GALWAY-MAYO INSTITUTE OF TECHNOLOGY

Department of Computer Science & Applied Physics

B.Sc. Software Development – Distributed Systems (2018/2019)
ASSIGNMENT DESCRIPTION & SCHEDULE

A RESTful Car Hire Booking System

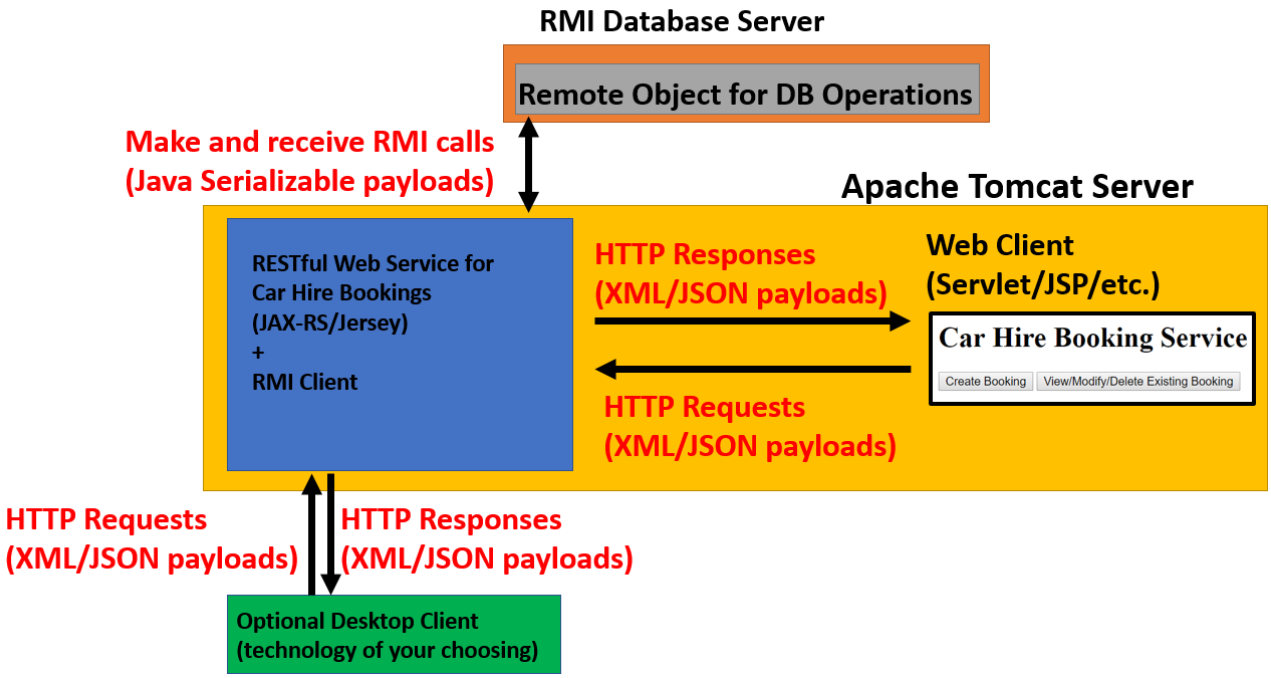
***Note*: This assignment will constitute 40% of the total marks for this module.**

Your solution must be submitted by midnight on Monday 3rd December 2018.

1. Overview

You are required to use the JAX-RS/Jersey, Java RMI and JAXB frameworks to develop a simple Car Hire Booking System. A Web Client page should provide users with the ability to Create/Modify/Update/Delete bookings for a specific vehicle for a given set of dates. The Web Client will interact with a RESTful JAX-RS Web Service for bookings which is deployed on Apache Tomcat Server. The RESTful Web Service will act as a RMI client to a RMI Database Server which will handle persistence.

The following diagram depicts the overall system architecture:



The point of this exercise is to give you some "hands-on" programming experience with the key technologies which are covered in this module (e.g. RESTful web services, Inter-Process Communication, Data Externalisation). It is expected that you will conduct a significant amount of independent study on the specified technologies as part of this project (beyond what has been covered in lectures and labs).

2. Minimum Requirements

Your implementation should include the following features (at a minimum):

(A) Simple Web Client (Java JSP/Servlet or .NET equivalent if preferred)

1. A **Web Client** will act as a GUI for the entire Car Hire Booking System. This GUI will allow a booking to be Created, Read, Updated or Deleted. (N.B. do not spend too much time on the styling and layout of the **Web Client**; basic unstyled HTML forms, tables and buttons etc. will suffice as this is not a client-side web development project). The Web Client will communicate with the RESTful Web Service below, using XML for marshalling and unmarshalling of data.

(B) RESTful Web Service (JAX-RS/Jersey)

1. Design a **RESTful Web Service** using JAX-RS/Jersey which will act as the gateway for all clients which wish to use the Car Hire Booking System. Clients will be able to access CRUD functionality for car hire bookings using the GET, POST, PUT and DELETE methods. This class will be responsible for marshalling/unmarshalling data to/from XML for all Web Client requests/responses. This class will also act as a client for the **RMI Database Server**.

(C) Data Modelling

1. An appropriate data model will be required for all classes/entities which are part of a car hire booking (e.g. Customer, Vehicle, Booking). This data model will be in the form of an XML Schema Definition. The xjc (XML to Java Converter) utility may then be used to generate the appropriate Java classes from the schema.

(D) RMI Database Server

1. A remote interface called ***DatabaseService*** should expose remote methods which provide CRUD (Create, Read, Update and Delete) functionality for each of the entities which have you modelled. This interface definition must be available to the **RMI Database Server** and the **RMI Client**. Objects sent or received via RMI will be required to implement the java.io.Serializable interface.
2. An implementation of the ***DatabaseService*** interface called ***DatabaseServiceImpl*** will handle persistence and CRUD functionality using a database of your choice (e.g. JDBC).
3. ***ServiceSetup*** will contain a main method which instantiates ***DatabaseServiceImpl*** and binds the Remote Object into the RMI registry using the name "databaseservice".

3. Deployment and Delivery

The project must be submitted by midnight on Monday 03rd December 2018 using both Moodle and GitHub.

Use the package name ie.gmit.sw throughout your project

- **GitHub**: All your source code should be available in a GitHub repo for this assignment. Your repo will also have a detailed readme containing a description of the entire application, extra functionality you have added and the steps required to deploy and run your application.

- **Moodle**: submit the URL of your GitHub repo using the Moodle upload facility (under "Project Upload") on or before the due date. Please note that any edits which are made to your GitHub repo after the submission deadline will not be considered for marking.

File	Description
README.txt	Contains a description of the application, extra functionality added and the steps required to run the application.

4. Extra Features

20% of the project marks will be awarded for documented (relevant) extra features. Some example extra features are listed below. Note it is not necessary to implement all of these additional features.

- Add different client roles (e.g. customer and manager, where specific functionality like viewing all bookings is only available to managers
- Add multithreading functionality, so that multiple RMI clients can make queries to the RMI Database Service concurrently. Could be implemented using e.g. Java Thread Pools.
- Create a desktop client which interacts with the RESTful web service. This may be implemented using any technology that you choose.
- Package the resources created into .jar and .war files as shown below:

File	Description
booking-server.war	A Web Application Archive containing the resources shown under Tomcat Web Application. All environmental variables should be declared in the file WEB-INF/web.xml. You can create the WAR file with the following command from inside the "booking-server" folder: **jar –cf booking-server.war ***
database-service.jar	A Java Archive containing the RMI Dictionary Service and a ServiceSetup class with a main() method. The application should be run as follows: **java –cp ./database-service.jar ie.gmit.sw.ServiceSetup** You can create the JAR file with the following command from inside the "bin" folder of the Eclipse project: **jar –cf database-service.jar ***

5. Marking Scheme

Marks for the project will be awarded using the following criteria:

Marks	Category
(15%)	Web Client
(20%)	RESTful Web Service
(15%)	Data modelling
(20%)	RMI Database Service (including RMI client functionality)
(10%)	Packaging & Distribution (Moodle and GitHub) Documentation (Readme.txt and code comments)
(20%)	Documented (and relevant) extra features beyond the minimum requirements outlined above

Each of the categories above will be scored using the following criteria:

- 0–30%: Not delivering on basic expectation
- 40-50%: Meeting basic expectation
- 60–70%: Tending to exceed expectation
- 80-90%: Exceeding expectations
- 90-100%: Exemplary

7. Copying & Plagiarism

Plagiarism is passing off the work of another person as one's own.

While you are allowed to collaborate with your classmates and review online and print resources for high-level problem solving and background research, you are each expected to code, write and complete this assignment **individually**. If you use material from an external source (e.g. textbook, webpage, lecture notes) as part of your answer(s), you must explicitly acknowledge the source of the material.

Please see Section 4 of the GMIT Code of Student Conduct 2018/2019 for further information on plagiarism: https://www.gmit.ie/sites/default/files/public/general/docs/7-1-code-student-conduct-2018-2019.pdf

Please also familiarise yourself with the Institute's Policy on Plagiarism: https://www.gmit.ie/sites/default/files/public/general/docs/3-5-plagiarism.pdf

Plagiarism is a serious academic offence and may lead to a loss of marks and/or disciplinary proceedings if it is detected in your submission.