Microsoft

Exam 70-494

Recertification for MCSD: Web Applications

Version: 8.0

[Total Questions: 249]

Topic break down

Topic	No. of Questions
Topic 1: Olympic Marathon	14
Topic 2: Web Application	16
Topic 3: Video Transcoding Service	13
Topic 4: Mixed Questions Set 1	93
Topic 5: Flight Information	16
Topic 6: ASP.NET MVC	16
Topic 7: Online Bookstore	15
Topic 8: Mix Questions B	66

Topic 1, Olympic Marathon

Background

You are developing an ASP.NET MVC application in Visual Studio 2012 that will be used by Olympic marathon runners to log data about training runs.

Business Requirements

The application stores date, distance, and duration information about a user's training runs. The user can view, insert, edit, and delete records.

The application must be optimized for accessibility.

All times must be displayed in the user's local time.

Technical Requirements

Data Access:

Database access is handled by a public class named RunnerLog.DataAccess.RunnerLogDb.
All data retrieval must be done by HTTP GET and all data updates must be done by HTTP POST.

Layout:

All pages in the application use a master layout file named \Views\Shared_Layout.cshtml.

Models:

The application uses the \Models\LogModel.cs model.

Views:

All views in the application use the Razor view engine.

Four views located in \Views\RunLog are named:

- _CalculatePace.cshtml
- EditLog.cshtml
- GetLog.cshtml
- InsertLog.cshtml

The application also contains a \Views\Home\Index.cshtml view.

Controllers:

The application contains a \Controllers\RunLogController.cs controller.

Images:

A stopwatch.png image is located in the \Images folder.

Videos:

A map of a runner's path is available when a user views a run log. The map is implemented

as an Adobe Flash application and video. The browser should display the video natively if possible, using H264, Ogg, or WebM formats, in that order. If the video cannot be displayed, then the Flash application should be used.

Security:

You have the following security requirements:

- The application is configured to use forms authentication.
- Users must be logged on to insert runner data.
- Users must be members of the Admin role to edit or delete runner data.
- There are no security requirements for viewing runner data.
- You need to protect the application against cross-site request forgery.
- Passwords are hashed by using the SHA1 algorithm.

RunnerLog.Providers.RunLogRoleProvider.cs contains a custom role provider.

Relevant portions of the application files follow. (Line numbers are included for reference only.)

Application Structure

Controllers\RunLogController.cs

```
RC01  public class RunLogController : Controller
RC02  {
RC03    public ActionResult GetLog()
RC04    {
RC05      List<LogModel> log = RunnerLogDb.GetLogsFromDatabase();
RC06      return View(log);
RC07    }
RC08
RC09    public ActionResult InsertLog()
RC10    {
RC11      LogModel log = new LogModel();
RC12      log.RunDate = DateTime.Now;
RC13      return View(log);
RC14    }
RC15
RC16    [HttpPost]
RC17    public ActionResult InsertLog(LogModel log)
RC18    {
RC19      RunnerLogDb.InsertLog(log);
RC20      return RedirectToAction("GetLog");
RC21    }
RC22
RC23    public ActionResult DeleteLog(int id)
RC24    {
RC25      RunnerLogDb.DeleteLog(id);
RC26      return RedirectToAction("GetLog");
RC27    }
RC28
RC29    public ActionResult EditLog(int id)
RC30    {
RC31      LogModel log = RunnerLogDb.GetRunnerLog(id);
RC32      return View(log);
RC33    }
RC34  }
```

Models\LogModel.cs

```
LM01  public class LogModel
LM02  {
LM03    [Required]
LM04    public int Id { get; set; }
LM05
LM06    [Required]
LM07    public DateTime RunDate { get; set; }
LM08
LM09    [Required]
LM10    [Range (0.01, 1000.00)]
LM11    public double Distance { get; set; }
LM12
LM13    [Required]
LM14    public TimeSpan Time { get; set; }
LM15
LM16    public string ShortDate
LM17    {
LM18      get
LM19      {
LM20        return RunDate.ToLocalTime().ToShortDateString();
LM21      }
LM22    }
LM23  }
```

Views\RunLog_CalculatePace.cshtml

```
CP01  @model RunnerLog.Models.LogModel
CP02  @(Convert.ToInt32(Model.Time.TotalMinutes / Model.Distance)) Min
CP03  @(Convert.ToInt32(Model.Time.TotalSeconds % 60 / Model.Distance)) Seconds
```

Views\RunLog\EditLog.cshtml

```
EL01  @model RunnerLog.Models.LogModel
EL02  <h2>Edit Log Item</h2>
EL03  <script src="@Url.Content("~/Scripts/jquery.validate.min.js")"></script>
EL04  <script src="@Url.Content("~/Scripts/jquery.validate.unobtrusive.min.js")"></
script>
EL05  @using (Html.BeginForm()) {
EL06    @Html.AntiForgeryToken()
EL07    @Html.ValidationSummary(true)
EL08    <fieldset>
EL09      <legend>LogModel</legend>
EL10      <h3>
EL11        Log Id:  @Model.Id
EL12      </h3>
EL13      <div>
EL14        @Html.LabelFor(model => model.Distance)
EL15      </div>
EL16      <div>
EL17        @Html.EditorFor(model => model.Distance)
EL18        @Html.ValidationMessageFor(model => model.Distance)
EL19      </div>
EL20      <div>
EL21        @Html.LabelFor(model => model.Time)
EL22      </div>
EL23      <div>
EL24        @Html.EditorFor(model => model.Time)
EL25        @Html.ValidationMessageFor(model => model.Time)
EL26      </div>
EL27      <p>
EL28        <input type="submit" value="Save" />
EL29      </p>
EL30    </fieldset>
EL31  }
```

Views\RunLog\GetLog.cshtml

```
GL01  @model List<RunnerLog.Models.LogModel>
GL02  <h2>View Runs </h2>
GL03  <table>
GL04    <tr>
GL05      <th>Id </th>
GL06      <th>Date </th>
GL07      <th>Distance </th>
GL08      <th>Duration </th>
GL09      <th>Avg Mile Pace </th>
GL10    </tr>
GL11    @foreach (RunnerLog.Models.LogModel log in Model)
GL12    {
GL13      <tr>
GL14        <td>
GL15          @Html.DisplayFor(model => log.Id)
GL16        </td>
GL17        <td>
GL18
GL19        </td>
GL20        <td>
GL21          @Html.DisplayFor(model => log.Distance)
GL22        </td>
GL23        <td>
GL24          @Html.DisplayFor(model => log.Time)
GL25        </td>
GL26        <td>
GL27
GL28        </td>
GL29        <td>
GL30          @Html.ActionLink("Edit", "EditLog", new { id = log.Id })
GL31        </td>
GL32        <td>
GL33          @Html.ActionLink("Delete", "DeleteLog", new { id = log.Id })
GL34        </td>
GL35      </tr>
GL36    }
GL37  </table>
```

Views\RunLog\InsertLog.cshtml

```
IL01  @model RunnerLog.Models.LogModel
IL02  <script src="@Url.Content("~/Scripts/jquery.validate.min.js")"></script>
IL03  <script src="@Url.Content("~/Scripts/jquery.validate.unobtrusive.min.js")"></
script>
IL04  @using (Html.BeginForm())
IL05  {
IL06    @Html.ValidationSummary(true)
IL07    <fieldset>
IL08      <legend>LogModel</legend>
IL09
IL10      <div>
IL11        @Html.LabelFor(model => model.RunDate)
IL12      </div>
IL13      <div>
IL14        @Html.EditorFor(model => model.RunDate)
IL15        @Html.ValidationMessageFor(model => model.RunDate)
IL16      </div>
IL17      <div>
IL18        @Html.LabelFor(model => model.Distance)
IL19      </div>
IL20      <div>
IL21        @Html.EditorFor(model => model.Distance)
IL22        @Html.ValidationMessageFor(model => model.Distance)
IL23      </div>
IL24      <div>
IL25        @Html.LabelFor(model => model.Time) HH:MM:SS
IL26      </div>
IL27      <div>
IL28        @Html.EditorFor(model => model.Time)
IL29        @Html.ValidationMessageFor(model => model.Time)
IL30      </div>
IL31      <p>
IL32        <input type="submit" value="Create" />
IL33      </p>
IL34    </fieldset>
IL35  }
```

Views\Shared_Layout.cshtml

```
LO01  <!DOCTYPE html>
LO02   <html lang="en">
LO03   <head>
LO04     ...
LO05   </head>
LO06   <body>
LO07   ...
LO08    <footer>
LO09
LO10      <script type="text/javascript">
LO11        var c = document.getElementById('myCanvas');
LO12        var ctx = c.getContext('2d');
LO13        ctx.font = '30pt Calibri';
LO14        ctx.strokeStyle = 'gray';
LO15        ctx.lineWidth = 3;
LO16        ctx.strokeText('London 2012', 80, 30);
LO17      </script>
LO18    </footer>
LO19   </body>
LO20   </html>
```

Question No : 1 - (Topic 1)

You need to extend the edit functionality of RunLogController.

Which code segment should you use?

A.
```
[HttpGet]
[ActionName("EditLog")]
[ValidateAntiForgeryToken]
public ActionResult EditLog(LogModel log)
{
  ...
}
```

B.
```
[HttpPost]
[ActionName("EditLog")]
public ActionResult EditLogValidated(LogModel log)
{
  ...
}
```

C.
```
[HttpPost]
[ActionName("EditLog")]
[ValidateAntiForgeryToken]
public ActionResult EditLogValidated(LogModel log)
{
  ...
}
```

D.
```
[HttpPost]
[ActionName("EditLog")]
[RequireHttps]
public ActionResult EditLogValidated(LogModel log)
{
  ...
}
```

A. Option A
B. Option B
C. Option C
D. Option D

Answer: C

Question No : 2 - (Topic 1)

The RunLog/Views/InsertLog.cshtml view must display the /Images/stopwatch.png image and the “Insert Run Data” header text below the image. The view should resemble the exhibit. (Click the Exhibit button.)

The application must display the image above the field set.

You need to add the HTML code to /Runlog/Views/InsertLog.cshtml to display the image and header text.

Which code segment should you use?

A.
```
<h2>
  Insert Run Data
</h2>
<div>
  <img src="../Images/StopWatch.png"/>
</div>
```

B.
```
<div style="background: url('../Images/StopWatch.png');">
  <h2>Insert Run Data</h2>
</div>
```

C.
```
<div style="width: 130px; height: 100px;">
  <a href="../Images/StopWatch.png"></a>
</div>
<h2>
  Insert Run Data
</h2>
```

D.
```
<div style="width: 130px; height: 100px; background: url
('../Images/StopWatch.png');">
</div>
<h2>
  Insert Run Data
</h2>
```

A. Option A
B. Option B
C. Option C
D. Option D

Answer: D

Explanation: Example:

<div style="background-image: url(../images/test-background.gif); height: 200px; width: