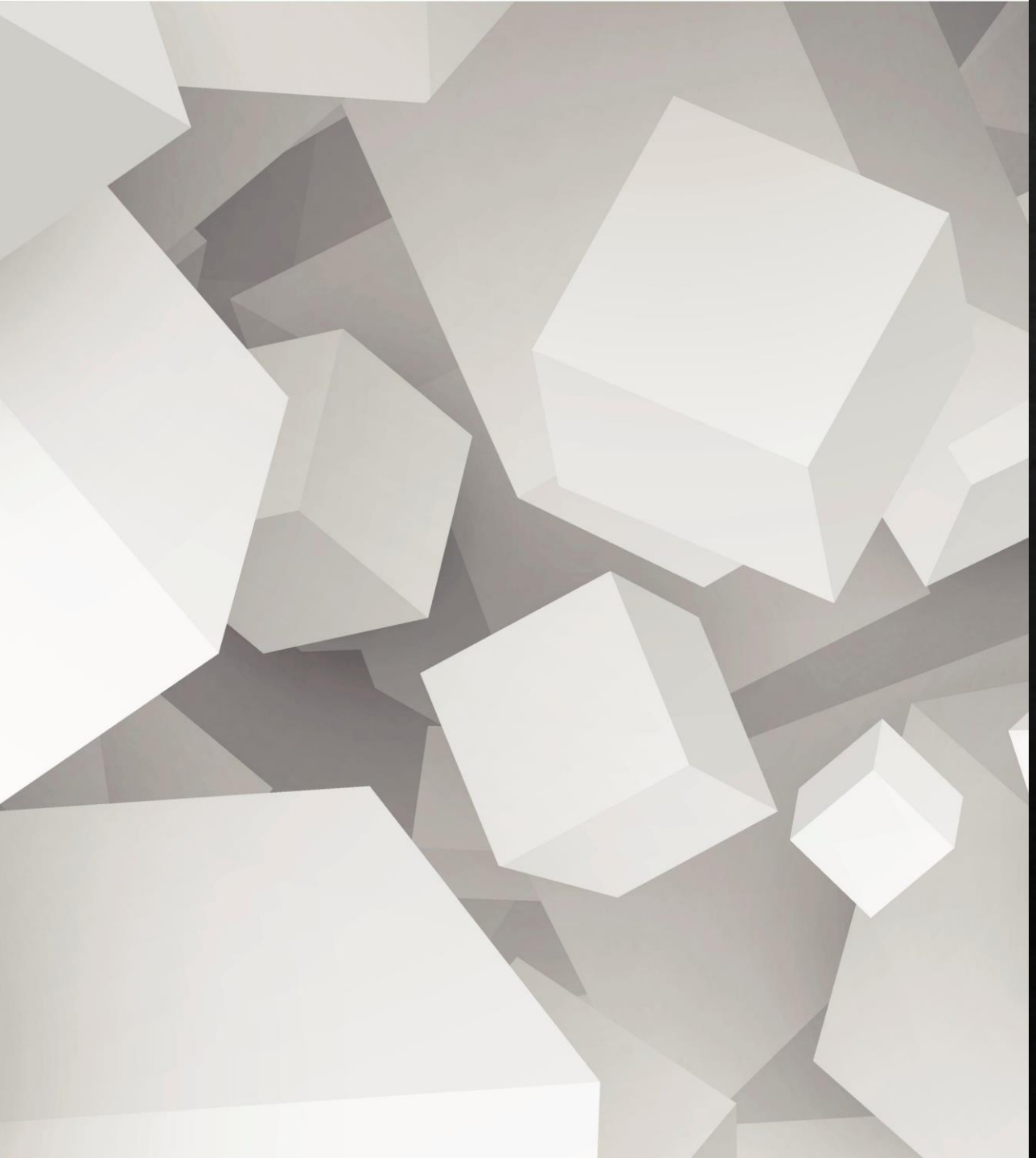
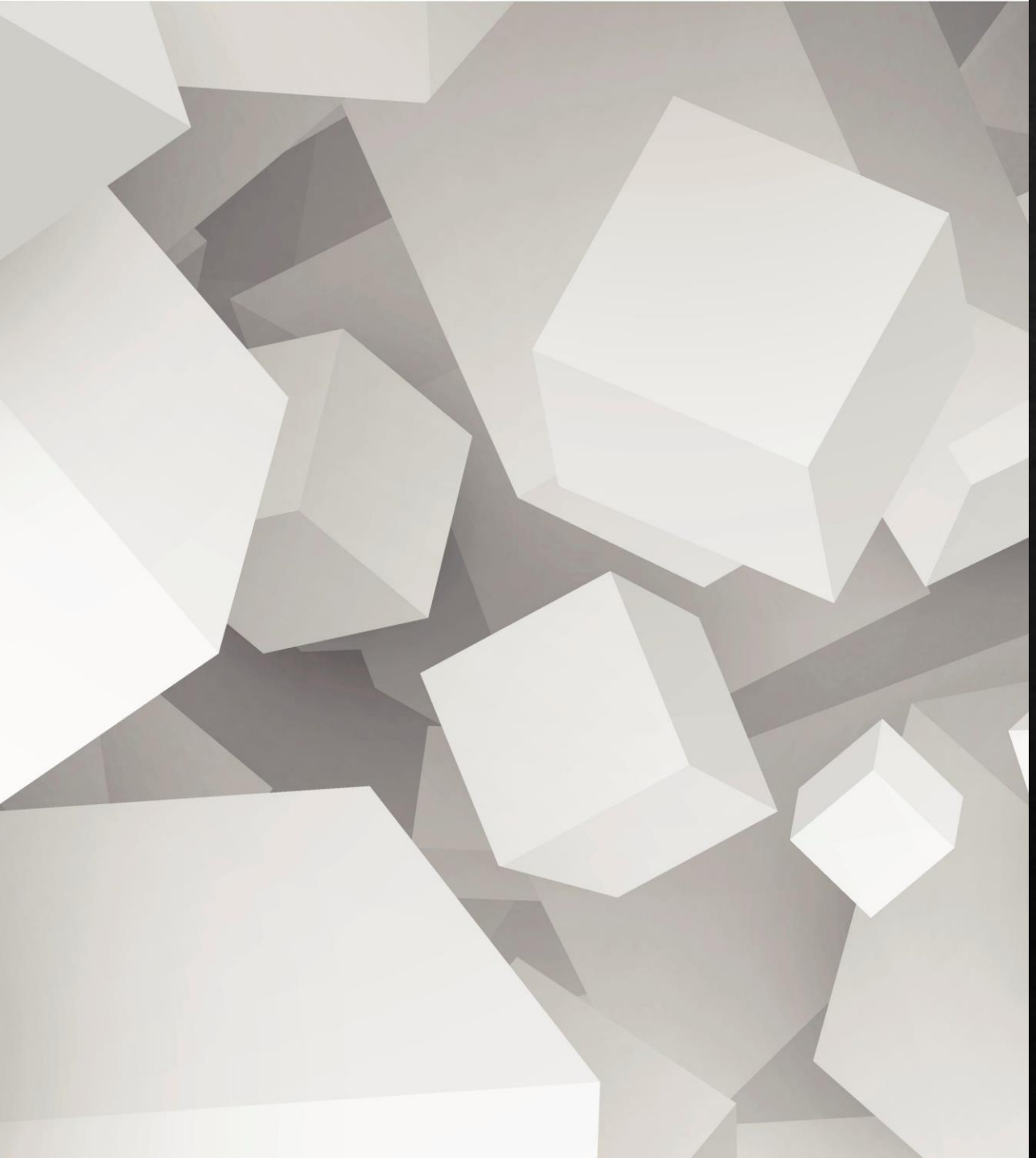
.NET Framework Web Forms

Arctech Info Private Limited



Topics 1/2

- HTTP revisited
- GET and POST revisited
- Client Side vs Server Side processing
- Asp.NET Project Structure in Visual Studio
- Web.config
- Server Side Web Application
- Comparison with Windows Forms
- PostBack with asp.net framework



Topics 2/2

- Asp.NET Web Form & Common Toolbox Controls
- Master Pages
- User Control
- Themes
- Asp.NET Framework Page lifecycle
- Stateless vs State & State Management
- ViewState & Hidden Fields, Session, Application, Cookies, Cache
- Request & Response
- Introduction to IIS Express & IIS

HTTP revisited

- HTML revision

```
<html>
  <head>
    <title>Hello World Page</title>
  </head>
  <body style="font-family: Arial;">
    <h1>Hello World</h1>
    <div style="font-size: 42">
      <p>Welcome to the Hello World Page </p>
    </div>
  </body>
</html>
```

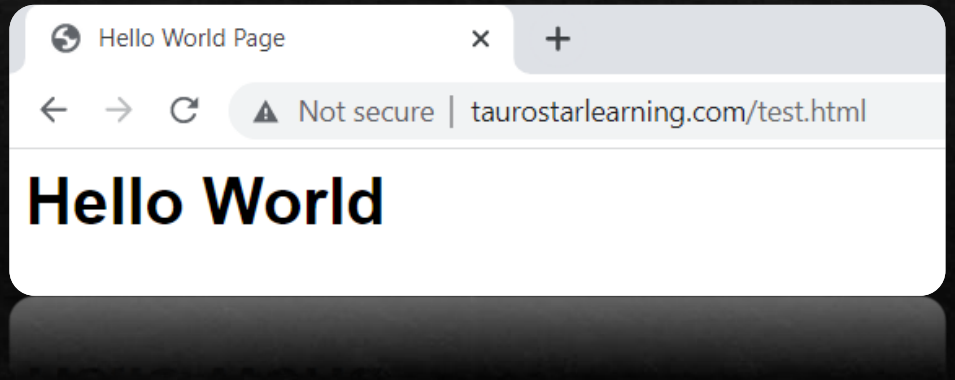
HTTP revisited

- Hyper Text Transfer Protocol
- Is the foundation of data communication between client and server
- If you type http://www.google.com on the URL bar of a browser
  - Browser packages this request using the HTTP Request rules
  - Sends the request package over the internet, eventually reaching the google servers.
  - The google web server parses the request package to understand what file is requested.
  - The web server then packages the response using data from the specified file (or index.html if not specified)
  - Sends the response package over the internet, eventually reaching our computer and browser
  - Browser then parses the response package (depending on type of data, either displays it, downloads it, etc.)

HTTP revisited

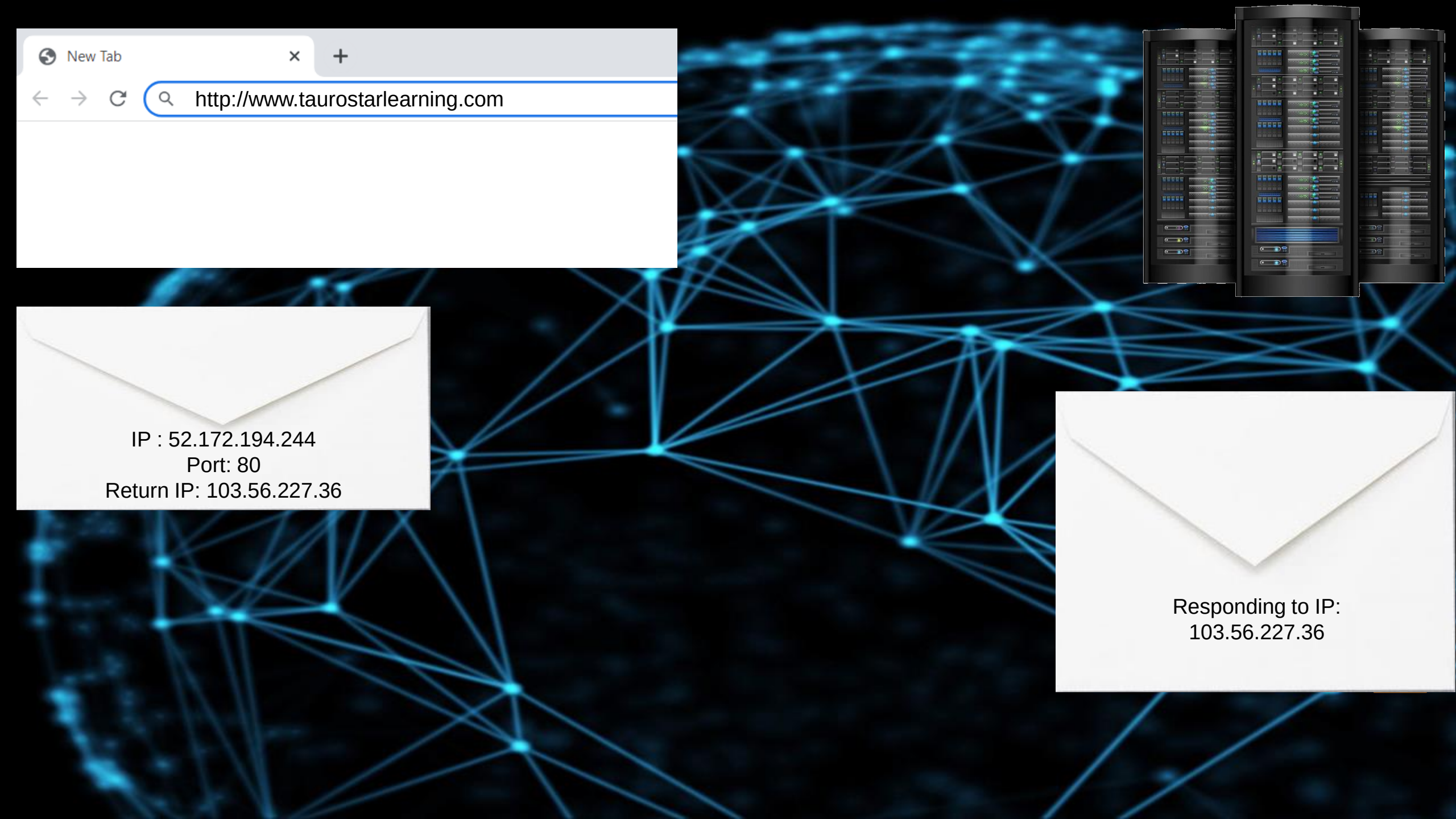
- HTTP Request

```
GET /test.html HTTP/1.1
Host: www.taurostarlearning.com
```



- HTTP Response

```
HTTP/1.1 200 OK
Content-Type: text/html
Content-Length: 387
<html>
 <body style="font-family: Arial;">
   <h1>Hello World</h1>
   ...
   ...
 </body>
</html>
```


New Tab
http://www.taurostarlearning.com
IP : 52.172.194.244
Port: 80
Return IP: 103.56.227.36
Responding to IP:
103.56.227.36

GET and POST revisited

- GET is used to request data from the specified server.
- GET is one of the most common HTTP methods.
- HTTP GET Example
  - `https://www.google.com/search?q=car`
  - ```
    GET /search?q=car HTTP/1.1
    Host: www.google.com
    ```

- POST is used to send data to a server to create/update a resource.
- POST is also one of the most common HTTP methods.
- HTTP POST Example
  - `http://www.somesite.com/login`
  - ```
    POST /login HTTP/1.1
    Host: www.somesite.com

    login=raman@gmail.com
    password=Test@123
    ```

Client-Side vs Server-Side

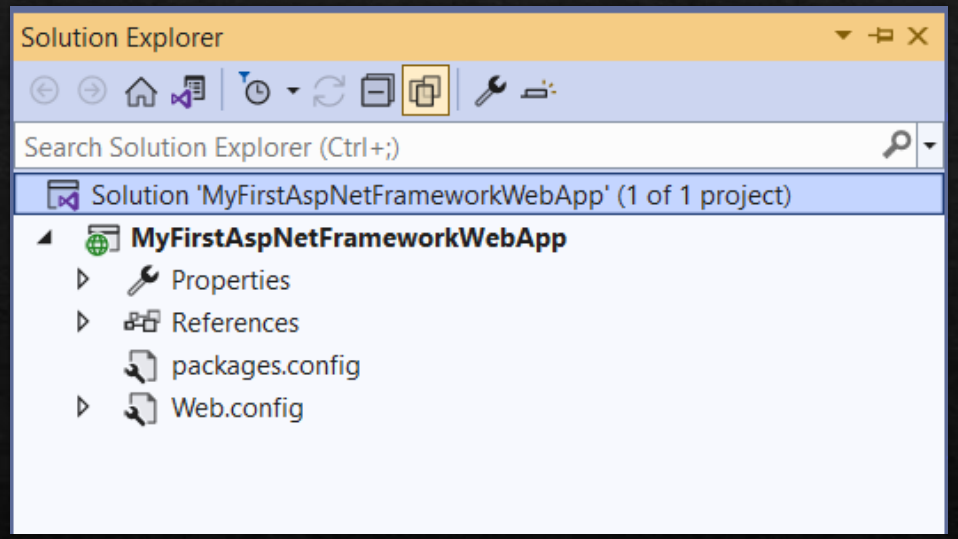
Client Side Processing

- It runs on the web browser on the user's computer. Google Chrome, Microsoft Edge, Firefox etc.
- Client-Side Scripts are usually written in JavaScript.
- It is used for validations, animations and other client-side logic.
- Client-side scripts running within a black box in the browser, which allow only limited functionality.
- You cannot access the file system on the client computer or other tabs in the browser.
- Users can view the source code of client-side scripting.

Server Side Processing

- It runs on the Web Server on a remote server machine or cloud machine. IIS, Apache, Azure Web App Service, etc.
- Server-Side Scripts are written in C#, Java, PHP, etc.
- It is used for creating dynamic pages.
- It can access databases, files etc on the server.
- The html content, can be altered to display data from the database before sending it to the browser.
- Users cannot view the source code of server-side scripting.

Asp.NET Project Structure in Visual Studio



Asp.NET Project Structure in Visual Studio

- App_Code
  - App_Code Folder stores cs files
- App_Data
  - Used as a data storage for the web application (.mdf, .mdb, and XML, etc.)
- Bin
  - Contains compiled assemblies (.dll files) for controls, components, or other code
- packages.config
  - Used by nuget package manager for maintaining the list of dependencies used by the current Project
- web.config
  - allows you to define configuration settings
  - are stored in XML format

Web.config

```
<?xml version="1.0" encoding="utf-8"?>

<configuration>

    <connectionStrings>
        <add name=“Connection1” connectionString=“”
    </connectionStrings>

    <appSettings>
        <add key="MinAllowedAge" value="18" />
        <add key="MaxAllowedAge" value="60" />
    </appSettings>

    <system.web>
    </system.web>

</configuration>
```

Server Side Web Application

- See Sample
  - Simple local HTML Page
  - HTML Page uploaded to a server
  - Chrome Developer Tab
  - Simple Asp.NET Framework WebApp with single page
  - Locally with dev web server

Comparison with Windows Forms

Windows Forms

- Create rich applications that install and run locally on a machine.
- Add controls on the Form either from the toolbox or writing C# code.
- Adding controls from the toolbox auto generates the C# code.
- You can set control properties and handle events in the code-behind file

Web Forms

- Create Web-based applications that display in a Web browser
- Add controls on the WebForm either from the toolbox or using C# code
- Adding controls from the toolbox auto generates html / aspx tags.
- You can set control properties and handle events in the code-behind file

PostBack with asp.net framework 1/2

- Browsers retrieve webpages from the server using the HTTP protocol
- Two of the most common HTTP methods/verbs are GET & POST
- Clicking on a hyperlink, sends a new HTTP request, usually GET
- Clicking on a button like save, sends a new HTTP request, usually POST
- When creating Asp.NET, the Microsoft Team wanted to make it simple for users to create web forms.
- So they implemented the framework to be similar to Windows Desktop Forms

PostBack with asp.net framework 2/2

- Properties & Event Handling similar to Windows Forms.
- For e.g. you can add a button and write code in the button click event handler.
- ASP.NET implements all the plumbing behind the scenes to
  - send HTTP POST request with head and body
  - The ASP.NET service installed on the server intercepts this request.
    - Execute the code inside the event handler like save data to database, generate html to display success/error message etc.
    - Send a HTTP response with the head and body, where the body contains dynamically generated HTML
- The POST request for the current url, is called a PostBack.

Asp.NET Web Form & Common Controls

- An Asp.NET Web Page is also called a Web Form
- A Web Form consists of two files
  - .aspx
    - Contains HTML tags and ASP tags
    - HTML e.g. `<div>,<input id="txtName" name="name" />`
    - ASPX e.g. `<asp:Button runat="server" id="ButttonSave"/>`
  - .aspx.cs
    - Contains the C# code behind with a class inheriting from `System.Web.UI.Page`
    - Handle events like `Page_Load`, `Button_Click`, etc.

Asp.NET Web Form & Common Controls

- WebForm1.aspx

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm1.aspx.cs"
Inherits="SampleFrameworkWebApp.WebForm1" %>

<html xmlns="http://www.w3.org/1999/xhtml">

<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
        <div>
            <asp:Button runat="server" id="ButttonSave" OnClick="ButttonSave_Click" />
        </div>
    </form>
</body>
</html>
```

Asp.NET Web Form & Common Controls

- WebForm1.aspx.cs

```
public partial class WebForm1 : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
    }

    protected void Button1_Click(object sender, EventArgs e)
    {
        TextBox1.Text = "Hello World";
    }
}
```

Asp.NET Web Form & Common Controls

- Assignment – Resume.aspx
  - Create Resume.aspx
  - Add controls for users to enter resume data
  - On Click of Save button save details into text file.
  - Save the resume to App_Data folder as follows. If candidate is Raman Gujral then create folder “RamanGujral” and inside save resume with filename = resume.txt
  - *Optional Feature: Allow users to upload their photo and save this photo along with resume as photo.?? (extension should be same as the one uploaded, e.g. jpg or png or pdf)*

Master Pages

- One attribute of a well-designed website is a consistent site-wide page layout.
- See `https://www.geeksforgeeks.org/csharp-programming-language`
- Every page has the same content at the top and bottom of the page.
- Another attribute of a well-designed site is the ease with which the site's appearance can be changed.
- A site-wide page template is created in ASP.NET using master pages.

```
<%@ Page Title="" Language="C#" MasterPageFile="~/Site1.Master" … />
```

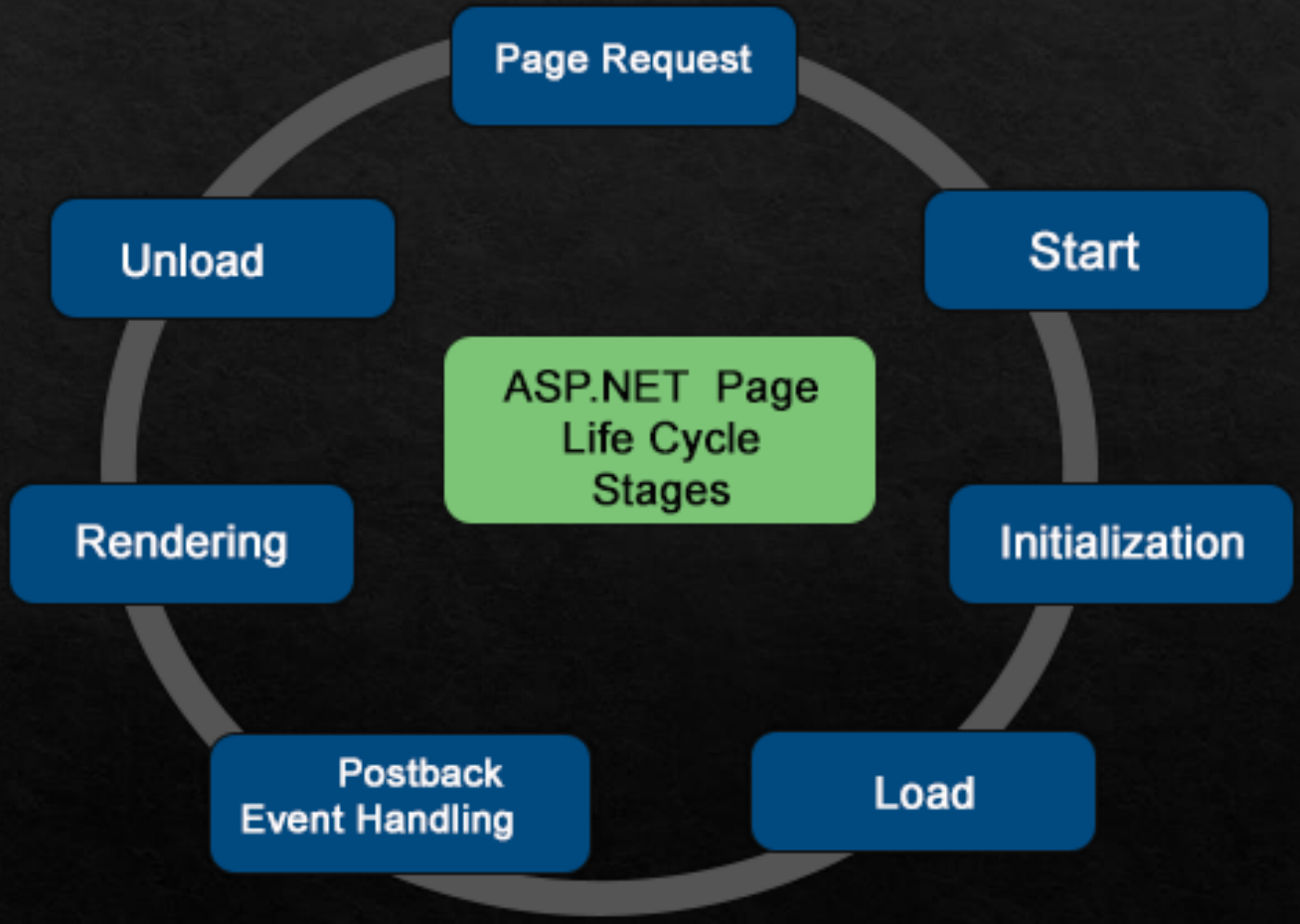
User Control

- An ASP.NET Web User Control is similar to a complete ASP.NET Web page.
- A user control can include code to manipulate its contents like a page.
- The advantage of user controls is that once you create one,
  - you can reuse it in multiple pages in the same web application.
  - you can add your own properties, events, and methods
- See sample...

Themes

- A theme decides the look and feel of the website.
- It is a collection of files that define the looks of a page.
- It can include skin files, CSS files & images.
- We define themes in a special App_Themes folder.
- Subfolders named Theme1, Halloween, etc, represent different themes
- See Sample

Asp.NET Framework Page lifecycle



IIS Express & IIS

IIS Express

- Lightweight, self-contained version of IIS optimized for developers.
- Makes it easy to use the most current version of IIS to develop and test websites
- It has all the core capabilities of IIS and additional features designed to ease website development
- Installed when you install the latest version of Visual Studio

IIS

- Internet Information Services (IIS) for Windows® Server is a flexible, secure and manageable Web server for hosting anything on the Web.
- From media streaming to web applications, IIS's scalable and open architecture is ready to handle the most demanding tasks.

Stateless vs State

Stateless

- A stateless process or application can be understood in isolation.
- There is no stored knowledge of or reference to past transactions.
- Each transaction is made as if from scratch for the first time.
- Http is a stateless protocol, therefore anything built on top of http is stateless.
- A static website is thus always stateless.

Stateful

- Stateful applications and processes, can be returned to again and again, like online banking or email.
- They're performed with the context of previous transactions
- The current transaction may be affected by what happened during previous transactions.
- You can think of stateful transactions as an ongoing periodic conversation with the same person.

State Management

- ASP.NET web applications are stateless by default
- Asp.NET State management preserves state control because.
- A new instance of the Web page class is created each time the page is posted to the server.
- If a user enters information into a web application, that information would be lost in the round trip from the browser.
- State management maintains and stores the information of any user till the end of the user session.

ViewState & Hidden Fields, Session, Application, Cookies, Cache

- By default, ASP.NET uses `ViewState` to maintain state for form fields
- E.g. Textboxes remember their last content after a round trip from the server.
- `ViewState` is a dictionary collection in the `System.Web.UI.Page` class.
- You can save custom data in the `ViewState`
- `ViewState` does not consume Server memory but is stored along with the page in a hidden field maintained by Asp.NET
- `ViewState` is scoped to the current Page
- `ViewState` can be turned off if not needed

ViewState & Hidden Fields, Session, Application, Cookies, Cache

- `Session` and `Application` are Dictionary Collection in the `System.Web.UI.Page` class
- `Session` is scoped to the current browser session while `Application` is global and accessible across all users of the Website.
- `Session` state is automatically destroyed if user does not interact with the website for some time. This is to conserve server resources.
- `Application` state is maintained till the Webserver is stopped or shut down.

ViewState & Hidden Fields, Session, Application, Cookies, Cache

- A cookie is a small amount of data that is stored either in a text file on the client file system or in-memory in the client browser session.
- It contains site-specific information that the server sends to the client along with page output.
- Cookies can be temporary (with specific expiration times and dates) or persistent.
- The cookies are saved on the client device, and when the browser requests a page, the client sends the information in the cookie along with the request information.
- The server can read the cookie and extract its value.
- A typical use is to store a token (perhaps encrypted) indicating that the user has already been authenticated in your application.

ViewState & Hidden Fields, Session, Application, Cookies, Cache

- ASP.NET supports output caching, which stores the generated output of pages, controls, and HTTP responses in memory.
- You can configure output caching declaratively in an ASP.NET Web page.
- `<%@ OutputCache Duration="15" VaryByParam="none" %>`
- You can also configure output caching globally in the Web.config file
- See walkthrough at https://docs.microsoft.com/en-us/previous-versions/aspnet/sfw2210t(v=vs.100)?redirectedfrom=MSDN

Request & Response

- Request allows ASP.NET to read the HTTP values sent by a client during a Web request.
- E.g., `Request["Id"], Request.Url, Request.RequestType`
- Response allows ASP.NET to change the HTTP-response before sending the data back to browser.
- E.g., `Response.Write, Response.Redirect, Response.Flush, Response.Clear`