FIT5032

Internet Applications Development

Week 1: Introduction to Web Development and ASP.NET

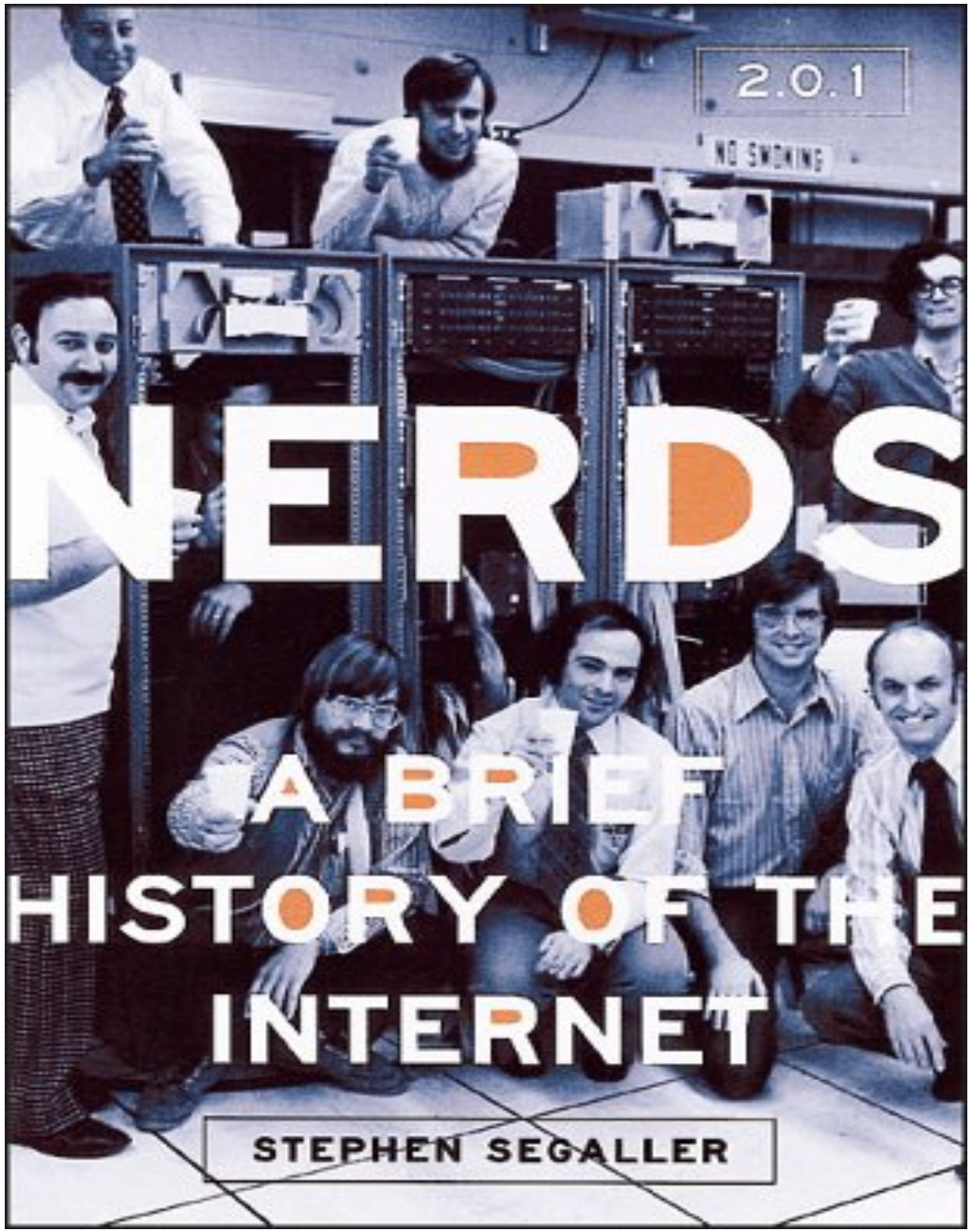
ABM Russel

www.shutterstock.com · 1141706672

Unit Topics

Week	Activities	Assessment
0		No formal assessment or activities are undertaken in week 0
1	Intro to Web development and ASP.NET	Note: Studio classes commence in week 1
2	The front end, user experience, accessibility and ASP.NET Scaffolding	
3	Introduction to C# & Version Control	
4	Entity Framework	
5	Fundamentals of Client side Javascript	Studio assessment task 1 due
6	Validation	
7	Security and Identity	
8	Sending Email, File Upload and Signal R	Studio assessment task 2 due
9	Web Optimisations & Evolution of ASP.NET CORE	
10	Modern JavaScript Web Development Approaches	
11	Testing and Deployment in Cloud	Studio assessment task 3 due
12	Review & Revision	Final Portfolio and Learning Summary due
	SWOT VAC	No formal assessment is undertaken in SWOT VAC
	Examination period	LINK to Assessment Policy:http://policy.monash.edu.au/policy-bank/academic/education/assessment/assessment-in-coursework-

History of the Internet



Ref: NERDS 2.01 A Brief History of the Internet, Stephen Segaller

FLUX: HTTP protocol Version 1.0

What was the main problem with the HTTP protocol Version 1.0?

1. No persistent connections
2. It was a stateless protocol
3. It was a Request-Response protocol
4. There was no security
5. All of the Above

KT33U4

Making a web request?

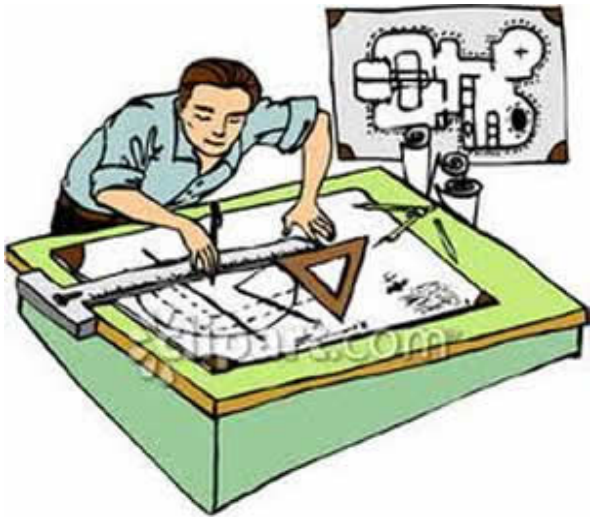
What happens when we make a request like:

http://www.monash.edu/

Internet Applications Development

Internet Applications Development

Where do we start?

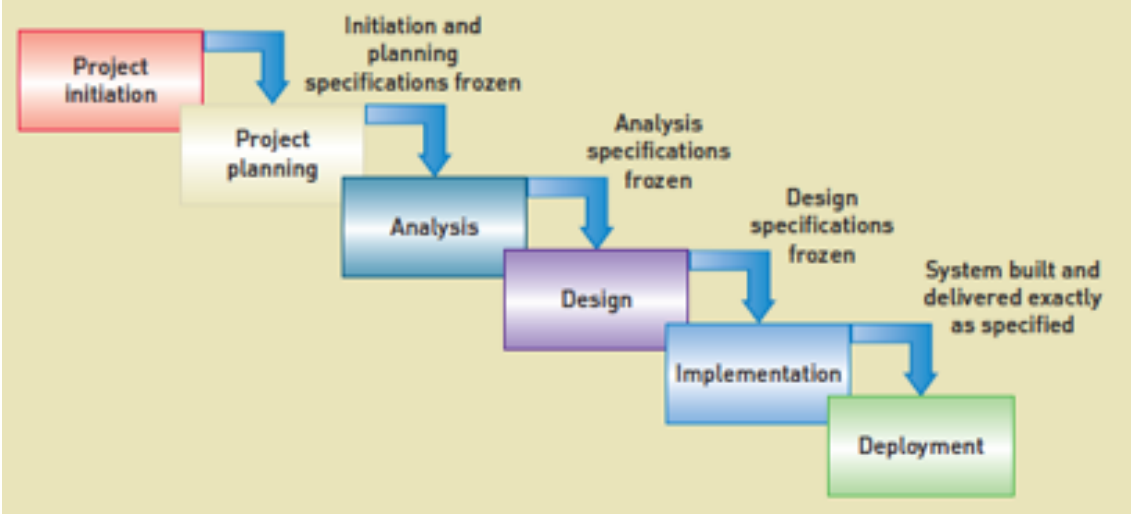


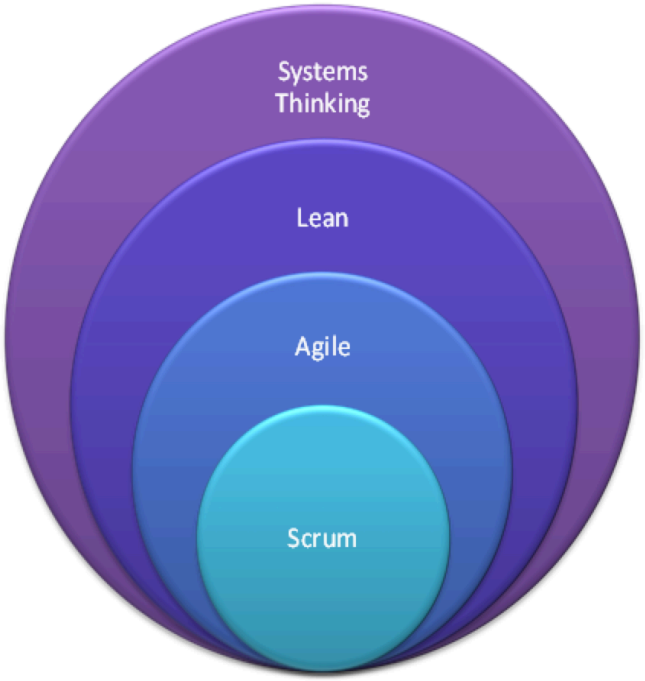
Need to avoid problems

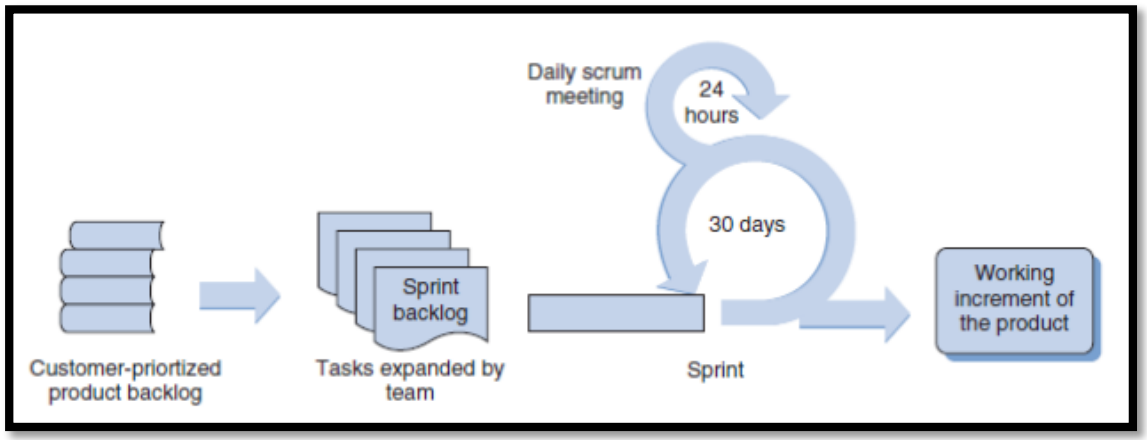
IBM's "Rule of Ten's"

- What is the rule of ten's?

Development Life-cycle





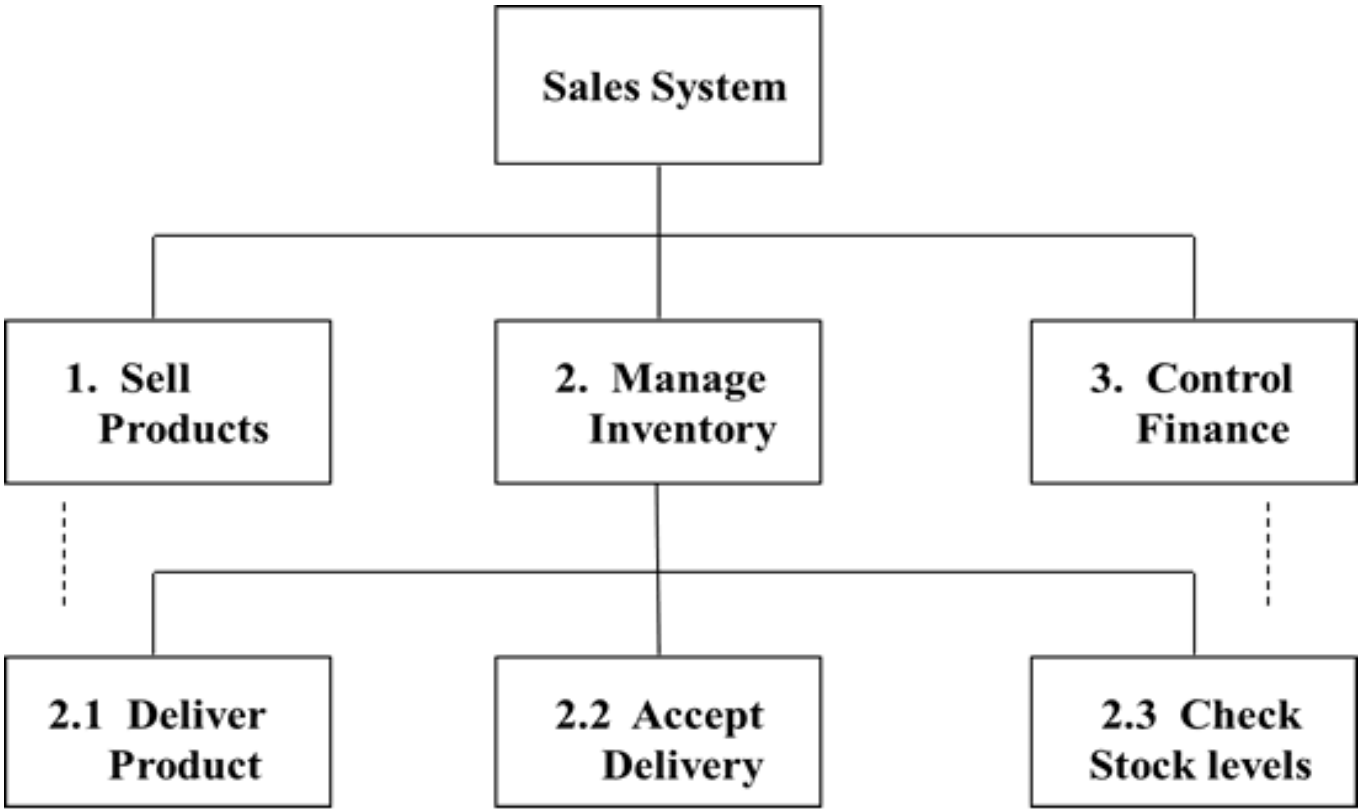


Development Methodology

- Proposed development evaluation
  - Why is it important to evaluate a proposal?
- Functional decomposition
  - Specification of a project
- Prototyping
  - Demonstrating project

Functional Decomposition

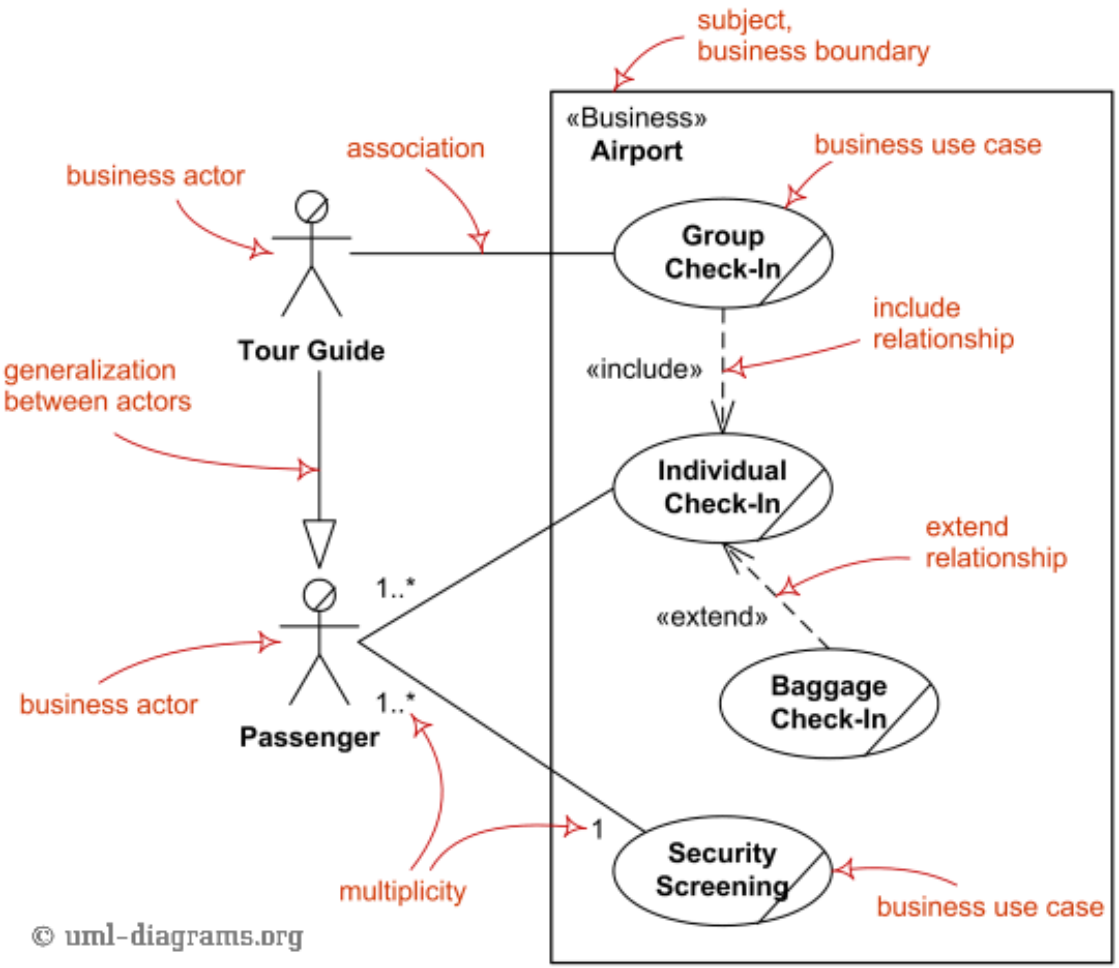
- Detailed specification of application



Prototyping

- Prototypes are mock ups. They are not the final system.
  - Low fidelity
  - Medium fidelity
  - High fidelity

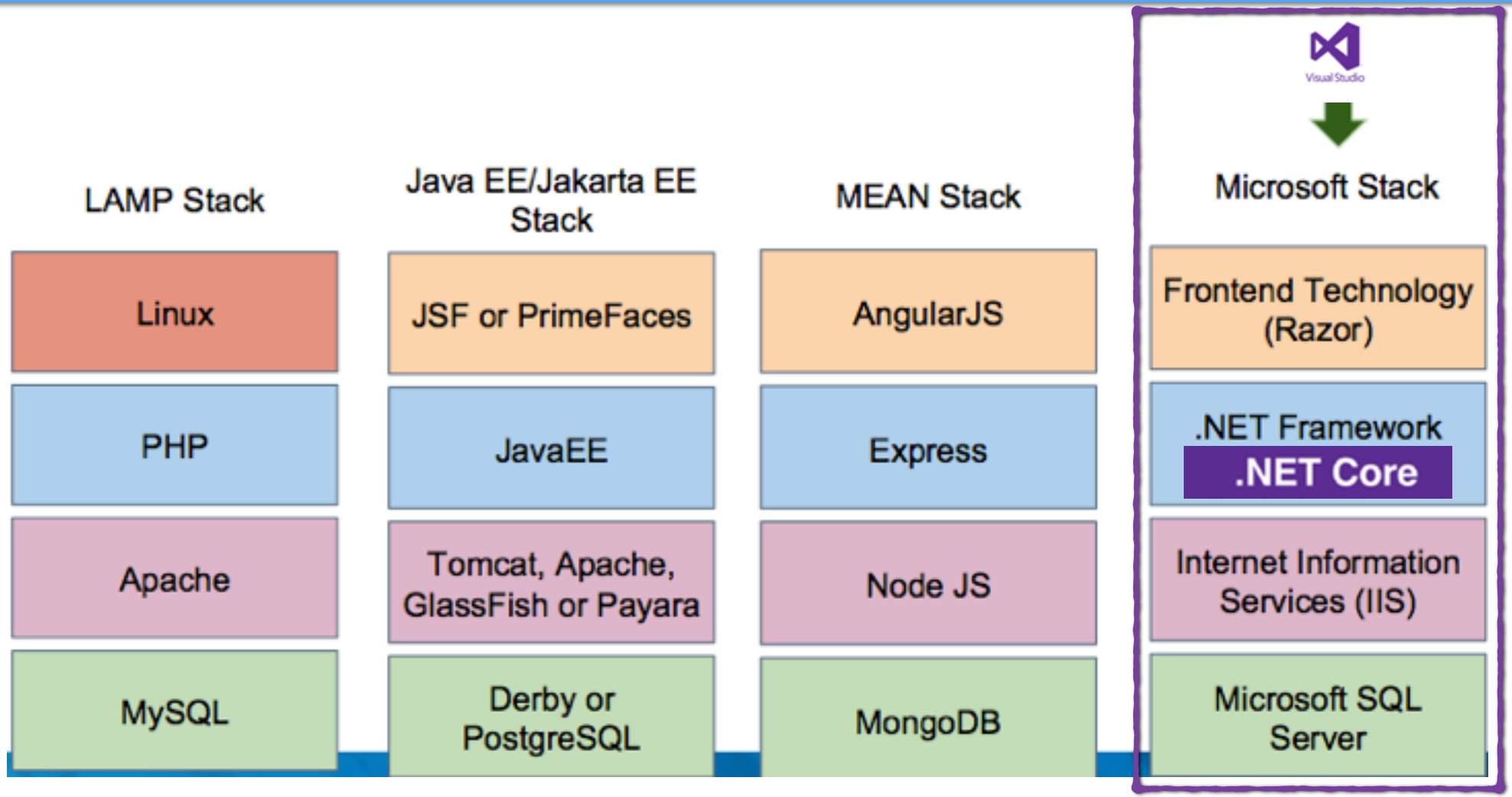
Use Cases and User Stories



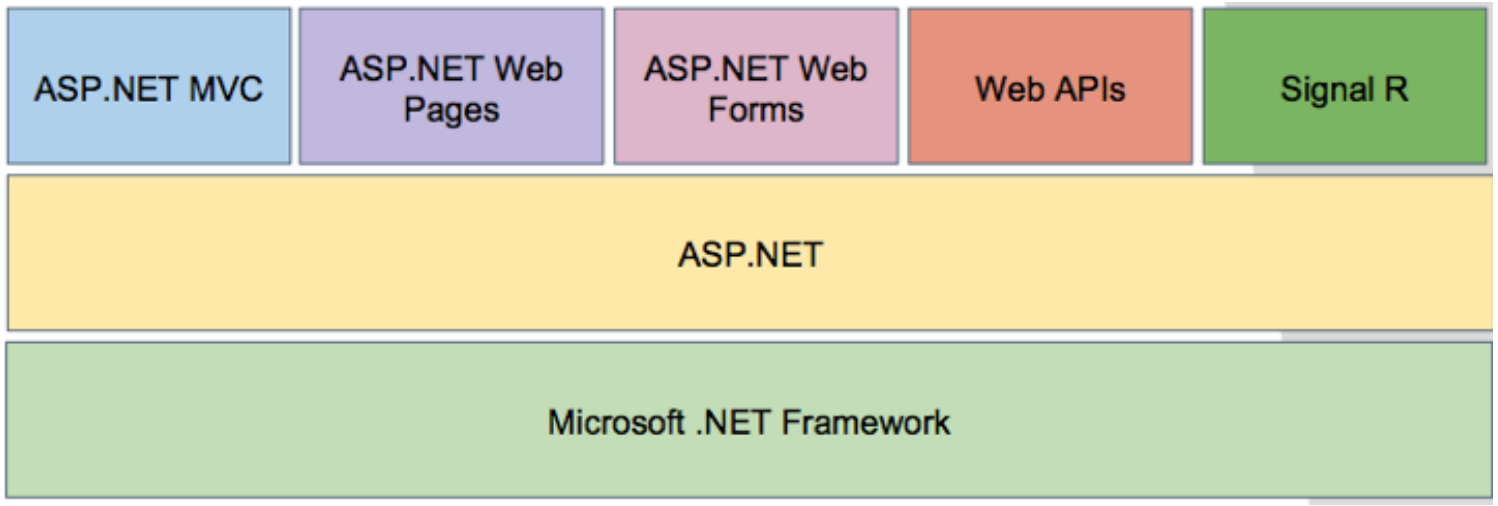
- As a {type of user}, I want to {some goals} so that {some reasons}

Software Development

Software Development Stacks



ASP.NET Overview



Empty
An empty project template for creating ASP.NET applications. This template does not have any content in it.

Web Forms
A project template for creating ASP.NET Web Forms applications. ASP.NET Web Forms lets you build dynamic websites using a familiar drag-and-drop, event-driven model. A design surface and hundreds of controls and components let you rapidly build sophisticated, powerful UI-driven sites with data access.

MVC
A project template for creating ASP.NET MVC applications. ASP.NET MVC allows you to build applications using the Model-View-Controller architecture. ASP.NET MVC includes many features that enable fast, test-driven development for creating applications that use the latest standards.

Web API
A project template for creating RESTful HTTP services that can reach a broad range of clients including browsers and mobile devices.

Single Page Application
A project template for creating rich client side JavaScript driven HTML5 applications using ASP.NET Web API. Single Page Applications provide a rich user experience which includes client-side interactions using HTML5, CSS3, and JavaScript.

Towards .NET Core

Empty
An empty project template for creating an ASP.NET Core application. This template does not have any content in it.

API
A project template for creating an ASP.NET Core application with an example Controller for a RESTful HTTP service. This template can also be used for ASP.NET Core MVC Views and Controllers.

Web Application
A project template for creating an ASP.NET Core application with example ASP.NET Core Razor Pages content.

Web Application (Model-View-Controller)
A project template for creating an ASP.NET Core application with example ASP.NET Core MVC Views and Controllers. This template can also be used for RESTful HTTP services.

Razor Class Library
A project template for creating a Razor class library.

Angular
A project template for creating an ASP.NET Core application with Angular.

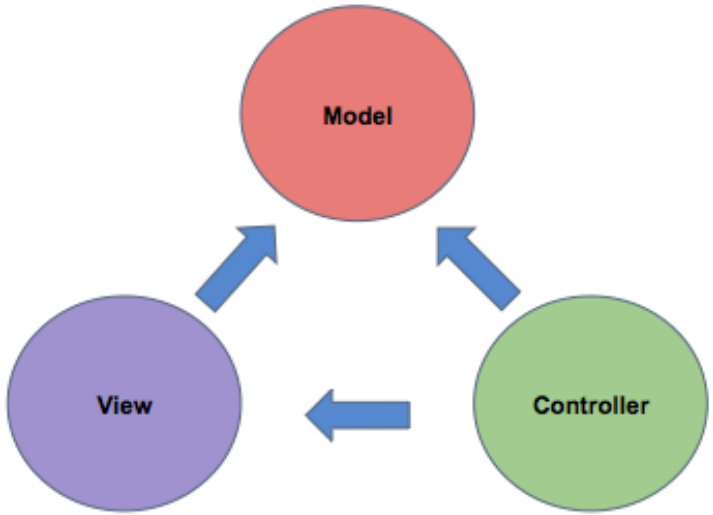
React.js
A project template for creating an ASP.NET Core application with React.js.

React.js and Redux
A project template for creating an ASP.NET Core application with React.js and Redux.

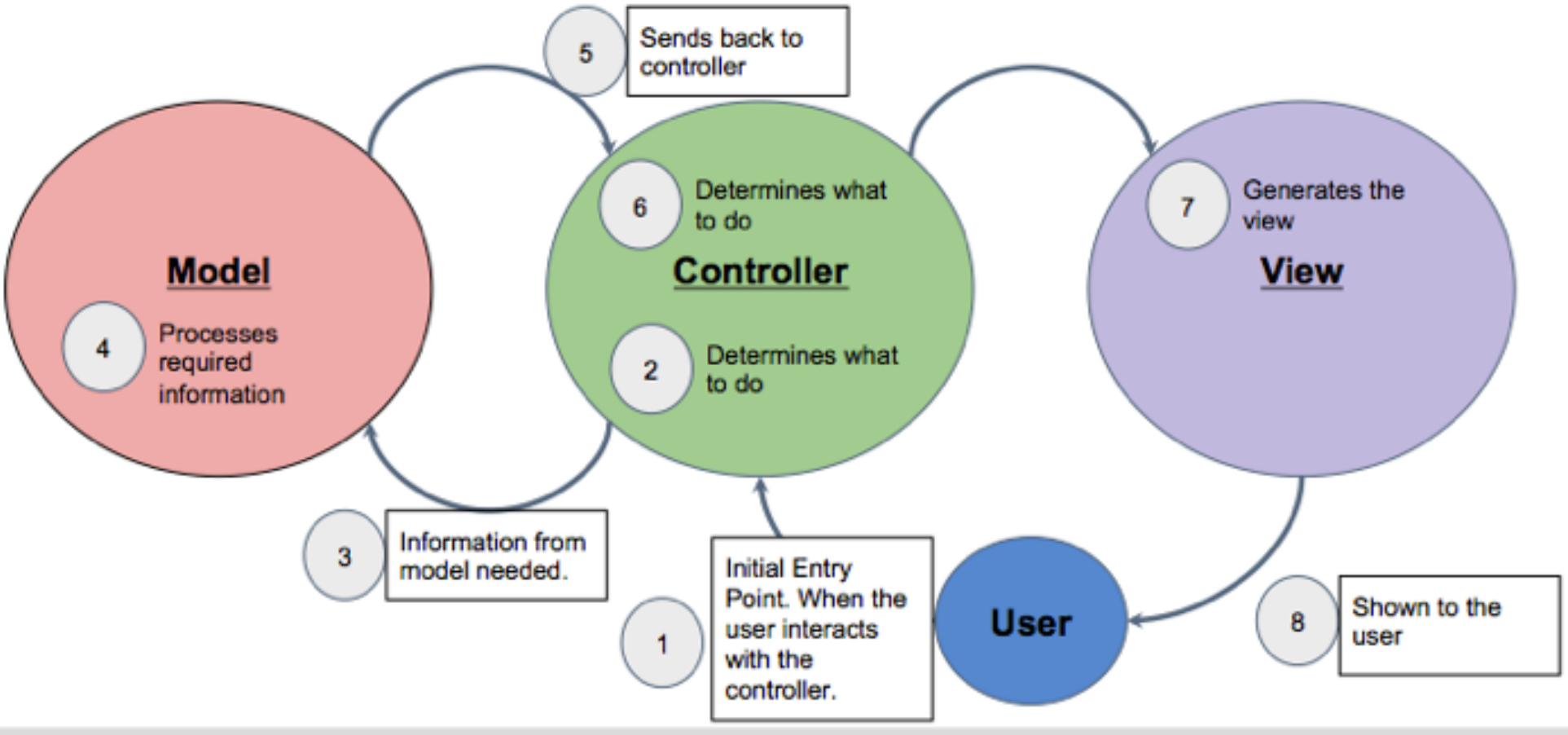
What is MVC

- The Model-View-Controller (MVC) **architectural pattern** separates an application into three main groups of components: Models, Views, and Controllers.

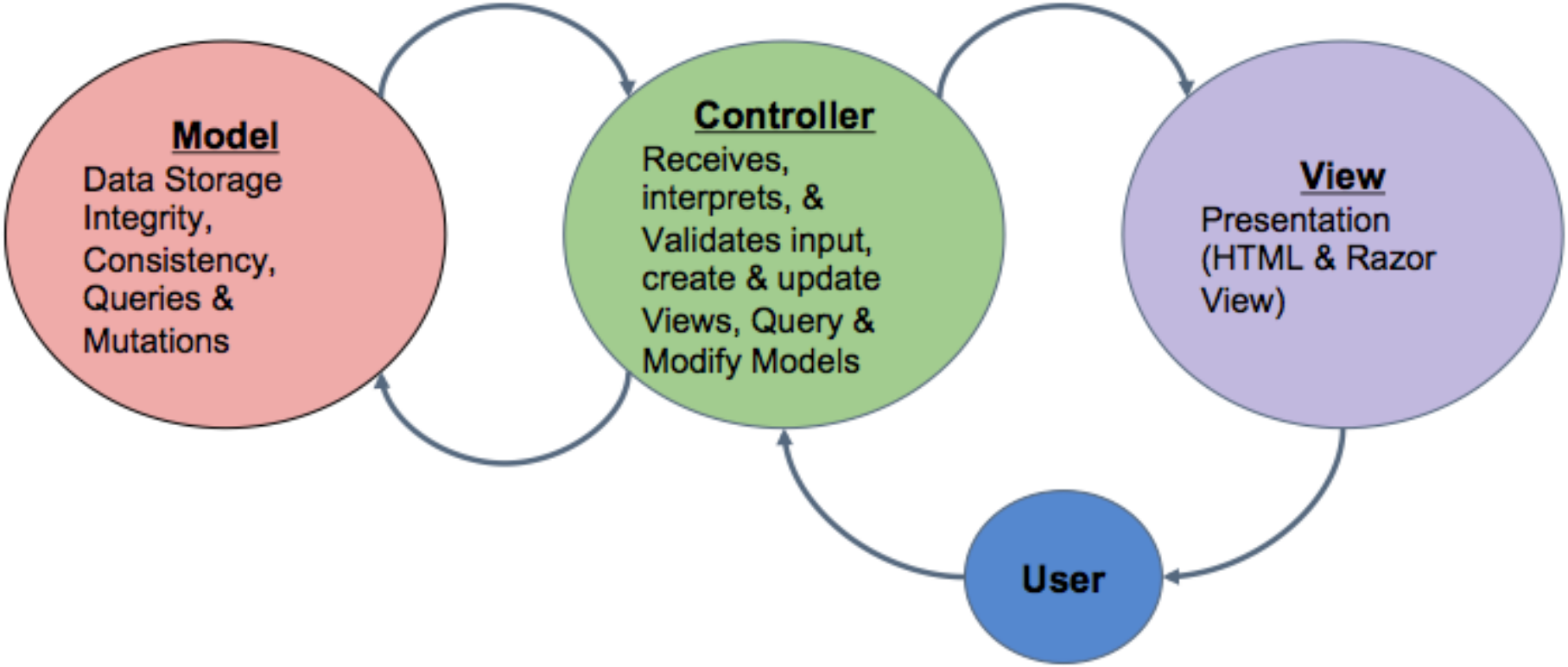
- This architectural pattern helps to achieve **separation of concerns.**



MVC in Action



ASP.NET MVC



- Separation of Concerns (SoC)
- Single Responsibility Principle
- Dependency Inversion Principle

Model

A class or set of classes that describes all the business logic and additionally handles data access for an application.

Also contains code that defines its relationship with other models.

Defines the data validation rules to be used when adding or updating data.

Controller

Controls the application flow or logic of the application

Controller logic decides what response is to be generated

Controller logic normally contains calls to models to access data, and also other functionalities like access control checks etc

Controller passes the response (output) to the view

View

View is the outputs or responses that are sent back to the user once a request is processed.

Consist of markup (like HTML) code with embedded .NET code. Can also be other forms of output like XML, PDF documents etc.

Views can be thought of as the presentation layer of an application and ideally should be as "dumb" as possible

Advantages of ASP.NET MVC

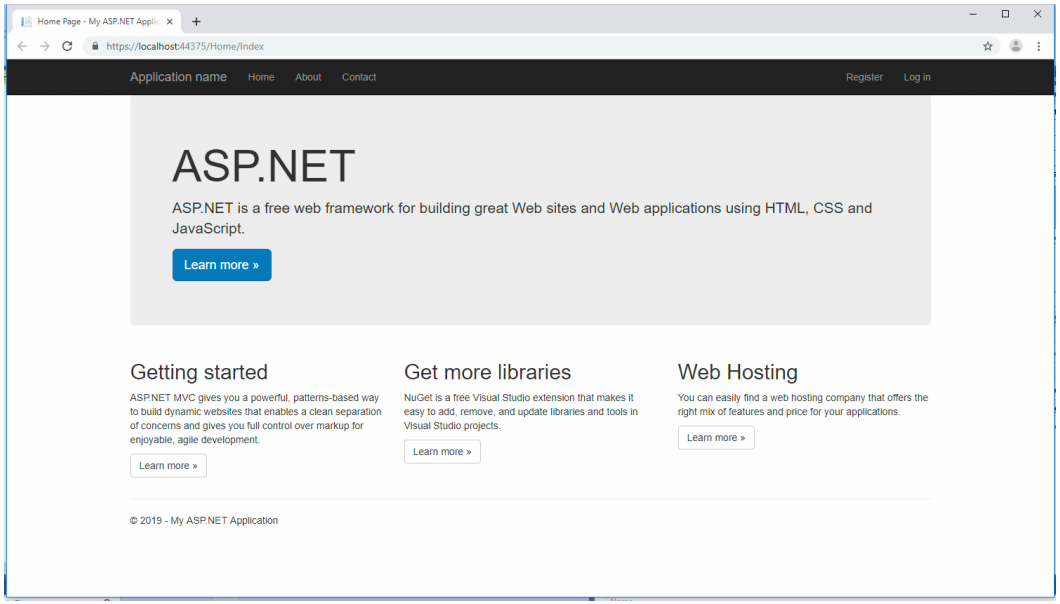
- Enables the full control over the rendered HTML.
- Provides clean separation of concerns(SoC).
- Enables Test Driven Development (TDD).
- Easy integration with JavaScript frameworks.
- Following the design of stateless nature of the web.
- RESTful urls that enables Search Engine Optimisation.
- No ViewState and PostBack events in comparison to ASP.NET Web Forms.

Lecture Summary

- Introduction to Internet Applications Development
- Understand the purpose of ASP.NET MVC
- Understand the benefits of the MVC architecture

Week 1 Studio Overview

- Introduction and setup eFolio
- Run Visual Studio 2019 Community Edition IDE
- Research different IDEs
- Core features of Visual Studio IDE



Next week: Front End & Scaffolding

- The front end, user experience, accessibility
- ASP.NET Scaffolding