**Building the Model in MVC – Displayig Details of a Recording**

|  |  |  |
| --- | --- | --- |
| **Introduction**  This tutorial will build on the practices and the concepts you were introduced to in the lecture. | | **Required Software**  For this tutorial, you will need access to the following software applications:  MS Explorer  Visual Studio  SQL SERVER |
| **Reference**  Walther et al, ‘ASP.NET MVC Framework Unleashed’, SAMS | **Breakdown of Tutorial**  This tutorial consists of the following task:  You will extend *'Forest'* to include new functionalities. You will be given a series of practical exercises, along with selected questions to consolidate your learning. | | |

**Review questions**

|  |  |
| --- | --- |
| **Refreshers:** Recall from memory as much as possible to answer the following refresher questions. Then open the *'Forest'* MVC projects you built during previous tutorials to help you develop and extend your answers. | |
|  | MVC is a design pattern. It is also sometimes referred to as an architecture. What is a design pattern? What are the benefits fo pattern-based development? |
|  | |
|  | MVC stands for Model-View-Contrller. Briefly explain the three components in terms of what they do. You are encouraged to draw a diagram that depicts the interactions between the three components. |
|  | |
|  | In particular about the model: What are examples of models? What sort of thing maybe considered a model? |
|  | |
|  | In particular, what are the benefits of MVC in developing web applications? |
|  | |
|  | Explain 'Forest' application in terms of its functionalities. What does it so far do? |
|  | |

**Tutorial objectives**

In the previous tutorial, we built the *Forest* MVC application. The object of this exercise is to extend the application to increase its functionalities. We will learn how MVC's modular development approach helps us build extensible systems.

**Extending the Forest MVC application**

|  |
| --- |
| Open the *Forest* solution. Build the solution and run the application to make sure all is well. |

|  |  |  |  |
| --- | --- | --- | --- |
| **About the Application**  Here we introduce an application flow to facilitate the exercise. | | | |
| C:\Users\mo\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\H5P0LERU\MP900437370[1].jpg | | The application opens in *Categories.cshtm* View which is *ActionResult* of *Categories()* Action of *MusicController*. We get a list of *Genres*. | |
| *A genre is click.*  *Recordings(genre)* Action returns a list of *Recordings* in that *Genre*. The list could be limited to few of the fields, for example *Artist* and *Title*. | |
| A *Details* link is clicked.  *Recording(id)* Action returns a full details of a *Recording*. | |
|  | | Sketch the application's work flow. | |
|  | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **About the Architecture** | | | | |
| Capture  **Always working from right to left ...** | | | | |
| C:\Users\mo\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\H5P0LERU\MP900437370[1].jpg | Data Tire | | Our application tends to work with a variety of data sources including various servers such as SQL Server, Oracle Database, MySQL, XML databases and local database files. In our case, we will make use of a SQL Server Local Database file. This database is integrated into the Integration tier. |
| Model | Integration Tier | The integration tier facilitates access to data sources. Looking at code in this layer we expect to see SQL, LINQ, XPath, etc for querying the data provider and data manipulator objects. The integration tier also facilitates access to 3rd party data services such as Web Services. |
| Service Tier | The service tier is concerned with business logic and data manipulation. We implement all the business rules that the application performs in this layer. Looking at code in this layer, we would expect to see references to integration tier objects as well as implementation of mathematical calculations, conditional decisions, etc. |
| MVC | | MVC consists of the controllers, views, JScripts and the various components that deal with the user interaction. MVC is the presentation layer of the architecture. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Building the Application**  Objective;  \_ Display details of a *music recording*. Having displayed music recordings in a *genre*, we would like to hyperlink *Details* to display comprehensive information about one *music recording*. | | | |
| **C:\Users\mo\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\H5P0LERU\MP900437370[1].jpg** | MVC facilities extensibility of the application. That is, you will add additional functionality to the application with ease and without affecting the existing code. As part of this, we have an established regime for coding.  **We always code from right to left. We always code starting with Data and finishing with the controller and the actions.** | | |
| \_ Ensure that you have completed the previous tutorials and that your solution successfully displays music categories and recordings in a genre. | |  |  |

|  |
| --- |
| **C:\Users\mo\Desktop\Capture.PNG Working on *Forest.Data* project** |

|  |  |
| --- | --- |
| \_ Extend the *IMusicDAO* Interface; |  |
| \_ Extend the *MusicDAO* class to implement the new method definition in the interface; |  |
| Save and build *Forest.Data* project. | |

|  |  |
| --- | --- |
| **C:\Users\mo\Desktop\Capture1.PNG Working on *Forest.Services* project** | |
| \_ Extend the *IMusicService* Interface; |  |
| \_ Extend the *MusicService* class to implement the new method definition in the interface; |  |
| Save and build *Forest.Services* project. | |

|  |  |  |
| --- | --- | --- |
| **Capture Working on *Forest* project**  Objective;  Object of this exercise is to add a new action to the *MusicController* called *Recording*. This action would display music recordings in a particular *genre*. To achieve this we will do the following;  \_ Create the controller action.  \_ Create the *View* for the action.  \_ Create the navigation through linking the *Details* hyperlink to the *Recording* action. | | |
| Working on the *MusicController*; | | |
| \_ Create a new controller action named *Recording*. | |  |
| \_ Create a *View* for the action. | |  |
| \_ Save and close the controller. | | |
|  | Working on GetMusicRecordings.cshtml view;  \_ Alter the *Details* *Html.ActionLink* helper to point to *Recording* action, passing *id* as parameter. | |
| \_ Save all, build the project and press *F5* to run the application. | | |

**Student-centred exercise**

You are to work on your own to extend the application for Video;

|  |  |
| --- | --- |
| * Create the Video Data Accessor (VideoDAO) | C:\Users\cmsmr2\Desktop\Capture.JPG |
| * Create the Video Search Service (VideoService) | C:\Users\cmsmr2\Desktop\2.JPG |