...ulita - copia\LaPaulita.Core\Common\EntityCommon.cs 1

1 namespace LaPaulita.Entity.Common 2 {

3 public class EntityCommon 4 {

1. public int Id { get; set; }
2. public int CreatedBy { get; set; }
3. public DateTime CreatedDate { get; set; } = DateTime.UtcNow;
4. public int UpdatedBy { get; set; }
5. public DateTime UpdatedDate { get; set; } = DateTime.UtcNow; 10 }

11 }

12

...lita - copia\LaPaulita.Core\Entities\OrderHeader.cs 1

1. using LaPaulita.Entity.Common;
2. using LaPaulita.Entity.Enums; 3

4 namespace LaPaulita.Entity.Entities 5 {

6 public class OrderHeader : EntityCommon 7 {

1. public int ClientId { get; set; }
2. public string ShippingAddress { get; set; }
3. public int ShippingCity { get; set; }
4. public int ShippingCountry { get; set; }
5. public string ShippingZip { get; set; }
6. public DateTime DateOrder { get; set; } = DateTime.UtcNow;
7. public TransportType TransportType { get; set; } = TransportType.Road;
8. public DiscountType DiscountType { get; set; } = DiscountType.Percentage;
9. public int Discount { get; set; } = 10; 17 }

18 }

19

...aulita - copia\LaPaulita.Core\Enums\DiscountType.cs 1

1 namespace LaPaulita.Entity.Enums 2 {

|  |  |  |
| --- | --- | --- |
| 3 |  | public enum DiscountType |
| 4 |  | { |
| 5 |  | Percentage = 1, |
| 6 |  | Flat = 2 |
| 7 |  |  |
| 8 |  | } |
| 9 | } |  |
| 10 |  |  |

...ulita - copia\LaPaulita.Core\Enums\TransportType.cs 1

1 namespace LaPaulita.Entity.Enums 2 {

|  |  |  |
| --- | --- | --- |
| 3 |  | public enum TransportType |
| 4 |  | { |
| 5 |  | Maritime = 1, |
| 6 |  | Air = 2, |
| 7 |  | Rail = 3, |
| 8 |  | Road = 4 |
| 9 |  | } |
| 10 | } |  |
| 11 |  |  |

...ta - copia\LaPaulita.Core\Interfaces\IUnitOfWork.cs 1

1 namespace LaPaulita.Entity.Interfaces 2 {

3 public interface IUnitOfWork 4 {

5 Task SaveChange(); 6 }

7 }

8

...ta - copia\LaPaulita.Sales\Agregates\CreateOrder.cs 1

1. using LaPaulita.Sales.BusinessRules.DTOs;
2. using LaPaulita.Sales.BusinessRules.ValueObject; 3

4 namespace LaPaulita.Sales.BusinessRules.Agregates 5 {

6 public class CreateOrder : OrderHeader 7 {

8 //

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*

9 // Esto si lo hago con una clase tipo record inmutable. 10 //

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*

11

1. //readonly List<OrdersDetails> records = new List<OrdersDetails> ();
2. //public IReadOnlyList<OrdersDetails> Details => records; 14
3. ///// <summary>
4. ///// Método que agrega un nuevo record a <c>OrdersDetails</c> de una orden de compra
5. ///// <c>OrderHeader</c>.
6. ///// Si el nuevo record a agregar tiene un <c>ProductId</c> que ya existe en
7. ///// <c>OrdersDetails</c>, solo se modifica la cantidad del producto adicinando al
8. ///// valor existente la nueva cantidad. 21 ///// </summary>
9. ///// <param name="record">Objeto que contiene los datos del nuevo registro a agregar.</param>
10. //public void AddRecord(OrdersDetails record) 24 //{
11. // var ExistingOrderDetail = records.FirstOrDefault(r => r.ProductId == record.ProductId);
12. // if (ExistingOrderDetail != default) 27 // {
13. // // Con with creamos una copia del objetoque esta a su izquierda pero con
14. // // las propiedades con valores modificado (entre la llaves)
15. // // Es decir agrega a la lista records una copia del objeto ExistingOrderDetail
16. // // pero modificando el valor de la propiedad ProductQuantity, asignandole como valor
17. // // lo que tiene ExistingOrderDetail.ProductQuantity sumado a lo que tiene
18. // // record.ProductQuantity 34

35 // records.Add(ExistingOrderDetail with 36 // {

37 // ProductQuantity = (short) (ExistingOrderDetail.ProductQuantity + record.ProductQuantity)

|  |  |  |
| --- | --- | --- |
| ...ta | - copia\LaPaulita.Sales\Agregates\CreateOrder.cs | 2 |
| 38 | // }); |  |
| 39 | // records.Remove(ExistingOrderDetail); |  |
| 40 | // } |  |
| 41 | // else |  |
| 42 | // { |  |
| 43 | // records.Add(record); |  |
| 44 | // } |  |
| 45 | //} |  |
| 46 | //public void AddRecord(int \_productId, decimal productPrice, |  |

short productQuantity)

47 //{

48 // AddRecord(new OrdersDetails(\_productId, productPrice, productQuantity));

49 //}

50

51 //

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*

1. // Esto si lo hago con una clase que es preparada por nosotros para ser
2. // inmutable.

54 //

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*

55

56

1. // Campo
2. readonly List<OrderDetail> records = new List<OrderDetail>();
3. // Propiedad
4. public IReadOnlyList<OrderDetail> Details => records; 61

62 public void AddDetail(OrderDetail record) 63 {

1. var ExistingOrderDetail = records.FirstOrDefault(r => r.ProductId == record.ProductId);
2. if (ExistingOrderDetail != default)

66 {

1. records.Add(new OrderDetail(record.ProductId, record.ProductPrice, (short)(record.ProductQuantity + ExistingOrderDetail.ProductQuantity)));
2. records.Remove(ExistingOrderDetail); 69 }

70 else

71 {

72 records.Add(record); 73 }

74 }

75 public void AddRecord(int productId, decimal productPrice, short productQuantity)

76 {

77

78 }

AddDetail(new OrderDetail(productId, productPrice, productQuantity));

|  |  |  |
| --- | --- | --- |
| ...ta | - copia\LaPaulita.Sales\Agregates\CreateOrder.cs | 3 |
| 79 | public static CreateOrder From(OrderHeaderDto orderHeaderDto) |  |
| 80 | { |  |
| 81 | // Aqui realizamos el mapeo de las propiedades del DTO con |  |

la Entidad.

82 CreateOrder createOrder = new CreateOrder

83 {

1. ClientId = orderHeaderDto.ClientId,
2. ShippingAddress = orderHeaderDto.ShippingAddress,
3. ShippingCity = orderHeaderDto.ShippingCity,
4. ShippingCountry = orderHeaderDto.ShippingCountry,
5. ShippingZip = orderHeaderDto.ShippingZip 89 };

90

91 foreach (var item in orderHeaderDto.OrderDetails)

92 {

93 createOrder.AddRecord(item.ProductId, item.ProductPrice, item.ProductQuantty);

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 94 |  |  |  | } |
| 95 |  |  |  | return createOrder; |
| 96 |  |  | } |  |
| 97 |  | } |  |  |
| 98 | } |  |  |  |
| 99 |  |  |  |  |

...lita - copia\LaPaulita.Sales\DTOs\OrderDetailDto.cs 1

1 namespace LaPaulita.Sales.BusinessRules.DTOs 2 {

|  |  |  |
| --- | --- | --- |
| 3 | public class OrderDetailDto |  |
| 4 | { |
| 5 | public int ProductId { get; | set; } |
| 6 | public decimal ProductPrice | { get; set; } |
| 7 | public short ProductQuantty | { get; set; } |
| 8 | } |  |
| 9 } |  |  |

...lita - copia\LaPaulita.Sales\DTOs\OrderHeaderDto.cs 1

1 namespace LaPaulita.Sales.BusinessRules.DTOs 2 {

|  |  |  |
| --- | --- | --- |
| 3 |  | public class OrderHeaderDto |
| 4 |  | { |
| 5 |  | public int ClientId { get; set; } |
| 6 |  | public string ShippingAddress { get; set; } |
| 7 |  | public int ShippingCity { get; set; } |
| 8 |  | public int ShippingCountry { get; set; } |
| 9 |  | public string ShippingZip { get; set; } |
| 10 |  |  |
| 11 |  | public List<OrderDetailDto> OrderDetails { get; set; } |
| 12 |  | } |
| 13 | } |  |

...ulita.Sales\DTOs\ValidatorDTO\ValidationErrorDto.cs 1

1 namespace LaPaulita.Sales.BusinessRules.DTOs.ValidatorDTO 2 {

3 public class ValidationErrorDto 4 {

1. public string PropertyName { get; set; }
2. public string ErrorMessage { get; set; } 7 }

8 }

9

...les\Interface\Controllers\ICreateOrderController.cs 1

* 1. using LaPaulita.Sales.BusinessRules.DTOs;
  2. using LaPaulita.Sales.BusinessRules.Wrappers; 3

4 namespace LaPaulita.Sales.BusinessRules.Interface.Controllers 5 {

|  |  |  |
| --- | --- | --- |
| 6 |  | public interface ICreateOrderController |
| 7 |  | { |
| 8 |  | Task<WrappersSalesOrder> CreateOrder(OrderHeaderDto order); |
| 9 |  | } |
| 10 | } |  |
| 11 |  |  |

...ta.Sales\Interface\Getways\ICreateOrderInputPort.cs 1

1 using LaPaulita.Sales.BusinessRules.DTOs; 2

3 namespace LaPaulita.Sales.BusinessRules.Interface.Getways 4 {

|  |  |  |
| --- | --- | --- |
| 5 |  | public interface ICreateOrderInputPort |
| 6 |  | { |
| 7 |  | Task Handle(OrderHeaderDto createOrderDto); |
| 8 |  | } |
| 9 | } |  |
| 10 |  |  |

...a.Sales\Interface\Getways\ICreateOrderOutputPort.cs 1

1 using LaPaulita.Sales.BusinessRules.Wrappers; 2

3 namespace LaPaulita.Sales.BusinessRules.Interface.Getways 4 {

5 public interface ICreateOrderOutputPort 6 {

1. //ValueTask Handle(int orderId);
2. ValueTask Handle(WrappersSalesOrder order);
3. ValueTask ValidationFaild(List<ValidationErrorDto> validationError);

10 }

11 }

12

... - copia\LaPaulita.Sales\ValueObject\OrderDetail.cs 1

1 namespace LaPaulita.Sales.BusinessRules.ValueObject 2 {

3 public class OrderDetail : IEquatable<OrderDetail> 4 {

1. #region "Campos"
2. readonly int \_productId;
3. readonly decimal \_productPrice;
4. readonly short \_productQuantity;
5. #endregion 10
6. #region "Propiedades"
7. //public int ProductId { get { return \_productId; } }
8. public int ProductId => \_productId; //expresión Lambda
9. public decimal ProductPrice => \_productPrice;
10. public short ProductQuantity => \_productQuantity;
11. #endregion 17
12. #region "Cosntructor"
13. public OrderDetail(int productId, decimal productPrice, short productQuantity)

20 {

21

22

23

24 }

\_productId = productId;

\_productPrice = productPrice;

\_productQuantity = productQuantity;

25 #endregion 26

1. #region"Equals and GetHashCode"
2. public override bool Equals(object? obj) 29 {

30 return Equals(obj as OrderDetail); 31 }

32

33 public bool Equals(OrderDetail? other) 34 {

1. //return other is not null &&
2. // ProductId == other.ProductId &&
3. // ProductPrice == other.ProductPrice &&
4. // ProductQuantity == other.ProductQuantity; 39

40 //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

41 // Otra forma usando GetHashCode.

42 //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1. if (other != null) { return GetHashCode() == other.GetHashCode(); }
2. else { return false; } 45 }

46

47 public override int GetHashCode()

48 {

49 return HashCode.Combine(ProductId, ProductPrice, ProductQuantity);

50 }

... - copia\LaPaulita.Sales\ValueObject\OrderDetail.cs 2

51

52

53 }

54

55

#endregion

}

... copia\LaPaulita.Sales\ValueObject\OrdersDetails.cs 1

1 namespace LaPaulita.Sales.BusinessRules.ValueObject 2 {

3 public record class OrdersDetails(int ProductId, decimal ProductPrice, short ProductQuantity);

4 }

5

...opia\LaPaulita.Sales\Wrappers\WrappersSalesOrder.cs 1

1 namespace LaPaulita.Sales.BusinessRules.Wrappers 2 {

3 public class WrappersSalesOrder 4 {

1. public int OrderId { get; set; }
2. public List<ValidationErrorDto> Errors { get; set; } 7 }

8 }

9

...a\LaPaulita.UsesCase\Create\CreateOrderIteractor.cs 1

* 1. using LaPaulita.Sales.BusinessRules.Agregates;
  2. using LaPaulita.Sales.BusinessRules.Interface.Presenters;
  3. using LaPaulita.Sales.BusinessRules.Interface.Repositories;
  4. using LaPaulita.Sales.BusinessRules.Wrappers;
  5. using LaPaulita.UsesCase.Specifications; 6

7 namespace LaPaulita.Sales.UsesCase.Create 8 {

9 /// <summary>

1. /// <b>Use Case Interactor</b>. Es el elemento que contiene el código con la lógica de
2. /// negocios que resuelve un caso de uso. Este elemento implementa la abstracción
3. /// representada por el elemento <i>Use Case Input Port</i>. En términos de programación
4. /// orientada a objetos, el <b>Interactor</b> es una clase que implementa una
5. /// Interface o clase abstracta <i>(InputPort)</i>.
6. /// </summary>
7. public class CreateOrderIteractor : ICreateOrderInputPort 17 {
8. //readonly ICreateOrderOutputPort \_outputPort;
9. readonly ISalesCommandRepository \_repository;
10. readonly ICreateOrderPresenter \_presenter; 21

22 public CreateOrderIteractor(ICreateOrderOutputPort outputPort, ISalesCommandRepository repository, ICreateOrderPresenter presenter)

23 {

24

25

26

27 }

28

//\_outputPort = outputPort;

\_repository = repository;

\_presenter = presenter;

29 public async Task Handle(OrderHeaderDto createOrderDto) 30 {

1. // Instanciamos un objeto del tipo List<ValidationErrorDto> y le asignamos
2. // lo que nos devuelva el método privado ValidateOrder. 33
3. List<ValidationErrorDto> validationErrors = new List<ValidationErrorDto>();
4. validationErrors = ValidateOrder(createOrderDto);
5. WrappersSalesOrder order = new();
6. // Consultamos si la lista validationErrors posee algún elemento.
7. if (validationErrors.Count > 0)

39 {

1. // Si la lista poseía algún elemento, es que hay por lo menos un error
2. // Entonses retormaos el OutputPort al presentador con la lista de errores.

42

...a\LaPaulita.UsesCase\Create\CreateOrderIteractor.cs 2

|  |  |  |
| --- | --- | --- |
| 43 |  | order.Errors = validationErrors; |
| 44 |  | await \_presenter.Handle(order); |
| 45 |  | return; |
| 46 | } |  |
| 47 |  |  |

48

49

50

51

52

53

54

55

56

57

58

59

60

61

62

63

64

65 }

// Si no hay errores contiunuamos con la ejecución del código y

// creamos la orden, luego guardamos los cambios y finalmente retornamos

// el Id del registro creado.

CreateOrder createOrder = CreateOrder.From(createOrderDto); try

{

await \_repository.CreateOrder(createOrder); await \_repository.SaveChange(); order.OrderId = createOrder.Id;

await \_presenter.Handle(order);

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

1. /// <summary>
2. /// Método que valida los datos de la Orden de compra antes de ser presistida
3. /// en la base de datos.

69 /// <br/> <br/>

1. /// <i>Posteriormente deberemos realizar la validación de la existencia de los
2. /// datos que debene existir previamente a la orden de compra, como el id del
3. /// cliente o el id del producto.</i>
4. /// </summary>
5. /// <param name="createOrderDto">Objeto que contiene las propiedades a validar</param>
6. /// <returns>La lista de propiedades que no cumplen con la validación
7. /// y la descripción del error específico.</returns>
8. private List<ValidationErrorDto> ValidateOrder(OrderHeaderDto createOrderDto)

|  |  |  |  |
| --- | --- | --- | --- |
| 78 |  |  | { |
| 79 |  |  | var specification = new OrderHeaderSpecification |
|  |  |  | (createOrderDto); |
| 80 |  |  |  |
| 81 |  |  | return specification.IsValid(); |
| 82 |  |  | } |
| 83 |  | } |  |
| 84 | } |  |  |
| 85 |  |  |  |

...UsesCase\Specifications\OrderHeaderSpecification.cs 1

1 namespace LaPaulita.UsesCase.Specifications 2 {

3 public partial class OrderHeaderSpecification : ISpecification<OrderHeaderDto>

4 {

1. readonly List<ValidationErrorDto> validationErrors = new List<ValidationErrorDto>();
2. readonly OrderHeaderDto entity; 7

8 public OrderHeaderSpecification(OrderHeaderDto entity) 9 {

10 this.entity = entity; 11 }

12

13 public List<ValidationErrorDto> IsValid() 14 {

1. IsClientIdValid();
2. IsAddressValid();
3. IsCityValid();
4. IsCountryValid();
5. IsZipValid(); 20

21 return validationErrors;

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 22 |  |  | } |  | | |
| 23 |  |  |  |
| 24 |  |  | private | partial | void | IsClientIdValid(); |
| 25 |  |  | private | partial | void | IsAddressValid(); |
| 26 |  |  | private | partial | void | IsCityValid(); |
| 27 |  |  | private | partial | void | IsCountryValid(); |
| 28 |  |  | private | partial | void | IsZipValid(); |
| 29 |  | } |  |  |  |  |
| 30 | } |  |  |  |  |  |
| 31 |  |  |  |  |  |  |

...ns\OrderHeaderSpecification\AddressSpecification.cs 1

1 namespace LaPaulita.UsesCase.Specifications 2 {

3 public partial class OrderHeaderSpecification : ISpecification<OrderHeaderDto>



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 4 | { | |  | | |
| 5 |  | | private partial void IsAddressValid() | | |
| 6 |  | | { | | |
| 7 |  | | if (string.IsNullOrEmpty(entity.ShippingAddress)) | | |
| 8 |  | | { | | |
| 9 |  | | validationErrors.Add(new ValidationErrorDto | | |
| 10 |  | | { | | |
| 11 |  | | PropertyName = "ShippingAddress", | | |
| 12 |  | | ErrorMessage = "La dirección de envío es requerida." | | |
| 13 |  | | }); | | |
| 14 |  | | } | | |
| 15 |  | |  | | |
| 16 |  | | if (entity.ShippingAddress.Length > 50) | | |
| 17 |  | | { | | |
| 18 |  | | validationErrors.Add(new ValidationErrorDto | | |
| 19 |  | | { | | |
| 20 |  | | PropertyName = "ShippingAddres", | | |
| 21 |  | | ErrorMessage = "La dirección de envío no puede | | |
| 22 |  |  |  |  | exceder los 50 caracteres."  }); |
| 23 |  |  |  | } |  |
| 24 |  |  | } |  |  |
| 25 |  | } |  |  |  |
| 26 | } |  |  |  |  |
| 27 |  |  |  |  |  |

...tions\OrderHeaderSpecification\CitySpecification.cs 1

1. using LaPaulita.Sales.BusinessRules.DTOs;
2. using LaPaulita.Sales.BusinessRules.Interface; 3

4 namespace LaPaulita.UsesCase.Specifications 5 {

6 public partial class OrderHeaderSpecification : ISpecification<OrderHeaderDto>

# 

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 7 | { | |  | | |
| 8 |  | | private partial void IsCityValid() | | |
| 9 |  | | { | | |
| 10 |  | | if (entity.ShippingCity <= 0) | | |
| 11 |  | | { | | |
| 12 |  | | validationErrors.Add(new ValidationErrorDto | | |
| 13 |  | | { | | |
| 14 |  | | PropertyName = "SippingCity", | | |
| 15 |  | | ErrorMessage = "La ciudad de entrega es | | |
| 16 |  |  |  |  | obligatorio."  }); |
| 17 |  |  |  | } |  |
| 18 |  |  | } |  |  |
| 19 |  | } |  |  |  |
| 20 | } |  |  |  |  |
| 21 |  |  |  |  |  |

...s\OrderHeaderSpecification\ClientIdSpecification.cs 1

1 namespace LaPaulita.UsesCase.Specifications 2 {

3 public partial class OrderHeaderSpecification : ISpecification<OrderHeaderDto>

|  |  |  |  |
| --- | --- | --- | --- |
| 4 |  | { |  |
| 5 |  |  | private partial void IsClientIdValid() |
| 6 |  |  | { |
| 7 |  |  | if (entity.ClientId <= 0) |
| 8 |  |  | { |
| 9 |  |  | validationErrors.Add(new ValidationErrorDto |
| 10 |  |  | { |
| 11 |  |  | PropertyName = "ClientId", |
| 12 |  |  | ErrorMessage = "El Id del cliente es obligatorio." |
| 13 |  |  | }); |
| 14 |  |  | } |
| 15 |  |  | } |
| 16 |  | } |  |
| 17 | } |  |  |
| 18 |  |  |  |

...ons\OrderHeaderSpecification\CountrSpecification.cs 1

1 namespace LaPaulita.UsesCase.Specifications 2 {

3 public partial class OrderHeaderSpecification : ISpecification<OrderHeaderDto>

|  |  |  |  |
| --- | --- | --- | --- |
| 4 |  | { |  |
| 5 |  |  | private partial void IsCountryValid() |
| 6 |  |  | { |
| 7 |  |  | if (entity.ShippingCountry <= 0) |
| 8 |  |  | { |
| 9 |  |  | validationErrors.Add(new ValidationErrorDto |
| 10 |  |  | { |
| 11 |  |  | PropertyName = "SippingCountry", |
| 12 |  |  | ErrorMessage = "El país de entrega es obligatorio." |
| 13 |  |  | }); |
| 14 |  |  | } |
| 15 |  |  | } |
| 16 |  | } |  |
| 17 | } |  |  |
| 18 |  |  |  |

...ations\OrderHeaderSpecification\ZipSpecification.cs 1

1 using System.Text.RegularExpressions; 2

3 namespace LaPaulita.UsesCase.Specifications 4 {

5 public partial class OrderHeaderSpecification : ISpecification<OrderHeaderDto>

6 {

7 private partial void IsZipValid()

8 {

9 if (string.IsNullOrEmpty(entity.ShippingZip))

10 {

11 validationErrors.Add(new ValidationErrorDto

12 {

1. PropertyName = "ShippingZip",
2.  ErrorMessage = "El código postal de entrega es obligatorio."

15 });

16 }

17

18 if (entity.ShippingZip.Length != 4)

19 {

20 validationErrors.Add(new ValidationErrorDto

21 {

1. PropertyName = "ShippingZip",
2.  ErrorMessage = "El código postal de entrega debe contener 4 caracteres."

24 });

25 }

26

1. // Patrón de expresión regular para verificar si solo contiene números
2. string pattern = @"^[0-9]+$"; 29
3. // Verificar si la entrada coincide con el patrón
4. bool isMatch = Regex.IsMatch(entity.ShippingZip, pattern); 32

33 if (!isMatch)

34 {

35 validationErrors.Add(new ValidationErrorDto

36 {

1. PropertyName = "SippingZip",
2.  ErrorMessage = "El código postal de entrega debe contener solo números."

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 39 |  |  |  |  | }); |
| 40 |  |  |  | } |  |
| 41 |  |  | } |  |  |
| 42 |  | } |  | | |
| 43 | } |  |
| 44 |  |  |

...a - copia\LaPaulita.UsesCase\DependencyContainer.cs 1

1 namespace LaPaulita.Sales.UsesCase 2 {

3 public static class DependencyContainer 4 {

5 public static IServiceCollection AddServicesUseCase(this IServiceCollection services)

# 

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 6 |  |  | { |  |
| 7 |  |  |  | services.AddScoped<ICreateOrderInputPort, |
|  |  |  |  | CreateOrderIteractor>(); |
| 8 |  |  |  |  |
| 9 |  |  |  | return services; |
| 10 |  |  | } |  |
| 11 |  | } |  |  |
| 12 | } |  |  |  |
| 13 |  |  |  |  |

...aPaulita.Sales.Controllers\CreateOrderController.cs 1

1 using LaPaulita.Sales.BusinessRules.Wrappers; 2

3 namespace LaPaulita.Sales.Controllers 4 {

5 public class CreateOrderController : ICreateOrderController 6 {

1. readonly ICreateOrderInputPort \_inputPort;
2. readonly ICreateOrderPresenter \_presenter; 9

10 //public CreateOrderController(ICreateOrderInputPort inputPort, ICreateOrderPresenter presenter)

11 //{

1. // \_inputPort = inputPort;
2. // \_presenter = presenter;

14 //}

15 public CreateOrderController(ICreateOrderInputPort inputPort, ICreateOrderPresenter presenter) => (\_presenter, \_inputPort) = (presenter, inputPort); //Expresión lamda.

16

17 public async Task<WrappersSalesOrder> CreateOrder(OrderHeaderDto order)

|  |  |  |  |
| --- | --- | --- | --- |
| 18 |  |  | { |
| 19 |  |  | await \_inputPort.Handle(order); |
| 20 |  |  | return \_presenter.Order; |
| 21 |  |  |  |
| 22 |  |  | } |
| 23 |  | } |  |
| 24 | } |  |  |
| 25 |  |  |  |

...\LaPaulita.Sales.Controllers\DependencyContiners.cs 1

1 namespace LaPaulita.Sales.Controllers 2 {

3 public static class DependencyContiners 4 {

1. public static IServiceCollection AddServicesSalesControllers(
2. this IServiceCollection services) 7 {
3. services.AddScoped<ICreateOrderController, CreateOrderController>();
4. return services; 10 }

11 }

12 }

13

... - copia\DependencyInversion\DependencyContainer.cs 1

1

2 namespace DependencyInversion 3 {

4 public static class DependencyContainer 5 {

6 public static IServiceCollection AddLaPaulitaSalesServices(this IServiceCollection services, IConfiguration configuration, string ConnectionString)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 7 |  |  | { |  |
| 8 |  |  |  | services |
| 9 |  |  |  | .AddRepositories(configuration, ConnectionString) |
| 10 |  |  |  | .AddServicesUseCase() |
| 11 |  |  |  | .AddServicesPresenter() |
| 12 |  |  |  | .AddServicesSalesControllers(); |
| 13 |  |  |  |  |
| 14 |  |  |  | return services; |
| 15 |  |  | } |  |
| 16 |  | } |  |  |
| 17 | } |  |  |  |
| 18 |  |  |  |  |

...y\EFCore\Configurations\OrderDetailConfiguration.cs 1

1 namespace LaPaulita.Sales.Repositories.EFCore.Configurations 2 {

3 public class OrderDetailConfiguration : IEntityTypeConfiguration<OrderDetail>

4

5

6

7

8

9

10

11

12

13

14 }

15

{

public void Configure(EntityTypeBuilder<OrderDetail> builder)

{

builder.HasKey(d => new { d.Id, d.ProductId }); builder.Property(d => d.Id)

.UseIdentityColumn(); builder.Property(d => d.ProductPrice)

.HasPrecision(8, 2);

}

}

...ory\EFCore\Configurations\OrderHeaderConfiguration.cs 1

1 namespace LaPaulita.Sales.Repositories.EFCore.Configurations 2 {

3 public class OrderHeaderConfiguration : IEntityTypeConfiguration<OrderHeader>

4 {

5 public void Configure(EntityTypeBuilder<OrderHeader> builder) 6 {

1. builder.Property(o => o.ClientId)
2. .IsRequired()
3. .HasMaxLength(5); 10
4. builder.Property(o => o.ShippingAddress)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 12 |  |  |  | .IsRequired() |  |
| 13 |  |  |  | .HasMaxLength(50); |
| 14 |  |  |  |  |
| 15 |  |  |  | builder.Property(o => | o.ShippingCity) |
| 16 |  |  |  | .IsRequired(); |  |
| 17 |  |  |  |  |  |
| 18 |  |  |  | builder.Property(o => | o.ShippingCountry) |
| 19 |  |  |  | .IsRequired(); |  |
| 20 |  |  |  |  |  |
| 21 |  |  |  | builder.Property(o => | o.ShippingZip) |
| 22 |  |  |  | .HasMaxLength(4); |  |
| 23 |  |  | } |  |  |
| 24 |  | } |  |  |  |
| 25 | } |  |  |  |  |
| 26 |  |  |  |  |  |

...Sales.Repository\EFCore\Context\LaPaulitaContext.cs 1

1 namespace LaPaulita.Sales.Repositories.EFCore.Context 2 {

3 internal class LaPaulitaContext : DbContext 4 {

5 protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)

6 {

7

8

9

10 }

11

optionsBuilder.UseSqlServer("Data Source=DanielPagano; Initial Catalog=LaPaulitaDB; User ID=sa; Password=MsSqlServer; TrustServerCertificate=True");

//"Data Source=(LocalDb)\\MSSQLLocalDB;Initial Catalog=LaPaulitaDB;Integrated Security=SSPI;"

1. public DbSet<OrderHeader> OrderHeaders { get; set; }
2. public DbSet<OrderDetail> OrderDetails { get; set; } 14

15 protected override void OnModelCreating(ModelBuilder modelBuilder)

16 {

17

modelBuilder.ApplyConfigurationsFromAssembly(

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 18 |  |  |  | Assembly.GetExecutingAssembly()); |
| 19 |  |  | } |  |
| 20 |  | } |  |  |
| 21 | } |  |  |  |
| 22 |  |  |  |  |

....Repository\EFCore\Context\LaPaulitaSalesContext.cs 1

1 namespace LaPaulita.Sales.Repositories.EFCore.Context 2 {

3 public class LaPaulitaSalesContext : DbContext 4 {

1. public LaPaulitaSalesContext (DbContextOptions<LaPaulitaSalesContext> options)
2. : base(options) { } 7
3. public DbSet<OrderHeader> OrderHeaders { get; set; }
4. public DbSet<OrderDetail> OrderDetails { get; set; } 10
5. protected override void OnModelCreating(ModelBuilder modelBuilder)

12 {

13

modelBuilder.ApplyConfigurationsFromAssembly(

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 14 |  |  |  | Assembly.GetExecutingAssembly()); |
| 15 |  |  | } |  |
| 16 |  | } |  |  |
| 17 | } |  |  |  |
| 18 |  |  |  |  |

...a\LaPaulita.Sales.Repository\EFCore\Migraciones.txt 1

* 1. Crear la migración
  2. Add-Migration InitialCreate -p LaPaulita.Sales.Repositories -s LaPaulita.Sales.Repositories -c LaPaulitaContext

3

1. Actualización de la DB
2. Update-Database -p LaPaulita.Sales.Repositories -s LaPaulita.Sales.Repositories -context LaPaulitaContext

6

7

1. Cadena de conexión
2. Data Source=(LocalDb)\\MSSQLLocalDB;Initial Catalog=LaPaulitaDB;Integrated Security=SSPI;

10

11 {

1. "clientId": 1,
2. "shippingAddress": "La Pampa 132",
3. "shippingCity": 1,
4. "shippingCountry": 1,
5. "shippingZip": "5620",
6. "orderDetails": [ 18 {
7. "productId": 1,
8. "productPrice": 175.26,
9. "productQuantty": 1 22 },

23 {

1. "productId": 1,
2. "productPrice": 175.26,
3. "productQuantty": 5 27 },

28 {

1. "productId": 2,
2. "productPrice": 375.26,
3. "productQuantty": 1 32 }

33 ]

34 }

...s.Repository\Repositories\SalesCommandRepository.cs 1

1 namespace LaPaulita.Sales.Repositories.Repositories 2 {

3 public class SalesCommandRepository : ISalesCommandRepository 4 {

5 readonly LaPaulitaSalesContext \_context; 6

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 7 |  |  | public SalesCommandRepository(LaPaulitaSalesContext | context) |
| 8 |  |  | { |  |
| 9 |  |  | \_context = context; |  |
| 10 |  |  | } |  |
| 11 |  |  |  |  |
| 12 |  |  | public async Task CreateOrder(CreateOrder order) |  |
| 13 |  |  | { |  |
| 14 |  |  | await \_context.AddAsync(order); |  |
| 15 |  |  | foreach (var item in order.Details) |  |
| 16 |  |  | { |  |
| 17 |  |  | await \_context.AddAsync(new OrderDetail |  |
| 18 |  |  | { |  |
| 19 |  |  | Order = order, |  |
| 20 |  |  | ProductId = item.ProductId, |  |
| 21 |  |  | ProductPrice = item.ProductPrice, |  |
| 22 |  |  | ProductQuantity = item.ProductQuantity |  |
| 23 |  |  | }); |  |
| 24 |  |  | } |  |
| 25 |  |  | } |  |
| 26 |  |  |  |  |
| 27 |  |  | public async Task SaveChange() |  |
| 28 |  |  | { |  |
| 29 |  |  | await \_context.SaveChangesAsync(); |  |
| 30 |  |  | } |  |
| 31 |  | } |  |  |
| 32 | } |  |  |  |
| 33 |  |  |  |  |

...a\LaPaulita.Sales.Repository\DependencyContainer.cs 1

|  |  |  |
| --- | --- | --- |
| 1 | namespace LaPaulita.Sales.Repositories | |
| 2 | { | |
| 3 |  | public static class DependencyContainer |
| 4 |  | { |
| 5 |  | public static IServiceCollection AddRepositories( |
| 6 |  | this IServiceCollection services, |
| 7 |  | IConfiguration configuration, |
| 8 |  | string connectionStringName) |
| 9 |  | { |
| 10 |  | services.AddDbContext<LaPaulitaSalesContext>(options => |
| 11 |  | options.UseSqlServer(configuration |
| 12 |  | .GetConnectionString(connectionStringName))); |
| 14 |  | services.AddScoped<ISalesCommandRepository, |
| 15 |  | SalesCommandRepository>(); |
| 16 |  |  |
| 17 |  | return services; |
| 18 |  | } |
| 19 |  | } |
| 20 | } |  |
| 21 |  |  |

13

...\LaPaulita.Sales.Presenters\CreateOrderPresenter.cs 1

1 using LaPaulita.Sales.BusinessRules.Wrappers; 2

3 namespace LaPaulita.Sales.Presenters 4 {

5 public class CreateOrderPresenter : ICreateOrderPresenter 6 {

7 //public int OrderId { get; private set; } 8

9 //public List<ValidationErrorDto> ErrorsList { get; private set; }

10

11 public WrappersSalesOrder Order { get; private set; } = new WrappersSalesOrder();

12

13 //public ValueTask Handle(int orderId) 14 //{

1. // OrderId = orderId;
2. // return ValueTask.CompletedTask; 17 //}

18

19 public ValueTask Handle(WrappersSalesOrder order) 20 {

1. Order.OrderId = order.OrderId;
2. Order.Errors = order.Errors;
3. return ValueTask.CompletedTask; 24 }

25

26 public ValueTask ValidationFaild(List<ValidationErrorDto> validationError)

|  |  |  |  |
| --- | --- | --- | --- |
| 27 |  |  | { |
| 28 |  |  | Order.Errors = validationError; |
| 29 |  |  | return ValueTask.CompletedTask; |
| 30 |  |  | } |
| 31 |  | } |  |
| 32 | } |  |  |
| 33 |  |  |  |

...a\LaPaulita.Sales.Presenters\DependencyContainer.cs 1

1 namespace LaPaulita.Sales.Presenters 2 {

3 public static class DependencyContainer 4 {

5 public static IServiceCollection AddServicesPresenter(this IServiceCollection services)

6

7

8

9

10

11

12

13

14

15

16 }

17

{

services.AddScoped<CreateOrderPresenter>();

services.AddScoped<ICreateOrderOutputPort, CreateOrderPresenter>();

services.AddScoped<ICreateOrderPresenter, CreateOrderPresenter>();

return services;

}

}

...Paulita - copia\LaPaulita.Sales.WebApi\EndPoints.cs 1

1. using LaPaulita.Sales.BusinessRules.DTOs;
2. using LaPaulita.Sales.BusinessRules.Interface.Controllers; 3

4 namespace LaPaulita.Sales.WebApi 5 {

6 public static class EndPoints 7 {

8 public static WebApplication LaPaulitaSalesEndPoint(this WebApplication app)

9 {

10 app.MapPost("/create", async (OrderHeaderDto order, ICreateOrderController controller) =>

13

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 11 |  |  |  | { |  |
| 12 |  |  |  | var result = await controller.CreateOrder(order); |
| 14 |  |  |  | if (result.Errors != null && result.Errors.Count > | 0) |
| 15 |  |  |  | { |  |
| 16 |  |  |  | return Results.BadRequest(result); |  |
| 17 |  |  |  | } |  |
| 18 |  |  |  | else |  |
| 19 |  |  |  | { |  |
| 20 |  |  |  | return Results.Ok(result); |  |
| 21 |  |  |  | } |  |
| 22 |  |  |  |  |  |
| 23 |  |  |  | }); |  |
| 24 |  |  |  |  |  |
| 25 |  |  |  | return app; |  |
| 26 |  |  | } |  |  |
| 27 |  | } |  |  |  |
| 28 | } |  |  |  |  |
| 29 |  |  |  |  |  |

...opia\LaPaulita.Sales.WebApi\WebApplicationHelper.cs 1

1 using DependencyInversion; 2

3 namespace LaPaulita.Sales.WebApi 4 {

5 public static class WebApplicationHelper 6 {

7 public static WebApplication CreateWebApplication(this WebApplicationBuilder builder)

8 {

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

32

33

34 }

// Configurar APIExplorer para descubrir y exponer

// los metadatos de los endpoints de la aplicación. builder.Services.AddEndpointsApiExplorer();

// Agregar el generador que construye los objetos de

// documentación de Swagger con la funcionalidad del APIExplorer.

builder.Services.AddSwaggerGen();

// Registrar los servicios de la aplicación builder.Services.AddLaPaulitaSalesServices(

builder.Configuration, "LaPaulitaDB");

// Agregar el servicio CORS para clientes que se ejecutan

// en el navegador Web (como Blazor WebAssembly). builder.Services.AddCors(options =>

{

options.AddDefaultPolicy(config =>

{

config.AllowAnyMethod(); config.AllowAnyHeader(); config.AllowAnyOrigin();

});

});

return builder.Build();

1. public static WebApplication ConfigureWebApplication(
2. this WebApplication app) 37 {
3. // Habilitar el middleware para servir el documento
4. // JSON generado y la interfaz UI de Swagger en el
5. // ambiente de desarrollo.
6. if (app.Environment.IsDevelopment())

42 {

1. app.UseSwagger();
2. app.UseSwaggerUI(); 45 }

46

1. // Registrar los endpoints de la aplicación
2. app.LaPaulitaSalesEndPoint(); 49
3. // Agregar el Middleware CORS
4. app.UseCors();

...opia\LaPaulita.Sales.WebApi\WebApplicationHelper.cs 2

52

53

54

55

56 }

57

return app;

}

}

...LaPaulita - copia\LaPaulita.Sales.WebApi\Program.cs 1

1 using LaPaulita.Sales.WebApi; 2

1. var builder = WebApplication.CreateBuilder(args)
2. .CreateWebApplication()
3. .ConfigureWebApplication();
4. builder.Run();
5. //var app = builder.Build(); 8

9 //app.MapGet("/", () => "Hello World!"); 10

11 //app.Run(); 12

...ita - copia\LaPaulita.Sales.WebApi\appsettings.json 1

1 {

2

3

4

5

6

7

8

9

10 }

11

"Logging": {

"LogLevel": {

"Default": "Information", "Microsoft.AspNetCore": "Warning"

}

},

"AllowedHosts": "\*",

"ConnectionStrings": { "LaPaulitaDB": "Data Source=DanielPagano; Initial Catalog=LaPaulitaDB; User ID=sa; Password=MsSqlServer; TrustServerCertificate=True" }