

```

1  =====pom.xml=====
2  <?xml version="1.0" encoding="UTF-8"?>
3  <project xmlns="http://maven.apache.org/POM/4.0.0"
4  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
6  https://maven.apache.org/xsd/maven-4.0.0.xsd">
7  <modelVersion>4.0.0</modelVersion>
8  <parent>
9  <groupId>org.springframework.boot</groupId>
10 <artifactId>spring-boot-starter-parent</artifactId>
11 <version>2.3.0.RELEASE</version>
12 <relativePath/> <!-- lookup parent from repository -->
13 </parent>
14 <groupId>com.ameya</groupId>
15 <artifactId>003-course-api-jdbc</artifactId>
16 <version>0.0.1-SNAPSHOT</version>
17 <name>003-course-api-jdbc</name>
18 <description>Demo project for Spring Boot</description>
19 <properties>
20 <java.version>1.8</java.version>
21 </properties>
22 <dependencies>
23 <dependency>
24 <groupId>org.springframework.boot</groupId>
25 <artifactId>spring-boot-starter-jdbc</artifactId>
26 </dependency>
27 <dependency>
28 <groupId>org.springframework.boot</groupId>
29 <artifactId>spring-boot-starter-web</artifactId>
30 </dependency>
31 <dependency>
32 <groupId>mysql</groupId>
33 <artifactId>mysql-connector-java</artifactId>
34 <scope>runtime</scope>
35 </dependency>
36 <dependency>
37 <groupId>org.springframework.boot</groupId>
38 <artifactId>spring-boot-starter-test</artifactId>
39 <scope>test</scope>
40 <exclusions>
41 <exclusion>
42 <groupId>org.junit.vintage</groupId>
43 <artifactId>junit-vintage-engine</artifactId>
44 </exclusion>
45 </exclusions>
46 </dependency>
47 </dependencies>
48 <build>
49 <plugins>
50 <plugin>
51 <groupId>org.springframework.boot</groupId>
52 <artifactId>spring-boot-maven-plugin</artifactId>
53 </plugin>
54 </plugins>
55 </build>
56
57 </project>
58 =====application.properties=====
59 server.port=9001
60 logging.level.web=trace
61
62 spring.datasource.url=jdbc:mysql://localhost:3306/nineleapsdb
63 spring.datasource.username=root
64 spring.datasource.password=root
65 spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
66 #spring.datasource.initialization-mode=always
67 =====schema.sql=====
68 CREATE TABLE topic(id INT(11) NOT NULL, topicname VARCHAR(50) NOT NULL, topicdesc
69 VARCHAR(100) NOT NULL, PRIMARY KEY (id));
70 =====data.sql=====
71 insert into topic values(101,'JavaSE','Introduction To Core Java 3 Days');

```

```

71 insert into topic values(102,'JavaEE','Introduction To JavaEE 3 Days');
72 =====Topic.java=====
73 package com.ameya.models;
74
75 public class Topic {
76
77     private int id;
78     private String topicName;
79     private String topicDesc;
80     public Topic() {
81
82     }
83     public Topic(int id, String topicName, String topicDesc) {
84         super();
85         this.id = id;
86         this.topicName = topicName;
87         this.topicDesc = topicDesc;
88     }
89     public int getId() {
90         return id;
91     }
92     public void setId(int id) {
93         this.id = id;
94     }
95     public String getTopicName() {
96         return topicName;
97     }
98     public void setTopicName(String topicName) {
99         this.topicName = topicName;
100     }
101     public String getTopicDesc() {
102         return topicDesc;
103     }
104     public void setTopicDesc(String topicDesc) {
105         this.topicDesc = topicDesc;
106     }
107
108 }
109 =====TopicDAO.java=====
110 package com.ameya.daos;
111
112 import java.util.List;
113
114 import com.ameya.models.Topic;
115
116 public interface TopicDAO {
117
118     List<Topic> getAllTopics();
119     void addTopic(Topic topic);
120     void updateTopic(Topic topic);
121     Topic getTopic(int id);
122     void deleteTopic(int id);
123 }
124 =====TopicDAOImpl.java=====
125 package com.ameya.daos.impl;
126
127 import java.util.List;
128
129 import org.springframework.beans.factory.annotation.Autowired;
130 import org.springframework.jdbc.core.JdbcTemplate;
131 import org.springframework.stereotype.Repository;
132
133 import com.ameya.daos.TopicDAO;
134 import com.ameya.daos.util.TopicRowMapper;
135 import com.ameya.models.Topic;
136
137 @Repository
138 public class TopicDAOImpl implements TopicDAO {
139
140     @Autowired
141     private JdbcTemplate jdbcTemplate;
142
143     @Override

```

```

144     public List<Topic> getAllTopics() {
145         final String SQL="select * from topic";
146         List<Topic> topics=jdbcTemplate.query(SQL,new TopicRowMapper());
147         return topics;
148     }
149
150     @Override
151     public void addTopic(Topic topic) {
152         final String SQL="insert into topic (id,topicname,topicdesc)"
153             + " values (?, ?, ?)";
154         jdbcTemplate.update(SQL,topic.getId(),topic.getTopicName(),
155             topic.getTopicDesc());
156         System.out.println("++++ Topic Added To DB ++++");
157     }
158
159
160     @Override
161     public void updateTopic(Topic topic) {
162         final String SQL="update topic set topicname = ? , "
163             + "topicdesc = ? where id = ?";
164         jdbcTemplate.update(SQL,topic.getTopicName(),
165             topic.getTopicDesc(),topic.getId());
166         System.out.println("++++ Topic Updated ++++");
167     }
168
169
170     @Override
171     public Topic getTopic(int id) {
172         final String SQL="select * from topic where id = ?";
173         Topic topic=jdbcTemplate.queryForObject(SQL,new Object[] {id},
174             new TopicRowMapper());
175         return topic;
176     }
177
178     @Override
179     public void deleteTopic(int id) {
180         final String SQL="delete from topic where id = ?";
181         jdbcTemplate.update(SQL,id);
182         System.out.println("++++ Topic Deleted ++++");
183     }
184
185 }
186
187 =====TopicService.java=====
188 package com.ameya.services;
189
190 import java.util.List;
191
192 import com.ameya.models.Topic;
193
194 public interface TopicService {
195     List<Topic> getAllTopics();
196     void addTopic(Topic topic);
197     void updateTopic(Topic topic);
198     Topic getTopic(int id);
199     void deleteTopic(int id);
200 }
201 =====TopicServiceImpl.java=====
202 package com.ameya.services.impl;
203
204 import java.util.List;
205
206 import org.springframework.beans.factory.annotation.Autowired;
207 import org.springframework.stereotype.Service;
208
209 import com.ameya.daos.TopicDAO;
210 import com.ameya.models.Topic;
211 import com.ameya.services.TopicService;
212
213 @Service
214 public class TopicServiceImpl implements TopicService {
215
216     @Autowired

```

```

217     private TopicDAO topicDao;
218
219     @Override
220     public List<Topic> getAllTopics() {
221         return topicDao.getAllTopics();
222     }
223
224     @Override
225     public void addTopic(Topic topic) {
226         topicDao.addTopic(topic);
227     }
228
229
230     @Override
231     public void updateTopic(Topic topic) {
232         topicDao.updateTopic(topic);
233     }
234
235
236     @Override
237     public Topic getTopic(int id) {
238         return topicDao.getTopic(id);
239     }
240
241     @Override
242     public void deleteTopic(int id) {
243         topicDao.deleteTopic(id);
244     }
245
246     }
247
248     =====TopicController.java=====
249     package com.ameya.controllers;
250
251     import java.util.List;
252
253     import org.springframework.beans.factory.annotation.Autowired;
254     import org.springframework.web.bind.annotation.DeleteMapping;
255     import org.springframework.web.bind.annotation.GetMapping;
256     import org.springframework.web.bind.annotation.PathVariable;
257     import org.springframework.web.bind.annotation.PostMapping;
258     import org.springframework.web.bind.annotation.PutMapping;
259     import org.springframework.web.bind.annotation.RequestBody;
260     import org.springframework.web.bind.annotation.RestController;
261
262     import com.ameya.models.Topic;
263     import com.ameya.services.TopicService;
264
265     @RestController
266     public class TopicController {
267         @Autowired
268         private TopicService topicService;
269
270         @GetMapping(path="/topics")
271         public List<Topic> findAll(){
272             return topicService.getAllTopics();
273         }
274         @GetMapping(path="/topics/{id}")
275         public Topic findById(@PathVariable("id") int id) {
276             return topicService.getTopic(id);
277         }
278         @PostMapping(path="/topics/create")
279         public void create(@RequestBody Topic topic) {
280             topicService.addTopic(topic);
281         }
282         @PutMapping(path="/topics/update/{id}")
283         public void modify(@RequestBody Topic topic) {
284             topicService.updateTopic(topic);
285         }
286         @DeleteMapping(path="/topics/delete/{id}")
287         public void delete(@PathVariable("id") int id) {
288             topicService.deleteTopic(id);
289         }

```

```
290 }
291 =====
292
293 http://localhost:9001/topics/
294 http://localhost:9001/topics/create
295
296 {
297     "id": 103,
298     "topicName": "Spring",
299     "topicDesc": "Introduction To Spring 3 Days"
300 }
```