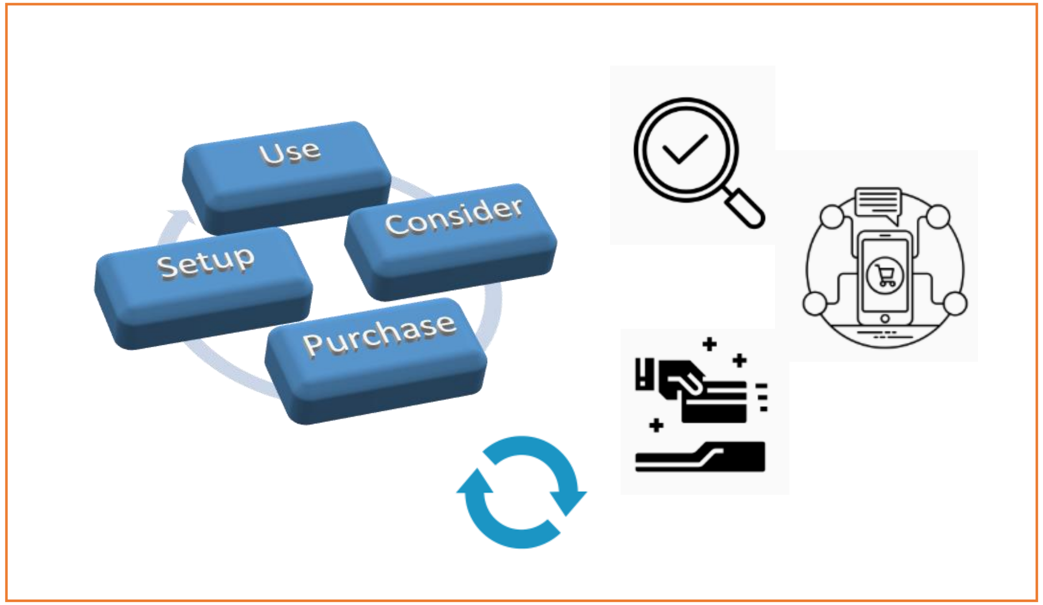
Software Architecture | Retail Store

Presentation



Agenda

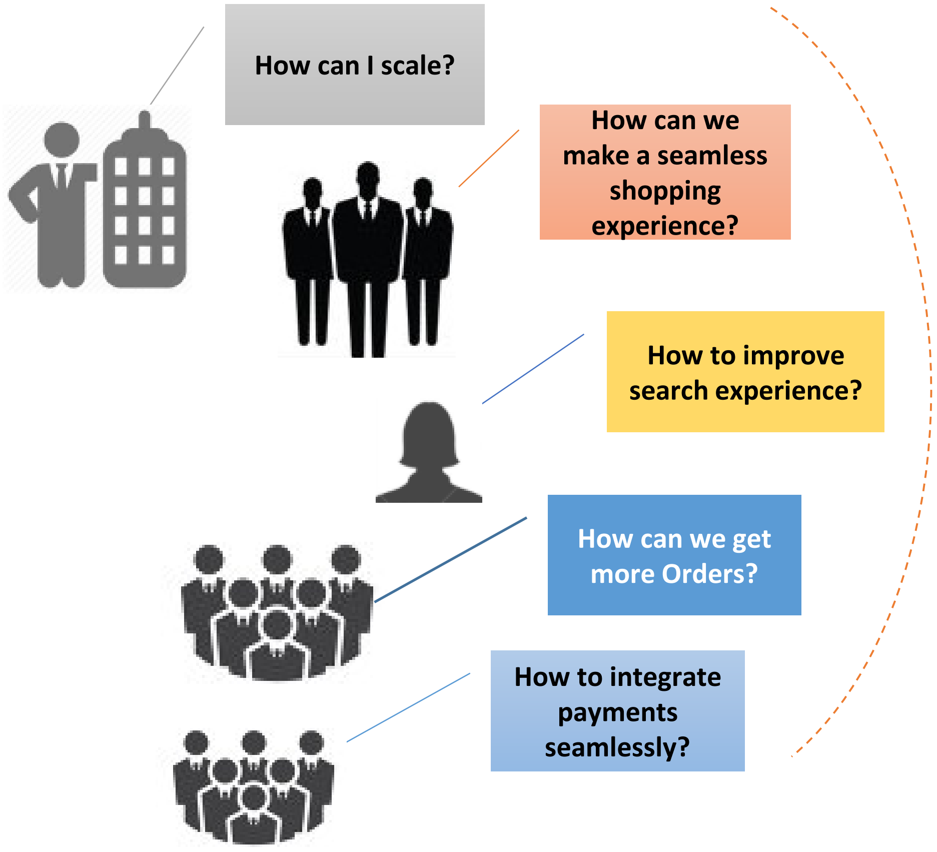
Needs | Layered View | Database | Steps | Summary

Abhilash G

Source:
Picture -

Software Architecture| Need

The Need



How can I scale?

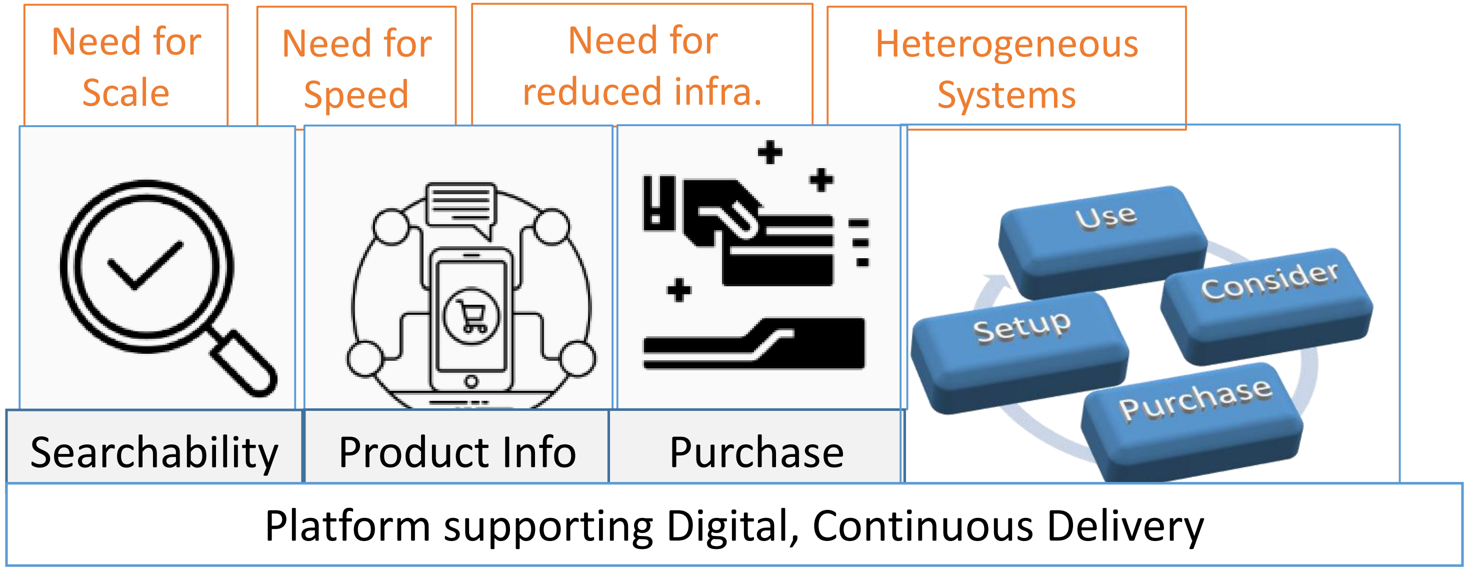
How can we make a seamless shopping experience?

How to improve search experience?

How can we get more Orders?

How to integrate payments seamlessly?

Driving Forces!



Need for Scale

Need for Speed

Need for reduced infra.

Heterogeneous Systems

Searchability

Product Info

Purchase

Platform supporting Digital, Continuous Delivery

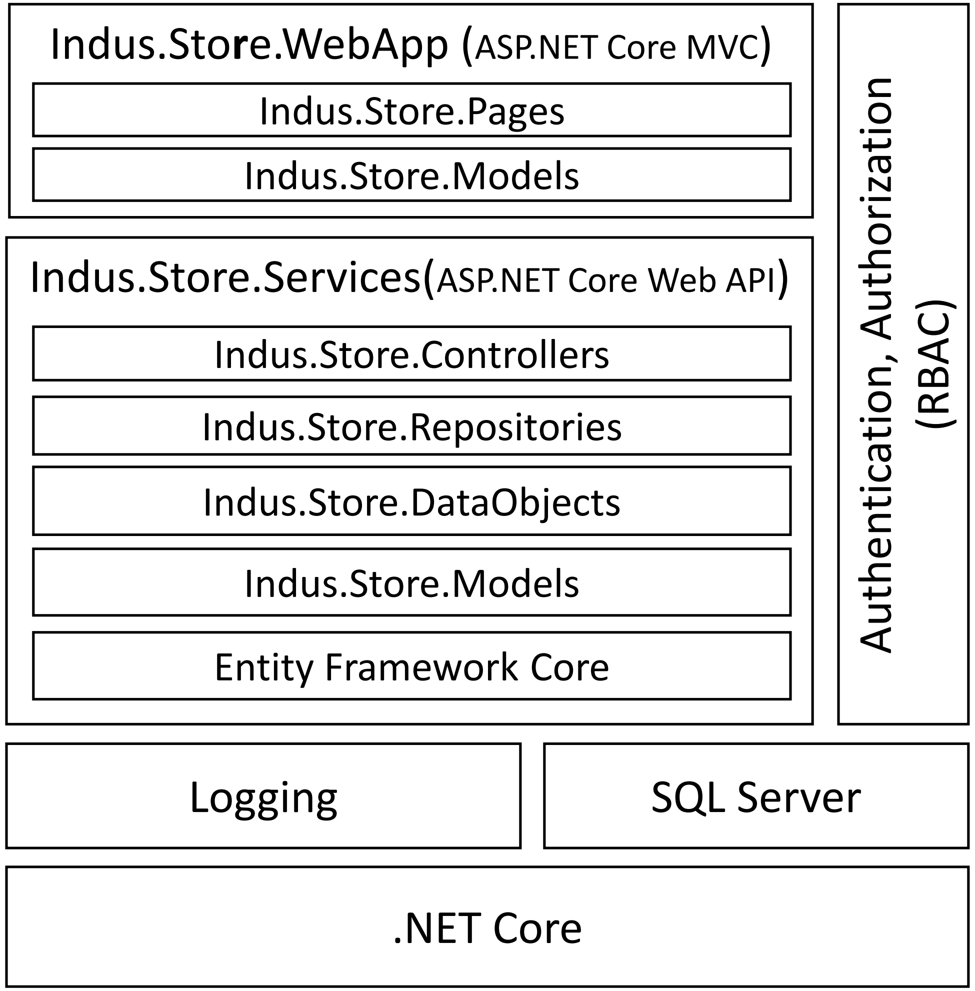
The key aspects is to support customer in buying process. This includes Consider, Purchase, Setup ad Use as the customer life cycle. Consideration to Purchase includes searchability, product education or making customer aware of the product and guide them to purchase integrating payments interface. Delivery, Ordering and return options are as well key elements of the experience. Intelligent Connected Systems aspect play the main digital element.

Sources: Qualtrics
https://www.qualtrics.com/blog/amazon-customer-experience-leader/

Driven by searchability, awareness and payments interactions!

Software Architecture| Layered View

Definitions and Standards



Indus.Store.WebApp (ASP.NET Core MVC)

Indus.Store.Pages

Indus.Store.Models

Indus.Store.Services(ASP.NET Core Web API)

Indus.Store.Controllers

Indus.Store.Repositories

Indus.Store.DataObjects

Indus.Store.Models

Entity Framework Core

Authentication, Authorization (RBAC)

Logging

SQL Server

.NET Core

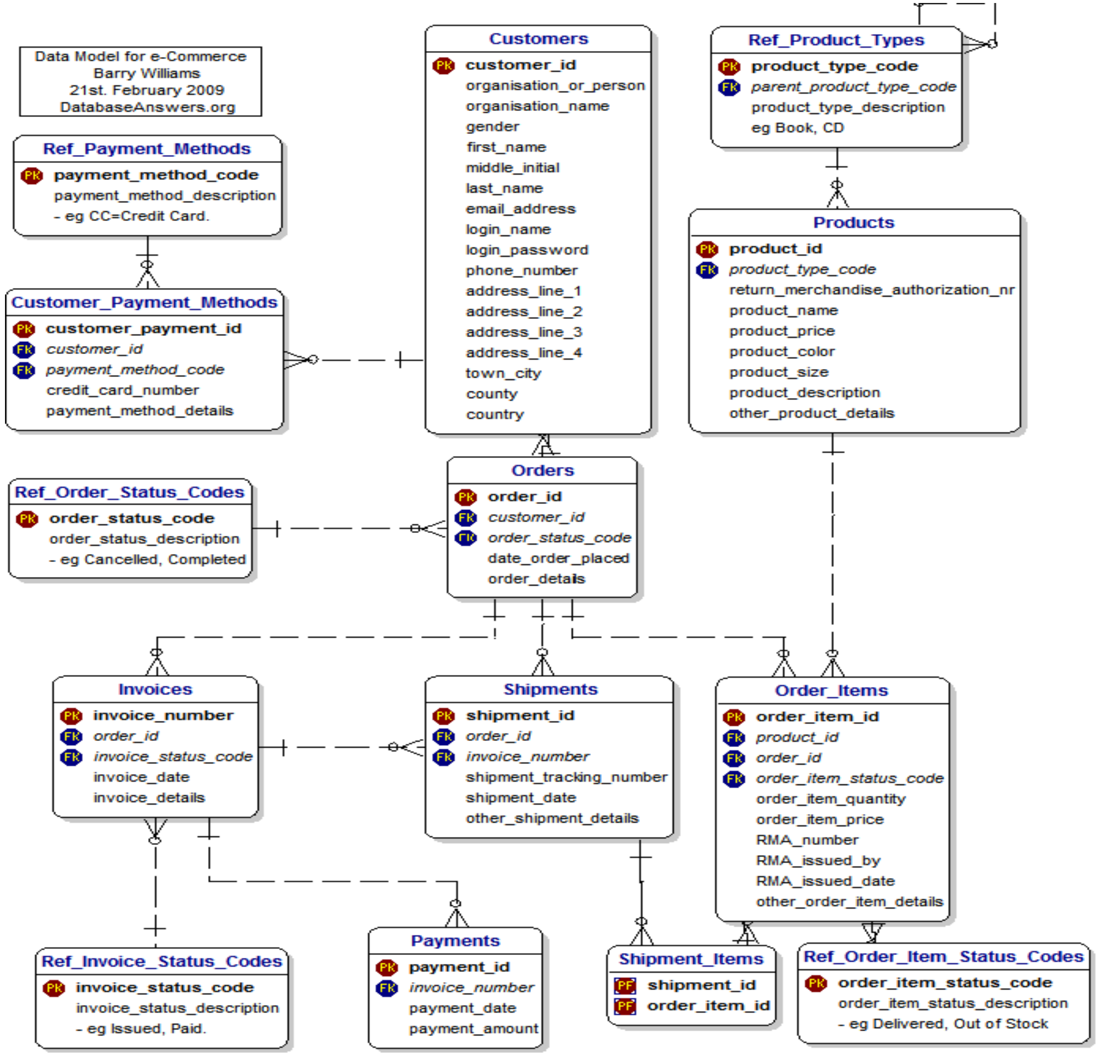
The overall approach is based on .NET Core. Database is SQL Server. For ORM we use Entity Framework Core.

The Services are developed using ASP.NET Core Web API.

The WebApp is developed using ASP.NET Core MVC.

Software Architecture| Database

Definitions and Standards



Data Model for e-Commerce
Barry Williams
21st. February 2009
DatabaseAnswers.org

Ref_Payment_Methods
- PK payment_method_code
- payment_method_description
- eg CC=Credit Card.

Customer_Payment_Methods
- PK customer_payment_id
- FK *customer_id*
- FK *payment_method_code*
- credit_card_number
- payment_method_details

Customers
- PK customer_id
- organisation_or_person
- organisation_name
- gender
- first_name
- middle_initial
- last_name
- email_address
- login_name
- login_password
- phone_number
- address_line_1
- address_line_2
- address_line_3
- address_line_4
- town_city
- county
- country

Ref_Product_Types
- PK product_type_code
- FK *parent_product_type_code*
- product_type_description
- eg Book, CD

Products
- PK product_id
- FK *product_type_code*
- return_merchandise_authorization_nr
- product_name
- product_price
- product_color
- product_size
- product_description
- other_product_details

Ref_Order_Status_Codes
- PK order_status_code
- order_status_description
- eg Cancelled, Completed

Orders
- PK order_id
- FK *customer_id*
- FK *order_status_code*
- date_order_placed
- order_details

Invoices
- PK invoice_number
- FK *order_id*
- FK *invoice_status_code*
- invoice_date
- invoice_details

Shipments
- PK shipment_id
- FK *order_id*
- FK *invoice_number*
- shipment_tracking_number
- shipment_date
- other_shipment_details

Order_Items
- PK order_item_id
- FK *product_id*
- FK *order_id*
- FK *order_item_status_code*
- order_item_quantity
- order_item_price
- RMA_number
- RMA_issued_by
- RMA_issued_date
- other_order_item_details

Ref_Invoice_Status_Codes
- PK invoice_status_code
- invoice_status_description
- eg Issued, Paid.

Payments
- PK payment_id
- FK *invoice_number*
- payment_date
- payment_amount

Shipment_Items
- PF shipment_id
- PF order_item_id

Ref_Order_Item_Status_Codes
- PK order_item_status_code
- order_item_status_description
- eg Delivered, Out of Stock

A relational database is defined here. The Basis of schema is an open model defined in below link.

https://stackoverflow.com/questions/35612778/database-schema-for-an-online-shop

http://www.databaseanswers.org/data_models/index.htm

Images : Credit – the https://thenounproject.com/browse/?i=2864641

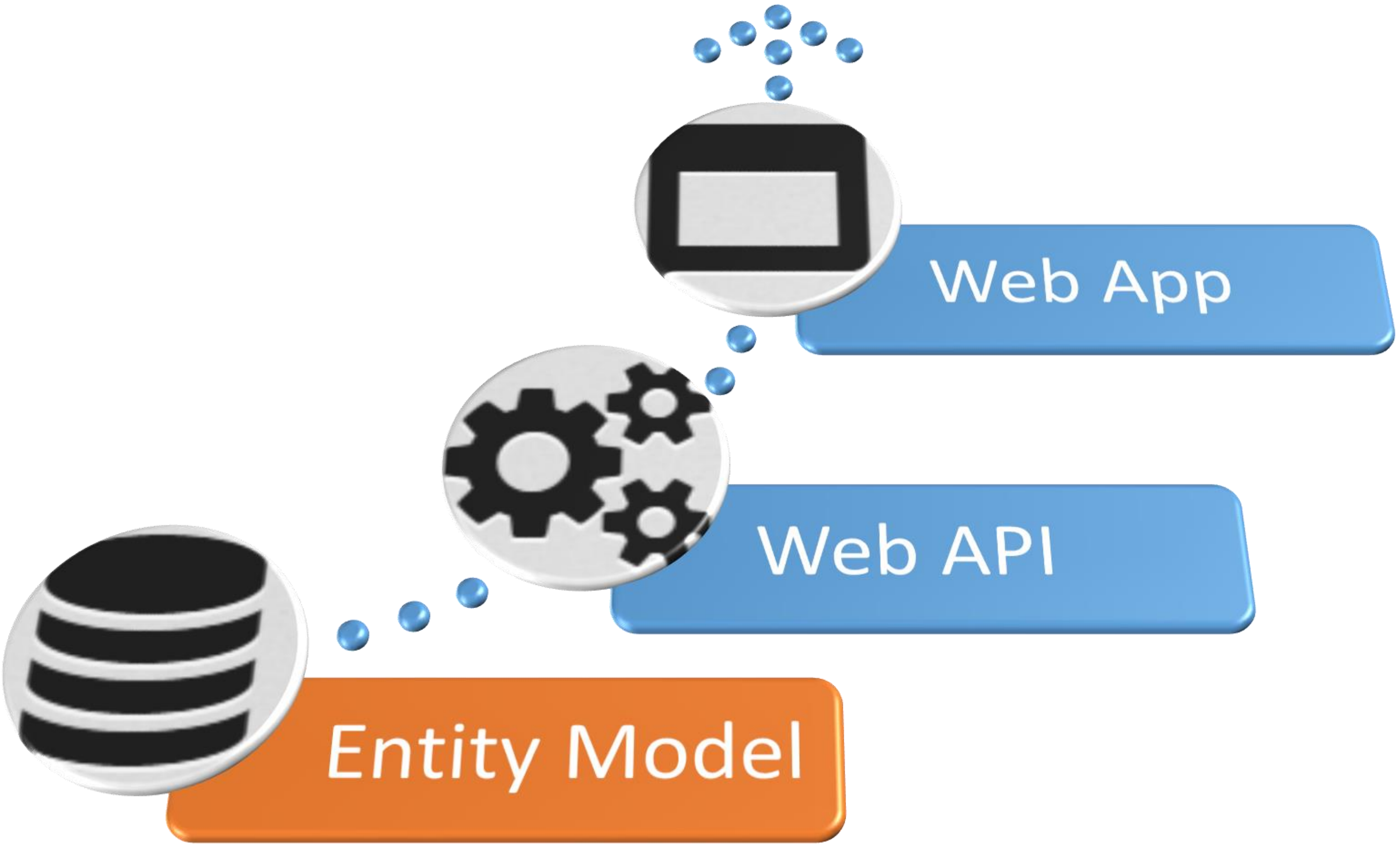
https://fontawesome.com/v4.7.0/icons/

Amazon Case Study
https://www.smartinsights.com/digital-marketing-strategy/online-business-revenue-models/amazon-case-study/

https://www.shopkeep.com/blog/retail-shopping-experience#step-1

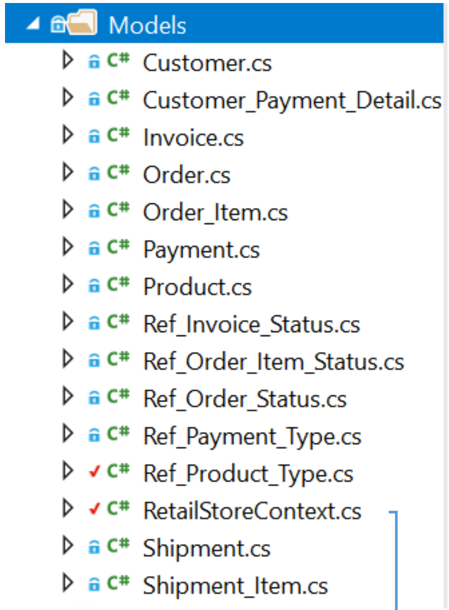
Software Architecture| Steps

Steps Involved in Development



1. Start with Entity Model

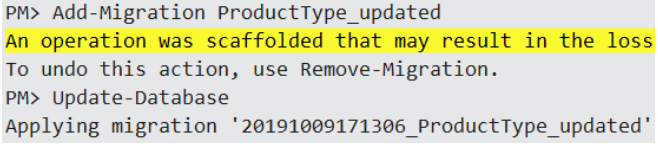
1.1 Prepare Model Classes



```
◢ Models
   ▷ C# Customer.cs
   ▷ C# Customer_Payment_Detail.cs
   ▷ C# Invoice.cs
   ▷ C# Order.cs
   ▷ C# Order_Item.cs
   ▷ C# Payment.cs
   ▷ C# Product.cs
   ▷ C# Ref_Invoice_Status.cs
   ▷ C# Ref_Order_Item_Status.cs
   ▷ C# Ref_Order_Status.cs
   ▷ C# Ref_Payment_Type.cs
   ▷ ✓ C# Ref_Product_Type.cs
   ▷ ✓ C# RetailStoreContext.cs
   ▷ C# Shipment.cs
   ▷ C# Shipment_Item.cs
```

1.2 Define DbContext

1.3 Now in Package Manager Console ->
Add-Migration Initial
Update-Database

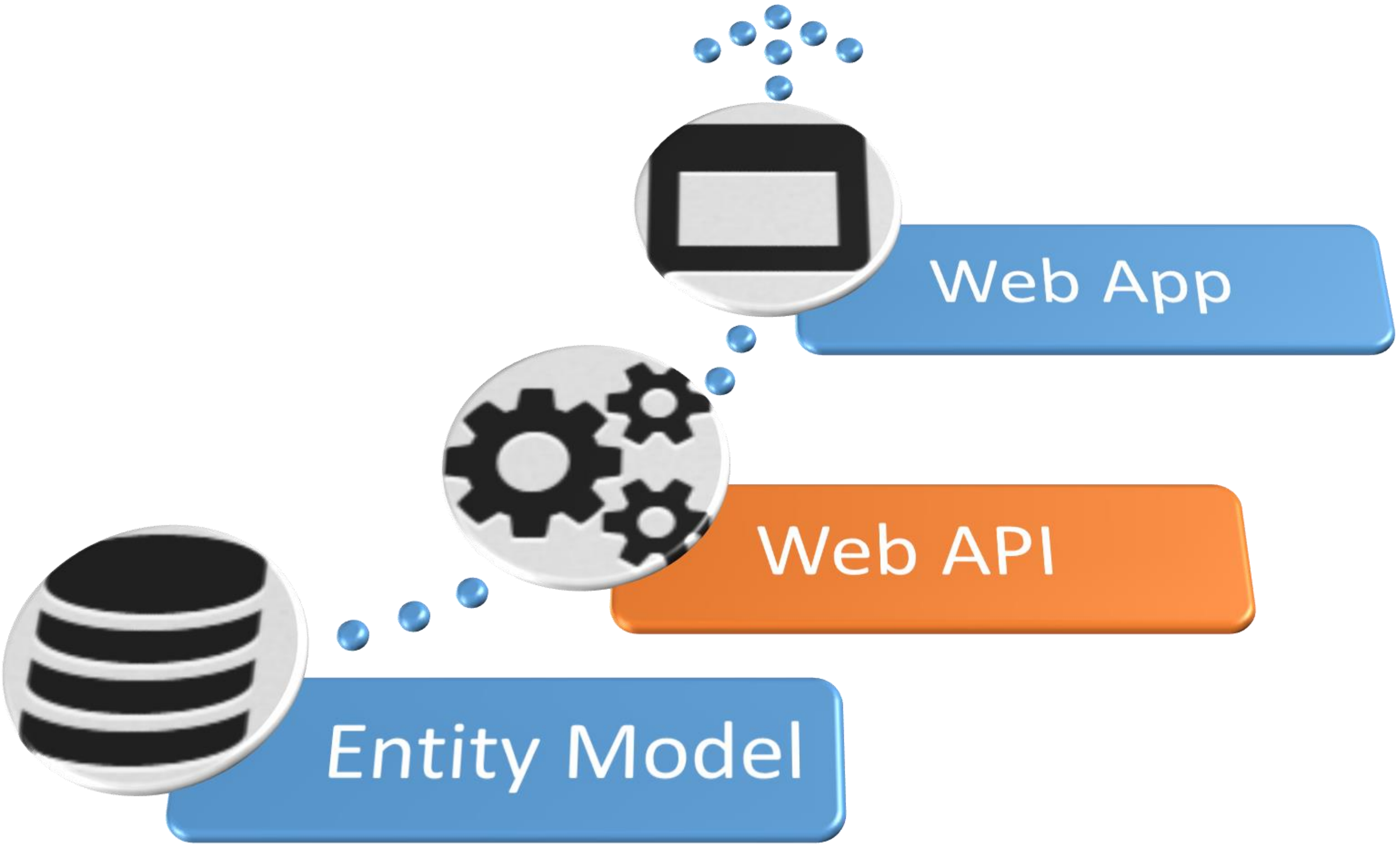


```
PM> Add-Migration ProductType_updated
An operation was scaffolded that may result in the loss
To undo this action, use Remove-Migration.
PM> Update-Database
Applying migration '20191009171306_ProductType_updated'
```

The Steps start with Entity Model, Web API and Web App Development

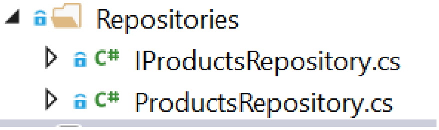
Software Architecture | Steps

Steps Involved in Development



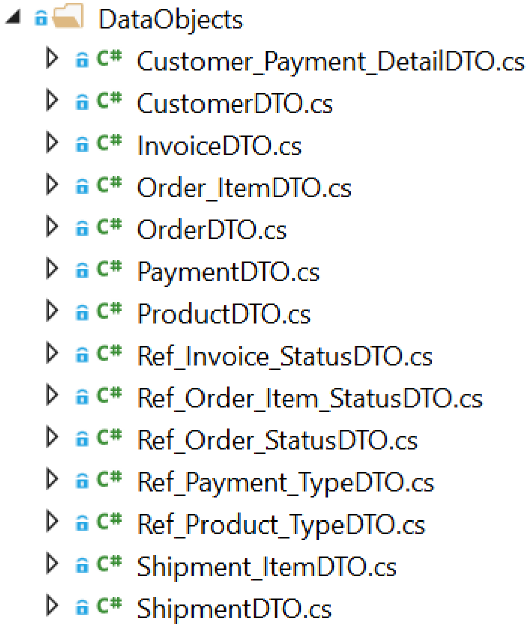
2. Define Web API

2.1 Prepare Repositories



- Repositories
  - IProductsRepository.cs
  - ProductsRepository.cs

2.2 Create Data Transfer Objects



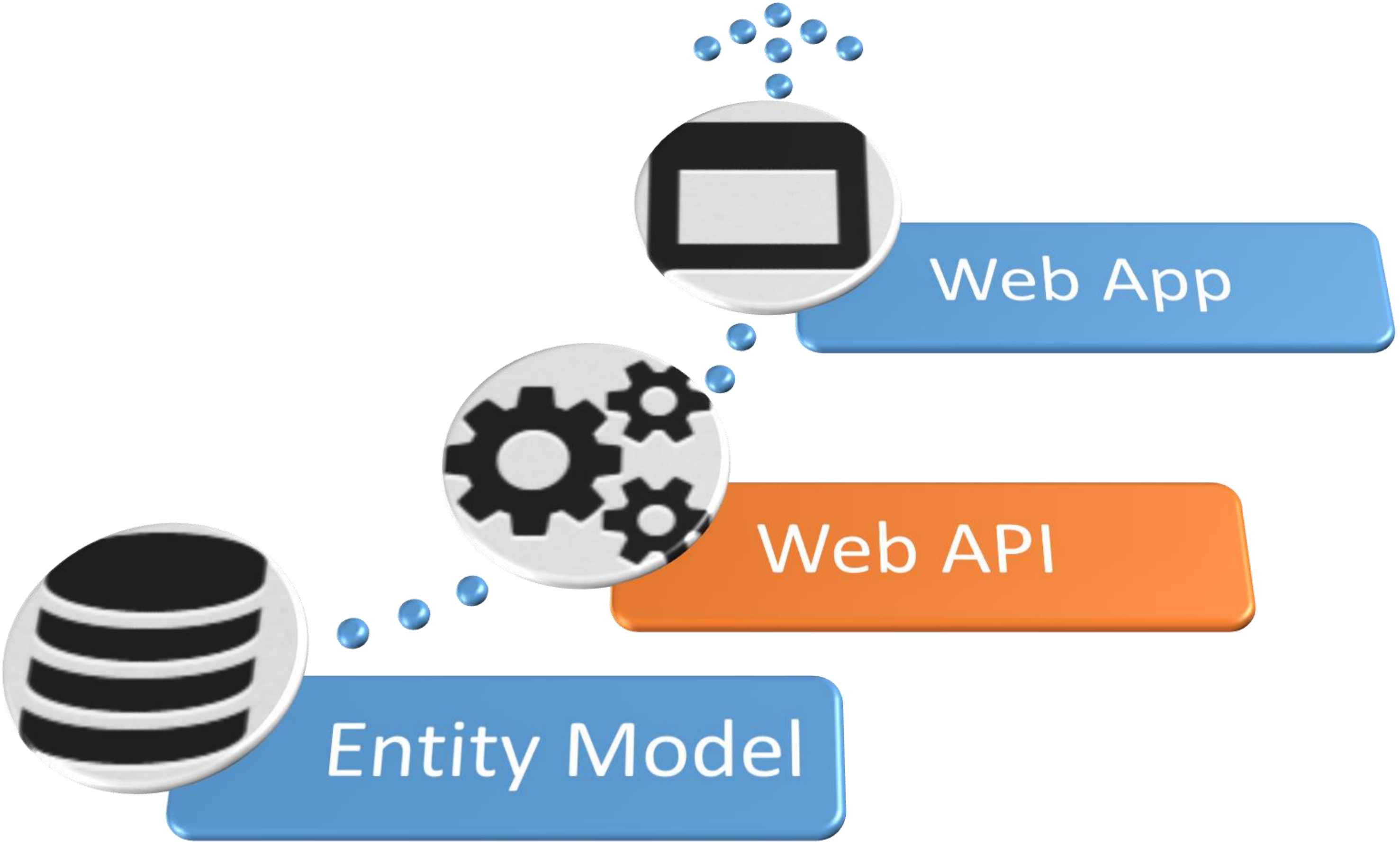
- DataObjects
  - Customer_Payment_DetailDTO.cs
  - CustomerDTO.cs
  - InvoiceDTO.cs
  - Order_ItemDTO.cs
  - OrderDTO.cs
  - PaymentDTO.cs
  - ProductDTO.cs
  - Ref_Invoice_StatusDTO.cs
  - Ref_Order_Item_StatusDTO.cs
  - Ref_Order_StatusDTO.cs
  - Ref_Payment_TypeDTO.cs
  - Ref_Product_TypeDTO.cs
  - Shipment_ItemDTO.cs
  - ShipmentDTO.cs

2.3 Define Controllers -> Next Page

The Web API involved defining Repositories, Data Transfer Objects and Controllers

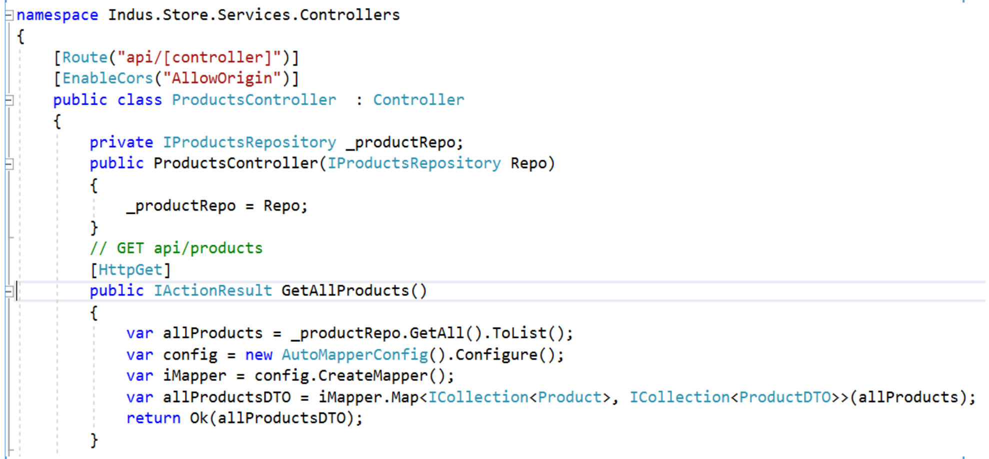
Software Architecture| Steps

Steps Involved in Development



2. Define Web API

2.3 Define Controllers

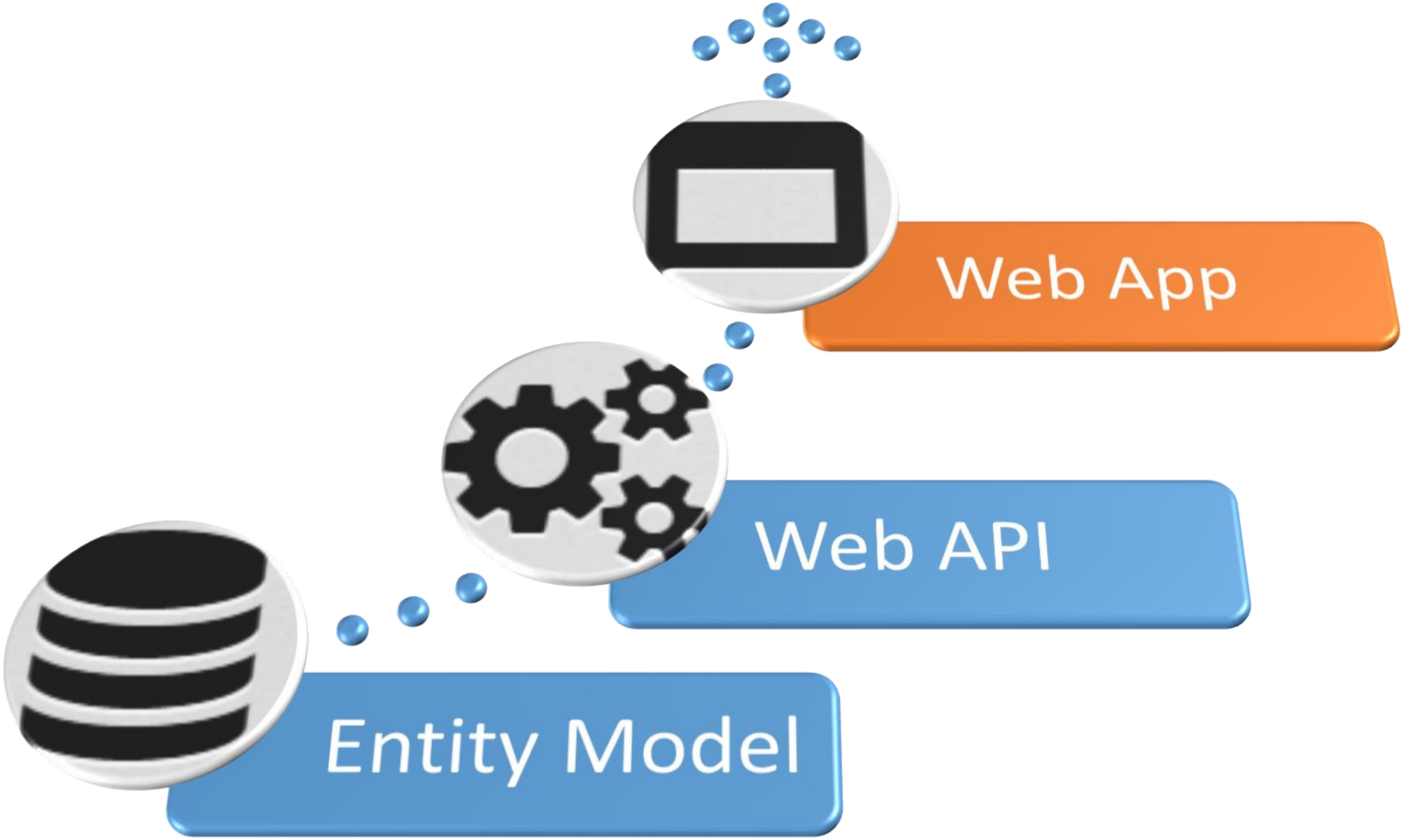


```
namespace Indus.Store.Services.Controllers
{
    [Route("api/[controller]")]
    [EnableCors("AllowOrigin")]
    public class ProductsController  : Controller
    {
        private IProductsRepository _productRepo;
        public ProductsController(IProductsRepository Repo)
        {
            _productRepo = Repo;
        }
        // GET api/products
        [HttpGet]
        public IActionResult GetAllProducts()
        {
            var allProducts = _productRepo.GetAll().ToList();
            var config = new AutoMapperConfig().Configure();
            var iMapper = config.CreateMapper();
            var allProductsDTO = iMapper.Map<ICollection<Product>, ICollection<ProductDTO>>(allProducts);
            return Ok(allProductsDTO);
        }
```

Controllers handle the HTTP Routes.

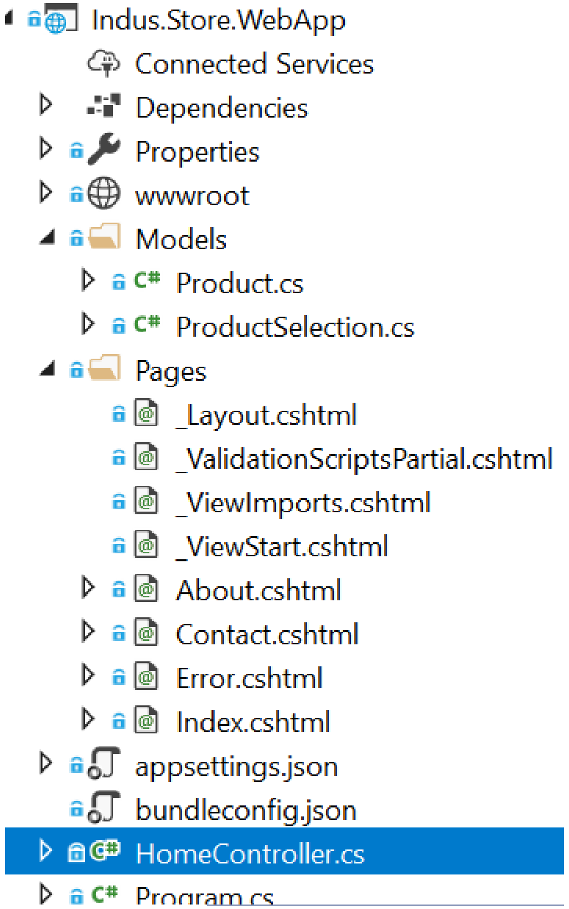
Software Architecture| Steps

Steps Involved in Development



3. Define Web App

3.1 Define Razor Pages



```
Indus.Store.WebApp
    Connected Services
    Dependencies
    Properties
    wwwroot
    Models
        Product.cs
        ProductSelection.cs
    Pages
        _Layout.cshtml
        _ValidationScriptsPartial.cshtml
        _ViewImports.cshtml
        _ViewStart.cshtml
        About.cshtml
        Contact.cshtml
        Error.cshtml
        Index.cshtml
    appsettings.json
    bundleconfig.json
    HomeController.cs
    Program.cs
```

The Web App based on Razor pages provide the presentation Layer

“Be the change you want to see in the world!” *MAHATMA GANDHI*