

Name	OFFICIAL (CLOSED) \ NON-SENSITIVE		
Student ID			
Assessment Venue		Seat Number	



C236 Web Application Development in .NET

AY2019 Semester 2 End-Semester Exam (ESE)

Instructions to student:

- 1) Do not turn over this question paper until you are instructed to do so by the invigilator.
- 2) Write your name, student ID, assessment venue and seat number in the table provided at the top of each page.
- 3) For this question paper, there are 22 pages (including this cover page).
- 4) For this assessment, you are allowed to:
 - Refer to bounded hardcopy notes.
 - Refer to materials stored in your laptop.
 - Have a blank piece of paper for rough working purpose. (Note that the sheet of rough working paper will not be accepted for submission at the end of the assessment.)
- 5) For this assessment, you are **NOT** allowed to:
 - Share any material, such as calculators, with another student.
 - Communicate with any person other than the invigilator.
 - Use any communication devices such as handphone or smart watches while at the assessment venue.
- 6) All rules and regulations pertaining to summative assessments and academic integrity stated in the Student Handbook shall also apply.

This segment is to be used by staff grader(s) only.		
Question Number	Marks Awarded	Max Marks
Section A		
1 - 10		10
Section B		
11		3
12		7
Section C		
13		10
Section D		
14		3
15		9
16		6
17		5
18		7
Total		60

This segment is to be used by the invigilator only.		
Please tick the box below if the student has done part of the assessment online:	Invigilator's Name:	Invigilator's Signature:
<input type="checkbox"/> Partially done online		

SECTION A

Question 1

Consider the following unsafe code for authenticating users.

```
String sql = @"SELECT * FROM AppUser
                WHERE userpw=HASHBYTES('SHA1','{1}') AND userid='{0}';
String select = String.Format(sql, userid, passwd);
DataTable dt = DBUt1.GetTable(select)
```

Which of the following userid/passwd combinations can a hacker use to login into Alfred's account without knowing his password?

- A. **userid** = "hahaha"; **passwd** = "') OR userid='alfred' --";
- B. **userid** = "alfred'--"; **passwd** = "hahaha";
- C. **userid** = "hahaha"; **passwd** = "alfred";
- D. **userid** = "'--"; **passwd** = "alfred";

[____]

Question 2

With respect to using a Layout in ASP.NET Core projects, which of the following is **FALSE**?

- A. It allows a common site template to be defined
- B. It includes common user interface elements
- C. Only one layout can be used for each Web application project
- D. It reduces duplicate code in views

[____]

Question 3

Which validation attribute would you use to validate whether an ID has already been taken by another user?

- A. [Required]
- B. [RegularExpression]
- C. [Remote]
- D. [Range]

[____]

Question 4

The connection string to connect to a cloud database is stored in

- A. Startup.cs
- B. libman.json
- C. launchSettings.json
- D. appsettings.json

[____]

Question 5

Which file needs to be changed in order to start a Web application to a specific controller/action?

- A. Appsettings.json
- B. launchSettings.json
- C. _ViewStart.cshtml
- D. _ViewImports.cshtml

[____]

Question 6

Which of the following is **FALSE** with regard to TempData?

- A. It is a dictionary that stores key-value pairs
- B. It can be used to pass data from a controller to a view
- C. TempData values are read-only
- D. It is used to pass data from a controller to another controller via redirection

[____]

Question 7

Which file should the directive @addTagHelper be placed in so that its html tag helpers are available to all views in an ASP.NET Core web application?

- A. _ViewStart.cshtml
- B. _ViewImports.cshtml
- C. _Layout.cshtml
- D. _Index.cshtml

[____]

Question 8

Which of the following statements is **FALSE** with regard to Model Binding in ASP.NET core

- A. Allows data from HTTP requests to be mapped to action method parameters easily
- B. Uses the directive @model and the tag-helper attribute asp-for to accomplish the mapping
- C. Offers productivity gains in data conversion and validation
- D. Requires the use of the tag-helper attribute asp-validation-summary

[____]

Question 9

A Bootstrap CSS class used to create a <div> occupying 25% of the screen of a large device is:

- A. col-lg-25
- B. col-lg-3
- C. col-lg-1/4
- D. col-lg-4

[____]

Question 10

Which option is NOT essential to enable sorting, searching and paging on a jQuery DataTable?

- A. paging: true
- B. ordering: true
- C. info: true
- D. searching: true

[____]

SECTION B

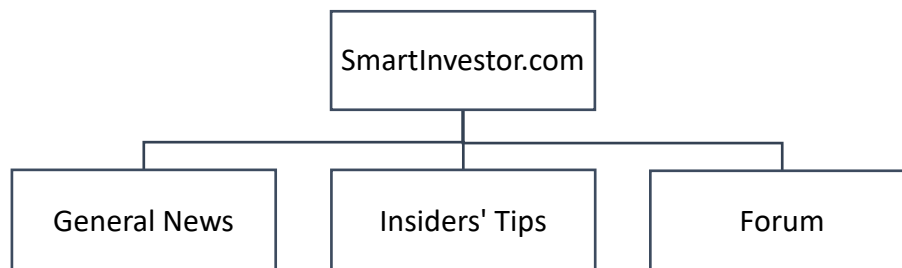
Question 11 (3 marks)

SmartInvestor.com is revamping their Web application using ASP.NET Core. Their Website has three kinds of users which are **visitors**, **login users** and **subscribers**. **Visitors** are those who have not registered. **Login users** are those who have registered their details with the website. **Subscribers** are those who have registered and have paid a monthly subscription.

SmartInvestor.com Web application comprises three sections, **General News**, **Insiders' Tips** and **Forum**. Visitors can only access pages in **General News**. Login users can access to the **Forum**. Only subscribers can view the pages in the **Insiders' Tips** section.

Sections	Visitors	Login Users	Subscribers
General News	✓	✓	✓
Insiders' Tips			✓
Forum		✓	✓

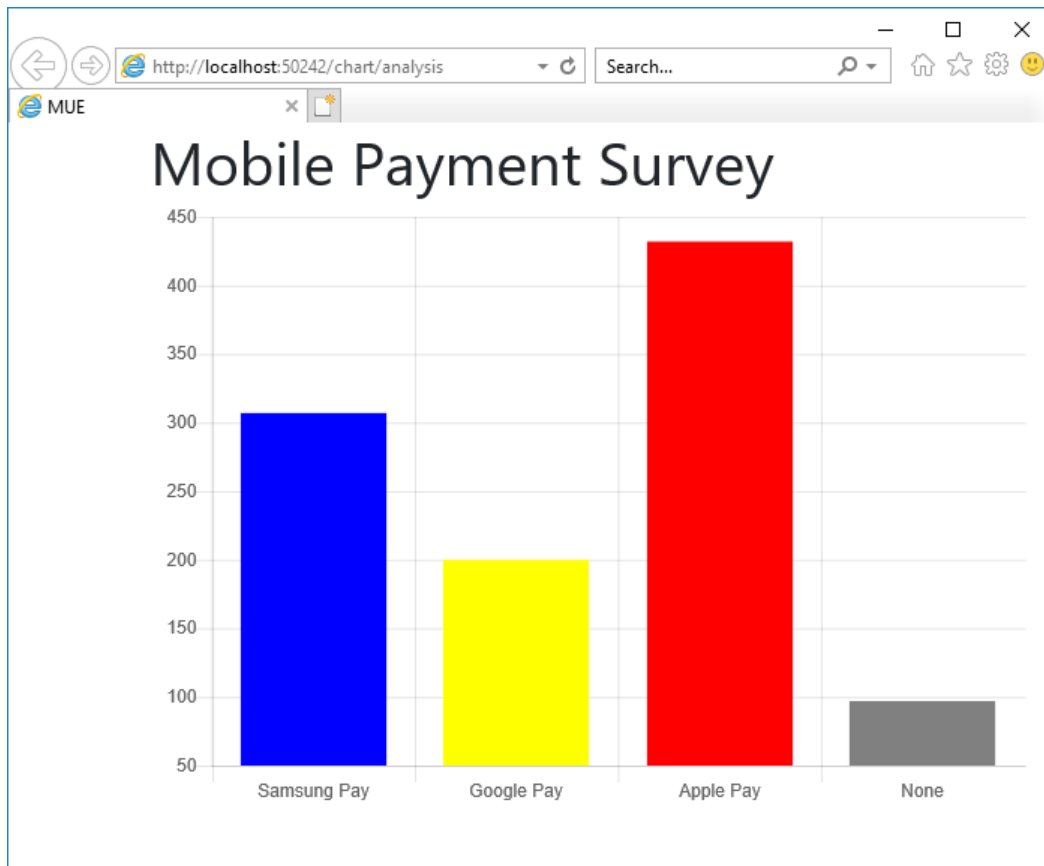
Each section has a different look and feel and the menu structure may change based on the type of user. However, the pages within each section have similar look and feel.



List three ways you can use layouts to simplify the development of SmartInvestor.com web application in terms of user interface and navigation control.

Question 12 (7 Marks)

A web page uses the Chart.js library to generate the following chart.



The controller code for the above web page is given as follows

```
public class ChartController : Controller
{
    public IActionResult Analysis()
    {
        string[] labels = new[] { "Samsung Pay", "Google Pay", "Apple Pay", "None"};
        ViewData["Platforms"] = labels;
        string[] colors = new[] { "blue", "yellow", "red", "gray" };
        ViewData["Bars"] = colors;
        int[] data = new[] { 307, 200, 432, 97};
        ViewData["Numbers"] = data;
        return View();
    }
}
```

Complete the code for the associated view

```
@section MoreScripts {
    <script>
        $(_____).ready(function () {
            new Chart(
                document.getElementById("_____").getContext('2d'),
                {
                    type: _____,
                    data: {
                        labels: @Json.Serialize(_____),
                        datasets: [{
                            backgroundColor:
                                @Json.Serialize(_____),
                            data: @Json.Serialize(_____)
                        }]
                    },
                    options: {
                        responsive: false,
                        legend: {
                            display: _____
                        }
                    }
                }
            );
        });
    </script>
}

<h1>Mobile Payment Survey</h1>
<canvas id="stat" width="600" height="400"></canvas>
```

SECTION C

Question 13 (10 Marks)

A web application enables a course administrator to email students their certification results.

Email Certification Results

Course :

Email successfully sent.

Upon clicking the **[Send Email]** button, the web application retrieves all certification records for the course entered. For each certification record, the final score is calculated. The details of the certification are then emailed to the relevant student.

For example, if **C11A** is entered as a Course and the button **[Send Email]** is clicked, the following emails will be sent to the respective students.

<p>Hi Cheong Sau Lai,</p> <p>Certification result for Course C11A</p> <p>Practical : 76</p> <p>Assignment : 60</p> <p>Examination : 59</p> <p>Final : 62.7</p> <p>Administrator</p>	<p>Hi Abdul Aziz Shajahan,</p> <p>Certification result for Course C11A</p> <p>Practical : 59</p> <p>Assignment : 62</p> <p>Examination : 61</p> <p>Final : 60.9</p> <p>Administrator</p>
---	--

The **Practical**, **Assignment** and **Examination** are each scored from 0 to **100%**. These scores are stored in the **Certification** table. The **Final** score is calculated by adding **20%** of the Practical score to **30%** of the Assignment score and **50%** of the Examination score.

The following diagram is a screen shot of the web application when the entered **Course** code is not found in the database.

Email Certification Results

Course :

Course Not Found

Complete the code for the **SendEmail** action in the **Certification** controller.


```

[HttpPost]
public IActionResult SendEmail(IFormCollection form)
{
    string course = form["Course"].ToString().Trim();

    string select = @"SELECT * FROM CERTIFICATION WHERE course='{0}'";
    DataTable table = DBUt1._____ (select, course);

    if (_____ == 0)
    {
        ViewData["Message"] = "Course Not Found";
        return View();
    }

    string title = "Your Certification Results";
    string template = @"Hi {0},<br/>
        <br/>
        Certifcation result for Course {1}<br/>
        Practical    : {2}<br/>
        Assignment   : {3}<br/>
        Examination  : {4}<br/>
        Final        : {5}<br/>
        <br/>
        Administrator";

    foreach (_____ row in table.Rows)
    {
        string email    = row["Email"].ToString();
        string name     = row["Name"].ToString();
        int practical   = (int)row["Practical"];
        int assignment  = (int)row["Assignment"];
        int exam        = (int)row["Exam"];

        double final = (_____ * 0.2) +
            (assignment * _____) +
            (_____ * 0.5);

        string message = _____ (template,
            name, course, practical, assignment,
            exam, final);
    }
}

```

```
string result = "";
EmailUtl.SendEmail(_____,
                  _____,
                  _____, out result);

}
ViewData["Message"] = "Email successfully sent.";
return View();
}
```

SECTION D

The CREATE TABLE and sample INSERT statements for the **UsedCar** table are as follows.

```
CREATE TABLE UsedCar
(
    Sno          INT PRIMARY KEY IDENTITY,
    CModel       VARCHAR(20) NOT NULL,
    Make         VARCHAR(20) NOT NULL,
    VRegno       VARCHAR(10) NOT NULL,
    Manufact     DATE NOT NULL,
    Mileage      INT NOT NULL,
    Sold         BIT NOT NULL
);

SET IDENTITY_INSERT UsedCar On;
INSERT INTO UsedCar(Sno, CModel, Make, VRegno, Manufact, Mileage, Sold) VALUES
(100, 'Accent 1.5', 'Hyundai', 'SHQ543A', '2010-06-28', 159812, 1),
(101, 'Elantra 1.8', 'Hyundai', 'SLX12Z', '2012-06-28', 200300, 0),
(102, 'Stinger 2.0', 'Kia', 'SMA5522B', '2016-05-01', 99890, 0),
(103, 'Cerato 1.6', 'Kia', 'SHQ543C', '2015-08-10', 178229, 0);
SET IDENTITY_INSERT UsedCar OFF;
```


The following diagram shows a screen shot of the web page at **UsedCar/Index**. Using this web page, a user can view all the used car records.

Used Cars

Sno	VRN	Model	Make	Mileage	Manufacture	Sold	Action
100	SHQ543A	Accent 1.5	Hyundai	159812 km	2010 Jun	Yes	Update
101	SLX12Z	Elantra 1.8	Hyundai	200300 km	2012 Jun	No	Update
102	SMA5522B	Stinger 2.0	Kia	99890 km	2016 May	No	Update
103	SHQ543C	Cerato 1.6	Kia	178229 km	2015 Aug	No	Update

If the user clicks on any of the **Update** links in the Action column, it will allow the user to update the details of the user car at **UsedCar/Update/{id}**. The following diagram illustrates the screen shot of UsedCar/Update/101.

Update Used Car Record

Vehicle Reg# :	<input type="text" value="SMA4325X"/>
Model :	<input type="text" value="Elantra 1.6"/>
Make :	<input type="text" value="Hyundai"/>
Mileage :	<input type="text" value="100000"/>
Manufacture :	<input type="text" value="2015-03-01"/> 
<input checked="" type="checkbox"/> Sold	
<input type="button" value="Update"/>	

After the user has updated the record and clicks the [Update] button, it will take the user back to the page at **UsedCar/Index** with a success message as illustrated as follows.

Used Cars

Used Car Record Updated

Sno	VRN	Model	Make	Mileage	Manufacture	Sold	Action
100	SHQ543A	Accent 1.5	Hyundai	159812 km	2010 Jun	Yes	Update
101	SMA4325X	Elantra 1.6	Hyundai	100000 km	2015 Mar	Yes	Update
102	SMA5522B	Stinger 2.0	Kia	99890 km	2016 May	No	Update
103	SHQ543C	Cerato 1.6	Kia	178229 km	2015 Aug	No	Update

Question 14 (3 Marks)

The following page is generated by the action **Index** of the controller **UsedCar**. You are to provide the Razor code to output the columns in the exact format shown in the rectangle.

Used Cars

Used Car Record Updated

Sno	VRN	Model	Make	Mileage	Manufacture	Sold	Action
100	SHQ543A	Accent 1.5	Hyundai	159812 km	2010 Jun	Yes	Update
101	SLX12Z	Elantra 1.8	Hyundai	200300 km	2012 Jun	No	Update
102	SMA5522B	Stinger 2.0	Kia	99890 km	2016 May	No	Update
103	SHQ543C	Cerato 1.6	Kia	178229 km	2015 Aug	No	Update

The code for the action **UsedCar/Index** is given as follows.

```
public class UsedCarController : Controller
{
    public IActionResult Index()
    {
        List<UsedCar> list = DBUtl1.GetList<UsedCar>("SELECT * FROM UsedCar");
        return View("Index", list);
    }
    .....
}
```

Complete the code for the view **Index.cshtml** to generate the status column.

```
@model List<UsedCar>
<h2>Used Cars</h2>
@if (TempData["Message"] != null)
{
    <div class="alert alert-@TempData["MsgType"]">@TempData["Message"]</div>
}
<table class="table table-condensed table-hover">
    <tr>
        <th scope="col">Sno</th>
        <th scope="col">VRN</th>
        <th scope="col">Model</th>
        <th scope="col">Make</th>
        <th scope="col">Mileage</th>
        <th scope="col">Manufacture</th>
        <th scope="col">Sold</th><th scope="col">Action</th>
    </tr>
    @foreach (UsedCar ucar in Model)
    {
        <tr>
            <td>@ucar.Sno</td>
            <td>@ucar.VRegno</td>
            <td>@ucar.CModel</td>
            <td>@ucar.Make</td>

            @*-- Write your code here --*@

            <td>
                <a asp-controller="UsedCar" asp-action="Update"
                  asp-route-id="@ucar.Sno">Update</a>
            </td>
        </tr>
    }
</table>
```

Question 15 (9 marks)

The following diagram illustrates some of the possible error messages that can be output for the Update Used Car page.

Update Used Car Record

Vehicle Reg# :	<input type="text" value="EX1234X"/>	Invalid Vehicle Reg #
Model :	<input type="text"/>	Please enter Model
Make :	<input type="text"/>	Please enter Make
Mileage :	<input type="text" value="99"/>	Mileage 100-500000
Manufacture :	<input type="text"/>	Please enter Manufacture Date

☐ Sold

Update

Here is a complete list of all error messages that will be output.

Property	Error Message	Comment
VRegNo	Please enter Vehicle Reg #	Mandatory
VRegNo	Invalid Vehicle Reg #	Starts with "S", then 2 alphabets, then 1-4 digits and ends with 1 alphabet. E.g. SFT1324X, SMA12K
CModel	Please enter Model	Mandatory
CModel	Model max 30 chars	1 – 30 characters
Make	Please enter Make	Mandatory
Make	Enter Hyundai, Kia or Daewoo	Permitted values are Hyundai, Kia, Daewoo
Mileage	Please enter Mileage	Mandatory
Mileage	Mileage 100 - 500000	100 – 500000
Manufact	Please enter Manufacture Date	Mandatory

```

public class UsedCar
{
    public int Sno { get; set; }
    public string VRegno { get; set; }
    public string CModel { get; set; }
    public string Make { get; set; }
    public int Mileage { get; set; }
    public DateTime Manufact { get; set; }
    public bool Sold { get; set; }
}

```

Provide validation attributes for the basic definition of **UsedCar.cs** so that all possible errors will be output. Your submitted answer shall be the complete code for the class **UsedCar**.

Question 16 (6 marks)

The following code for the **Views\UsedCar\Update.cshtml** is incomplete. Fill in the blanks to complete the code.

```
@model UsedCar

@section MoreScripts {
    <script src="~/lib/moment/moment.min.js"></script>
    <link href="~/lib/dtpicker/css/tempusdominus-bootstrap-4.min.css"
        rel="stylesheet" />
    <script src="~/lib/dtpicker/js/tempusdominus-bootstrap-4.min.js"></script>

    <script language="javascript">
        $(document).ready(function () {
            $(' BLANK A ')
                .datetimepicker({
                    format: 'YYYY-MM-DD',
                });
        });
    </script>
}

<form asp-controller="UsedCar" asp-action="Update" BLANK B >

    <div class="form-group row">
        <div class="offset-sm-2 col-sm-5"><h2>Update Used Car Record</h2></div>
    </div>

    <input type="text" asp-for="Sno" hidden />

    <div class="form-group row">
        <label class="control-label col-sm-2" asp-for="VRegno">Vehicle Reg# :</label>
        <div class="col-sm-5">
            <input type="text" BLANK C ="VRegno" class="form-control" />
        </div>
        <div class="col-sm-4">
            <span BLANK D ="VRegno" class="text-danger"></span>
        </div>
    </div>

    <div class="form-group row">
        <label class="control-label col-sm-2" asp-for="CModel">Model :</label>
        <div class="col-sm-5">
            <input type="text" BLANK C ="CModel" class="form-control" />
        </div>
        <div class="col-sm-4">
            <span BLANK D ="CModel" class="text-danger"></span>
        </div>
    </div>

    <div class="form-group row">
        <label class="control-label col-sm-2" asp-for="Make">Make :</label>
        <div class="col-sm-5">
            <input type="text" BLANK C ="Make" class="form-control" />
        </div>
        <div class="col-sm-4">
            <span BLANK D ="Make" class="text-danger"></span>
        </div>
    </div>
```

```

<div class="form-group row">
  <label class="control-label col-sm-2" asp-for="Mileage">Mileage :</label>
  <div class="col-sm-5">
    <input type="text" BLANK C ="Mileage" class="form-control" />
  </div>
  <div class="col-sm-4">
    <span BLANK D ="Mileage" class="text-danger"></span>
  </div>
</div>

<div class="form-group row">
  <label class="control-label col-sm-2" asp-for="Manufact">Manufact :</label>
  <div class="col-sm-5">
    <div class="input-group date" id="JSManuDT" data-target-input="nearest">
      <input type="text" class="form-control"
        BLANK C ="Manufact" asp-format="{0:yyyy-MM-dd}" />
      <div class="input-group-append"
        data-target="#JSManuDT" data-toggle="datetimepicker">
        <div class="input-group-text"><i class="fa fa-calendar"></i></div>
      </div>
    </div>
  </div>
  <div class="col-sm-4">
    <span BLANK D ="Manufact" class="text-danger"></span>
  </div>
</div>

<div class="form-group row">
  <div class="offset-sm-2 col-sm-3">
    <div class="form-check">
      <input type="checkbox" class="form-check-input" asp-for="Sold" />
      <label class="form-check-label" for="Sold">Sold</label>
    </div>
  </div>
</div>

<div class="form-group row">
  <div class="offset-sm-2 col-sm-5">
    <input type="submit" BLANK E class="btn btn-primary" /><br />
  </div>
</div>

@if ( BLANK F != null)
{
  <div class="form-group row">
    <div class="offset-sm-2 col-sm-5">
      <div class="alert alert-@ViewData["MsgType"]">
        <text>@ BLANK F </text>
      </div>
    </div>
  </div>
}
</form>

```

Blank	Answers
A	
B	
C	
D	
E	
F	

Question 17 (5 marks)

The following incomplete code provides the framework to complete the **Update UsedCar** page of the Web application.

```
public class UsedCarController : Controller
{
    .....

    [HttpGet]
    public IActionResult Update(int id)
    {
        // Question 17
        // Get the record from the database using the id
        // If the record is found, pass the model to the View
        // Otherwise, redirect to the UsedCar/Index page
        // with the message "Used Car record does not exist"

    }

    [HttpPost]
    public IActionResult Update(UsedCar ucar)
    {
        // Question 18
        // Check the ModelState
        // If not valid, display the message "Invalid Input" in the same View
        // Otherwise,
        // Write SQL Update statement
        // Execute the statement with model's properties
        // Check for success
        // If success, redirect to the UsedCar/Index page
        // with "Used Car Record Updated"
        // Otherwise, redirect to the UsedCar/Index page with db error message

    }
}
```

Complete the code for the **[HttpGet] Update** action in the UsedCarController.cs

Question 18 (7 marks)

Complete the code for the **[HttpPost] Update** action in the UsedCarController.cs.

– END OF PAPER –