



**CCITFTUI**  
FAKULTAS TEKNIK



# **Practical Approaches with Next.js & Tailwind CSS:** Exploring Modern Web Development Trends Alongside an Academic Overview of ASP.NET

FINAL PROJECT | INTEGRATED SYSTEM

# Abstract

## ABSTRACT

Name : Riki Awal Syahputra  
No registration : 2120010136  
Title : Practical Approaches with Next.js & Tailwind CSS: Exploring  
Modern Web Development Trends Alongside an Academic Overview  
of ASP.NET  
Advisor : Listyo Edi Prabowo, S.T.,M.T.

*Rapid progress and development in the business world has driven increased scalability and accelerated operations to support economic growth, both individually, in groups and at the national level. In this context, the use of technology, especially in the form of digitization, has proven itself as an effective approach to achieving this goal. Digital transformation, which is primarily reflected through online platforms, continues to evolve to achieve flexibility in development, reduced time and costs, and attractive results for consumers.*

*Especially in the development of online platforms, the NextJS framework based on JavaScript and the Tailwind CSS design approach have become highly recommended solutions. NextJS enables high-efficiency web application development through support for server-side rendering (SSR) and static site generation (SSG). Meanwhile, Tailwind CSS, as a class-based layout framework, enables fast responsive interface design through the use of utility classes.*

*It is important to note that on a larger scale, such as large enterprises, web development frameworks such as ASP.NET have a significant role. ASP.NET stands out in the development of large-scale applications with impressive processing speeds. In addition, its advanced features support server-based development and solid integration with the Microsoft services ecosystem.*

*However, in choosing a technology, it is necessary to consider the business context, the complexity of the project, as well as the development of the latest technology trends. Successful web development requires an in-depth understanding of the principles of design, security, performance, and compliance with a company's long-term goals.*

Keywords:  
*Digitalization, scalability, web development*

Rapid progress & development of businesses increased **scalability** that push **digitalization** and put their trust in **web development** for the more advanced economy

## TABLE OF CONTENTS

ORIGINALITY DECLARATION PAGE .....	ii
ENDORSEMENT PAGE .....	iii
ACKNOWLEDGMENTS .....	iv
ABSTRACT .....	iv
ABSTRACT .....	vi
TABLE OF CONTENTS .....	vii
LIST OF FIGURES .....	ix
<b>1. INTRODUCTION .....</b>	<b>1</b>
1.1 Background .....	1
1.2 Problem Domain .....	2
1.3 Writing Objective .....	3
1.4 Writing Methodology .....	4
1.5 Place and Time of Activity .....	6
1.6 Writing Framework .....	6
<b>2 LITERATURE REVIEW .....</b>	<b>8</b>
2.1 Historical Overview of Company .....	8
2.2 Purpose, Vision, and Mission .....	9
<b>3 RESULT AND PROBLEM ANALYSIS .....</b>	<b>10</b>
3.1 Exploring NextJS .....	10
3.1.1 What is NextJS .....	10
3.1.2 Implement NextJS Project with TODO LIST .....	17
3.1.3 Used Software .....	30
3.1.4 Project Result .....	30
3.2 Exploring ASP.NET .....	35
3.2.1 What is ASP.NET .....	36
3.2.2 ASP.NET Abstraction Programming Technique Overview .....	42
<b>4 CONCLUSION &amp; SUGGESTION .....</b>	<b>57</b>
4.1 Conclusions .....	57
4.2 Suggestions .....	58
<b>REFERENCES .....</b>	<b>60</b>

# 01.

## Introduction & Literature review

(About the Topic & Internship company)

# 02.

## Result & Problem Analysis

Explore & Implement NextJS |  
Overviewing ASP.NET

# 03.

## Conclusion & Suggestion

Wrap up the paper



# 01.

## Introduction & Literature Review



This paper can be done with the help of data & experience by PT. IDX Consulting & VIX program

## **Background (company):**

**IDX Partners** is providing data analytics consulting and solutions through data-driven strategies, risk assessment, decision management, and business intelligence integration

## Problem Domain

1. NextJS and ASP.NET exploration and implementation
2. Encompassing understanding of their concepts
3. Advantages, and disadvantages
4. Identifying adopters
5. Designing projects
6. Analyzing software and technologies
7. Evaluating project outcomes
8. Delving into advanced cases
9. Exploring ASP.NET abstraction programming techniques
10. Drawing conclusions and suggesting future improvements for both technologies.

## Writing Objective

1. Explore the impact of digitalization on scalability and business operations
2. Assess NextJS and Tailwind CSS frameworks
3. Evaluate ASP.NET's role
4. Examine technology selection factors
5. Understand web development prerequisites
6. Fulfill a graduation requirement.

# Historical Overview of Company

Founded in 2002, id/x Partners is led by experts in banking and management, offering data-driven consulting across diverse industries like finance, telecom, and retail, seamlessly integrating data analytics, decisioning solutions, and risk management for innovation and growth.

## VISION

provides consulting services that specializes in utilizing data analytic and decisioning (DAD) solutions combined with an integrated risk management and marketing discipline to help clients optimize the portfolio profitability and business process.

## MISSION

- C: Customer** (Deliver right solution and superior service for the customer to realize maximum business value)
- H: Honorable** (Focus on profitable growth without sacrificing integrity & quality)
- A: Agile** (Respond quickly and meaningfully to new opportunities or threats)
- M: Mentorship** (Develop team's continuous personal and professional growth)
- P: Proactive** (Put forth new ideas in order to improve our work.)
- I: Innovative** (Cultivate a creative working environment to solve tough problems.)
- O: Ownership** (Exercise accountability and ownership of our work and responsibilities.)
- N: Numeric** (Make decisions based on facts and numbers.)





# 02.

## **Result & Problem Analysis**

# What is NextJS?

1. **Framework:** NextJS is a modern web development framework.
2. **SSR and SSG:** Speeds up loading and user experience with server-side rendering (SSR) and static site generation (SSG).
3. **SEO Boost:** using SSR by pre-rendering pages.
4. **Balanced Interaction:** NextJS balances dynamic interactions and fast content delivery.
5. **Easy Integration:** Intuitive routing and smart data fetching help integrate projects with data sources.
6. **Language Support:** Supports TypeScript and Babel.
7. **User Focus:** combining speed, interactivity, and SEO.
8. **Key Web Tools:** optimizing performance and user engagement.

# Advantage & Disadvantage

## Advantages:

1. Enhanced Performance: SSR and SSG boost speed and UX.
2. SEO Friendly: Pre-rendering improves SEO.
3. Intuitive Routing: Streamlined navigation enhances engagement.
4. Data Flexibility: Diverse data fetching options.
5. Developer-Friendly: Supports TypeScript, Babel, HMR.
6. Hybrid Approach: Combines SSR and CSR effectively.

## Disadvantages:

1. Learning Curve: SSR, routing concepts need learning.
2. Build Complexity: SSR, SSG can complicate builds.
3. Server Load: SSR strains servers for complex apps.
4. Dependency Management: Careful integration of third-party libs.

**NEXT.js**

*Enabling Performance and Scalability*



**Tailwind CSS**

*Streamlined Interface Design*

## The Company who Implemented NextJS

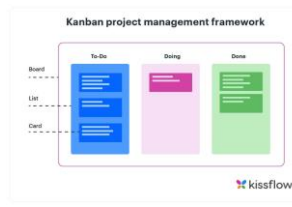


Typeform

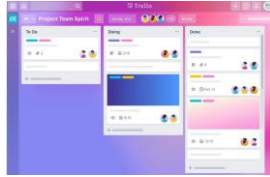
*ticketmaster*

**Implement NextJS with Todo List App**

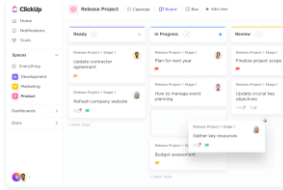
Kanban



Trello

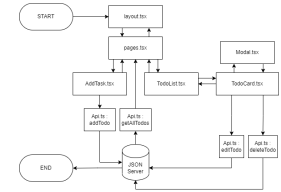


ClickUp



Task Management, Subtasks, Due Dates, Assignees, Comments, Attachments, Checklists, Labels, Custom Fields, Calendar View, Kanban Boards, List View, Goals and OKRs, Document Management, Time Tracking, Gantt Charts (ClickUp), Power-Ups (Trello), Automation, Integrations, Basic Reporting, Mobile Apps, Workload Management (ClickUp), Collaboration Features, Templates, Card Voting (Trello with Power-Ups), Time Estimates (ClickUp), Advanced Reporting (ClickUp), Board and List Customization.

- Input task functionality
- Todo list card
- Edit functionality
- Delete functionality



Research Reference



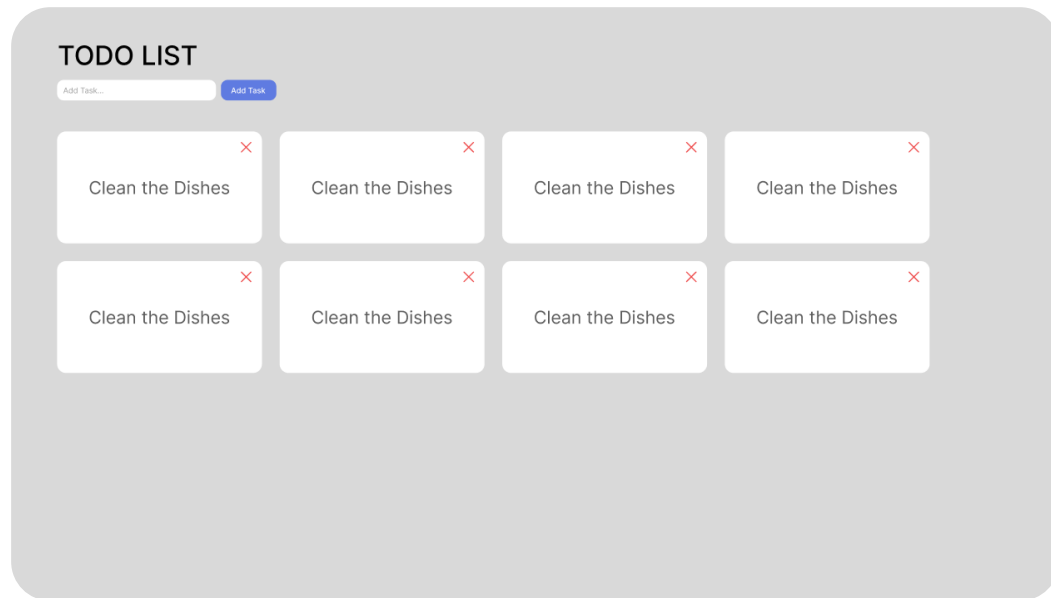
Functionality Feature  
Extraction



Decide Functionality  
Feature

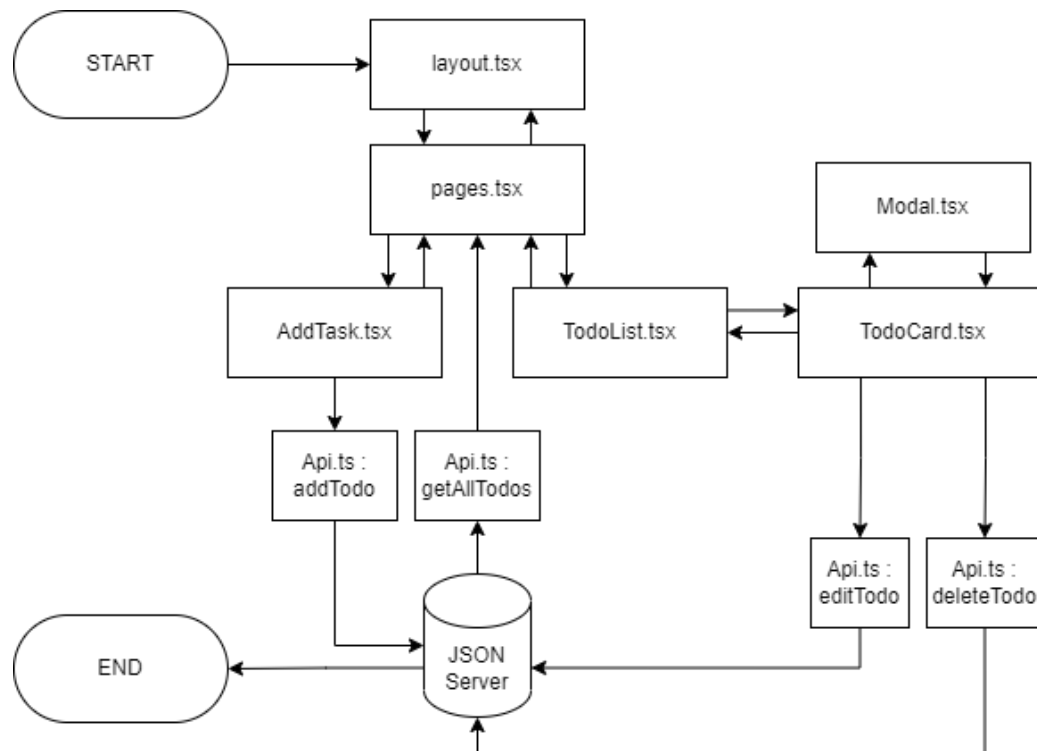
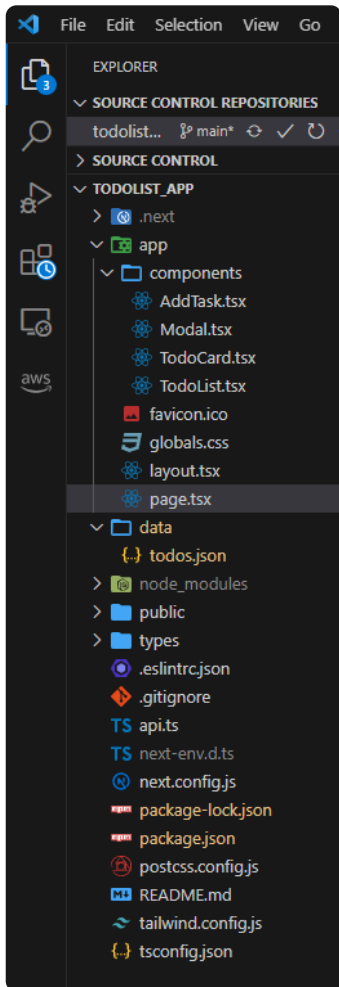


Plans & Build



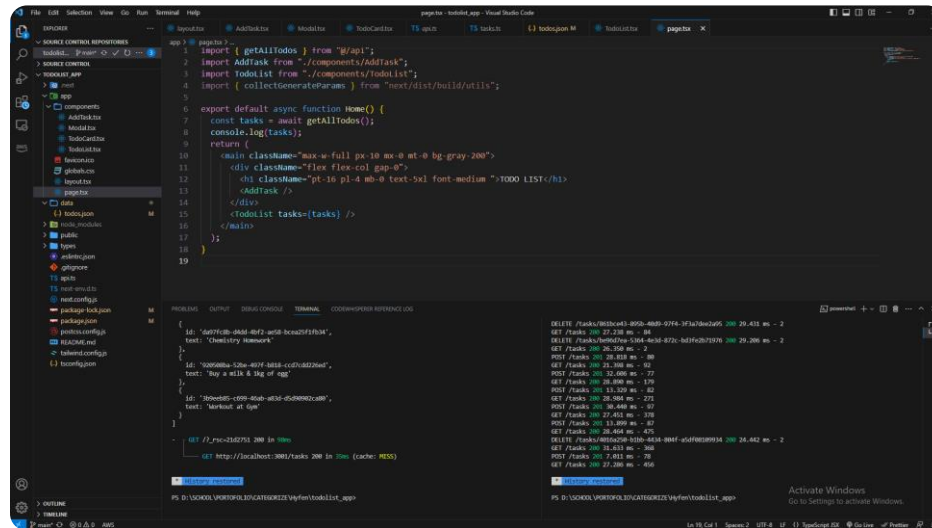
UI  
Plans

# Flowchart

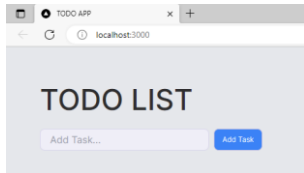
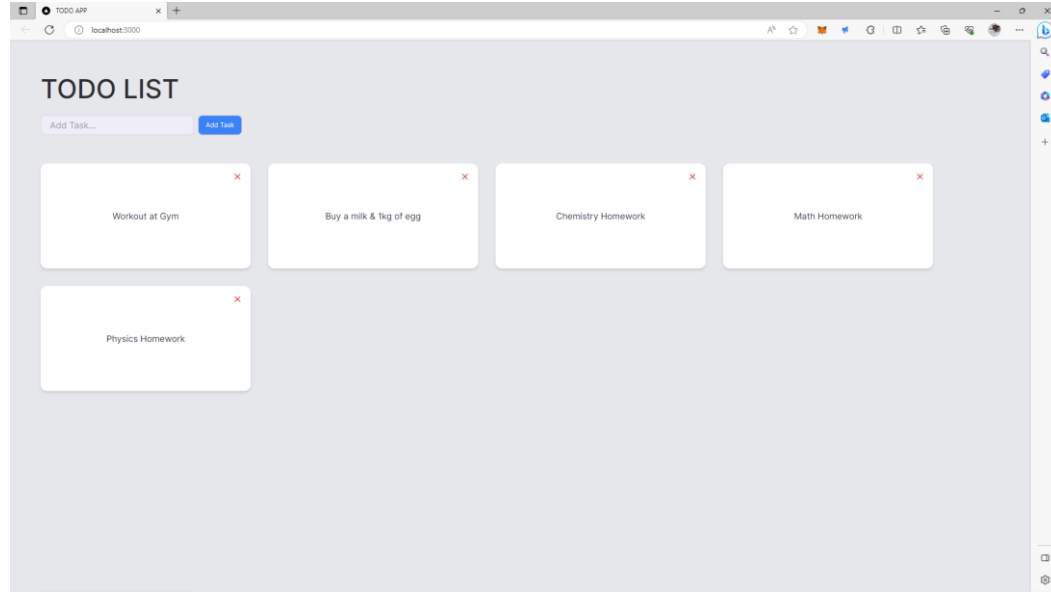




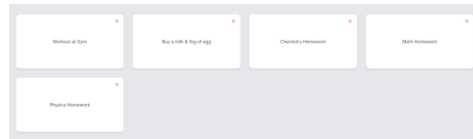
# Used software



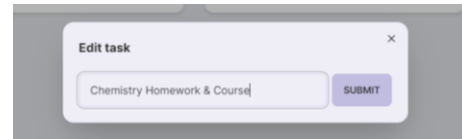
# Project result



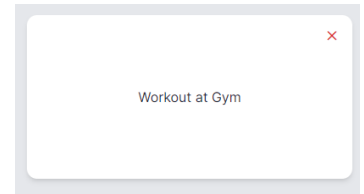
Add taskbar



Task card



Edit feature  
functionality



Delete feature  
functionality


# Advanced Case Reference



Products

Hardware

Solutions



Have questions?

Contact us

## Pricing Calculator

Setup Monthly or Yearly Pricing Plans

Search...

Avatar Name  
Detail here

### Basic

For every user

- ✓ All analytics features
- ✓ Up to 250,000 tracked
- ✓ Base Support
- ✓ Up to 3 team members

Add Plan

### Pro

For every user

- ✓ All analytics features
- ✓ Up to 250,000 tracked
- ✓ Base Support
- ✓ Up to 3 team members

Add Plan

### Enterprise

For every user

- ✓ All analytics features
- ✓ Up to 250,000 tracked
- ✓ Base Support
- ✓ Up to 3 team members

Add Plan

### Testing

For every user

- ✓ All analytics features
- ✓ Up to 250,000 tracked
- ✓ Base Support
- ✓ Up to 3 team members

Add Plan

Sales team target

82%

Achieved


Cleared Queue

1.4k

No. of Bills  +15%



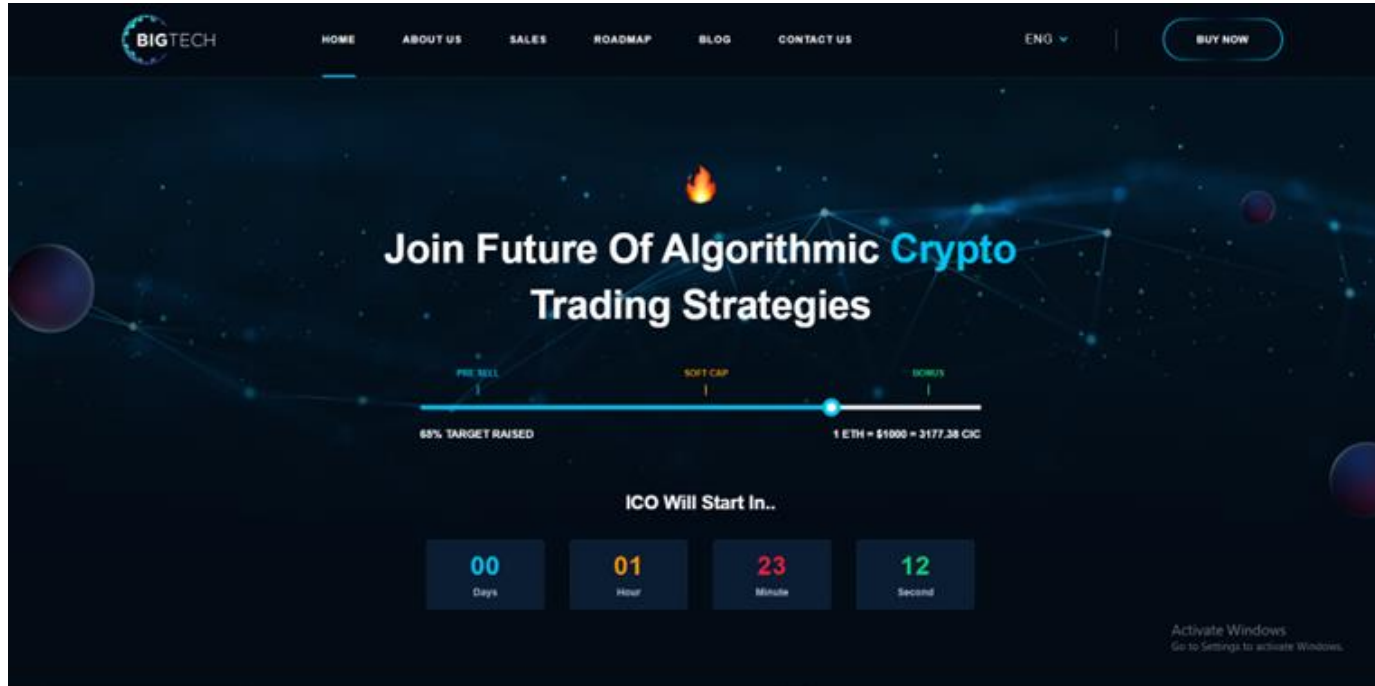
+4






Pricing Calculator

# Advanced Case Reference



ICO Page


# Advanced Case Reference

 **NEFTRON**

[Home](#) [Activity](#) [Collectors](#) [Blog](#) [Pages](#) [Contact](#)

[Connect Wallet](#)


### 3d Aesthetics with shapes




Creators  
**Leslie Alexander**


Current Bids  
**1.24 ETH**


Auction ending in:  
**13 h 53 m 10 s**

 Rewards  
**\$400k+**


 Ticket  
**Thailand Gateway**

[Place bid now](#) [View items](#)







By barry tone  
**3d aesthetics with shapes**  
0.0-74 ETH




By barry tone  
**Anonymous Genesis**  
0.0-74 ETH




By barry tone  
**GoldSilver Pirates**  
0.0-74 ETH



By barry tone  
**Shadowy Super Coder**  
0.0-74 ETH




By barry tone  
**Chameleonkin Series NFT**




#### Multilayer security

This process usually takes 4-8 hours. Upon immediately on the Marketplace as either an



#### Forming & royalties

This process usually takes 4-8 hours. Upon immediately on the Marketplace as either an



#### Multichain minting

This process usually takes 4-8 hours. Upon immediately on the Marketplace as either an

Activate Windows  
Go to Settings to activate Windows.

NFT & Crypto Marketplace

## **Exploring ASP.NET**

# What is ASP.NET?

ASP.NET is an **open-source web application development framework** that comprises two primary variants: **ASP.NET Web Forms** and **ASP.NET MVC**. It enables developers to create **powerful and dynamic web applications** by utilizing reusable controls and structured architectures that provide precise control over components.

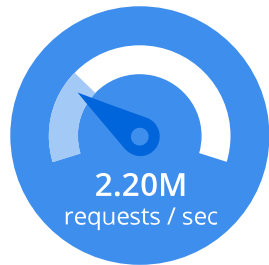
## Advantage & Disadvantage

ASP.NET Advantages:

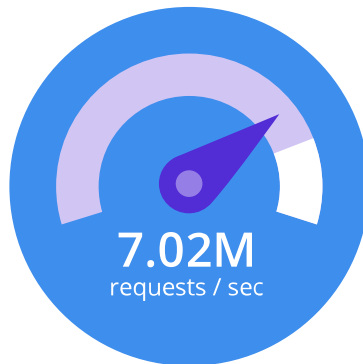
1. Integration: Seamlessly works with Microsoft tools.
2. Security: Offers strong protection features.
3. Modularity: Easy component-focused work.
4. Rich Resources: Abundance of libraries and tools.
5. Clean Code: Object-oriented for readability.

ASP.NET Disadvantages:

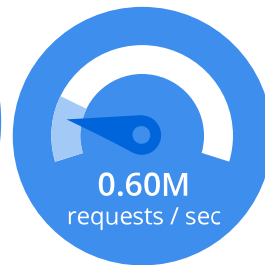
1. Learning Curve: Steeper learning process.
2. Proprietary Links: Tied to Microsoft, limits options.
3. Variable Performance: Windows-optimized.
4. Resource Use: Heavier than alternatives.



Java Servlet



.NET



Node.js



# The Company who Implemented ASP.NET



# Vast Implementation of ASP.NET

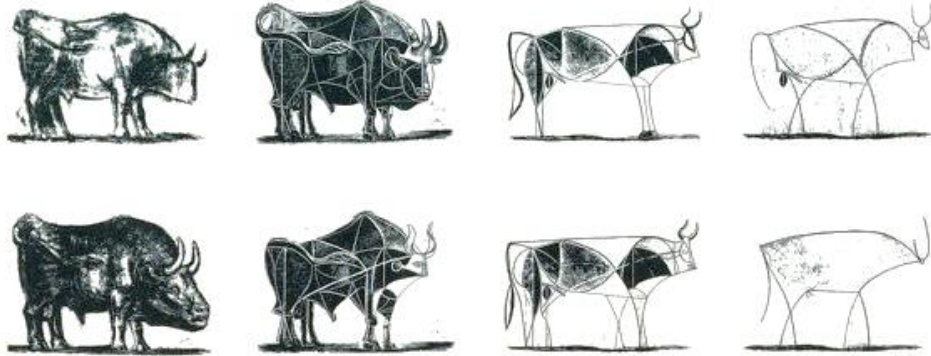
ASP.NET is like a Swiss Army knife for making all kinds of computer stuff. It's used in many areas:

1. **Business Management:** Helps companies run smoothly.
2. **Online Stores:** Builds advanced e-commerce.
3. **Healthcare:** Improves health systems.
4. **Financial Apps:** Creates secure money tools.
5. **Online Education:** Enables web-based learning.
6. **Robotics:** Works with robots and machines.
7. **Entertainment:** Develops games and online fun.
8. **Smart Devices:** Connects gadgets and devices.
9. **Logistics:** Manages shipping and inventory.
10. **Government Services:** Creates official websites and services.
11. **Aerospace & Defense:** Builds tools for aviation and military.
12. **Energy Control:** Manages energy and utilities.
13. **Telecom:** Assists phone and communication companies.
14. **Car Technology:** Develops automotive apps and tools.
15. **Environmental Monitoring:** Tracks environmental data for nature.

ASP.NET is strong and flexible, so it's great for lots of different things businesses and tech need.

# **ASP.NET Abstraction Programming Technique Overview**

# ABSTRACTION



**Simplifies complex software** by focusing on the important stuff and **hiding the rest**. giving a clear view without showing all the inner workings. In software, abstraction makes **things easier to build** by breaking them **into simpler parts**. This way, developers can concentrate on the big picture rather than tiny details.

**Main goal:**

- separate what something does from how it's done
- smoother development
- code reuse.

delegates | events | generics | extension methods | lambda  
expressions | LINQ | asynchronous programming

```
using System;

namespace DelegateExample
{
    delegate int MathOperation(int a, int b);

    class Calculator
    {
        public int Add(int x, int y) { return x + y; }
        public int Subtract(int x, int y) { return x - y; }
    }

    class Program
    {
        static void Main(string[] args)
        {
            Calculator calculator = new Calculator();

            MathOperation addDelegate = calculator.Add;
            MathOperation subtractDelegate = calculator.Subtract;

            int resultAdd = addDelegate(5, 3);           // Output: 8
            int resultSubtract = subtractDelegate(7, 2); // Output: 5

            Console.WriteLine("Result of addition: " + resultAdd);
            Console.WriteLine("Result of subtraction: " + resultSubtract);
        }
    }
}
```

reference to a method, allowing developers to invoke methods indirectly through the delegate instance

# Events

```
using System;

namespace EventExample
{
    class EventPublisher
    {
        public event EventHandler MyEvent;

        public void RaiseEvent()
        {
            MyEvent?.Invoke(this, EventArgs.Empty);
        }
    }

    class EventSubscriber
    {
        public void HandleEvent(object sender, EventArgs e)
        {
            Console.WriteLine("Event handled!");
        }
    }

    class Program
    {
        static void Main(string[] args)
        {
            EventPublisher publisher = new EventPublisher();
            EventSubscriber subscriber = new EventSubscriber();

            publisher.MyEvent += subscriber.HandleEvent;

            publisher.RaiseEvent(); // Output: Event handled!
        }
    }
}
```

a mechanism that enables an object to notify other objects when a specific action or state change occurs



# Generics

```
using System;

namespace GenericsExample
{
    class GenericList<T>
    {
        private T[] items = new T[10];
        private int count = 0;

        public void Add(T item)
        {
            if (count < items.Length)
            {
                items[count] = item;
                count++;
            }
        }

        public void DisplayItems()
        {
            foreach (T item in items)
            {
                if (item != null)
                {
                    Console.WriteLine(item);
                }
            }
        }
    }

    class Program
    {
        static void Main(string[] args)
        {
            GenericList<int> intList = new GenericList<int>();
            intList.Add(5);
            intList.Add(10);
            intList.DisplayItems(); // Output: 5, 10

            GenericList<string> stringList = new GenericList<string>();
            stringList.Add("Hello");
            stringList.Add("World");
            stringList.DisplayItems(); // Output: Hello, World
        }
    }
}
```

enable the creation of components that can work with a wide range of data types while maintaining strong type safety

# Extensions Method

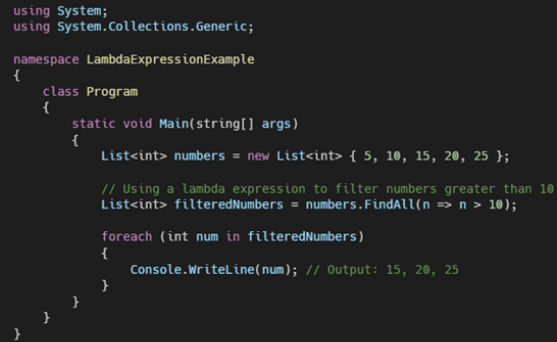
```
using System;

namespace ExtensionMethodExample
{
    public static class StringExtensions
    {
        public static string Capitalize(this string input)
        {
            if (string.IsNullOrEmpty(input))
            {
                return input;
            }
            return char.ToUpper(input[0]) + input.Substring(1);
        }
    }

    class Program
    {
        static void Main(string[] args)
        {
            string text = "hello, world!";
            string capitalizedText = text.Capitalize(); // Output: Hello, world!
            Console.WriteLine(capitalizedText);
        }
    }
}
```

enable the addition of new  
methods to types that are already  
defined

# Lambda Expressions



```
using System;
using System.Collections.Generic;

namespace LambdaExpressionExample
{
    class Program
    {
        static void Main(string[] args)
        {
            List<int> numbers = new List<int> { 5, 10, 15, 20, 25 };

            // Using a lambda expression to filter numbers greater than 10
            List<int> filteredNumbers = numbers.FindAll(n => n > 10);

            foreach (int num in filteredNumbers)
            {
                Console.WriteLine(num); // Output: 15, 20, 25
            }
        }
    }
}
```

enable developers to define small, inline functions without the need to explicitly declare a separate method or delegate

```
using System;
using System.Collections.Generic;
using System.Linq;

namespace LINQExample
{
    class Program
    {
        static void Main(string[] args)
        {
            List<int> numbers = new List<int> { 5, 10, 15, 20, 25 };

            // Using LINQ to filter numbers greater than 10
            IEnumerable<int> filteredNumbers = numbers.Where(n => n > 10);

            foreach (int num in filteredNumbers)
            {
                Console.WriteLine(num); // Output: 15, 20, 25
            }
        }
    }
}
```

enables developers to perform  
data querying and manipulation  
tasks using a unified syntax

# Asynchronous Programming

```
using System;
using System.Net.Http;
using System.Threading.Tasks;

namespace AsynchronousExample
{
    class Program
    {
        static async Task Main(string[] args)
        {
            await PerformAsyncOperation();
            Console.WriteLine("Async operation completed.");
        }

        static async Task PerformAsyncOperation()
        {
            using (HttpClient client = new HttpClient())
            {
                HttpResponseMessage response = await
client.GetAsync("https://www.example.com");
                string content = await response.Content.ReadAsStringAsync();
                Console.WriteLine("Content length: " + content.Length);
            }
        }
    }
}
```

allowing tasks to be executed  
concurrently

# THANKS!

