Swinburne University of Technology

Creating Secure and Scalable Software (S1 2020)

Doubtfire Submission

04 Pass_Task 4.1P

Submitted By:
Ashen Lakshitha Fonseka Hewa Fonsekage
102238586
2020/05/19 13:11

Tutor:
Shibli Saleheen

May 19, 2020

Name: Ashen Lakshitha Fonseka
Student ID: 102238586
Tutor: Shibli Saleheen

```
    @Override
    public boolean createRecord(MyuserDTO myuserDTO) {
        if (find(myuserDTO.getUserid()) != null) {
// user whose userid can be found
            return false;
        }
// user whose userid could not be found
        try {
            Myuser myuser = this.myDTO2DAO(myuserDTO);
            this.create(myuser); // add to database
            return true;
        } catch (Exception ex) {
            return false; // something is wrong, should not be here though
        }
    }

    private MyuserDTO myDAO2DTO(Myuser myuser) {
        MyuserDTO dto = new MyuserDTO(myuser.getUserid(), myuser.getName(),
                myuser.getPassword(), myuser.getEmail(), myuser.getPhone(),
                myuser.getAddress(), myuser.getSecqn(), myuser.getSecans());
        return dto;
    }

    public MyuserDTO getRecord(String userId) {
        MyuserDTO record = null;
        Myuser user = find(userId);
        if (user != null) {
            record = myDAO2DTO(user);
        }
        return record;
    }
```

```
@Override
public boolean updateRecord(MyuserDTO myuserDTO) {
    if (find(myuserDTO.getUserid()) != null) {
user whose userid can be found
        return false;
    }
user whose userid could not be found
    try {
        Myuser myuser = this.myDTO2DAO(myuserDTO);
        this.edit(myuser); // add to database
        return true;
    } catch (Exception ex) {
        return false; // something is wrong, should not be here though
    }
}

@Override
public boolean deleteRecord(String userId) {
    MyuserDTO record = null;
    Myuser user = find(userId);
    if (user != null) {
        remove(user);
    }
    return false;
}
```

```
    public ArrayList<MyuserDTO> getRecordsByAddress(String address) {
        ArrayList<MyuserDTO> dtolist = new ArrayList<MyuserDTO>();
        List<Myuser> daoList;
        //List<Myuser> temList;
        Query query = em.createNamedQuery("Myuser.findByAddress").setParameter("address", address);
        daoList = query.getResultList();
        if (daoList.size() == 0) {
            return null;
        }

        for (Myuser user : daoList) {

            dtolist.add(myDAO2DTO(user));
        }
        return dtolist;
    }
}
```

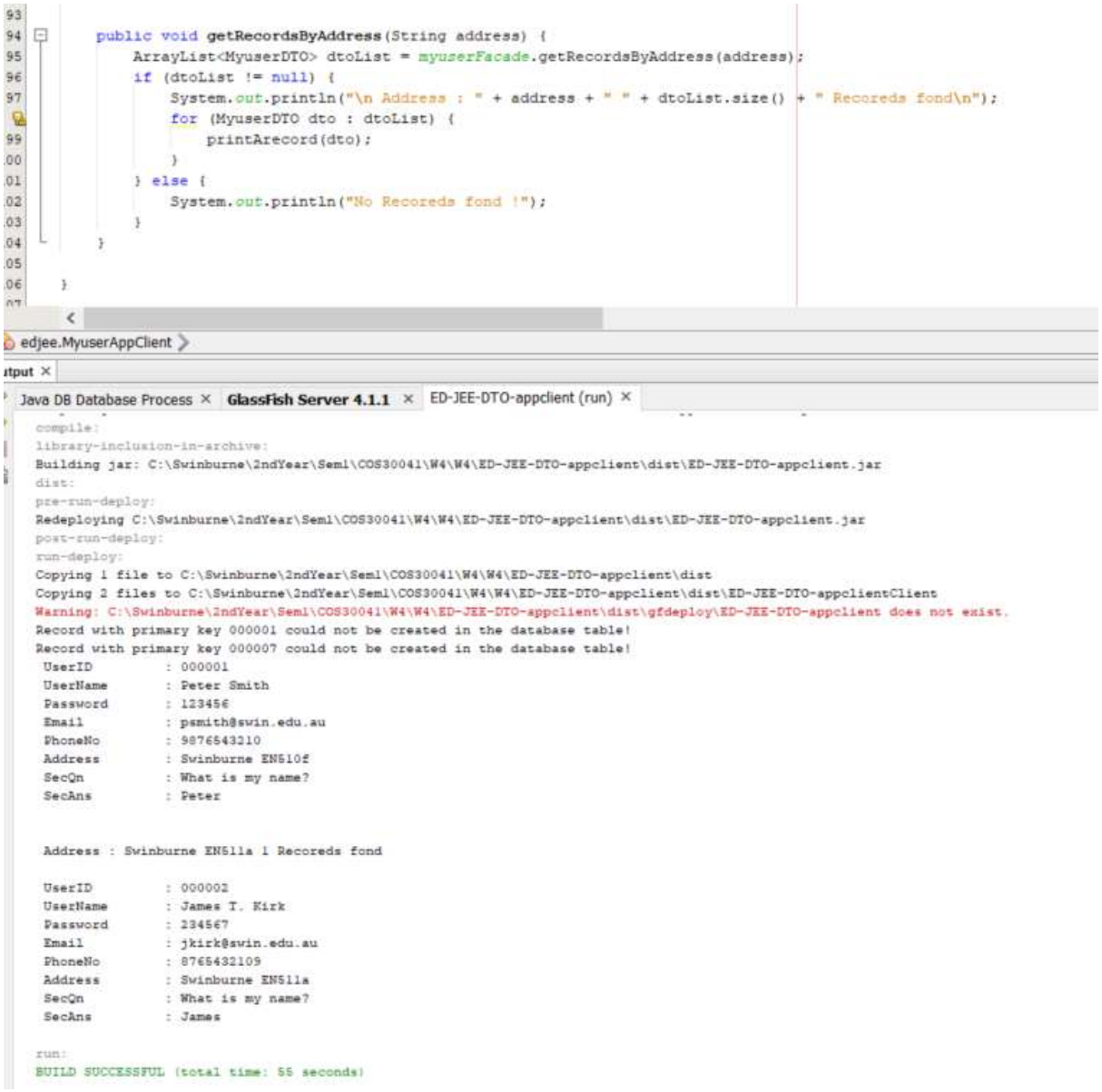
```
        MyuserDTO myuserDTO2 = new MyuserDTO("000007", "David Lee", "e54321",
                "dlee@swin.edu.au", "0123456789", "Swinburne EN510g",
                "What is my name?", "David");
        result = client.createRecord(myuserDTO2);
        result = client.updateRecord(myuserDTO);
        client.showCreateResult(result, myuserDTO2);

        String uID = "000001";
        MyuserDTO record = client.getRecord(uID);
        client.printArecord(record);

        String address = "Swinburne EN511a";
        client.getRecordsByAddress(address);
    }

    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 void printArecord(MyuserDTO dto) {
        if (dto == null) {
            System.out.println("Record can not be found !");
        } else {
            System.out.println(" UserID \t: " + dto.getUserid()
                    + "\n UserName \t: " + dto.getName()
                    + "\n Password \t: " + dto.getPassword()
                    + "\n Email \t\t: " + dto.getEmail()
                    + "\n PhoneNo \t: " + dto.getPhone()
                    + "\n Address \t: " + dto.getAddress()
                    + "\n SecQn \t\t: " + dto.getSecQn()
```



```
    public void getRecordsByAddress(String address) {
        ArrayList<MyuserDTO> dtoList = myuserFacade.getRecordsByAddress(address);
        if (dtoList != null) {
            System.out.println("\n Address : " + address + " " + dtoList.size() + " Recoreds fond\n");
            for (MyuserDTO dto : dtoList) {
                printArecord(dto);
            }
        } else {
            System.out.println("No Recoreds fond !");
        }
    }

}
```

edjee.MyuserAppClient

utput

Java DB Database Process | GlassFish Server 4.1.1 | ED-JEE-DTO-appclient (run)

```
compile:
library-inclusion-in-archive:
Building jar: C:\Swinburne\2ndYear\Seml\COS30041\W4\W4\ED-JEE-DTO-appclient\dist\ED-JEE-DTO-appclient.jar
dist:
pre-run-deploy:
Redeploying C:\Swinburne\2ndYear\Seml\COS30041\W4\W4\ED-JEE-DTO-appclient\dist\ED-JEE-DTO-appclient.jar
post-run-deploy:
run-deploy:
Copying 1 file to C:\Swinburne\2ndYear\Seml\COS30041\W4\W4\ED-JEE-DTO-appclient\dist
Copying 2 files to C:\Swinburne\2ndYear\Seml\COS30041\W4\W4\ED-JEE-DTO-appclient\dist\ED-JEE-DTO-appclientClient
Warning: C:\Swinburne\2ndYear\Seml\COS30041\W4\W4\ED-JEE-DTO-appclient\dist\gfdeploy\ED-JEE-DTO-appclient does not exist.
Record with primary key 000001 could not be created in the database table!
Record with primary key 000007 could not be created in the database table!
 UserID         : 000001
 UserName       : Peter Smith
 Password       : 123456
 Email          : psmith@swin.edu.au
 PhoneNo        : 9876543210
 Address        : Swinburne EN510f
 SecQn          : What is my name?
 SecAns         : Peter


 Address : Swinburne EN511a 1 Recoreds fond

 UserID         : 000002
 UserName       : James T. Kirk
 Password       : 234567
 Email          : jkirk@swin.edu.au
 PhoneNo        : 8765432109
 Address        : Swinburne EN511a
 SecQn          : What is my name?
 SecAns         : James

run:
BUILD SUCCESSFUL (total time: 55 seconds)
```

Myuser is a DAO type object. Which means it directly access to database when the object created. MyuserFacade is a bean type class. This makes responsible for ORM work.

Stateless session beans are pooled. This means that, for each managed bean, the container keeps a certain number of beans instances ready in a pool. For each client request, an instances from the pool is quickly assigned to service the request. When the client finishes, the instance is returned to the pool for later reuse. This setup using a pool means that a small number of bean instances can service a relatively large number of clients.