**Where we are:**

|  |  |
| --- | --- |
| Intro to MVC | Annotation and validation |
| Controllers and views | Identity (Authentication and authorization) |
| Models and Entity Framework | Routing |
| Forms and HTML helpers | Dependency Injection |
| Customizing Scaffolding | Unit Testing |

**Announcements**  
A preliminary version of your term project is due along with lab 5

**Review Last Week’s Code**

**Validation**

* Expected in the browser (Ux), required on the server (security)
* Good Ux provides a way for users to easily fix entry errors
* In ASP.NET MVC, validation is done on the models
* Implemented using C# attributes – MVC calls them “data annotations”

Validation Annotations

Example: [Required]

Discussion

All non-nullable model fields are required by default. Make fields nullable by adding ? – this will also make them nullable in the database.

Hidden fields: look at the Edit scaffolding. They are used to hold the entity ID so that it is sent back with the post data. Explain why models that are sent to the view aren’t kept in memory and can’t automatically be posted with all the original data in them.

Updating edited fields. db.State(entity)

Partial views:

* Created like any other view. Put it in Views/Shared/ (or the same folder as the view that consumes it).
* Only contain a fragment of HTML, not a full HTML document
* Include a partial view with @Html.Partial(“PartialViewName”) // no extension on the name
* @Html.ActionLink helpers that don’t specify a controller will be directed to the controller for the view that consumes the partial view.
* Strongly typed partial views: The model can be passed to @Html.Partial as the second parameter. It could be the same model used in the parent view or it could be a new object created in the parent’s razor markup.