**Restful Web Services:**

1. Create Restful Web service that will process credit card approvals. To keep things simple, we’ll say that any credit card ending in an even number is considered valid and any card with an odd number is invalid. (This obviously isn’t true in real life; we’re just simplifying the problem for demonstration purposes.) In order to process the card, we’ll need to know the card number. We’ll send back a true if it’s valid—false if it’s invalid.

2. Create a RESTful Web service that accepts a date parameter and tells you how old you will be on that date. Create a client that can gather date of birth from the user and display the answer.

3. Create a RESTful Web service that doubles any integer value passed in. Create a client that can gather integer value from the user and display the answer.

4. Create a RESTful Web service that converts your name to all lowercase letters when you pass it in by reference. Create a client that can gather String from the user and display the answer.

5. Create a calculator RESTful Web service and a client on your computer. Make the Web service support add() and subtract() methods. Create a client that can gather this Information from the user and display the answer.

6. Create Employee Management RESTful web service, which will provide CRUD (Create, Retrieve, Update, Delete) operations, you can complete integration with hibernate and oracle. Crate a client that can gather this information from the user and display the result.

7. Create an Account RESTful web service and a client on your computer. Make the web service support deposit(), withdraw() and getBalance() methods. Create a client that can gather this information from the user and display the result.

8. Create Simple object Album object which have attributes like title and singer, and user JACKSON to convert object to /from JSON. Create a Web service Client to access the Album details.

9. Create a RESTful Web Service to support the below requirements:

a. The Model/Entity class is as follows:

|  |  |  |
| --- | --- | --- |
| Class : Movie |  |  |
| movieId | String | E.g.: M001 |
| movieName | String | E.g.: Top Gun |
| movieActor | String | E.g.: Tom Cruise |
| movieCollection | Float | E.g.: 350000000.00 |

b. Store some sample Movie objects in a collection (Map/List) (around 10 movie objects)

b. The Root resource class should provide the below “GET” methods:

* “GET” method that returns all the movie objects in “text/xml” format
* “GET” method that returns all the movie objects in “application/xml” format
* “GET” method that returns all the movie objects in “application/json” format
* “GET” method that takes a “movieId” as the argument and returns only the Movie details for that MovieId in “application/xml” format.

c. Hint: Annotate the Model class with relevant JAXB annotations so that the above GET methods can return the objects directly and the objects will be automatically marshalled to XML/JSON by JAXB.

(You could use any JAX-RS RI like Jersey or RESTEasy for this program)