**Chapter 8: Asp.Net Applications Configuration and Deployment of Application**

Explain web.config file

* Each and every web application has its own web.config file which can configure entire web application or particular web page of your web application.

* Initially, web.config file is generated from machine.config fie.

* It allows you to define the configuration settings at time of developing application.

* Web.config files are stored in XML format.

* Web.config file contains following configuration settings:

1. **<ConnectionString>**: It allows you to create connection string variables which can be used in any web form of your web application.

1. **<appsettings>:** It allows you to add ore remove any application specific variables.

1. **<System.web>:** It is used to configure your application like <caching>, <<Authentication>, <trace>, etc.

1. **<System.Codedom>:** This section allows compiler specific information used in your web application.
   1. **<System.WebServer>**: This section stores the information about the web server which will be used for running your website.

1. **<runtime>:** This section is used to specify some runtime setting for your web application.

Explain Common Configuration Settings.

* Some Common Configuration settings are:
* **Tracing Web Application:**

Tracing allows you to check for any type of bugs that your web application may have. Along with tracing your web application, you can also maintain log files which give you some additional information. These log files can help you to find out some problems that occur during web application usage.

* **Customizing Errors**:

While using web site, sometimes different error occur. You are not aware that which error is going to come. Web Site users feel uncomfortable when they get some unexpected screen. By customizing errors, you can redirect your error to specific page which can display error but in your design specific way.

* **Authentication and Authorization**:

Authenticated users are requirement for most of the web sites. To allow authenticated users, you must authorize users in such a way that any even any unauthorized user visits your web site, should not be able to harm your information. This can be done with the help of combining Authentication and Authorization.

* **Enabling Role Manager:**

Enabling Role Manager is a feature which is used along with Authentication and Authorization. This feature can be used to allow some specific users. For example, some of the web page of your website are important, you want only “Administrator” level user can access. “Guest” users should not be able to access these pages

* **Session Configuration**:

 Session Configuration allows you to specify session settings like Timeout Period, Session Mode, Cookie Based Session or CookielessSession, SqlCommandTimeout, NetworkTimeout, etc.

* **Trust Levels**:

There Can be different types of web site Users. For example, Web master will have access to all the web pages of web site. Web Developer will have some limited access to web pages of Website; Administrator will have access to all the back end pages for web site administration where at last level is Users who will have access to only those pages which are informative. So different types of trust levels can be defined using trust levels.

* **Web Service Configuration**:

Web services are configured to lock some resources from being accessed by some specific users.

* **Caching:**

    Caching in short means, keeping your output or data into cache memory so that it can be accessed in a faster way instead of gathering output from web Server or searching data from Database.

Explain Error Handling

* Error Handling is one of the most important part of any web application. Each and every error has to be caught and suitable action has to be taken to solve that error. Asp.net provides a simple yet powerful way to handle errors in application.

* Asp.net provides three methods or levels to catch errors which are as below.

1. Page\_Error() event of .aspx file.
2. Application\_Error() event in Global.asax file
3. <customErrors>section of the web.config file.

* These all methods are occur one after another in sequence. This sequence are given below.

**Level-1**:-First of all Page\_Error() event of particular .aspx file (Web Form) is executed if you have written and handled it.

**Level-2:**-After going into Page\_Error () next,control goes to Application\_Error() event which is written in Global.asax file.

**Level-3:-**At last ,if you have written <customErrors>section in Web.config,control goes to specified page.

* Control goes from one level to another if error is not cleared. If in any level error is cleared,control is not transferred to next level.

**Server.ClearError() Method:**

* If you want to resolve any error this can be done by using Server.ClearError() method. If you clear error in Page\_Error() event,control will not goes to the next level.

**Server.GetLastError() Method:**

* To know which error is occurred you can use Server.GetLastError() method. This method gives you the most recent error.

* You need to save this method under exception object as follows:

* Exception err=Server.GetLastError();

* <custom Errors> section of web.config file comes at third level after any error occurs in your web application. This section used to handle different types of errors.

* This section helps you to display your own customized error message. This is useful in following way.

* Customized error message can be understood by user because it is written in easy language.
* Users do not feel that they are thrown out of the scope due to some error.

* In this section you can specify redirect page as a default error page based on the HTTP error code.

* This section contains following attributes:

**1.defaultRedirect:** This is optional attribute. It specifies the URL of the custom error page .It indicates this page should be displayed instead of Runtime Error.

**2.mode:**The mode attribute have three values.

* On:Specifies that custom errors are enabled.
* Off:Specifies that custom errors are disabled. This allows the display of detailed errors.
* RemoteOnly:This option specifies that custom errors are displays only to remote clients and ASP.NET errors are displays to the local host.

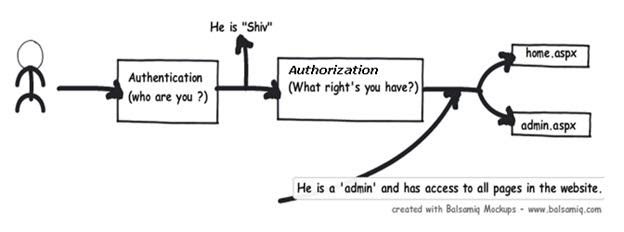
* ***Authentication*** is knowing the identity of the user. For example, Alice logs in with her username and password, and the server uses the password to authenticate Alice.

Explain Authentication and Authorization in detail.

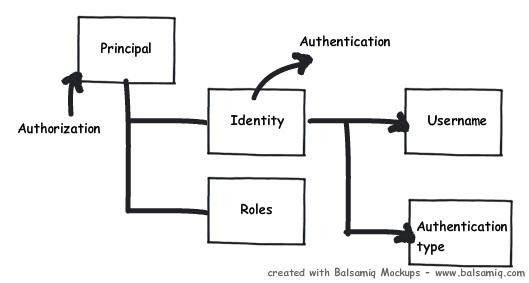
* ***Authorization*** is deciding whether a user is allowed to perform an action. For example, Alice has permission to get a resource but not create a resource.

* The same dictionary meaning applies to ASP.NET as well. In ASP.NET authentication means to identify the user or in other words its nothing but to validate that he exists in your database and he is the proper user. **Authorization m**eans does he have access to a particular resource on the IIS website. A resource can be an ASP.NET web page, media files (MP4, GIF, JPEG etc), compressed file (ZIP, RAR) etc.

* So the first process which happens is authentication and then authorization. Below is a simple graphical representation of authentication and authorization. So when the user enters ‘userid’ and ‘password’ he is first authenticated and identified by the user name.   
  Now when the user starts accessing resources like pages, **ASPDOTNETauthentication**, videos etc, he is checked whether he has the necessary access for the resources. The process of identifying the rights for resources is termed as ‘Authorization’.

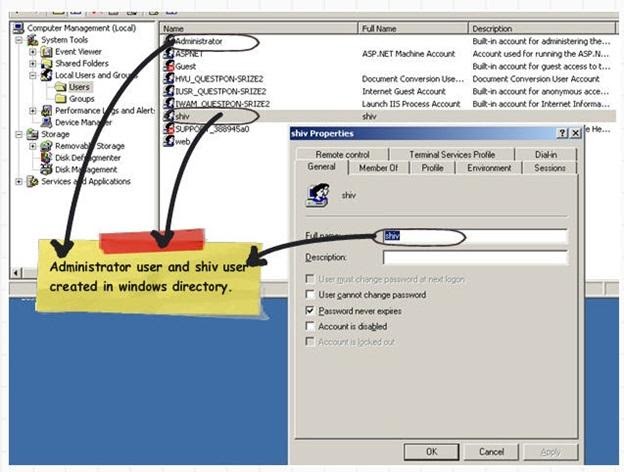


* To put it in simple words to identify “he is shiv” is authentication and to identify that “Shiv is admin” is authorization.
* Detecting authentication and authorization: - The principal and identity objects
* At any moment of time if you want to know who the user is and what kind of authentication type he using you can use the identity object. If you want to know what kind of roles it’s associated with then we need to use the principal object. In other words to get authentication details we need to the identity object and to know about authorization details of that identity we need the principal object.



* **Types of authentication and authorization in ASP.NET:**

There are three ways of doing authentication and authorization in ASP.NET:-  
• **Windows authentication: -** In this methodology ASP.NET web pages will use local windows users and groups to authenticate and authorize resources.  
  
• **Forms Authentication: -** This is a cookie based authentication where username and password are stored on client machines as cookie files or they are sent through URL for every request. Form-based authentication presents the user with an HTML-based Web page that prompts the user for credentials.

**• Passport authentication :-** Passport authentication is based on the passport website provided  
by the Microsoft .So when user logins with credentials it will be reached to the passport website ( i.e. hotmail,devhood,windows live etc) where authentication will happen.

 If Authentication is successful it will return a token to your website.