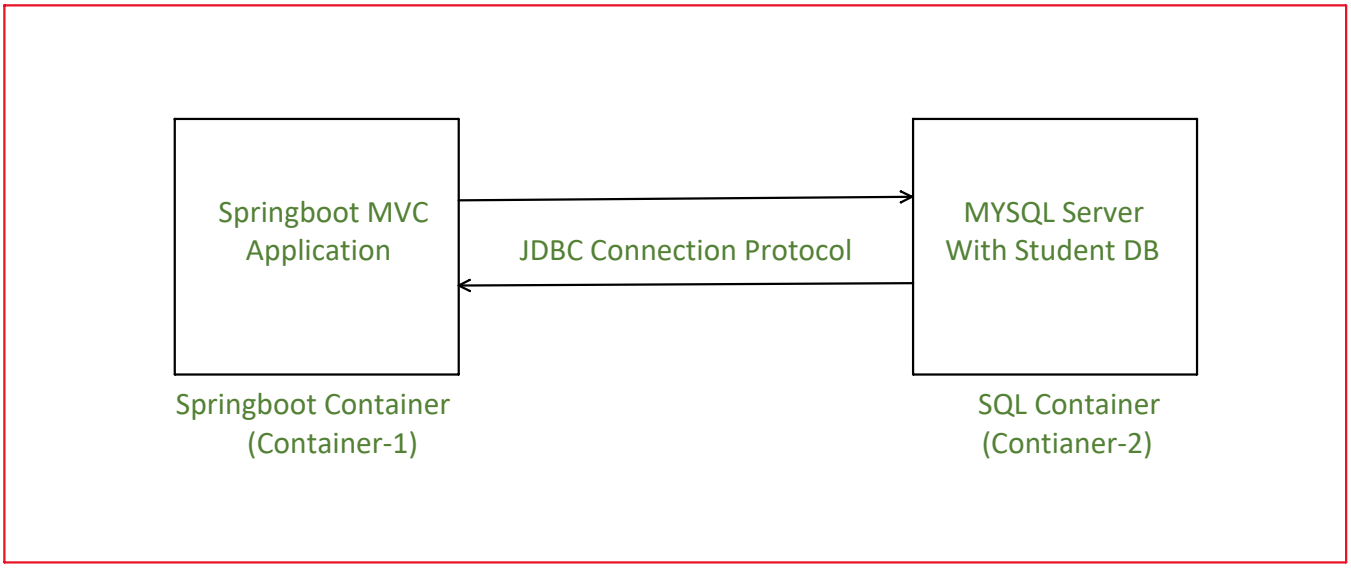
Chapter-8 (Assignment)

09 August 2022 03:35

- Create a web application which have capability to perform CRUD operation of Student.

Application Architecture



Steps for configuring MYSQL Server

- First we need to write SQL script to initialize our database with user & DB creation. And also providing user privileges to perform operation in DB.
  - Sample SQL script:

```
CREATE USER 'dockerUsr'@'%' IDENTIFIED BY 'dockerPass';
CREATE DATABASE IF NOT EXISTS dockerdb DEFAULT CHARACTER SET utf8 DEFAULT COLLATE utf8_general_ci;
ALTER USER 'dockerUsr'@'%' IDENTIFIED WITH mysql_native_password BY 'dockerPass';
GRANT SELECT, INSERT, CREATE, ALTER, DROP, INDEX, LOCK TABLES, CREATE TEMPORARY TABLES, DELETE, UPDATE, EXECUTE ON dockerdb.* TO 'dockerUsr'@'%';
FLUSH PRIVILEGES;
```

  - Sample Dockerfile:

```
FROM mysql
MAINTAINER CodeHop
ARG DB_PASS
ARG DB_PORT
ENV MYSQL_ROOT_PASSWORD $DB_PASS
COPY init.sql /docker-entrypoint-initdb.d/
EXPOSE $DB_PORT
```

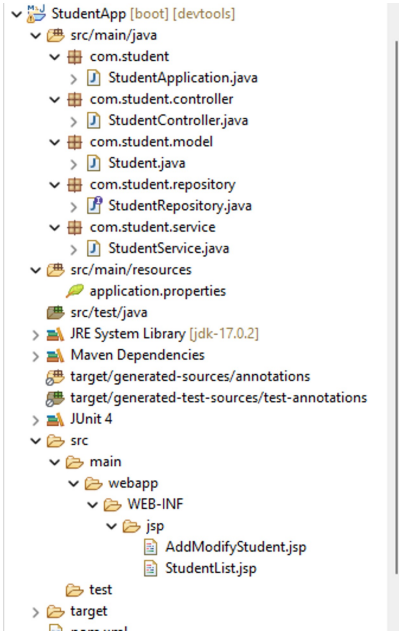
  - Command to build & execute:
    - `docker build . -t studentdbserver:v1 --build-arg DB_PASS=dockerPass --build-arg DB_PORT=3306`

docker build -t studentdbserver:v1 --build-arg DB_PASS=dockerpass --build-arg DB_PORT=3306

- docker run -d -p 3309:3306 --name studentdbservice studentdbserver:v1

Steps for configuring Springboot Application

- Project Structure:



- Pom.xml :

```
<?xml version="1.0" encoding="UTF-8"?>
<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 http://maven.apache.org/xsd/maven-4.0.0.xsd">
	<modelVersion>4.0.0</modelVersion>
	<groupId>com.student</groupId>
	<artifactId>StudentApp</artifactId>
	<version>0.0.1-SNAPSHOT</version>
	<packaging>war</packaging>
	<name>StudentApp</name>
	<description>StudentApp</description>
	<parent>
		<groupId>org.springframework.boot</groupId>
		<artifactId>spring-boot-starter-parent</artifactId>
		<version>2.7.2</version>
		<relativePath /> <!-- lookup parent from repository -->
	</parent>
	<properties>
		<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
		<project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>
		<java.version>17</java.version>
	</properties>
	<dependencies>
		<dependency>
			<groupId>javax.servlet</groupId>
			<artifactId>jstl</artifactId>
			<scope>provided</scope>
		</dependency>
		<dependency>
			<groupId>org.apache.tomcat.embed</groupId>
			<artifactId>tomcat-embed-jasper</artifactId>
			<scope>provided</scope>
		</dependency>
```

```
                <dependency>
                    <groupId>org.apache.tomcat.embed</groupId>
                    <artifactId>tomcat-embed-jasper</artifactId>
                    <scope>provided</scope>
                </dependency>
                <dependency>
                    <groupId>org.springframework.boot</groupId>
                    <artifactId>spring-boot-starter-data-jpa</artifactId>
                </dependency>
                <dependency>
                    <groupId>org.springframework.boot</groupId>
                    <artifactId>spring-boot-starter-web</artifactId>
                </dependency>
                <dependency>
                    <groupId>org.springframework.boot</groupId>
                    <artifactId>spring-boot-devtools</artifactId>
                    <scope>runtime</scope>
                </dependency>
                <dependency>
                    <groupId>mysql</groupId>
                    <artifactId>mysql-connector-java</artifactId>
                    <scope>runtime</scope>
                </dependency>
                <dependency>
                    <groupId>org.springframework.boot</groupId>
                    <artifactId>spring-boot-starter-test</artifactId>
                    <scope>test</scope>
                </dependency>
            </dependencies>
            <build>
                <plugins>
                    <plugin>
                        <groupId>org.springframework.boot</groupId>
                        <artifactId>spring-boot-maven-plugin</artifactId>
                    </plugin>
                </plugins>
            </build>
        </project>
```

- Controller Class :

```java
package com.student.controller;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.MediaType;
import org.springframework.stereotype.Controller;
import org.springframework.ui.ModelMap;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;

import com.student.model.Student;
import com.student.service.StudentService;

@Controller
public class StudentController {

    @Autowired
    private StudentService service;

    @RequestMapping(value = {"/","/list"}, method = {RequestMethod.GET,RequestMethod.POST}, consumes= {MediaType.ALL_VALUE})
    public String listStudent(Student student,ModelMap model) {
            List<Student> list = service.listStudent(student);
            model.put("studentList", list);
            model.put("student", student);
            model.put("message", "Welcome to Student Application");
        return "StudentList";
    }

    @GetMapping(value = {"/add","/edit"}, consumes= {MediaType.ALL_VALUE})
    public String addStudent(Student student,ModelMap model) {
        if(student!=null && student.getRollNumber()!=null) {
            Student studentFrmDb = service.readStudent(student);
            model.put("student", studentFrmDb);
            model.put("heading", "Modify Student");
            model.put("url", "/modify");
```

```
        }else {
            model.put("student", new Student());
            model.put("heading", "Add New Student");
            model.put("url", "/addnew");
        }
        return "AddModifyStudent";
    }

    @PostMapping(value = {"/modify","/addnew"}, consumes= {MediaType.ALL_VALUE})
    public String addModifyStudent(Student student,ModelMap model){
            String msg ="Error";
            try {
                if(student.getRollNumber()==null) {
                    msg = "Student Added Successfully";
                    service.addNewStudent(student);
                }else {
                    msg = "Student Modified Successfully";
                    service.modifyStudent(student);
                }
                model.put("message", msg);
            } catch (Exception e) {
                model.put("message", msg);
            }
            List<Student> list = service.listStudent(student);
            model.put("studentList", list);
            return "StudentList";
    }

}

```

- Sample application.properties :

```
## Spring view resolver set up
spring.mvc.view.prefix=/WEB-INF/jsp/
spring.mvc.view.suffix=.jsp

## Spring DATASOURCE (DataSourceAutoConfiguration & DataSourceProperties)
spring.datasource.url = jdbc:mysql://${DB_HOST:localhost}:${DB_PORT:3306}/${DB_NAME:studentdb}
spring.datasource.username = ${DB_UNAME:root}
spring.datasource.password = ${DB_PASS:root}


## Hibernate Properties
# The SQL dialect makes Hibernate generate better SQL for the chosen database
spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.MySQL5InnoDBDialect

# Hibernate ddl auto (create, create-drop, validate, update)
spring.jpa.hibernate.ddl-auto = update

logging.level.root=INFO

```

- Sample Dockerfile :

```
FROM openjdk:17-alpine
MAINTAINER CodeHop
ARG DB_HOST
ARG DB_PORT
ENV DB_PASS=dockerPass
ENV DB_NAME=dockerdb
ENV DB_UNAME=dockerUsr
ENV DB_HOST=$DB_HOST
ENV DB_PORT=$DB_PORT
RUN mkdir -p /opt/student/
WORKDIR /opt/student/
ADD StudentApp.war StudentService.war
ENTRYPOINT ["java","-jar","StudentService.war"]
```

- Command to build & execute:
  - To build Springboot image we need to pass argument as DB_HOST, which is the IP of docker mysql container:
    - `docker inspect studentdbservice`
  - Now build docker image :
    - `docker build . -t studentservice:v1 --build-arg DB_HOST=`**`172.17.0.2`** `--build-arg DB_PORT=`**`3306`**

- docker build . -t studentservice:v1 --build-arg DB_HOST=**172.17.0.2** --build-arg DB_PORT=**3306**

- docker run -d -p 8081:8080 --name studentservice studentservice:v1