Bài thực hành 04
Restful Web Services

1. Mục tiêu

- *Xây dựng ứng dụng Restful Web Service với Jersey để cung cấp các phương thức xử lý với database.*

2. Bài thực hành Step by Step

Bài thực hành 1: Tạo CSDL gồm 2 bảng City và Country, insert một số dữ liệu ban đầu cho 2 bảng này. Tạo rest web service cung cấp các chức năng xử lý CRUD với bảng City.

Bước 1: Tạo database cho ứng dụng

```
create database DBService_Restful
go
use DBService_Restful
go
create table Country(
	CountryId varchar(15) not null primary key,
	CountrName nvarchar(100))


create table City(
	CityId int identity primary key,
	CountryId varchar(15) foreign key references Country(CountryId),
	CityName nvarchar(100),
	FoundationDate datetime,
	Area float,
	Population int)


insert into Country values ('VN',N'Viet Nam')
insert into Country values ('LA',N'Lao')
insert into Country values ('CAM',N'Campuchia')
insert into Country values ('TL',N'Thai Lan')
insert into Country values ('MAR',N'Myanmar')
insert into Country values ('IDO',N'Indonesia')

insert into City values
('VN','Ha Noi','1885-01-12',30000,8000000),
('VN','TP Ho Chi Minh','1887-03-05',50000,9000000),
('VN','Da Nang','1886-11-21',20000,4000000)
```

Bước 2: Tạo ứng dụng Dynamic Web Project có tên WEBSERV_Lab04_Webservices

File → New → Other → Web → Dynamic Web Project . Đặt tên project **"WEBSERV_Lab04_Webservices"**

Bước 3: **Convert to maven project**

Convert project về maven project để quản lý các thư viện của dự án trong file pom.xml

Click phải vào project → Configure → Convert to Maven Project → Finish

Bước 4: **Thêm các thư viện cho ứng dụng trong file pom.xml**

Mở file pom.xml chèn cặp thẻ <dependencies>…</dependencies> giữa thẻ <packaging> và thẻ <build>

Thêm các thư viện sau:

- hibernate-core
- hibernate-entitymanager
- jersey-server
- gson

- **Thêm thư viện sqljdbc4-4.0.jar vào thư mục lib của ứng dụng và chọn Build Path/ Add to Build Path.**

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd
/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <groupId>Demo_WebService</groupId>
    <artifactId>Demo_WebService</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <packaging>war</packaging>
    <dependencies>
        <!-- https://mvnrepository.com/artifact/org.hibernate/hibernate-core -->
        <dependency>
            <groupId>org.hibernate</groupId>
            <artifactId>hibernate-core</artifactId>
            <version>5.4.27.Final</version>
        </dependency>

        <!-- https://mvnrepository.com/artifact/org.hibernate/hibernate-
```

```
entitymanager -->
        <dependency>
            <groupId>org.hibernate</groupId>
            <artifactId>hibernate-entitymanager</artifactId>
            <version>5.4.27.Final</version>
        </dependency>
        <!-- https://mvnrepository.com/artifact/com.sun.jersey/jersey-server -->
        <dependency>
            <groupId>com.sun.jersey</groupId>
            <artifactId>jersey-server</artifactId>
            <version>1.9</version>
        </dependency>
        <!-- https://mvnrepository.com/artifact/com.google.code.gson/gson -->
        <dependency>
            <groupId>com.google.code.gson</groupId>
            <artifactId>gson</artifactId>
            <version>2.8.6</version>
        </dependency>

    </dependencies>
    <build>
        <sourceDirectory>src</sourceDirectory>
        <plugins>
            <plugin>
                <artifactId>maven-compiler-plugin</artifactId>
                <version>3.8.1</version>
                <configuration>
                    <source>1.8</source>
                    <target>1.8</target>
                </configuration>
            </plugin>
            <plugin>
                <artifactId>maven-war-plugin</artifactId>
                <version>3.2.3</version>
                <configuration>
                    <warSourceDirectory>WebContent</warSourceDirectory>
                </configuration>
            </plugin>
        </plugins>
    </build>
</project>
```

Bước 5: Cấu hình file web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-
```

```
instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.
jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-
app_4_0.xsd" id="WebApp_ID" version="4.0">
  <display-name>Demo_WebService</display-name>
     <servlet>
        <servlet-name>myservlet</servlet-name>
        <servlet-class>com.sun.jersey.spi.container.servlet.ServletContainer</servlet-
class>
        <init-param>
            <param-name>com.sun.jersey.config.property.packages</param-name>
            <param-value>webapi.services</param-value>
        </init-param>
     </servlet>
     <servlet-mapping>
        <servlet-name>myservlet</servlet-name>
        <url-pattern>/rest/*</url-pattern>
     </servlet-mapping>
</web-app>
```

<u>Bước 6:</u> **Tạo các class entities**

Tạo class Country

```
package country_city.entities;

import java.util.List;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.OneToMany;
import javax.persistence.Table;

@Entity
@Table(name = "Country")
public class Country {
    @Id
    @Column(name = "CountryId")
    private String countryId;
    @Column(name = "CountrName")
    private String countryName;

    @OneToMany(mappedBy = "countryId")
    private List<City> listCity;

    public Country() {
        super();
        // TODO Auto-generated constructor stub
    }

    public Country(String countryId, String countryName, List<City> listCity) {
        super();
        this.countryId = countryId;
        this.countryName = countryName;
        this.listCity = listCity;
    }

    public String getCountryId() {
        return countryId;
    }

    public void setCountryId(String countryId) {
        this.countryId = countryId;
    }

    public String getCountryName() {
        return countryName;
    }

    public void setCountryName(String countryName) {
        this.countryName = countryName;
    }

    public List<City> getListCity() {
        return listCity;
    }

    public void setListCity(List<City> listCity) {
        this.listCity = listCity;
    }
}
```

Tạo class City

```
package country_city.entities;

import java.util.Date;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;
import javax.persistence.Table;

@Entity
@Table(name = "City")
public class City {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name = "CityId")
    private Integer cityId;
    @Column(name = "CityName")
    private String cityName;
    @Column(name = "FoundationDate")
    private Date foundationDate;
    @Column(name = "Area")
    private Double area;
    @Column(name = "Population")
    private Integer population;

    @ManyToOne
    @JoinColumn(name = "CountryId",referencedColumnName = "CountryId")
    private Country countryId;

    public City() {
        super();
        // TODO Auto-generated constructor stub
    }

    public City(Integer cityId, String cityName, Date foundationDate, Double area, Int
eger population,
            Country countryId) {
        super();
        this.cityId = cityId;
        this.cityName = cityName;
        this.foundationDate = foundationDate;
        this.area = area;
        this.population = population;
        this.countryId = countryId;
    }

    public Integer getCityId() {
        return cityId;
    }

    public void setCityId(Integer cityId) {
        this.cityId = cityId;
    }

    public String getCityName() {
        return cityName;
    }

    public void setCityName(String cityName) {
        this.cityName = cityName;
    }

    public Date getFoundationDate() {
        return foundationDate;
    }

    public void setFoundationDate(Date foundationDate) {
        this.foundationDate = foundationDate;
    }

    public Double getArea() {
        return area;
    }

    public void setArea(Double area) {
        this.area = area;
    }

    public Integer getPopulation() {
        return population;
    }

    public void setPopulation(Integer population) {
        this.population = population;
    }

    public Country getCountryId() {
        return countryId;
    }

    public void setCountryId(Country countryId) {
        this.countryId = countryId;
    }

}
```

Tạo class CityDTO

```
package webapi.entities.dto;

import java.util.Date;

public class CityDTO {
    private Integer cityId;
    private String countryId;
    private String cityName;
    private Date foundationDate;
    private Double area;
    private Long population;
    public CityDTO() {
        super();
        // TODO Auto-generated constructor stub
    }
    public CityDTO(Integer cityId, String countryId, String cityName, Date foundationD
ate, Double area,
            Long population) {
        super();
        this.cityId = cityId;
        this.countryId = countryId;
        this.cityName = cityName;
        this.foundationDate = foundationDate;
        this.area = area;
        this.population = population;
    }
    public Integer getCityId() {
        return cityId;
    }
    public void setCityId(Integer cityId) {
        this.cityId = cityId;
    }
    public String getCountryId() {
        return countryId;
    }
    public void setCountryId(String countryId) {
        this.countryId = countryId;
    }
    public String getCityName() {
        return cityName;
    }
    public void setCityName(String cityName) {
        this.cityName = cityName;
    }
    public Date getFoundationDate() {
        return foundationDate;
    }
    public void setFoundationDate(Date foundationDate) {
        this.foundationDate = foundationDate;
    }
    public Double getArea() {
        return area;
    }
    public void setArea(Double area) {
        this.area = area;
    }
    public Long getPopulation() {
        return population;
    }
    public void setPopulation(Long population) {
        this.population = population;
    }
}
```

<u>Bước 7:</u> **Tạo file cấu hình hibernate.cfg.xml**

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-configuration PUBLIC
        "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
        "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
<hibernate-configuration>
    <session-factory>
        <property name="hibernate.connection.driver_class">com.microsoft.sqlserver.jdb
c.SQLServerDriver</property>
        <property name="hibernate.connection.password">1234$</property>
        <property name="hibernate.connection.url">jdbc:sqlserver://localhost:1433;data
baseName=DBService_RestFul</property>
        <property name="hibernate.connection.username">sa</property>
        <property name="hibernate.dialect">org.hibernate.dialect.SQLServer2008Dialect<
/property>
        <mapping class="webapi.entities.City"/>
        <mapping class="webapi.entities.Country"/>
    </session-factory>
</hibernate-configuration>
```

<u>Bước 8:</u> **Cài đặt class HibernateUtil.java**

```
package webapi.hibernate.util;

import org.hibernate.SessionFactory;
import org.hibernate.boot.Metadata;
import org.hibernate.boot.MetadataSources;
import org.hibernate.boot.registry.StandardServiceRegistryBuilder;
import org.hibernate.service.ServiceRegistry;

public class HibernateUtil {
    private static SessionFactory sessionFactory;
    static {
        if(sessionFactory==null) {
            ServiceRegistry reg = new StandardServiceRegistryBuilder().configure().bui
ld();
            MetadataSources source = new MetadataSources(reg);
            Metadata metadata = source.getMetadataBuilder().build();
            sessionFactory = metadata.getSessionFactoryBuilder().build();
        }
    }

    public static SessionFactory getSessionFactory() {
        return sessionFactory;
    }
}
```

Bước 9: Tạo giao diện CityDAO

```
package webapi.dao;

import java.util.List;

import webapi.entities.City;

public interface CityDAO {
    public List<City> getListCitites();
    public City getCityById(Integer cityId);
    public boolean insertCity(City city);
    public boolean updateCity(City city);
    public boolean deleteCity(Integer cityId);
    public List<City> getCitiesByCountryName(String countryName);
}
```

Bước 10: Cài đặt phương thức trong lớp CityDAOImpl

```
package webapi.dao;

import java.util.List;

import org.hibernate.Session;
import org.hibernate.SessionFactory;
```

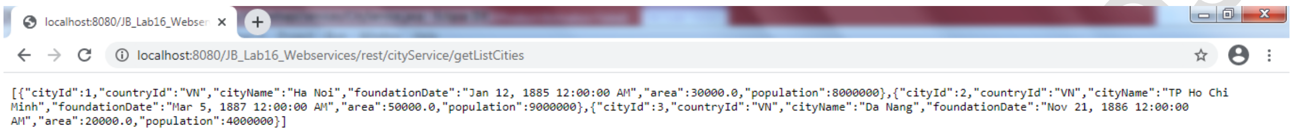
```java
import webapi.entities.City;
import webapi.hibernate.util.HibernateUtil;

public class CityDAOImpl implements CityDAO{

    @Override
    public List<City> getListCitites() {
        // TODO Auto-generated method stub
        SessionFactory sessionFactory = HibernateUtil.getSessionFactory();
        Session session = sessionFactory.openSession();
        try {
            session.beginTransaction();
            List list = session.createQuery("from City").list();
            session.getTransaction().commit();
            return list;
        } catch (Exception e) {
            // TODO: handle exception
            e.printStackTrace();
            session.getTransaction().rollback();
        }finally {
            session.close();
        }
        return null;
    }

    @Override
    public City getCityById(Integer cityId) {
        // TODO Auto-generated method stub
        SessionFactory sessionFactory = HibernateUtil.getSessionFactory();
        Session session = sessionFactory.openSession();
        try {
            session.beginTransaction();
//          City c = (City) session.createQuery("from City where cityId = :cityId")
//          .setParameter("cityId", cityId)
//          .uniqueResult();
            City c = session.get(City.class, cityId);
            session.getTransaction().commit();
            return c;
        } catch (Exception e) {
            // TODO: handle exception
            e.printStackTrace();
            session.getTransaction().rollback();
        }finally {
            session.close();
        }
        return null;
    }

    @Override
    public boolean insertCity(City city) {
        SessionFactory sessionFactory = HibernateUtil.getSessionFactory();
        Session session = sessionFactory.openSession();
        try {
            session.beginTransaction();
            session.save(city);
            session.getTransaction().commit();
            return true;
        } catch (Exception e) {
            // TODO: handle exception
            e.printStackTrace();
            session.getTransaction().rollback();
        }finally {
            session.close();
        }
        return false;
    }

    @Override
    public boolean updateCity(City city) {
        SessionFactory sessionFactory = HibernateUtil.getSessionFactory();
        Session session = sessionFactory.openSession();
        try {
            session.beginTransaction();
            session.update(city);
            session.getTransaction().commit();
            return true;
        } catch (Exception e) {
            // TODO: handle exception
            e.printStackTrace();
            session.getTransaction().rollback();
        }finally {
            session.close();
        }
        return false;
    }

    @Override
    public boolean deleteCity(Integer cityId) {
        SessionFactory sessionFactory = HibernateUtil.getSessionFactory();
        Session session = sessionFactory.openSession();
        try {
            session.beginTransaction();
            int i = session.createQuery("delete from City where cityId = :cityId")
            .setParameter("cityId", cityId)
            .executeUpdate();
            session.getTransaction().commit();
            if(i>0)
                return true;
        } catch (Exception e) {
            // TODO: handle exception
            e.printStackTrace();
            session.getTransaction().rollback();
        }finally {
            session.close();
        }
        return false;
    }

    @Override
    public List<City> getCitiesByCountryName(String countryName) {
        SessionFactory sessionFactory = HibernateUtil.getSessionFactory();
        Session session = sessionFactory.openSession();
        try {
            session.beginTransaction();
            if(countryName.length()==0) {
                countryName = "%";
            }else {
                countryName = "%"+countryName+"%";
            }
            List list = session.createQuery("from City where objCountry.countryName li
ke :countryName")
                    .setParameter("countryName", countryName)
                    .list();
            session.getTransaction().commit();
            return list;
        } catch (Exception e) {
            // TODO: handle exception
            e.printStackTrace();
            session.getTransaction().rollback();
        }finally {
            session.close();
        }
        return null;
    }
}
```

Bước 11: Cài đặt lớp CityService

```
package webapi.services;

import java.util.ArrayList;
import java.util.List;

import javax.ws.rs.Consumes;
import javax.ws.rs.GET;
import javax.ws.rs.POST;
import javax.ws.rs.Path;
import javax.ws.rs.PathParam;
import javax.ws.rs.Produces;
import javax.ws.rs.core.MediaType;

import com.google.gson.Gson;

import webapi.dao.CityDAOImpl;
import webapi.entities.City;
import webapi.entities.Country;
import webapi.entities.dto.CityDTO;

@Path(value = "/cityService")
public class CityService {

    @GET
    @Path("/getListCities")
    @Produces(MediaType.APPLICATION_JSON)
    public String getListCities() {
        List<City> listCitites = new CityDAOImpl().getListCitites();
        Gson son = new Gson();
        List<CityDTO> listData = new ArrayList<CityDTO>();
        for (City c : listCitites) {
            CityDTO cdto = new CityDTO(c.getCityId(), c.getObjCountry().getCountryId()
, c.getCityName(), c.getFoundationDate(), c.getArea(), c.getPopulation());
            listData.add(cdto);
        }
        String data = son.toJson(listData);
        return data;
    }

    @POST
    @Path("/insertCity")
    @Consumes(MediaType.APPLICATION_JSON)
    public String insertCity(String c) {
        Gson son = new Gson();
        CityDTO objDTO = son.fromJson(c, CityDTO.class);
        Country objCountry = new Country();
        objCountry.setCountryId(objDTO.getCountryId());
        City objCity = new City(0, objDTO.getCityName(), objDTO.getFoundationDate(), o
bjDTO.getArea(), objDTO.getPopulation(), objCountry);
        boolean bl = new CityDAOImpl().insertCity(objCity);
        String data = son.toJson(bl);
        return data;
    }

    @POST
    @Path("/updateCity")
    @Consumes(MediaType.APPLICATION_JSON)
    public String updateCity(String c) {
        Gson son = new Gson();
        CityDTO objDTO = son.fromJson(c, CityDTO.class);
        Country objCountry = new Country();
        objCountry.setCountryId(objDTO.getCountryId());
        City objCity = new City(objDTO.getCityId(), objDTO.getCityName(), objDTO.getFo
undationDate(), objDTO.getArea(), objDTO.getPopulation(), objCountry);
        boolean bl = new CityDAOImpl().updateCity(objCity);
        String data = son.toJson(bl);
        return data;
    }

    @POST
    @Path("/deleteCity/{cityId}")
    @Consumes(MediaType.APPLICATION_JSON)
    public String deleteCity(@PathParam("cityId")Integer cityId) {
        Gson son = new Gson();
        boolean bl = new CityDAOImpl().deleteCity(cityId);
        String data = son.toJson(bl);
        return data;
    }

    @GET
    @Path("/getCitiesByCountryName/{countryName}")
    @Produces(MediaType.APPLICATION_JSON)
    public String getCitiesByCountryName(@PathParam("countryName")String countryName)
{
        List<City> listCitites = new CityDAOImpl().getCitiesByCountryName(countryName)
;
        Gson son = new Gson();
        List<CityDTO> listData = new ArrayList<CityDTO>();
        for (City c : listCitites) {
            CityDTO cdto = new CityDTO(c.getCityId(), c.getObjCountry().getCountryId()
, c.getCityName(), c.getFoundationDate(), c.getArea(), c.getPopulation());
            listData.add(cdto);
        }
        String data = son.toJson(listCitites);
        return data;
    }
}
```

Chạy thử ứng dụng: Right click project / Run As / Run on Server

Gõ url để chạy vào hàm getListCities


localhost:8080/JB_Lab16_Webser
localhost:8080/JB_Lab16_Webservices/rest/cityService/getListCities

```
[{"cityId":1,"countryId":"VN","cityName":"Ha Noi","foundationDate":"Jan 12, 1885 12:00:00 AM","area":30000.0,"population":8000000},{"cityId":2,"countryId":"VN","cityName":"TP Ho Chi
Minh","foundationDate":"Mar 5, 1887 12:00:00 AM","area":50000.0,"population":9000000},{"cityId":3,"countryId":"VN","cityName":"Da Nang","foundationDate":"Nov 21, 1886 12:00:00
AM","area":20000.0,"population":4000000}]
```

3. ***Bài tập tự làm***

Bài 1. Tạo database: DB_WebAPI1

Tên bảng: TblEmployee

Tên cột	Kiểu dữ liệu	Mô tả
EmpId	int – tự tăng	Khóa chính
EmpName	Nvarchar(50)	
Gender	Bit	
Birthday	Date	
Address	Nvarchar(200)	

Position	Nvarchar(100)	
Salary	float	

- Thêm một số dữ liệu ban đầu cho bảng này.
- Tạo web service cung cấp các phương thức:
    - Get all employees
    - Get employee by id
    - Insert new employee
    - Update employee
    - Delete employee

Bài 2: Tạo database như sau

```
create database DB_WebAPI2
go
use DB_Class_Student
go
create table Classes(
       ClassId varchar(15) not null primary key,
       ClassName nvarchar(100) unique)

insert into Classes values ('C1808M',N'Lập trình quốc tế ACCPi17 Lớp C1808M')
insert into Classes values ('C1808I',N'Lập trình quốc tế ACCPi17 Lớp C1808I')
insert into Classes values ('C1807G',N'Lập trình quốc tế ACCPi17 Lớp C1807G')
insert into Classes values ('C1807G2H',N'Lập trình quốc tế ACCPi17 Lớp C1807G2H')
insert into Classes values ('C18010M',N'Lập trình quốc tế ACCPi17 Lớp C18010M')

create table Student(
       StuId int identity primary key,
       ClassId varchar(15) foreign key references Classes(ClassId),
       FullName nvarchar(70),
       Gender bit,
       Birthday datetime,
       Address nvarchar(200))

insert into Student values ('C1808M',N'Nguyễn Đức Trọng',1,'2001-05-05',N'Ba Đình
- Hà Nội')
insert into Student values ('C1807G',N'Trần Khánh Huyền',0,'2000-11-23',N'Sóc Sơn
- Hà Nội')
insert into Student values ('C1808I',N'Nguyễn Thành Nam',1,'2000-12-24',N'Gia Lâm
- Hà Nội')

select * from Classes
select * from Student
```

- Tạo web service cung cấp các phương thức sau:
  - Get all students
  - Get student by id
  - Insert new student
  - Update student
  - Delete student