1. **What is InProc and OutProc session?**

**In-proc session state:** An inproc server runs in the same process as the calling application. It’s close to a normal function call on a dll.

In-Process stores the session in memory on the web server. This requires the “sticky-server” (or no load-balancing) so that the user is always reconnected to the same web server.

In-Proc session mode indicates that session state is stored locally, means that with In-Proc session state mode is store objects in the AppDomain of the web application. Because of this session state is lost when IIS (Internet Information System) restarts. For small, very stable sites,

InProc is perfectly acceptable and possibly even ideal. InProc also has the benefit of being able to hold any memory object in session. This can also be problematic if you attempt to hold enormous objects.

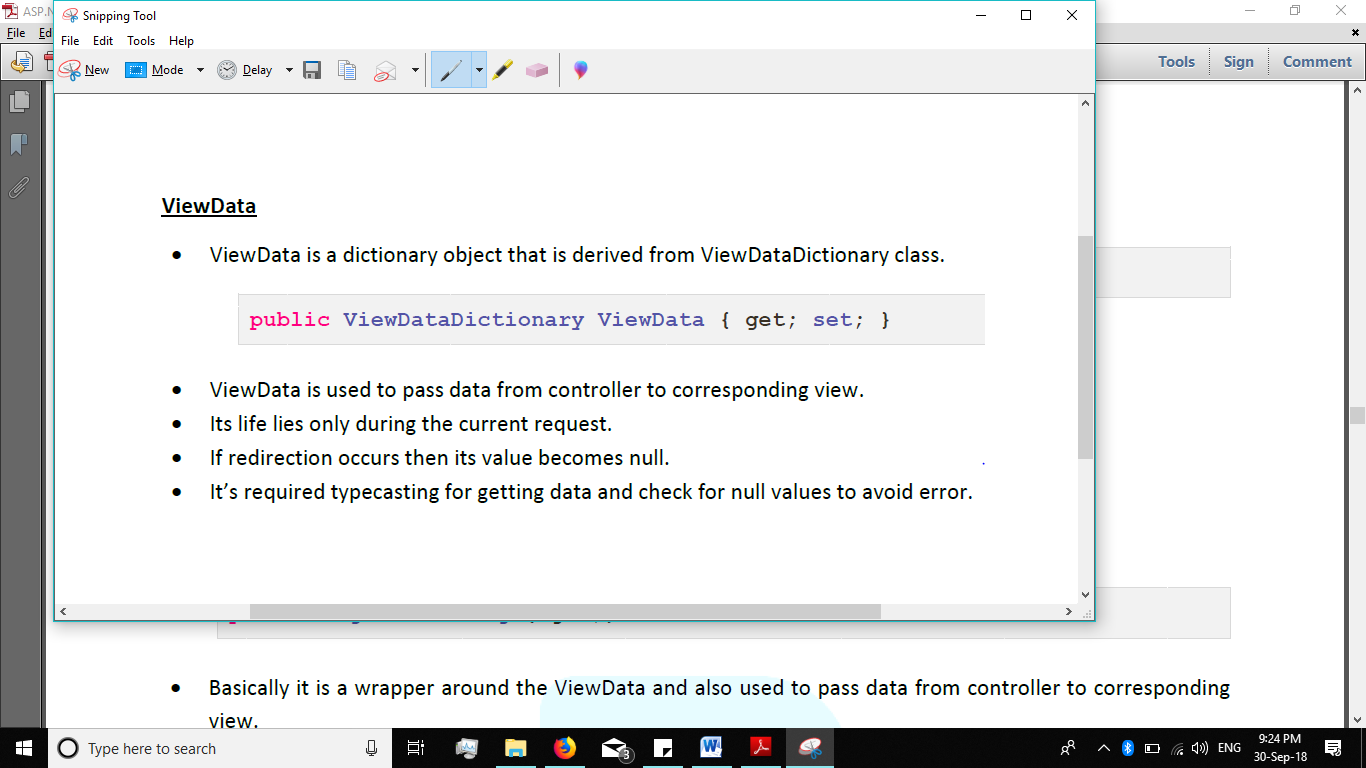
**Out-Proc session state:** State server mode / OutProc, which stores session state in a separate process called the ASP.NET state service. The ensures that session state is preserved if the Web Application is restarted and also makes session state available to multiple web service in a web forms. Calling an Out-Proc server, data needs to be marshaled across the process boundry which is an expensive operation.

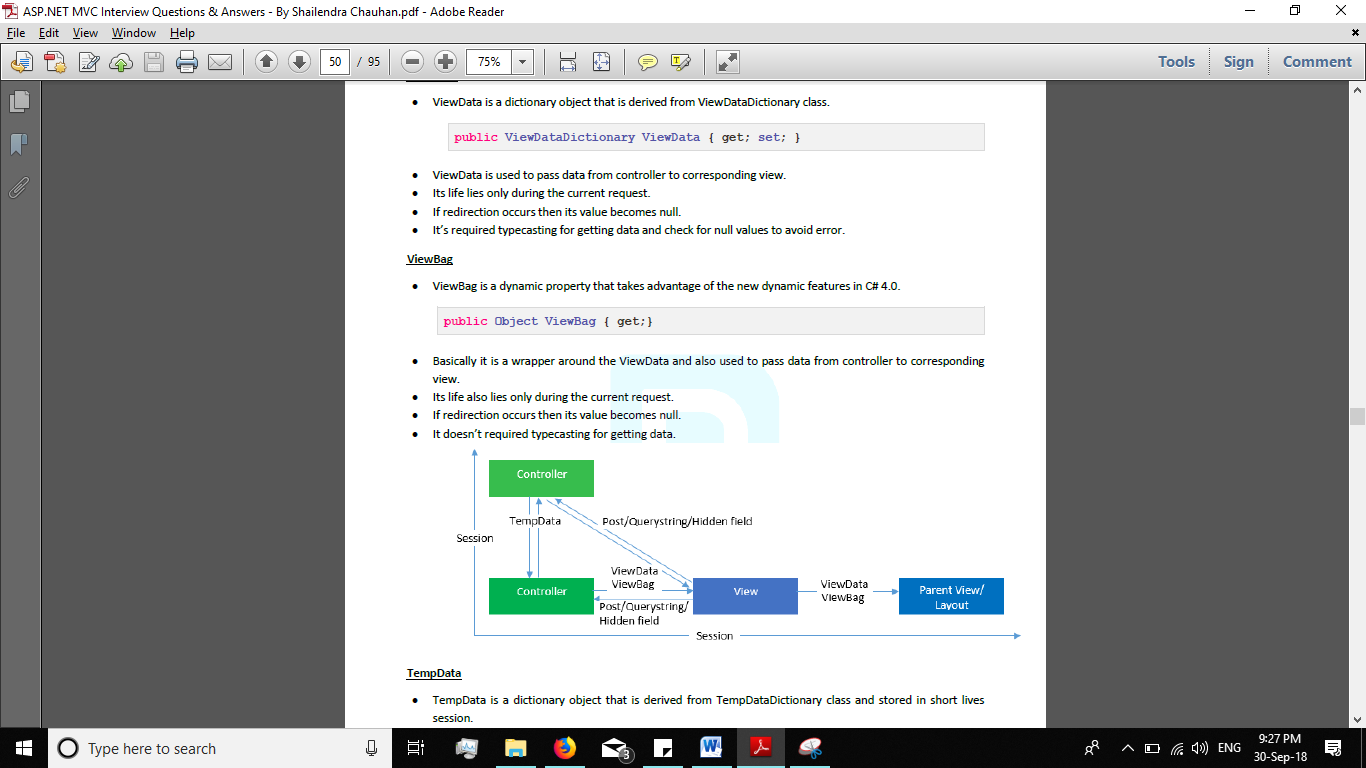
The basic necessity of using the Out-Proc session is when you have deployed your application in web form or Web garden. Out-Proc session should be used when you expect issue like if you are modifying you web.config file or bin folder at runtime or IIS restart on server. As state server mode gives a advantages that it will not lose the data even if your application crash.

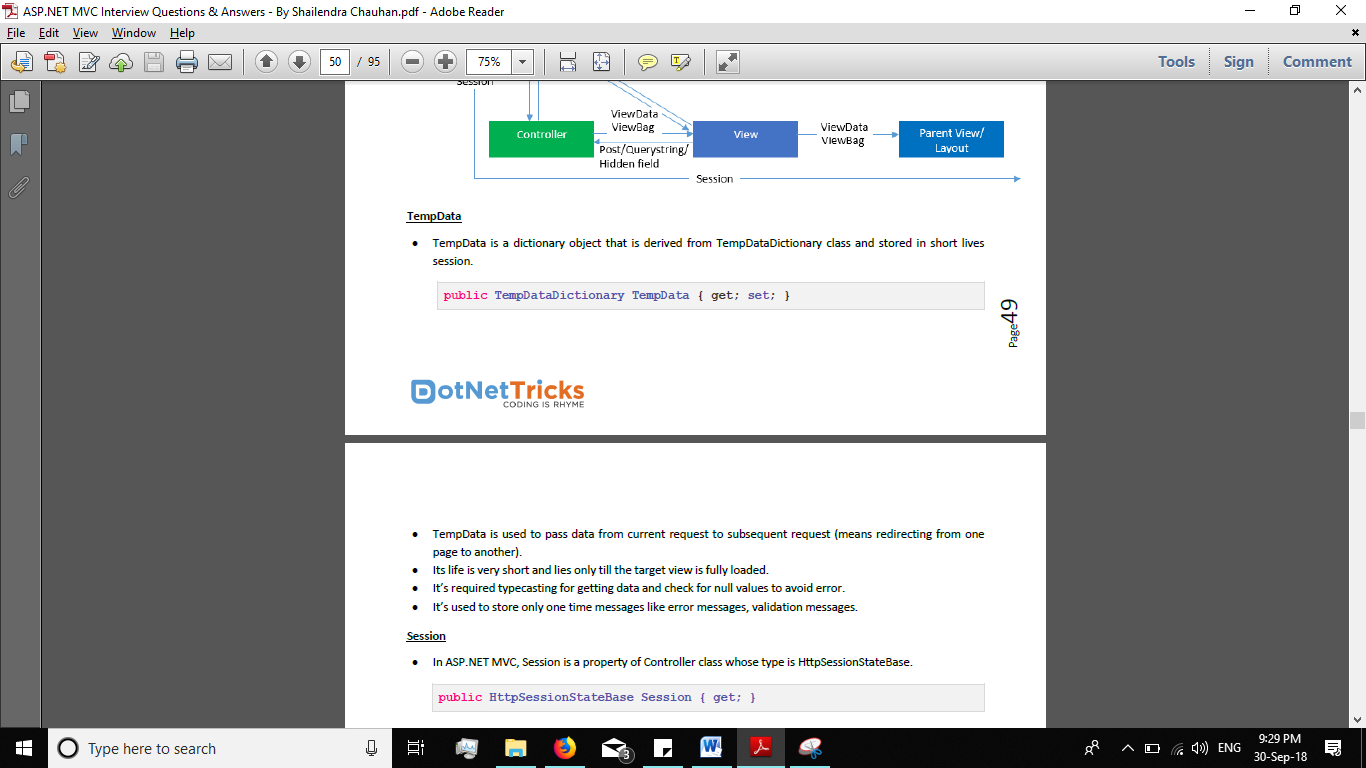
1. **What is ViewData, ViewBag, TempData and Session?**
2. **What is difference among ViewData, ViewBag, TempData and Session?**

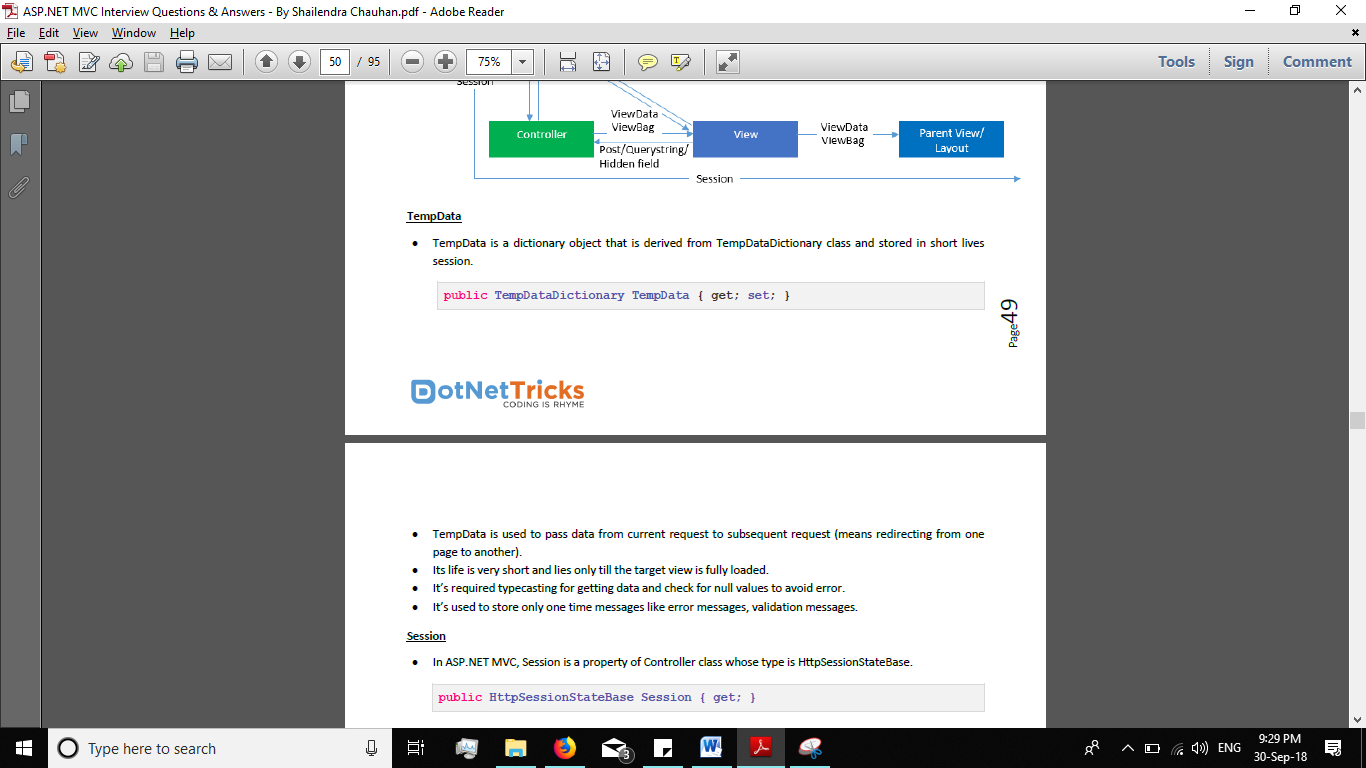
**Answer:**

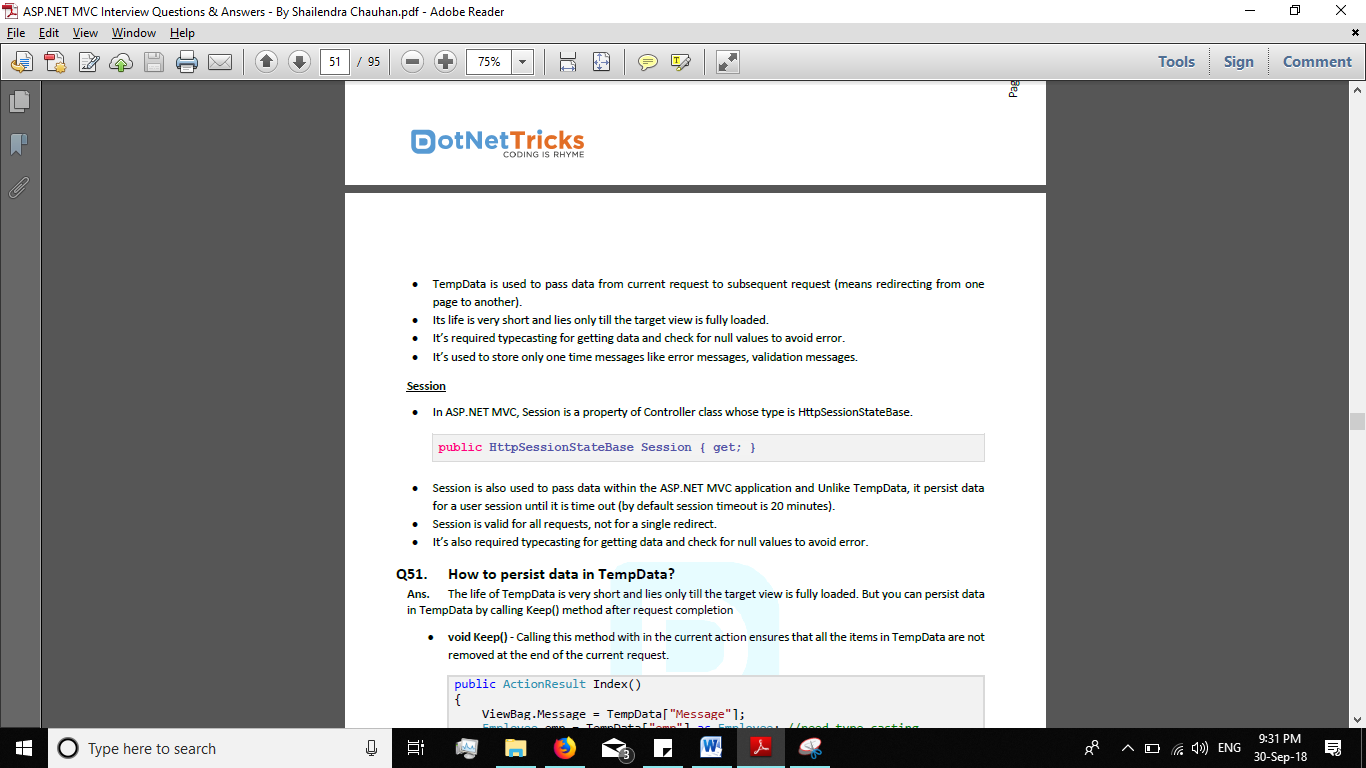
In Asp.Net MVC there are three ways – VigData, ViewBag and TempData to pass data from controller to view and in next request. Link WebForm, you can also use session to persist data during a user session.











**--------------------------------------------------------------------------------------------------------------------------------------------------**

**Que: What is Filter in MVC?**

**Ans:**

* ASP.NET MVC Filter provides a simple way to inject a piece of code or logic either before or after an action is executed.
* For achieving this functionality, ASP.Net MVC provide a feature to add Pre and Post-action behaviors on the controller’s action methods.
* Filters can be applied to an action method or controller in a declarative or programmatic way.

**--------------------------------------------------------------------------------------------------------------------------------------------------**

**Que: When to use Filters?**

**Ans:** Filter are use to perform following common functionalities in your ASP.NET MVC application.

* **Custom Authentication**
* **Custom Authorization (user based or Role based)**
* **Error handling or logging.**
* **User Activity logging.**
* **Data Caching.**
* **Data Compression**.

**--------------------------------------------------------------------------------------------------------------------------------------------------**

**Que: What are different types of Filters in ASP.NET MVC? / Action Filters Types?**

**Ans:** ASP.NET MVC framework provides 5 types filters.

* **Authentication Filters**
* **Authorization Filters**
* **Action Filters**
* **Result Filters**
* **Exception Filters**

**--------------------------------------------------------------------------------------------------------------------------------------------------**

**Que: What is the sequence of running of various filters?**

**Ans:**

* The **Authentication filter** **runs before any other filter or action** method.
* The **Authorization filter** **runs after the Authentication filter and before any other filter or action** method.
* The **Action filter** **runs before and after any action** method.
* The **Result filter** **runs before and after execution of any view** result.
* The **Exception filter** **runs only if action methods, filters or action results throw an exception.**

**--------------------------------------------------------------------------------------------------------------------------------------------------**

**Que: What is Custom Action Filter?**

**Ans:**

* We write **custom action filters for logging, or for saving data to database before any action execution.**
* We could also have one for **fetching data from the database and setting it as the global value of the application.**

**--------------------------------------------------------------------------------------------------------------------------------------------------**

**Que: What are the reason to create custom action filters?**

**Ans:** There are following reasons to create custom action filters.

* **Creating a privileged authorization.**
* **Logging the user request.**
* **Pre-processing image upload.**
* **Fetching data to display in layout menu.**
* **Localization of the application.**
* **Reading browser user agent information to perform the particular task.**
* **Caching.**

**--------------------------------------------------------------------------------------------------------------------------------------------------**

**Que: What is Authentication Filter in MVC application?**

**Ans: Authentication Filters:**

* **Authentication filter run before any other filter or action method.**
* **Authentication confirms that you are a valid or invalid user.**

**Apply Filters**

**Authentication Filters**

**Run First**

**Other filters like Authorization filter, Action Filter, Result Filters, Exception Filter) or Action Method (like Inbox)**

**Run Second**

**--------------------------------------------------------------------------------------------------------------------------------------------------**

**Que: How to use and create “Custom Authentication Filters” in MVC application?**

**Ans: Step 1:** You have to Implements the IAuthenticationFilter interface in your own custom class.

public interface IAuthenticationFilter

{

//OnAuthentication method is excuted first and can be used to perform any needed authentication.

void OnAuthentication(AuthenticationContext filterContext);

// OnAuthenticationChallenge method is used to restrict access based upon the authenticated users principle.

void OnAuthenticationChallenge(AuthenticationChallengeContext filterContext);

}

**Step 2:**

* Create own custom class to implement the ActionFilterAttribute, IAuthenticationFilter.
* Override the OnAuthentication and OnAuthenticationChallenge method.

public class MyOwnAuthAttribute:ActionFilterAttribute, IAuthenticationFilter

{

public void OnAuthentication(AuthenticationContextFilterContext)

{

//Logic for authenticating a user.

}

//Runs after the OnAuthentication method

public void OnAuthenticationChallenge(AuthenticationChallengeContextFilterContext)

{

//TODO: Additional tasks on the request.

}

}

**Step 3:**

Now we can apply **MyOwnAuthAttribute** on H**omeController** class.

[MyOwnAuthAttribute]

public class ActionFilterDemoController : Controller

{

public ActionResult Index()

{

throw new NullReferenceException();

}

public ActionResult About()

{

return View();

}

}

**Que: What is Authorization Filter in MVC application?**

**Ans: Authorization Filters:**

* Authorization filters run before executing the Action Method.
* Authorize filter attribute implements the IAuthorizationFilter Interface.

public interface IAuthoriztionFilter

{

Void OnAuthorization(AuthorizationContext filterContext);

}

* The AuthorizeAttribute class provides the following methods to override in the CustomAutorizeAttribute class

**--------------------------------------------------------------------------------------------------------------------------------------------------**

**Que: How to create and use “Custom Authorization Filter” in MVC application?**

**Ans:**

Step 1:

* Create a custom Authorize filter class which implements by IAuthorizationFilter interface and FilterAttribute class.

public class MyAuthorizationFilterAttribute : FilterAttribute, IAuthorizationFilter

{

public void OnAuthorization(AuthorizationContext filterContext)

{

if(filterContext.HttpContext.Request.IsAuthenticated)

{

//The Action Filter Logic

}

}

}

* In this way you can make CustomAuthorize filter attribute either by implementing IAuthorizationFilter interface or bye inheriting and overriding above the required methods of AuthorizeAttribute class.

Step 2:

* Now we can apply MyAuthorizationFilterAttribute on Action Method.

public class HomeController: Controller

{

public ActionResult Index()

{

viewBag.Message=”Welcome to Wiez Tech”;

return View();

}

}

* Examples of Built-in Authorization Filter
* AuthorizeAttribute
* RequestHttpsAttribute

**--------------------------------------------------------------------------------------------------------------------------------------------------**

**Que: What is AuthorizeAttribute and How to use in MVC application?**

**Ans:**

* Authorization filter **allowing only authorized user** to log in the application.

public class HomeController :Controller

{

[Authorize]

public ActionFilter Index()

{

ViewBag.Message=”Wiez tech”;

return View();

}

[Authorize(Role=”admin”)]

public ActionFilter AdminIndex()

{

ViewBag.Message=”Admin Wiez tech”;

return View();

}

}

* If you would try to access the application without logging in, it will throw and error like “Server Error in ‘/’ Application.”

**--------------------------------------------------------------------------------------------------------------------------------------------------**

**Que: What is RequireHttpsAttribute in MVC and How to use in MVC application?**

**Ans:**

* RequireHttpsAttribute indicate that **this action method should only accessible by SSL or HTTPS.**
* Whenever action methods are marked with RequiredHttps, at that time **we cannot browse it with HTTP protocol.**
* If we try to browse it with HTTP then it will redirected to HTTPS url.
* We can also use **this attribute at Controller level.** When RequireHttps is applied at controller level, all action method **within controller would be only accessible via HTTPS.**
* We should **apply this attribute**, whenever we are **obtaining sensitive data from user for e.g. login credential, credit card information, etc.**

[RequireHttps]

public ActionResult SignIn()

{

Return View();

}

If your server does not support SSL then requested HTTP url will give issue an HTTP 302 redirection and it will be redirected to HTTPS url. (eg. <https://localhost>)

**--------------------------------------------------------------------------------------------------------------------------------------------------**

Que: What is Action Filter in MVC

**--------------------------------------------------------------------------------------------------------------------------------------------------**

Que: Give me Example of Action Filter?

**--------------------------------------------------------------------------------------------------------------------------------------------------**

Que:

**--------------------------------------------------------------------------------------------------------------------------------------------------**

Que:

**--------------------------------------------------------------------------------------------------------------------------------------------------**

Que:

**--------------------------------------------------------------------------------------------------------------------------------------------------**

Que:

**--------------------------------------------------------------------------------------------------------------------------------------------------**

Que:

**--------------------------------------------------------------------------------------------------------------------------------------------------**

Que:

**--------------------------------------------------------------------------------------------------------------------------------------------------**

Que:

**--------------------------------------------------------------------------------------------------------------------------------------------------**

Que:

**--------------------------------------------------------------------------------------------------------------------------------------------------**

Que:

**--------------------------------------------------------------------------------------------------------------------------------------------------**

Que:

**--------------------------------------------------------------------------------------------------------------------------------------------------**

Que:

**--------------------------------------------------------------------------------------------------------------------------------------------------**

Que: