FAST NUCES – IPT – Assignment 2

Due: Saturday 21st October 2017

Please submit via Slate. Submit a single .rar/.zip file named ‘IPT Assignment 2 <Reg #>’

PLEASE DO NOT COPY

We are asked to make a small application to maintain vehicle data by a ride hailing business. Each vehicle has the following information:

* Vehicle registration number, an 8 character alphanumeric which is unique.
* Vehicle make & model (Toyota is Make, Corolla XLi is Model)
* Vehicle year (Year of manufacturing)
* Vehicle Type (corresponds to the ride type; E.g., Economy, Business, Luxury/VIP, Large Capacity)
* IMEI number of the tracking device, optional
* Accessories: Optional list of fittings/accessories/features. Example includes (Rear air-conditioner, Cup-holders, Sun/Moon Roof, Fog lights, TV, Navigation system, Rear-view cameras, proximity sensors, Leather seats, Cold storage, etc.)
* One or more drivers assigned to the vehicle. For each driver, store the following information:
  + Driver name
  + CNIC
  + Driver Rating

1. Create a form based application to maintain this small application.
2. Come up with a design whereby the data persistence mechanism is decoupled from the application, i.e., the application may use XML or JSON or flat files or NoSQL or RDBMS as data storage mechanism
3. Implement the application using both XML and JSON; Generate information about 1,000 vehicles
4. List down all assumptions that you may have taken.
5. Minimum functionality of application:
   1. Add, Edit, Delete Vehicles
   2. Add, Edit, Delete drivers for a particular vehicle
   3. Search vehicle with following search criteria:
      1. Make & Model
      2. Year of manufacturing range
      3. Filter based on accessory
      4. Filter based on driver ratings
6. Use LINQ for collections in the implementation

Grading Criteria:

Assignment will be graded according to the following general criteria:

* The solution adequately addresses the requirements for the assignment.
* The program compiles, links, and executes.
* The program runs correctly (or at least appears to be correct based on testing done by the grader).
* The program is easy to read and to understand, i.e., it is well commented and adheres to programming conventions. For example, method and object names are appropriate (or as defined by the assignment), and all potentially confusing/complex program code (including non-trivial if-statements, loops, method calls, etc.) is adequately documented.
* The general design of the program is clear and reasonable. For example, the program makes good use of classes and methods and is implemented in a sensible, understandable way.

Assignment will receive a grade according to the following table:

|  |  |
| --- | --- |
| Quality of Submitted Solution | Points (%) |
| The solution meets all criteria well | 100 |
| The solution meets most criteria, but there is some room for improvement | 80 |
| The solution is just satisfactory; it meets some criteria but there is significant room for improvement | 40 |
| The solution is not acceptable | 0 |

Zero will be awarded for assignment that is copied (either amongst students or from another source)