**Capabilities covered:**

1. Configure Environment to create RESTful web services using Spring Boot
2. Use Eclipse IDE for programming Java application
3. Analyze complex business scenarios and create a data model
4. Use Java to implement business layer of your application
5. Handle exceptions to maintain the normal flow of the application.
6. Configure Environment to create RESTful web services using Spring Boot
7. Use Hibernate/JPA framework to implement basic DAO layer
8. Implement Unit testing
9. Automate the build using Maven

**Note:**

1. Do not change the table structures.
2. Use appropriate data types and precisions for the variables
3. Handle all exception and alternate flows
4. **Loose coupling:** Create a separate Dao interfaces and implementation classes. Database interaction code should be only in DAO classes.

# Problem statement: Ticket Tracking System

Design and implement an internal ticket tracking application for a software company.

Create the below 3 end points:

1. open a new ticket
2. close an old ticket
3. view tickets with the turnaround

Database design with sample data is listed below. Do not add/ remove columns to this table, create the tables “EMPLOYEE” and “TICKETS” as listed.

**EMPLOYEE table: MID is Primary Key**

|  |  |  |  |
| --- | --- | --- | --- |
| **MID** | **EMPLOYEE\_NAME** | **HIRE\_DATE** | **DEPT** |
| M100100 | Karthik | 2004-1-10 | BIZC |
| M100101 | Swetha | 2011-12-12 | HiTech |
| M100102 | James | 2010-12-12 | ISSD |
| M100103 | Rahul | 2012-2-22 | ISSD |
| M100104 | Zaheer | 2012-3-19 | T&T |

**Note:** Create a Stored Procedure to insert records into **‘EMPLOYEE’** table. [Provide database scripts file for creating stored procedure and inserting records].

**TICKETS table: Ticket\_ID is Primary key, LOGGED\_BY and RESOLVED\_BY are foreign keys referencing MID of Employee table**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **TICKET\_ID** | **LOGGED\_BY** | **RAISED\_DATE** | **SEVERITY** | **TICKET\_DESC** | **RESOLVED\_BY** | **RESOLUTION** | **RESOLVED\_DATE** | **STATUS** |
| 1 | M100101 | 2012-10-3 | Major | AppV not working | M100103 | Need to restart with LAN cable | 2012-10-4 | CLOSED |
| 2 | M100100 | 2013-7-10 | Critical | Laptop restart problem | NULL | NULL | NULL | OPEN |

The below end points need to be developed:

* **“Log a Ticket”**
* **“Close a Ticket”**
* **“View Turnaround time”**

|  |  |  |
| --- | --- | --- |
| **Use case #1** | | **Log a ticket** |
| Pre-Conditions | | |  | | --- | | The table containing the details of “EMPLOYEES” should be created and pre-populated with a set of values manually from the back-end. | |
| Post-Conditions | | |  | | --- | | The complete information about employee’s ticket is stored in the application database. | |
| Data validations | | Ticket Date-Time should be in the specified format  Ticket Date-Time should be earlier than the current date-time |
| Business rules | | |  | | --- | | An employee can have more than one ticket logged in the system. | | |  | | --- | | The list of employees does not include employees from the ISSD department. | | |
| **Use case #2** | |  |
| Pre-Conditions | | |  | | --- | | A ticket has been logged in the application database. | |
| Post-Conditions | | The status of the ticket is changed from “Open” to “Closed”. |
| Main flow | | |  | | --- | | User provides the required details in form as end points and the following message is displayed: “Ticket: <Ticket-No> is closed”. [ Ex: Ticket 5 is closed] | |
| Business rules | | |  | | --- | | The current system timestamp is recorded as the time of closing the ticket | |
|  | |  |
| **Use case #3** | **View Tickets with the Turnaround Time** | |
| Pre-Conditions | |  | | --- | | A ticket has been closed in the application database. | | |
| UI Screen details | The details all tickets, with the turnaround time [total time taken between the submission and resolving] are displayed in the exact format given below [Given below is just a sample data]: | |