Introduction of ASP.NET

BCA SEM-VI

Adv. NET using c#

UNIT-1

THE INSB IITMS BCA & PGDCA COLLEGE, IDAR

Introduction of .Net

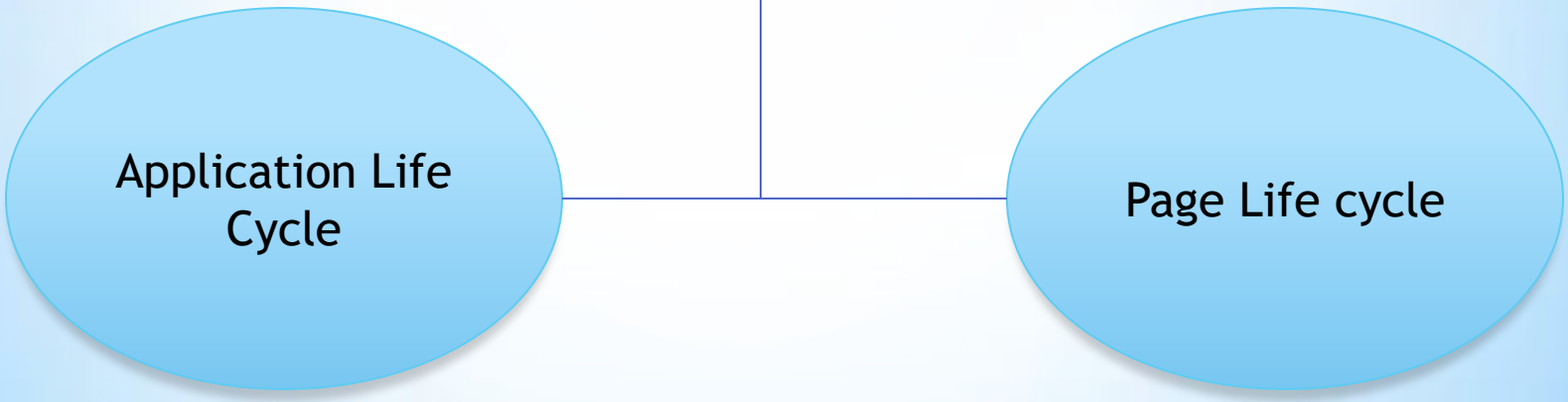
- Asp.net is a server side technology used for developing dynamic websites, web services, and web application.
- Asp.net helps in making website and web application in a colorful way.
- ASP.NET Stand for **Active Server Page and .NET is Network Enabled technologies.**
- Its is provided by Microsoft.

- **What is Asp.Net?**

ASP.NET Page Life Cycle

- ASP.NET Processes pages to produce dynamic output.
- The application and its pages are instantiated and processed.
- ASP.NET compiles the pages dynamically.

ASP.NET Life cycle is divided into two groups



Asp.NET Application Life Cycle

Application Start

Application start is a method that can be executed by the web server when a user makes a request.

Object Creation

The HTTP requests contain all the information about the current request like cookies and browser information.

HTTP Application

An object helps to process each subsequent request sent to the application that is created by the webserver.

Dispose

Dispose is an event that can be called before the application is destroyed. This helps to release manually unmanaged resources.

Application End

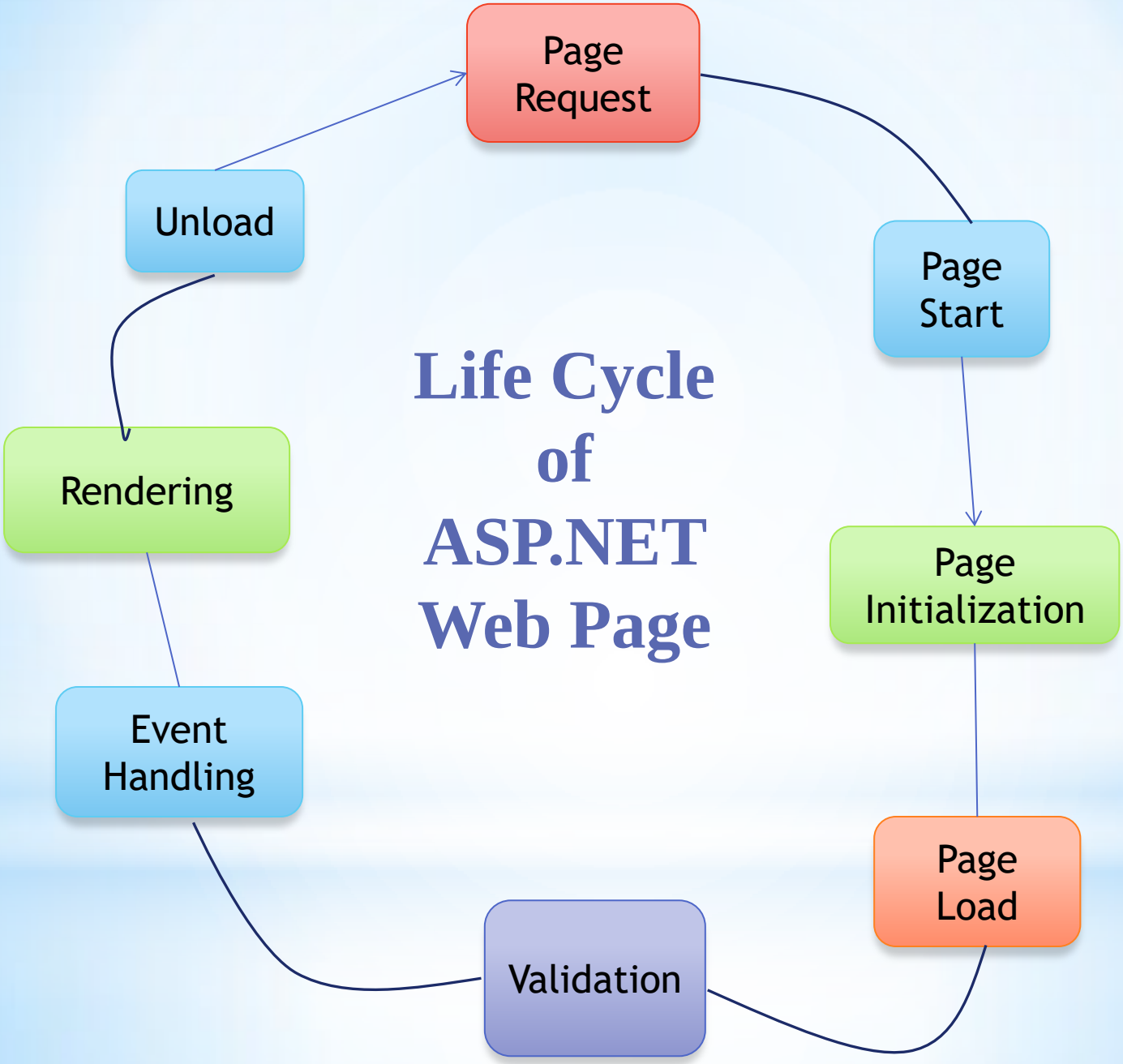
Application end is the final part of the application this can help to unload the memory.

ASP.NET Page Life cycle

ASP.NET web page life cycle has specific steps that carried out during the execution.

These phases include initialization, restoring and execution.

ASP.NET Web page also automatic event, which means that asp.net searches for methods that have particular names and automatically executes those methods when contain event are raised.


Page Request
Page Start
Page Initialization
Page Load
Validation
Event Handling
Rendering
Unload
Life Cycle of ASP.NET Web Page

- **Page Request:-** When the user requests a page the server checks the request and then compiles the pages.
- **Page Start:-** in this phase two steps are carried out request and response the request holds all information form the page.
- **Page Initialization:-** in this phase all the controls on the page are set and each has a particular ID, and themes are applied to the pages.
- **Page load:-** in this phase all the control properties are loaded and information is set using view state.
- **Validation:-** Validation happens when the page execution goes successful, and then it returns true else the execution fails it returns false.
- **Event handling:-** Event handling happens in response to validation in this case the page is loaded again. So the Postback event handler is called.
- **Rendering:-** Rendering occurs before all the response information is sent to the user it also stores all information sent to the user.
- **Unload:-** Unload phase helps to clean all the unwanted information it also cleans the memory once the page output is sent to the user.

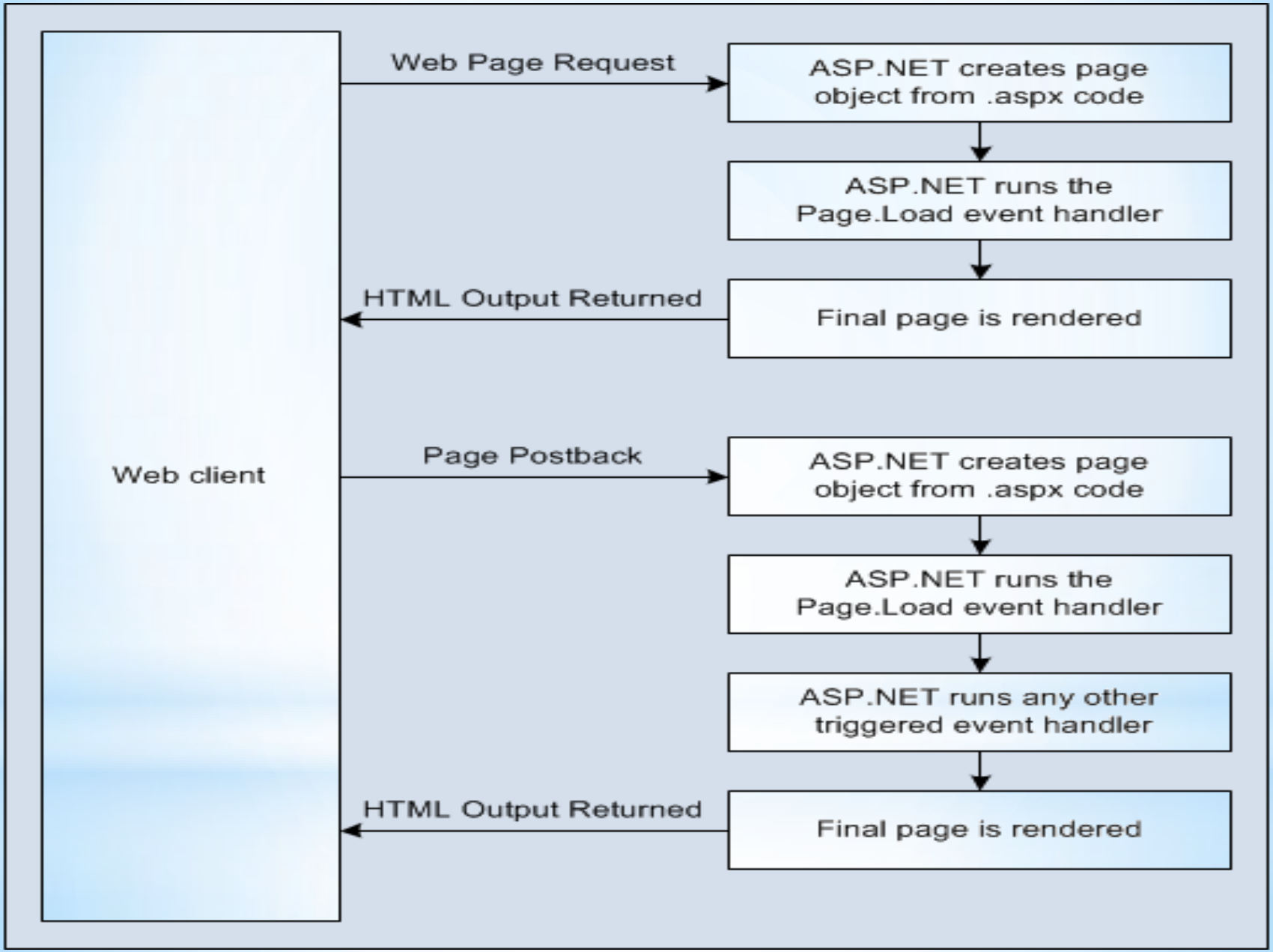
Page Processing Sequence

→ The ASP.NET page processing sequence, or page life cycle, is a series of steps that a page goes through when it's requested, processed, and rendered by the server

→These phases include page request to unload page process are the page processing sequence.

→ If you want to a change event for a web control, you need to set its postback property to true. that means that when the user click a radio button or checkbox the page will be resubmitted to the server.

→ Ex. When the user changes the text in a textbox (textChanged event) or chooses a new item in a list (selectedIndexChanged Event). You might want to response to these events but without postback you code has no way to run.


Web client
Web Page Request
ASP.NET creates page object from .aspx code
ASP.NET runs the Page.Load event handler
Final page is rendered
HTML Output Returned
Page Postback
ASP.NET creates page object from .aspx code
ASP.NET runs the Page.Load event handler
ASP.NET runs any other triggered event handler
Final page is rendered
HTML Output Returned

The following are the most common page events.

- **PreInit**
- **Init**
- **Load**
- **Control Events**
- **Load Complete**
- **Render**
- **Unload**

Preinit:- Preinit is the first event that occurs during the page life cycle. After the start stage is complete and before the initialization stage.

- Create or re-create dynamic controls.
- **IsPostback** property to determine whether this is the first time the page process.
- Set a master page dynamically.

Init:- This event after all controls have been initializad and any skin settings have been applied. The Init event of individual controls before the **Init** event of the page.

Load:- Use the OnLoad event method to set properties in controls and to establish database connections.

- The page class calls the unload event recursively does the same for each child control until the page and all control are loaded.

Control Events:- Use these events to handle specific control events such as a Button control's Click event.

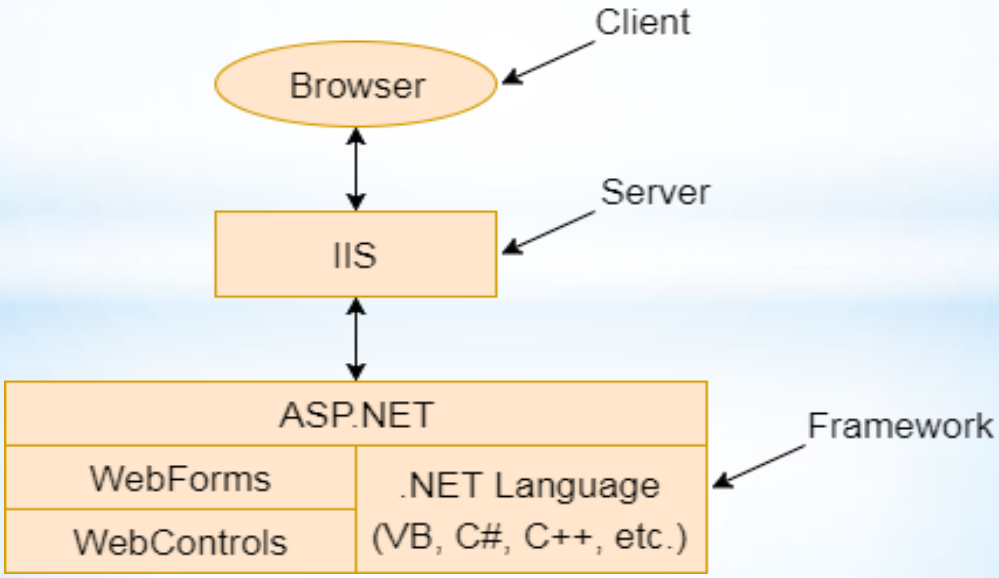
Render:- This is not an event. At this stage of processing the page object calls this method on each control.

- All asp.net web server controls have a render method that writes out the control's markup to send to the browser.

Unload:- This event to do final cleanup for specific controls such as closing control specific database connections or finishing up logging or other request specific tasks.

Introduction to Web Form & Event

- Web Forms are web pages built on the ASP.NET Technology. It executes on the server and generates output to the browser. It is compatible to any browser to any language supported by .NET common language runtime. It is flexible and allows us to create and add custom controls.



Event	Attribute	Controls
Click	OnClick	Button, image button, link button, image map
Command	OnCommand	Button, image button, link button
TextChanged	OnTextChanged	Text box
SelectedIndexChanged	OnSelectedIndexChanged	Drop-down list, list box, radio button list, check box list.
CheckedChanged	OnCheckedChanged	Check box, radic button

Some events cause the form to be posted back to the server immediately, these are called the postback events. For example, the click event such as, Button.Click.

Control	Default Event
AdRotator	AdCreated
BulletedList	Click
Button	Click
Calender	SelectionChanged
CheckBox	CheckedChanged
CheckBoxList	SelectedIndexChanged
DataGrid	SelectedIndexChanged
DataList	SelectedIndexChanged
DropDownList	SelectedIndexChanged
HyperLink	Click
ImageButton	Click
ImageMap	Click
LinkButton	Click

LinkButton	Click
ListBox	SelectedIndexChanged
Menu	MenuItemClick
RadioButton	CheckedChanged
RadioButtonList	SelectedIndexChanged

Coding Model

Coding Model

There are two type of Coding Model.

- **Inline Code**
- **Code behind**

Inline Code

- When we developing a web form or a web page. We will having some designer code which is required for designing a user interface(UI). I want to design a login form required for buttons, textbox's, and some controls all the design of code is called design code.

- When the user use the login and enter the username & password and click on login button. There should be some login validating the user name correct the password or not with the database code with after comparing with the database it make take the next page.

- Inline coding the designer code and business code both of them existing in single file and that particular file is going to be share with extension of **.aspx.**

- **C# code**

<% @ page language =“c#” %>

- .aspx.page

{

coding

….

design

}

Code-behind

- Code behind you can create a separate file (.cs file for c#) to match every .aspx.file. The .aspx file contains the user interface logic, which is the series of html code and asp.net tags that create controls on the page.

- The .cs file contains code only at the top of the aspx file you use a special page directive that identifies the matching code behind file.

C# code

<% @ page language =“c#” codebehind=“home.aspx.cs” inherits=“practical_default” %>

- **Codebehind.aspx**

  {

  …….

  Deign code

  ……..

  }

- **Codebehind.aspx.cs**

  {

  …….

  Coding Part

  ……..

  }

Application Directory Structure

App_data:- App_Data is the default director for the datastorage, database or XML file like Acces, SQL server express etc.

App_code:- Contains source code files, compiled dynamically to be used in the ASP.NET application.

Bin:- Contains .net assemblies containing precompiled web page and web service classes. You can use directly use assemblies in your application.

Global.asax Application File

- It is also known as asp.net application file.
- It is optional file that is located file in a application's root directory and is counterpart of Global.asa file in ASP.
- It is contains code for responding application and session level event by asp.net.
- At a runtime it is parsed and compiled into dynamically generated .net framework class derived from Httpapplication base class.
- To add this class into the application.=> select add new item=> select global application class
- By default its name is Global.asax click add button so it will be add the file.

Application Event

- Application_start():- when an application start , this event is fired it is executed only one time in the application life cycle. It executes again when we restart the application.
- Application_End():- it is fired when application ends its execution.
- Application_Error():- it is executed when an unhandled error or execution occurs in an application.
- Session_Start():- it is executed whenever new session starts. This is the best place to count user session for application.
- Session_End():- it is executed when any user session ends.

Web .Config

- Web configuration file is used to manage various settings that define a website.
- The settings are stored in xml files that are separate from your application.
- In this way you can configure settings independently from your code.
- Generally a website contains a single web.config file stored inside the application root directory.
- ASP.NET in particular use XML file formatted ‘config’ file to configuration application.

You add custom settings to a special element called <appsettings> note that this nested in the root **<configuration>** element.

```
<?XML version="1" encoding="utf-8">
    <appsettings>
        <!--- special application settings go here.-->
    </appsettings>
<system.web>
    <!--- special application settings go here.-->
</system.web>
</configuration>
```

- That means the default settings are used from the server machine.config file.
- It is never locked → its is easily accessed and replicated. → its easy to edit and understand.

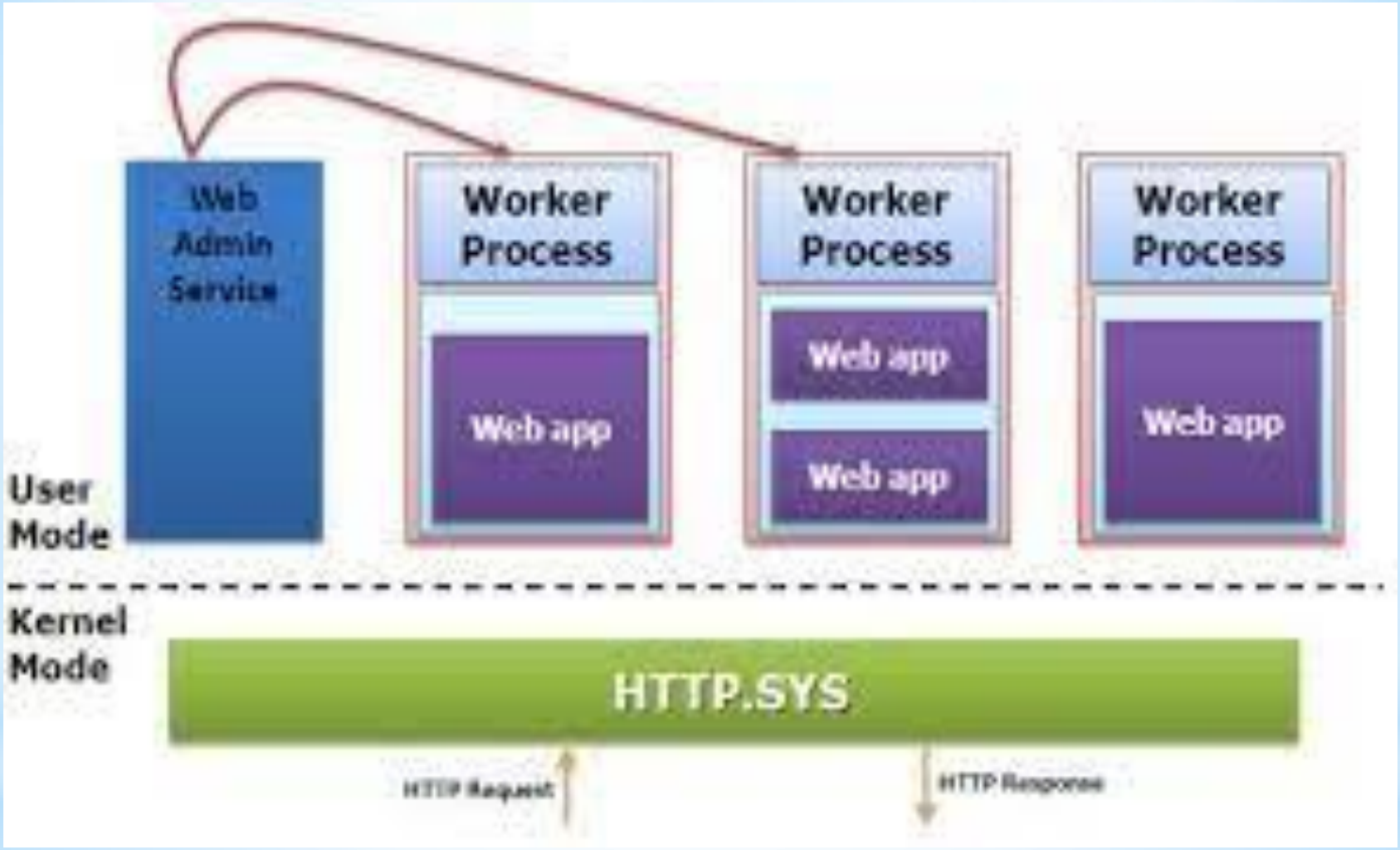
IIS Request Handling Process

- **IIS (Internet Information Server)** is one of the most powerful web servers from Microsoft that is used to host your ASP.NET Web application. IIS has it’s own ASP.NET Process Engine to handle the ASP.NET request. So, when a request comes from client to server, IIS takes that request and process it and send response back to clients.

- Before starting with a virtual directory and application pool and all other stuff let us have a quick look into the IIS process module and IIS request processing.

- We can divide the whole architecture into two layers.

- Kernel Mode
  → HTTP.SYS
- User Mode
  → Web Admin service
  → Virtual Directory


Web Admin Service
Worker Process
Web app
Worker Process
Web app
Web app
Worker Process
Web app
User Mode
Kernel Mode
HTTP.SYS
HTTP Request
HTTP Response

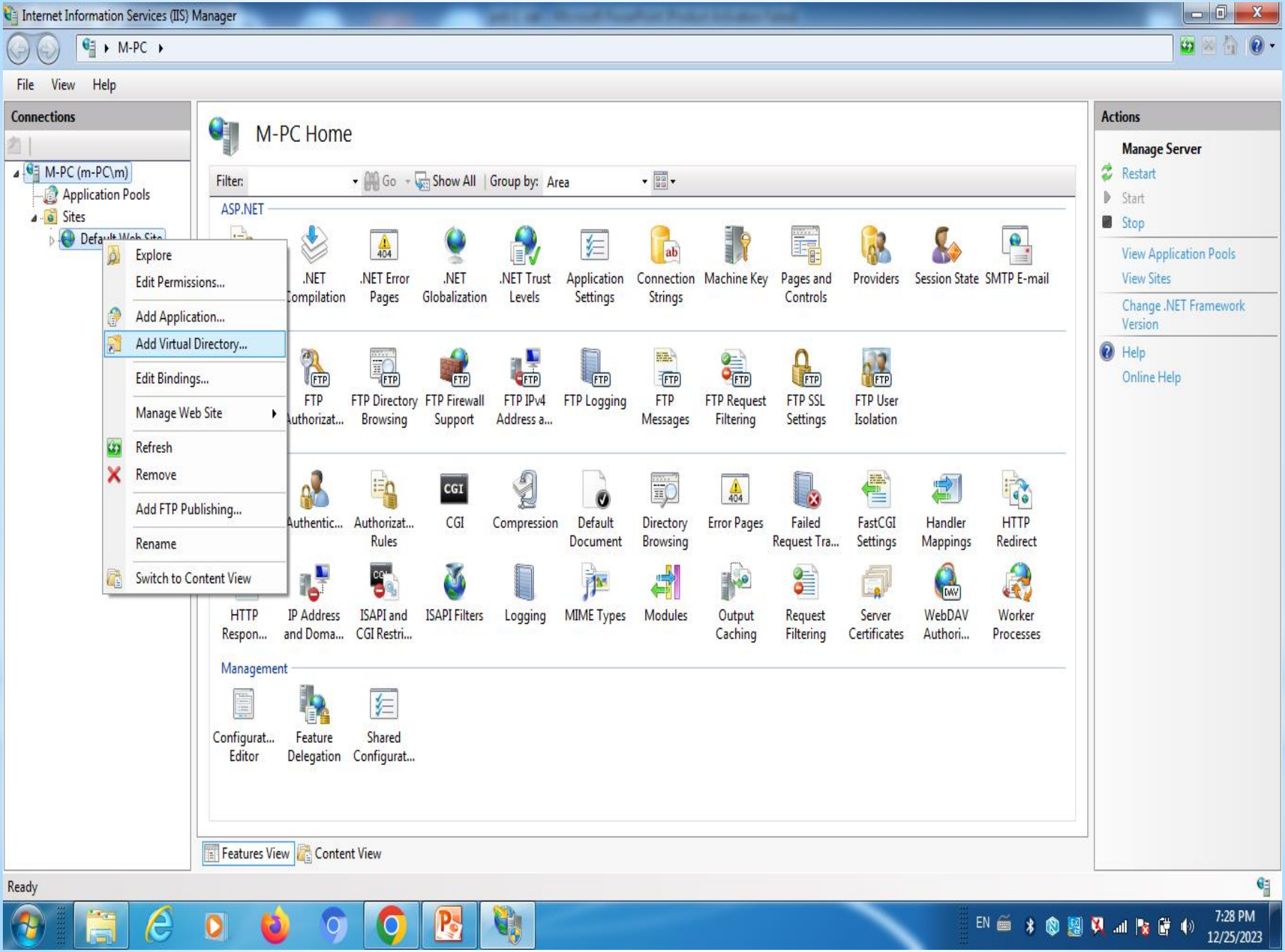
- **Kernel or user** model are http.sys is the heart of kernel mode which accepts raw requests from the client and pass it to a particular application pool.
- Below are the step of IIS request processing.

1. Client Request hits the web server. Internally this request comes to Kernel Layer of IIS means at HTTP.SYS.
2. HTTP.SYS indentifies the name and ID of Application Pool for that ASP.NET Request.
3. Now Request comes to user level of IIS. WAS (Web Admin Service) puts the request from HTTP.SYS to Application Pool.
4. When Application pool receives the request, it simply passes the request to worker process.

5. The worker process “w3wp.exe” looks up the URL of the request in order to load the correct ISAPI extension (aspnet_isapi.dll).
6. ISAPI creates an HTTPRuntime Object to Process the request via HTTPModule and HTTPHanlder.HTTPRuntime is an entry point of the application.
→After these steps, after that the ASP.NET Page LifeCycle events starts

Creating a Virtual Directory

- → A virtual directory in Internet Information Services (IIS) is a directory name that maps to a physical directory on a remote or local server. The directory name becomes part of the application's URL, allowing users to access the physical directory's content through a browser.
- → When you deploy your web application to a web server, its exposed through something called a virtual directory. A virtual directory is simply the public face of your website directory.


Internet Information Services (IIS) Manager
M-PC
File View Help
Connections
M-PC (m-PC\m)
Application Pools
Sites
Default Web Site
Explore
Edit Permissions...
Add Application...
Add Virtual Directory...
Edit Bindings...
Manage Web Site
Refresh
Remove
Add FTP Publishing...
Rename
Switch to Content View
M-PC Home
Filter:
Go
Show All
Group by: Area
ASP.NET
.NET Compilation
.NET Error Pages
.NET Globalization
.NET Trust Levels
Application Settings
Connection Strings
Machine Key
Pages and Controls
Providers
Session State
SMTP E-mail
FTP Authorizat...
FTP Directory Browsing
FTP Firewall Support
FTP IPv4 Address a...
FTP Logging
FTP Messages
FTP Request Filtering
FTP SSL Settings
FTP User Isolation
Authentic...
Authorizat... Rules
CGI
Compression
Default Document
Directory Browsing
Error Pages
Failed Request Tra...
FastCGI Settings
Handler Mappings
HTTP Redirect
HTTP Respon...
IP Address and Doma...
ISAPI and CGI Restri...
ISAPI Filters
Logging
MIME Types
Modules
Output Caching
Request Filtering
Server Certificates
WebDAV Authori...
Worker Processes
Management
Configurat... Editor
Feature Delegation
Shared Configurat...
Features View
Content View
Actions
Manage Server
Restart
Start
Stop
View Application Pools
View Sites
Change .NET Framework Version
Help
Online Help
Ready
EN
7:28 PM
12/25/2023