out.println("Enter UserID");
            String userID = s.nextLine();
```

```
            System.out.println("Name: ");
            String name = s.nextLine();

            System.out.println("Password: ");
            String password = s.nextLine();

            System.out.println("Email: ");
            String email = s.nextLine();

            System.out.println("Phone: ");
            String phone = s.nextLine();

            System.out.println("Address: ");
            String address = s.nextLine();

            System.out.println("Secqn:");
            String secqn = s.nextLine();

            System.out.println("Secans:");
            String secans = s.nextLine();

            MyuserDTO newUser = new MyuserDTO(userID, name, password, email, phone,
address, secqn, secans);
            client.createRecord(newUser);
        } else if (choice == 3) {
            Scanner s = new Scanner(System.in);

            System.out.println("Enter UserID to Update");
            String userID = s.nextLine();

            System.out.println("Name: ");
            String name = s.nextLine();

            System.out.println("Password: ");
            String password = s.nextLine();

            System.out.println("Email: ");
            String email = s.nextLine();

            System.out.println("Phone: ");
            String phone = s.nextLine();

            System.out.println("Address: ");
            String address = s.nextLine();

            System.out.println("Secqn:");
```

```
            String secqn = s.nextLine();

            System.out.println("Secans:");
            String secans = s.nextLine();

            MyuserDTO updatedUser = new MyuserDTO(userID, name, password, email,
phone, address, secqn, secans);
            client.updateRecord(updatedUser);
         } else if (choice == 4) {

            Scanner s = new Scanner(System.in);
            System.out.println("Enter UserID to Delete");
            String userID = s.nextLine();
            client.deleteRecord(userID);

         } else if (choice == 5) {
            System.exit(1);
         } else {
            System.out.println("Invalid Selection");
         }
      }

   }
   public void showCreateResult(boolean result, MyuserDTO myuserDTO) {
      if (result) {
         System.out.println("Record with primary key " + myuserDTO.getUserid()
               + " has been created in the database table.");
      } else {
         System.out.println("Record with primary key " + myuserDTO.getUserid()
               + " could not be created in the database table!");
      }
   }
   public boolean createRecord(MyuserDTO myuserDTO) {
      return mydb.createRecord(myuserDTO);
   }

   public MyuserDTO getRecord(String userId) {
      return mydb.getRecord(userId);
   }

   public boolean updateRecord(MyuserDTO myuserDTO) {
      return mydb.updateRecord(myuserDTO);
   }

   public boolean deleteRecord(String userId) {
      return mydb.deleteRecord(userId);
   }
```

}

MYUSERDB.JAVA

```
package ed.jpa;

import entity.Myuser;

import javax.persistence.EntityManager;
import javax.persistence.EntityManagerFactory;
import javax.persistence.Persistence;

public class MyuserDB {

   private EntityManager em = null;
   public MyuserDB() {
      // using default persistence unit defined in the persistence.xml file
      EntityManagerFactory emf = Persistence.createEntityManagerFactory("ED-EntityPU");
      em = emf.createEntityManager();
   }
   public EntityManager getEntityManger() {
      return em;
   }

   public Myuser findMyuser(String userid) {
      return em.find(Myuser.class, userid);
   }
   public boolean createMyuser(Myuser myuser) throws Exception {
      try {
         if (findMyuser(myuser.getUserid()) != null) {
            // myuser exists already
            return false;
         }
         // myuser does not exist in database
         em.getTransaction().begin();
         em.persist(myuser); // to add an object to database
         em.getTransaction().commit();
         return true;
      } catch (Exception ex) {
         throw ex;
      }
   }

   public boolean createRecord(MyuserDTO myuserDTO) {
      Myuser myuser = this.myDTOtoDAO(myuserDTO);
      boolean result = false;
      try {
```

```
        result = this.createMyuser(myuser);
    } catch (Exception ex) {
    }
    return result;
}

private Myuser myDTOtoDAO(MyuserDTO myDTO) {
    Myuser myuser = new Myuser();
    myuser.setUserid(myDTO.getUserid());
    myuser.setName(myDTO.getName());
    myuser.setPassword(myDTO.getPassword());
    myuser.setEmail(myDTO.getEmail());
    myuser.setPhone(myDTO.getPhone());
    myuser.setAddress(myDTO.getAddress());
    myuser.setSecqn(myDTO.getSecQn());
    myuser.setSecans(myDTO.getSecAns());
    return myuser;
}


public MyuserDTO getRecord(String userId) {

    Myuser myuser = new Myuser();
    myuser.setUserid(userId);

    em.getTransaction().begin();
    Myuser user = em.find(Myuser.class, userId);
    em.getTransaction().commit();
    MyuserDTO myuserDTO = new MyuserDTO(user.getUserid(), user.getName(),
            user.getPassword(), user.getEmail(), user.getPhone(),
            user.getAddress(), user.getSecans(), user.getSecqn());
    return myuserDTO;

}

public boolean updateRecord(MyuserDTO myuserDTO) {
    MyuserDTO foundUser = getRecord(myuserDTO.getUserid());
    if (foundUser == null) {
        return false;
    }
    Myuser myuser = this.myDTOtoDAO(myuserDTO);
    System.out.println("MERGING :" + myuserDTO.getName());
    em.getTransaction().begin();
    em.merge(myuser);
    em.getTransaction().commit();

    return true;
```

```
    }

    public boolean deleteRecord(String userId) {
        MyuserDTO foundUser = getRecord(userId);
        if (foundUser == null) {
            return false;
        }
        Myuser myuser = this.myDTOtoDAO(foundUser);
        System.out.println("DELETING: " + myuser.getName());
        em.getTransaction().begin();
        Myuser myuser2 = em.merge(myuser);
        em.remove(myuser2);
        em.getTransaction().commit();

        return true;
    }
}
```

```
run:
Welcome to the database
1. Check current record
2. Create a record
3. Update a user
4. Delete a record
5. Exit
1
Enter User ID
000001
Name: PeterSmith
Password: 123456
Email: psmith@swin.edu.au
Phone: 9876543210
Address: Swinburne EN510f
SecQn: What is my name?
SecAns: Peter

1. Check current record
2. Create a record
3. Update a user
4. Delete a record
5. Exit
2
Enter UserID
000006
Name:
Duy
Password:
111111
Email:
kevin@gmail.com
Phone:
0123456789
Address:
1 Melbourne ave
Secqn:
How r u
Secans:
im gud
User Created
```

```
User Created
1. Check current record
2. Create a record
3. Update a user
4. Delete a record
5. Exit
1
Enter User ID
000006
Name: Duy
Password: 111111
Email: kevin@gmail.com
Phone: 0123456789
Address: 1 Melbourne ave
SecQn: How r u
SecAns: im gud

1. Check current record
2. Create a record
3. Update a user
4. Delete a record
5. Exit
3
Enter UserID to Update
000006
Name:
KEVIN
Password:
111111
Email:
kevin@yahoo
Phone:
000000
Address:
Road of Mel
Secqn:
U good
Secans:
yes
User Id:000006 has been updated
1. Check current record
```

```
1. Check current record
2. Create a record
3. Update a user
4. Delete a record
5. Exit
1
Enter User ID
000006
Name: KEVIN
Password: 111111
Email: kevin@yahoo
Phone: 000000
Address: Road of Mel
SecQn: U good
SecAns: yes

1. Check current record
2. Create a record
3. Update a user
4. Delete a record
5. Exit
4
Enter UserID to Delete
000006
Record deleted successfully
1. Check current record
2. Create a record
3. Update a user
4. Delete a record
5. Exit
5
```

4.1 MyUser.java is responsible for doing ORM, because it hides SQL queries from OO logic, together with design implementation (i.e. public constructors)
4.2 Myuser class here is different from Myuser class in lab 2, since they have different responsibilities. One does ORM and the other acts as DTO. Myuser class of lab 3 holds public constructors whereas the one in lab 2 holds private constructors.
4.3 Typically, the domain object defines the business object and it's properties and methods. It's used to manipulate and move the data within the processing system. The Entity object exists to take those domain properties and map them to a persistent storage object, such as a database table.