**B1. What is model?**

**A:** The model classes represents domain-specific data and business logic in the MVC application. It represents the shape of the data as public properties and business logic as methods.

**B2. What is routing in MVC?**

**A:** Route defines the URL pattern and handler information. All the configured routes of an application stored in Route Table and will be used by the Routing engine to determine appropriate handler class or file for an incoming request.

**B3. What is the difference between Temp data, View, and View Bag?**

**View bag:** View Bag is a dynamic object to pass the data from Controller to View. And, this will pass the data as a property of object view Bag. And we have no need to typecast to read the data or for null checking. The scope of View Bag is permitted to the current request and the value of view Bag will become null while redirecting.

**View data:** View data is a dictionary object to pass the data from Controller to View where data is passed in the form of key-value pair. And typecasting is required to read the data in View if the data is complex and we need to ensure null check to avoid null exceptions. The scope of View data is similar to View bag and it is restricted to the current request and the value of View data will become null while redirecting.

**Template:** Template is a dictionary object to pass the data from one action to other action in the same Controller or different Controllers. Usually, Template object will be stored in a session object. Template is also required to typecast and for null checking before reading data from it. Template scope is limited to the next request and if we want Template to be available even further, we should use Keep and peek.

**B4. What is difference between MVC and Web Forms?**

**Mvc:** 1. MVC focuses on separation of concern, i.e., there is no fixed code behind page for every view. A view can be called from multiple action.

2. First request comes to controller and action, then view gets called.

3. MVC provides HTML Helpers to create form controls. This is optional. You can use simple HTML controls also.

4. There is no View State for state management in View.

5. We create partial views for reusable views.

6. It is very easy to use jquery and JavaScript Using CSS is also easy.

**Web forms:**

1. Web form based on functions and page behind code, i.e., there is code behind page for each view. You have to write code in that class related to this view only.

2. First request comes to Page (View) then it will go to code behind page.

3. For everything in webforms, you have a server control.

4. Viewstate is used to maintain state of form in view. This also makes page heavy.

5. We create user controls for reusable view or control.

6. It is a little difficult to use jquery and JavaScripts in web forms. It provides themes and it is difficult to manage design of server controls.

**B5. What is session? What is the default time for session?**

A: Session variables are stored in a [**SessionStateItemCollection**](https://msdn.microsoft.com/en-us/library/93f46x05(v=vs.100))object that is exposed through the [HttpContext.Session](https://msdn.microsoft.com/en-us/library/ye8zyye9(v=vs.100)) property. In an ASP.NET page, the current session variables are exposed through the Session property of the Page obj.

2. Session default time out is 20min.

**B6. What is Partial View in MVC? With example?**

A: The partial view is instantiated with its own copy of a ViewDataDictionary object which is available with the parent view so that partial view can access the data of the parent view. If we made the change in this data (ViewDataDictionary object), the parent view's data is not affected. Generally the Partial rendering method of the view is used when related data that we want to render in a partial view is part of our model.

**B7. What is the difference between View and Partial View?**

|  |  |
| --- | --- |
| **View**     * View contains the layout page. * 2.      Before any view is rendered, viewstart page is rendered. * 3.      View might have markup tags like body, html, head, title, Meta etc. * 4.      View is not lightweight as compare to Partial View | **Partial View**   * Partial View does not contain the layout page * 2.      Partial view does not verify for a viewstart.cshtml page. * 3.      Partial view is designed specially to render within the view and just because of that it does not consist any mark up * 4.      We can pass a regular view to the RenderPartial method |

**B8. Explain the concept of MVC Scaffolding?**

Scaffolding is a technique used by many MVC frameworks like ASP.NET MVC, Ruby on Rails, Cake PHP and Node.JS etc., to generate code for basic CRUD (create, read, update, and delete) operations against your database effectively. Further you can edit or customize this auto generated code according to your need.

**B9.How to change time of session?**

ASP.NET MVC Session state is used to **temporarily** store and retrieve the values for a user when the user navigates to another view in an ASP.NET MVC application.

Usually, the session timeout configuration setting will be stored in the web.config file in the MVC application. If you want to increase the session timeout then open your application web.config file which is placed under your application root folder.

**B10. What is query string? What are disadvantages of query string?**

A **Query String** is information sent to the server appended to the end of a page URL which means that Query String is way to transfer information from one page to another through the URL. Query String is attached to the URL with "?"

**Disadvantages:**

1.    There is a limit to URL length of 255 characters.

2.    Query String data is directly visible to user thus leading to security problems.

**B11.What is cookie? What are limitations for cookie?**

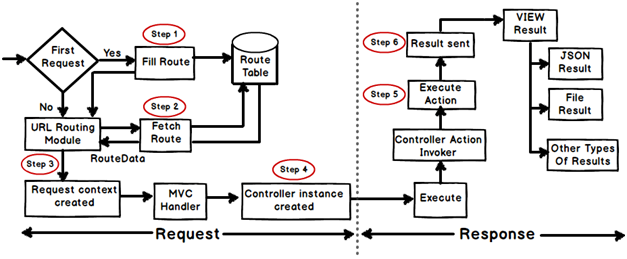
 Cookies is a small piece of information stored on the client machine. This file is located on client machines "C:\Document and Settings\Currently\_Login user\Cookie" path.  Its is used to store user preference information like Username, Password,City and PhoneNo etc on client machines. We need to import namespace called Systen.Web.HttpCookie before we use cookie

**Limitations for cookie:**

* Most browsers support cookies of up to 4096 bytes. That is plenty of space for storing a few values on the user's computers, but you would not want to try to store a dataset or some other potentially large piece of data in a cookie. More practically, you probably do not want to store a big collection of user information in a cookie. Instead, you would want to store a user number or other identifier. Then, when the user visits your site again, you would use the user ID to look up user details in a database.
* Browsers also impose limitations on how many cookies your site can store on the user's computer. Most browsers allow only 20 cookies per site; if you try to store more, the oldest cookies are discarded. Some browsers also put an absolute limit, usually 300, on the number of cookies they will accept from all sites combined.
* A cookie limitation that you are likelier to run into is that users can set their browser to refuse cookies. You cannot do much to get around this problem except to avoid cookies altogether and use a different mechanism to store user-specific information. A common method for storing user information is Session state, but Session state depends on cookies, as I explain later in Cookies and Session.

**B13. Explain MVC application life cycle.**

**A1. Explain MVC application life cycle**



**Step 1 Fill route: -**MVC requests are mapped to route tables which in turn specify which controller and action to be invoked. So if the request is the first request the first thing is to fill the route table with routes collection. This filling of route table happens in the global.asax file.

**Step 2 Fetch route:-**Depending on the URL sent "UrlRoutingModule" searches the route table to create "RouteData" object which has the details of which controller and action to invoke.

**Step 3 Request context created: -**The "RouteData" object is used to create the "RequestContext" object.

**Step 4 Controller instance created: -**This request object is sent to "MvcHandler" instance to create the controller class instance. Once the controller class object is created it calls the "Execute" method of the controller class.

**Creating Response object: -**This phase has two steps executing the action and finally sending the response as a result to the view.

**Step 5 Execute Action: -**The "ControllerActionInvoker" determines which action to executed and executes the action.

**Step 6 Result sent: -**The action method executes and creates the type of result which can be a view result , file result , JSON result etc.

So in all there are six broad steps which get executed in MVC application life cycle.

**B14. List out different return types of a controller action method ?**

**Return View:** This is a most common and very frequently used type. We see that we can pass eight parameters when we return the view. We can specify the view name explicitly or may not.

**Return partial View**: The concept of a partial view is very similar to the master page concept in Web Form applications. The partial view is nothing but platelet that we can return from the controller and that merges with the main view and generates one concrete HTML page.

**Redirect**:This is equivalent to Response.redirect() or Server.Transfer() functions. It takes the URL path to be redirect , though we can use Response.Redirect() or Server.Transfer() in MVC too.

Redirect To Action**:**Sometimes it is necessary to call another action after completion of one action, this is very similar to a function call in traditional function oriented programming or Object Oriented Programming. It may take 6 parameters. The first parameter is very simple, only action name.

**Return JSON**: This is very useful when we don't want an entire HTML page but only want a value. Generally in AJAX-based single-page applications we do not load an entire page again and again but load fresh data from the DB using AJAX. In this scenario we can return only a JSON object and in the success function of jQuery Ajax (let's assume we are using the jqury library to implement AJAX) we can just manipulate data.

Return JavaScript: When we wanted to return a JavaScript string, we may use this function. It takes only one parameter, the string only.

Return File:We are allowed to return a binary file if needed from a controller. It takes 6 parameters maximum.

**B15. What are filters in mvc?**

There are five types of Filters in ASP.NET MVC 5:

1. **Authentication filters :**

Authentication filter runs before any other filter or action method. Authentication confirms that you are a valid or invalid user. Action filters implement the IAuthenticationFilter interface.

1. **Authorization Filters:**

The AuthorizeAttribute and RequireHttpsAttribute are examples of Authorization Filters. Authorization Filters are responsible for checking User Access; these implement the IAuthorizationFilterinterface in the framework. These filters used to implement authentication and authorization for controller actions. For example, the Authorize filter is an example of an Authorization filter.

1. **Action Filters:**

Action Filter is an attribute that you can apply to a controller action or an entire controller. This filter will be called before and after the action starts executing and after the action has executed.Action filters implement the IActionFilter interface that has two methods OnActionExecuting andOnActionExecuted. OnActionExecuting runs before the Action and gives an opportunity to cancel the Action call. These filters contain logic that is executed before and after a controller action executes, you can use an action filter, for instance, to modify the view data that a controller action returns.

1. **Result Filters:**

The OutputCacheAttribute class is an example of Result Filters. These implement the IResultFilter interface which like the IActionFilter has OnResultExecuting and OnResultExecuted. These filters contain logic that is executed before and after a view result is executed. Like if you want to modify a view result right before the view is rendered to the browser.

1. **ExceptionFilters:**

The HandleErrorAttribute class is an example of ExceptionFilters. These implement the IExceptionFilter interface and they execute if there are any unhandled exceptions thrown during the execution pipeline. These filters can be used as an exception filter to handle errors raised by either your controller actions or controller action results.

**B16. What are HTML helpers in MVC?**

HTML Helpers are categorized into three types,

1. Inline HTML helpers
2. Built-in HTML helpers
   1. Standard HTML Helpers
   2. Strongly Typed HTML helpers
   3. Templated HTML helpers
3. Custom HTML helper

**B17. Differences between Razor and ASPX View Engine in MVC?**

|  |  |
| --- | --- |
| **Razor View Engine** | **ASPX View Engine in MVC** |
| The namespace used by the Razor View Engine is System.Web.Razor | The namespace used by the ASPX View Engine is System.Web.Mvc.WebForm  ViewEngine |
| Razor has a syntax that is very compact and helps us to reduce typing. | The web form view engine has syntax that is the same as an ASP.Net forms application. |
| The Razor View Engine uses @ to render server-side content. | The ASPX/web form view engine uses "<%= %>" or "<%: %>" to render server-side content. |
| By default all text from an @ expression is HTML encoded | There is a different syntax ("<%: %>") to make text HTML encoded. |
| There is only three transition characters with the Razor View Engine. | There are only three transition characters with the Razor View Engine. |