Parul® University

05201302

Web Application Development

MCA MSc-IT Semester 3

Prof Faruk Abdulla

Assistant Professor PIET - MCA

Parul®
University

DIGITAL LEARNING CONTENT

Unit 1: Introduction to ASP.NET Framework

ASP.NET is an open-source, server-side web-application framework designed for web development to produce dynamic web pages, developed by Microsoft.

It allows developers to build robust web applications using .NET languages like C# or VB.NET.

Parul®
University

DIGITAL LEARNING CONTENT

Unit 1: Introduction to ASP.NET Framework

ASP.NET extends the .NET platform with tools and libraries specifically for building web apps. It uses the HTTP protocol and works on a request-response model. It simplifies the development process by offering pre-built controls, rich server-side features, and seamless integration with databases.

Parul®
University

DIGITAL LEARNING CONTENT

Unit 1: Introduction to ASP.NET Framework

ASP.NET is like a toolkit for building web pages that can respond to user input (like buttons, forms, etc.) and connect to databases, perform logic, and display results.

When you create a login form using ASP.NET, you don’t have to write the entire code from scratch. ASP.NET provides ready-to-use controls and libraries to simplify your work.

Parul®
University

DIGITAL LEARNING CONTENT

Unit 1: Introduction to ASP.NET Framework

Introduced by Microsoft in the early 2000s to simplify application development.

Emerged as a response to the complexity of COM and Win32 APIs.

First released as .NET Framework 1.0 in 2002.

Parul®
University

DIGITAL LEARNING CONTENT

Unit 1: Introduction to ASP.NET Framework

Year	Version / Name	Type	Key Features / Highlights	Trivia / Notes
2002	.NET Framework 1.0	Classic .NET	Base Class Library, CLR, ASP.NET, Windows Forms, ADO.NET	First official release of .NET Framework
2003	.NET Framework 1.1	Classic .NET	Built-in support for mobile ASP.NET apps, ODBC and Oracle database support	Shipped with Visual Studio .NET 2003
2005	.NET Framework 2.0	Classic .NET	Generics, Anonymous methods, ASP.NET 2.0, Master Pages, Data controls	Introduced System.Collections.Generic namespace
2006	.NET Framework 3.0	Classic .NET	Windows Communication Foundation (WCF), Windows Presentation Foundation (WPF), Windows Workflow Foundation (WF)	Released with Windows Vista

Parul® University

DIGITAL LEARNING CONTENT

Unit 1: Introduction to ASP.NET Framework

Year	Version / Name	Type	Key Features / Highlights	Trivia / Notes
2007	.NET Framework 3.5	Classic .NET	LINQ, ASP.NET AJAX, Extension methods, Lambda expressions	Introduced Language Integrated Query (LINQ)
2010	.NET Framework 4.0	Classic .NET	Dynamic Language Runtime (DLR), Parallel programming, MEF, Improved CLR	Shipped with Visual Studio 2010
2012	.NET Framework 4.5	Classic .NET	Async and await keywords, WebSockets, WPF improvements	Helped simplify asynchronous programming
2015	.NET Framework 4.6	Classic .NET	RyuJIT compiler, HTTP/2 support, Touch screen enhancements	Introduced 64-bit JIT compiler
2016	.NET Core 1.0	Cross-platform	Open source, cross-platform, modular CLR (CoreCLR), CLI support	First version of modern cross-platform .NET. Developed under .NET Foundation

Parul® University

DIGITAL LEARNING CONTENT

Unit 1: Introduction to ASP.NET Framework

Year	Version / Name	Type	Key Features / Highlights	Trivia / Notes
2017	.NET Core 2.0	Cross-platform	Razor Pages, Span, Performance improvements, API compatibility	Became more production-ready with increased APIs
2018	.NET Core 2.1	Cross-platform	Global tools, SignalR, HTTPS by default, Web API enhancements	LTS (Long-Term Support) release
2019	.NET Core 3.0	Cross-platform	Desktop app support (WPF, WinForms on Windows), C# 8.0 support, Blazor (client-side)	Major step towards unifying .NET
2020	.NET 5.0 (One .NET)	Unified Platform	Single platform (combines .NET Core and .NET Framework), Performance improvements, Source generators	.NET Core branding dropped — "One .NET" begins

Parul® University

DIGITAL LEARNING CONTENT

Unit 1: Introduction to ASP.NET Framework

Year	Version / Name	Type	Key Features / Highlights	Trivia / Notes
2017	.NET Core 2.0	Cross-platform	Razor Pages, Span, Performance improvements, API compatibility	Became more production-ready with increased APIs
2018	.NET Core 2.1	Cross-platform	Global tools, SignalR, HTTPS by default, Web API enhancements	LTS (Long-Term Support) release
2019	.NET Core 3.0	Cross-platform	Desktop app support (WPF, WinForms on Windows), C# 8.0 support, Blazor (client-side)	Major step towards unifying .NET
2020	.NET 5.0 (One .NET)	Unified Platform	Single platform (combines .NET Core and .NET Framework), Performance improvements, Source generators	.NET Core branding dropped — "One .NET" begins

Parul® University

DIGITAL LEARNING CONTENT

Unit 1: Introduction to ASP.NET Framework

Year	Version / Name	Type	Key Features / Highlights	Trivia / Notes
2021	.NET 6.0	Unified Platform	MAUI preview (Multi-platform App UI), Hot Reload, Minimal APIs, LTS release	Strong focus on developer productivity
2022	.NET 7.0	Unified Platform	Cloud-native support, Performance-focused, Updated APIs	Continuation of yearly releases
2023	.NET 8.0	Unified Platform	Enhanced Blazor support, Native AOT, JSON source generators	Released at .NET Conf 2023
2024	.NET 9.0 (Preview/Planning)	Unified Platform	More AI integration, deeper cloud and microservice improvements	Expected to release in Nov 2024

Parul®
University

DIGITAL LEARNING CONTENT

Unit 1: Introduction to ASP.NET Framework

- .NET was originally named **NGWS** (Next Generation Windows Services) before its release.
- The name **".NET"** was initially intended to reflect **connectivity and the internet**, aiming to bridge desktop and web technologies.
- .NET Core was **open-sourced** in 2014 and hosted on **GitHub**.
- Microsoft collaborates with **community and Linux developers** for cross-platform improvements.
- **Xamarin** was acquired by Microsoft to add native mobile support into .NET (now part of .NET MAUI).
- C# and ASP.NET have evolved side-by-side — ASP.NET Core is now based on **modern C# features**.

Parul®
University

DIGITAL LEARNING CONTENT

Key Features:

- Event-driven programming model
- Built-in state management
- Rich toolbox and designer support
- Security features like authentication and authorization
- Cross-platform development with .NET Core

Parul®
University

DIGITAL LEARNING CONTENT

Key Features:

- You can build dynamic websites (like Gmail, Amazon) that change based on user actions.
- It provides built-in security (like login systems).
- It supports event-driven programming (you write code to respond to clicks, typing, etc.).

  - When you create a login form using ASP.NET, you don’t have to write the entire code from scratch. ASP.NET provides ready-to-use controls and libraries to simplify your work.

Parul®
University

DIGITAL LEARNING CONTENT

The Origin of .NET Technology

•.NET was introduced by Microsoft to overcome the limitations of COM and Win32 programming models. It provides a managed environment for software execution.

Microsoft introduced **.NET Framework** in the early 2000s to solve many problems with earlier systems (like Windows APIs and COM).

Parul®
University

DIGITAL LEARNING CONTENT

The Origin of .NET Technology

- **Before .NET:**
- Developers had to manually manage memory.
- Code written in one language could not easily talk to code written in another.
- Application deployment was messy.
- **With .NET:**
- Everything runs in a **managed environment** (called CLR).
- Code can be written in any .NET language (C#, VB.NET, F#).
- It supports web apps, desktop apps, and even mobile.

Parul®
University

DIGITAL LEARNING CONTENT

Why .NET was developed:

.NET was developed by Microsoft to provide a **unified, secure, and scalable framework** for building modern applications. It enables **language interoperability**, simplifies development and deployment, supports **web and enterprise-grade apps**, and ensures better **security and memory management** through the Common Language Runtime (CLR).

- Platform independence (through CLR)
- Language interoperability
- Easy deployment and versioning
- Enhanced security
- Better performance via JIT

Parul®
University

DIGITAL LEARNING CONTENT

Common Language Runtime (CLR)

•CLR is the execution engine of the .NET Framework. It handles memory management, security, exception handling, and more.

CLR is like a teacher supervising students. It ensures students follow the rules, manages discipline (exceptions), gives space (memory), and throws away unused papers (garbage collection).

Responsibilities of CLR:

- Code execution
- Memory management
- Thread management
- Security enforcement
- Exception handling
- Garbage collection

.NET Framework Version 2.0 2005 (Basic Architecture of .Net Framework)	VERSION 3.0 2006	VERSION 3.5 2007	VERSION 4.0 2010	VERSION 4.5 2012
CLR(Common Language Runtime)	WF (Windows Workflow Foundation)	Entity Framework	Task - Based Asynchronous Model	.NET API For Store / UWP Apps
FCL(Framework Class Library)	Card Space		Task Parallel Library	
ADO.NET	WCF (Windows Communication Foundation)	LINQ (Language Integrated Query)	Parallel LINQ	
ASP.NET	WPF (Windows Presentation Foundation)			
WinForms				

Can Be Categorized As Types Of Applications That Can Built On .Net Framework

Parul®
University

DIGITAL LEARNING CONTENT

Managed and Unmanaged Code

Managed Code:

Code that runs under the control of CLR. It benefits from features like garbage collection, type safety, and exception handling.

Unmanaged Code:

Code that executes directly by the operating system outside the control of the CLR. Examples: C/C++ programs, Win32 APIs.

Parul®
University

DIGITAL LEARNING CONTENT

Managed and Unmanaged Code

Example:

- ✓ C# code compiled to run inside .NET = Managed
- ✓ C++ program using malloc() and free() = Unmanaged

Why it matters?
Managed code is safer and less error-prone for new developers.

Parul®
University

DIGITAL LEARNING CONTENT

Common Type System (CTS)

CTS defines how types are declared, used, and managed in the runtime. It ensures that data types are common across all .NET languages.

Purpose:

- Supports cross-language
- integration Ensures type safety
- Defines base types (int, string, float, etc.)

Parul®
University

DIGITAL LEARNING CONTENT

Common Language Specification (CLS)

CLS is a set of basic language features needed by many applications. It defines rules to which .NET languages must conform.

Importance

- Enables code reuse across .NET languages
- Ensures interoperability
- Only CLS-compliant features should be used in public APIs

Parul®
University

DIGITAL LEARNING CONTENT

Common Language Specification (CLS)

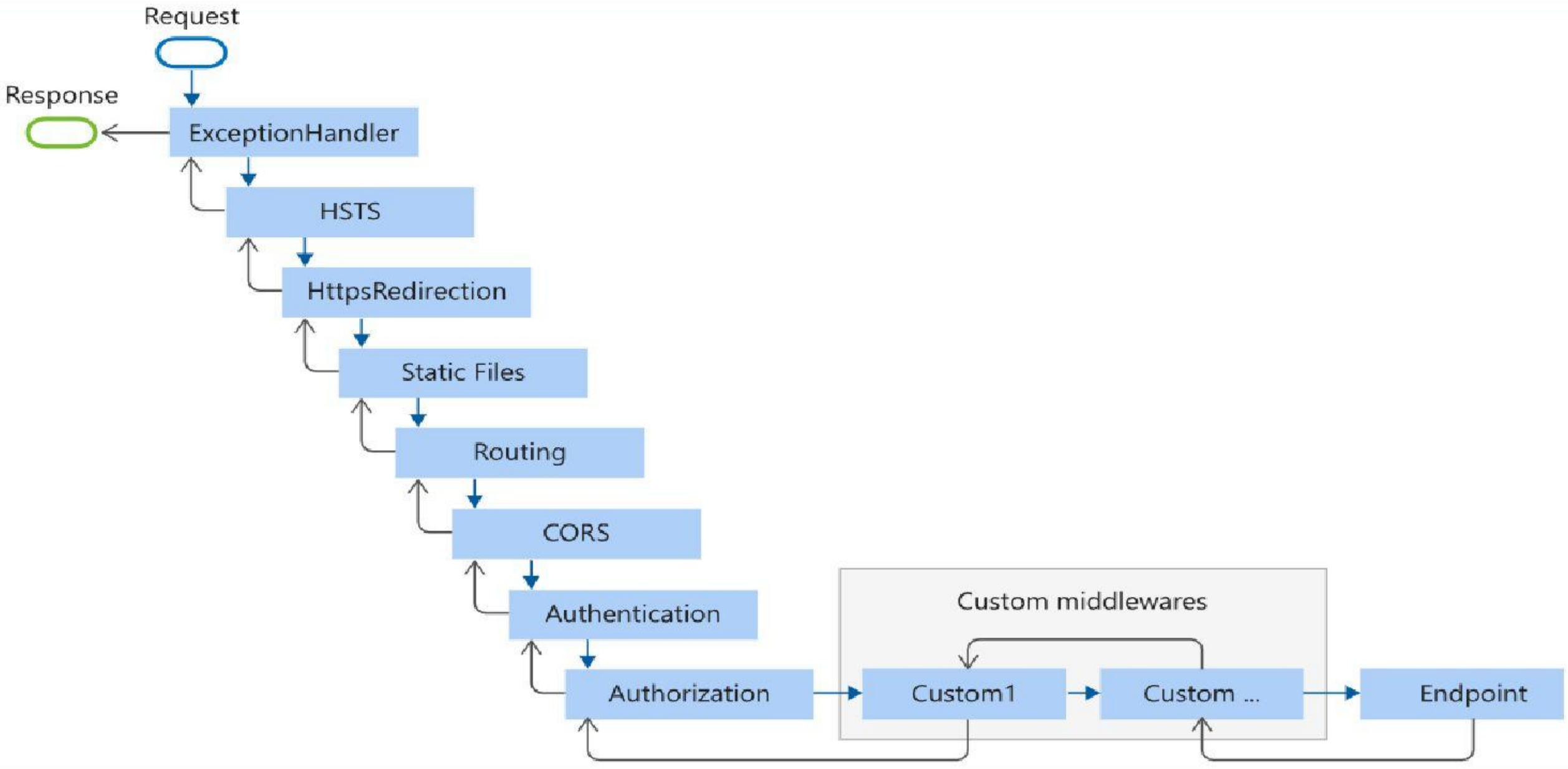
CLS is a set of rules that .NET languages must follow to ensure **cross-language compatibility**.

If you create a class in C#, it can be used in VB.NET as long as both follow CLS rules.

Parul®
University

DIGITAL LEARNING CONTENT

ASP.NET Request Processing Pipeline



Parul®
University

DIGITAL LEARNING CONTENT

Microsoft Intermediate Language (MSIL)

MSIL is a CPU-independent set of instructions that can be converted to native code. All .NET code is compiled to MSIL first.

MSIL is the intermediate code generated by the .NET compiler from source code.

Parul®
University

Microsoft Intermediate Language (MSIL)

MSIL Features:

- Platform-agnostic
- Contains metadata
- Used for language interoperability
- Later compiled to native code by JIT

Think of MSIL like Java's bytecode – it’s not machine code yet, but it's a standardized format ready to be converted.

Parul®
University

DIGITAL LEARNING CONTENT

Just-In-Time (JIT) Compiler

JIT compiles the MSIL code to native machine code at runtime.

Types of JIT Compilation:

- **Pre-JIT:** Entire code is compiled at deployment time
- **Econo-JIT:** Only required methods are compiled
- **Normal JIT:** Compiles methods as they are called

Reduces memory usage, compiles only necessary code, faster execution.

JIT compiles methods only when they are called, improving startup performance and reducing memory footprint.

Parul®
University

Framework Base Classes

These are reusable types provided by the .NET Framework Class Library (FCL). They include collections, file I/O, threading, XML, etc.

Namespaces Examples:

- System
- System.Collections
- System.IO
- System.Net
- System.Web

Parul®
University

Framework Base Classes

FCL (Framework Class Library) is a collection of **ready-made classes** and **functions** provided by .NET.

File handling (System.IO)
Collections (System.Collections)
Networking (System.Net)
Web forms (System.Web)

Benefit: You don’t have to write everything from scratch.

Parul®
University

DIGITAL LEARNING CONTENT

Assemblies – Private and Shared

An **Assembly** is a **compiled file** (.dll or .exe) that contains your code + metadata.

A compiled code library used for deployment, versioning, and security in .NET applications. Contains metadata, MSIL, and resources.

Private Assemblies: Used by a single application

Shared Assemblies: Stored in Global Assembly Cache (GAC) and shared by multiple apps

Parul®
University

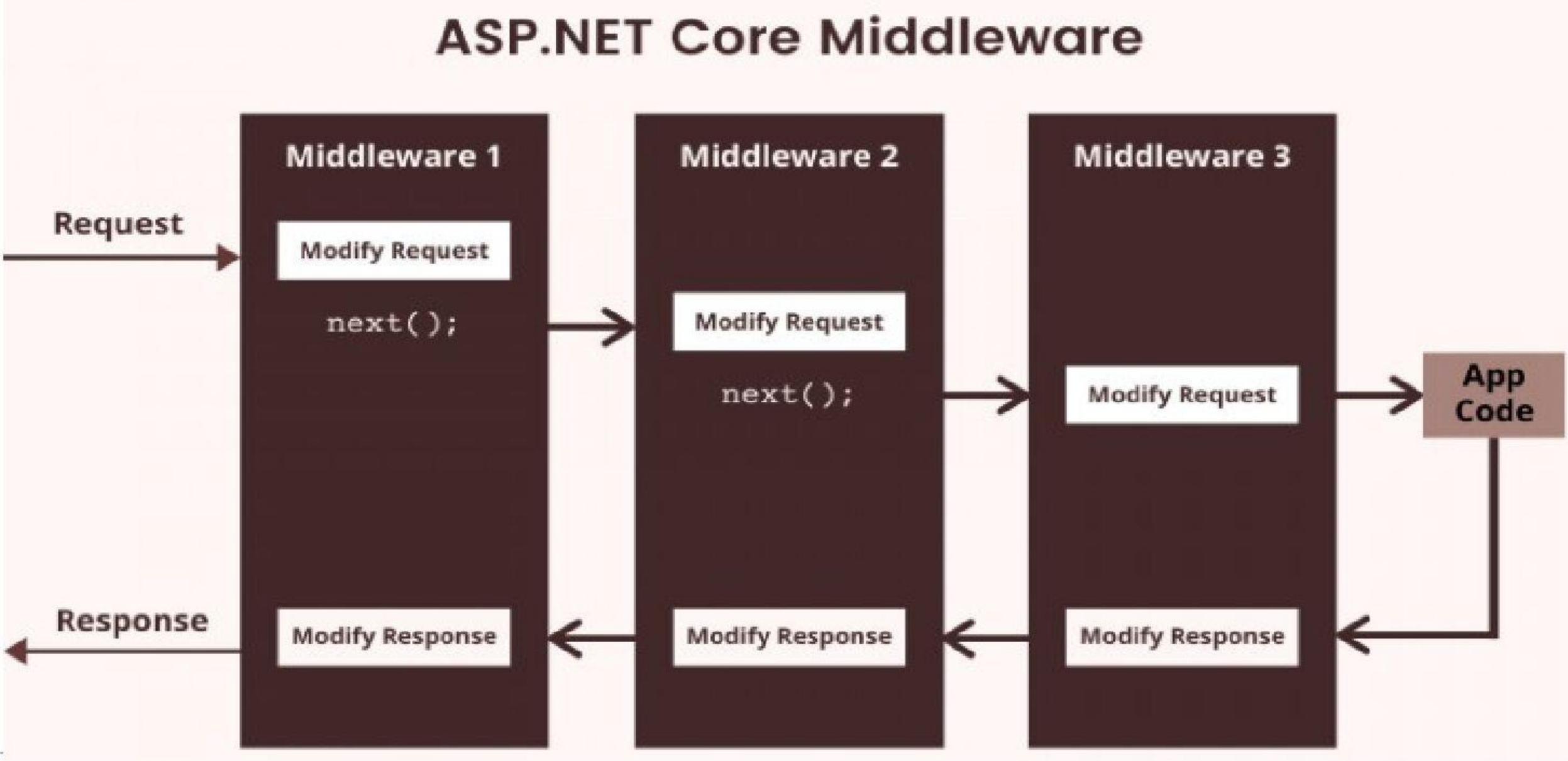
Assemblies – Components of an Assembly:

- Manifest
- Metadata
- MSIL
- Resources

Parul®
University

DIGITAL LEARNING CONTENT

ASP.NET Core Middleware Pipeline



Parul®
University

DIGITAL LEARNING CONTENT

Garbage Collection

- Automatic memory management feature of CLR. It reclaims memory occupied by unreachable objects.

- Objects are stored in managed heap
- GC identifies unused objects
- Frees memory and compacts the heap
- Operates in generations (0, 1, 2)

Benefits of Garbage Collection

.NET uses **automatic garbage collection**. You don't need to manually delete unused objects.

- No manual memory management
- Avoids memory leaks
- Efficient application performance

Parul®
University

DIGITAL LEARNING CONTENT

END of Unit 1

THANK YOU