# IN710 OOSD 2017 Assignment 2 - Trivia Quiz Challenge

For this assignment you will build an online Trivia Quiz game that allows users to participate in monthly "tournaments". The application will dynamically fetch trivia questions from the Open Trivia Database (opentdb.com), using its RESTful API. At each game iteration, the user is presented with 10 randomly selected questions from the database. Their score (number correct out of 10) is preserved in persistent storage (on bitdev). Users are able to view a "high scores" display showing the average score in the current tournament for all registered users. A new tournament starts each month.

NB: No multi-threaded or asynchronous programming is required for this assignment. LINQ should be used, as appropriate.

**The application must:**

|  |  |
| --- | --- |
| **Requirement** | **Comments** |
| Be implemented in ASP.NET MVC5 using "database first" ORM. | * Do not use Core or VS 2017. The application must open and run correctly on the standard student machine image. There will not be time for me to port (or ask you to port) your assignment before grades are due. If your application does not run on the standard image, it will receive an automatic 0 marks for robustness. |
| Have a well-designed, easy-to-use interface for user input and display.  The interface should be designed **to run full-screen** on a 1920 x 1080 display. Responsive design is not required.  The look and feel should be attractive and appropriate to the context. You may use whatever CSS you wish (including frameworks). | * It must be clear to the user how to access all functionality. * Navigation must be clear and efficient. * The opportunity for user errors must be minimised. * User errors, when they do occur, must be dealt with gracefully. * It is sensible to do some user-testing with friends and/or flatmates to make sure your interface is well-designed. |
| Maintain multiple users.  Allow login for existing users.  Allow registration for new users, which will add them to the database. | * Industrial strength security is not required. You may manage your users with a simple username & password system. * It is acceptable to require users to type in both their username and their password. * It is acceptable to store the passwords in your database as plain text. (If this makes you really uncomfortable, feel free to explore the C# facilities for hashing passwords, but this is not required for this assignment.) |
| Parse and use the JSON data returned by the API. | * Use JSON.NET |
| Maintain persistent storage in a relational database on bitdev.ict.op.ac.nz | * For the minimum required functionality, this is a simple database; my solution has only two tables in it. * If you haven't taken DB2 and need help with the ERD, ask me. |
| Present multiple-choice questions fetched dynamically from opentdb.com | * See Extra Credit below for additional options |
| Present a set of 10 questions as one game.  Each question must be presented as a separate screen (i.e. **do not** display multiple questions on a single screen). |  |
| Provide feedback, after each response is submitted, as to whether the response is correct for that question. |  |
| Allow the user to decide when to proceed to the next question. | * That is, there is no clock or time limit of any kind. |
| When a game (i.e. a set of 10 questions) is finished, present the user with their total score for that game (number correct out of 10). |  |
| Display a "high scores" screen for the current "tournament". | * A new tournament starts each month. So the scores screen displays the results only of those games played during **the current month**. * The scores screen should show the average score for each registered player, for the current month, ordered from highest to lowest. * See Extra Credit below for additional options. |

**Extra Credit Options: Marks earned for extra credit are added to your total project mark. Maximum mark is therefore 104%.**

|  |  |
| --- | --- |
| **Feature** | **Maximum Extra Credit** |
| Include a mixture of multiple choice and true-false format questions in your games (opentdb provides both). Screen displays must be appropriate for each of the two different question formats. | 2% |
| Allow the user to select a previous month, and see the "high scores" table for that month. Screen controls must be correct (as judged by me; Hint: no typing) to receive credit for this feature. | 2% |

**Seed Data:**

Your submission must have the seed data specified below. Before you submit your final version, clear all your tables and refresh with these data, as needed. Your robustness mark will include comparison to solutions based on a system that has **exactly** these data in it, so be very careful to also have exactly these data.

*Users:* If you set your UserID field as IDENTITY and insert your users in this order, the IDs will come out correctly. Be sure to use all lower-case, as shown.

|  |  |  |
| --- | --- | --- |
| **User ID** | **User name** | **Password** |
| 1 | fred | fred |
| 2 | barney | barney |
| 3 | wilma | wilma |
| 4 | betty | betty |

*Results:* The accompanying file "insertResultsSQL.txt" contains 50 SQL INSERT statements, each corresponding to one game play iteration. This is the SQL required to insert the seed data ***into my database****.* The values in each INSERT statement are playerID, score, and date, in that order. (The table into which these records are being inserted also has an IDENTITY field called ResultID, but that value is, of course, not specified in the INSERT statement.) You will (almost certainly) need to modify these statements to match your database. Do whatever is required to insure that you have the same set of results seeded into your database when you submit your assignment.

**Marking Schedule:**

|  |  |
| --- | --- |
| **Component** | **Weight** |
| Architecture (models, views, controllers) | .15 |
| Database correctness | .05 |
| Code quality (includes commenting) | .30 |
| Functionality & Robustness | .30 |
| User Interface (usability) | .15 |
| User Interface (look and feel) | .05 |

Submission must be via your private IN710 GitBucket repo. A minimum of two commits per week is required.