**Case study**

**Application name to be built for hands-on:** COVID Test Repository Service

**Description**

Through this COVID repository service, we will collect information about the test, people are taking. To hide the identity of the person who is taking the test, we will not collect any information about the person (like name or address). We will uniquely assign a subject number for every person who is taking the test for the first time. There after we will use this subject number to update the description of the latest test.

Information we will be uploading into the repository:

1. Subject number
2. Test description
3. Test location
4. Status
5. Date of the test

If an existing subject takes a new test, we will update the above information accordingly. We should also provide a service to fetch the information based on the subject number.

**Download link for general resources**

<https://github.com/digiterati-aug2020/resources.git>

**Day 1**

**Topics: Spring basics, Spring Boot, Spring REST**

**Core repository:** <https://github.com/digiterati-aug2020/day1>.git

**References:**

1. <https://spring.io/guides/gs/spring-boot/>
2. <https://spring.io/guides/gs/rest-service/>

**TO-DO**

We will build the basic core of the application without any database connectivity. But will enhance the application with database connectivity as we progress into other modules.

**Steps**

1. In the code which has been downloaded from Github open the project springbootbasics\_lab.
2. The structure and code have been built already and is available for reference in the package structure instructor.
3. The student is supposed to create the package under student and complete the following.
4. Create the following interface and classes:
   1. com.digiterati.basics.lab.student.repository.SubjectRepository (this will have a Map key being subject number and value being the Subject object)
   2. com.digiterati.basics.lab.student.model.Subject – with above values.
   3. SubjectController inside the package com.digiterati.basics.lab.student.controller that will have following methods:
      1. updateSubject (Subject): If the subject number does not exist, insert else update. [POST, Response message: success, produces: JSON, consumes: JSON]
      2. getSubject(String subjectNumber): Fetches subject info if it exists else returns error message.

[GET, Response: Subject, produces: JSON, param: subjectNumber=<>]

* 1. Run the web service using the embedded tomcat.
  2. Write a driver program to invoke the configuration class – so that the service is deployed. (inside com.digiterati.basics.lab.student.driver)
  3. Write a client program inside com.digiterati.basics.lab.student.driver to invoke both the services.
  4. Write test cases for SubjectRepository and controller methods.

**Day 2**

**Topics: Spring jdbc, jpa and data**

**Core repository:** [https://github.com/digiterati-aug2020/day](https://github.com/digiterati-aug2020/day1)2.git

**References:**

1. <https://spring.io/guides/gs/relational-data-access/>
2. <https://spring.io/projects/spring-data>
3. <https://spring.io/projects/spring-data-jpa>
4. <https://spring.io/projects/spring-data-mongodb>
5. <https://github.com/OAI/OpenAPI-Specification/blob/master/versions/2.0.md>

**TO-DO**

Let us plan to bring database connectivity to our “day 1” task.

**Steps**

1. In the code which has been download from Github open the project springdata\_lab.
2. The structure and code have been built already and is available for reference in the package structure instructor.
3. The student is supposed to create the package under student and complete the following. **This time, the student has to convert the repository into a JPA Compliant Spring Data repository.**
4. Create the following interface and classes:
   1. com.digiterati.data.lab.student.repository.SubjectRepository – this will access database to store/retrieve information.
   2. com.digiterati.data.lab.student.model.Subject – with above values.
   3. SubjectController inside the package com.digiterati.data.lab.student.controller that will have following methods:
      1. updateSubject (Subject): If the subject number does not exist, insert else update. [POST, Response message: success, produces: JSON, consumes: JSON]
      2. getSubject(String subjectNumber): Fetches subject info if it exists else returns error message.

[GET, Response: Subject, produces: JSON, param: subjectNumber=<>]

* 1. Run the web service using the embedded tomcat.
  2. Write a driver program to invoke the configuration class – so that the service is deployed. (inside com.digiterati.data.lab.student.driver)
  3. Write a client program inside com.digiterati.data.lab.student.driver to invoke both the services.
  4. Write test cases for SubjectRepository and controller methods.

**Day 3**

**Topics: Microservices part 1**

**Core repository:** <https://github.com/digiterati-aug2020/day3.git>

**References**

1. <https://spring.io/microservices>

**Steps**

1. Convert the web services created in Day 2 into microservices.

**Day 4 (Continuation day 3)**

**Topics: Microservices part 2**

**Core repository:** <https://github.com/digiterati-aug2020/day3.git>

**References**

1. <https://spring.io/microservices>

**Hands-on exercise**

1. Develop circuit breaker with Hystrix.
2. Develop sleuth and zipkin trace for sample restful web services