|  |  |
| --- | --- |
| MIT%20Award%20smal(bw) | **BDT – Bachelor of Digital Technologies**  **502.632 - Full-Stack Web Development**  **TEST 02**  **Assessment Cover Sheet** |
| Course | 502.632 - Full-Stack Web Development |
| Assessment name | TEST02 |
| Assessment type | Practical Test – "Open Book" – reference allowed to The Internet, notes, examples, Canvas Resources. Information exchange is not allowed during the test, except with the test supervisor(s). See details in the "Statement" below. |
| Date | Thu, 11 Apr 2019 |
| Course contribution | **30% -** (Test has 30 marks - a test mark is a course mark). |
| Time Allowed | 150 minutes |
| **Student ID:** |  |
| **Student NAME:** |  |
| **Acknowledgement of Database Copyright** I understand that the "Tennis" sample database is from a published copyrighted work and we may only use it for "exam purposes" which is in this case means this test only.  Attribution: Rick F. van der Lans  **Statement of Valid Authorship** I hereby confirm that these test answers are my own work done in the test time. This means that to the best of my knowledge and belief, this test contains only my own created and assembled responses to these particular questions without communication with any other person, or AI equivalent to a person, except the course lecturer. I make this statement fully understanding that, should it be found false, I will, in most circumstances, receive zero marks for this test.  Signed by student: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date: \_\_\_\_\_\_\_\_\_\_\_\_\_ | |
| **Relates to all 4 Learning Outcomes**   1. Demonstrate an understanding of the architecture of web applications including the roles of clients and servers and the use of protocols for interaction. 2. Learn and apply a server-side programming platform and/or language. 3. Implement dynamic web user interfaces using HTML, CSS, JavaScript, and AJAX. 4. Implement and publish web-based software solutions that interact with a variety of data sources and public APIs. | |

**PRACTICAL TEST02 - Tennis Club Reports and Information**

Download the file provided **"Tennis\_Test.zip"** from the Canvas page provided for this test.  
Download or move this to folder **"C:\Users\<your username>"** on your machine (or lab machine).  
Unzip there by right-click and "Extract Here".  
Rename the extracted folder to **"Tennis\_<your name>"**

At the END of the test, zip the folder **"****Tennis\_<your name>"** and hand-in this zip to Canvas.

**SUPPLIED is a STARTER APP**Model Classes have already been created for you.

Controller "PlayersController" and its Views have already been created for you.

Check that "PlayersController" is working.

**Question ONE** **[7 marks]**  
1-1 Provide a Players Search. [3 marks]  
 Do this by modifying the Players Controller and View code   
 to provide a search in the fields "LastName", "AddressStreet" and "Locality".  
 The user enters a single Search String and your code looks for a match in any  
 of these 3 fields.

1-2 Make this an auto-suggest search [2 marks]

1-3 Make this an auto suggest search with user-friendly display of the suggestions   
 as clickable list items under the search input field. [2 marks]

**Question TWO [5 marks]**

Add a method "ReportResult” to the “PlayersController” from Question ONE

In that method, create an SQL statement, with LINQ as an acceptable alternative, to display the following fields from tables "Player" and "Result". The purpose of this statement is to add value to table "Result" by displaying Player names and other readable values.

The first part of this SQL statement is provided below. You need to finish it off. [1 mark]

SELECT ResultId, Initials, LastName, DateOfBirth, Gender, YearJoined, Won, Lost, TeamId

. . .

Analyse your completed SQL (or LINQ), and its data fields, to create a ViewModel.

Then use that ViewModel with the SQL(or LINQ) to create an on-screen report

by adding a View to method “ReportResult”. [2 marks]

Improve this report by calculating the

Total of games Won and displaying this total at the bottom of the Won column.  
Total of games Lost and displaying this total at the bottom of the Lost column.

[2 marks]

**Question THREE [3 marks]**

Add a method "ReportCommittee” to the supplied “PlayersController”.

In that method, create an SQL statement, with LINQ as an acceptable alternative, to display the following fields from tables "CommitteeMember", "Player" and "CommitteePosition". The purpose of this statement is to add value to table "CommitteeMember" by displaying names and other readable values instead of Id numbers.

The first part of this SQL statement is provided below. You need to finish it off.   
 [1 mark]

SELECT CommitteeMemberId, Initials, LastName, Description, Phone, DateBegin, DateEnd

. . .

Analyse your completed SQL (or LINQ), and its data fields, to create a ViewModel.

Then use that ViewModel with the SQL(or LINQ) to create an on-screen report

by adding a View to method “ReportCommittee”. [2 marks]

**Question FOUR [4 marks]**

Analyse the SQL supplied below for this question, and its data fields, to create a ViewModel.

Then use that ViewModel with this SQL (or equivalent LINQ) to create an on-screen display

by adding a Method "ReportPenalties" and its View to the supplied "PlayersController".

SELECT PenaltyId, Initials, LastName, Phone, Locality, DateOfPayment, Amount

FROM Penalty INNER JOIN Player ON Penalty.PlayerId = Player.PlayerId

Improve this report by calculating the Total of Amounts and displaying it at the bottom of the Amount column.

All currency amounts need to display with 2 decimal places like 47.32, 47.30 and 47.00

**QUESTION FIVE** **[5 marks]**  
Demonstrate "AngularJS" by displaying the same report as Question FOUR above   
using AngularJS  
For the Server-Side code, create a new Method "ReportPenaltiesAngular" in "PlayersController"   
with a copy of your code from Question FOUR as the starter code.

**QUESTION SIX [6 marks]**

SUPPLIED is a plain text file *“Lawn Tennis for Ladies by Mrs Lambert Chambers.txt”*

The text is divided into a 5 blocks of text information.

6-1 Include this text in your working app by rewriting the existing "About" page   
 associated with the "HomeController".

Apply headings and paragraphs.

6-2 Apply the Bootstrap resource already existing in MVC, to this page  
 Produce a result that has:

For large screens, 1st block as header plus 3 columns   
The first column is half the parent row width, the others are each one sixth row width.

|  |  |  |  |
| --- | --- | --- | --- |
|  | | | |
|  |  |  |  |

6-3 For medium screens. - the large screen half-width column becomes full width,   
 the others are each one third row width.

|  |  |  |
| --- | --- | --- |
|  | | |
|  | | |
|  |  |  |

6-4 For small screens – the first 3 blocks of text are full row width,   
 with the other 2 as half-row-width.

|  |  |
| --- | --- |
|  | |
|  | |
|  | |
|  |  |

For extra small screens the single column default applies.

6-5 Select and display a suitable Youtube Video in the header block.

6-6 Customise Bootstrap by using the additional custom file "Site.css"   
 to give different background colours to each block of text and better height displays.

**END OF TEST**