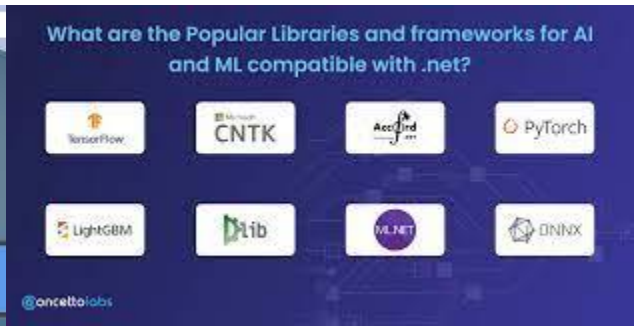
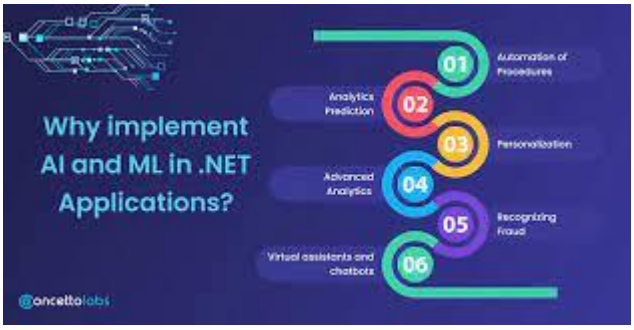
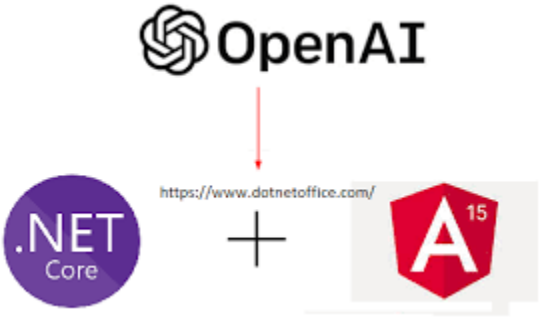
Visual Studio .NET Core App Development with ASP.Net Web API With AI

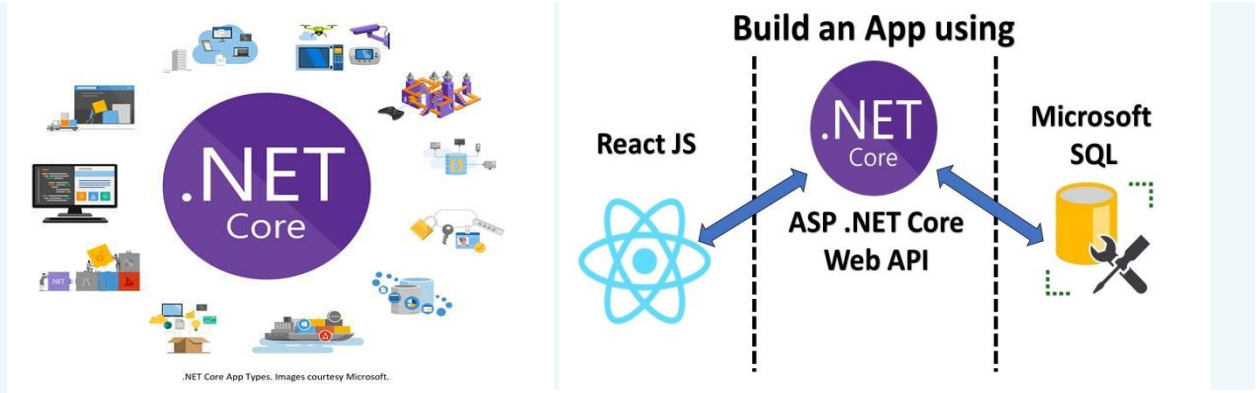
Cloud Azure, SQL Server, WPF Win Form and SAP Crystal Report

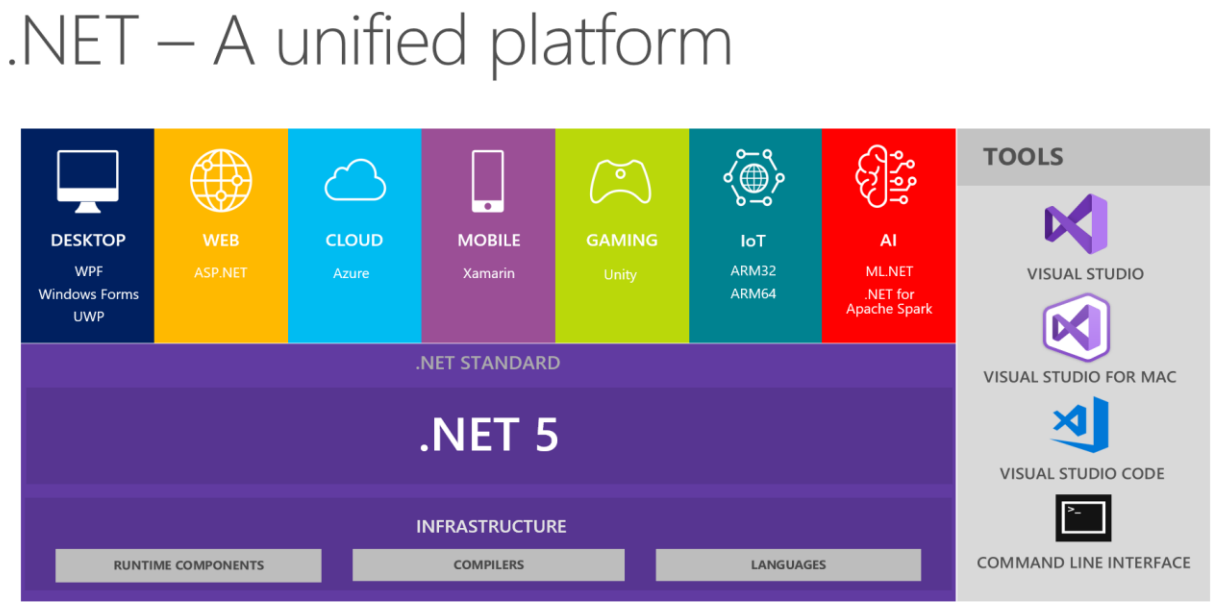
WHAT ARE THE REASONS TO LEARN DOT NET?











- ASP.NET Core is the new version of the ASP.NET web framework mainly targeted to run on .NET Core platform.
- ASP.NET Core is a free, open-source, and cross-platform framework for building cloud-based applications, **such as web apps, IoT apps, and mobile backends**. It is designed to run on the cloud as well as on-premises.
- Same as .NET Core, it was architected modular with minimum overhead, and then other more advanced features can be added as NuGet packages as per application requirement. This results in high performance, require less memory, less deployment size, and easy to maintain.
- ASP.NET Core is an open source framework supported by Microsoft and the community, so you can also contribute or download the source code from the ASP.NET Core Repository on Github.
- ASP.NET 3.x runs only on .NET Core 3.x, whereas ASP.NET Core 2.x runs on .NET Core 2.x as well as .NET Framework.

Why ASP.NET Core?

- **Supports Multiple Platforms:** ASP.NET Core applications can run on Windows, Linux, and Mac. So you dont need to build different apps for different platforms using different frameworks.
- **Fast:** ASP.NET Core no longer depends on System.Web.dll for browser-server communication. ASP.NET Core allows us to include packages that we need for our application. This reduces the request pipeline and improves performance and scalability.
- **IoC Container:** It includes the built-in IoC container for automatic dependency injection which makes it maintainable and testable.
- **Integration with Modern UI Frameworks:** It allows you to use and manage modern UI frameworks such as **AngularJS, ReactJS, Umber, Bootstrap**, etc. using Bower (a package manager for the web).
- **Hosting:** ASP.NET Core web application can be hosted on multiple platforms with any web server such as IIS, Apache etc. It is not dependent only on IIS as a standard .NET Framework.
- **Code Sharing:** It allows you to build a class library that can be used with other .NET frameworks such as .NET Framework 4.x or Mono. Thus a single code base can be shared across frameworks.

Who Should Attend

- Fresh Graduates looking for their first job
- Beginners who want to acquire .NET, .NET Core, MVC and React JS scripting skills
- Advanced .NET users, who want to take their skills to the next level
- Web Developers
- Mobile Developers

C#.net

The .NET Framework is an integral Windows component that supports building and running the next generation applications. Microsoft's C# .NET has matured along with the entire Visual Studio .NET development environment. Students will also be introduced to Integrated Development Environments (IDE)

Microsoft SQL Server

- Introduction to Microsoft SQL Server 2014
- Using Transact-SQL on a SQL Server Database
- Designing a SQL Server Database
- Implementing SQL Server Databases and Tables
- Implementing Data Integrity

- Accessing and Modifying Data
- Managing and Manipulating Data
- Implementing Stored Procedures
- Implementing Triggers
- Implementing Views
- Implementing Indexes
- Managing SQL Server Transactions and Locks
- Designing and Administering SQL Server 2014 Security
- SQL Server Monitoring and Tuning

.NET Windows Form

- Windows Forms and the User Interface
- Configuring Controls and Creating the User Interface
- Advanced Windows Forms Controls Tool Strips, Menus, and Events
- Configuring Connections and Connecting to Data
- Working with Data in a Connected Environment
- Create, Add, Delete, and Edit Data in a Disconnected Environment
- Implementing Data-Bound Controls
- Working with XML
- Advanced Topics in Windows Forms
- Enhancing Usability
- Asynchronous Programming Techniques
- MDI AND SDI FORM
- Validation Control
- Dialog Boxes (Open File Dialog for image saving app by path)
- 3rd Party Component Dev Component
- Excel File reading
- Text File Reading
- Database Image Saving
- Back Ground Process using threading
- Lamda Expression
- Window Service
- Globalization
- Socket Programming

Crystal Report

- Introduction to Crystal Reports
- Accessing Your Data
- Using the Default Report Wizards
- Fundamentals of the Crystal Reports Design Environment
- Formatting Reports
- Enhancing Crystal Reports

- Advanced Report Design Concepts
- Sharing and Distributing Crystal Reports

ASP.NET MVC Application

- Software Requirements for ASP.NET MVC
- Installing ASP.NET MVC
- Installing the MVC Development Components
- Installing MVC on a Server
- Creating an ASP.NET MVC Application
- The New ASP.NET MVC Dialog
- Application Templates
- View Engines
- Testing
- Understanding the MVC Application Structure
- ASP.NET MVC and Conventions
- Convention over Configuration
- Conventions Simplify Communication

CONTROLLERS

- The Controller's Role
- Controller Basics
- Creating the New Controller
- Writing Your Action Methods
- Parameters in Controller Actions

VIEWS

- What a View
- Strongly Typed Views
- View Models
- Adding a View
- Understanding the Add View Dialog Options
- Razor View Engine
- What is Razor?
- Code Expressions
- Html Encoding
- Code Blocks
- Razor Syntax Samples
- Implicit Code Expression
- Explicit Code Expression
- Un encoded Code Expression
- Code Block
- Combining Text and Markup
- Mixing Code and Plain Text

- Escaping the Code Delimiter
- Server Side Comment
- Calling a Generic Method
- Layouts
- Specifying a Partial View
- The View Engine

MODELS

- Empty Controller
- Controller with Empty Read/Write Actions
- Controller with Read/Write Actions and Views,
- Using Entity Framework
- Code First Conventions
- The DbContext
- The Data Context
- The Views
- Creating Databases with the Entity Framework
- Using Database Initializers
- Model Binding

FORMS AND HTML HELPERS

- Using Forms
- The Action and the Method
- To GET or To POST
- HTML Helpers
- Automatic Encoding
- Make Helpers Do Your Bidding
- Inside HTML Helpers
- Html.BeginForm
- Html.Validation
- Adding Inputs
- Html.TextBox (and Html.TextArea)
- Html.Label
- Html.DropDownList (and Html.ListBox)
- Html.ValidationMessage
- Helpers, Models, and View Data
- Strongly-Typed Helpers
- Helpers and Model Metadata
- Templated Helpers
- Helpers and ModelState
- Other Input Helpers
- Html.Hidden
- Html.Password
- Html.RadioButton
- Html.CheckBox
- Rendering Helpers

- Html.ActionLink and Html.RouteLink
- URL Helpers
- Html.Partial and Html.RenderPartial
- Html.Action and Html.RenderAction
- Passing Values to RenderAction
- Cooperating with the ActionName Attribute
-

DATA ANNOTATIONS AND VALIDATION

- Annotating Orders for Validation
- Using Validation Annotations
- Required
- StringLength
- RegularExpression
- Range
- Validation Attributes from System.Web.Mvc
- Custom Error Messages and Localization
- Looking Behind the Annotation Curtain
- Validation and Model Binding
- Validation and Model State
- Controller Actions and Validation Errors

SECURING YOUR APPLICATION

- Using the Authorize Attribute to Require Login
- Securing Controller Actions
- How the AuthorizeAttribute Works with Forms Authentication and the
- AccountController
- Windows Authentication in the Intranet Application Template
- Securing Entire Controllers

ASP.NET CORE

- ASP.NET Core
- ASP.NET Core Framework
- Setup ASP.NET Core Environment
- Creat ASP.NET Core Web Application
- ASP.NET Core Project File and Folder
- ASP.NET Core Main Method
- ASP.NET Core InProcess Hosting
- Kestrel Web Server in ASP.NET Core
- Model View Controller MVC in ASP.NET Core MVC Framework
- Set up MVC in ASP.NET Core
- AddController vs AddMvc vs AddControllersWithViews vs AddRazorPages
- Models in ASP.NET Core MVC
- ASP.NET Core Dependency Injection

- Controllers in ASP.NET Core MVC
- Views in ASP.NET Core MVC
- Creat ASP.NET Core Application using MVC Template
- ViewData in ASP.NET Core MVC
- ViewBag in ASP.NET Core MVC
- Strongly Typed View in ASP.NET Core MVC
- ViewModel in ASP.NET Core MVC
- Routing in ASP.NET Core MVC
- Custom Routing in ASP.NET Core MVC
- Attribute Routing in ASP.NET Core MVC
- ASP.NET Core Attribute Routing using Tokens
- Layout View in ASP.NET Core MVC
- Sections in Layout Page in ASP.NET Core MVC
- ViewStart in ASP.NET Core MVC
- ViewImports in ASP.NET Core MVC
- Install Bootstrap in ASP.NET Core MVC
- Use Bootstrap in ASP.NET Core MVC
- Tag Helpers in ASP.NET Core MVC
- Image Tag Helper in ASP.NET Core
- Model Binding in ASP.NET Core

ASP.NET WEB API

- ASP.NET Web API Application
- Swagger in Web API Application
- Use Fiddler to test Web API
- Use POSTMAN to test Web API Services
- Use ASP.NET Web API using SQL Server
- Content Negotiation in Web API
- Media Type Formatter in Web API
- Implement GET Method in Web API
- Implement the POST Method in Web API Application
- Implement PUT Method in Web API Application
- Implement DELETE Method in Web API Application
- Custom Method Names in Web API
- Parameter Binding in Web API
- Consuming Web API Service From jQuery
- Calling Web API Service in a Cross-Domain Using jQuery AJAX
- Cross-Origin Resource Sharing in Web API
- Enable SSL in Visual Studio Development Server
- Enable HTTPS in Web API Service