trabalhoPOO_27967_Fase2

Generated by Doxygen 1.12.0

1 Namespace Index 1

1.1 Package List . . . 1

2 Hierarchical Index 3

2.1 Class Hierarchy . . . 3

3 Class Index 5

3.1 Class List . . . 5

4 File Index 9

4.1 File List . . . 9

5 Namespace Documentation 11

5.1 BestSale Namespace Reference . . . 11

5.2 BestSale_Validations Namespace Reference . . . 11

5.3 Business_Layer Namespace Reference . . . 11

5.4 Business_Object Namespace Reference . . . 11

5.5 Data_BestSale Namespace Reference . . . 12

5.6 Exceptions Namespace Reference . . . 13

5.7 trabalhoPOO_27967 Namespace Reference . . . 13

5.8 trabalhoPOO_27967.Interface Namespace Reference . . . 14

5.9 trabalhoPOO_27967.Store Namespace Reference . . . 14

6 Class Documentation 15

6.1 BestSale.BestSale Class Reference . . . 15

6.1.1 Detailed Description . . . 15

6.2 Data_BestSale.Campaign Class Reference . . . 15

6.2.1 Detailed Description . . . 16

6.2.2 Constructor & Destructor Documentation . . . 16

6.2.2.1 Campaign() [1/2] . . . 16

6.2.2.2 Campaign() [2/2] . . . 16

6.2.3 Member Function Documentation . . . 17

6.2.3.1 Equals() . . . 17

6.2.3.2 operator"!=() . . . 17

6.2.3.3 operator==() . . . 17

6.2.3.4 VerifyApplicability() . . . 18

6.2.4 Property Documentation . . . 18

6.2.4.1 CampaignCount . . . 18

6.2.4.2 Discount . . . 18

6.2.4.3 EndDate . . . 19

6.2.4.4 Id . . . 19

6.2.4.5 Name . . . 19

6.2.4.6 StartDate . . . 19

6.3 trabalhoPOO_27967.Campaign Class Reference . . . 19

6.3.1 Detailed Description . . . 20
6.3.2 Constructor & Destructor Documentation . . . 20
6.3.2.1 Campaign() [1/2] . . . 20
6.3.2.2 Campaign() [2/2] . . . 20
6.3.3 Member Function Documentation . . . 21
6.3.3.1 Equals() . . . 21
6.3.3.2 operator"!=() . . . 21
6.3.3.3 operator==() . . . 21
6.3.3.4 VerifyApplicability() . . . 22
6.3.4 Property Documentation . . . 22
6.3.4.1 CampaignCount . . . 22
6.3.4.2 Discount . . . 22
6.3.4.3 EndDate . . . 23
6.3.4.4 Id . . . 23
6.3.4.5 Name . . . 23
6.3.4.6 StartDate . . . 23
6.4 Data_BestSale.Campaigns Class Reference . . . 23
6.5 trabalhoPOO_27967.Campaigns Class Reference . . . 24
6.5.1 Detailed Description . . . 24
6.5.2 Constructor & Destructor Documentation . . . 24
6.5.2.1 Campaigns() [1/2] . . . 24
6.5.2.2 Campaigns() [2/2] . . . 24
6.5.3 Member Function Documentation . . . 25
6.5.3.1 Add() . . . 25
6.5.3.2 Exist() . . . 25
6.5.3.3 Remove() . . . 25
6.5.4 Property Documentation . . . 26
6.5.4.1 Camps . . . 26
6.6 Data_BestSale.Categories Class Reference . . . 26
6.6.1 Detailed Description . . . 27
6.6.2 Constructor & Destructor Documentation . . . 27
6.6.2.1 Categories() [1/2] . . . 27
6.6.2.2 Categories() [2/2] . . . 27
6.6.3 Member Function Documentation . . . 27
6.6.3.1 Add() . . . 27
6.6.3.2 ClearCategories() . . . 28
6.6.3.3 Exist() . . . 28
6.6.3.4 GetCategory() . . . 28
6.6.3.5 Remove() . . . 29
6.6.4 Property Documentation . . . 29
6.6.4.1 Cats . . . 29
6.7 trabalhoPOO_27967.Categories Class Reference . . . 29

6.7.1 Detailed Description . . . 30
6.7.2 Constructor & Destructor Documentation . . . 30
6.7.2.1 Categories() [1/2] . . . 30
6.7.2.2 Categories() [2/2] . . . 30
6.7.3 Member Function Documentation . . . 30
6.7.3.1 Add() . . . 30
6.7.3.2 Exist() . . . 31
6.7.3.3 Remove() . . . 31
6.7.4 Property Documentation . . . 32
6.7.4.1 Cats . . . 32
6.8 Data_BestSale.Category Class Reference . . . 32
6.8.1 Detailed Description . . . 33
6.8.2 Constructor & Destructor Documentation . . . 33
6.8.2.1 Category() [1/2] . . . 33
6.8.2.2 Category() [2/2] . . . 33
6.8.3 Member Function Documentation . . . 33
6.8.3.1 CreateCategory() . . . 33
6.8.3.2 Equals() . . . 34
6.8.3.3 operator"!=() . . . 35
6.8.3.4 operator==() . . . 35
6.8.4 Property Documentation . . . 36
6.8.4.1 Id . . . 36
6.8.4.2 Name . . . 36
6.9 trabalhoPOO_27967.Category Class Reference . . . 36
6.9.1 Detailed Description . . . 37
6.9.2 Constructor & Destructor Documentation . . . 37
6.9.2.1 Category() [1/2] . . . 37
6.9.2.2 Category() [2/2] . . . 37
6.9.3 Member Function Documentation . . . 37
6.9.3.1 Equals() . . . 37
6.9.3.2 operator"!=() . . . 37
6.9.3.3 operator==() . . . 38
6.9.4 Property Documentation . . . 38
6.9.4.1 Id . . . 38
6.9.4.2 Name . . . 38
6.10 Data_BestSale.Client Class Reference . . . 39
6.10.1 Detailed Description . . . 39
6.10.2 Constructor & Destructor Documentation . . . 39
6.10.2.1 Client() [1/2] . . . 39
6.10.2.2 Client() [2/2] . . . 40
6.10.3 Member Function Documentation . . . 40
6.10.3.1 CreateClientFromNameContact() . . . 40

6.10.3.2 Equals() . . . 40
6.10.3.3 operator"!=() . . . 40
6.10.3.4 operator==() . . . 41
6.10.3.5 ToString() . . . 41
6.10.4 Property Documentation . . . 41
6.10.4.1 ClientCount . . . 41
6.10.4.2 ClientID . . . 42
6.10.4.3 Contact . . . 42
6.10.4.4 Name . . . 42
6.11 trabalhoPOO_27967.Client Class Reference . . . 42
6.11.1 Detailed Description . . . 43
6.11.2 Constructor & Destructor Documentation . . . 43
6.11.2.1 Client() [1/2] . . . 43
6.11.2.2 Client() [2/2] . . . 43
6.11.3 Member Function Documentation . . . 43
6.11.3.1 Equals() . . . 43
6.11.3.2 operator"!=() . . . 44
6.11.3.3 operator==() . . . 44
6.11.3.4 ToString() . . . 45
6.11.4 Property Documentation . . . 45
6.11.4.1 ClientCount . . . 45
6.11.4.2 ClientID . . . 45
6.11.4.3 Contact . . . 45
6.11.4.4 Name . . . 45
6.12 Data_BestSale.Clients Class Reference . . . 46
6.12.1 Detailed Description . . . 46
6.12.2 Constructor & Destructor Documentation . . . 46
6.12.2.1 Clients() . . . 46
6.12.3 Member Function Documentation . . . 46
6.12.3.1 Add() . . . 46
6.12.3.2 ClearClients() . . . 47
6.12.3.3 Exist() . . . 47
6.12.3.4 GetClient() . . . 47
6.12.3.5 Remove() . . . 48
6.12.4 Property Documentation . . . 48
6.12.4.1 ClientList . . . 48
6.13 trabalhoPOO_27967.Clients Class Reference . . . 48
6.13.1 Detailed Description . . . 49
6.13.2 Constructor & Destructor Documentation . . . 49
6.13.2.1 Clients() . . . 49
6.13.3 Member Function Documentation . . . 49
6.13.3.1 Add() . . . 49

6.13.3.2 Exist() . . . 50
6.13.3.3 GetClient() . . . 50
6.13.3.4 Remove() . . . 50
6.13.4 Property Documentation . . . 51
6.13.4.1 ClientList . . . 51
6.14 Data_BestSale.IListManagement Interface Reference . . . 51
6.14.1 Detailed Description . . . 51
6.14.2 Member Function Documentation . . . 51
6.14.2.1 Add() . . . 51
6.14.2.2 Exist() . . . 52
6.14.2.3 Remove() . . . 52
6.15 trabalhoPOO_27967.Interface.IListManagement Interface Reference . . . 52
6.15.1 Detailed Description . . . 52
6.15.2 Member Function Documentation . . . 52
6.15.2.1 Add() . . . 52
6.15.2.2 Exist() . . . 53
6.15.2.3 Remove() . . . 53
6.16 Exceptions.InvalidPhoneNumberException Class Reference . . . 53
6.16.1 Detailed Description . . . 53
6.16.2 Constructor & Destructor Documentation . . . 53
6.16.2.1 InvalidPhoneNumberException() [1/3] . . . 53
6.16.2.2 InvalidPhoneNumberException() [2/3] . . . 54
6.16.2.3 InvalidPhoneNumberException() [3/3] . . . 54
6.17 Data_BestSale.Make Class Reference . . . 54
6.17.1 Detailed Description . . . 55
6.17.2 Constructor & Destructor Documentation . . . 55
6.17.2.1 Make() [1/2] . . . 55
6.17.2.2 Make() [2/2] . . . 55
6.17.3 Member Function Documentation . . . 55
6.17.3.1 CreateMake() . . . 55
6.17.3.2 Equals() . . . 56
6.17.3.3 GetMakeID() . . . 56
6.17.3.4 operator"!=() . . . 56
6.17.3.5 operator==() . . . 57
6.17.3.6 ToString() . . . 57
6.17.4 Property Documentation . . . 57
6.17.4.1 ID . . . 57
6.17.4.2 Name . . . 58
6.18 trabalhoPOO_27967.Make Class Reference . . . 58
6.18.1 Detailed Description . . . 58
6.18.2 Constructor & Destructor Documentation . . . 59
6.18.2.1 Make() [1/2] . . . 59

6.18.2.2 Make() [2/2] . . . 59
6.18.3 Member Function Documentation . . . 59
6.18.3.1 Equals() . . . 59
6.18.3.2 operator"!=() . . . 59
6.18.3.3 operator==() . . . 60
6.18.3.4 ToString() . . . 60
6.18.4 Property Documentation . . . 60
6.18.4.1 ID . . . 60
6.18.4.2 Name . . . 61
6.19 Data_BestSale.Makes Class Reference . . . 61
6.19.1 Detailed Description . . . 62
6.19.2 Constructor & Destructor Documentation . . . 62
6.19.2.1 Makes() [1/2] . . . 62
6.19.2.2 Makes() [2/2] . . . 62
6.19.3 Member Function Documentation . . . 62
6.19.3.1 Add() . . . 62
6.19.3.2 ClearMakes() . . . 63
6.19.3.3 Exist() . . . 63
6.19.3.4 GetMake() . . . 63
6.19.3.5 Remove() . . . 63
6.19.4 Property Documentation . . . 64
6.19.4.1 MakeList . . . 64
6.20 trabalhoPOO_27967.Makes Class Reference . . . 64
6.20.1 Detailed Description . . . 65
6.20.2 Constructor & Destructor Documentation . . . 65
6.20.2.1 Makes() [1/2] . . . 65
6.20.2.2 Makes() [2/2] . . . 65
6.20.3 Member Function Documentation . . . 65
6.20.3.1 Add() . . . 65
6.20.3.2 Exist() . . . 66
6.20.3.3 Remove() . . . 66
6.20.4 Property Documentation . . . 67
6.20.4.1 MakeList . . . 67
6.21 Data_BestSale.Product Class Reference . . . 67
6.21.1 Detailed Description . . . 68
6.21.2 Constructor & Destructor Documentation . . . 68
6.21.2.1 Product() [1/3] . . . 68
6.21.2.2 Product() [2/3] . . . 68
6.21.2.3 Product() [3/3] . . . 69
6.21.3 Member Function Documentation . . . 69
6.21.3.1 CreateProductWithWarranty() . . . 69
6.21.3.2 Equals() . . . 69

6.21.3.3 operator"!=() . . . 70
6.21.3.4 operator==() . . . 70
6.21.3.5 ToString() . . . 71
6.21.4 Property Documentation . . . 71
6.21.4.1 CategoryID . . . 71
6.21.4.2 MakeID . . . 71
6.21.4.3 Price . . . 71
6.21.4.4 Reference . . . 71
6.21.4.5 Stock . . . 72
6.21.4.6 Warranty . . . 72
6.22 trabalhoPOO_27967.Product Class Reference . . . 72
6.22.1 Detailed Description . . . 73
6.22.2 Constructor & Destructor Documentation . . . 73
6.22.2.1 Product() [1/2] . . . 73
6.22.2.2 Product() [2/2] . . . 73
6.22.3 Member Function Documentation . . . 73
6.22.3.1 Equals() . . . 73
6.22.3.2 operator"!=() . . . 74
6.22.3.3 operator==() . . . 74
6.22.3.4 ToString() . . . 75
6.22.4 Property Documentation . . . 75
6.22.4.1 Category . . . 75
6.22.4.2 Make . . . 75
6.22.4.3 Price . . . 75
6.22.4.4 Reference . . . 75
6.22.4.5 Stock . . . 76
6.22.4.6 Warranty . . . 76
6.23 Data_BestSale.Products Class Reference . . . 76
6.23.1 Detailed Description . . . 77
6.23.2 Constructor & Destructor Documentation . . . 77
6.23.2.1 Products() [1/2] . . . 77
6.23.2.2 Products() [2/2] . . . 77
6.23.3 Member Function Documentation . . . 77
6.23.3.1 Add() . . . 77
6.23.3.2 ClearProducts() . . . 78
6.23.3.3 Exist() . . . 78
6.23.3.4 Remove() . . . 78
6.23.3.5 SearchProduct() . . . 79
6.23.3.6 ToString() . . . 79
6.23.3.7 TotalPrice() . . . 80
6.23.3.8 ValueInPosition() . . . 80
6.23.3.9 WarratyExpirationDateForProduct() . . . 81

6.23.4 Property Documentation . . . 81
6.23.4.1 Prods . . . 81
6.24 trabalhoPOO_27967.Products Class Reference . . . 81
6.24.1 Detailed Description . . . 82
6.24.2 Constructor & Destructor Documentation . . . 82
6.24.2.1 Products() [1/2] . . . 82
6.24.2.2 Products() [2/2] . . . 82
6.24.3 Member Function Documentation . . . 83
6.24.3.1 Add() . . . 83
6.24.3.2 Exist() . . . 83
6.24.3.3 Remove() . . . 83
6.24.3.4 SearchProduct() . . . 84
6.24.3.5 ToString() . . . 84
6.24.3.6 TotalPrice() . . . 84
6.24.3.7 ValueInPosition() . . . 84
6.24.3.8 WarratyExpirationDateForProduct() . . . 85
6.24.4 Property Documentation . . . 85
6.24.4.1 Prods . . . 85
6.25 Data_BestSale.Sale Class Reference . . . 85
6.25.1 Detailed Description . . . 86
6.25.2 Constructor & Destructor Documentation . . . 87
6.25.2.1 Sale() [1/2] . . . 87
6.25.2.2 Sale() [2/2] . . . 87
6.25.3 Member Function Documentation . . . 87
6.25.3.1 Equals() . . . 87
6.25.3.2 ExistProductOnSale() . . . 87
6.25.3.3 InsertProductOnSale() . . . 88
6.25.3.4 operator"!=() . . . 88
6.25.3.5 operator==() . . . 88
6.25.3.6 RemoveProductFromSale() . . . 89
6.25.3.7 ToString() . . . 89
6.25.3.8 TotalPrice() . . . 89
6.25.3.9 WarrantyExpirationDate() . . . 89
6.25.4 Property Documentation . . . 90
6.25.4.1 Campaigns . . . 90
6.25.4.2 Client . . . 90
6.25.4.3 Id . . . 90
6.25.4.4 Products . . . 90
6.25.4.5 SaleDate . . . 91
6.25.4.6 TotPrice . . . 91
6.26 trabalhoPOO_27967.Sale Class Reference . . . 91
6.26.1 Detailed Description . . . 92

6.26.2 Constructor & Destructor Documentation . . . 92
6.26.2.1 Sale() [1/2] . . . 92
6.26.2.2 Sale() [2/2] . . . 92
6.26.3 Member Function Documentation . . . 93
6.26.3.1 Equals() . . . 93
6.26.3.2 ExistProductOnSale() . . . 93
6.26.3.3 InsertProductOnSale() . . . 93
6.26.3.4 operator"!=() . . . 94
6.26.3.5 operator==() . . . 94
6.26.3.6 RemoveProductFromSale() . . . 94
6.26.3.7 ToString() . . . 95
6.26.3.8 TotalPrice() . . . 95
6.26.3.9 WarrantyExpirationDate() . . . 95
6.26.4 Property Documentation . . . 96
6.26.4.1 Campaigns . . . 96
6.26.4.2 Client . . . 96
6.26.4.3 Id . . . 96
6.26.4.4 Products . . . 96
6.26.4.5 SaleDate . . . 96
6.26.4.6 TotPrice . . . 96
6.27 Data_BestSale.Sales Class Reference . . . 97
6.27.1 Detailed Description . . . 97
6.27.2 Constructor & Destructor Documentation . . . 97
6.27.2.1 Sales() [1/2] . . . 97
6.27.2.2 Sales() [2/2] . . . 97
6.27.3 Member Function Documentation . . . 98
6.27.3.1 Add() . . . 98
6.27.3.2 ClearSales() . . . 98
6.27.3.3 Exist() . . . 98
6.27.3.4 GetSale() . . . 98
6.27.3.5 Remove() . . . 99
6.27.4 Property Documentation . . . 99
6.27.4.1 SalesStored . . . 99
6.28 trabalhoPOO_27967.Sales Class Reference . . . 99
6.28.1 Detailed Description . . . 100
6.28.2 Constructor & Destructor Documentation . . . 100
6.28.2.1 Sales() [1/2] . . . 100
6.28.2.2 Sales() [2/2] . . . 100
6.28.3 Member Function Documentation . . . 101
6.28.3.1 Add() . . . 101
6.28.3.2 Exist() . . . 101
6.28.3.3 GetSale() . . . 101

6.28.3.4 Remove() . . . 102
6.28.4 Property Documentation . . . 102
6.28.4.1 SalesStored . . . 102
6.29 Business_Object.SimpleProduct Class Reference . . . 102
6.29.1 Detailed Description . . . 103
6.29.2 Constructor & Destructor Documentation . . . 103
6.29.2.1 SimpleProduct() [1/2] . . . 103
6.29.2.2 SimpleProduct() [2/2] . . . 103
6.29.3 Property Documentation . . . 104
6.29.3.1 Make . . . 104
6.29.3.2 Price . . . 104
6.29.3.3 Reference . . . 104
6.30 Data_BestSale.Store Class Reference . . . 104
6.30.1 Detailed Description . . . 105
6.30.2 Constructor & Destructor Documentation . . . 105
6.30.2.1 Store() [1/2] . . . 105
6.30.2.2 Store() [2/2] . . . 106
6.30.3 Member Function Documentation . . . 106
6.30.3.1 ClearStore() . . . 106
6.30.3.2 GetCategoryIdFromNameInStore() . . . 106
6.30.3.3 GetMakeIdFromNameInStore() . . . 106
6.30.3.4 GetMakeNameFromID() . . . 107
6.30.3.5 GetProductPriceInStoreFromReference() . . . 107
6.30.3.6 InsertCategoryInStore() . . . 107
6.30.3.7 InsertClientInStore() . . . 108
6.30.3.8 InsertMakeInStore() . . . 108
6.30.3.9 InsertProductInStore() . . . 108
6.30.3.10 LoadStoreBin() . . . 109
6.30.3.11 SaveStoreBin() . . . 109
6.30.4 Property Documentation . . . 110
6.30.4.1 CatList . . . 110
6.30.4.2 ClientLIst . . . 110
6.30.4.3 MakeList . . . 110
6.30.4.4 ProdList . . . 110
6.30.4.5 SaleList . . . 110
6.31 trabalhoPOO_27967.Store.Store Class Reference . . . 111
6.31.1 Detailed Description . . . 111
6.31.2 Constructor & Destructor Documentation . . . 111
6.31.2.1 Store() [1/2] . . . 111
6.31.2.2 Store() [2/2] . . . 112
6.31.3 Member Function Documentation . . . 112
6.31.3.1 GetMakeNameFromID() . . . 112

6.31.4 Property Documentation . . . 112
6.31.4.1 CatList . . . 112
6.31.4.2 ClientLIst . . . 113
6.31.4.3 MakeList . . . 113
6.31.4.4 ProdList . . . 113
6.31.4.5 SaleList . . . 113
6.31.4.6 WarrantList . . . 113
6.32 Data_BestSale.Warranties Class Reference . . . 114
6.32.1 Detailed Description . . . 114
6.32.2 Constructor & Destructor Documentation . . . 114
6.32.2.1 Warranties() [1/2] . . . 114
6.32.2.2 Warranties() [2/2] . . . 114
6.32.3 Member Function Documentation . . . 115
6.32.3.1 Add() . . . 115
6.32.3.2 ClearWarranties() . . . 115
6.32.3.3 Exist() . . . 115
6.32.3.4 Remove() . . . 115
6.32.4 Property Documentation . . . 116
6.32.4.1 Warrants . . . 116
6.33 trabalhoPOO_27967.Warranties Class Reference . . . 116
6.33.1 Detailed Description . . . 117
6.33.2 Constructor & Destructor Documentation . . . 117
6.33.2.1 Warranties() [1/2] . . . 117
6.33.2.2 Warranties() [2/2] . . . 117
6.33.3 Member Function Documentation . . . 117
6.33.3.1 Add() . . . 117
6.33.3.2 Exist() . . . 118
6.33.3.3 Remove() . . . 118
6.33.4 Property Documentation . . . 119
6.33.4.1 Warrants . . . 119
6.34 Data_BestSale.Warranty Class Reference . . . 119
6.34.1 Detailed Description . . . 120
6.34.2 Constructor & Destructor Documentation . . . 120
6.34.2.1 Warranty() [1/2] . . . 120
6.34.2.2 Warranty() [2/2] . . . 120
6.34.3 Member Function Documentation . . . 120
6.34.3.1 CreateWarranty() . . . 120
6.34.3.2 Equals() . . . 121
6.34.3.3 ExpirationDate() . . . 121
6.34.3.4 operator"!=() . . . 121
6.34.3.5 operator==() . . . 122
6.34.3.6 ToString() . . . 122

6.34.4 Property Documentation . . . 122
6.34.4.1 Conditions . . . 122
6.34.4.2 DurationInYears . . . 123
6.34.4.3 ProdID . . . 123
6.35 trabalhoPOO_27967.Warranty Class Reference . . . 123
6.35.1 Detailed Description . . . 124
6.35.2 Constructor & Destructor Documentation . . . 124
6.35.2.1 Warranty() [1/2] . . . 124
6.35.2.2 Warranty() [2/2] . . . 124
6.35.3 Member Function Documentation . . . 124
6.35.3.1 Equals() . . . 124
6.35.3.2 ExpirationDate() . . . 125
6.35.3.3 operator"!=() . . . 125
6.35.3.4 operator==() . . . 125
6.35.3.5 ToString() . . . 126
6.35.4 Property Documentation . . . 126
6.35.4.1 Conditions . . . 126
6.35.4.2 DurationInYears . . . 126
6.35.4.3 ProdID . . . 126

7 File Documentation **127**

7.1 BestSale.cs . . . 127
7.2 BestSale.cs . . . 128
7.3 .NETFramework,Version=v4.7.2.AssemblyAttributes.cs . . . 128
7.4 .NETFramework,Version=v4.7.2.AssemblyAttributes.cs . . . 128
7.5 .NETFramework,Version=v4.7.2.AssemblyAttributes.cs . . . 128
7.6 .NETFramework,Version=v4.7.2.AssemblyAttributes.cs . . . 129
7.7 .NETFramework,Version=v4.7.2.AssemblyAttributes.cs . . . 129
7.8 .NETFramework,Version=v4.7.2.AssemblyAttributes.cs . . . 129
7.9 .NETFramework,Version=v4.7.2.AssemblyAttributes.cs . . . 129
7.10 AssemblyInfo.cs . . . 129
7.11 AssemblyInfo.cs . . . 130
7.12 AssemblyInfo.cs . . . 130
7.13 AssemblyInfo.cs . . . 130
7.14 AssemblyInfo.cs . . . 131
7.15 AssemblyInfo.cs . . . 131
7.16 AssemblyInfo.cs . . . 132
7.17 BestSale_Validations.cs . . . 132
7.18 ClientManagement.cs . . . 133
7.19 FileManagement.cs . . . 133
7.20 ProductManagement.cs . . . 134
7.21 SimpleProduct.cs . . . 135

7.22 Campaign.cs . . . 136
7.23 Campaign.cs . . . 138
7.24 Campaigns.cs . . . 140
7.25 Campaigns.cs . . . 141
7.26 Categories.cs . . . 142
7.27 Categories.cs . . . 144
7.28 Category.cs . . . 145
7.29 Category.cs . . . 147
7.30 Client.cs . . . 148
7.31 Client.cs . . . 150
7.32 Clients.cs . . . 151
7.33 Clients.cs . . . 153
7.34 IListManagement.cs . . . 154
7.35 IListManagement.cs . . . 154
7.36 Make.cs . . . 155
7.37 Make.cs . . . 156
7.38 Makes.cs . . . 158
7.39 Makes.cs . . . 159
7.40 Product.cs . . . 161
7.41 Product.cs . . . 163
7.42 Products.cs . . . 164
7.43 Products.cs . . . 166
7.44 Sale.cs . . . 168
7.45 Sale.cs . . . 170
7.46 Sales.cs . . . 172
7.47 Sales.cs . . . 174
7.48 Store.cs . . . 175
7.49 Store.cs . . . 178
7.50 Warranties.cs . . . 179
7.51 Warranties.cs . . . 181
7.52 Warranty.cs . . . 182
7.53 Warranty.cs . . . 183
7.54 InvalidPhoneNumberException.cs . . . 185

Index **187**

Chapter 1

Namespace Index

1.1 Package List

Here are the packages with brief descriptions (if available):

BestSale 11
BestSale_Validations 11
Business_Layer 11
Business_Object 11
Data_BestSale 12
Exceptions 13
trabalhoPOO_27967 13
trabalhoPOO_27967.Interface 14
trabalhoPOO_27967.Store 14

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

ApplicationException
Exceptions.InvalidPhoneNumberException . . . 53
BestSale.BestSale . . . 15
Data_BestSale.Campaign . . . 15
trabalhoPOO_27967.Campaign . . . 19
Data_BestSale.Category . . . 32
trabalhoPOO_27967.Category . . . 36
Data_BestSale.Client . . . 39
trabalhoPOO_27967.Client . . . 42
Data_BestSale.IListManagement . . . 51
Data_BestSale.Campaigns . . . 23
Data_BestSale.Categories . . . 26
Data_BestSale.Clients . . . 46
Data_BestSale.Makes . . . 61
Data_BestSale.Products . . . 76
Data_BestSale.Sales . . . 97
Data_BestSale.Warranties . . . 114
trabalhoPOO_27967.Interface.IListManagement . . . 52
trabalhoPOO_27967.Campaigns . . . 24
trabalhoPOO_27967.Categories . . . 29
trabalhoPOO_27967.Clients . . . 48
trabalhoPOO_27967.Makes . . . 64
trabalhoPOO_27967.Products . . . 81
trabalhoPOO_27967.Sales . . . 99
trabalhoPOO_27967.Warranties . . . 116
Data_BestSale.Make . . . 54
trabalhoPOO_27967.Make . . . 58
Data_BestSale.Product . . . 67
trabalhoPOO_27967.Product . . . 72
Data_BestSale.Sale . . . 85
trabalhoPOO_27967.Sale . . . 91
Business_Object.SimpleProduct . . . 102
Data_BestSale.Store . . . 104
trabalhoPOO_27967.Store.Store . . . 111
Data_BestSale.Warranty . . . 119
trabalhoPOO_27967.Warranty . . . 123

Chapter 3

Class Index

3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

BestSale.BestSale 15

Data_BestSale.Campaign
Purpose: Definition of Campaign and methods to deal with Campaign operations. Created by: Jose Alves a27967 Created on: 11/6/2024 11:21:43 AM 15

trabalhoPOO_27967.Campaign
Purpose: Definition of Campaign and methods to deal with Campaign operations. Created by: Jose Alves a27967 Created on: 11/6/2024 11:21:43 AM 19

Data_BestSale.Campaigns
Purpose:This file has the definition and methods to work with the plurality of Campaign. Created by: Jose Alves a27967 Created on: 11/14/2024 3:57:04 PM 23

trabalhoPOO_27967.Campaigns
Purpose:This file has the definition and methods to work with the plurality of Campaign. Created by: Jose Alves a27967 Created on: 11/14/2024 3:57:04 PM 24

Data_BestSale.Categories
Purpose:This file has the definition and methods to work with the plurality of Category. Created by: Jose Alves a27967 Created on: 11/14/2024 4:45:58 PM 26

trabalhoPOO_27967.Categories
Purpose:This file has the definition and methods to work with the plurality of Category. Created by: Jose Alves a27967 Created on: 11/14/2024 4:45:58 PM 29

Data_BestSale.Category
Purpose: Definition of Category and methods to deal with Category operations. Created by: Jose Alves a27967 Created on: 11/6/2024 11:20:47 AM 32

trabalhoPOO_27967.Category
Purpose: Definition of Category and methods to deal with Category operations. Created by: Jose Alves a27967 Created on: 11/6/2024 11:20:47 AM 36

Data_BestSale.Client
Purpose: Definition of Client and methods to deal with Client operations. Created by: Jose Alves a27967 Created on: 10/29/2024 4:23:56 PM 39

trabalhoPOO_27967.Client
Purpose: Definition of Client and methods to deal with Client operations. Created by: Jose Alves a27967 Created on: 10/29/2024 4:23:56 PM 42

Data_BestSale.Clients
Purpose: Class with the definition and methods to manage a list of clients. Created by: Jose Alves a27967 Created on: 11/12/2024 9:25:28 PM 46

trabalhoPOO_27967.Clients
Purpose: Class with the definition and methods to manage a list of clients. Created by: Jose Alves a27967 Created on: 11/12/2024 9:25:28 PM . . . 48
Data_BestSale.IListManagement . . . 51
trabalhoPOO_27967.Interface.IListManagement . . . 52
Exceptions.InvalidPhoneNumberException
The exception to be throws when a string doesn't match the phone number pattern . . . 53
Data_BestSale.Make
Purpose: Definition of Make and methods to deal with Make operations. Created by: Jose Alves a27967 Created on: 11/6/2024 11:22:09 AM . . . 54
trabalhoPOO_27967.Make
Purpose: Definition of Make and methods to deal with Make operations. Created by: Jose Alves a27967 Created on: 11/6/2024 11:22:09 AM . . . 58
Data_BestSale.Makes
Purpose:This file has the definition and methods to work with the plurality of Make. Created by: Jose Alves a27967 Created on: 11/14/2024 4:33:51 PM . . . 61
trabalhoPOO_27967.Makes
Purpose:This file has the definition and methods to work with the plurality of Make. Created by: Jose Alves a27967 Created on: 11/14/2024 4:33:51 PM . . . 64
Data_BestSale.Product
Purpose: Definition of product and methods to deal with product operations. Created by: Jose Alves a27967 Created on: 11/2/2024 4:40:12 PM . . . 67
trabalhoPOO_27967.Product
Purpose: Definition of product and methods to deal with product operations. Created by: Jose Alves a27967 Created on: 11/2/2024 4:40:12 PM . . . 72
Data_BestSale.Products
Purpose: Class to manage a group of more than one product. Created by: Jose Alves a27967 Created on: 11/9/2024 6:34:19 PM . . . 76
trabalhoPOO_27967.Products
Purpose: Class to manage a group of more than one product. Created by: Jose Alves a27967 Created on: 11/9/2024 6:34:19 PM . . . 81
Data_BestSale.Sale
Purpose: Definition of Sale and methods to deal with Sale operations. Created by: Jose Alves a27967 Created on: 11/6/2024 11:21:53 AM . . . 85
trabalhoPOO_27967.Sale
Purpose: Definition of Sale and methods to deal with Sale operations. Created by: Jose Alves a27967 Created on: 11/6/2024 11:21:53 AM . . . 91
Data_BestSale.Sales
Purpose: Class with the agregation of sales of a store. Created by: Jose Alves a27967 Created on: 11/10/2024 7:42:03 PM . . . 97
trabalhoPOO_27967.Sales
Purpose: Class with the agregation of sales of a store. Created by: Jose Alves a27967 Created on: 11/10/2024 7:42:03 PM . . . 99
Business_Object.SimpleProduct
Purpose:This File contains the definition and methods to manage a SimpleClient Created by: zecun Created on: 12/11/2024 11:18:40 AM . . . 102
Data_BestSale.Store
Purpose: This class has the definition and properties to manage a store. Created by: Jose Alves a27967 Created on: 11/14/2024 5:01:23 PM . . . 104
trabalhoPOO_27967.Store.Store
Purpose: This class has the definition and properties to manage a store. Created by: Jose Alves a27967 Created on: 11/14/2024 5:01:23 PM . . . 111
Data_BestSale.Warranties
Purpose:This file has the definition and methods to work with the plurality of Warranty. Created by: Jose Alves a27967 Created on: 11/14/2024 4:20:11 PM . . . 114
trabalhoPOO_27967.Warranties
Purpose:This file has the definition and methods to work with the plurality of Warranty. Created by: Jose Alves a27967 Created on: 11/14/2024 4:20:11 PM . . . 116

Data_BestSale.Warranty
Purpose: This class contains the definition and methods to manage warranties. Created by: Jose Alves a27967 Created on: 11/13/2024 4:17:18 PM 119
trabalhoPOO_27967.Warranty
Purpose: This class contains the definition and methods to manage warranties. Created by: Jose Alves a27967 Created on: 11/13/2024 4:17:18 PM 123

Chapter 4

File Index

4.1 File List

Here is a list of all documented files with brief descriptions:

BestSale/BestSale.cs . . . 127
BestSale/obj/Debug/.NETFramework,Version=v4.7.2.AssemblyAttributes.cs . . . 128
BestSale/Properties/AssemblyInfo.cs . . . 129
BestSale_Validations/BestSale_Validations.cs . . . 132
BestSale_Validations/obj/Debug/.NETFramework,Version=v4.7.2.AssemblyAttributes.cs . . . 128
BestSale_Validations/Properties/AssemblyInfo.cs . . . 130
Business_Layer/ClientManagement.cs . . . 133
Business_Layer/FileManagement.cs . . . 133
Business_Layer/ProductManagement.cs . . . 134
Business_Layer/obj/Debug/.NETFramework,Version=v4.7.2.AssemblyAttributes.cs . . . 128
Business_Layer/Properties/AssemblyInfo.cs . . . 130
Business_Object/SimpleProduct.cs . . . 135
Business_Object/obj/Debug/.NETFramework,Version=v4.7.2.AssemblyAttributes.cs . . . 129
Business_Object/Properties/AssemblyInfo.cs . . . 130
Data_BestSale/Campaign/Campaign.cs . . . 136
Data_BestSale/Campaign/Campaigns.cs . . . 140
Data_BestSale/Category/Categories.cs . . . 142
Data_BestSale/Category/Category.cs . . . 145
Data_BestSale/Client/Client.cs . . . 148
Data_BestSale/Client/Clients.cs . . . 151
Data_BestSale/Interface/IListManagement.cs . . . 154
Data_BestSale/Make/Make.cs . . . 155
Data_BestSale/Make/Makes.cs . . . 158
Data_BestSale/obj/Debug/.NETFramework,Version=v4.7.2.AssemblyAttributes.cs . . . 129
Data_BestSale/Product/Product.cs . . . 161
Data_BestSale/Product/Products.cs . . . 164
Data_BestSale/Properties/AssemblyInfo.cs . . . 131
Data_BestSale/Sale/Sale.cs . . . 168
Data_BestSale/Sale/Sales.cs . . . 172
Data_BestSale/Store/Store.cs . . . 175
Data_BestSale/Warranty/Warranties.cs . . . 179
Data_BestSale/Warranty/Warranty.cs . . . 182
Exceptions/InvalidPhoneNumberException.cs . . . 185
Exceptions/obj/Debug/.NETFramework,Version=v4.7.2.AssemblyAttributes.cs . . . 129
Exceptions/Properties/AssemblyInfo.cs . . . 131

trabalhoPOO_27967/BestSale.cs . . . 128
trabalhoPOO_27967/Campaign/Campaign.cs . . . 138
trabalhoPOO_27967/Campaign/Campaigns.cs . . . 141
trabalhoPOO_27967/Category/Categories.cs . . . 144
trabalhoPOO_27967/Category/Category.cs . . . 147
trabalhoPOO_27967/Client/Client.cs . . . 150
trabalhoPOO_27967/Client/Clients.cs . . . 153
trabalhoPOO_27967/Interface/IListManagement.cs . . . 154
trabalhoPOO_27967/Make/Make.cs . . . 156
trabalhoPOO_27967/Make/Makes.cs . . . 159
trabalhoPOO_27967/obj/Debug/.NETFramework,Version=v4.7.2.AssemblyAttributes.cs . . . 129
trabalhoPOO_27967/Product/Product.cs . . . 163
trabalhoPOO_27967/Product/Products.cs . . . 166
trabalhoPOO_27967/Properties/AssemblyInfo.cs . . . 132
trabalhoPOO_27967/Sale/Sale.cs . . . 170
trabalhoPOO_27967/Sale/Sales.cs . . . 174
trabalhoPOO_27967/Store/Store.cs . . . 178
trabalhoPOO_27967/Warranty/Warranties.cs . . . 181
trabalhoPOO_27967/Warranty/Warranty.cs . . . 183

Chapter 5

Namespace Documentation

5.1 BestSale Namespace Reference

Classes

- class BestSale

5.2 BestSale_Validations Namespace Reference

Classes

- class **BestSale_Validations**

5.3 Business_Layer Namespace Reference

Classes

- class **ClientManagement**

  Purpose:This namespace has all the necessary calls to the back end to manage clients. Created by: zecun Created on: 12/10/2024 10:57:31 PM.
- class **FileManagement**

  Purpose:This File contains the calls to the methods to save data to a file. Created by: zecun Created on: 12/11/2024 12:04:37 PM.
- class **ProductManagement**

  Purpose: This namespace has all the necessary calls to the back end to manage products, categories, makes and warranties. Created by: zecun Created on: 12/14/2024 4:28:51 PM.

5.4 Business_Object Namespace Reference

Classes

- class SimpleProduct

  Purpose:This File contains the definition and methods to manage a SimpleClient Created by: zecun Created on: 12/11/2024 11:18:40 AM.

5.5 Data_BestSale Namespace Reference

Classes

- class Campaign

  Purpose: Definition of Campaign and methods to deal with Campaign operations. Created by: Jose Alves a27967 Created on: 11/6/2024 11:21:43 AM.
- class Campaigns

  Purpose:This file has the definition and methods to work with the plurality of Campaign. Created by: Jose Alves a27967 Created on: 11/14/2024 3:57:04 PM.
- class Categories

  Purpose:This file has the definition and methods to work with the plurality of Category. Created by: Jose Alves a27967 Created on: 11/14/2024 4:45:58 PM.
- class Category

  Purpose: Definition of Category and methods to deal with Category operations. Created by: Jose Alves a27967 Created on: 11/6/2024 11:20:47 AM.
- class Client

  Purpose: Definition of Client and methods to deal with Client operations. Created by: Jose Alves a27967 Created on: 10/29/2024 4:23:56 PM.
- class Clients

  Purpose: Class with the definition and methods to manage a list of clients. Created by: Jose Alves a27967 Created on: 11/12/2024 9:25:28 PM.
- interface IListManagement
- class Make

  Purpose: Definition of Make and methods to deal with Make operations. Created by: Jose Alves a27967 Created on: 11/6/2024 11:22:09 AM.
- class Makes

  Purpose:This file has the definition and methods to work with the plurality of Make. Created by: Jose Alves a27967 Created on: 11/14/2024 4:33:51 PM.
- class Product

  Purpose: Definition of product and methods to deal with product operations. Created by: Jose Alves a27967 Created on: 11/2/2024 4:40:12 PM.
- class Products

  Purpose: Class to manage a group of more than one product. Created by: Jose Alves a27967 Created on: 11/9/2024 6:34:19 PM.
- class Sale

  Purpose: Definition of Sale and methods to deal with Sale operations. Created by: Jose Alves a27967 Created on: 11/6/2024 11:21:53 AM.
- class Sales

  Purpose: Class with the agregation of sales of a store. Created by: Jose Alves a27967 Created on: 11/10/2024 7:42:03 PM.
- class Store

  Purpose: This class has the definition and properties to manage a store. Created by: Jose Alves a27967 Created on: 11/14/2024 5:01:23 PM.
- class Warranties

  Purpose:This file has the definition and methods to work with the plurality of Warranty. Created by: Jose Alves a27967 Created on: 11/14/2024 4:20:11 PM.
- class Warranty

  Purpose: This class contains the definition and methods to manage warranties. Created by: Jose Alves a27967 Created on: 11/13/2024 4:17:18 PM.

5.6 Exceptions Namespace Reference

Classes

- class InvalidPhoneNumberException

  The exception to be throws when a string doesn't match the phone number pattern.

5.7 trabalhoPOO_27967 Namespace Reference

Classes

- class Campaign

  Purpose: Definition of Campaign and methods to deal with Campaign operations. Created by: Jose Alves a27967 Created on: 11/6/2024 11:21:43 AM.
- class Campaigns

  Purpose:This file has the definition and methods to work with the plurality of Campaign. Created by: Jose Alves a27967 Created on: 11/14/2024 3:57:04 PM.
- class Categories

  Purpose:This file has the definition and methods to work with the plurality of Category. Created by: Jose Alves a27967 Created on: 11/14/2024 4:45:58 PM.
- class Category

  Purpose: Definition of Category and methods to deal with Category operations. Created by: Jose Alves a27967 Created on: 11/6/2024 11:20:47 AM.
- class Client

  Purpose: Definition of Client and methods to deal with Client operations. Created by: Jose Alves a27967 Created on: 10/29/2024 4:23:56 PM.
- class Clients

  Purpose: Class with the definition and methods to manage a list of clients. Created by: Jose Alves a27967 Created on: 11/12/2024 9:25:28 PM.
- class Make

  Purpose: Definition of Make and methods to deal with Make operations. Created by: Jose Alves a27967 Created on: 11/6/2024 11:22:09 AM.
- class Makes

  Purpose:This file has the definition and methods to work with the plurality of Make. Created by: Jose Alves a27967 Created on: 11/14/2024 4:33:51 PM.
- class Product

  Purpose: Definition of product and methods to deal with product operations. Created by: Jose Alves a27967 Created on: 11/2/2024 4:40:12 PM.
- class Products

  Purpose: Class to manage a group of more than one product. Created by: Jose Alves a27967 Created on: 11/9/2024 6:34:19 PM.
- class Sale

  Purpose: Definition of Sale and methods to deal with Sale operations. Created by: Jose Alves a27967 Created on: 11/6/2024 11:21:53 AM.
- class Sales

  Purpose: Class with the agregation of sales of a store. Created by: Jose Alves a27967 Created on: 11/10/2024 7:42:03 PM.
- class Warranties

  Purpose:This file has the definition and methods to work with the plurality of Warranty. Created by: Jose Alves a27967 Created on: 11/14/2024 4:20:11 PM.
- class Warranty

  Purpose: This class contains the definition and methods to manage warranties. Created by: Jose Alves a27967 Created on: 11/13/2024 4:17:18 PM.

5.8 trabalhoPOO_27967.Interface Namespace Reference

Classes

- interface IListManagement

5.9 trabalhoPOO_27967.Store Namespace Reference

Classes

- class Store

  Purpose: This class has the definition and properties to manage a store. Created by: Jose Alves a27967 Created on: 11/14/2024 5:01:23 PM.

Chapter 6

Class Documentation

6.1 BestSale.BestSale Class Reference

6.1.1 Detailed Description

Definition at line 12 of file BestSale.cs.

The documentation for this class was generated from the following files:

- BestSale/BestSale.cs
- trabalhoPOO_27967/BestSale.cs

6.2 Data_BestSale.Campaign Class Reference

Purpose: Definition of Campaign and methods to deal with Campaign operations. Created by: Jose Alves a27967 Created on: 11/6/2024 11:21:43 AM.

Public Member Functions

- Campaign ()

  The default Constructor.
- Campaign (string name, decimal discount, DateTime startDate, DateTime endDate)

  Constructor used to create a campaign when a name, a decimal value of discount, start and ending dates are given.
- override bool Equals (object obj)

  Redefine the Equals operator to verify if a campaign matches the other.

Static Public Member Functions

- static bool operator== (Campaign camp1, Campaign camp2)

  Redefinition of the == operator.
- static bool operator!= (Campaign camp1, Campaign camp2)

  Redefinition of the != operator.
- static bool VerifyApplicability (Campaign camp)

  Method to verify if a certaign campaign is applicable, according to its start and end dates.

Properties

- int Id `[get, set]`

  Property used to get and set the ID of a Campaign.
- string Name `[get, set]`

  Property used to get and set the Name of a Campaign.
- decimal Discount `[get, set]`

  Property used to get and set the Decimal Discount of a Campaign.
- DateTime StartDate `[get, set]`

  Property used to get and set the Start Date of a Campaign.
- DateTime EndDate `[get, set]`

  Property used to get and set the End Date of a Campaign.
- int CampaignCount `[get, set]`

  Property used to get and set the Campaign Counter.

6.2.1 Detailed Description

Purpose: Definition of Campaign and methods to deal with Campaign operations. Created by: Jose Alves a27967 Created on: 11/6/2024 11:21:43 AM.

Definition at line 22 of file Campaign.cs.

6.2.2 Constructor & Destructor Documentation

6.2.2.1 Campaign() [1/2]

```
Data_BestSale.Campaign.Campaign ()
```

The default Constructor.

Definition at line 40 of file Campaign.cs.

6.2.2.2 Campaign() [2/2]

```
Data_BestSale.Campaign.Campaign (
            string name,
            decimal discount,
            DateTime startDate,
            DateTime endDate)
```

Constructor used to create a campaign when a name, a decimal value of discount, start and ending dates are given.

Parameters

id	
name	
discount	
startDate	
endDate	

Definition at line 57 of file Campaign.cs.

6.2.3 Member Function Documentation

6.2.3.1 Equals()

```
override bool Data_BestSale.Campaign.Equals (
            object obj)
```

Redefine the Equals operator to verify if a campaign matches the other.

Parameters

obj	

Returns

Veriffies if the object given is null.

Casts the object to be Campaign.

Definition at line 136 of file Campaign.cs.

6.2.3.2 operator"!=()

```
static bool Data_BestSale.Campaign.operator!= (
            Campaign camp1,
            Campaign camp2)  [static]
```

Redefinition of the != operator.

Parameters

cli1	
cli2	

Returns

Definition at line 166 of file Campaign.cs.

6.2.3.3 operator==()

```
static bool Data_BestSale.Campaign.operator== (
            Campaign camp1,
            Campaign camp2)  [static]
```

Redefinition of the == operator.

Parameters

cli1	
cli2	

Returns

Definition at line 155 of file Campaign.cs.

6.2.3.4 VerifyApplicability()

```
static bool Data_BestSale.Campaign.VerifyApplicability (
            Campaign camp)  [static]
```

Method to verify if a certaign campaign is applicable, according to its start and end dates.

Parameters

camp	

Returns

Verifies if the actual date is in between the start and end dates of the campaign.

Definition at line 179 of file Campaign.cs.

6.2.4 Property Documentation

6.2.4.1 CampaignCount

```
int Data_BestSale.Campaign.CampaignCount  [get], [set]
```

Property used to get and set the Campaign Counter.

Definition at line 121 of file Campaign.cs.

6.2.4.2 Discount

```
decimal Data_BestSale.Campaign.Discount  [get], [set]
```

Property used to get and set the Decimal Discount of a Campaign.

Definition at line 94 of file Campaign.cs.

6.2.4.3 EndDate

```
DateTime Data_BestSale.Campaign.EndDate  [get], [set]
```

Property used to get and set the End Date of a Campaign.

Definition at line 112 of file Campaign.cs.

6.2.4.4 Id

```
int Data_BestSale.Campaign.Id  [get], [set]
```

Property used to get and set the ID of a Campaign.

Definition at line 77 of file Campaign.cs.

6.2.4.5 Name

```
string Data_BestSale.Campaign.Name  [get], [set]
```

Property used to get and set the Name of a Campaign.

Definition at line 85 of file Campaign.cs.

6.2.4.6 StartDate

```
DateTime Data_BestSale.Campaign.StartDate  [get], [set]
```

Property used to get and set the Start Date of a Campaign.

Definition at line 103 of file Campaign.cs.

The documentation for this class was generated from the following file:

- Data_BestSale/Campaign/Campaign.cs

6.3 trabalhoPOO_27967.Campaign Class Reference

Purpose: Definition of Campaign and methods to deal with Campaign operations. Created by: Jose Alves a27967 Created on: 11/6/2024 11:21:43 AM.

Public Member Functions

- Campaign ()

  The default Constructor.
- Campaign (string name, decimal discount, DateTime startDate, DateTime endDate)

  Constructor used to create a campaign when a name, a decimal value of discount, start and ending dates are given.
- override bool Equals (object obj)

  Redefine the Equals operator to verify if a campaign matches the other.

Static Public Member Functions

- static bool operator== (Campaign camp1, Campaign camp2)

  Redefinition of the == operator.
- static bool operator!= (Campaign camp1, Campaign camp2)

  Redefinition of the != operator.
- static bool VerifyApplicability (Campaign camp)

  Method to verify if a certaign campaign is applicable, according to its start and end dates.

Properties

- int Id `[get, set]`

  Property used to get and set the ID of a Campaign.
- string Name `[get, set]`

  Property used to get and set the Name of a Campaign.
- decimal Discount `[get, set]`

  Property used to get and set the Decimal Discount of a Campaign.
- DateTime StartDate `[get, set]`

  Property used to get and set the Start Date of a Campaign.
- DateTime EndDate `[get, set]`

  Property used to get and set the End Date of a Campaign.
- int CampaignCount `[get, set]`

  Property used to get and set the Campaign Counter.

6.3.1 Detailed Description

Purpose: Definition of Campaign and methods to deal with Campaign operations. Created by: Jose Alves a27967 Created on: 11/6/2024 11:21:43 AM.

Definition at line 20 of file Campaign.cs.

6.3.2 Constructor & Destructor Documentation

6.3.2.1 Campaign() [1/2]

```
trabalhoPOO_27967.Campaign.Campaign ()
```

The default Constructor.

Definition at line 38 of file Campaign.cs.

6.3.2.2 Campaign() [2/2]

```
trabalhoPOO_27967.Campaign.Campaign (
            string name,
            decimal discount,
            DateTime startDate,
            DateTime endDate)
```

Constructor used to create a campaign when a name, a decimal value of discount, start and ending dates are given.

Parameters

id	
name	
discount	
startDate	
endDate	

Definition at line 55 of file Campaign.cs.

6.3.3 Member Function Documentation

6.3.3.1 Equals()

```
override bool trabalhoPOO_27967.Campaign.Equals (
            object obj)
```

Redefine the Equals operator to verify if a campaign matches the other.

Parameters

obj	

Returns

Veriffies if the object given is null.

Casts the object to be Campaign.

Definition at line 134 of file Campaign.cs.

6.3.3.2 operator"!=()

```
static bool trabalhoPOO_27967.Campaign.operator!= (
            Campaign camp1,
            Campaign camp2)  [static]
```

Redefinition of the != operator.

Parameters

cli1	
cli2	

Returns

Definition at line 164 of file Campaign.cs.

6.3.3.3 operator==()

```
static bool trabalhoPOO_27967.Campaign.operator== (
            Campaign camp1,
            Campaign camp2)  [static]
```

Redefinition of the == operator.

Parameters

cli1	
cli2	

Returns

Definition at line 153 of file Campaign.cs.

6.3.3.4 VerifyApplicability()

```
static bool trabalhoPOO_27967.Campaign.VerifyApplicability (
            Campaign camp)  [static]
```

Method to verify if a certaign campaign is applicable, according to its start and end dates.

Parameters

camp	

Returns

Verifies if the actual date is in between the start and end dates of the campaign.

Definition at line 177 of file Campaign.cs.

6.3.4 Property Documentation

6.3.4.1 CampaignCount

```
int trabalhoPOO_27967.Campaign.CampaignCount  [get], [set]
```

Property used to get and set the Campaign Counter.

Definition at line 119 of file Campaign.cs.

6.3.4.2 Discount

```
decimal trabalhoPOO_27967.Campaign.Discount  [get], [set]
```

Property used to get and set the Decimal Discount of a Campaign.

Definition at line 92 of file Campaign.cs.

6.3.4.3 EndDate

```
DateTime trabalhoPOO_27967.Campaign.EndDate  [get], [set]
```

Property used to get and set the End Date of a Campaign.

Definition at line 110 of file Campaign.cs.

6.3.4.4 Id

```
int trabalhoPOO_27967.Campaign.Id  [get], [set]
```

Property used to get and set the ID of a Campaign.

Definition at line 75 of file Campaign.cs.

6.3.4.5 Name

```
string trabalhoPOO_27967.Campaign.Name  [get], [set]
```

Property used to get and set the Name of a Campaign.

Definition at line 83 of file Campaign.cs.

6.3.4.6 StartDate

```
DateTime trabalhoPOO_27967.Campaign.StartDate  [get], [set]
```

Property used to get and set the Start Date of a Campaign.

Definition at line 101 of file Campaign.cs.

The documentation for this class was generated from the following file:

- trabalhoPOO_27967/Campaign/Campaign.cs

6.4 Data_BestSale.Campaigns Class Reference

Purpose:This file has the definition and methods to work with the plurality of Campaign. Created by: Jose Alves a27967 Created on: 11/14/2024 3:57:04 PM.

Inheritance diagram for Data_BestSale.Campaigns:

6.5 trabalhoPOO_27967.Campaigns Class Reference

Purpose:This file has the definition and methods to work with the plurality of Campaign. Created by: Jose Alves a27967 Created on: 11/14/2024 3:57:04 PM.

Inheritance diagram for trabalhoPOO_27967.Campaigns:

Collaboration diagram for trabalhoPOO_27967.Campaigns:

Public Member Functions

- Campaigns ()
  The default Constructor.
- Campaigns (List< Campaign > p)
  The constructor to use when a list of Campaigns is given.
- bool Add (object obj)
  Method used to add a campaign to a list of campaigns.
- bool Remove (object obj)
  Method used to remove a campaign from a list of campaigns.
- bool Exist (object obj)
  Method used to confirm if a campaign exists on a list of campaigns.

Public Member Functions inherited from trabalhoPOO_27967.Interface.IListManagement

Properties

- List< Campaign > Camps `[get, set]`
  Property used to get and set a Campaign list.

6.5.1 Detailed Description

Purpose:This file has the definition and methods to work with the plurality of Campaign. Created by: Jose Alves a27967 Created on: 11/14/2024 3:57:04 PM.

Definition at line 23 of file Campaigns.cs.

6.5.2 Constructor & Destructor Documentation

6.5.2.1 Campaigns() [1/2]

```
trabalhoPOO_27967.Campaigns.Campaigns ()
```

The default Constructor.

Definition at line 36 of file Campaigns.cs.

6.5.2.2 Campaigns() [2/2]

```
trabalhoPOO_27967.Campaigns.Campaigns (
            List< Campaign > p)
```

The constructor to use when a list of Campaigns is given.

Parameters

p	

Definition at line 45 of file Campaigns.cs.

6.5.3 Member Function Documentation

6.5.3.1 Add()

```
bool trabalhoPOO_27967.Campaigns.Add (
            object obj)
```

Method used to add a campaign to a list of campaigns.

Parameters

c	

Returns

Implements trabalhoPOO_27967.Interface.IListManagement.

Definition at line 74 of file Campaigns.cs.

6.5.3.2 Exist()

```
bool trabalhoPOO_27967.Campaigns.Exist (
            object obj)
```

Method used to confirm if a campaign exists on a list of campaigns.

Parameters

id	

Returns

Implements trabalhoPOO_27967.Interface.IListManagement.

Definition at line 109 of file Campaigns.cs.

6.5.3.3 Remove()

```
bool trabalhoPOO_27967.Campaigns.Remove (
            object obj)
```

Method used to remove a campaign from a list of campaigns.

Parameters

camp	

Returns

Implements trabalhoPOO_27967.Interface.IListManagement.

Definition at line 90 of file Campaigns.cs.

6.5.4 Property Documentation

6.5.4.1 Camps

```
List<Campaign> trabalhoPOO_27967.Campaigns.Camps  [get], [set]
```

Property used to get and set a Campaign list.

Definition at line 55 of file Campaigns.cs.

The documentation for this class was generated from the following file:

- trabalhoPOO_27967/Campaign/Campaigns.cs

6.6 Data_BestSale.Categories Class Reference

Purpose:This file has the definition and methods to work with the plurality of Category. Created by: Jose Alves a27967 Created on: 11/14/2024 4:45:58 PM.

Inheritance diagram for Data_BestSale.Categories:

Collaboration diagram for Data_BestSale.Categories:

Public Member Functions

- Categories ()
  The default Constructor.
- Categories (List< Category > cats)
  The constructor to use when a list of categories is given.
- bool Add (object obj)
  Method used to add a category to a list of categories.
- bool Remove (object obj)
  Method used to remove a category from a list of categories.
- bool Exist (object obj)
  Method used to verify if a category exists on a list of makes, given its ID or name.
- bool ClearCategories ()
  Method used to Clear a list of Categories.
- Category GetCategory (object obj)
  This method returns a category, given its name or id.

Public Member Functions inherited from Data_BestSale.IListManagement

Properties

- List< Category > Cats `[get, set]`

  The property used to get and set the list of categories.

6.6.1 Detailed Description

Purpose:This file has the definition and methods to work with the plurality of Category. Created by: Jose Alves a27967 Created on: 11/14/2024 4:45:58 PM.

Definition at line 23 of file Categories.cs.

6.6.2 Constructor & Destructor Documentation

6.6.2.1 Categories() [1/2]

```
Data_BestSale.Categories.Categories ()
```

The default Constructor.

Definition at line 36 of file Categories.cs.

6.6.2.2 Categories() [2/2]

```
Data_BestSale.Categories.Categories (
            List< Category > cats)
```

The constructor to use when a list of categories is given.

Parameters

cats	

Definition at line 45 of file Categories.cs.

6.6.3 Member Function Documentation

6.6.3.1 Add()

```
bool Data_BestSale.Categories.Add (
            object obj)
```

Method used to add a category to a list of categories.

Parameters

c	

Returns

Implements Data_BestSale.IListManagement.

Definition at line 75 of file Categories.cs.

6.6.3.2 ClearCategories()

```
bool Data_BestSale.Categories.ClearCategories ()
```

Method used to Clear a list of Categories.

Definition at line 143 of file Categories.cs.

6.6.3.3 Exist()

```
bool Data_BestSale.Categories.Exist (
            object obj)
```

Method used to verify if a category exists on a list of makes, given its ID or name.

Parameters

id	

Returns

If the ID is given

The Name is given

Implements Data_BestSale.IListManagement.

Definition at line 109 of file Categories.cs.

6.6.3.4 GetCategory()

```
Category Data_BestSale.Categories.GetCategory (
            object obj)
```

This method returns a category, given its name or id.

Parameters

obj	The ID or Name of the Category.

Returns

The category

Definition at line 161 of file Categories.cs.

6.6.3.5 Remove()

```
bool Data_BestSale.Categories.Remove (
            object obj)
```

Method used to remove a category from a list of categories.

Parameters

c	

Returns

Implements Data_BestSale.IListManagement.

Definition at line 91 of file Categories.cs.

6.6.4 Property Documentation

6.6.4.1 Cats

```
List<Category> Data_BestSale.Categories.Cats  [get], [set]
```

The property used to get and set the list of categories.

Definition at line 57 of file Categories.cs.

The documentation for this class was generated from the following file:

- Data_BestSale/Category/Categories.cs

6.7 trabalhoPOO_27967.Categories Class Reference

Purpose:This file has the definition and methods to work with the plurality of Category. Created by: Jose Alves a27967 Created on: 11/14/2024 4:45:58 PM.

Inheritance diagram for trabalhoPOO_27967.Categories:

Collaboration diagram for trabalhoPOO_27967.Categories:

Public Member Functions

- Categories ()
  The default Constructor.
- Categories (List< Category > cats)
  The constructor to use when a list of categories is given.
- bool Add (object obj)
  Method used to add a category to a list of categories.
- bool Remove (object obj)
  Method used to remove a category from a list of categories.
- bool Exist (object obj)
  Method used to verify if a category exists on a list of makes, given its ID or name.

Public Member Functions inherited from trabalhoPOO_27967.Interface.IListManagement

Properties

- List< Category > Cats `[get, set]`
  The property used to get and set the list of categories.

6.7.1 Detailed Description

Purpose:This file has the definition and methods to work with the plurality of Category. Created by: Jose Alves a27967 Created on: 11/14/2024 4:45:58 PM.

Definition at line 23 of file Categories.cs.

6.7.2 Constructor & Destructor Documentation

6.7.2.1 Categories() [1/2]

```
trabalhoPOO_27967.Categories.Categories ()
```

The default Constructor.

Definition at line 36 of file Categories.cs.

6.7.2.2 Categories() [2/2]

```
trabalhoPOO_27967.Categories.Categories (
            List< Category > cats)
```

The constructor to use when a list of categories is given.

Parameters

cats	

Definition at line 45 of file Categories.cs.

6.7.3 Member Function Documentation

6.7.3.1 Add()

```
bool trabalhoPOO_27967.Categories.Add (
            object obj)
```

Method used to add a category to a list of categories.

Parameters

c	

Returns

Implements trabalhoPOO_27967.Interface.IListManagement.

Definition at line 75 of file Categories.cs.

6.7.3.2 Exist()

```
bool trabalhoPOO_27967.Categories.Exist (
            object obj)
```

Method used to verify if a category exists on a list of makes, given its ID or name.

Parameters

id	

Returns

If the ID is given

The Name is given

Implements trabalhoPOO_27967.Interface.IListManagement.

Definition at line 109 of file Categories.cs.

6.7.3.3 Remove()

```
bool trabalhoPOO_27967.Categories.Remove (
            object obj)
```

Method used to remove a category from a list of categories.

Parameters

c	

Returns

Implements trabalhoPOO_27967.Interface.IListManagement.

Definition at line 91 of file Categories.cs.

6.7.4 Property Documentation

6.7.4.1 Cats

```
List<Category> trabalhoPOO_27967.Categories.Cats  [get], [set]
```

The property used to get and set the list of categories.

Definition at line 57 of file Categories.cs.

The documentation for this class was generated from the following file:

- trabalhoPOO_27967/Category/Categories.cs

6.8 Data_BestSale.Category Class Reference

Purpose: Definition of Category and methods to deal with Category operations. Created by: Jose Alves a27967 Created on: 11/6/2024 11:20:47 AM.

Public Member Functions

- Category ()
  The default Constructor.
- Category (string name)
  The constructor to use when the name of a category is given.
- override bool Equals (object obj)
  Method that overrides Equals() and verifies if a categry matches another one.

Static Public Member Functions

- static bool operator== (Category cat1, Category cat2)
  Redefinition of the Equal operator.
- static bool operator!= (Category cat1, Category cat2)
  Redefinition of the Not Equal Operator.
- static bool CreateCategory (string name, out Category category)
  This method creates a new category instance.

Properties

- int Id `[get, set]`
  The property to get and set the ID of a category.
- string Name `[get, set]`
  The property to get and set the name of a Category.

6.8.1 Detailed Description

Purpose: Definition of Category and methods to deal with Category operations. Created by: Jose Alves a27967 Created on: 11/6/2024 11:20:47 AM.

Definition at line 21 of file Category.cs.

6.8.2 Constructor & Destructor Documentation

6.8.2.1 Category() [1/2]

```
Data_BestSale.Category.Category ()
```

The default Constructor.

Definition at line 36 of file Category.cs.

6.8.2.2 Category() [2/2]

```
Data_BestSale.Category.Category (
            string name)
```

The constructor to use when the name of a category is given.

Parameters

name	

Definition at line 46 of file Category.cs.

6.8.3 Member Function Documentation

6.8.3.1 CreateCategory()

```
static bool Data_BestSale.Category.CreateCategory (
            string name,
            out Category category)  [static]
```

This method creates a new category instance.

Parameters

name	
category	

Returns

True - if succeeded

Exception - An error occurred.

Returns

Definition at line 126 of file Category.cs.

6.8.3.2 Equals()

```
override bool Data_BestSale.Category.Equals (
            object obj)
```

Method that overrides Equals() and verifies if a categry matches another one.

Parameters

obj	

Returns

Veriffies if the object given is null.

Casts the object to be Category.

Definition at line 82 of file Category.cs.

6.8.3.3 operator"!=()

```
static bool Data_BestSale.Category.operator!= (
            Category cat1,
            Category cat2)  [static]
```

Redefinition of the Not Equal Operator.

Parameters

cat1	
cat2	

Returns

Definition at line 112 of file Category.cs.

6.8.3.4 operator==()

```
static bool Data_BestSale.Category.operator== (
            Category cat1,
            Category cat2)  [static]
```

Redefinition of the Equal operator.

Parameters

cat1	
cat2	

Returns

Definition at line 101 of file Category.cs.

6.8.4 Property Documentation

6.8.4.1 Id

```
int Data_BestSale.Category.Id  [get], [set]
```

The property to get and set the ID of a category.

Definition at line 58 of file Category.cs.

6.8.4.2 Name

```
string Data_BestSale.Category.Name  [get], [set]
```

The property to get and set the name of a Category.

Definition at line 67 of file Category.cs.

The documentation for this class was generated from the following file:

- Data_BestSale/Category/Category.cs

6.9 trabalhoPOO_27967.Category Class Reference

Purpose: Definition of Category and methods to deal with Category operations. Created by: Jose Alves a27967 Created on: 11/6/2024 11:20:47 AM.

Public Member Functions

- Category ()
  The default Constructor.
- Category (string name)
  The constructor to use when the name of a category is given.
- override bool Equals (object obj)
  Method that overrides Equals() and verifies if a categry matches another one.

Static Public Member Functions

- static bool operator== (Category cat1, Category cat2)
  Redefinition of the Equal operator.
- static bool operator!= (Category cat1, Category cat2)
  Redefinition of the Not Equal Operator.

Properties

- int Id `[get, set]`
  The property to get and set the ID of a category.
- string Name `[get, set]`
  The property to get and set the name of a Category.

6.9.1 Detailed Description

Purpose: Definition of Category and methods to deal with Category operations. Created by: Jose Alves a27967 Created on: 11/6/2024 11:20:47 AM.

Definition at line 20 of file Category.cs.

6.9.2 Constructor & Destructor Documentation

6.9.2.1 Category() [1/2]

```
trabalhoPOO_27967.Category.Category ()
```

The default Constructor.

Definition at line 35 of file Category.cs.

6.9.2.2 Category() [2/2]

```
trabalhoPOO_27967.Category.Category (
            string name)
```

The constructor to use when the name of a category is given.

Parameters

name	

Definition at line 45 of file Category.cs.

6.9.3 Member Function Documentation

6.9.3.1 Equals()

```
override bool trabalhoPOO_27967.Category.Equals (
            object obj)
```

Method that overrides Equals() and verifies if a categry matches another one.

Parameters

obj	

Returns

Veriffies if the object given is null.

Casts the object to be Category.

Definition at line 81 of file Category.cs.

6.9.3.2 operator"!=()

```
static bool trabalhoPOO_27967.Category.operator!= (
            Category cat1,
            Category cat2)  [static]
```

Redefinition of the Not Equal Operator.

Parameters

cat1	
cat2	

Returns

Definition at line 111 of file Category.cs.

6.9.3.3 operator==()

```
static bool trabalhoPOO_27967.Category.operator== (
            Category cat1,
            Category cat2)  [static]
```

Redefinition of the Equal operator.

Parameters

cat1	
cat2	

Returns

Definition at line 100 of file Category.cs.

6.9.4 Property Documentation

6.9.4.1 Id

```
int trabalhoPOO_27967.Category.Id  [get], [set]
```

The property to get and set the ID of a category.

Definition at line 57 of file Category.cs.

6.9.4.2 Name

```
string trabalhoPOO_27967.Category.Name  [get], [set]
```

The property to get and set the name of a Category.

Definition at line 66 of file Category.cs.

The documentation for this class was generated from the following file:

- trabalhoPOO_27967/Category/Category.cs

6.10 Data_BestSale.Client Class Reference

Purpose: Definition of Client and methods to deal with Client operations. Created by: Jose Alves a27967 Created on: 10/29/2024 4:23:56 PM.

Public Member Functions

- Client ()

  The default Constructor.
- Client (string n, string c)

  Constructor to use when a name and a contact are given.
- override string ToString ()

  Redefine the ToString Function to show a client's info.
- override bool Equals (object obj)

  Redefine the Equals operator to verify if a client matches the other.

Static Public Member Functions

- static bool operator== (Client cli1, Client cli2)

  Redefinition of the == operator.
- static bool operator!= (Client cli1, Client cli2)

  Redefinition of the != operator.
- static bool CreateClientFromNameContact (string name, string contact, out Client newClient)

Properties

- int ClientID `[get, set]`

  Property that sets or returns the ID of a client.
- string Name `[get, set]`

  Property that sets or returns the Name of a client.
- string Contact `[get, set]`

  Property that sets or returns the Contact of a client.
- static int ClientCount `[get, set]`

  Property that sets or returns the amount of clients.

6.10.1 Detailed Description

Purpose: Definition of Client and methods to deal with Client operations. Created by: Jose Alves a27967 Created on: 10/29/2024 4:23:56 PM.

Definition at line 28 of file Client.cs.

6.10.2 Constructor & Destructor Documentation

6.10.2.1 Client() `[1/2]`

```
Data_BestSale.Client.Client ()
```

The default Constructor.

Definition at line 44 of file Client.cs.

6.10.2.2 Client() [2/2]

```
Data_BestSale.Client.Client (
            string n,
            string c)
```

Constructor to use when a name and a contact are given.

Parameters

n	
c	

Definition at line 56 of file Client.cs.

6.10.3 Member Function Documentation

6.10.3.1 CreateClientFromNameContact()

```
static bool Data_BestSale.Client.CreateClientFromNameContact (
            string name,
            string contact,
            out Client newClient)  [static]
```

Definition at line 180 of file Client.cs.

6.10.3.2 Equals()

```
override bool Data_BestSale.Client.Equals (
            object obj)
```

Redefine the Equals operator to verify if a client matches the other.

Parameters

obj	

Returns

Veriffies if the object given is null.

Casts the object to be Client.

Definition at line 142 of file Client.cs.

6.10.3.3 operator"!=()

```
static bool Data_BestSale.Client.operator!= (
            Client cli1,
            Client cli2)  [static]
```

Redefinition of the != operator.

Parameters

cli1	
cli2	

Returns

Definition at line 172 of file Client.cs.

6.10.3.4 operator==()

```
static bool Data_BestSale.Client.operator== (
            Client cli1,
            Client cli2)  [static]
```

Redefinition of the == operator.

Parameters

cli1	
cli2	

Returns

Definition at line 161 of file Client.cs.

6.10.3.5 ToString()

```
override string Data_BestSale.Client.ToString ()
```

Redefine the ToString Function to show a client's info.

Returns

Definition at line 127 of file Client.cs.

6.10.4 Property Documentation

6.10.4.1 ClientCount

```
int Data_BestSale.Client.ClientCount  [static], [get], [set]
```

Property that sets or returns the amount of clients.

Definition at line 111 of file Client.cs.

6.10.4.2 ClientID

```
int Data_BestSale.Client.ClientID  [get], [set]
```

Property that sets or returns the ID of a client.

Definition at line 70 of file Client.cs.

6.10.4.3 Contact

```
string Data_BestSale.Client.Contact  [get], [set]
```

Property that sets or returns the Contact of a client.

Definition at line 88 of file Client.cs.

6.10.4.4 Name

```
string Data_BestSale.Client.Name  [get], [set]
```

Property that sets or returns the Name of a client.

Definition at line 79 of file Client.cs.

The documentation for this class was generated from the following file:

- Data_BestSale/Client/Client.cs

6.11 trabalhoPOO_27967.Client Class Reference

Purpose: Definition of Client and methods to deal with Client operations. Created by: Jose Alves a27967 Created on: 10/29/2024 4:23:56 PM.

Public Member Functions

- Client ()
  The default Constructor.
- Client (string n, string c)
  Constructor to use when a name and a contact are given.
- override string ToString ()
  Redefine the ToString Function to show a client's info.
- override bool Equals (object obj)
  Redefine the Equals operator to verify if a client matches the other.

Static Public Member Functions

- static bool operator== (Client cli1, Client cli2)

  Redefinition of the == operator.
- static bool operator!= (Client cli1, Client cli2)

  Redefinition of the != operator.

Properties

- int ClientID `[get, set]`

  Property that sets or returns the ID of a client.
- string Name `[get, set]`

  Property that sets or returns the Name of a client.
- string Contact `[get, set]`

  Property that sets or returns the Contact of a client.
- static int ClientCount `[get, set]`

  Property that sets or returns the amount of clients.

6.11.1 Detailed Description

Purpose: Definition of Client and methods to deal with Client operations. Created by: Jose Alves a27967 Created on: 10/29/2024 4:23:56 PM.

Definition at line 23 of file Client.cs.

6.11.2 Constructor & Destructor Documentation

6.11.2.1 Client() [1/2]

```
trabalhoPOO_27967.Client.Client ()
```

The default Constructor.

Definition at line 39 of file Client.cs.

6.11.2.2 Client() [2/2]

```
trabalhoPOO_27967.Client.Client (
            string n,
            string c)
```

Constructor to use when a name and a contact are given.

Parameters

n	
c	

Definition at line 51 of file Client.cs.

6.11.3 Member Function Documentation

6.11.3.1 Equals()

```
override bool trabalhoPOO_27967.Client.Equals (
            object obj)
```

Redefine the Equals operator to verify if a client matches the other.

Parameters

obj	

Returns

Veriffies if the object given is null.

Casts the object to be Client.

Definition at line 131 of file Client.cs.

6.11.3.2 operator"!=()

```
static bool trabalhoPOO_27967.Client.operator!= (
            Client cli1,
            Client cli2)  [static]
```

Redefinition of the != operator.

Parameters

cli1	
cli2	

Returns

Definition at line 161 of file Client.cs.

6.11.3.3 operator==()

```
static bool trabalhoPOO_27967.Client.operator== (
            Client cli1,
            Client cli2)  [static]
```

Redefinition of the == operator.

Parameters

cli1	
cli2	

Returns

Definition at line 150 of file Client.cs.

6.11.3.4 ToString()

```
override string trabalhoPOO_27967.Client.ToString ()
```

Redefine the ToString Function to show a client's info.

Returns

Definition at line 116 of file Client.cs.

6.11.4 Property Documentation

6.11.4.1 ClientCount

```
int trabalhoPOO_27967.Client.ClientCount  [static], [get], [set]
```

Property that sets or returns the amount of clients.

Definition at line 100 of file Client.cs.

6.11.4.2 ClientID

```
int trabalhoPOO_27967.Client.ClientID  [get], [set]
```

Property that sets or returns the ID of a client.

Definition at line 65 of file Client.cs.

6.11.4.3 Contact

```
string trabalhoPOO_27967.Client.Contact  [get], [set]
```

Property that sets or returns the Contact of a client.

Definition at line 83 of file Client.cs.

6.11.4.4 Name

```
string trabalhoPOO_27967.Client.Name  [get], [set]
```

Property that sets or returns the Name of a client.

Definition at line 74 of file Client.cs.

The documentation for this class was generated from the following file:

- trabalhoPOO_27967/Client/Client.cs

6.12 Data_BestSale.Clients Class Reference

Purpose: Class with the definition and methods to manage a list of clients. Created by: Jose Alves a27967 Created on: 11/12/2024 9:25:28 PM.

Inheritance diagram for Data_BestSale.Clients:

Collaboration diagram for Data_BestSale.Clients:

Public Member Functions

- Clients ()
  The default Constructor.
- bool Add (object obj)
  Method to add a client to a clients' list.
- bool Remove (object obj)
  Method to remove a client from the store's client list.
- bool Exist (object obj)
  Method to check if a client is listed on a clients' list.
- Client GetClient (int id)
  Method used to get a client from a clients' list.
- bool ClearClients ()
  Method used to Clear a list of Clients.

Public Member Functions inherited from Data_BestSale.IListManagement

Properties

- List< Client > ClientList `[get, set]`
  Property used to get and set the client list.

6.12.1 Detailed Description

Purpose: Class with the definition and methods to manage a list of clients. Created by: Jose Alves a27967 Created on: 11/12/2024 9:25:28 PM.

Definition at line 22 of file Clients.cs.

6.12.2 Constructor & Destructor Documentation

6.12.2.1 Clients()

```
Data_BestSale.Clients.Clients ()
```

The default Constructor.

Definition at line 35 of file Clients.cs.

6.12.3 Member Function Documentation

6.12.3.1 Add()

```
bool Data_BestSale.Clients.Add (
            object obj)
```

Method to add a client to a clients' list.

Parameters

cli	

Returns

True - Client has been successfully added to the list.

False - The list already contains the client or an error occurred.

Returns

Implements Data_BestSale.IListManagement.

Definition at line 67 of file Clients.cs.

6.12.3.2 ClearClients()

```
bool Data_BestSale.Clients.ClearClients ()
```

Method used to Clear a list of Clients.

Definition at line 139 of file Clients.cs.

6.12.3.3 Exist()

```
bool Data_BestSale.Clients.Exist (
            object obj)
```

Method to check if a client is listed on a clients' list.

Parameters

id	

Returns

Implements Data_BestSale.IListManagement.

Definition at line 104 of file Clients.cs.

6.12.3.4 GetClient()

```
Client Data_BestSale.Clients.GetClient (
            int id)
```

Method used to get a client from a clients' list.

Parameters

id	

Returns

Definition at line 124 of file Clients.cs.

6.12.3.5 Remove()

```
bool Data_BestSale.Clients.Remove (
            object obj)
```

Method to remove a client from the store's client list.

Parameters

cli	

Returns

Implements Data_BestSale.IListManagement.

Definition at line 87 of file Clients.cs.

6.12.4 Property Documentation

6.12.4.1 ClientList

```
List<Client> Data_BestSale.Clients.ClientList  [get], [set]
```

Property used to get and set the client list.

Definition at line 48 of file Clients.cs.

The documentation for this class was generated from the following file:

- Data_BestSale/Client/Clients.cs

6.13 trabalhoPOO_27967.Clients Class Reference

Purpose: Class with the definition and methods to manage a list of clients. Created by: Jose Alves a27967 Created on: 11/12/2024 9:25:28 PM.

Inheritance diagram for trabalhoPOO_27967.Clients:

Collaboration diagram for trabalhoPOO_27967.Clients:

Public Member Functions

- Clients ()

  The default Constructor.
- bool Add (object obj)

  Method to add a client to the store's client list.
- bool Remove (object obj)

  Method to remove a client from the store's client list.
- bool Exist (object obj)

  Method to check if a client is listed on a clients' list.
- Client GetClient (int id)

  Method used to get a client from a clients' list.

Public Member Functions inherited from trabalhoPOO_27967.Interface.IListManagement

Properties

- List< Client > ClientList `[get, set]`

  Property used to get and set the client list.

6.13.1 Detailed Description

Purpose: Class with the definition and methods to manage a list of clients. Created by: Jose Alves a27967 Created on: 11/12/2024 9:25:28 PM.

Definition at line 22 of file Clients.cs.

6.13.2 Constructor & Destructor Documentation

6.13.2.1 Clients()

```
trabalhoPOO_27967.Clients.Clients ()
```

The default Constructor.

Definition at line 35 of file Clients.cs.

6.13.3 Member Function Documentation

6.13.3.1 Add()

```
bool trabalhoPOO_27967.Clients.Add (
            object obj)
```

Method to add a client to the store's client list.

Parameters

cli	

Returns

Implements trabalhoPOO_27967.Interface.IListManagement.

Definition at line 66 of file Clients.cs.

6.13.3.2 Exist()

```
bool trabalhoPOO_27967.Clients.Exist (
            object obj)
```

Method to check if a client is listed on a clients' list.

Parameters

id	

Returns

Implements trabalhoPOO_27967.Interface.IListManagement.

Definition at line 99 of file Clients.cs.

6.13.3.3 GetClient()

```
Client trabalhoPOO_27967.Clients.GetClient (
            int id)
```

Method used to get a client from a clients' list.

Parameters

id	

Returns

Definition at line 119 of file Clients.cs.

6.13.3.4 Remove()

```
bool trabalhoPOO_27967.Clients.Remove (
            object obj)
```

Method to remove a client from the store's client list.

Parameters

cli	

Returns

Implements trabalhoPOO_27967.Interface.IListManagement.

Definition at line 82 of file Clients.cs.

6.13.4 Property Documentation

6.13.4.1 ClientList

```
List<Client> trabalhoPOO_27967.Clients.ClientList  [get], [set]
```

Property used to get and set the client list.

Definition at line 48 of file Clients.cs.

The documentation for this class was generated from the following file:

- trabalhoPOO_27967/Client/Clients.cs

6.14 Data_BestSale.IListManagement Interface Reference

Inheritance diagram for Data_BestSale.IListManagement:

Public Member Functions

- bool Add (object obj)
- bool Remove (object obj)
- bool Exist (object obj)

6.14.1 Detailed Description

Definition at line 17 of file IListManagement.cs.

6.14.2 Member Function Documentation

6.14.2.1 Add()

```
bool Data_BestSale.IListManagement.Add (
            object obj)
```

Implemented in Data_BestSale.Campaigns, Data_BestSale.Categories, Data_BestSale.Clients, Data_BestSale.Makes, Data_BestSale.Products, Data_BestSale.Sales, and Data_BestSale.Warranties.

6.14.2.2 Exist()

```
bool Data_BestSale.IListManagement.Exist (
            object obj)
```

Implemented in Data_BestSale.Campaigns, Data_BestSale.Categories, Data_BestSale.Clients, Data_BestSale.Makes, Data_BestSale.Products, Data_BestSale.Sales, and Data_BestSale.Warranties.

6.14.2.3 Remove()

```
bool Data_BestSale.IListManagement.Remove (
            object obj)
```

Implemented in Data_BestSale.Campaigns, Data_BestSale.Categories, Data_BestSale.Clients, Data_BestSale.Makes, Data_BestSale.Products, Data_BestSale.Sales, and Data_BestSale.Warranties.

The documentation for this interface was generated from the following file:

- Data_BestSale/Interface/IListManagement.cs

6.15 trabalhoPOO_27967.Interface.IListManagement Interface Reference

Inheritance diagram for trabalhoPOO_27967.Interface.IListManagement:

Public Member Functions

- bool Add (object obj)
- bool Remove (object obj)
- bool Exist (object obj)

6.15.1 Detailed Description

Definition at line 17 of file IListManagement.cs.

6.15.2 Member Function Documentation

6.15.2.1 Add()

```
bool trabalhoPOO_27967.Interface.IListManagement.Add (
            object obj)
```

Implemented in trabalhoPOO_27967.Campaigns, trabalhoPOO_27967.Categories, trabalhoPOO_27967.Clients, trabalhoPOO_27967.Makes, trabalhoPOO_27967.Products, trabalhoPOO_27967.Sales, and trabalhoPOO_27967.Warranties.

6.15.2.2 Exist()

```
bool trabalhoPOO_27967.Interface.IListManagement.Exist (
            object obj)
```

Implemented in trabalhoPOO_27967.Campaigns, trabalhoPOO_27967.Categories, trabalhoPOO_27967.Clients, trabalhoPOO_27967.Makes, trabalhoPOO_27967.Products, trabalhoPOO_27967.Sales, and trabalhoPOO_27967.Warranties.

6.15.2.3 Remove()

```
bool trabalhoPOO_27967.Interface.IListManagement.Remove (
            object obj)
```

Implemented in trabalhoPOO_27967.Campaigns, trabalhoPOO_27967.Categories, trabalhoPOO_27967.Clients, trabalhoPOO_27967.Makes, trabalhoPOO_27967.Products, trabalhoPOO_27967.Sales, and trabalhoPOO_27967.Warranties.

The documentation for this interface was generated from the following file:

- trabalhoPOO_27967/Interface/IListManagement.cs

6.16 Exceptions.InvalidPhoneNumberException Class Reference

The exception to be throws when a string doesn't match the phone number pattern.

Inheritance diagram for Exceptions.InvalidPhoneNumberException:

Collaboration diagram for Exceptions.InvalidPhoneNumberException:

Public Member Functions

- InvalidPhoneNumberException (string message)
- InvalidPhoneNumberException (string message, Exception e)

6.16.1 Detailed Description

The exception to be throws when a string doesn't match the phone number pattern.

Definition at line 21 of file InvalidPhoneNumberException.cs.

6.16.2 Constructor & Destructor Documentation

6.16.2.1 InvalidPhoneNumberException() [1/3]

```
Exceptions.InvalidPhoneNumberException.InvalidPhoneNumberException ()
```

Definition at line 23 of file InvalidPhoneNumberException.cs.

6.16.2.2 InvalidPhoneNumberException() [2/3]

```
Exceptions.InvalidPhoneNumberException.InvalidPhoneNumberException (
            string message)
```

Definition at line 25 of file InvalidPhoneNumberException.cs.

6.16.2.3 InvalidPhoneNumberException() [3/3]

```
Exceptions.InvalidPhoneNumberException.InvalidPhoneNumberException (
            string message,
            Exception e)
```

Definition at line 27 of file InvalidPhoneNumberException.cs.

The documentation for this class was generated from the following file:

- Exceptions/InvalidPhoneNumberException.cs

6.17 Data_BestSale.Make Class Reference

Purpose: Definition of Make and methods to deal with Make operations. Created by: Jose Alves a27967 Created on: 11/6/2024 11:22:09 AM.

Public Member Functions

- Make ()
  The default Constructor.
- Make (string name)
  Constructor when the name of the make is given.
- override string ToString ()
  Override of the ToString() method to convert the data of a Make to a string.
- override bool Equals (object obj)
  The redefinition of the Equals() Method, to verify if a make matches another one.
- int GetMakeID ()
  Method used to get the ID of a make.

Static Public Member Functions

- static bool operator== (Make m1, Make m2)
  The redefinition of the Equal operator.
- static bool operator!= (Make m1, Make m2)
  The redefinition of the NOt Equal operator.
- static bool CreateMake (string name, out Make make)
  Method to create a new make, given its name.

Properties

- int ID `[get, set]`

  Property to set and get the ID of a Make.
- string Name `[get, set]`

  Property to get and set the Name of a Make.

6.17.1 Detailed Description

Purpose: Definition of Make and methods to deal with Make operations. Created by: Jose Alves a27967 Created on: 11/6/2024 11:22:09 AM.

Definition at line 24 of file Make.cs.

6.17.2 Constructor & Destructor Documentation

6.17.2.1 Make() [1/2]

```
Data_BestSale.Make.Make ()
```

The default Constructor.

Definition at line 39 of file Make.cs.

6.17.2.2 Make() [2/2]

```
Data_BestSale.Make.Make (
            string name)
```

Constructor when the name of the make is given.

Parameters

id	
name	

Definition at line 50 of file Make.cs.

6.17.3 Member Function Documentation

6.17.3.1 CreateMake()

```
static bool Data_BestSale.Make.CreateMake (
            string name,
            out Make make)  [static]
```

Method to create a new make, given its name.

Parameters

name	
make	

Returns

Definition at line 142 of file Make.cs.

6.17.3.2 Equals()

```
override bool Data_BestSale.Make.Equals (
            object obj)
```

The redefinition of the Equals() Method, to verify if a make matches another one.

Parameters

obj	

Returns

Veriffies if the object given is null.

Casts the object to be Make.

Definition at line 99 of file Make.cs.

6.17.3.3 GetMakeID()

```
int Data_BestSale.Make.GetMakeID ()
```

Method used to get the ID of a make.

Returns

The ID of the make.

Definition at line 159 of file Make.cs.

6.17.3.4 operator"!=()

```
static bool Data_BestSale.Make.operator!= (
            Make m1,
            Make m2)  [static]
```

The redefinition of the NOt Equal operator.

Parameters

m1	
m2	

Returns

Definition at line 129 of file Make.cs.

6.17.3.5 operator==()

```
static bool Data_BestSale.Make.operator== (
            Make m1,
            Make m2)  [static]
```

The redefinition of the Equal operator.

Parameters

m1	
m2	

Returns

Definition at line 118 of file Make.cs.

6.17.3.6 ToString()

```
override string Data_BestSale.Make.ToString ()
```

Override of the ToString() method to convert the data of a Make to a string.

Returns

Definition at line 89 of file Make.cs.

6.17.4 Property Documentation

6.17.4.1 ID

```
int Data_BestSale.Make.ID  [get], [set]
```

Property to set and get the ID of a Make.

Definition at line 65 of file Make.cs.

6.17.4.2 Name

```
string Data_BestSale.Make.Name  [get], [set]
```

Property to get and set the Name of a Make.

Definition at line 74 of file Make.cs.

The documentation for this class was generated from the following file:

- Data_BestSale/Make/Make.cs

6.18 trabalhoPOO_27967.Make Class Reference

Purpose: Definition of Make and methods to deal with Make operations. Created by: Jose Alves a27967 Created on: 11/6/2024 11:22:09 AM.

Public Member Functions

- Make ()
  The default Constructor.
- Make (string name)
  Constructor when the name of the make is given.
- override string ToString ()
  Override of the ToString() method to convert the data of a Make to a string.
- override bool Equals (object obj)
  The redefinition of the Equals() Method, to verify if a make matches another one.

Static Public Member Functions

- static bool operator== (Make m1, Make m2)
  The redefinition of the Equal operator.
- static bool operator!= (Make m1, Make m2)
  The redefinition of the NOt Equal operator.

Properties

- int ID `[get, set]`
  Property to set and get the ID of a Make.
- string Name `[get, set]`
  Property to get and set the Name of a Make.

6.18.1 Detailed Description

Purpose: Definition of Make and methods to deal with Make operations. Created by: Jose Alves a27967 Created on: 11/6/2024 11:22:09 AM.

Definition at line 23 of file Make.cs.

6.18.2 Constructor & Destructor Documentation

6.18.2.1 Make() [1/2]

```
trabalhoPOO_27967.Make.Make ()
```

The default Constructor.

Definition at line 38 of file Make.cs.

6.18.2.2 Make() [2/2]

```
trabalhoPOO_27967.Make.Make (
            string name)
```

Constructor when the name of the make is given.

Parameters

id	
name	

Definition at line 49 of file Make.cs.

6.18.3 Member Function Documentation

6.18.3.1 Equals()

```
override bool trabalhoPOO_27967.Make.Equals (
            object obj)
```

The redefinition of the Equals() Method, to verify if a make matches another one.

Parameters

obj	

Returns

Veriffies if the object given is null.

Casts the object to be Make.

Definition at line 98 of file Make.cs.

6.18.3.2 operator"!=()

```
static bool trabalhoPOO_27967.Make.operator!= (
            Make m1,
            Make m2)  [static]
```

The redefinition of the NOt Equal operator.

Parameters

m1	
m2	

Returns

Definition at line 128 of file Make.cs.

6.18.3.3 operator==()

```
static bool trabalhoPOO_27967.Make.operator== (
            Make m1,
            Make m2)  [static]
```

The redefinition of the Equal operator.

Parameters

m1	
m2	

Returns

Definition at line 117 of file Make.cs.

6.18.3.4 ToString()

```
override string trabalhoPOO_27967.Make.ToString ()
```

Override of the ToString() method to convert the data of a Make to a string.

Returns

Definition at line 88 of file Make.cs.

6.18.4 Property Documentation

6.18.4.1 ID

```
int trabalhoPOO_27967.Make.ID  [get], [set]
```

Property to set and get the ID of a Make.

Definition at line 64 of file Make.cs.

6.18.4.2 Name

```
string trabalhoPOO_27967.Make.Name  [get], [set]
```

Property to get and set the Name of a Make.

Definition at line 73 of file Make.cs.

The documentation for this class was generated from the following file:

- trabalhoPOO_27967/Make/Make.cs

6.19 Data_BestSale.Makes Class Reference

Purpose:This file has the definition and methods to work with the plurality of Make. Created by: Jose Alves a27967 Created on: 11/14/2024 4:33:51 PM.

Inheritance diagram for Data_BestSale.Makes:

Collaboration diagram for Data_BestSale.Makes:

Public Member Functions

- Makes ()
  The default Constructor.
- Makes (List< Make > m)
  The constructor to use when a list of Make is given.
- bool Add (object obj)
  Method used to add a make to a list of makes.
- bool Remove (object obj)
  Method used to remove a make from a list of makes.
- bool Exist (object obj)
  Method used to verify if a make exists on a list of makes, given its ID or name.
- bool ClearMakes ()
  Method used to Clear a list of Makes.
- Make GetMake (object obj)
  This method finds a make instance, given its ID or Name.

Public Member Functions inherited from Data_BestSale.IListManagement

Properties

- List< Make > MakeList [get, set]
  The property to get and set a list of Make.

6.19.1 Detailed Description

Purpose:This file has the definition and methods to work with the plurality of Make. Created by: Jose Alves a27967 Created on: 11/14/2024 4:33:51 PM.

Definition at line 24 of file Makes.cs.

6.19.2 Constructor & Destructor Documentation

6.19.2.1 Makes() [1/2]

```
Data_BestSale.Makes.Makes ()
```

The default Constructor.

Definition at line 37 of file Makes.cs.

6.19.2.2 Makes() [2/2]

```
Data_BestSale.Makes.Makes (
            List< Make > m)
```

The constructor to use when a list of Make is given.

Parameters

m	

Definition at line 46 of file Makes.cs.

6.19.3 Member Function Documentation

6.19.3.1 Add()

```
bool Data_BestSale.Makes.Add (
            object obj)
```

Method used to add a make to a list of makes.

Parameters

m	

Returns

Implements Data_BestSale.IListManagement.

Definition at line 74 of file Makes.cs.

6.19.3.2 ClearMakes()

```
bool Data_BestSale.Makes.ClearMakes ()
```

Method used to Clear a list of Makes.

Definition at line 135 of file Makes.cs.

6.19.3.3 Exist()

```
bool Data_BestSale.Makes.Exist (
            object obj)
```

Method used to verify if a make exists on a list of makes, given its ID or name.

Parameters

id	

Returns

Implements Data_BestSale.IListManagement.

Definition at line 106 of file Makes.cs.

6.19.3.4 GetMake()

```
Make Data_BestSale.Makes.GetMake (
            object obj)
```

This method finds a make instance, given its ID or Name.

Parameters

id	The Make ID

Returns

The make instance

Definition at line 152 of file Makes.cs.

6.19.3.5 Remove()

```
bool Data_BestSale.Makes.Remove (
            object obj)
```

Method used to remove a make from a list of makes.

Parameters

m	

Returns

Implements Data_BestSale.IListManagement.

Definition at line 89 of file Makes.cs.

6.19.4 Property Documentation

6.19.4.1 MakeList

```
List<Make> Data_BestSale.Makes.MakeList  [get], [set]
```

The property to get and set a list of Make.

Definition at line 56 of file Makes.cs.

The documentation for this class was generated from the following file:

- Data_BestSale/Make/Makes.cs

6.20 trabalhoPOO_27967.Makes Class Reference

Purpose:This file has the definition and methods to work with the plurality of Make. Created by: Jose Alves a27967 Created on: 11/14/2024 4:33:51 PM.

Inheritance diagram for trabalhoPOO_27967.Makes:

Collaboration diagram for trabalhoPOO_27967.Makes:

Public Member Functions

- Makes ()
  The default Constructor.
- Makes (List< Make > m)
  The constructor to use when a list of Make is given.
- bool Add (object obj)
  Method used to add a make to a list of makes.
- bool Remove (object obj)
  Method used to remove a make from a list of makes.
- bool Exist (object obj)
  Method used to verify if a make exists on a list of makes, given its ID or name.

Public Member Functions inherited from trabalhoPOO_27967.Interface.IListManagement

Properties

- List< Make > MakeList `[get, set]`

  The property to get and set a list of Make.

6.20.1 Detailed Description

Purpose:This file has the definition and methods to work with the plurality of Make. Created by: Jose Alves a27967 Created on: 11/14/2024 4:33:51 PM.

Definition at line 23 of file Makes.cs.

6.20.2 Constructor & Destructor Documentation

6.20.2.1 Makes() [1/2]

```
trabalhoPOO_27967.Makes.Makes ()
```

The default Constructor.

Definition at line 36 of file Makes.cs.

6.20.2.2 Makes() [2/2]

```
trabalhoPOO_27967.Makes.Makes (
            List< Make > m)
```

The constructor to use when a list of Make is given.

Parameters

m	

Definition at line 45 of file Makes.cs.

6.20.3 Member Function Documentation

6.20.3.1 Add()

```
bool trabalhoPOO_27967.Makes.Add (
            object obj)
```

Method used to add a make to a list of makes.

Parameters

m	

Returns

Implements trabalhoPOO_27967.Interface.IListManagement.

Definition at line 73 of file Makes.cs.

6.20.3.2 Exist()

```
bool trabalhoPOO_27967.Makes.Exist (
            object obj)
```

Method used to verify if a make exists on a list of makes, given its ID or name.

Parameters

id	

Returns

Implements trabalhoPOO_27967.Interface.IListManagement.

Definition at line 105 of file Makes.cs.

6.20.3.3 Remove()

```
bool trabalhoPOO_27967.Makes.Remove (
            object obj)
```

Method used to remove a make from a list of makes.

Parameters

m	

Returns

Implements trabalhoPOO_27967.Interface.IListManagement.

Definition at line 88 of file Makes.cs.

6.20.4 Property Documentation

6.20.4.1 MakeList

```
List<Make> trabalhoPOO_27967.Makes.MakeList  [get], [set]
```

The property to get and set a list of Make.

Definition at line 55 of file Makes.cs.

The documentation for this class was generated from the following file:

- trabalhoPOO_27967/Make/Makes.cs

6.21 Data_BestSale.Product Class Reference

Purpose: Definition of product and methods to deal with product operations. Created by: Jose Alves a27967 Created on: 11/2/2024 4:40:12 PM.

Public Member Functions

- Product ()
  The default Constructor.
- Product (string reff, decimal price, int makeID, int categoryID)
  The constructor to use when reference, price, makeID and categoryID are given.
- Product (string reff, decimal price, Warranty warranty, int make, int category)
  Constructor for when the reference, price and warranty duration are given.
- override bool Equals (object obj)
  Redefinition of the method to compare two products.
- override string ToString ()
  Override of the ToString() Method to convert the data of a product into a string.

Static Public Member Functions

- static bool operator== (Product p1, Product p2)
  Redefinition of the equal operator.
- static bool operator!= (Product p1, Product p2)
  Redefinition of the different operator.
- static Product CreateProductWithWarranty (string reff, decimal price, int makeID, int categoryID, int warrantyDuration, string warrantyConditions)
  Method that creates a product and its warranty.

Properties

- string Reference `[get, set]`

  Property to set and get the reference of a product.
- decimal Price `[get, set]`

  Property to get and set the price of a product.
- int MakeID `[get, set]`

  Property to set and get the Make of a product.
- int CategoryID `[get, set]`

  Property to set and get the Category of a product.
- int Stock `[get, set]`

  Property to get and set the existing stock of a product.
- Warranty Warranty `[get, set]`

6.21.1 Detailed Description

Purpose: Definition of product and methods to deal with product operations. Created by: Jose Alves a27967 Created on: 11/2/2024 4:40:12 PM.

Definition at line 28 of file Product.cs.

6.21.2 Constructor & Destructor Documentation

6.21.2.1 Product() [1/3]

```
Data_BestSale.Product.Product ()
```

The default Constructor.

Definition at line 46 of file Product.cs.

6.21.2.2 Product() [2/3]

```
Data_BestSale.Product.Product (
            string reff,
            decimal price,
            int makeID,
            int categoryID)
```

The constructor to use when reference, price, makeID and categoryID are given.

Parameters

reff	
price	
makeID	
categoryID	

Definition at line 62 of file Product.cs.

6.21.2.3 Product() [3/3]

```
Data_BestSale.Product.Product (
            string reff,
            decimal price,
            Warranty warranty,
            int make,
            int category)
```

Constructor for when the reference, price and warranty duration are given.

Parameters

reff	
pri	

Definition at line 76 of file Product.cs.

6.21.3 Member Function Documentation

6.21.3.1 CreateProductWithWarranty()

```
static Product Data_BestSale.Product.CreateProductWithWarranty (
            string reff,
            decimal price,
            int makeID,
            int categoryID,
            int warrantyDuration,
            string warrantyConditions)  [static]
```

Method that creates a product and its warranty.

Parameters

reff	The reference of the product
price	The price of the product
makeID	The ID of the make of the product
categoryID	The ID of the category of the product
warrantyDuration	The duration, in years, of the warranty
warrantyConditions	The terms of the warranty

Returns

The instance of product created.

Definition at line 210 of file Product.cs.

6.21.3.2 Equals()

```
override bool Data_BestSale.Product.Equals (
            object obj)
```

Redefinition of the method to compare two products.

Parameters

obj	

Returns

Definition at line 152 of file Product.cs.

6.21.3.3 operator"!=()

```
static bool Data_BestSale.Product.operator!= (
            Product p1,
            Product p2)  [static]
```

Redefinition of the different operator.

Parameters

p1	
p2	

Returns

Definition at line 177 of file Product.cs.

6.21.3.4 operator==()

```
static bool Data_BestSale.Product.operator== (
            Product p1,
            Product p2)  [static]
```

Redefinition of the equal operator.

Parameters

p1	
p2	

Returns

Definition at line 166 of file Product.cs.

6.21.3.5 ToString()

```
override string Data_BestSale.Product.ToString ()
```

Override of the ToString() Method to convert the data of a product into a string.

Returns

Definition at line 186 of file Product.cs.

6.21.4 Property Documentation

6.21.4.1 CategoryID

```
int Data_BestSale.Product.CategoryID  [get], [set]
```

Property to set and get the Category of a product.

Definition at line 122 of file Product.cs.

6.21.4.2 MakeID

```
int Data_BestSale.Product.MakeID  [get], [set]
```

Property to set and get the Make of a product.

Definition at line 112 of file Product.cs.

6.21.4.3 Price

```
decimal Data_BestSale.Product.Price  [get], [set]
```

Property to get and set the price of a product.

Definition at line 103 of file Product.cs.

6.21.4.4 Reference

```
string Data_BestSale.Product.Reference  [get], [set]
```

Property to set and get the reference of a product.

Definition at line 93 of file Product.cs.

6.21.4.5 Stock

```
int Data_BestSale.Product.Stock  [get], [set]
```

Property to get and set the existing stock of a product.

Definition at line 131 of file Product.cs.

6.21.4.6 Warranty

```
Warranty Data_BestSale.Product.Warranty  [get], [set]
```

Definition at line 137 of file Product.cs.

The documentation for this class was generated from the following file:

- Data_BestSale/Product/Product.cs

6.22 trabalhoPOO_27967.Product Class Reference

Purpose: Definition of product and methods to deal with product operations. Created by: Jose Alves a27967 Created on: 11/2/2024 4:40:12 PM.

Public Member Functions

- Product ()
  The default Constructor.
- Product (string reff, decimal price, Warranty warranty, int make, int category)
  Constructor for when the reference, price and warranty duration are given.
- override bool Equals (object obj)
  Redefinition of the method to compare two products.
- override string ToString ()
  Override of the ToString() Method to convert the data of a product into a string.

Static Public Member Functions

- static bool operator== (Product p1, Product p2)
  Redefinition of the equal operator.
- static bool operator!= (Product p1, Product p2)
  Redefinition of the different operator.

Properties

- string Reference [get, set]
  Property to set and get the reference of a product.
- decimal Price [get, set]
  Property to get and set the price of a product.
- int Make [get, set]
  Property to set and get the Make of a product.
- int Category [get, set]
  Property to set and get the Category of a product.
- int Stock [get, set]
  Property to get and set the existing stock of a product.
- Warranty Warranty [get, set]

6.22.1 Detailed Description

Purpose: Definition of product and methods to deal with product operations. Created by: Jose Alves a27967 Created on: 11/2/2024 4:40:12 PM.

Definition at line 25 of file Product.cs.

6.22.2 Constructor & Destructor Documentation

6.22.2.1 Product() [1/2]

```
trabalhoPOO_27967.Product.Product ()
```

The default Constructor.

Definition at line 43 of file Product.cs.

6.22.2.2 Product() [2/2]

```
trabalhoPOO_27967.Product.Product (
            string reff,
            decimal price,
            Warranty warranty,
            int make,
            int category)
```

Constructor for when the reference, price and warranty duration are given.

Parameters

reff	
pri	

Definition at line 57 of file Product.cs.

6.22.3 Member Function Documentation

6.22.3.1 Equals()

```
override bool trabalhoPOO_27967.Product.Equals (
            object obj)
```

Redefinition of the method to compare two products.

Parameters

obj	

Returns

Definition at line 133 of file Product.cs.

6.22.3.2 operator"!=()

```
static bool trabalhoPOO_27967.Product.operator!= (
            Product p1,
            Product p2)  [static]
```

Redefinition of the different operator.

Parameters

p1	
p2	

Returns

Definition at line 158 of file Product.cs.

6.22.3.3 operator==()

```
static bool trabalhoPOO_27967.Product.operator== (
            Product p1,
            Product p2)  [static]
```

Redefinition of the equal operator.

Parameters

p1	
p2	

Returns

Definition at line 147 of file Product.cs.

6.22.3.4 ToString()

```
override string trabalhoPOO_27967.Product.ToString ()
```

Override of the ToString() Method to convert the data of a product into a string.

Returns

Definition at line 167 of file Product.cs.

6.22.4 Property Documentation

6.22.4.1 Category

```
int trabalhoPOO_27967.Product.Category  [get], [set]
```

Property to set and get the Category of a product.

Definition at line 103 of file Product.cs.

6.22.4.2 Make

```
int trabalhoPOO_27967.Product.Make  [get], [set]
```

Property to set and get the Make of a product.

Definition at line 93 of file Product.cs.

6.22.4.3 Price

```
decimal trabalhoPOO_27967.Product.Price  [get], [set]
```

Property to get and set the price of a product.

Definition at line 84 of file Product.cs.

6.22.4.4 Reference

```
string trabalhoPOO_27967.Product.Reference  [get], [set]
```

Property to set and get the reference of a product.

Definition at line 74 of file Product.cs.

6.22.4.5 Stock

```
int trabalhoPOO_27967.Product.Stock  [get], [set]
```

Property to get and set the existing stock of a product.

Definition at line 112 of file Product.cs.

6.22.4.6 Warranty

```
Warranty trabalhoPOO_27967.Product.Warranty  [get], [set]
```

Definition at line 118 of file Product.cs.

The documentation for this class was generated from the following file:

- trabalhoPOO_27967/Product/Product.cs

6.23 Data_BestSale.Products Class Reference

Purpose: Class to manage a group of more than one product. Created by: Jose Alves a27967 Created on↩ : 11/9/2024 6:34:19 PM.

Inheritance diagram for Data_BestSale.Products:

Collaboration diagram for Data_BestSale.Products:

Public Member Functions

- Products ()
  The default Constructor.
- Products (List< Product > products)
  The constructor to use when list of Product is given.
- override string ToString ()
  Override of the ToString() Method to convert the data of a list fo products to a string.
- decimal ValueInPosition (int p)
  This method returns the price of a product, given a certain array position.
- bool Add (object obj)
  This method inserts a product in a list of products.
- bool Exist (object obj)
  Method used to verify if a product is on a products' list, given its Refference.
- bool Remove (object obj)
  Method used to remove a product from a Products' list.
- Product SearchProduct (string reff)
  This method searches for a product in an array, given its refference.
- decimal TotalPrice ()
  Method used to calculate the total price of products in a list of products.
- DateTime WarratyExpirationDateForProduct (DateTime date, string reff)
  Method used to calculate the warranty's expiration date of a product on a list of products.
- bool ClearProducts ()
  Method used to Clear a list of Products.

Public Member Functions inherited from Data_BestSale.IListManagement

Properties

- List< Product > Prods [get, set]
  Property used to get and set the list of products.

6.23.1 Detailed Description

Purpose: Class to manage a group of more than one product. Created by: Jose Alves a27967 Created on↩ : 11/9/2024 6:34:19 PM.

Definition at line 26 of file Products.cs.

6.23.2 Constructor & Destructor Documentation

6.23.2.1 Products() [1/2]

```
Data_BestSale.Products.Products ()
```

The default Constructor.

Definition at line 39 of file Products.cs.

6.23.2.2 Products() [2/2]

```
Data_BestSale.Products.Products (
            List< Product > products)
```

The constructor to use when list of Product is given.

Parameters

products	

Definition at line 48 of file Products.cs.

6.23.3 Member Function Documentation

6.23.3.1 Add()

```
bool Data_BestSale.Products.Add (
            object obj)
```

This method inserts a product in a list of products.

Parameters

p	

Returns

Returns true or False, depending on whether or not it succeeded in inserting the product into the list.

Implements Data_BestSale.IListManagement.

Definition at line 104 of file Products.cs.

6.23.3.2 ClearProducts()

```
bool Data_BestSale.Products.ClearProducts ()
```

Method used to Clear a list of Products.

Definition at line 196 of file Products.cs.

6.23.3.3 Exist()

```
bool Data_BestSale.Products.Exist (
            object obj)
```

Method used to verify if a product is on a products' list, given its Refference.

Parameters

reff	

Returns

Implements Data_BestSale.IListManagement.

Definition at line 124 of file Products.cs.

6.23.3.4 Remove()

```
bool Data_BestSale.Products.Remove (
            object obj)
```

Method used to remove a product from a Products' list.

Parameters

p	

Returns

Product removed successfully

Product was not removed.

Implements Data_BestSale.IListManagement.

Definition at line 142 of file Products.cs.

6.23.3.5 SearchProduct()

```
Product Data_BestSale.Products.SearchProduct (
            string reff)
```

This method searches for a product in an array, given its refference.

Parameters

reff	

Returns

Returns the product if found

Definition at line 159 of file Products.cs.

6.23.3.6 ToString()

```
override string Data_BestSale.Products.ToString ()
```

Override of the ToString() Method to convert the data of a list fo products to a string.

Returns

Definition at line 76 of file Products.cs.

6.23.3.7 TotalPrice()

```
decimal Data_BestSale.Products.TotalPrice ()
```

Method used to calculate the total price of products in a list of products.

Returns

lambda funtion tells the Sum() function that, for each Product p in _prods, it should use the price value.

Definition at line 172 of file Products.cs.

6.23.3.8 ValueInPosition()

```
decimal Data_BestSale.Products.ValueInPosition (
            int p)
```

This method returns the price of a product, given a certain array position.

Parameters

p	Position in array.

Returns

Definition at line 94 of file Products.cs.

6.23.3.9 WarratyExpirationDateForProduct()

```
DateTime Data_BestSale.Products.WarratyExpirationDateForProduct (
            DateTime date,
            string reff)
```

Method used to calculate the warranty's expiration date of a product on a list of products.

Parameters

date	
reff	

Returns

Definition at line 185 of file Products.cs.

6.23.4 Property Documentation

6.23.4.1 Prods

```
List<Product> Data_BestSale.Products.Prods  [get], [set]
```

Property used to get and set the list of products.

Definition at line 61 of file Products.cs.

The documentation for this class was generated from the following file:

- Data_BestSale/Product/Products.cs

6.24 trabalhoPOO_27967.Products Class Reference

Purpose: Class to manage a group of more than one product. Created by: Jose Alves a27967 Created on↩ : 11/9/2024 6:34:19 PM.

Inheritance diagram for trabalhoPOO_27967.Products:

Collaboration diagram for trabalhoPOO_27967.Products:

Public Member Functions

- Products ()

  The default Constructor.
- Products (List< Product > products)

  The constructor to use when list of Product is given.
- override string ToString ()

  Override of the ToString() Method to convert the data of a list fo products to a string.
- decimal ValueInPosition (int p)

  This method returns the price of a product, given a certain array position.
- bool Add (object obj)

  This method inserts a product in a list of products.
- bool Exist (object obj)

  Method used to verify if a product is on a products' list, given its Refference.
- bool Remove (object obj)

  Method used to remove a product from a Products' list.
- Product SearchProduct (string reff)

  This method searches for a product in an array, given its refference.
- decimal TotalPrice ()

  Method used to calculate the total price of products in a list of products.
- DateTime WarratyExpirationDateForProduct (DateTime date, string reff)

  Method used to calculate the warranty's expiration date of a product on a list of products.

Public Member Functions inherited from trabalhoPOO_27967.Interface.IListManagement

Properties

- List< Product > Prods `[get, set]`

  Property used to get and set the list of products.

6.24.1 Detailed Description

Purpose: Class to manage a group of more than one product. Created by: Jose Alves a27967 Created on↩ : 11/9/2024 6:34:19 PM.

Definition at line 27 of file Products.cs.

6.24.2 Constructor & Destructor Documentation

6.24.2.1 Products() [1/2]

```
trabalhoPOO_27967.Products.Products ()
```

The default Constructor.

Definition at line 40 of file Products.cs.

6.24.2.2 Products() [2/2]

```
trabalhoPOO_27967.Products.Products (
            List< Product > products)
```

The constructor to use when list of Product is given.

Parameters

products	

Definition at line 49 of file Products.cs.

6.24.3 Member Function Documentation

6.24.3.1 Add()

```
bool trabalhoPOO_27967.Products.Add (
            object obj)
```

This method inserts a product in a list of products.

Parameters

p	

Returns

Returns true or False, depending on whether or not it succeeded in inserting the product into the list.

Implements trabalhoPOO_27967.Interface.IListManagement.

Definition at line 105 of file Products.cs.

6.24.3.2 Exist()

```
bool trabalhoPOO_27967.Products.Exist (
            object obj)
```

Method used to verify if a product is on a products' list, given its Refference.

Parameters

reff	

Returns

Implements trabalhoPOO_27967.Interface.IListManagement.

Definition at line 125 of file Products.cs.

6.24.3.3 Remove()

```
bool trabalhoPOO_27967.Products.Remove (
            object obj)
```

Method used to remove a product from a Products' list.

Parameters

p	

Returns

Product removed successfully

Product was not removed.

Implements trabalhoPOO_27967.Interface.IListManagement.

Definition at line 143 of file Products.cs.

6.24.3.4 SearchProduct()

```
Product trabalhoPOO_27967.Products.SearchProduct (
            string reff)
```

This method searches for a product in an array, given its refference.

Parameters

reff	

Returns

Returns the product if found

Definition at line 160 of file Products.cs.

6.24.3.5 ToString()

```
override string trabalhoPOO_27967.Products.ToString ()
```

Override of the ToString() Method to convert the data of a list fo products to a string.

Returns

Definition at line 77 of file Products.cs.

6.24.3.6 TotalPrice()

```
decimal trabalhoPOO_27967.Products.TotalPrice ()
```

Method used to calculate the total price of products in a list of products.

Returns

Definition at line 173 of file Products.cs.

6.24.3.7 ValueInPosition()

```
decimal trabalhoPOO_27967.Products.ValueInPosition (
            int p)
```

This method returns the price of a product, given a certain array position.

Parameters

p	Position in array.

Returns

Definition at line 95 of file Products.cs.

6.24.3.8 WarratyExpirationDateForProduct()

```
DateTime trabalhoPOO_27967.Products.WarratyExpirationDateForProduct (
            DateTime date,
            string reff)
```

Method used to calculate the warranty's expiration date of a product on a list of products.

Parameters

date	
reff	

Returns

Definition at line 185 of file Products.cs.

6.24.4 Property Documentation

6.24.4.1 Prods

```
List<Product> trabalhoPOO_27967.Products.Prods  [get], [set]
```

Property used to get and set the list of products.

Definition at line 62 of file Products.cs.

The documentation for this class was generated from the following file:

- trabalhoPOO_27967/Product/Products.cs

6.25 Data_BestSale.Sale Class Reference

Purpose: Definition of Sale and methods to deal with Sale operations. Created by: Jose Alves a27967 Created on: 11/6/2024 11:21:53 AM.

Public Member Functions

- Sale ()

  The default Constructor.
- Sale (Client client, Products products, Campaign camp)

  Constructor used when a client, a product array and a campaign to be used are given.
- override bool Equals (object obj)

  Redefinition of the Equals method to compare two sales.
- override string ToString ()
- decimal TotalPrice ()

  Method to calculate the total price of a sale, given the products list and a campaign code.
- bool InsertProductOnSale (Product p)

  Method used to insert a product on a sale's list.
- bool RemoveProductFromSale (Product p)

  Method used to remove a product from a sale.
- bool ExistProductOnSale (Product p)

  Method used to verify if a product is on a sale.
- DateTime WarrantyExpirationDate (string reff)

  Method to calculate when a warranty is due to expire.

Static Public Member Functions

- static bool operator== (Sale s1, Sale s2)

  Redefiniiton of the equal operator.
- static bool operator!= (Sale s1, Sale s2)

  Redefinition of the different operator.

Properties

- int Id `[get, set]`

  Property used to get and set the ID of a Sale.
- Client Client `[get, set]`

  Property used to get and set the information of the Client who made the purchase.
- Products Products `[get, set]`

  Property used to get and set the list of products in a Sale.
- decimal TotPrice `[get, set]`

  Property used to get and set the total price of a Sale.
- DateTime SaleDate `[get, set]`

  Property used to get and set the Date of a Sale.
- Campaign Campaigns `[get, set]`

  Property used to get and set the Campaigns applicable to a Sale.

6.25.1 Detailed Description

Purpose: Definition of Sale and methods to deal with Sale operations. Created by: Jose Alves a27967 Created on: 11/6/2024 11:21:53 AM.

Definition at line 26 of file Sale.cs.

6.25.2 Constructor & Destructor Documentation

6.25.2.1 Sale() [1/2]

```
Data_BestSale.Sale.Sale ()
```

The default Constructor.

Definition at line 46 of file Sale.cs.

6.25.2.2 Sale() [2/2]

```
Data_BestSale.Sale.Sale (
            Client client,
            Products products,
            Campaign camp)
```

Constructor used when a client, a product array and a campaign to be used are given.

Parameters

client	
products	
camp	

Definition at line 59 of file Sale.cs.

6.25.3 Member Function Documentation

6.25.3.1 Equals()

```
override bool Data_BestSale.Sale.Equals (
            object obj)
```

Redefinition of the Equals method to compare two sales.

Parameters

obj	

Returns

Definition at line 136 of file Sale.cs.

6.25.3.2 ExistProductOnSale()

```
bool Data_BestSale.Sale.ExistProductOnSale (
            Product p)
```

Method used to verify if a product is on a sale.

Parameters

p	

Returns

Definition at line 219 of file Sale.cs.

6.25.3.3 InsertProductOnSale()

```
bool Data_BestSale.Sale.InsertProductOnSale (
            Product p)
```

Method used to insert a product on a sale's list.

Parameters

p	

Returns

Definition at line 199 of file Sale.cs.

6.25.3.4 operator"!=()

```
static bool Data_BestSale.Sale.operator!= (
            Sale s1,
            Sale s2)  [static]
```

Redefinition of the different operator.

Parameters

s1	
s2	

Returns

Definition at line 159 of file Sale.cs.

6.25.3.5 operator==()

```
static bool Data_BestSale.Sale.operator== (
            Sale s1,
            Sale s2)  [static]
```

Redefiniiton of the equal operator.

Parameters

s1	
s2	

Returns

Definition at line 148 of file Sale.cs.

6.25.3.6 RemoveProductFromSale()

```
bool Data_BestSale.Sale.RemoveProductFromSale (
            Product p)
```

Method used to remove a product from a sale.

Parameters

p	

Returns

Definition at line 209 of file Sale.cs.

6.25.3.7 ToString()

```
override string Data_BestSale.Sale.ToString ()
```

\u20AC é o unicode para o símbolo de euro.

Definition at line 164 of file Sale.cs.

6.25.3.8 TotalPrice()

```
decimal Data_BestSale.Sale.TotalPrice ()
```

Method to calculate the total price of a sale, given the products list and a campaign code.

Returns

The total price to pay.

Definition at line 180 of file Sale.cs.

6.25.3.9 WarrantyExpirationDate()

```
DateTime Data_BestSale.Sale.WarrantyExpirationDate (
            string reff)
```

Method to calculate when a warranty is due to expire.

Parameters

s	
reff	

Returns

Definition at line 230 of file Sale.cs.

6.25.4 Property Documentation

6.25.4.1 Campaigns

```
Campaign Data_BestSale.Sale.Campaigns  [get], [set]
```

Property used to get and set the Campaigns applicable to a Sale.

Definition at line 121 of file Sale.cs.

6.25.4.2 Client

```
Client Data_BestSale.Sale.Client  [get], [set]
```

Property used to get and set the information of the Client who made the purchase.

Definition at line 85 of file Sale.cs.

6.25.4.3 Id

```
int Data_BestSale.Sale.Id  [get], [set]
```

Property used to get and set the ID of a Sale.

Definition at line 76 of file Sale.cs.

6.25.4.4 Products

```
Products Data_BestSale.Sale.Products  [get], [set]
```

Property used to get and set the list of products in a Sale.

Definition at line 94 of file Sale.cs.

6.25.4.5 SaleDate

```
DateTime Data_BestSale.Sale.SaleDate  [get], [set]
```

Property used to get and set the Date of a Sale.

Definition at line 112 of file Sale.cs.

6.25.4.6 TotPrice

```
decimal Data_BestSale.Sale.TotPrice  [get], [set]
```

Property used to get and set the total price of a Sale.

Definition at line 103 of file Sale.cs.

The documentation for this class was generated from the following file:

- Data_BestSale/Sale/Sale.cs

6.26 trabalhoPOO_27967.Sale Class Reference

Purpose: Definition of Sale and methods to deal with Sale operations. Created by: Jose Alves a27967 Created on: 11/6/2024 11:21:53 AM.

Public Member Functions

- Sale ()
  The default Constructor.
- Sale (Client client, Products products, Campaign camp)
  Constructor used when a client, a product array and a campaign to be used are given.
- override bool Equals (object obj)
  Redefinition of the Equals method to compare two sales.
- override string ToString ()
- decimal TotalPrice ()
  Method to calculate the total price of a sale, given the products list and a campaign code.
- bool InsertProductOnSale (Product p)
  Method used to insert a product on a sale's list.
- bool RemoveProductFromSale (Product p)
  Method used to remove a product from a sale.
- bool ExistProductOnSale (Product p)
  Method used to verify if a product is on a sale.
- DateTime WarrantyExpirationDate (string reff)
  Method to calculate when a warranty is due to expire.

Static Public Member Functions

- static bool operator== (Sale s1, Sale s2)

  Redefiniiton of the equal operator.
- static bool operator!= (Sale s1, Sale s2)

  Redefinition of the different operator.

Properties

- int Id `[get, set]`

  Property used to get and set the ID of a Sale.
- Client Client `[get, set]`

  Property used to get and set the information of the Client who made the purchase.
- Products Products `[get, set]`

  Property used to get and set the list of products in a Sale.
- decimal TotPrice `[get, set]`

  Property used to get and set the total price of a Sale.
- DateTime SaleDate `[get, set]`

  Property used to get and set the Date of a Sale.
- Campaign Campaigns `[get, set]`

  Property used to get and set the Campaigns applicable to a Sale.

6.26.1 Detailed Description

Purpose: Definition of Sale and methods to deal with Sale operations. Created by: Jose Alves a27967 Created on: 11/6/2024 11:21:53 AM.

Definition at line 25 of file Sale.cs.

6.26.2 Constructor & Destructor Documentation

6.26.2.1 Sale() [1/2]

```
trabalhoPOO_27967.Sale.Sale ()
```

The default Constructor.

Definition at line 45 of file Sale.cs.

6.26.2.2 Sale() [2/2]

```
trabalhoPOO_27967.Sale.Sale (
            Client client,
            Products products,
            Campaign camp)
```

Constructor used when a client, a product array and a campaign to be used are given.

Parameters

client	
products	
camp	

Definition at line 58 of file Sale.cs.

6.26.3 Member Function Documentation

6.26.3.1 Equals()

```
override bool trabalhoPOO_27967.Sale.Equals (
            object obj)
```

Redefinition of the Equals method to compare two sales.

Parameters

obj	

Returns

Definition at line 135 of file Sale.cs.

6.26.3.2 ExistProductOnSale()

```
bool trabalhoPOO_27967.Sale.ExistProductOnSale (
            Product p)
```

Method used to verify if a product is on a sale.

Parameters

p	

Returns

Definition at line 218 of file Sale.cs.

6.26.3.3 InsertProductOnSale()

```
bool trabalhoPOO_27967.Sale.InsertProductOnSale (
            Product p)
```

Method used to insert a product on a sale's list.

Parameters

p	

Returns

Definition at line 198 of file Sale.cs.

6.26.3.4 operator"!=()

```
static bool trabalhoPOO_27967.Sale.operator!= (
            Sale s1,
            Sale s2)  [static]
```

Redefinition of the different operator.

Parameters

s1	
s2	

Returns

Definition at line 158 of file Sale.cs.

6.26.3.5 operator==()

```
static bool trabalhoPOO_27967.Sale.operator== (
            Sale s1,
            Sale s2)  [static]
```

Redefiniiton of the equal operator.

Parameters

s1	
s2	

Returns

Definition at line 147 of file Sale.cs.

6.26.3.6 RemoveProductFromSale()

```
bool trabalhoPOO_27967.Sale.RemoveProductFromSale (
            Product p)
```

Method used to remove a product from a sale.

Parameters

p	

Returns

Definition at line 208 of file Sale.cs.

6.26.3.7 ToString()

```
override string trabalhoPOO_27967.Sale.ToString ()
```

\u20AC é o unicode para o símbolo de euro.

Definition at line 163 of file Sale.cs.

6.26.3.8 TotalPrice()

```
decimal trabalhoPOO_27967.Sale.TotalPrice ()
```

Method to calculate the total price of a sale, given the products list and a campaign code.

Returns

The total price to pay.

Definition at line 179 of file Sale.cs.

6.26.3.9 WarrantyExpirationDate()

```
DateTime trabalhoPOO_27967.Sale.WarrantyExpirationDate (
            string reff)
```

Method to calculate when a warranty is due to expire.

Parameters

s	
reff	

Returns

Definition at line 229 of file Sale.cs.

6.26.4 Property Documentation

6.26.4.1 Campaigns

```
Campaign trabalhoPOO_27967.Sale.Campaigns  [get], [set]
```

Property used to get and set the Campaigns applicable to a Sale.

Definition at line 120 of file Sale.cs.

6.26.4.2 Client

```
Client trabalhoPOO_27967.Sale.Client  [get], [set]
```

Property used to get and set the information of the Client who made the purchase.

Definition at line 84 of file Sale.cs.

6.26.4.3 Id

```
int trabalhoPOO_27967.Sale.Id  [get], [set]
```

Property used to get and set the ID of a Sale.

Definition at line 75 of file Sale.cs.

6.26.4.4 Products

```
Products trabalhoPOO_27967.Sale.Products  [get], [set]
```

Property used to get and set the list of products in a Sale.

Definition at line 93 of file Sale.cs.

6.26.4.5 SaleDate

```
DateTime trabalhoPOO_27967.Sale.SaleDate  [get], [set]
```

Property used to get and set the Date of a Sale.

Definition at line 111 of file Sale.cs.

6.26.4.6 TotPrice

```
decimal trabalhoPOO_27967.Sale.TotPrice  [get], [set]
```

Property used to get and set the total price of a Sale.

Definition at line 102 of file Sale.cs.

The documentation for this class was generated from the following file:

- trabalhoPOO_27967/Sale/Sale.cs

6.27 Data_BestSale.Sales Class Reference

Purpose: Class with the agregation of sales of a store. Created by: Jose Alves a27967 Created on: 11/10/2024 7:42:03 PM.

Inheritance diagram for Data_BestSale.Sales:

Collaboration diagram for Data_BestSale.Sales:

Public Member Functions

- Sales ()
  The default Constructor.
- Sales (List< Sale > sales)
  The constructor to use when the sales' List is given.
- Sale GetSale (int idSale)
  Method to find a sale in a list of sales, given its ID.
- bool Add (object obj)
  Method used to add a sale to a sales' list.
- bool Remove (object obj)
  Method used to remove a sale from a list of sales.
- bool Exist (object obj)
  Method used to check if a sale exists in a list of sales.
- bool ClearSales ()
  Method used to Clear a list of Sales.

Public Member Functions inherited from Data_BestSale.IListManagement

Properties

- List< Sale > SalesStored `[get, set]`
  Property used to get and set a list of sales.

6.27.1 Detailed Description

Purpose: Class with the agregation of sales of a store. Created by: Jose Alves a27967 Created on: 11/10/2024 7:42:03 PM.

Definition at line 22 of file Sales.cs.

6.27.2 Constructor & Destructor Documentation

6.27.2.1 Sales() [1/2]

```
Data_BestSale.Sales.Sales ()
```

The default Constructor.

Definition at line 35 of file Sales.cs.

6.27.2.2 Sales() [2/2]

```
Data_BestSale.Sales.Sales (
            List< Sale > sales)
```

The constructor to use when the sales' List is given.

Parameters

sales	

Definition at line 44 of file Sales.cs.

6.27.3 Member Function Documentation

6.27.3.1 Add()

```
bool Data_BestSale.Sales.Add (
            object obj)
```

Method used to add a sale to a sales' list.

Parameters

sale	

Returns

Implements Data_BestSale.IListManagement.

Definition at line 89 of file Sales.cs.

6.27.3.2 ClearSales()

```
bool Data_BestSale.Sales.ClearSales ()
```

Method used to Clear a list of Sales.

Definition at line 144 of file Sales.cs.

6.27.3.3 Exist()

```
bool Data_BestSale.Sales.Exist (
            object obj)
```

Method used to check if a sale exists in a list of sales.

Parameters

obj	

Returns

Implements Data_BestSale.IListManagement.

Definition at line 125 of file Sales.cs.

6.27.3.4 GetSale()

```
Sale Data_BestSale.Sales.GetSale (
            int idSale)
```

Method to find a sale in a list of sales, given its ID.

Parameters

idSale	

Returns

Definition at line 75 of file Sales.cs.

6.27.3.5 Remove()

```
bool Data_BestSale.Sales.Remove (
            object obj)
```

Method used to remove a sale from a list of sales.

Parameters

obj	

Returns

Implements Data_BestSale.IListManagement.

Definition at line 105 of file Sales.cs.

6.27.4 Property Documentation

6.27.4.1 SalesStored

```
List<Sale> Data_BestSale.Sales.SalesStored  [get], [set]
```

Property used to get and set a list of sales.

Definition at line 56 of file Sales.cs.

The documentation for this class was generated from the following file:

- Data_BestSale/Sale/Sales.cs

6.28 trabalhoPOO_27967.Sales Class Reference

Purpose: Class with the agregation of sales of a store. Created by: Jose Alves a27967 Created on: 11/10/2024 7:42:03 PM.

Inheritance diagram for trabalhoPOO_27967.Sales:

Collaboration diagram for trabalhoPOO_27967.Sales:

Public Member Functions

- Sales ()

  The default Constructor.
- Sales (List< Sale > sales)

  The constructor to use when the sales' List is given.
- Sale GetSale (int idSale)

  Method to find a sale in a list of sales, given its ID.
- bool Add (object obj)

  Method used to add a sale to a sales' list.
- bool Remove (object obj)

  Method used to remove a sale from a list of sales.
- bool Exist (object obj)

  Method used to check if a sale exists in a list of sales.

Public Member Functions inherited from trabalhoPOO_27967.Interface.IListManagement

Properties

- List< Sale > SalesStored `[get, set]`

  Property used to get and set a list of sales.

6.28.1 Detailed Description

Purpose: Class with the agregation of sales of a store. Created by: Jose Alves a27967 Created on: 11/10/2024 7:42:03 PM.

Definition at line 22 of file Sales.cs.

6.28.2 Constructor & Destructor Documentation

6.28.2.1 Sales() [1/2]

```
trabalhoPOO_27967.Sales.Sales ()
```

The default Constructor.

Definition at line 35 of file Sales.cs.

6.28.2.2 Sales() [2/2]

```
trabalhoPOO_27967.Sales.Sales (
            List< Sale > sales)
```

The constructor to use when the sales' List is given.

Parameters

sales	

Definition at line 44 of file Sales.cs.

6.28.3 Member Function Documentation

6.28.3.1 Add()

```
bool trabalhoPOO_27967.Sales.Add (
            object obj)
```

Method used to add a sale to a sales' list.

Parameters

sale	

Returns

Implements trabalhoPOO_27967.Interface.IListManagement.

Definition at line 89 of file Sales.cs.

6.28.3.2 Exist()

```
bool trabalhoPOO_27967.Sales.Exist (
            object obj)
```

Method used to check if a sale exists in a list of sales.

Parameters

obj	

Returns

Implements trabalhoPOO_27967.Interface.IListManagement.

Definition at line 125 of file Sales.cs.

6.28.3.3 GetSale()

```
Sale trabalhoPOO_27967.Sales.GetSale (
            int idSale)
```

Method to find a sale in a list of sales, given its ID.

Parameters

idSale	

Returns

Definition at line 75 of file Sales.cs.

6.28.3.4 Remove()

```
bool trabalhoPOO_27967.Sales.Remove (
            object obj)
```

Method used to remove a sale from a list of sales.

Parameters

obj	

Returns

Implements trabalhoPOO_27967.Interface.IListManagement.

Definition at line 105 of file Sales.cs.

6.28.4 Property Documentation

6.28.4.1 SalesStored

```
List<Sale> trabalhoPOO_27967.Sales.SalesStored  [get], [set]
```

Property used to get and set a list of sales.

Definition at line 56 of file Sales.cs.

The documentation for this class was generated from the following file:

- trabalhoPOO_27967/Sale/Sales.cs

6.29 Business_Object.SimpleProduct Class Reference

Purpose:This File contains the definition and methods to manage a SimpleClient Created by: zecun Created on: 12/11/2024 11:18:40 AM.

Public Member Functions

- SimpleProduct ()

  The default Constructor.
- SimpleProduct (string reff, decimal price, int make)

  The constructor to use when a reference, price and makeID are given.

Properties

- string Reference `[get, set]`

  Property to set and get the reference of a SimpleProduct object.
- decimal Price `[get, set]`

  Property to get and set the price of a SimpleProduct object.
- int Make `[get, set]`

  Property to set and get the Make of a SimpleProduct object.

6.29.1 Detailed Description

Purpose:This File contains the definition and methods to manage a SimpleClient Created by: zecun Created on: 12/11/2024 11:18:40 AM.

Definition at line 20 of file SimpleProduct.cs.

6.29.2 Constructor & Destructor Documentation

6.29.2.1 SimpleProduct() [1/2]

```
Business_Object.SimpleProduct.SimpleProduct ()
```

The default Constructor.

Definition at line 35 of file SimpleProduct.cs.

6.29.2.2 SimpleProduct() [2/2]

```
Business_Object.SimpleProduct.SimpleProduct (
            string reff,
            decimal price,
            int make)
```

The constructor to use when a reference, price and makeID are given.

Parameters

reff	Product Reference
price	Product Price
make	Make ID

Definition at line 45 of file SimpleProduct.cs.

6.29.3 Property Documentation

6.29.3.1 Make

```
int Business_Object.SimpleProduct.Make  [get], [set]
```

Property to set and get the Make of a SimpleProduct object.

Definition at line 76 of file SimpleProduct.cs.

6.29.3.2 Price

```
decimal Business_Object.SimpleProduct.Price  [get], [set]
```

Property to get and set the price of a SimpleProduct object.

Definition at line 67 of file SimpleProduct.cs.

6.29.3.3 Reference

```
string Business_Object.SimpleProduct.Reference  [get], [set]
```

Property to set and get the reference of a SimpleProduct object.

Definition at line 57 of file SimpleProduct.cs.

The documentation for this class was generated from the following file:

- Business_Object/SimpleProduct.cs

6.30 Data_BestSale.Store Class Reference

Purpose: This class has the definition and properties to manage a store. Created by: Jose Alves a27967 Created on: 11/14/2024 5:01:23 PM.

Public Member Functions

- Store ()
  The default Constructor.
- Store (Clients cl, Products p, Sales s, Makes m, Categories c)
  The constructor to use when all the lists are given.
- bool SaveStoreBin (string fileName)
  Save a Store to a binary file.

Static Public Member Functions

- static string GetMakeNameFromID (int makeID)

  Method that gets a make's name from the list of makes in a store.
- static bool InsertClientInStore (Client client)

  Inserts a client in the store's list of clients, if it's not already there.
- static bool ClearStore ()

  Method used to clear the data of a store from memory.
- static bool LoadStoreBin (string fileName)

  Load a Stor from a binary file.
- static bool InsertProductInStore (Product prod)

  Method used to add a product to the list of products of a store.
- static decimal GetProductPriceInStoreFromReference (string reference)

  Method that returns a product price from the list of products in a store, given its reference.
- static int GetMakeIdFromNameInStore (string name)

  Method to get the ID of a make in a store's list, given its name.
- static bool InsertMakeInStore (Make make)

  Method to insert a make on a store's list of makes.
- static int GetCategoryIdFromNameInStore (string name)

  This method finds the ID of a Category, given its name.
- static bool InsertCategoryInStore (Category cat)

  This method adds a Category to a store's list of categories.

Properties

- Clients ClientLIst `[get, set]`

  The property used to get and set the clients' list.
- Products ProdList `[get, set]`

  The property to get and set de products' list.
- Sales SaleList `[get, set]`

  The property to get and set de sales' list.
- Makes MakeList `[get, set]`

  The property to get and set de makes' list.
- Categories CatList `[get, set]`

  The property to get and set de categories' list.

6.30.1 Detailed Description

Purpose: This class has the definition and properties to manage a store. Created by: Jose Alves a27967 Created on: 11/14/2024 5:01:23 PM.

Definition at line 26 of file Store.cs.

6.30.2 Constructor & Destructor Documentation

6.30.2.1 Store() [1/2]

```
Data_BestSale.Store.Store ()
```

The default Constructor.

Definition at line 43 of file Store.cs.

6.30.2.2 Store() [2/2]

```
Data_BestSale.Store.Store (
            Clients cl,
            Products p,
            Sales s,
            Makes m,
            Categories c)
```

The constructor to use when all the lists are given.

Parameters

cl	
p	
s	
m	
c	

Definition at line 60 of file Store.cs.

6.30.3 Member Function Documentation

6.30.3.1 ClearStore()

```
static bool Data_BestSale.Store.ClearStore ()  [static]
```

Method used to clear the data of a store from memory.

Definition at line 182 of file Store.cs.

6.30.3.2 GetCategoryIdFromNameInStore()

```
static int Data_BestSale.Store.GetCategoryIdFromNameInStore (
            string name)  [static]
```

This method finds the ID of a Category, given its name.

Parameters

name	The name of the Make

Returns

The ID of the Category

-100 - There's no category with that name on the list.

Returns

Definition at line 298 of file Store.cs.

6.30.3.3 GetMakeIdFromNameInStore()

```
static int Data_BestSale.Store.GetMakeIdFromNameInStore (
            string name)  [static]
```

Method to get the ID of a make in a store's list, given its name.

Parameters

name	

Returns

The ID of the make

-50 if the make does not exist on the list.

Returns

Definition at line 269 of file Store.cs.

6.30.3.4 GetMakeNameFromID()

```
static string Data_BestSale.Store.GetMakeNameFromID (
            int makeID)  [static]
```

Method that gets a make's name from the list of makes in a store.

Parameters

makeID	

Returns

The name of the make, if found. Otherwise, returns 'Not Found'

Definition at line 131 of file Store.cs.

6.30.3.5 GetProductPriceInStoreFromReference()

```
static decimal Data_BestSale.Store.GetProductPriceInStoreFromReference (
            string reference)  [static]
```

Method that returns a product price from the list of products in a store, given its reference.

Parameters

reference	The reference wanted.

Returns

The product that matches that reference.

Definition at line 254 of file Store.cs.

6.30.3.6 InsertCategoryInStore()

```
static bool Data_BestSale.Store.InsertCategoryInStore (
            Category cat)  [static]
```

This method adds a Category to a store's list of categories.

Parameters

cat	

Returns

True or false, wheter it succeeded or not.

Definition at line 313 of file Store.cs.

6.30.3.7 InsertClientInStore()

```
static bool Data_BestSale.Store.InsertClientInStore (
            Client client)  [static]
```

Inserts a client in the store's list of clients, if it's not already there.

Parameters

client	

Returns

True - Client has been successfully added to the list.

False - Client already exists or an error occurred.

Returns

Definition at line 147 of file Store.cs.

6.30.3.8 InsertMakeInStore()

```
static bool Data_BestSale.Store.InsertMakeInStore (
            Make make)  [static]
```

Method to insert a make on a store's list of makes.

Parameters

make	The make to insert on the list

Returns

True or false, wheter it was added or not.

Definition at line 284 of file Store.cs.

6.30.3.9 InsertProductInStore()

```
static bool Data_BestSale.Store.InsertProductInStore (
            Product prod)  [static]
```

Method used to add a product to the list of products of a store.

Parameters

prod	The product to add.

Returns

True - Product added to the list.

False - The product already exists on the list.

Returns

Definition at line 239 of file Store.cs.

6.30.3.10 LoadStoreBin()

```
static bool Data_BestSale.Store.LoadStoreBin (
            string fileName)  [static]
```

Load a Stor from a binary file.

Parameters

fileName	Name of file where the data is stored.

Returns

Verify if a file with that name exists and has content in it.

Definition at line 204 of file Store.cs.

6.30.3.11 SaveStoreBin()

```
bool Data_BestSale.Store.SaveStoreBin (
            string fileName)
```

Save a Store to a binary file.

Parameters

fileName	

Returns

Definition at line 158 of file Store.cs.

6.30.4 Property Documentation

6.30.4.1 CatList

```
Categories Data_BestSale.Store.CatList  [get], [set]
```

The property to get and set de categories' list.

Definition at line 111 of file Store.cs.

6.30.4.2 ClientLIst

```
Clients Data_BestSale.Store.ClientLIst  [get], [set]
```

The property used to get and set the clients' list.

Definition at line 75 of file Store.cs.

6.30.4.3 MakeList

```
Makes Data_BestSale.Store.MakeList  [get], [set]
```

The property to get and set de makes' list.

Definition at line 102 of file Store.cs.

6.30.4.4 ProdList

```
Products Data_BestSale.Store.ProdList  [get], [set]
```

The property to get and set de products' list.

Definition at line 84 of file Store.cs.

6.30.4.5 SaleList

```
Sales Data_BestSale.Store.SaleList  [get], [set]
```

The property to get and set de sales' list.

Definition at line 93 of file Store.cs.

The documentation for this class was generated from the following file:

- Data_BestSale/Store/Store.cs

6.31 trabalhoPOO_27967.Store.Store Class Reference

Purpose: This class has the definition and properties to manage a store. Created by: Jose Alves a27967 Created on: 11/14/2024 5:01:23 PM.

Public Member Functions

- Store ()
  The default Constructor.
- Store (Clients cl, Products p, Sales s, Makes m, Categories c, Warranties w)
  The constructor to use when all the lists are given.

Static Public Member Functions

- static string GetMakeNameFromID (int makeID)
  Method that gets a make's name from the list of makes in a store.

Properties

- Clients ClientLIst `[get, set]`
  The property used to get and set the clients' list.
- Products ProdList `[get, set]`
  The property to get and set de products' list.
- Sales SaleList `[get, set]`
  The property to get and set de sales' list.
- Makes MakeList `[get, set]`
  The property to get and set de makes' list.
- Categories CatList `[get, set]`
  The property to get and set de categories' list.
- Warranties WarrantList `[get, set]`
  The property to get and set de warranties' list.

6.31.1 Detailed Description

Purpose: This class has the definition and properties to manage a store. Created by: Jose Alves a27967 Created on: 11/14/2024 5:01:23 PM.

Definition at line 20 of file Store.cs.

6.31.2 Constructor & Destructor Documentation

6.31.2.1 Store() `[1/2]`

```
trabalhoPOO_27967.Store.Store.Store ()
```

The default Constructor.

Definition at line 38 of file Store.cs.

6.31.2.2 Store() [2/2]

```
trabalhoPOO_27967.Store.Store.Store (
            Clients cl,
            Products p,
            Sales s,
            Makes m,
            Categories c,
            Warranties w)
```

The constructor to use when all the lists are given.

Parameters

cl	
p	
s	
m	
c	
w	

Definition at line 57 of file Store.cs.

6.31.3 Member Function Documentation

6.31.3.1 GetMakeNameFromID()

```
static string trabalhoPOO_27967.Store.Store.GetMakeNameFromID (
            int makeID)  [static]
```

Method that gets a make's name from the list of makes in a store.

Parameters

makeID	

Returns

The name of the make, if found. Otherwise, returns 'Not Found'

Definition at line 138 of file Store.cs.

6.31.4 Property Documentation

6.31.4.1 CatList

```
Categories trabalhoPOO_27967.Store.Store.CatList  [get], [set]
```

The property to get and set de categories' list.

Definition at line 109 of file Store.cs.

6.31.4.2 ClientLIst

```
Clients trabalhoPOO_27967.Store.Store.ClientLIst  [get], [set]
```

The property used to get and set the clients' list.

Definition at line 73 of file Store.cs.

6.31.4.3 MakeList

```
Makes trabalhoPOO_27967.Store.Store.MakeList  [get], [set]
```

The property to get and set de makes' list.

Definition at line 100 of file Store.cs.

6.31.4.4 ProdList

```
Products trabalhoPOO_27967.Store.Store.ProdList  [get], [set]
```

The property to get and set de products' list.

Definition at line 82 of file Store.cs.

6.31.4.5 SaleList

```
Sales trabalhoPOO_27967.Store.Store.SaleList  [get], [set]
```

The property to get and set de sales' list.

Definition at line 91 of file Store.cs.

6.31.4.6 WarrantList

```
Warranties trabalhoPOO_27967.Store.Store.WarrantList  [get], [set]
```

The property to get and set de warranties' list.

Definition at line 118 of file Store.cs.

The documentation for this class was generated from the following file:

- trabalhoPOO_27967/Store/Store.cs

6.32 Data_BestSale.Warranties Class Reference

Purpose:This file has the definition and methods to work with the plurality of Warranty. Created by: Jose Alves a27967 Created on: 11/14/2024 4:20:11 PM.

Inheritance diagram for Data_BestSale.Warranties:

Collaboration diagram for Data_BestSale.Warranties:

Public Member Functions

- Warranties ()

  The default Constructor.
- Warranties (List< Warranty > warrants)

  The constructor to use when a list of Warranties is given.
- bool Add (object obj)
- bool Remove (object obj)

  Method used to remove a warranty from a list of warranties.
- bool Exist (object obj)

  Method used to confirm if a warranty exists on a list of warranties, given the product ID.
- void ClearWarranties ()

  Method used to Clear a list of Warranties.

Public Member Functions inherited from Data_BestSale.IListManagement

Properties

- List< Warranty > Warrants `[get, set]`

  Property used to get and set the list of warranties.

6.32.1 Detailed Description

Purpose:This file has the definition and methods to work with the plurality of Warranty. Created by: Jose Alves a27967 Created on: 11/14/2024 4:20:11 PM.

Definition at line 23 of file Warranties.cs.

6.32.2 Constructor & Destructor Documentation

6.32.2.1 Warranties() [1/2]

```
Data_BestSale.Warranties.Warranties ()
```

The default Constructor.

Definition at line 36 of file Warranties.cs.

6.32.2.2 Warranties() [2/2]

```
Data_BestSale.Warranties.Warranties (
            List< Warranty > warrants)
```

The constructor to use when a list of Warranties is given.

Parameters

warrants	

Definition at line 45 of file Warranties.cs.

6.32.3 Member Function Documentation

6.32.3.1 Add()

```
bool Data_BestSale.Warranties.Add (
            object obj)
```

Method used to add a warranty to a list of warranties.

Parameters

c	

Returns

Implements Data_BestSale.IListManagement.

Definition at line 76 of file Warranties.cs.

6.32.3.2 ClearWarranties()

```
void Data_BestSale.Warranties.ClearWarranties ()
```

Method used to Clear a list of Warranties.

Definition at line 138 of file Warranties.cs.

6.32.3.3 Exist()

```
bool Data_BestSale.Warranties.Exist (
            object obj)
```

Method used to confirm if a warranty exists on a list of warranties, given the product ID.

Parameters

id	

Returns

True - If Warranty exists in the list of Warranties

False - If Warranty does not exist in the list of Warranties

Implements Data_BestSale.IListManagement.

Definition at line 119 of file Warranties.cs.

6.32.3.4 Remove()

```
bool Data_BestSale.Warranties.Remove (
            object obj)
```

Method used to remove a warranty from a list of warranties.

Parameters

camp	

Returns

Implements Data_BestSale.IListManagement.

Definition at line 92 of file Warranties.cs.

6.32.4 Property Documentation

6.32.4.1 Warrants

```
List<Warranty> Data_BestSale.Warranties.Warrants  [get], [set]
```

Property used to get and set the list of warranties.

Definition at line 58 of file Warranties.cs.

The documentation for this class was generated from the following file:

- Data_BestSale/Warranty/Warranties.cs

6.33 trabalhoPOO_27967.Warranties Class Reference

Purpose:This file has the definition and methods to work with the plurality of Warranty. Created by: Jose Alves a27967 Created on: 11/14/2024 4:20:11 PM.

Inheritance diagram for trabalhoPOO_27967.Warranties:

Collaboration diagram for trabalhoPOO_27967.Warranties:

Public Member Functions

- Warranties ()

  The default Constructor.
- Warranties (List< Warranty > warrants)

  The constructor to use when a list of Warranties is given.
- bool Add (object obj)
- bool Remove (object obj)

  Method used to remove a warranty from a list of warranties.
- bool Exist (object obj)

  Method used to confirm if a warranty exists on a list of warranties, given the product ID.

Public Member Functions inherited from trabalhoPOO_27967.Interface.IListManagement

Properties

- List< Warranty > Warrants `[get, set]`

  Property used to get and set the list of warranties.

6.33.1 Detailed Description

Purpose:This file has the definition and methods to work with the plurality of Warranty. Created by: Jose Alves a27967 Created on: 11/14/2024 4:20:11 PM.

Definition at line 22 of file Warranties.cs.

6.33.2 Constructor & Destructor Documentation

6.33.2.1 Warranties() [1/2]

```
trabalhoPOO_27967.Warranties.Warranties ()
```

The default Constructor.

Definition at line 35 of file Warranties.cs.

6.33.2.2 Warranties() [2/2]

```
trabalhoPOO_27967.Warranties.Warranties (
            List< Warranty > warrants)
```

The constructor to use when a list of Warranties is given.

Parameters

warrants	

Definition at line 44 of file Warranties.cs.

6.33.3 Member Function Documentation

6.33.3.1 Add()

```
bool trabalhoPOO_27967.Warranties.Add (
            object obj)
```

Method used to add a warranty to a list of warranties.

Parameters

c	

Returns

Implements trabalhoPOO_27967.Interface.IListManagement.

Definition at line 75 of file Warranties.cs.

6.33.3.2 Exist()

```
bool trabalhoPOO_27967.Warranties.Exist (
            object obj)
```

Method used to confirm if a warranty exists on a list of warranties, given the product ID.

Parameters

id	

Returns

Implements trabalhoPOO_27967.Interface.IListManagement.

Definition at line 116 of file Warranties.cs.

6.33.3.3 Remove()

```
bool trabalhoPOO_27967.Warranties.Remove (
            object obj)
```

Method used to remove a warranty from a list of warranties.

Parameters

camp	

Returns

Implements trabalhoPOO_27967.Interface.IListManagement.

Definition at line 91 of file Warranties.cs.

6.33.4 Property Documentation

6.33.4.1 Warrants

```
List<Warranty> trabalhoPOO_27967.Warranties.Warrants  [get], [set]
```

Property used to get and set the list of warranties.

Definition at line 57 of file Warranties.cs.

The documentation for this class was generated from the following file:

- trabalhoPOO_27967/Warranty/Warranties.cs

6.34 Data_BestSale.Warranty Class Reference

Purpose: This class contains the definition and methods to manage warranties. Created by: Jose Alves a27967 Created on: 11/13/2024 4:17:18 PM.

Public Member Functions

- Warranty ()
  The default Constructor.
- Warranty (string prodID, int durationInYears, string conditions)
  The constructor to use when the ID of the product, duration (in years) and the conditions are given.
- override string ToString ()
  Method to show the information of a Warranty.
- override bool Equals (object obj)
  Redefine the Equals operator to verify if a warranty matches the other.
- DateTime ExpirationDate (Sale s, string reff)
  Method to calculate when a warranty is due to expire.

Static Public Member Functions

- static bool operator== (Warranty w1, Warranty w2)
  Redefinition of the == operator.
- static bool operator!= (Warranty w1, Warranty w2)
  Redefinition of the != operator.
- static Warranty CreateWarranty (string reff, int warrantyDuration, string warrantyConditions)
  Method that creates a warranty instance, given the reference of the product, warranty duration and its terms.

Properties

- string ProdID [get, set]
  Property to get and set the ID of the product to which the warranty is about.
- int DurationInYears [get, set]
  The property to set and get the duration of a warranty (in years).
- string Conditions [get, set]
  The property to get and set the terms of a warranty.

6.34.1 Detailed Description

Purpose: This class contains the definition and methods to manage warranties. Created by: Jose Alves a27967 Created on: 11/13/2024 4:17:18 PM.

Definition at line 24 of file Warranty.cs.

6.34.2 Constructor & Destructor Documentation

6.34.2.1 Warranty() [1/2]

```
Data_BestSale.Warranty.Warranty ()
```

The default Constructor.

Definition at line 39 of file Warranty.cs.

6.34.2.2 Warranty() [2/2]

```
Data_BestSale.Warranty.Warranty (
            string prodID,
            int durationInYears,
            string conditions)
```

The constructor to use when the ID of the product, duration (in years) and the conditions are given.

Parameters

prodID	
durationInYears	
conditions	

Definition at line 52 of file Warranty.cs.

6.34.3 Member Function Documentation

6.34.3.1 CreateWarranty()

```
static Warranty Data_BestSale.Warranty.CreateWarranty (
            string reff,
            int warrantyDuration,
            string warrantyConditions)  [static]
```

Method that creates a warranty instance, given the reference of the product, warranty duration and its terms.

Parameters

reff	
warrantyDuration	
warrantyConditions	

Returns

Definition at line 171 of file Warranty.cs.

6.34.3.2 Equals()

```
override bool Data_BestSale.Warranty.Equals (
            object obj)
```

Redefine the Equals operator to verify if a warranty matches the other.

Parameters

obj	

Returns

Veriffies if the object given is null.

Casts the object to be Warranty.

Definition at line 115 of file Warranty.cs.

6.34.3.3 ExpirationDate()

```
DateTime Data_BestSale.Warranty.ExpirationDate (
            Sale s,
            string reff)
```

Method to calculate when a warranty is due to expire.

Parameters

s	
reff	

Returns

Definition at line 158 of file Warranty.cs.

6.34.3.4 operator"!=()

```
static bool Data_BestSale.Warranty.operator!= (
            Warranty w1,
            Warranty w2)  [static]
```

Redefinition of the != operator.

Parameters

w1	
w2	

Returns

Definition at line 145 of file Warranty.cs.

6.34.3.5 operator==()

```
static bool Data_BestSale.Warranty.operator== (
            Warranty w1,
            Warranty w2)  [static]
```

Redefinition of the == operator.

Parameters

w1	
w2	

Returns

Definition at line 134 of file Warranty.cs.

6.34.3.6 ToString()

```
override string Data_BestSale.Warranty.ToString ()
```

Method to show the information of a Warranty.

Returns

Definition at line 100 of file Warranty.cs.

6.34.4 Property Documentation

6.34.4.1 Conditions

```
string Data_BestSale.Warranty.Conditions  [get], [set]
```

The property to get and set the terms of a warranty.

Definition at line 86 of file Warranty.cs.

6.34.4.2 DurationInYears

```
int Data_BestSale.Warranty.DurationInYears  [get], [set]
```

The property to set and get the duration of a warranty (in years).

Definition at line 77 of file Warranty.cs.

6.34.4.3 ProdID

```
string Data_BestSale.Warranty.ProdID  [get], [set]
```

Property to get and set the ID of the product to which the warranty is about.

Definition at line 68 of file Warranty.cs.

The documentation for this class was generated from the following file:

- Data_BestSale/Warranty/Warranty.cs

6.35 trabalhoPOO_27967.Warranty Class Reference

Purpose: This class contains the definition and methods to manage warranties. Created by: Jose Alves a27967 Created on: 11/13/2024 4:17:18 PM.

Public Member Functions

- Warranty ()
  The default Constructor.
- Warranty (string prodID, int durationInYears, string conditions)
  The constructor to use when the ID of the product, duration (in years) and the conditions are given.
- override string ToString ()
  Method to show the information of a Warranty.
- override bool Equals (object obj)
  Redefine the Equals operator to verify if a warranty matches the other.
- DateTime ExpirationDate (Sale s, string reff)
  Method to calculate when a warranty is due to expire.

Static Public Member Functions

- static bool operator== (Warranty w1, Warranty w2)
  Redefinition of the == operator.
- static bool operator!= (Warranty w1, Warranty w2)
  Redefinition of the != operator.

Properties

- string ProdID `[get, set]`

  Property to get and set the ID of the product to which the warranty is about.
- int DurationInYears `[get, set]`

  The property to set and get the duration of a warranty (in years).
- string Conditions `[get, set]`

  The property to get and set the terms of a warranty.

6.35.1 Detailed Description

Purpose: This class contains the definition and methods to manage warranties. Created by: Jose Alves a27967 Created on: 11/13/2024 4:17:18 PM.

Definition at line 23 of file Warranty.cs.

6.35.2 Constructor & Destructor Documentation

6.35.2.1 Warranty() `[1/2]`

```
trabalhoPOO_27967.Warranty.Warranty ()
```

The default Constructor.

Definition at line 38 of file Warranty.cs.

6.35.2.2 Warranty() `[2/2]`

```
trabalhoPOO_27967.Warranty.Warranty (
            string prodID,
            int durationInYears,
            string conditions)
```

The constructor to use when the ID of the product, duration (in years) and the conditions are given.

Parameters

prodID	
durationInYears	
conditions	

Definition at line 51 of file Warranty.cs.

6.35.3 Member Function Documentation

6.35.3.1 Equals()

```
override bool trabalhoPOO_27967.Warranty.Equals (
            object obj)
```

Redefine the Equals operator to verify if a warranty matches the other.

Parameters

obj	

Returns

Veriffies if the object given is null.

Casts the object to be Warranty.

Definition at line 114 of file Warranty.cs.

6.35.3.2 ExpirationDate()

```
DateTime trabalhoPOO_27967.Warranty.ExpirationDate (
            Sale s,
            string reff)
```

Method to calculate when a warranty is due to expire.

Parameters

s	
reff	

Returns

Definition at line 157 of file Warranty.cs.

6.35.3.3 operator"!=()

```
static bool trabalhoPOO_27967.Warranty.operator!= (
            Warranty w1,
            Warranty w2)  [static]
```

Redefinition of the != operator.

Parameters

w1	
w2	

Returns

Definition at line 144 of file Warranty.cs.

6.35.3.4 operator==()

```
static bool trabalhoPOO_27967.Warranty.operator== (
            Warranty w1,
            Warranty w2)  [static]
```

Redefinition of the == operator.

Parameters

w1	
w2	

Returns

Definition at line 133 of file Warranty.cs.

6.35.3.5 ToString()

```
override string trabalhoPOO_27967.Warranty.ToString ()
```

Method to show the information of a Warranty.

Returns

Definition at line 99 of file Warranty.cs.

6.35.4 Property Documentation

6.35.4.1 Conditions

```
string trabalhoPOO_27967.Warranty.Conditions  [get], [set]
```

The property to get and set the terms of a warranty.

Definition at line 85 of file Warranty.cs.

6.35.4.2 DurationInYears

```
int trabalhoPOO_27967.Warranty.DurationInYears  [get], [set]
```

The property to set and get the duration of a warranty (in years).

Definition at line 76 of file Warranty.cs.

6.35.4.3 ProdID

```
string trabalhoPOO_27967.Warranty.ProdID  [get], [set]
```

Property to get and set the ID of the product to which the warranty is about.

Definition at line 67 of file Warranty.cs.

The documentation for this class was generated from the following file:

- trabalhoPOO_27967/Warranty/Warranty.cs

Chapter 7

File Documentation

7.1 BestSale.cs

```
00001 /*
00002 *   <copyright file="BestSale.cs" company="IPCA">
00003 *       Copyright (c) 2024 All Rights Reserved
00004 *   </copyright>
00005 *   <author>Jose Alves a27967</author>
00006 *   <date>12/17/2024 6:23:56 PM</date>
00007 *   <description>This file contains the frond end of the app..</description>
00008 **/
00009 using System;
00010 using System.Collections.Generic;
00011 using System.Linq;
00012 using System.Text;
00013 using System.Threading.Tasks;
00014 using System.Text.RegularExpressions; //Used to verify if a string meets certain criteria.
00015 using Business_Layer;
00016 using System.IO;
00017
00018 namespace BestSale
00019 {
00020     class BestSale
00021     {
00022         static void Main(string[] args)
00023         {
00024             Console.OutputEncoding = System.Text.Encoding.UTF8;
00025
00026             bool a;
00027
00028             try
00029             {
00030                 a = ClientManagement.CreateClientInStore("Jose Alves", "962310421");
00031             }
00032             catch (IOException ioExcep)
00033             {
00034                 Console.WriteLine(ioExcep.Message);
00035             }
00036             catch (Exception excep)
00037             {
00038                 Console.WriteLine(excep.Message);
00039             }
00040
00041             try
00042             {
00043                 a = ProductManagement.CreateMakeInStore("Benfica");
00044                 int id = ProductManagement.GetMakeIdFromName("Benfica");
00045                 a = ProductManagement.CreateCategoryInStore("Melhor do Mundo");
00046                 int cat = ProductManagement.GetCategoryIdFromName("Melhor do Mundo");
00047                 a = ProductManagement.CreateNewProductInStore("1A34", 23.9m, id, cat , 3, "Nao gostar
      de batatas.");
00048             }
00049             catch (Exception e)
00050             {
00051                 Console.WriteLine(e.Message);
00052             }
00053
00054
00055
00056
00057         }
00058     }
00059 }
```

7.2 BestSale.cs

```
00001 using System;
00002 using System.Collections.Generic;
00003 using System.Linq;
00004 using System.Text;
00005 using System.Threading.Tasks;
00006 using System.Text.RegularExpressions; //Used to verify if a string meets certain criteria.
00007
00008
00009
00010 namespace BestSale
00011 {
00012     class BestSale
00013     {
00014         static void Main(string[] args)
00015         {
00016             Console.OutputEncoding = System.Text.Encoding.UTF8;
00017
00018             //Client clientTest = new Client("Jose Alves","962310421");
00019             //Client clientTest2 = new Client("Rui Jordao", "931250420");
00020             //Client clientTest1 = new Client();
00021
00022             //Make benfica = new Make("Benfica");
00023             //Category futebol = new Category("Futebol");
00024             //Product product1 = new Product("1A34", 23.9m, new Warranty("1A34",3, "Nao gostar de
      batatas."), benfica, futebol );
00025
00026             //Make braga = new Make("Braga");
00027             //Product product2 = new Product("2V45", 15.9m, new Warranty("2V45", 3, "Nao gostar de
      peixe."), braga, futebol);
00028
00029             //Sale sale = new Sale();
00030             //sale.InsertProductOnSale(product1);
00031             //sale.InsertProductOnSale(product2);
00032
00033             //Console.WriteLine(sale.ToString());
00034
00035             //Clients clientes = new Clients();
00036             //clientes.AddClient(clientTest);
00037             //clientes.AddClient(clientTest2);
00038
00039             //foreach (Client client in clientes.ClientList)
00040             //{
00041             //    Console.WriteLine(client.ToString());
00042             //}
00043
00044             bool teste=
00045
00046         }
00047     }
00048 }
```

7.3 .NETFramework,Version=v4.7.2.AssemblyAttributes.cs

```
00001 // <autogenerated />
00002 using System;
00003 using System.Reflection;
00004 [assembly: global::System.Runtime.Versioning.TargetFrameworkAttribute(".NETFramework,Version=v4.7.2",
      FrameworkDisplayName = ".NET Framework 4.7.2")]
```

7.4 .NETFramework,Version=v4.7.2.AssemblyAttributes.cs

```
00001 // <autogenerated />
00002 using System;
00003 using System.Reflection;
00004 [assembly: global::System.Runtime.Versioning.TargetFrameworkAttribute(".NETFramework,Version=v4.7.2",
      FrameworkDisplayName = ".NET Framework 4.7.2")]
```

7.5 .NETFramework,Version=v4.7.2.AssemblyAttributes.cs

```
00001 // <autogenerated />
00002 using System;
00003 using System.Reflection;
00004 [assembly: global::System.Runtime.Versioning.TargetFrameworkAttribute(".NETFramework,Version=v4.7.2",
      FrameworkDisplayName = ".NET Framework 4.7.2")]
```

7.6 .NETFramework,Version=v4.7.2.AssemblyAttributes.cs

```
00001 // <autogenerated />
00002 using System;
00003 using System.Reflection;
00004 [assembly: global::System.Runtime.Versioning.TargetFrameworkAttribute(".NETFramework,Version=v4.7.2",
      FrameworkDisplayName = ".NET Framework 4.7.2")]
```

7.7 .NETFramework,Version=v4.7.2.AssemblyAttributes.cs

```
00001 // <autogenerated />
00002 using System;
00003 using System.Reflection;
00004 [assembly: global::System.Runtime.Versioning.TargetFrameworkAttribute(".NETFramework,Version=v4.7.2",
      FrameworkDisplayName = ".NET Framework 4.7.2")]
```

7.8 .NETFramework,Version=v4.7.2.AssemblyAttributes.cs

```
00001 // <autogenerated />
00002 using System;
00003 using System.Reflection;
00004 [assembly: global::System.Runtime.Versioning.TargetFrameworkAttribute(".NETFramework,Version=v4.7.2",
      FrameworkDisplayName = ".NET Framework 4.7.2")]
```

7.9 .NETFramework,Version=v4.7.2.AssemblyAttributes.cs

```
00001 // <autogenerated />
00002 using System;
00003 using System.Reflection;
00004 [assembly: global::System.Runtime.Versioning.TargetFrameworkAttribute(".NETFramework,Version=v4.7.2",
      FrameworkDisplayName = ".NET Framework 4.7.2")]
```

7.10 AssemblyInfo.cs

```
00001 using System.Reflection;
00002 using System.Runtime.CompilerServices;
00003 using System.Runtime.InteropServices;
00004
00005 // General Information about an assembly is controlled through the following
00006 // set of attributes. Change these attribute values to modify the information
00007 // associated with an assembly.
00008 [assembly: AssemblyTitle("BestSale")]
00009 [assembly: AssemblyDescription("")]
00010 [assembly: AssemblyConfiguration("")]
00011 [assembly: AssemblyCompany("")]
00012 [assembly: AssemblyProduct("BestSale")]
00013 [assembly: AssemblyCopyright("Copyright ©  2024")]
00014 [assembly: AssemblyTrademark("")]
00015 [assembly: AssemblyCulture("")]
00016
00017 // Setting ComVisible to false makes the types in this assembly not visible
00018 // to COM components.  If you need to access a type in this assembly from
00019 // COM, set the ComVisible attribute to true on that type.
00020 [assembly: ComVisible(false)]
00021
00022 // The following GUID is for the ID of the typelib if this project is exposed to COM
00023 [assembly: Guid("e2c10845-2973-4a05-9fdb-fcd73e9c1fc9")]
00024
00025 // Version information for an assembly consists of the following four values:
00026 //
00027 //      Major Version
00028 //      Minor Version
00029 //      Build Number
00030 //      Revision
00031 //
00032 [assembly: AssemblyVersion("1.0.0.0")]
00033 [assembly: AssemblyFileVersion("1.0.0.0")]
```

7.11 AssemblyInfo.cs

```
00001 using System.Reflection;
00002 using System.Runtime.CompilerServices;
00003 using System.Runtime.InteropServices;
00004
00005 // General Information about an assembly is controlled through the following
00006 // set of attributes. Change these attribute values to modify the information
00007 // associated with an assembly.
00008 [assembly: AssemblyTitle("BestSale_Validations")]
00009 [assembly: AssemblyDescription("")]
00010 [assembly: AssemblyConfiguration("")]
00011 [assembly: AssemblyCompany("")]
00012 [assembly: AssemblyProduct("BestSale_Validations")]
00013 [assembly: AssemblyCopyright("Copyright ©  2024")]
00014 [assembly: AssemblyTrademark("")]
00015 [assembly: AssemblyCulture("")]
00016
00017 // Setting ComVisible to false makes the types in this assembly not visible
00018 // to COM components.  If you need to access a type in this assembly from
00019 // COM, set the ComVisible attribute to true on that type.
00020 [assembly: ComVisible(false)]
00021
00022 // The following GUID is for the ID of the typelib if this project is exposed to COM
00023 [assembly: Guid("bd0cbcce-6f1b-4250-868d-a9a63c08bf64")]
00024
00025 // Version information for an assembly consists of the following four values:
00026 //
00027 //      Major Version
00028 //      Minor Version
00029 //      Build Number
00030 //      Revision
00031 //
00032 [assembly: AssemblyVersion("1.0.0.0")]
00033 [assembly: AssemblyFileVersion("1.0.0.0")]
```

7.12 AssemblyInfo.cs

```
00001 using System.Reflection;
00002 using System.Runtime.CompilerServices;
00003 using System.Runtime.InteropServices;
00004
00005 // General Information about an assembly is controlled through the following
00006 // set of attributes. Change these attribute values to modify the information
00007 // associated with an assembly.
00008 [assembly: AssemblyTitle("Business_Layer")]
00009 [assembly: AssemblyDescription("")]
00010 [assembly: AssemblyConfiguration("")]
00011 [assembly: AssemblyCompany("")]
00012 [assembly: AssemblyProduct("Business_Layer")]
00013 [assembly: AssemblyCopyright("Copyright ©  2024")]
00014 [assembly: AssemblyTrademark("")]
00015 [assembly: AssemblyCulture("")]
00016
00017 // Setting ComVisible to false makes the types in this assembly not visible
00018 // to COM components.  If you need to access a type in this assembly from
00019 // COM, set the ComVisible attribute to true on that type.
00020 [assembly: ComVisible(false)]
00021
00022 // The following GUID is for the ID of the typelib if this project is exposed to COM
00023 [assembly: Guid("aa1c97bc-4777-42b0-a03c-aea33be3adbd")]
00024
00025 // Version information for an assembly consists of the following four values:
00026 //
00027 //      Major Version
00028 //      Minor Version
00029 //      Build Number
00030 //      Revision
00031 //
00032 [assembly: AssemblyVersion("1.0.0.0")]
00033 [assembly: AssemblyFileVersion("1.0.0.0")]
```

7.13 AssemblyInfo.cs

```
00001 using System.Reflection;
00002 using System.Runtime.CompilerServices;
00003 using System.Runtime.InteropServices;
00004
00005 // General Information about an assembly is controlled through the following
```

```
00006 // set of attributes. Change these attribute values to modify the information
00007 // associated with an assembly.
00008 [assembly: AssemblyTitle("Business_Object")]
00009 [assembly: AssemblyDescription("")]
00010 [assembly: AssemblyConfiguration("")]
00011 [assembly: AssemblyCompany("")]
00012 [assembly: AssemblyProduct("Business_Object")]
00013 [assembly: AssemblyCopyright("Copyright ©  2024")]
00014 [assembly: AssemblyTrademark("")]
00015 [assembly: AssemblyCulture("")]
00016
00017 // Setting ComVisible to false makes the types in this assembly not visible
00018 // to COM components.  If you need to access a type in this assembly from
00019 // COM, set the ComVisible attribute to true on that type.
00020 [assembly: ComVisible(false)]
00021
00022 // The following GUID is for the ID of the typelib if this project is exposed to COM
00023 [assembly: Guid("5b92d0e5-bc9e-4abe-b03c-4545fa28219f")]
00024
00025 // Version information for an assembly consists of the following four values:
00026 //
00027 //      Major Version
00028 //      Minor Version
00029 //      Build Number
00030 //      Revision
00031 //
00032 [assembly: AssemblyVersion("1.0.0.0")]
00033 [assembly: AssemblyFileVersion("1.0.0.0")]
```

7.14 AssemblyInfo.cs

```
00001 using System.Reflection;
00002 using System.Runtime.CompilerServices;
00003 using System.Runtime.InteropServices;
00004 using System;
00005
00006 // General Information about an assembly is controlled through the following
00007 // set of attributes. Change these attribute values to modify the information
00008 // associated with an assembly.
00009 [assembly: AssemblyTitle("Data_BestSale")]
00010 [assembly: AssemblyDescription("")]
00011 [assembly: AssemblyConfiguration("")]
00012 [assembly: AssemblyCompany("")]
00013 [assembly: AssemblyProduct("Data_BestSale")]
00014 [assembly: AssemblyCopyright("Copyright ©  2024")]
00015 [assembly: AssemblyTrademark("")]
00016 [assembly: AssemblyCulture("")]
00017
00018 // Setting ComVisible to false makes the types in this assembly not visible
00019 // to COM components.  If you need to access a type in this assembly from
00020 // COM, set the ComVisible attribute to true on that type.
00021 [assembly: ComVisible(false)]
00022
00023 // The following GUID is for the ID of the typelib if this project is exposed to COM
00024 [assembly: Guid("93769237-722c-4488-8157-9b0f2f568bab")]
00025
00026 // Version information for an assembly consists of the following four values:
00027 //
00028 //      Major Version
00029 //      Minor Version
00030 //      Build Number
00031 //      Revision
00032 //
00033 [assembly: AssemblyVersion("1.0.0.0")]
00034 [assembly: AssemblyFileVersion("1.0.0.0")]
00035 [assembly: CLSCompliant(true)]
```

7.15 AssemblyInfo.cs

```
00001 using System.Reflection;
00002 using System.Runtime.CompilerServices;
00003 using System.Runtime.InteropServices;
00004
00005 // General Information about an assembly is controlled through the following
00006 // set of attributes. Change these attribute values to modify the information
00007 // associated with an assembly.
00008 [assembly: AssemblyTitle("Exceptions")]
00009 [assembly: AssemblyDescription("")]
00010 [assembly: AssemblyConfiguration("")]
```

```
00011 [assembly: AssemblyCompany("")]
00012 [assembly: AssemblyProduct("Exceptions")]
00013 [assembly: AssemblyCopyright("Copyright ©  2024")]
00014 [assembly: AssemblyTrademark("")]
00015 [assembly: AssemblyCulture("")]
00016
00017 // Setting ComVisible to false makes the types in this assembly not visible
00018 // to COM components.  If you need to access a type in this assembly from
00019 // COM, set the ComVisible attribute to true on that type.
00020 [assembly: ComVisible(false)]
00021
00022 // The following GUID is for the ID of the typelib if this project is exposed to COM
00023 [assembly: Guid("fabe7b09-51fd-47b8-89e3-d85b3554d76b")]
00024
00025 // Version information for an assembly consists of the following four values:
00026 //
00027 //      Major Version
00028 //      Minor Version
00029 //      Build Number
00030 //      Revision
00031 //
00032 [assembly: AssemblyVersion("1.0.0.0")]
00033 [assembly: AssemblyFileVersion("1.0.0.0")]
```

7.16 AssemblyInfo.cs

```
00001 using System.Reflection;
00002 using System.Runtime.CompilerServices;
00003 using System.Runtime.InteropServices;
00004
00005 // General Information about an assembly is controlled through the following
00006 // set of attributes. Change these attribute values to modify the information
00007 // associated with an assembly.
00008 [assembly: AssemblyTitle("trabalhoPOO_27967")]
00009 [assembly: AssemblyDescription("")]
00010 [assembly: AssemblyConfiguration("")]
00011 [assembly: AssemblyCompany("")]
00012 [assembly: AssemblyProduct("trabalhoPOO_27967")]
00013 [assembly: AssemblyCopyright("Copyright ©  2024")]
00014 [assembly: AssemblyTrademark("")]
00015 [assembly: AssemblyCulture("")]
00016
00017 // Setting ComVisible to false makes the types in this assembly not visible
00018 // to COM components.  If you need to access a type in this assembly from
00019 // COM, set the ComVisible attribute to true on that type.
00020 [assembly: ComVisible(false)]
00021
00022 // The following GUID is for the ID of the typelib if this project is exposed to COM
00023 [assembly: Guid("c7a4f6de-be70-4f08-92bd-b54d2a0111f1")]
00024
00025 // Version information for an assembly consists of the following four values:
00026 //
00027 //      Major Version
00028 //      Minor Version
00029 //      Build Number
00030 //      Revision
00031 //
00032 // You can specify all the values or you can default the Build and Revision Numbers
00033 // by using the '*' as shown below:
00034 // [assembly: AssemblyVersion("1.0.*")]
00035 [assembly: AssemblyVersion("1.0.0.0")]
00036 [assembly: AssemblyFileVersion("1.0.0.0")]
```

7.17 BestSale_Validations.cs

```
00001 /*
00002 *   <copyright file="BestSale_Validations.cs" company="IPCA">
00003 *       Copyright (c) 2024 All Rights Reserved
00004 *   </copyright>
00005 *   <author>Jose Alves a27967</author>
00006 *   <date>12/17/2024 6:23:56 PM</date>
00007 *   <description>This file contains the validations to be made by the app.</description>
00008 **/
00009 using System;
00010 using System.Collections.Generic;
00011 using System.Linq;
00012 using System.Text;
00013 using System.Text.RegularExpressions;
00014 using System.Threading.Tasks;
```

```
00015 using Exceptions;
00016
00017 namespace BestSale_Validations
00018 {
00019     public static class BestSale_Validations
00020     {
00021         public static bool ValidatePhoneNumber(string phoneNumber)
00022         {
00023             string pattern = @"^(2|9)\d{8}$"; //Defines the pattern to be a number starting by 9 or 2
      with 8 more numbers after (as a portuguese mobile or landline number).
00024             if (Regex.IsMatch(phoneNumber, pattern)) return true; //Verifies if the value meets the
      criteria.
00025
00026             else throw new InvalidPhoneNumberException();
00027
00028         }
00029     }
00030 }
```

7.18 ClientManagement.cs

```
00001 /*
00002 *   <copyright file="Business_Layer.cs" company="IPCA">
00003 *       Copyright (c) 2024 All Rights Reserved
00004 *   </copyright>
00005 *   <author>zecun</author>
00006 *   <date>12/10/2024 10:57:31 PM</date>
00007 *   <description>This file has all the necessary calls to the back end to manage
      clients.</description>
00008 **/
00009 using System;
00010 using System.Xml.Schema;
00011 using Business_Object;
00012 using Data_BestSale;
00013 using BestSale_Validations;
00014
00015 namespace Business_Layer
00016 {
00024     public static class ClientManagement
00025     {
00026
00027         #region Methods
00028
00029         #region Overrides
00030         #endregion
00031
00032         #region OtherMethods
00040         public static bool CreateClientInStore(string name, string contact)
00041         {
00042             try
00043             {
00044                 if (BestSale_Validations.BestSale_Validations.ValidatePhoneNumber(contact))
00045                 {
00046                     bool aux = Client.CreateClientFromNameContact(name, contact, out Client
      newClient);
00047                     aux = Store.InsertClientInStore(newClient);
00048                     return aux;
00049                 }
00050                 return false;
00051             }
00052             catch(Exceptions.InvalidPhoneNumberException invalidPhoneNumber)
00053             {
00054                 throw invalidPhoneNumber;
00055             }
00056             catch (Exception excep)
00057             {
00058                 throw (excep);
00059             }
00060         }
00061         #endregion
00062
00063         #endregion
00064     }
00065 }
```

7.19 FileManagement.cs

```
00001 /*
00002 *   <copyright file="Business_Layer.cs" company="IPCA">
```

```
00003 *       Copyright (c) 2024 All Rights Reserved
00004 *   </copyright>
00005 *   <author>zecun</author>
00006 *   <date>12/11/2024 12:04:37 PM</date>
00007 *   <description>This File contains the calls to the methods to save data to a file.</description>
00008 **/
00009 using Data_BestSale;
00010 using System;
00011 using System.IO;
00012
00013 namespace Business_Layer
00014 {
00022     public static class FileManagement
00023     {
00024
00025         #region Methods
00026
00034         public static bool SaveStore(Store store, string fileName) {
00035             try
00036             {
00037                 bool a= store.SaveStoreBin(fileName);
00038                 return a;
00039             }
00040             catch (IOException e)
00041             {
00042                 throw e;
00043
00044             }
00045             catch (Exception excep)
00046             {
00047                 throw excep;
00048             }
00049
00050         }
00051
00052         public static bool LoadStore(string fileName)
00053         {
00054             try
00055             {
00056                 Store.LoadStoreBin(fileName);
00057                 return true;
00058
00059             }
00060             catch (IOException e)
00061             {
00062                 throw e;
00063
00064             }
00065             catch (Exception excep)
00066             {
00067                 throw excep;
00068             }
00069         }
00070
00071         public static void ClearStoreMemory()
00072         {
00073             Store.ClearStore();
00074         }
00075
00076
00077         #endregion
00078     }
00079 }
```

7.20 ProductManagement.cs

```
00001 /*
00002 *   <copyright file="Business_Layer.cs" company="IPCA">
00003 *       Copyright (c) 2024 All Rights Reserved
00004 *   </copyright>
00005 *   <author>zecun</author>
00006 *   <date>12/14/2024 4:28:51 PM</date>
00007 *   <description>This file has all the necessary calls to the back end to manage products, categories,
      makes and warranties.</description>
00008 **/
00009 using Data_BestSale;
00010 using System;
00011
00012 namespace Business_Layer
00013 {
00021     public static class ProductManagement
00022     {
00023
```

```
00024
00025         #region Methods
00026
00027         #region Overrides
00028         #endregion
00029
00030         #region OtherMethods
00031         #region Products
00043         public static bool CreateNewProductInStore(string reff, decimal price, int makeID, int
      categoryID, int warrantyDuration, string warrantyConditions)
00044         {
00045             try
00046             {
00047                 Product prod = Product.CreateProductWithWarranty(reff, price, makeID, categoryID,
      warrantyDuration, warrantyConditions);
00048                 Store.InsertProductInStore(prod);
00049                 return true;
00050             }
00051             catch(Exception excep)
00052             {
00053                 throw excep;
00054             }
00055         }
00056
00062         public static decimal GetProductPriceFromReference(string reference)
00063         {
00064             return Store.GetProductPriceInStoreFromReference(reference);
00065         }
00066         #endregion
00067
00068         #region Make
00074         public static bool CreateMakeInStore(string name)
00075         {
00076             Make.CreateMake(name, out Make newMake);
00077             return Store.InsertMakeInStore(newMake);
00078         }
00079
00085         public static int GetMakeIdFromName(string name)
00086         {
00087             return Store.GetMakeIdFromNameInStore(name);
00088         }
00089
00090         #endregion
00091
00092         #region Category
00099         public static bool CreateCategoryInStore(string name)
00100         {
00101             Category.CreateCategory(name, out Category newCategory);
00102             return Store.InsertCategoryInStore(newCategory);
00103         }
00104
00110         public static int GetCategoryIdFromName(string name)
00111         {
00112             return Store.GetCategoryIdFromNameInStore(name);
00113         }
00114         #endregion
00115
00116
00117         #endregion
00118
00119
00120
00121         #endregion
00122     }
00123 }
```

7.21 SimpleProduct.cs

```
00001 /*
00002 *   <copyright file="Business_Object.cs" company="IPCA">
00003 *       Copyright (c) 2024 All Rights Reserved
00004 *   </copyright>
00005 *   <author>zecun</author>
00006 *   <date>12/11/2024 11:18:40 AM</date>
00007 *   <description>This File contains the definition and methods to manage a SimpleClient</description>
00008 **/
00009 using System;
00010
00011 namespace Business_Object
00012 {
00020     public class SimpleProduct
00021     {
00022         #region Attributes
```

```
00023         string _reference;
00024         decimal _price;
00025         int _makeID;
00026         #endregion
00027
00028         #region Methods
00029
00030         #region Constructors
00031
00035         public SimpleProduct()
00036         {
00037         }
00038
00045         public SimpleProduct(string reff, decimal price, int make) {
00046             _reference = reff;
00047             _price = price;
00048             _makeID = make;
00049         }
00050
00051         #endregion
00052
00053         #region Properties
00057         public string Reference
00058         {
00059             get { return _reference; }
00060             set { _reference = value; }
00061
00062         }
00063
00067         public decimal Price
00068         {
00069             get { return _price; }
00070             set { _price = value; }
00071         }
00072
00076         public int Make
00077         {
00078             get { return _makeID; }
00079             set { _makeID = value; }
00080
00081         }
00082
00083
00084         #endregion
00085
00086
00087
00088         #region Overrides
00089         #endregion
00090
00091         #region OtherMethods
00092         #endregion
00093
00094         #region Destructor
00098         ~SimpleProduct()
00099         {
00100         }
00101         #endregion
00102
00103         #endregion
00104     }
00105 }
```

7.22 Campaign.cs

```
00001 /*
00002 *   <copyright file="Data_BestSale.cs" company="IPCA">
00003 *       Copyright (c) 2024 All Rights Reserved
00004 *   </copyright>
00005 *   <author>Jose Alves a27967</author>
00006 *   <date>11/6/2024 11:21:43 AM</date>
00007 *   <description>Definition of Campaign and methods to deal with Campaign operations.</description>
00008 **/
00009 using System;
00010
00011 namespace Data_BestSale
00012 {
00013
00014     [Serializable]
00022     public class Campaign
00023     {
00024         #region Attributes
00025         int _id;
```

```
00026         string _name;
00027         decimal _discount;
00028         DateTime _startDate;
00029         DateTime _endDate;
00030         static int _campaignCount;
00031         #endregion
00032
00033         #region Methods
00034
00035         #region Constructors
00036
00040         public Campaign()
00041         {
00042             _id = ++_campaignCount;
00043             _name = string.Empty;
00044             _discount = 0m;
00045             _startDate = DateTime.MinValue;
00046             _endDate = DateTime.MaxValue;
00047         }
00048
00057         public Campaign(string name, decimal discount, DateTime startDate, DateTime endDate)
00058         {
00059             _id = ++_campaignCount;
00060             _name = name;
00061             _discount = discount;
00062             _startDate = startDate;
00063             _endDate = endDate;
00064         }
00065
00066
00067
00068
00069
00070         #endregion
00071
00072         #region Properties
00073
00077         public int Id{
00078             get { return _id; }
00079             set { _id = value; }
00080         }
00081
00085         public string Name
00086         {
00087             get { return _name; }
00088             set { _name = value; }
00089         }
00090
00094         public decimal Discount
00095         {
00096             get { return _discount; }
00097             set { _discount = value; }
00098         }
00099
00103         public DateTime StartDate
00104         {
00105             get { return _startDate; }
00106             set { _startDate = value; }
00107         }
00108
00112         public DateTime EndDate
00113         {
00114             get { return _endDate; }
00115             set { _endDate = value; }
00116         }
00117
00121         public int CampaignCount
00122         {
00123             get { return _campaignCount; }
00124             set { _campaignCount = value; }
00125         }
00126         #endregion
00127
00128
00129
00130         #region Overrides
00136         public override bool Equals(object obj)
00137         {
00139             if (obj == null)
00140             {
00141                 return false;
00142             }
00143
00145             Campaign camp = obj as Campaign;
00146             return (this.Id == camp.Id && _name == camp.Name);
00147         }
00148
```

```
00155         public static bool operator ==(Campaign camp1, Campaign camp2)
00156         {
00157             return (camp1.Equals(camp2));
00158         }
00159
00166         public static bool operator !=(Campaign camp1, Campaign camp2)
00167         {
00168             return !(camp1.Equals(camp2));
00169         }
00170         #endregion
00171
00172         #region OtherMethods
00173
00179         static public bool VerifyApplicability(Campaign camp)
00180         {
00182             if (camp.StartDate <= DateTime.Now && camp.EndDate >= DateTime.Now)
00183             {
00184                 return true;
00185             }
00186
00187             return false;
00188         }
00189         #endregion
00190
00191         #region Destructor
00195         ~Campaign()
00196         {
00197         }
00198         #endregion
00199
00200         #endregion
00201     }
00202 }
```

7.23 Campaign.cs

```
00001 /*
00002 *   <copyright file="trabalhoPOO_27967.cs" company="IPCA">
00003 *       Copyright (c) 2024 All Rights Reserved
00004 *   </copyright>
00005 *   <author>Jose Alves a27967</author>
00006 *   <date>11/6/2024 11:21:43 AM</date>
00007 *   <description></description>
00008 **/
00009 using System;
00010
00011 namespace trabalhoPOO_27967
00012 {
00020     public class Campaign
00021     {
00022         #region Attributes
00023         int _id;
00024         string _name;
00025         decimal _discount;
00026         DateTime _startDate;
00027         DateTime _endDate;
00028         static int _campaignCount;
00029         #endregion
00030
00031         #region Methods
00032
00033         #region Constructors
00034
00038         public Campaign()
00039         {
00040             _id = ++_campaignCount;
00041             _name = string.Empty;
00042             _discount = 0m;
00043             _startDate = DateTime.MinValue;
00044             _endDate = DateTime.MaxValue;
00045         }
00046
00055         public Campaign(string name, decimal discount, DateTime startDate, DateTime endDate)
00056         {
00057             _id = ++_campaignCount;
00058             _name = name;
00059             _discount = discount;
00060             _startDate = startDate;
00061             _endDate = endDate;
00062         }
00063
00064
00065
```

```
00066
00067
00068         #endregion
00069
00070         #region Properties
00071
00075         public int Id{
00076             get { return _id; }
00077             set { _id = value; }
00078         }
00079
00083         public string Name
00084         {
00085             get { return _name; }
00086             set { _name = value; }
00087         }
00088
00092         public decimal Discount
00093         {
00094             get { return _discount; }
00095             set { _discount = value; }
00096         }
00097
00101         public DateTime StartDate
00102         {
00103             get { return _startDate; }
00104             set { _startDate = value; }
00105         }
00106
00110         public DateTime EndDate
00111         {
00112             get { return _endDate; }
00113             set { _endDate = value; }
00114         }
00115
00119         public int CampaignCount
00120         {
00121             get { return _campaignCount; }
00122             set { _campaignCount = value; }
00123         }
00124         #endregion
00125
00126
00127
00128         #region Overrides
00134         public override bool Equals(object obj)
00135         {
00137             if (obj == null)
00138             {
00139                 return false;
00140             }
00141
00143             Campaign camp = obj as Campaign;
00144             return (this.Id == camp.Id && _name == camp.Name);
00145         }
00146
00153         public static bool operator ==(Campaign camp1, Campaign camp2)
00154         {
00155             return (camp1.Equals(camp2));
00156         }
00157
00164         public static bool operator !=(Campaign camp1, Campaign camp2)
00165         {
00166             return !(camp1.Equals(camp2));
00167         }
00168         #endregion
00169
00170         #region OtherMethods
00171
00177         static public bool VerifyApplicability(Campaign camp)
00178         {
00180             if (camp.StartDate <= DateTime.Now && camp.EndDate >= DateTime.Now)
00181             {
00182                 return true;
00183             }
00184
00185             return false;
00186         }
00187         #endregion
00188
00189         #region Destructor
00193         ~Campaign()
00194         {
00195         }
00196         #endregion
00197
00198         #endregion
```

```
00199     }
00200 }
```

7.24 Campaigns.cs

```
00001 /*
00002 *   <copyright file="Data_BestSale.cs" company="IPCA">
00003 *       Copyright (c) 2024 All Rights Reserved
00004 *   </copyright>
00005 *   <author>Jose Alves a27967</author>
00006 *   <date>11/14/2024 3:57:04 PM</date>
00007 *   <description>This file has the definition and methods to work with the plurality of
      Campaign.</description>
00008 **/
00009 using System;
00010 using System.Collections.Generic;
00011 using System.Xml.Linq;
00012
00013 namespace Data_BestSale
00014 {
00015     [Serializable]
00023     public class Campaigns : IListManagement
00024     {
00025         #region Attributes
00026         List<Campaign> _camps;
00027         #endregion
00028
00029         #region Methods
00030
00031         #region Constructors
00032
00036         public Campaigns()
00037         {
00038             _camps = new List<Campaign>();
00039         }
00040
00045         public Campaigns(List<Campaign> p) {
00046             _camps = p;
00047         }
00048
00049         #endregion
00050
00051         #region Properties
00055         public List<Campaign> Camps
00056         {
00057             get{  return _camps; }
00058             set{ _camps = value; }
00059         }
00060         #endregion
00061
00062
00063
00064         #region Overrides
00065
00066         #endregion
00067
00068         #region OtherMethods
00074         public bool Add(object obj)
00075         {
00076             if(obj==null)return false;
00077             if (obj is Campaign)
00078             {
00079                 _camps.Add((Campaign)obj);
00080                 return true;
00081             }
00082             return false;
00083         }
00084
00090         public bool Remove(object obj)
00091         {
00092             if(obj==null) return false;
00093             var aux=obj as Campaign;
00094             if (Exist(aux.Id) || Exist(aux.Name))
00095             {
00096
00097                 _camps.Remove((Campaign)obj);
00098                 return true;
00099
00100             }
00101             return false;
00102         }
00103
00109         public bool Exist(object obj)
```

```
00110         {
00111             if (obj == null) return false;
00112             var aux=obj as Campaign;
00113             foreach (Campaign c in _camps)
00114             {
00115                 if (c.Id == aux.Id || c.Name==aux.Name)
00116                 {
00117                     return true;
00118                 }
00119             }
00120             return false;
00121         }
00122 
00126         public void ClearCampaigns()
00127         {
00128             _camps.Clear();
00129         }
00130         #endregion
00131 
00132         #region Destructor
00136         ~Campaigns()
00137         {
00138         }
00139         #endregion
00140 
00141         #endregion
00142     }
00143 }
```

7.25 Campaigns.cs

```
00001 /*
00002 *   <copyright file="trabalhoPOO_27967.cs" company="IPCA">
00003 *       Copyright (c) 2024 All Rights Reserved
00004 *   </copyright>
00005 *   <author>Jose Alves a27967</author>
00006 *   <date>11/14/2024 3:57:04 PM</date>
00007 *   <description></description>
00008 **/
00009 using System;
00010 using System.Collections.Generic;
00011 using System.Xml.Linq;
00012 using trabalhoPOO_27967.Interface;
00013 
00014 namespace trabalhoPOO_27967
00015 {
00023     public class Campaigns : IListManagement
00024     {
00025         #region Attributes
00026         List<Campaign> _camps;
00027         #endregion
00028 
00029         #region Methods
00030 
00031         #region Constructors
00032 
00036         public Campaigns()
00037         {
00038             _camps = new List<Campaign>();
00039         }
00040 
00045         public Campaigns(List<Campaign> p) {
00046             _camps = p;
00047         }
00048 
00049         #endregion
00050 
00051         #region Properties
00055         public List<Campaign> Camps
00056         {
00057             get{  return _camps; }
00058             set{ _camps = value; }
00059         }
00060         #endregion
00061 
00062 
00063 
00064         #region Overrides
00065 
00066         #endregion
00067 
00068         #region OtherMethods
00074         public bool Add(object obj)
```

```
00075         {
00076             if(obj==null)return false;
00077             if (obj is Campaign)
00078             {
00079                 _camps.Add((Campaign)obj);
00080                 return true;
00081             }
00082             return false;
00083         }
00084 
00090         public bool Remove(object obj)
00091         {
00092             if(obj==null) return false;
00093             var aux=obj as Campaign;
00094             if (Exist(aux.Id) || Exist(aux.Name))
00095             {
00096 
00097                 _camps.Remove((Campaign)obj);
00098                 return true;
00099 
00100             }
00101             return false;
00102         }
00103 
00109         public bool Exist(object obj)
00110         {
00111             if (obj == null) return false;
00112             var aux=obj as Campaign;
00113             foreach (Campaign c in _camps)
00114             {
00115                 if (c.Id == aux.Id || c.Name==aux.Name)
00116                 {
00117                     return true;
00118                 }
00119             }
00120             return false;
00121         }
00122         #endregion
00123 
00124         #region Destructor
00128         ~Campaigns()
00129         {
00130         }
00131         #endregion
00132 
00133         #endregion
00134     }
00135 }
```

7.26 Categories.cs

```
00001 /*
00002 *   <copyright file="Data_BestSale.cs" company="IPCA">
00003 *       Copyright (c) 2024 All Rights Reserved
00004 *   </copyright>
00005 *   <author>Jose Alves a27967</author>
00006 *   <date>11/14/2024 4:45:58 PM</date>
00007 *   <description>This file has the definition and methods to work with the plurality of
      Category.</description>
00008 **/
00009 using System;
00010 using System.Collections.Generic;
00011 using System.Xml.Linq;
00012 
00013 namespace Data_BestSale
00014 {
00015     [Serializable]
00023     public class Categories : IListManagement
00024     {
00025         #region Attributes
00026         List<Category> _cats;
00027         #endregion
00028 
00029         #region Methods
00030 
00031         #region Constructors
00032 
00036         public Categories()
00037         {
00038             _cats = new List<Category>();
00039         }
00040 
00045         public Categories(List<Category> cats)
```

```
00046         {
00047             _cats = cats;
00048         }
00049 
00050 
00051         #endregion
00052 
00053         #region Properties
00057         public List<Category> Cats
00058         {
00059             get { return _cats; }
00060             set { _cats = value; }
00061         }
00062         #endregion
00063 
00064 
00065 
00066         #region Overrides
00067         #endregion
00068 
00069         #region OtherMethods
00075         public bool Add(object obj)
00076         {
00077             if (obj == null) return false;
00078             if (obj is Category)
00079             {
00080                 _cats.Add((Category)obj);
00081                 return true;
00082             }
00083             return false;
00084         }
00085 
00091         public bool Remove(object obj)
00092         {
00093             if (obj == null) return false;
00094             var aux = obj as Category;
00095             if (Exist(aux.Id) || Exist(aux.Name))
00096             {
00097                 _cats.Remove((Category)obj);
00098                 return true;
00099 
00100             }
00101             return false;
00102         }
00103 
00109         public bool Exist(object obj)
00110         {
00111             if (obj == null) return false;
00112             var aux=obj as Category;
00113 
00115             if (obj is int)
00116             {
00117                 foreach (Category cate in _cats)
00118                 {
00119                     if (cate.Id == aux.Id)
00120                     {
00121                         return true;
00122                     }
00123                 }
00124             }
00125 
00127             if(obj is string)
00128             {
00129                 foreach (Category cate in _cats)
00130                 {
00131                     if (cate.Name == aux.Name)
00132                     {
00133                         return true;
00134                     }
00135                 }
00136             }
00137             return false;
00138         }
00139 
00143         public bool ClearCategories()
00144         {
00145             try
00146             {
00147                 _cats.Clear();
00148                 return true;
00149             }
00150             catch
00151             {
00152                 return false;
00153             }
00154         }
00155 
```

```
00161         public Category GetCategory(object obj)
00162         {
00163             if (obj == null) return null;
00164             if (obj is int)
00165             {
00166                 if (this.Exist((int)obj))
00167                 {
00168                     foreach (Category cat in _cats)
00169                     {
00170                         if (cat.Id == (int)obj)
00171                         {
00172                             return cat;
00173                         }
00174                     }
00175                 }
00176             }
00177             if (obj is string)
00178             {
00179                 if (this.Exist((string)obj))
00180                 {
00181                     foreach (Category cat in _cats)
00182                     {
00183                         if (cat.Name == (string)obj)
00184                         {
00185                             return cat;
00186                         }
00187                     }
00188                 }
00189             }
00190             return null;
00191         }
00192 
00193         #endregion
00194 
00195         #region Destructor
00199         ~Categories()
00200         {
00201         }
00202         #endregion
00203 
00204         #endregion
00205     }
00206 }
```

7.27 Categories.cs

```
00001 /*
00002 *   <copyright file="trabalhoPOO_27967.Category.cs" company="IPCA">
00003 *       Copyright (c) 2024 All Rights Reserved
00004 *   </copyright>
00005 *   <author>Jose Alves a27967</author>
00006 *   <date>11/14/2024 4:45:58 PM</date>
00007 *   <description></description>
00008 **/
00009 using System;
00010 using System.Collections.Generic;
00011 using System.Xml.Linq;
00012 using trabalhoPOO_27967.Interface;
00013 
00014 namespace trabalhoPOO_27967
00015 {
00023     public class Categories : IListManagement
00024     {
00025         #region Attributes
00026         List<Category> _cats;
00027         #endregion
00028 
00029         #region Methods
00030 
00031         #region Constructors
00032 
00036         public Categories()
00037         {
00038             _cats = new List<Category>();
00039         }
00040 
00045         public Categories(List<Category> cats)
00046         {
00047             _cats = cats;
00048         }
00049 
00050 
00051         #endregion
```

```
00052
00053         #region Properties
00057         public List<Category> Cats
00058         {
00059             get { return _cats; }
00060             set { _cats = value; }
00061         }
00062         #endregion
00063
00064
00065
00066         #region Overrides
00067         #endregion
00068
00069         #region OtherMethods
00075         public bool Add(object obj)
00076         {
00077             if (obj == null) return false;
00078             if (obj is Category)
00079             {
00080                 _cats.Add((Category)obj);
00081                 return true;
00082             }
00083             return false;
00084         }
00085
00091         public bool Remove(object obj)
00092         {
00093             if (obj == null) return false;
00094             var aux = obj as Category;
00095             if (Exist(aux.Id) || Exist(aux.Name))
00096             {
00097                 _cats.Remove((Category)obj);
00098                 return true;
00099
00100             }
00101             return false;
00102         }
00103
00109         public bool Exist(object obj)
00110         {
00111             if (obj == null) return false;
00112             var aux=obj as Category;
00113
00115             if (obj is int)
00116             {
00117                 foreach (Category cate in _cats)
00118                 {
00119                     if (cate.Id == aux.Id)
00120                     {
00121                         return true;
00122                     }
00123                 }
00124             }
00125
00127             if(obj is string)
00128             {
00129                 foreach (Category cate in _cats)
00130                 {
00131                     if (cate.Name == aux.Name)
00132                     {
00133                         return true;
00134                     }
00135                 }
00136             }
00137             return false;
00138         }
00139
00140
00141         #endregion
00142
00143         #region Destructor
00147         ~Categories()
00148         {
00149         }
00150         #endregion
00151
00152         #endregion
00153     }
00154 }
```

7.28 Category.cs

```
00001 /*
```

```
00002 *   <copyright file="Data_BestSale.cs" company="IPCA">
00003 *       Copyright (c) 2024 All Rights Reserved
00004 *   </copyright>
00005 *   <author>Jose Alves a27967</author>
00006 *   <date>11/6/2024 11:20:47 AM</date>
00007 *   <description>Definition of Category and methods to deal with Category operations.</description>
00008 **/
00009 using System;
00010
00011 namespace Data_BestSale
00012 {
00013     [Serializable]
00021     public class Category
00022     {
00023         #region Attributes
00024         int _id;
00025         string _name;
00026         static int _catCount=0;
00027         #endregion
00028
00029         #region Methods
00030
00031         #region Constructors
00032
00036         public Category()
00037         {
00038             _id = ++_catCount;
00039             _name = string.Empty;
00040         }
00041
00046         public Category(string name)
00047         {
00048             _id = ++_catCount;
00049             _name = name;
00050         }
00051         #endregion
00052
00053         #region Properties
00054
00058         public int Id
00059         {
00060             get { return _id; }
00061             set { _id = value; }
00062         }
00063
00067         public string Name
00068         {
00069             get { return _name; }
00070             set { _name = value; }
00071         }
00072         #endregion
00073
00074
00075
00076         #region Overrides
00082         public override bool Equals(object obj)
00083         {
00085             if (obj == null)
00086             {
00087                 return false;
00088             }
00089
00091             Category cat = obj as Category;
00092             return (this.Id == cat.Id || _name == cat.Name);
00093         }
00094
00101         public static bool operator ==(Category cat1, Category cat2)
00102         {
00103             return (cat1.Equals(cat2));
00104         }
00105
00112         public static bool operator !=(Category cat1, Category cat2)
00113         {
00114             return !(cat1.Equals(cat2));
00115         }
00116         #endregion
00117
00118         #region OtherMethods
00126         public static bool CreateCategory(string name, out Category category)
00127         {
00128             try
00129             {
00130                 category = new Category(name);
00131                 return true;
00132             }
00133             catch (Exception e)
00134             {
```

```
00135                 throw e;
00136             }
00137         }
00138         #endregion
00139
00140         #region Destructor
00144         ~Category()
00145         {
00146         }
00147         #endregion
00148
00149         #endregion
00150     }
00151 }
```

7.29 Category.cs

```
00001 /*
00002 *   <copyright file="trabalhoPOO_27967.cs" company="IPCA">
00003 *       Copyright (c) 2024 All Rights Reserved
00004 *   </copyright>
00005 *   <author>Jose Alves a27967</author>
00006 *   <date>11/6/2024 11:20:47 AM</date>
00007 *   <description></description>
00008 **/
00009 using System;
00010
00011 namespace trabalhoPOO_27967
00012 {
00020     public class Category
00021     {
00022         #region Attributes
00023         int _id;
00024         string _name;
00025         static int _catCount=0;
00026         #endregion
00027
00028         #region Methods
00029
00030         #region Constructors
00031
00035         public Category()
00036         {
00037             _id = ++_catCount;
00038             _name = string.Empty;
00039         }
00040
00045         public Category(string name)
00046         {
00047             _id = ++_catCount;
00048             _name = name;
00049         }
00050         #endregion
00051
00052         #region Properties
00053
00057         public int Id
00058         {
00059             get { return _id; }
00060             set { _id = value; }
00061         }
00062
00066         public string Name
00067         {
00068             get { return _name; }
00069             set { _name = value; }
00070         }
00071         #endregion
00072
00073
00074
00075         #region Overrides
00081         public override bool Equals(object obj)
00082         {
00084             if (obj == null)
00085             {
00086                 return false;
00087             }
00088
00090             Category cat = obj as Category;
00091             return (this.Id == cat.Id || _name == cat.Name);
00092         }
00093
```

```
00100         public static bool operator ==(Category cat1, Category cat2)
00101         {
00102             return (cat1.Equals(cat2));
00103         }
00104
00111         public static bool operator !=(Category cat1, Category cat2)
00112         {
00113             return !(cat1.Equals(cat2));
00114         }
00115         #endregion
00116
00117         #region OtherMethods
00118         #endregion
00119
00120         #region Destructor
00124         ~Category()
00125         {
00126         }
00127         #endregion
00128
00129         #endregion
00130     }
00131 }
```

7.30 Client.cs

```
00001 /*
00002 *   <copyright file="Data_BestSale.cs" company="IPCA">
00003 *       Copyright (c) 2024 All Rights Reserved
00004 *   </copyright>
00005 *   <author>Jose Alves a27967</author>
00006 *   <date>10/29/2024 4:23:56 PM</date>
00007 *   <description>Definition of Client and methods to deal with Client operations.
00009 **/
00010 using System;
00011 using System.Text;
00012 using System.Text.RegularExpressions;
00013 using static System.Net.Mime.MediaTypeNames;
00014 using Business_Object;
00015 using BestSale_Validations;
00016 using Exceptions;
00017
00018 namespace Data_BestSale
00019 {
00020     [Serializable]
00028     public class Client
00029     {
00030         #region Attributes
00031         int _clientID;
00032         string _name;
00033         string _contact;
00034         static int _clientCount=0;
00035         #endregion
00036
00037         #region Methods
00038
00039         #region Constructors
00040
00044         public Client()
00045         {
00046             _clientID = ++_clientCount ;
00047             _name = "No Name";
00048             _contact = "999999999";
00049         }
00050
00056         public Client(string n, string c)
00057         {
00058             _clientID = ++_clientCount;
00059             _name = n;
00060             _contact = c;
00061             //COMO POSSO FAZER PARA TESTAR SE A STRING PODE SER CONTATCO? DEVO FAZE-LO NO CONSTRUTOR
      OU FORA?
00062         }
00063
00064         #endregion
00065
00066         #region Properties
00070         public int ClientID
00071         {
00072             get { return _clientID; }
00073             set { _clientID = value; }
00074         }
00075
```

```
00079         public string Name
00080         {
00081             get { return _name; }
00082             set { _name = value; }
00083         }
00084 
00088         public string Contact
00089         {
00090             get { return _contact; }
00091             set
00092             {
00093                 try
00094                 {
00095                 if (BestSale_Validations.BestSale_Validations.ValidatePhoneNumber(value)) _contact =
      value;
00096                 }
00097                 catch(InvalidPhoneNumberException excep)
00098                 {
00099                     throw excep;
00100                 }
00101                 catch(Exception)
00102                 {
00103                     throw new Exception("Invalid Phone Number");
00104                 }
00105             }
00106         }
00107 
00111         public static int ClientCount
00112         {
00113             get { return _clientCount; }
00114 
00115             set { _clientCount = value; }
00116         }
00117         #endregion
00118 
00119 
00120 
00121         #region Overrides
00122 
00127         public override string ToString()
00128         {
00129             StringBuilder sb = new StringBuilder();
00130             sb.AppendLine($"Client ID: {_clientID}");
00131             sb.AppendLine($"Name: {_name}");
00132             sb.AppendLine($"Contact: {_contact}");
00133 
00134             return sb.ToString();
00135         }
00136 
00142         public override bool Equals(object obj)
00143         {
00145             if (obj == null)
00146             {
00147                 return false;
00148             }
00149 
00151             Client client = obj as Client;
00152             return (this._clientID == client._clientID && _name == client._name);
00153         }
00154 
00161         public static bool operator == (Client cli1, Client cli2)
00162         {
00163             return( cli1.Equals(cli2) );
00164         }
00165 
00172         public static bool operator !=(Client cli1, Client cli2)
00173         {
00174             return !(cli1.Equals(cli2));
00175         }
00176         #endregion
00177 
00178 
00179         #region OtherMethods
00180         public static bool CreateClientFromNameContact(string name, string contact, out Client
      newClient)
00181         {
00182             try
00183             {
00184                 newClient = new Client(name, contact);
00185                 return true;
00186             }
00187             catch (Exception excep)
00188             {
00189                 throw (excep);
00190 
00191             }
00192         }
```

```
00193         #endregion
00194
00195         #region Destructor
00199         ~Client()
00200         {
00201         }
00202         #endregion
00203
00204         #endregion
00205     }
00206 }
```

7.31 Client.cs

```
00001 /*
00002 *   <copyright file="trabalhoPOO_27967.cs" company="IPCA">
00003 *       Copyright (c) 2024 All Rights Reserved
00004 *   </copyright>
00005 *   <author>Jose Alves a27967</author>
00006 *   <date>10/29/2024 4:23:56 PM</date>
00007 *   <description></description>
00008 **/
00009 using System;
00010 using System.Text;
00011 using System.Text.RegularExpressions;
00012 using static System.Net.Mime.MediaTypeNames;
00013
00014 namespace trabalhoPOO_27967
00015 {
00023     public class Client
00024     {
00025         #region Attributes
00026         int _clientID;
00027         string _name;
00028         string _contact;
00029         static int _clientCount=0;
00030         #endregion
00031
00032         #region Methods
00033
00034         #region Constructors
00035
00039         public Client()
00040         {
00041             _clientID = ++_clientCount ;
00042             _name = "No Name";
00043             _contact = "999999999";
00044         }
00045
00051         public Client(string n, string c)
00052         {
00053             _clientID = ++_clientCount;
00054             _name = n;
00055             _contact = c;
00056             //COMO POSSO FAZER PARA TESTAR SE A STRING PODE SER CONTATCO? DEVO FAZE-LO NO CONSTRUTOR
      OU FORA?
00057         }
00058
00059         #endregion
00060
00061         #region Properties
00065         public int ClientID
00066         {
00067             get { return _clientID; }
00068             set { _clientID = value; }
00069         }
00070
00074         public string Name
00075         {
00076             get { return _name; }
00077             set { _name = value; }
00078         }
00079
00083         public string Contact
00084         {
00085             get { return _contact; }
00086             set
00087             {
00088                 string pattern = @"^(2|9)\d{8}$"; //Defines the pattern to be a number starting by 9
      or 2 with 8 more numbers after (as a portuguese mobile or landline number).
00089                 bool isGood = Regex.IsMatch(value, pattern); //Verifies if the value meets the
      criteria.
00090
```

```
00091                 if (isGood) _contact = value;
00092
00093                 //COMO POSSO RETORNAR UM ERRO CASO A STRING NAO CORRESPONDA?
00094             }
00095         }
00096
00100         public static int ClientCount
00101         {
00102             get { return _clientCount; }
00103
00104             set { _clientCount = value; }
00105         }
00106         #endregion
00107
00108
00109
00110         #region Overrides
00111
00116         public override string ToString()
00117         {
00118             StringBuilder sb = new StringBuilder();
00119             sb.AppendLine($"Client ID: {_clientID}");
00120             sb.AppendLine($"Name: {_name}");
00121             sb.AppendLine($"Contact: {_contact}");
00122
00123             return sb.ToString();
00124         }
00125
00131         public override bool Equals(object obj)
00132         {
00134             if (obj == null)
00135             {
00136                 return false;
00137             }
00138
00140             Client client = obj as Client;
00141             return (this._clientID == client._clientID && _name == client._name);
00142         }
00143
00150         public static bool operator == (Client cli1, Client cli2)
00151         {
00152             return( cli1.Equals(cli2) );
00153         }
00154
00161         public static bool operator !=(Client cli1, Client cli2)
00162         {
00163             return !(cli1.Equals(cli2));
00164         }
00165         #endregion
00166
00167
00168         #region OtherMethods
00169         #endregion
00170
00171         #region Destructor
00175         ~Client()
00176         {
00177         }
00178         #endregion
00179
00180         #endregion
00181     }
00182 }
```

7.32 Clients.cs

```
00001 /*
00002 *   <copyright file="Data_BestSale.cs" company="IPCA">
00003 *       Copyright (c) 2024 All Rights Reserved
00004 *   </copyright>
00005 *   <author>Jose Alves a27967</author>
00006 *   <date>11/12/2024 9:25:28 PM</date>
00007 *   <description>Class with the definition and methods to manage a list of clients.</description>
00008 **/
00009 using System;
00010 using System.Collections.Generic;
00011
00012 namespace Data_BestSale
00013 {
00014     [Serializable]
00022     public class Clients : IListManagement
00023     {
00024         #region Attributes
```

```
00025         static List<Client> _clientList;
00026         #endregion
00027
00028         #region Methods
00029
00030         #region Constructors
00031
00035         public Clients()
00036         {
00037             _clientList = new List<Client>();
00038         }
00039
00040
00041         #endregion
00042
00043         #region Properties
00044
00048         public List<Client> ClientList
00049         {
00050             get{ return _clientList; }
00051             set{ _clientList = value; }
00052         }
00053         #endregion
00054
00055
00056
00057         #region Overrides
00058         #endregion
00059
00060         #region OtherMethods
00067         public bool Add(object obj)
00068         {
00069             if (obj == null) return false;
00070             if (obj is Client)
00071             {
00072                 if (this.ClientList.Contains((Client)obj)) return false;
00073                 else
00074                 {
00075                     this.ClientList.Add((Client)obj);
00076                     return true;
00077                 }
00078             }
00079             return false;
00080         }
00081
00087         public bool Remove(object obj)
00088         {
00089             if (obj == null) return false;
00090             var aux = (Client) obj;
00091             if (Exist(aux.ClientID))
00092             {
00093                 _clientList.Remove((Client)obj);
00094                 return true;
00095             }
00096             return false;
00097         }
00098
00104         public bool Exist(object obj)
00105         {
00106             if (obj is int)
00107             {
00108                 foreach (Client client in _clientList)
00109                 {
00110                     if (client.ClientID == (int)obj)
00111                     {
00112                         return true;
00113                     }
00114                 }
00115             }
00116             return false;
00117         }
00118
00124         public Client GetClient(int id)
00125         {
00126             foreach (Client client in _clientList)
00127             {
00128                 if (client.ClientID == id)
00129                 {
00130                     return client;
00131                 }
00132             }
00133             return null;
00134         }
00135
00139         public bool ClearClients()
00140         {
00141             try
```

```
00142             {
00143                 _clientList.Clear();
00144                 return true;
00145             }
00146             catch
00147             {
00148                 return false;
00149             }
00150         }
00151         #endregion
00152 
00153         #region Destructor
00157         ~Clients()
00158         {
00159         }
00160         #endregion
00161 
00162         #endregion
00163     }
00164 }
```

7.33 Clients.cs

```
00001 /*
00002 *   <copyright file="trabalhoPOO_27967.cs" company="IPCA">
00003 *       Copyright (c) 2024 All Rights Reserved
00004 *   </copyright>
00005 *   <author>Jose Alves a27967</author>
00006 *   <date>11/12/2024 9:25:28 PM</date>
00007 *   <description></description>
00008 **/
00009 using System;
00010 using System.Collections.Generic;
00011 using trabalhoPOO_27967.Interface;
00012 
00013 namespace trabalhoPOO_27967
00014 {
00022     public class Clients : IListManagement
00023     {
00024         #region Attributes
00025         static List<Client> _clientList;
00026         #endregion
00027 
00028         #region Methods
00029 
00030         #region Constructors
00031 
00035         public Clients()
00036         {
00037             _clientList = new List<Client>();
00038         }
00039 
00040 
00041         #endregion
00042 
00043         #region Properties
00044 
00048         public List<Client> ClientList
00049         {
00050             get{ return _clientList; }
00051             set{ _clientList = value; }
00052         }
00053         #endregion
00054 
00055 
00056 
00057         #region Overrides
00058         #endregion
00059 
00060         #region OtherMethods
00066         public bool Add(object obj)
00067         {
00068             if (obj == null) return false;
00069             if (obj is Client)
00070             {
00071                 this.ClientList.Add((Client)obj);
00072                 return true;
00073             }
00074             return false;
00075         }
00076 
00082         public bool Remove(object obj)
00083         {
```

```
00084             if (obj == null) return false;
00085             var aux = (Client) obj;
00086             if (Exist(aux.ClientID))
00087             {
00088                 _clientList.Remove((Client)obj);
00089                 return true;
00090             }
00091             return false;
00092         }
00093
00099         public bool Exist(object obj)
00100         {
00101             if (obj is int)
00102             {
00103                 foreach (Client client in _clientList)
00104                 {
00105                     if (client.ClientID == (int)obj)
00106                     {
00107                         return true;
00108                     }
00109                 }
00110             }
00111             return false;
00112         }
00113
00119         public Client GetClient(int id)
00120         {
00121             foreach (Client client in _clientList)
00122             {
00123                 if (client.ClientID == id)
00124                 {
00125                     return client;
00126                 }
00127             }
00128             return null;
00129         }
00130         #endregion
00131
00132         #region Destructor
00136         ~Clients()
00137         {
00138         }
00139         #endregion
00140
00141         #endregion
00142     }
00143 }
```

7.34 IListManagement.cs

```
00001 /*
00002 *   <copyright file="Data_BestSale.cs" company="IPCA">
00003 *       Copyright (c) 2024 All Rights Reserved
00004 *   </copyright>
00005 *   <author>Jose Alves a27967</author>
00006 *   <date>11/15/2024 04:21:43 PM</date>
00007 *   <description>Defines the interface to use in List Management</description>
00008 **/
00009 using System;
00010 using System.Collections.Generic;
00011 using System.Linq;
00012 using System.Text;
00013 using System.Threading.Tasks;
00014
00015 namespace Data_BestSale
00016 {
00017     public interface IListManagement
00018     {
00019         bool Add(object obj);
00020
00021         bool Remove(object obj);
00022
00023         bool Exist(object obj);
00024     }
00025 }
```

7.35 IListManagement.cs

```
00001 /*
```

```
00002 *   <copyright file="trabalhoPOO_27967.cs" company="IPCA">
00003 *       Copyright (c) 2024 All Rights Reserved
00004 *   </copyright>
00005 *   <author>Jose Alves a27967</author>
00006 *   <date>11/15/2024 04:21:43 PM</date>
00007 *   <description>Defines the interface to use in List Management</description>
00008 **/
00009 using System;
00010 using System.Collections.Generic;
00011 using System.Linq;
00012 using System.Text;
00013 using System.Threading.Tasks;
00014
00015 namespace trabalhoPOO_27967.Interface
00016 {
00017     public interface IListManagement
00018     {
00019         bool Add(object obj);
00020
00021         bool Remove(object obj);
00022
00023         bool Exist(object obj);
00024     }
00025 }
```

7.36 Make.cs

```
00001 /*
00002 *   <copyright file="Data_BestSale.cs" company="IPCA">
00003 *       Copyright (c) 2024 All Rights Reserved
00004 *   </copyright>
00005 *   <author>Jose Alves a27967</author>
00006 *   <date>11/6/2024 11:22:09 AM</date>
00007 *   <description>Definition of Make and methods to deal with Make operations.</description>
00008 **/
00009 using System;
00010 using System.Diagnostics.Contracts;
00011 using System.Runtime.Remoting.Messaging;
00012 using System.Text;
00013
00014 namespace Data_BestSale
00015 {
00016     [Serializable]
00024     public class Make
00025     {
00026         #region Attributes
00027         int _id;
00028         string _name;
00029         static int _makeCount=0;
00030         #endregion
00031
00032         #region Methods
00033
00034         #region Constructors
00035
00039         public Make()
00040         {
00041             _id = ++_makeCount;
00042             _name = string.Empty;
00043         }
00044
00050         public Make(string name)
00051         {
00052             _id = ++_makeCount;
00053             _name = name;
00054         }
00055
00056
00057
00058         #endregion
00059
00060         #region Properties
00061
00065         public int ID
00066         {
00067             get { return _id; }
00068             set { _id = value; }
00069         }
00070
00074         public string Name
00075         {
00076             get { return _name; }
00077             set { _name = value; }
```

```
00078         }
00079
00080         #endregion
00081
00082
00083
00084         #region Overrides
00089         public override string ToString()
00090         {
00091             return ("Make : " + _name);
00092         }
00093
00099         public override bool Equals(object obj)
00100         {
00102             if (obj == null)
00103             {
00104                 return false;
00105             }
00106
00108             Make make = obj as Make;
00109             return (this.ID == make.ID || this.Name == make.Name);
00110         }
00111
00118         public static bool operator ==(Make m1, Make m2)
00119         {
00120             return (m1.Equals(m2));
00121         }
00122
00129         public static bool operator !=(Make m1, Make m2)
00130         {
00131             return !(m1.Equals(m2));
00132         }
00133         #endregion
00134
00135         #region OtherMethods
00142         public static bool CreateMake(string name, out Make make)
00143         {
00144             try
00145             {
00146                 make = new Make(name);
00147                 return true;
00148             }
00149             catch (Exception e)
00150             {
00151                 throw e;
00152             }
00153         }
00154
00159         public int GetMakeID()
00160         {
00161             return this.ID;
00162         }
00163         #endregion
00164
00165         #region Destructor
00169         ~Make()
00170         {
00171         }
00172         #endregion
00173
00174         #endregion
00175     }
00176 }
```

7.37 Make.cs

```
00001 /*
00002 *   <copyright file="trabalhoPOO_27967.cs" company="IPCA">
00003 *       Copyright (c) 2024 All Rights Reserved
00004 *   </copyright>
00005 *   <author>Jose Alves a27967</author>
00006 *   <date>11/6/2024 11:22:09 AM</date>
00007 *   <description></description>
00008 **/
00009 using System;
00010 using System.Diagnostics.Contracts;
00011 using System.Runtime.Remoting.Messaging;
00012 using System.Text;
00013
00014 namespace trabalhoPOO_27967
00015 {
00023     public class Make
00024     {
```

```
00025         #region Attributes
00026         int _id;
00027         string _name;
00028         static int _makeCount=0;
00029         #endregion
00030
00031         #region Methods
00032
00033         #region Constructors
00034
00038         public Make()
00039         {
00040             _id = ++_makeCount;
00041             _name = string.Empty;
00042         }
00043
00049         public Make(string name)
00050         {
00051             _id = ++_makeCount;
00052             _name = name;
00053         }
00054
00055
00056
00057         #endregion
00058
00059         #region Properties
00060
00064         public int ID
00065         {
00066             get { return _id; }
00067             set { _id = value; }
00068         }
00069
00073         public string Name
00074         {
00075             get { return _name; }
00076             set { _name = value; }
00077         }
00078
00079         #endregion
00080
00081
00082
00083         #region Overrides
00088         public override string ToString()
00089         {
00090             return ("Make : " + _name);
00091         }
00092
00098         public override bool Equals(object obj)
00099         {
00101             if (obj == null)
00102             {
00103                 return false;
00104             }
00105
00107             Make make = obj as Make;
00108             return (this.ID == make.ID || this.Name == make.Name);
00109         }
00110
00117         public static bool operator ==(Make m1, Make m2)
00118         {
00119             return (m1.Equals(m2));
00120         }
00121
00128         public static bool operator !=(Make m1, Make m2)
00129         {
00130             return !(m1.Equals(m2));
00131         }
00132         #endregion
00133
00134         #region OtherMethods
00135         #endregion
00136
00137         #region Destructor
00141         ~Make()
00142         {
00143         }
00144         #endregion
00145
00146         #endregion
00147     }
00148 }
```

7.38 Makes.cs

```
00001 /*
00002 *   <copyright file="Data_BestSale.cs" company="IPCA">
00003 *       Copyright (c) 2024 All Rights Reserved
00004 *   </copyright>
00005 *   <author>Jose Alves a27967</author>
00006 *   <date>11/14/2024 4:33:51 PM</date>
00007 *   <description>This file has the definition and methods to work with the plurality of
      Make.</description>
00008 **/
00009 using System;
00010 using System.Collections.Generic;
00011 using System.Xml.Linq;
00012
00013
00014 namespace Data_BestSale
00015 {
00016     [Serializable]
00024     public class Makes : IListManagement
00025     {
00026         #region Attributes
00027         List<Make> _makeList;
00028         #endregion
00029
00030         #region Methods
00031
00032         #region Constructors
00033
00037         public Makes()
00038         {
00039             _makeList = new List<Make>();
00040         }
00041
00046         public Makes(List<Make> m)
00047         {
00048             _makeList = m;
00049         }
00050         #endregion
00051
00052         #region Properties
00056         public List<Make> MakeList
00057         {
00058             get { return _makeList; }
00059             set { _makeList = value; }
00060         }
00061         #endregion
00062
00063
00064
00065         #region Overrides
00066         #endregion
00067
00068         #region OtherMethods
00074         public bool Add(object obj)
00075         {
00076             if (obj == null) return false;
00077             if (obj is Make) {
00078                 _makeList.Add((Make)obj);
00079                 return true;
00080             }
00081             return false;
00082         }
00083
00089         public bool Remove(object obj)
00090         {
00091             if (obj == null) return false;
00092             var aux = obj as Make;
00093             if (Exist(aux.ID))
00094             {
00095                 _makeList.Remove((Make)obj);
00096                 return true;
00097             }
00098             return false;
00099         }
00100
00106         public bool Exist(object obj)
00107         {
00108             if (obj == null) return false;
00109             if (obj is int)
00110             {
00111                 foreach (Make make in _makeList)
00112                 {
00113                     if (make.ID == (int)obj)
00114                     {
00115                         return true;
00116                     }
```

```
00117                 }
00118             }
00119             if(obj is string)
00120             {
00121                 foreach (Make make in _makeList)
00122                 {
00123                     if (make.Name == (string)obj)
00124                     {
00125                         return true;
00126                     }
00127                 }
00128             }
00129             return false;
00130         }
00131 
00135         public bool ClearMakes()
00136         {
00137             try{
00138             _makeList.Clear();
00139                 return true;
00140             }
00141             catch
00142             {
00143                 return false;
00144             }
00145         }
00146 
00152         public Make GetMake(object obj )
00153         {
00154             if (obj == null) return null;
00155             if (obj is int)
00156             {
00157                 if (this.Exist((int)obj))
00158                 {
00159                     foreach (Make make in _makeList)
00160                     {
00161                         if (make.ID == (int)obj)
00162                         {
00163                             return make;
00164                         }
00165                     }
00166                 }
00167             }
00168             if (obj is string)
00169             {
00170                 if (this.Exist((string)obj))
00171                 {
00172                     foreach (Make make in _makeList)
00173                     {
00174                         if (make.Name == (string)obj)
00175                         {
00176                             return make;
00177                         }
00178                     }
00179                 }
00180             }
00181             return null;
00182         }
00183 
00184         #endregion
00185 
00186         #region Destructor
00190         ~Makes()
00191         {
00192         }
00193         #endregion
00194 
00195         #endregion
00196     }
00197 }
```

7.39 Makes.cs

```
00001 /*
00002 *   <copyright file="trabalhoPOO_27967.cs" company="IPCA">
00003 *       Copyright (c) 2024 All Rights Reserved
00004 *   </copyright>
00005 *   <author>Jose Alves a27967</author>
00006 *   <date>11/14/2024 4:33:51 PM</date>
00007 *   <description></description>
00008 **/
00009 using System;
00010 using System.Collections.Generic;
```

```
00011 using System.Xml.Linq;
00012 using trabalhoPOO_27967.Interface;
00013
00014 namespace trabalhoPOO_27967
00015 {
00023     public class Makes : IListManagement
00024     {
00025         #region Attributes
00026         List<Make> _makeList;
00027         #endregion
00028
00029         #region Methods
00030
00031         #region Constructors
00032
00036         public Makes()
00037         {
00038             _makeList = new List<Make>();
00039         }
00040
00045         public Makes(List<Make> m)
00046         {
00047             _makeList = m;
00048         }
00049         #endregion
00050
00051         #region Properties
00055         public List<Make> MakeList
00056         {
00057             get { return _makeList; }
00058             set { _makeList = value; }
00059         }
00060         #endregion
00061
00062
00063
00064         #region Overrides
00065         #endregion
00066
00067         #region OtherMethods
00073         public bool Add(object obj)
00074         {
00075             if (obj == null) return false;
00076             if (obj is Make) {
00077                 _makeList.Add((Make)obj);
00078                 return true;
00079             }
00080             return false;
00081         }
00082
00088         public bool Remove(object obj)
00089         {
00090             if (obj == null) return false;
00091             var aux = obj as Make;
00092             if (Exist(aux.ID))
00093             {
00094                 _makeList.Remove((Make)obj);
00095                 return true;
00096             }
00097             return false;
00098         }
00099
00105         public bool Exist(object obj)
00106         {
00107             if (obj == null) return false;
00108             if (obj is int)
00109             {
00110                 foreach (Make make in _makeList)
00111                 {
00112                     if (make.ID == (int)obj)
00113                     {
00114                         return true;
00115                     }
00116                 }
00117             }
00118             if(obj is string)
00119             {
00120                 foreach (Make make in _makeList)
00121                 {
00122                     if (make.Name == (string)obj)
00123                     {
00124                         return true;
00125                     }
00126                 }
00127             }
00128             return false;
00129         }
```

```
00130
00131         #endregion
00132
00133         #region Destructor
00137         ~Makes()
00138         {
00139         }
00140         #endregion
00141
00142         #endregion
00143     }
00144 }
```

7.40 Product.cs

```
00001 /*
00002 *   <copyright file="Data_BestSale.cs" company="IPCA">
00003 *       Copyright (c) 2024 All Rights Reserved
00004 *   </copyright>
00005 *   <author>Jose Alves a27967</author>
00006 *   <date>11/2/2024 4:40:12 PM</date>
00007 *   <description> Definition of product and methods to deal with product operations.</description>
00008 **/
00009 using System;
00010 using System.Collections;
00011 using System.Collections.Generic;
00012 using System.Diagnostics.Contracts;
00013 using System.IO;
00014 using System.Runtime.Serialization.Formatters.Binary;
00015 using System.Text;
00016 using System.Xml.Linq;
00017
00018 namespace Data_BestSale
00019 {
00020     [Serializable]
00028     public class Product
00029     {
00030         #region Attributes
00031         string _reference;
00032         decimal _price;
00033         int _makeID;
00034         int _categoryID;
00035         Warranty _warranty; //in Years
00036         int _stock;
00037         #endregion
00038
00039         #region Methods
00040
00041         #region Constructors
00042
00046         public Product()
00047         {
00048             _reference = string.Empty;
00049             _price = -1;
00050             _makeID = -1;
00051             _categoryID = -1;
00052             _stock = 0;
00053         }
00054
00062         public Product(string reff, decimal price, int makeID, int categoryID)
00063         {
00064             _reference = reff;
00065             _price = price;
00066             _makeID = makeID;
00067             _categoryID = categoryID;
00068             _warranty = null;
00069         }
00070
00076         public Product(string reff, decimal price, Warranty warranty, int make, int category)
00077         {
00078             _reference = reff;
00079             _price = price;
00080             _makeID = make;
00081             _categoryID = category;
00082             _warranty = warranty;
00083
00084         }
00085
00086         #endregion
00087
00088         #region Properties
00089
00093         public string Reference
```

```
00094         {
00095             get { return _reference; }
00096             set { _reference = value; }
00097 
00098         }
00099 
00103         public decimal Price
00104         {
00105             get { return _price; }
00106             set { _price = value; }
00107         }
00108 
00112         public int MakeID
00113         {
00114             get { return _makeID; }
00115             set { _makeID = value; }
00116 
00117         }
00118 
00122         public int CategoryID
00123         {
00124             get { return _categoryID; }
00125             set { _categoryID = value; }
00126         }
00127 
00131         public int Stock
00132         {
00133             get { return _stock; }
00134             set { _stock = value; }
00135         }
00136 
00137         public Warranty Warranty
00138         {
00139             get { return _warranty; }
00140             set { _warranty = value; }
00141         }
00142         #endregion
00143 
00144 
00145 
00146         #region Overrides
00152         public override bool Equals(object obj)
00153         {
00154             if (obj == null) return false;
00155             Product product = obj as Product;
00156             if (product.Reference == _reference) return true;
00157             return false;
00158         }
00159 
00166         public static bool operator ==(Product p1, Product p2)
00167         {
00168             return (p1.Equals(p2));
00169         }
00170 
00177         public static bool operator !=(Product p1, Product p2)
00178         {
00179             return !(p1.Equals(p2));
00180         }
00181 
00186         public override string ToString()
00187         {
00188             StringBuilder sb = new StringBuilder();
00189             sb.AppendLine($"Product ID: {_reference}");
00190             sb.AppendLine($"Price: {_price}€");
00191             sb.AppendLine(_makeID.ToString());
00192             sb.AppendLine(_warranty.ToString());
00193 
00194             return sb.ToString();
00195         }
00196 
00197         #endregion
00198 
00199         #region OtherMethods
00210         public static Product CreateProductWithWarranty(string reff, decimal price, int makeID, int
      categoryID,int warrantyDuration, string warrantyConditions)
00211         {
00212             Warranty warranty = Warranty.CreateWarranty(reff, warrantyDuration, warrantyConditions);
00213             Product product = new Product(reff, price, warranty, makeID, categoryID);
00214             return product;
00215         }
00216 
00217         #endregion
00218 
00219         #region Destructor
00223         ~Product()
00224         {
00225         }
```

```
00226         #endregion
00227
00228         #endregion
00229     }
00230 }
```

7.41 Product.cs

```
00001 /*
00002 *   <copyright file="trabalhoPOO_27967.cs" company="IPCA">
00003 *       Copyright (c) 2024 All Rights Reserved
00004 *   </copyright>
00005 *   <author>Jose Alves a27967</author>
00006 *   <date>11/2/2024 4:40:12 PM</date>
00007 *   <description></description>
00008 **/
00009 using System;
00010 using System.Collections;
00011 using System.Collections.Generic;
00012 using System.Diagnostics.Contracts;
00013 using System.Text;
00014 using System.Xml.Linq;
00015
00016 namespace trabalhoPOO_27967
00017 {
00025     public class Product
00026     {
00027         #region Attributes
00028         string _reference;
00029         decimal _price;
00030         int _makeID;
00031         int _categoryID;
00032         Warranty _warranty; //in Years
00033         int _stock;
00034         #endregion
00035
00036         #region Methods
00037
00038         #region Constructors
00039
00043         public Product()
00044         {
00045             _reference = string.Empty;
00046             _price = -1;
00047             _makeID = -1;
00048             _categoryID = -1;
00049             _stock = 0;
00050         }
00051
00057         public Product(string reff, decimal price, Warranty warranty, int make, int category)
00058         {
00059             _reference = reff;
00060             _price = price;
00061             _makeID = make;
00062             _categoryID = category;
00063             _warranty = warranty;
00064
00065         }
00066
00067         #endregion
00068
00069         #region Properties
00070
00074         public string Reference
00075         {
00076             get { return _reference; }
00077             set { _reference = value; }
00078
00079         }
00080
00084         public decimal Price
00085         {
00086             get { return _price; }
00087             set { _price = value; }
00088         }
00089
00093         public int Make
00094         {
00095             get { return _makeID; }
00096             set { _makeID = value; }
00097
00098         }
00099
```

```
00103         public int Category
00104         {
00105             get { return _categoryID; }
00106             set { _categoryID = value; }
00107         }
00108 
00112         public int Stock
00113         {
00114             get { return _stock; }
00115             set { _stock = value; }
00116         }
00117 
00118         public Warranty Warranty
00119         {
00120             get { return _warranty; }
00121             set { _warranty = value; }
00122         }
00123         #endregion
00124 
00125 
00126 
00127         #region Overrides
00133         public override bool Equals(object obj)
00134         {
00135             if (obj == null) return false;
00136             Product product = obj as Product;
00137             if (product.Reference == _reference) return true;
00138             return false;
00139         }
00140 
00147         public static bool operator ==(Product p1, Product p2)
00148         {
00149             return (p1.Equals(p2));
00150         }
00151 
00158         public static bool operator !=(Product p1, Product p2)
00159         {
00160             return !(p1.Equals(p2));
00161         }
00162 
00167         public override string ToString()
00168         {
00169             StringBuilder sb = new StringBuilder();
00170             sb.AppendLine($"Product ID: {_reference}");
00171             sb.AppendLine($"Price: {_price}€");
00172             sb.AppendLine(_makeID.ToString());
00173             sb.AppendLine(_warranty.ToString());
00174 
00175             return sb.ToString();
00176         }
00177 
00178         #endregion
00179 
00180         #region OtherMethods
00181         #endregion
00182 
00183         #region Destructor
00187         ~Product()
00188         {
00189         }
00190         #endregion
00191 
00192         #endregion
00193     }
00194 }
```

7.42 Products.cs

```
00001 /*
00002 *   <copyright file="Data_BestSale.cs" company="IPCA">
00003 *       Copyright (c) 2024 All Rights Reserved
00004 *   </copyright>
00005 *   <author>Jose Alves a27967</author>
00006 *   <date>11/9/2024 6:34:19 PM</date>
00007 *   <description>Class to manage a group of more than one product.</description>
00008 **/
00009 using System;
00010 using System.Collections;
00011 using System.Collections.Generic;
00012 using System.Linq;
00013 using System.Text;
00014 using static System.Net.Mime.MediaTypeNames;
00015 
```

```
00016 namespace Data_BestSale
00017 {
00018     [Serializable]
00026     public class Products : IListManagement
00027     {
00028         #region Attributes
00029         List<Product> _prods; //como faço uma propriedade para isto? Não consigo dar return do array,
      certo?
00030         #endregion
00031
00032         #region Methods
00033
00034         #region Constructors
00035
00039         public Products()
00040         {
00041             _prods = new List<Product>();
00042         }
00043
00048         public Products(List<Product> products)
00049         {
00050             _prods = products;
00051         }
00052
00053
00054
00055         #endregion
00056
00057         #region Properties
00061         public List<Product> Prods
00062         {
00063             get { return _prods; }
00064             set { _prods = value; }
00065         }
00066         #endregion
00067
00068
00069
00070         #region Overrides
00071
00076         public override string ToString()
00077         {
00078             StringBuilder sb = new StringBuilder();
00079             foreach (var product in _prods)
00080             {
00081                 sb.AppendLine(product.ToString());
00082             }
00083             return sb.ToString();
00084         }
00085         #endregion
00086
00087         #region OtherMethods
00088
00094         public decimal ValueInPosition(int p)
00095         {
00096             return this.Prods[p].Price;
00097         }
00098
00104         public bool Add(object obj)
00105         {
00106             if (obj == null) return false;
00107             var aux=obj as Product;
00108             if (Exist(aux.Reference))
00109             {
00110                 if (obj is Product)
00111                 {
00112                     _prods.Add((Product)obj);
00113                     return true;
00114                 }
00115             }
00116             return false;
00117         }
00118
00124         public bool Exist(object obj)
00125         {
00126             if (obj == null) return false;
00127             if (obj is string)
00128             {
00129                 foreach (Product p in _prods)
00130                 {
00131                     if (p.Reference == (string)obj) return true;
00132                 }
00133             }
00134             return false;
00135         }
00136
00142         public bool Remove(object obj)
```

```
00143         {
00144             if (obj == null) return false;
00145             Product p = (Product)obj;
00146             if (Exist(p.Reference))
00147             {
00148                 _prods.Remove(p);
00149                 return true;
00150             }
00151             return false;
00152         }
00153
00159         public Product SearchProduct(string reff)
00160         {
00161             foreach (Product p in _prods)
00162             {
00163                 if(p.Reference==reff) return p;
00164             }
00165             return null;
00166         }
00167
00172         public decimal TotalPrice()
00173         {
00175             return _prods.Sum(p => p.Price);
00176
00177         }
00178
00185         public DateTime WarratyExpirationDateForProduct(DateTime date, string reff)
00186         {
00187             Product p = SearchProduct(reff);
00188             {
00189                 return (date.AddYears(p.Warranty.DurationInYears));
00190             }
00191         }
00192
00196         public bool ClearProducts()
00197         {
00198             try{
00199                 _prods.Clear();
00200                 return true;
00201             }
00202             catch
00203             {
00204                 return false;
00205             }
00206         }
00207         #endregion
00208
00209         #region Destructor
00213         ~Products()
00214         {
00215         }
00216         #endregion
00217
00218         #endregion
00219     }
00220 }
```

7.43 Products.cs

```
00001 /*
00002 *   <copyright file="trabalhoPOO_27967.cs" company="IPCA">
00003 *       Copyright (c) 2024 All Rights Reserved
00004 *   </copyright>
00005 *   <author>Jose Alves a27967</author>
00006 *   <date>11/9/2024 6:34:19 PM</date>
00007 *   <description></description>
00008 **/
00009 using System;
00010 using System.Collections;
00011 using System.Collections.Generic;
00012 using System.Linq;
00013 using System.Text;
00014 using trabalhoPOO_27967.Interface;
00015 using static System.Net.Mime.MediaTypeNames;
00016
00017 namespace trabalhoPOO_27967
00018 {
00019     [Serializable]
00027     public class Products : IListManagement
00028     {
00029         #region Attributes
00030         List<Product> _prods; //como faço uma propriedade para isto? Não consigo dar return do array,
      certo?
```

```
00031         #endregion
00032
00033         #region Methods
00034
00035         #region Constructors
00036
00040         public Products()
00041         {
00042             _prods = new List<Product>();
00043         }
00044
00049         public Products(List<Product> products)
00050         {
00051             _prods = products;
00052         }
00053
00054
00055
00056         #endregion
00057
00058         #region Properties
00062         public List<Product> Prods
00063         {
00064             get { return _prods; }
00065             set { _prods = value; }
00066         }
00067         #endregion
00068
00069
00070
00071         #region Overrides
00072
00077         public override string ToString()
00078         {
00079             StringBuilder sb = new StringBuilder();
00080             foreach (var product in _prods)
00081             {
00082                 sb.AppendLine(product.ToString());
00083             }
00084             return sb.ToString();
00085         }
00086         #endregion
00087
00088         #region OtherMethods
00089
00095         public decimal ValueInPosition(int p)
00096         {
00097             return this.Prods[p].Price;
00098         }
00099
00105         public bool Add(object obj)
00106         {
00107             if (obj == null) return false;
00108             var aux=obj as Product;
00109             if (Exist(aux.Reference))
00110             {
00111                 if (obj is Product)
00112                 {
00113                     _prods.Add((Product)obj);
00114                     return true;
00115                 }
00116             }
00117             return false;
00118         }
00119
00125         public bool Exist(object obj)
00126         {
00127             if (obj == null) return false;
00128             if (obj is string)
00129             {
00130                 foreach (Product p in _prods)
00131                 {
00132                     if (p.Reference == (string)obj) return true;
00133                 }
00134             }
00135             return false;
00136         }
00137
00143         public bool Remove(object obj)
00144         {
00145             if (obj == null) return false;
00146             Product p = (Product)obj;
00147             if (Exist(p.Reference))
00148             {
00149                 _prods.Remove(p);
00150                 return true;
00151             }
```

```
00152             return false;
00153         }
00154
00160         public Product SearchProduct(string reff)
00161         {
00162             foreach (Product p in _prods)
00163             {
00164                 if(p.Reference==reff) return p;
00165             }
00166             return null;
00167         }
00168
00173         public decimal TotalPrice()
00174         {
00175             return _prods.Sum(p => p.Price);
00176
00177         }
00178
00185         public DateTime WarratyExpirationDateForProduct(DateTime date, string reff)
00186         {
00187             Product p = SearchProduct(reff);
00188             {
00189                 return (date.AddYears(p.Warranty.DurationInYears));
00190             }
00191         }
00192         #endregion
00193
00194         #region Destructor
00198         ~Products()
00199         {
00200         }
00201         #endregion
00202
00203         #endregion
00204     }
00205 }
```

7.44 Sale.cs

```
00001 /*
00002 *   <copyright file="Data_BestSale.cs" company="IPCA">
00003 *       Copyright (c) 2024 All Rights Reserved
00004 *   </copyright>
00005 *   <author>Jose Alves a27967</author>
00006 *   <date>11/6/2024 11:21:53 AM</date>
00007 *   <description>Definition of Sale and methods to deal with Sale operations.</description>
00008 **/
00009 using System;
00010 using System.CodeDom;
00011 using System.Linq;
00012 using System.Text;
00013
00014 namespace Data_BestSale
00015 {
00016     [Serializable]
00025
00026     public class Sale
00027     {
00028         #region Attributes
00029         int _id;
00030         Client _client;
00031         Products _products;
00032         decimal _totPrice;
00033         DateTime _saleDate;
00034         static int _numSales;
00035         Campaign _campaigns;
00036         #endregion
00037
00038
00039         #region Methods
00040
00041         #region Constructors
00042
00046         public Sale()
00047         {
00048             _products= new Products();
00049             _client= new Client();
00050             _campaigns= new Campaign();
00051         }
00052
00059         public Sale(Client client, Products products, Campaign camp)
00060         {
00061             _id = ++_numSales;
```

```
00062             _client = client;
00063             _products = products;
00064             _totPrice = TotalPrice();
00065             _saleDate = DateTime.Now;
00066         }
00067 
00068 
00069 
00070         #endregion
00071 
00072         #region Properties
00076         public int Id
00077         {
00078             get { return _id; }
00079             set { _id = value; }
00080         }
00081 
00085         public Client Client
00086         {
00087             get { return (_client); }
00088             set { _client = value; }
00089         }
00090 
00094         public Products Products
00095         {
00096             get { return _products; }
00097             set { _products = value; }
00098         }
00099 
00103         public decimal TotPrice
00104         {
00105             get { return _totPrice; }
00106             set { _totPrice = value; }
00107         }
00108 
00112         public DateTime SaleDate
00113         {
00114             get { return _saleDate; }
00115             set { _saleDate = value; }
00116         }
00117 
00121         public Campaign Campaigns
00122         {
00123             get { return _campaigns; }
00124             set { _campaigns = value; }
00125         }
00126         #endregion
00127 
00128 
00129 
00130         #region Overrides
00136         public override bool Equals(object obj)
00137         {
00138             Sale sale = obj as Sale;
00139             return (sale.Id==this.Id);
00140         }
00141 
00148         public static bool operator ==(Sale s1, Sale s2)
00149         {
00150             return(s1.Equals(s2));
00151         }
00152 
00159         public static bool operator !=(Sale s1, Sale s2)
00160         {
00161             return !(s1.Equals(s2));
00162         }
00163 
00164         public override string ToString()
00165         {
00166             StringBuilder sb = new StringBuilder();
00167             sb.AppendLine(_products.ToString());
00168             sb.AppendLine("Total " + this.TotalPrice().ToString() + "\u20AC");
00169             return sb.ToString();
00170 
00171         }
00172         #endregion
00173 
00174         #region OtherMethods
00175 
00180         public decimal TotalPrice()
00181         {
00182             decimal total = 0;
00183 
00184             total=_products.TotalPrice();
00185 
00186             if (Campaign.VerifyApplicability(this.Campaigns))
00187             {
```

```
00188                 total *= (1-this.Campaigns.Discount);
00189             }
00190
00191             return total;
00192         }
00193
00199         public bool InsertProductOnSale(Product p)
00200         {
00201             return _products.Add(p);
00202         }
00203
00209         public bool RemoveProductFromSale(Product p)
00210         {
00211             return _products.Remove(p);
00212         }
00213
00219         public bool ExistProductOnSale(Product p)
00220         {
00221             return _products.Exist(p.Reference);
00222         }
00223
00230         public DateTime WarrantyExpirationDate(string reff)
00231         {
00232             return (_products.WarratyExpirationDateForProduct(this.SaleDate, reff));
00233         }
00234         #endregion
00235
00236         #region Destructor
00240         ~Sale()
00241         {
00242         }
00243         #endregion
00244
00245         #endregion
00246     }
00247 }
```

7.45 Sale.cs

```
00001 /*
00002 *   <copyright file="trabalhoPOO_27967.cs" company="IPCA">
00003 *       Copyright (c) 2024 All Rights Reserved
00004 *   </copyright>
00005 *   <author>Jose Alves a27967</author>
00006 *   <date>11/6/2024 11:21:53 AM</date>
00007 *   <description></description>
00008 **/
00009 using System;
00010 using System.CodeDom;
00011 using System.Linq;
00012 using System.Text;
00013
00014 namespace trabalhoPOO_27967
00015 {
00024
00025     public class Sale
00026     {
00027         #region Attributes
00028         int _id;
00029         Client _client;
00030         Products _products;
00031         decimal _totPrice;
00032         DateTime _saleDate;
00033         static int _numSales;
00034         Campaign _campaigns;
00035         #endregion
00036
00037
00038         #region Methods
00039
00040         #region Constructors
00041
00045         public Sale()
00046         {
00047             _products= new Products();
00048             _client= new Client();
00049             _campaigns= new Campaign();
00050         }
00051
00058         public Sale(Client client, Products products, Campaign camp)
00059         {
00060             _id = ++_numSales;
00061             _client = client;
```

```
00062             _products = products;
00063             _totPrice = TotalPrice();
00064             _saleDate = DateTime.Now;
00065         }
00066 
00067 
00068 
00069         #endregion
00070 
00071         #region Properties
00075         public int Id
00076         {
00077             get { return _id; }
00078             set { _id = value; }
00079         }
00080 
00084         public Client Client
00085         {
00086             get { return (_client); }
00087             set { _client = value; }
00088         }
00089 
00093         public Products Products
00094         {
00095             get { return _products; }
00096             set { _products = value; }
00097         }
00098 
00102         public decimal TotPrice
00103         {
00104             get { return _totPrice; }
00105             set { _totPrice = value; }
00106         }
00107 
00111         public DateTime SaleDate
00112         {
00113             get { return _saleDate; }
00114             set { _saleDate = value; }
00115         }
00116 
00120         public Campaign Campaigns
00121         {
00122             get { return _campaigns; }
00123             set { _campaigns = value; }
00124         }
00125         #endregion
00126 
00127 
00128 
00129         #region Overrides
00135         public override bool Equals(object obj)
00136         {
00137             Sale sale = obj as Sale;
00138             return (sale.Id==this.Id);
00139         }
00140 
00147         public static bool operator ==(Sale s1, Sale s2)
00148         {
00149             return(s1.Equals(s2));
00150         }
00151 
00158         public static bool operator !=(Sale s1, Sale s2)
00159         {
00160             return !(s1.Equals(s2));
00161         }
00162 
00163         public override string ToString()
00164         {
00165             StringBuilder sb = new StringBuilder();
00166             sb.AppendLine(_products.ToString());
00167             sb.AppendLine("Total " + this.TotalPrice().ToString() + "\u20AC");
00168             return sb.ToString();
00169 
00170         }
00171         #endregion
00172 
00173         #region OtherMethods
00174 
00179         public decimal TotalPrice()
00180         {
00181             decimal total = 0;
00182 
00183             total=_products.TotalPrice();
00184 
00185             if (Campaign.VerifyApplicability(this.Campaigns))
00186             {
00187                 total *= (1-this.Campaigns.Discount);
```

```
00188             }
00189
00190             return total;
00191         }
00192
00198         public bool InsertProductOnSale(Product p)
00199         {
00200             return _products.Add(p);
00201         }
00202
00208         public bool RemoveProductFromSale(Product p)
00209         {
00210             return _products.Remove(p);
00211         }
00212
00218         public bool ExistProductOnSale(Product p)
00219         {
00220             return _products.Exist(p.Reference);
00221         }
00222
00229         public DateTime WarrantyExpirationDate(string reff)
00230         {
00231             return (_products.WarratyExpirationDateForProduct(this.SaleDate, reff));
00232         }
00233         #endregion
00234
00235         #region Destructor
00239         ~Sale()
00240         {
00241         }
00242         #endregion
00243
00244         #endregion
00245     }
00246 }
```

7.46 Sales.cs

```
00001 /*
00002 *   <copyright file="Data_BestSale.cs" company="IPCA">
00003 *       Copyright (c) 2024 All Rights Reserved
00004 *   </copyright>
00005 *   <author>Jose Alves a27967</author>
00006 *   <date>11/10/2024 7:42:03 PM</date>
00007 *   <description>File with the agregation of sales of a store.</description>
00008 **/
00009 using System;
00010 using System.Collections.Generic;
00011
00012 namespace Data_BestSale
00013 {
00014     [Serializable]
00022     public class Sales : IListManagement
00023     {
00024         #region Attributes
00025         static List<Sale> _salesStored;
00026         #endregion
00027
00028         #region Methods
00029
00030         #region Constructors
00031
00035         public Sales()
00036         {
00037             _salesStored = new List<Sale>();
00038         }
00039
00044         public Sales(List<Sale> sales)
00045         {
00046             _salesStored = sales;
00047         }
00048
00049
00050         #endregion
00051
00052         #region Properties
00056         public List<Sale> SalesStored
00057         {
00058             get { return _salesStored; }
00059             set { _salesStored = value; }
00060         }
00061
00062         #endregion
```

```
00063
00064
00065
00066         #region Overrides
00067         #endregion
00068
00069         #region OtherMethods
00075         public Sale GetSale(int idSale)
00076         {
00077             foreach (Sale s in _salesStored)
00078             {
00079                 if (s.Id == idSale) return s;
00080             }
00081             return null;
00082         }
00083
00089         public bool Add(object obj)
00090         {
00091             if (obj == null) return false;
00092             if (obj is Sale)
00093             {
00094                 _salesStored.Add((Sale)obj);
00095                 return true;
00096             }
00097             return false;
00098         }
00099
00105         public bool Remove(object obj)
00106         {
00107             if (obj == null) return false;
00108             Sale aux = (Sale)obj;
00109             if (Exist(aux.Id))
00110             {
00111                 if (obj is Sale)
00112                 {
00113                     _salesStored.Remove(aux);
00114                     return true;
00115                 }
00116             }
00117             return false;
00118         }
00119
00125         public bool Exist(object obj)
00126         {
00127             if (obj == null) return false;
00128             if(obj is int)
00129             {
00130                 foreach (Sale s in _salesStored)
00131                 {
00132                     if (s.Id == (int)obj)
00133                     {
00134                         return true;
00135                     }
00136                 }
00137             }
00138             return false;
00139         }
00140
00144         public bool ClearSales()
00145         {
00146             try
00147             {
00148                 _salesStored.Clear();
00149                 return true;
00150             }
00151             catch
00152             {
00153                 return false;
00154             }
00155         }
00156         #endregion
00157
00158         #region Destructor
00162         ~Sales()
00163         {
00164         }
00165         #endregion
00166
00167         #endregion
00168     }
00169 }
```

7.47 Sales.cs

```
00001 /*
00002 *   <copyright file="trabalhoPOO_27967.cs" company="IPCA">
00003 *       Copyright (c) 2024 All Rights Reserved
00004 *   </copyright>
00005 *   <author>Jose Alves a27967</author>
00006 *   <date>11/10/2024 7:42:03 PM</date>
00007 *   <description></description>
00008 **/
00009 using System;
00010 using System.Collections.Generic;
00011 using trabalhoPOO_27967.Interface;
00012
00013 namespace trabalhoPOO_27967
00014 {
00022     public class Sales : IListManagement
00023     {
00024         #region Attributes
00025         static List<Sale> _salesStored;
00026         #endregion
00027
00028         #region Methods
00029
00030         #region Constructors
00031
00035         public Sales()
00036         {
00037             _salesStored = new List<Sale>();
00038         }
00039
00044         public Sales(List<Sale> sales)
00045         {
00046             _salesStored = sales;
00047         }
00048
00049
00050         #endregion
00051
00052         #region Properties
00056         public List<Sale> SalesStored
00057         {
00058             get { return _salesStored; }
00059             set { _salesStored = value; }
00060         }
00061
00062         #endregion
00063
00064
00065
00066         #region Overrides
00067         #endregion
00068
00069         #region OtherMethods
00075         public Sale GetSale(int idSale)
00076         {
00077             foreach (Sale s in _salesStored)
00078             {
00079                 if (s.Id == idSale) return s;
00080             }
00081             return null;
00082         }
00083
00089         public bool Add(object obj)
00090         {
00091             if (obj == null) return false;
00092             if (obj is Sale)
00093             {
00094                 _salesStored.Add((Sale)obj);
00095                 return true;
00096             }
00097             return false;
00098         }
00099
00105         public bool Remove(object obj)
00106         {
00107             if (obj == null) return false;
00108             Sale aux = (Sale)obj;
00109             if (Exist(aux.Id))
00110             {
00111                 if (obj is Sale)
00112                 {
00113                     _salesStored.Remove(aux);
00114                     return true;
00115                 }
00116             }
00117             return false;
```

```
00118         }
00119
00125         public bool Exist(object obj)
00126         {
00127             if (obj == null) return false;
00128             if(obj is int)
00129             {
00130                 foreach (Sale s in _salesStored)
00131                 {
00132                     if (s.Id == (int)obj)
00133                     {
00134                         return true;
00135                     }
00136                 }
00137             }
00138             return false;
00139         }
00140         #endregion
00141
00142         #region Destructor
00146         ~Sales()
00147         {
00148         }
00149         #endregion
00150
00151         #endregion
00152     }
00153 }
```

7.48 Store.cs

```
00001 /*
00002 *   <copyright file="Data_BestSale.cs" company="IPCA">
00003 *       Copyright (c) 2024 All Rights Reserved
00004 *   </copyright>
00005 *   <author>Jose Alves a27967</author>
00006 *   <date>11/14/2024 5:01:23 PM</date>
00007 *   <description>This class has the definition and properties to manage a store.</description>
00008 **/
00009 using System;
00010 using System.Collections.Generic;
00011 using System.Diagnostics;
00012 using System.IO;
00013 using System.Runtime.Serialization.Formatters.Binary;
00014 using Business_Object;
00015
00016 namespace Data_BestSale
00017 {
00018     [Serializable]
00026     public class Store
00027     {
00028         #region Attributes
00029         static Clients _clientList;
00030         static Products _prodList;
00031         static Sales _saleList;
00032         static Makes _makeList;
00033         static Categories _catList;
00034         #endregion
00035
00036         #region Methods
00037
00038         #region Constructors
00039
00043         public Store()
00044         {
00045             _clientList = new Clients();
00046             _prodList = new Products();
00047             _saleList = new Sales();
00048             _makeList = new Makes();
00049             _catList = new Categories();
00050         }
00051
00060         public Store(Clients cl, Products p, Sales s, Makes m, Categories c)
00061         {
00062             _clientList = cl;
00063             _prodList = p;
00064             _saleList = s;
00065             _makeList = m;
00066             _catList = c;
00067         }
00068
00069         #endregion
00070
```

```
00071         #region Properties
00075         public Clients ClientLIst
00076         {
00077             get { return _clientList; }
00078             set { _clientList = value; }
00079         }
00080
00084         public Products ProdList
00085         {
00086             get { return _prodList; }
00087             set { _prodList = value; }
00088         }
00089
00093         public Sales SaleList
00094         {
00095             get { return _saleList; }
00096             set { _saleList = value; }
00097         }
00098
00102         public Makes MakeList
00103         {
00104             get { return _makeList; }
00105             set { _makeList = value; }
00106         }
00107
00111         public Categories CatList
00112         {
00113             get { return _catList; }
00114             set { _catList = value; }
00115         }
00116
00117
00118         #endregion
00119
00120
00121
00122         #region Overrides
00123         #endregion
00124
00125         #region OtherMethods
00131         public static string GetMakeNameFromID(int makeID)
00132         {
00133             foreach(Make m in _makeList.MakeList)
00134             {
00135                 if (m.ID == makeID) return m.Name;
00136             }
00137
00138             return ("Not Found");
00139         }
00140
00147         public static bool InsertClientInStore(Client client)
00148         {
00149             return _clientList.Add(client);
00150         }
00151
00152         #region Files
00158         public bool SaveStoreBin(string fileName)
00159         {
00160             try
00161             {
00162                 FileStream stream = File.Open(fileName, FileMode.OpenOrCreate);
00163                 BinaryFormatter bin = new BinaryFormatter();
00164                 bin.Serialize(stream, this);
00165                 stream.Close();
00166                 return true;
00167             }
00168             catch (IOException ioExcep)
00169             {
00170
00171                 throw ioExcep;
00172             }
00173             catch (Exception excep)
00174             {
00175                 throw excep;
00176             }
00177         }
00178
00182         public static bool ClearStore()
00183         {
00184             try
00185             {
00186                 _clientList.ClearClients();
00187                 _catList.ClearCategories();
00188                 _makeList.ClearMakes();
00189                 _prodList.ClearProducts();
00190                 _saleList.ClearSales();
00191                 return true;
```

```
00192             }
00193             catch
00194             {
00195                 return false;
00196             }
00197         }
00198 
00204         public static bool LoadStoreBin(string fileName)
00205         {
00207             if (File.Exists(fileName) && (new FileInfo(fileName).Length > 0))
00208             {
00209                 try
00210                 {
00211                     Store store = new Store();
00212                     Stream stream = File.Open(fileName, FileMode.Open);
00213                     BinaryFormatter bin = new BinaryFormatter();
00214                     store = (Store)bin.Deserialize(stream);
00215                     stream.Close();
00216                     return true;
00217                 }
00218                 catch (IOException ioExcep)
00219                 {
00220 
00221                     throw ioExcep;
00222                 }
00223                 catch (Exception excep)
00224                 {
00225                     throw excep;
00226                 }
00227             }
00228             return false;
00229         }
00230         #endregion
00231 
00232         #region Products
00239         public static bool InsertProductInStore(Product prod)
00240         {
00241             if (_prodList.Exist(prod)) return false;
00242             else
00243             {
00244                 _prodList.Add(prod);
00245                 return true;
00246             }
00247         }
00248 
00254         public static decimal GetProductPriceInStoreFromReference(string reference)
00255         {
00256             Product prod = _prodList.SearchProduct(reference);
00257             return prod.Price;
00258         }
00259 
00260         #endregion
00261 
00262         #region Makes
00269         public static int GetMakeIdFromNameInStore(string name)
00270         {
00271             if (_makeList.Exist(name))
00272             {
00273                 Make aux = _makeList.GetMake(name);
00274                 return (aux.GetMakeID());
00275             }
00276             else return -50;
00277         }
00278 
00284         public static bool InsertMakeInStore(Make make)
00285         {
00286             return _makeList.Add(make);
00287         }
00288         #endregion
00289 
00290         #region Category
00291 
00298         public static int GetCategoryIdFromNameInStore(string name)
00299         {
00300             if(_catList.Exist(name))
00301             {
00302                 Category aux= _catList.GetCategory(name);
00303                 return aux.Id;
00304             }
00305             return -100;
00306         }
00307 
00313         public static bool InsertCategoryInStore(Category cat)
00314         {
00315             return _catList.Add(cat);
00316         }
00317         #endregion
```

```
00318
00319         #endregion
00320
00321         #region Destructor
00325         //~Store()
00326         //{
00327         //}
00328         #endregion
00329
00330         #endregion
00331     }
00332 }
```

7.49 Store.cs

```
00001 /*
00002 *   <copyright file="trabalhoPOO_27967.Store.cs" company="IPCA">
00003 *       Copyright (c) 2024 All Rights Reserved
00004 *   </copyright>
00005 *   <author>Jose Alves a27967</author>
00006 *   <date>11/14/2024 5:01:23 PM</date>
00007 *   <description></description>
00008 **/
00009 using System;
00010
00011 namespace trabalhoPOO_27967.Store
00012 {
00020     public class Store
00021     {
00022         #region Attributes
00023         static Clients _clientList;
00024         static Products _prodList;
00025         static Sales _saleList;
00026         static Makes _makeList;
00027         static Categories _catList;
00028         static Warranties _warrantList;
00029         #endregion
00030
00031         #region Methods
00032
00033         #region Constructors
00034
00038         public Store()
00039         {
00040             _clientList = new Clients();
00041             _prodList = new Products();
00042             _saleList = new Sales();
00043             _makeList = new Makes();
00044             _catList = new Categories();
00045             _warrantList = new Warranties();
00046         }
00047
00057         public Store(Clients cl, Products p, Sales s, Makes m, Categories c, Warranties w)
00058         {
00059             _clientList=cl;
00060             _prodList=p;
00061             _saleList=s;
00062             _makeList=m;
00063             _catList=c;
00064             _warrantList=w;
00065         }
00066
00067         #endregion
00068
00069         #region Properties
00073         public Clients ClientLIst
00074         {
00075             get { return _clientList; }
00076             set { _clientList = value; }
00077         }
00078
00082         public Products ProdList
00083         {
00084             get { return _prodList; }
00085             set { _prodList = value; }
00086         }
00087
00091         public Sales SaleList
00092         {
00093             get { return _saleList; }
00094             set { _saleList = value; }
00095         }
00096
```

```
00100         public Makes MakeList
00101         {
00102             get { return _makeList; }
00103             set { _makeList = value; }
00104         }
00105
00109         public Categories CatList
00110         {
00111             get { return _catList; }
00112             set { _catList = value; }
00113         }
00114
00118         public Warranties WarrantList
00119         {
00120             get { return _warrantList; }
00121             set { _warrantList = value; }
00122         }
00123
00124
00125         #endregion
00126
00127
00128
00129         #region Overrides
00130         #endregion
00131
00132         #region OtherMethods
00138         public static string GetMakeNameFromID(int makeID)
00139         {
00140             foreach(Make m in _makeList.MakeList)
00141             {
00142                 if (m.ID == makeID) return m.Name;
00143             }
00144
00145             return ("Not Found");
00146         }
00147         #endregion
00148
00149         #region Destructor
00153         ~Store()
00154         {
00155         }
00156         #endregion
00157
00158         #endregion
00159     }
00160 }
```

7.50 Warranties.cs

```
00001 /*
00002 *   <copyright file="Data_BestSale.cs" company="IPCA">
00003 *       Copyright (c) 2024 All Rights Reserved
00004 *   </copyright>
00005 *   <author>Jose Alves a27967</author>
00006 *   <date>11/14/2024 4:20:11 PM</date>
00007 *   <description>This file has the definition and methods to work with the plurality of
      Warranty.</description>
00008 **/
00009 using System;
00010 using System.Collections.Generic;
00011
00012
00013 namespace Data_BestSale
00014 {
00015     [Serializable]
00023     public class Warranties: IListManagement
00024     {
00025         #region Attributes
00026         List<Warranty> _warrants;
00027         #endregion
00028
00029         #region Methods
00030
00031         #region Constructors
00032
00036         public Warranties()
00037         {
00038             _warrants = new List<Warranty>();
00039         }
00040
00045         public Warranties(List<Warranty> warrants)
00046         {
```

```
00047             _warrants = warrants;
00048         }
00049
00050
00051
00052         #endregion
00053
00054         #region Properties
00058         public List<Warranty> Warrants
00059         {
00060             get { return _warrants; }
00061             set { _warrants = value; }
00062         }
00063         #endregion
00064
00065
00066
00067         #region Overrides
00068         #endregion
00069
00070         #region OtherMethods
00071         // <summary>
00076         public bool Add(object obj)
00077         {
00078             if (obj == null) return false;
00079             if(obj is Warranty)
00080             {
00081                 _warrants.Add((Warranty)obj);
00082                 return true;
00083             }
00084             return false;
00085         }
00086
00092         public bool Remove(object obj)
00093         {
00094             if (obj == null) return false;
00095             Warranty aux;
00096
00097             //ACRESCENTAR NAS OUTRAS CLASSES DE AGREGACAO!!!!
00098             if (obj is Warranty){
00099                 aux = obj as Warranty;
00100                 if (Exist(aux.ProdID))
00101                 {
00102                     if (obj is Warranty)
00103                     {
00104                         _warrants.Remove((Warranty)obj);
00105                         return true;
00106                     }
00107                 }
00108             }
00109
00110             return false;
00111         }
00112
00119         public bool Exist(object obj)
00120         {
00121             if (obj == null) return false;
00122             if (obj is string)
00123             {
00124                 foreach (Warranty w in _warrants)
00125                 {
00126                     if (w.ProdID == (string)obj)
00127                     {
00128                         return true;
00129                     }
00130                 }
00131             }
00132             return false;
00133         }
00134
00138         public void ClearWarranties()
00139         {
00140             _warrants.Clear();
00141         }
00142         #endregion
00143
00144         #region Destructor
00148         ~Warranties()
00149         {
00150         }
00151         #endregion
00152
00153         #endregion
00154     }
00155 }
```

7.51 Warranties.cs

```
00001 /*
00002 *   <copyright file="trabalhoPOO_27967.Category.cs" company="IPCA">
00003 *       Copyright (c) 2024 All Rights Reserved
00004 *   </copyright>
00005 *   <author>Jose Alves a27967</author>
00006 *   <date>11/14/2024 4:20:11 PM</date>
00007 *   <description></description>
00008 **/
00009 using System;
00010 using System.Collections.Generic;
00011 using trabalhoPOO_27967.Interface;
00012
00013 namespace trabalhoPOO_27967
00014 {
00022     public class Warranties: IListManagement
00023     {
00024         #region Attributes
00025         List<Warranty> _warrants;
00026         #endregion
00027
00028         #region Methods
00029
00030         #region Constructors
00031
00035         public Warranties()
00036         {
00037             _warrants = new List<Warranty>();
00038         }
00039
00044         public Warranties(List<Warranty> warrants)
00045         {
00046             _warrants = warrants;
00047         }
00048
00049
00050
00051         #endregion
00052
00053         #region Properties
00057         public List<Warranty> Warrants
00058         {
00059             get { return _warrants; }
00060             set { _warrants = value; }
00061         }
00062         #endregion
00063
00064
00065
00066         #region Overrides
00067         #endregion
00068
00069         #region OtherMethods
00070         // <summary>
00075         public bool Add(object obj)
00076         {
00077             if (obj == null) return false;
00078             if(obj is Warranty)
00079             {
00080                 _warrants.Add((Warranty)obj);
00081                 return true;
00082             }
00083             return false;
00084         }
00085
00091         public bool Remove(object obj)
00092         {
00093             if (obj == null) return false;
00094             Warranty aux=null;
00095
00096             //ACRESCENTAR NAS OUTRAS CLASSES DE AGREGACAO!!!!
00097             if (obj is Warranty){
00098                 aux = obj as Warranty;
00099             }
00100             if (Exist(aux.ProdID))
00101             {
00102                 if (obj is Warranty)
00103                 {
00104                     _warrants.Remove((Warranty)obj);
00105                     return true;
00106                 }
00107             }
00108             return false;
00109         }
00110
00116         public bool Exist(object obj)
```

```
00117         {
00118             if (obj == null) return false;
00119             if (obj is string)
00120             {
00121                 foreach (Warranty w in _warrants)
00122                 {
00123                     if (w.ProdID == (string)obj)
00124                     {
00125                         return true;
00126                     }
00127                 }
00128             }
00129             return false;
00130         }
00131         #endregion
00132
00133         #region Destructor
00137         ~Warranties()
00138         {
00139         }
00140         #endregion
00141
00142         #endregion
00143     }
00144 }
```

7.52 Warranty.cs

```
00001 /*
00002 *   <copyright file="Data_BestSale.cs" company="IPCA">
00003 *       Copyright (c) 2024 All Rights Reserved
00004 *   </copyright>
00005 *   <author>Jose Alves a27967</author>
00006 *   <date>11/13/2024 4:17:18 PM</date>
00007 *   <description>This class contains the definition and methods to manage warranties.</description>
00008 **/
00009 using System;
00010 using System.Diagnostics.Contracts;
00011 using System.Text;
00012 using System.Xml.Linq;
00013
00014 namespace Data_BestSale
00015 {
00016     [Serializable]
00024     public class Warranty
00025     {
00026         #region Attributes
00027         string _prodID;
00028         int _durationInYears;
00029         string _conditions;
00030         #endregion
00031
00032         #region Methods
00033
00034         #region Constructors
00035
00039         public Warranty()
00040         {
00041             _prodID = string.Empty;
00042             _durationInYears = -1;
00043             _conditions = string.Empty;
00044         }
00045
00052         public Warranty(string prodID, int durationInYears, string conditions)
00053         {
00054             _prodID = prodID;
00055             _durationInYears = durationInYears;
00056             _conditions = conditions;
00057         }
00058
00059
00060
00061         #endregion
00062
00063         #region Properties
00064
00068         public string ProdID
00069         {
00070             get {  return _prodID; }
00071             set { _prodID = value; }
00072         }
00073
00077         public int DurationInYears
```

```
00078         {
00079             get { return _durationInYears; }
00080             set { _durationInYears = value; }
00081         }
00082 
00086         public string Conditions
00087         {
00088             get { return _conditions; }
00089             set { _conditions = value; }
00090         }
00091         #endregion
00092 
00093 
00094 
00095         #region Overrides
00100         public override string ToString()
00101         {
00102             StringBuilder sb = new StringBuilder();
00103             sb.AppendLine($"Warranty Data: ");
00104             sb.AppendLine($"Duration: {_durationInYears} years");
00105             sb.AppendLine($"Terms: {_conditions}");
00106 
00107             return sb.ToString();
00108         }
00109 
00115         public override bool Equals(object obj)
00116         {
00118             if (obj == null)
00119             {
00120                 return false;
00121             }
00122 
00124             Warranty war = obj as Warranty;
00125             return (this.ProdID == war.ProdID && this.Conditions == war.Conditions);
00126         }
00127 
00134         public static bool operator ==(Warranty w1, Warranty w2)
00135         {
00136             return (w1.Equals(w2));
00137         }
00138 
00145         public static bool operator !=(Warranty w1, Warranty w2)
00146         {
00147             return !(w1.Equals(w2));
00148         }
00149         #endregion
00150 
00151         #region OtherMethods
00158         public DateTime ExpirationDate(Sale s, string reff)
00159         {
00160             Product p = s.Products.SearchProduct(reff);
00161             return (s.SaleDate.AddYears(_durationInYears));
00162         }
00163 
00171         public static Warranty CreateWarranty(string reff, int warrantyDuration, string
      warrantyConditions)
00172         {
00173             return new Warranty(reff, warrantyDuration, warrantyConditions);
00174         }
00175 
00176         #endregion
00177 
00178         #region Destructor
00182         ~Warranty()
00183         {
00184         }
00185         #endregion
00186 
00187         #endregion
00188     }
00189 }
```

7.53 Warranty.cs

```
00001 /*
00002 *   <copyright file="trabalhoPOO_27967.cs" company="IPCA">
00003 *       Copyright (c) 2024 All Rights Reserved
00004 *   </copyright>
00005 *   <author>Jose Alves a27967</author>
00006 *   <date>11/13/2024 4:17:18 PM</date>
00007 *   <description></description>
00008 **/
00009 using System;
```

```
00010 using System.Diagnostics.Contracts;
00011 using System.Text;
00012 using System.Xml.Linq;
00013
00014 namespace trabalhoPOO_27967
00015 {
00023     public class Warranty
00024     {
00025         #region Attributes
00026         string _prodID;
00027         int _durationInYears;
00028         string _conditions;
00029         #endregion
00030
00031         #region Methods
00032
00033         #region Constructors
00034
00038         public Warranty()
00039         {
00040             _prodID = string.Empty;
00041             _durationInYears = -1;
00042             _conditions = string.Empty;
00043         }
00044
00051         public Warranty(string prodID, int durationInYears, string conditions)
00052         {
00053             _prodID = prodID;
00054             _durationInYears = durationInYears;
00055             _conditions = conditions;
00056         }
00057
00058
00059
00060         #endregion
00061
00062         #region Properties
00063
00067         public string ProdID
00068         {
00069             get {  return _prodID; }
00070             set { _prodID = value; }
00071         }
00072
00076         public int DurationInYears
00077         {
00078             get { return _durationInYears; }
00079             set { _durationInYears = value; }
00080         }
00081
00085         public string Conditions
00086         {
00087             get { return _conditions; }
00088             set { _conditions = value; }
00089         }
00090         #endregion
00091
00092
00093
00094         #region Overrides
00099         public override string ToString()
00100         {
00101             StringBuilder sb = new StringBuilder();
00102             sb.AppendLine($"Warranty Data: ");
00103             sb.AppendLine($"Duration: {_durationInYears} years");
00104             sb.AppendLine($"Terms: {_conditions}");
00105
00106             return sb.ToString();
00107         }
00108
00114         public override bool Equals(object obj)
00115         {
00117             if (obj == null)
00118             {
00119                 return false;
00120             }
00121
00123             Warranty war = obj as Warranty;
00124             return (this.ProdID == war.ProdID && this.Conditions == war.Conditions);
00125         }
00126
00133         public static bool operator ==(Warranty w1, Warranty w2)
00134         {
00135             return (w1.Equals(w2));
00136         }
00137
00144         public static bool operator !=(Warranty w1, Warranty w2)
```

```
00145         {
00146             return !(w1.Equals(w2));
00147         }
00148         #endregion
00149 
00150         #region OtherMethods
00157         public DateTime ExpirationDate(Sale s, string reff)
00158         {
00159             Product p = s.Products.SearchProduct(reff);
00160             return (s.SaleDate.AddYears(_durationInYears));
00161         }
00162         #endregion
00163 
00164         #region Destructor
00168         ~Warranty()
00169         {
00170         }
00171         #endregion
00172 
00173         #endregion
00174     }
00175 }
```

7.54 InvalidPhoneNumberException.cs

```
00001 /*
00002 *   <copyright file="InvalidPhoneNumberException.cs" company="IPCA">
00003 *       Copyright (c) 2024 All Rights Reserved
00004 *   </copyright>
00005 *   <author>Jose Alves a27967</author>
00006 *   <date>12/17/2024 6:23:56 PM</date>
00007 *   <description>This file contains the exceptions to be handled by the app, when it comes to the
      validation of phone numbers.</description>
00008 */
00009 
00010 using System;
00011 using System.Collections.Generic;
00012 using System.Linq;
00013 using System.Text;
00014 using System.Threading.Tasks;
00015 
00016 namespace Exceptions
00017 {
00021     public class InvalidPhoneNumberException : ApplicationException
00022     {
00023         public InvalidPhoneNumberException() : base("Invalid Phone Number") { }
00024 
00025         public InvalidPhoneNumberException(string message) : base(message) { }
00026 
00027         public InvalidPhoneNumberException(string message, Exception e)
00028         {
00029             throw new InvalidPhoneNumberException(e.Message + "-" + message);
00030         }
00031 
00032     }
00033 }
```

Index

Add
Data_BestSale.Categories, 27
Data_BestSale.Clients, 46
Data_BestSale.IListManagement, 51
Data_BestSale.Makes, 62
Data_BestSale.Products, 77
Data_BestSale.Sales, 98
Data_BestSale.Warranties, 115
trabalhoPOO_27967.Campaigns, 25
trabalhoPOO_27967.Categories, 30
trabalhoPOO_27967.Clients, 49
trabalhoPOO_27967.Interface.IListManagement, 52
trabalhoPOO_27967.Makes, 65
trabalhoPOO_27967.Products, 83
trabalhoPOO_27967.Sales, 101
trabalhoPOO_27967.Warranties, 117

BestSale, 11
BestSale.BestSale, 15
BestSale/BestSale.cs, 127
BestSale/obj/Debug/.NETFramework,Version=v4.7.2.AssemblyAttributes.cs, 128
BestSale/Properties/AssemblyInfo.cs, 129
BestSale_Validations, 11
BestSale_Validations/BestSale_Validations.cs, 132
BestSale_Validations/obj/Debug/.NETFramework,Version=v4.7.2.AssemblyAttributes.cs, 128
BestSale_Validations/Properties/AssemblyInfo.cs, 130
Business_Layer, 11
Business_Layer/ClientManagement.cs, 133
Business_Layer/FileManagement.cs, 133
Business_Layer/obj/Debug/.NETFramework,Version=v4.7.2.AssemblyAttributes.cs, 128
Business_Layer/ProductManagement.cs, 134
Business_Layer/Properties/AssemblyInfo.cs, 130
Business_Object, 11
Business_Object.SimpleProduct, 102
Make, 104
Price, 104
Reference, 104
SimpleProduct, 103
Business_Object/obj/Debug/.NETFramework,Version=v4.7.2.AssemblyAttributes.cs, 129
Business_Object/Properties/AssemblyInfo.cs, 130
Business_Object/SimpleProduct.cs, 135

Campaign
Data_BestSale.Campaign, 16
trabalhoPOO_27967.Campaign, 20
CampaignCount
Data_BestSale.Campaign, 18
trabalhoPOO_27967.Campaign, 22
Campaigns
Data_BestSale.Sale, 90
trabalhoPOO_27967.Campaigns, 24
trabalhoPOO_27967.Sale, 96
Camps
trabalhoPOO_27967.Campaigns, 26
Categories
Data_BestSale.Categories, 27
trabalhoPOO_27967.Categories, 30
Category
Data_BestSale.Category, 33
trabalhoPOO_27967.Category, 37
trabalhoPOO_27967.Product, 75
CategoryID
Data_BestSale.Product, 71
CatList
Data_BestSale.Store, 110
trabalhoPOO_27967.Store.Store, 112
Cats
Data_BestSale.Categories, 29
trabalhoPOO_27967.Categories, 32
ClearCategories
Data_BestSale.Categories, 28
ClearClients
Data_BestSale.Clients, 47
ClearMakes
Data_BestSale.Makes, 62
ClearProducts
Data_BestSale.Products, 78
ClearSales
Data_BestSale.Sales, 98
ClearStore
Data_BestSale.Store, 106
ClearWarranties
Data_BestSale.Warranties, 115
Client
Data_BestSale.Client, 39
Data_BestSale.Sale, 90
trabalhoPOO_27967.Client, 43
trabalhoPOO_27967.Sale, 96
ClientCount
Data_BestSale.Client, 41
trabalhoPOO_27967.Client, 45
ClientID
Data_BestSale.Client, 41
trabalhoPOO_27967.Client, 45

ClientLIst
Data_BestSale.Store, 110
trabalhoPOO_27967.Store.Store, 112
ClientList
Data_BestSale.Clients, 48
trabalhoPOO_27967.Clients, 51
Clients
Data_BestSale.Clients, 46
trabalhoPOO_27967.Clients, 49
Conditions
Data_BestSale.Warranty, 122
trabalhoPOO_27967.Warranty, 126
Contact
Data_BestSale.Client, 42
trabalhoPOO_27967.Client, 45
CreateCategory
Data_BestSale.Category, 33
CreateClientFromNameContact
Data_BestSale.Client, 40
CreateMake
Data_BestSale.Make, 55
CreateProductWithWarranty
Data_BestSale.Product, 69
CreateWarranty
Data_BestSale.Warranty, 120

Data_BestSale, 12
Data_BestSale.Campaign, 15
Campaign, 16
CampaignCount, 18
Discount, 18
EndDate, 18
Equals, 17
Id, 19
Name, 19
operator!=, 17
operator==, 17
StartDate, 19
VerifyApplicability, 18
Data_BestSale.Campaigns, 23
Data_BestSale.Categories, 26
Add, 27
Categories, 27
Cats, 29
ClearCategories, 28
Exist, 28
GetCategory, 28
Remove, 29
Data_BestSale.Category, 32
Category, 33
CreateCategory, 33
Equals, 33
Id, 36
Name, 36
operator!=, 35
operator==, 35
Data_BestSale.Client, 39
Client, 39
ClientCount, 41
ClientID, 41
Contact, 42
CreateClientFromNameContact, 40
Equals, 40
Name, 42
operator!=, 40
operator==, 41
ToString, 41
Data_BestSale.Clients, 46
Add, 46
ClearClients, 47
ClientList, 48
Clients, 46
Exist, 47
GetClient, 47
Remove, 48
Data_BestSale.IListManagement, 51
Add, 51
Exist, 51
Remove, 52
Data_BestSale.Make, 54
CreateMake, 55
Equals, 56
GetMakeID, 56
ID, 57
Make, 55
Name, 57
operator!=, 56
operator==, 57
ToString, 57
Data_BestSale.Makes, 61
Add, 62
ClearMakes, 62
Exist, 63
GetMake, 63
MakeList, 64
Makes, 62
Remove, 63
Data_BestSale.Product, 67
CategoryID, 71
CreateProductWithWarranty, 69
Equals, 69
MakeID, 71
operator!=, 70
operator==, 70
Price, 71
Product, 68
Reference, 71
Stock, 71
ToString, 70
Warranty, 72
Data_BestSale.Products, 76
Add, 77
ClearProducts, 78
Exist, 78
Prods, 81
Products, 77
Remove, 78

SearchProduct, 79
ToString, 79
TotalPrice, 79
ValueInPosition, 80
WarratyExpirationDateForProduct, 81
Data_BestSale.Sale, 85
Campaigns, 90
Client, 90
Equals, 87
ExistProductOnSale, 87
Id, 90
InsertProductOnSale, 88
operator!=, 88
operator==, 88
Products, 90
RemoveProductFromSale, 89
Sale, 87
SaleDate, 90
ToString, 89
TotalPrice, 89
TotPrice, 91
WarrantyExpirationDate, 89
Data_BestSale.Sales, 97
Add, 98
ClearSales, 98
Exist, 98
GetSale, 98
Remove, 99
Sales, 97
SalesStored, 99
Data_BestSale.Store, 104
CatList, 110
ClearStore, 106
ClientLIst, 110
GetCategoryIdFromNameInStore, 106
GetMakeIdFromNameInStore, 106
GetMakeNameFromID, 107
GetProductPriceInStoreFromReference, 107
InsertCategoryInStore, 107
InsertClientInStore, 108
InsertMakeInStore, 108
InsertProductInStore, 108
LoadStoreBin, 109
MakeList, 110
ProdList, 110
SaleList, 110
SaveStoreBin, 109
Store, 105
Data_BestSale.Warranties, 114
Add, 115
ClearWarranties, 115
Exist, 115
Remove, 115
Warranties, 114
Warrants, 116
Data_BestSale.Warranty, 119
Conditions, 122
CreateWarranty, 120
DurationInYears, 122
Equals, 120
ExpirationDate, 121
operator!=, 121
operator==, 122
ProdID, 123
ToString, 122
Warranty, 120
Data_BestSale/Campaign/Campaign.cs, 136
Data_BestSale/Campaign/Campaigns.cs, 140
Data_BestSale/Category/Categories.cs, 142
Data_BestSale/Category/Category.cs, 145
Data_BestSale/Client/Client.cs, 148
Data_BestSale/Client/Clients.cs, 151
Data_BestSale/Interface/IListManagement.cs, 154
Data_BestSale/Make/Make.cs, 155
Data_BestSale/Make/Makes.cs, 158
Data_BestSale/obj/Debug/.NETFramework,Version=v4.7.2.AssemblyAttrib
129
Data_BestSale/Product/Product.cs, 161
Data_BestSale/Product/Products.cs, 164
Data_BestSale/Properties/AssemblyInfo.cs, 131
Data_BestSale/Sale/Sale.cs, 168
Data_BestSale/Sale/Sales.cs, 172
Data_BestSale/Store/Store.cs, 175
Data_BestSale/Warranty/Warranties.cs, 179
Data_BestSale/Warranty/Warranty.cs, 182
Discount
Data_BestSale.Campaign, 18
trabalhoPOO_27967.Campaign, 22
DurationInYears
Data_BestSale.Warranty, 122
trabalhoPOO_27967.Warranty, 126
EndDate
Data_BestSale.Campaign, 18
trabalhoPOO_27967.Campaign, 22
Equals
Data_BestSale.Campaign, 17
Data_BestSale.Category, 33
Data_BestSale.Client, 40
Data_BestSale.Make, 56
Data_BestSale.Product, 69
Data_BestSale.Sale, 87
Data_BestSale.Warranty, 120
trabalhoPOO_27967.Campaign, 21
trabalhoPOO_27967.Category, 37
trabalhoPOO_27967.Client, 43
trabalhoPOO_27967.Make, 59
trabalhoPOO_27967.Product, 73
trabalhoPOO_27967.Sale, 93
trabalhoPOO_27967.Warranty, 124
Exceptions, 13
Exceptions.InvalidPhoneNumberException, 53
InvalidPhoneNumberException, 53, 54
Exceptions/InvalidPhoneNumberException.cs, 185
Exceptions/obj/Debug/.NETFramework,Version=v4.7.2.AssemblyAttributes
129
Exceptions/Properties/AssemblyInfo.cs, 131

Exist
Data_BestSale.Categories, 28
Data_BestSale.Clients, 47
Data_BestSale.IListManagement, 51
Data_BestSale.Makes, 63
Data_BestSale.Products, 78
Data_BestSale.Sales, 98
Data_BestSale.Warranties, 115
trabalhoPOO_27967.Campaigns, 25
trabalhoPOO_27967.Categories, 31
trabalhoPOO_27967.Clients, 50
trabalhoPOO_27967.Interface.IListManagement, 52
trabalhoPOO_27967.Makes, 66
trabalhoPOO_27967.Products, 83
trabalhoPOO_27967.Sales, 101
trabalhoPOO_27967.Warranties, 118
ExistProductOnSale
Data_BestSale.Sale, 87
trabalhoPOO_27967.Sale, 93
ExpirationDate
Data_BestSale.Warranty, 121
trabalhoPOO_27967.Warranty, 125

GetCategory
Data_BestSale.Categories, 28
GetCategoryIdFromNameInStore
Data_BestSale.Store, 106
GetClient
Data_BestSale.Clients, 47
trabalhoPOO_27967.Clients, 50
GetMake
Data_BestSale.Makes, 63
GetMakeID
Data_BestSale.Make, 56
GetMakeIdFromNameInStore
Data_BestSale.Store, 106
GetMakeNameFromID
Data_BestSale.Store, 107
trabalhoPOO_27967.Store.Store, 112
GetProductPriceInStoreFromReference
Data_BestSale.Store, 107
GetSale
Data_BestSale.Sales, 98
trabalhoPOO_27967.Sales, 101

ID
Data_BestSale.Make, 57
trabalhoPOO_27967.Make, 60
Id
Data_BestSale.Campaign, 19
Data_BestSale.Category, 36
Data_BestSale.Sale, 90
trabalhoPOO_27967.Campaign, 23
trabalhoPOO_27967.Category, 38
trabalhoPOO_27967.Sale, 96
InsertCategoryInStore
Data_BestSale.Store, 107
InsertClientInStore
Data_BestSale.Store, 108
InsertMakeInStore
Data_BestSale.Store, 108
InsertProductInStore
Data_BestSale.Store, 108
InsertProductOnSale
Data_BestSale.Sale, 88
trabalhoPOO_27967.Sale, 93
InvalidPhoneNumberException
Exceptions.InvalidPhoneNumberException, 53, 54

LoadStoreBin
Data_BestSale.Store, 109

Make
Business_Object.SimpleProduct, 104
Data_BestSale.Make, 55
trabalhoPOO_27967.Make, 59
trabalhoPOO_27967.Product, 75
MakeID
Data_BestSale.Product, 71
MakeList
Data_BestSale.Makes, 64
Data_BestSale.Store, 110
trabalhoPOO_27967.Makes, 67
trabalhoPOO_27967.Store.Store, 113
Makes
Data_BestSale.Makes, 62
trabalhoPOO_27967.Makes, 65

Name
Data_BestSale.Campaign, 19
Data_BestSale.Category, 36
Data_BestSale.Client, 42
Data_BestSale.Make, 57
trabalhoPOO_27967.Campaign, 23
trabalhoPOO_27967.Category, 38
trabalhoPOO_27967.Client, 45
trabalhoPOO_27967.Make, 60

operator!=
Data_BestSale.Campaign, 17
Data_BestSale.Category, 35
Data_BestSale.Client, 40
Data_BestSale.Make, 56
Data_BestSale.Product, 70
Data_BestSale.Sale, 88
Data_BestSale.Warranty, 121
trabalhoPOO_27967.Campaign, 21
trabalhoPOO_27967.Category, 37
trabalhoPOO_27967.Client, 44
trabalhoPOO_27967.Make, 59
trabalhoPOO_27967.Product, 74
trabalhoPOO_27967.Sale, 94
trabalhoPOO_27967.Warranty, 125
operator==
Data_BestSale.Campaign, 17
Data_BestSale.Category, 35
Data_BestSale.Client, 41

Data_BestSale.Make, 57
Data_BestSale.Product, 70
Data_BestSale.Sale, 88
Data_BestSale.Warranty, 122
trabalhoPOO_27967.Campaign, 21
trabalhoPOO_27967.Category, 38
trabalhoPOO_27967.Client, 44
trabalhoPOO_27967.Make, 60
trabalhoPOO_27967.Product, 74
trabalhoPOO_27967.Sale, 94
trabalhoPOO_27967.Warranty, 125

Price
Business_Object.SimpleProduct, 104
Data_BestSale.Product, 71
trabalhoPOO_27967.Product, 75
ProdID
Data_BestSale.Warranty, 123
trabalhoPOO_27967.Warranty, 126
ProdList
Data_BestSale.Store, 110
trabalhoPOO_27967.Store.Store, 113
Prods
Data_BestSale.Products, 81
trabalhoPOO_27967.Products, 85
Product
Data_BestSale.Product, 68
trabalhoPOO_27967.Product, 73
Products
Data_BestSale.Products, 77
Data_BestSale.Sale, 90
trabalhoPOO_27967.Products, 82
trabalhoPOO_27967.Sale, 96

Reference
Business_Object.SimpleProduct, 104
Data_BestSale.Product, 71
trabalhoPOO_27967.Product, 75
Remove
Data_BestSale.Categories, 29
Data_BestSale.Clients, 48
Data_BestSale.IListManagement, 52
Data_BestSale.Makes, 63
Data_BestSale.Products, 78
Data_BestSale.Sales, 99
Data_BestSale.Warranties, 115
trabalhoPOO_27967.Campaigns, 25
trabalhoPOO_27967.Categories, 31
trabalhoPOO_27967.Clients, 50
trabalhoPOO_27967.Interface.IListManagement, 53
trabalhoPOO_27967.Makes, 66
trabalhoPOO_27967.Products, 83
trabalhoPOO_27967.Sales, 102
trabalhoPOO_27967.Warranties, 118
RemoveProductFromSale
Data_BestSale.Sale, 89
trabalhoPOO_27967.Sale, 94

Sale
Data_BestSale.Sale, 87
trabalhoPOO_27967.Sale, 92
SaleDate
Data_BestSale.Sale, 90
trabalhoPOO_27967.Sale, 96
SaleList
Data_BestSale.Store, 110
trabalhoPOO_27967.Store.Store, 113
Sales
Data_BestSale.Sales, 97
trabalhoPOO_27967.Sales, 100
SalesStored
Data_BestSale.Sales, 99
trabalhoPOO_27967.Sales, 102
SaveStoreBin
Data_BestSale.Store, 109
SearchProduct
Data_BestSale.Products, 79
trabalhoPOO_27967.Products, 84
SimpleProduct
Business_Object.SimpleProduct, 103
StartDate
Data_BestSale.Campaign, 19
trabalhoPOO_27967.Campaign, 23
Stock
Data_BestSale.Product, 71
trabalhoPOO_27967.Product, 75
Store
Data_BestSale.Store, 105
trabalhoPOO_27967.Store.Store, 111

ToString
Data_BestSale.Client, 41
Data_BestSale.Make, 57
Data_BestSale.Product, 70
Data_BestSale.Products, 79
Data_BestSale.Sale, 89
Data_BestSale.Warranty, 122
trabalhoPOO_27967.Client, 44
trabalhoPOO_27967.Make, 60
trabalhoPOO_27967.Product, 74
trabalhoPOO_27967.Products, 84
trabalhoPOO_27967.Sale, 95
trabalhoPOO_27967.Warranty, 126
TotalPrice
Data_BestSale.Products, 79
Data_BestSale.Sale, 89
trabalhoPOO_27967.Products, 84
trabalhoPOO_27967.Sale, 95
TotPrice
Data_BestSale.Sale, 91
trabalhoPOO_27967.Sale, 96
trabalhoPOO_27967, 13
trabalhoPOO_27967.Campaign, 19
Campaign, 20
CampaignCount, 22
Discount, 22
EndDate, 22

Equals, 21
Id, 23
Name, 23
operator!=, 21
operator==, 21
StartDate, 23
VerifyApplicability, 22
trabalhoPOO_27967.Campaigns, 24
Add, 25
Campaigns, 24
Camps, 26
Exist, 25
Remove, 25
trabalhoPOO_27967.Categories, 29
Add, 30
Categories, 30
Cats, 32
Exist, 31
Remove, 31
trabalhoPOO_27967.Category, 36
Category, 37
Equals, 37
Id, 38
Name, 38
operator!=, 37
operator==, 38
trabalhoPOO_27967.Client, 42
Client, 43
ClientCount, 45
ClientID, 45
Contact, 45
Equals, 43
Name, 45
operator!=, 44
operator==, 44
ToString, 44
trabalhoPOO_27967.Clients, 48
Add, 49
ClientList, 51
Clients, 49
Exist, 50
GetClient, 50
Remove, 50
trabalhoPOO_27967.Interface, 14
trabalhoPOO_27967.Interface.IListManagement, 52
Add, 52
Exist, 52
Remove, 53
trabalhoPOO_27967.Make, 58
Equals, 59
ID, 60
Make, 59
Name, 60
operator!=, 59
operator==, 60
ToString, 60
trabalhoPOO_27967.Makes, 64
Add, 65
Exist, 66
MakeList, 67
Makes, 65
Remove, 66
trabalhoPOO_27967.Product, 72
Category, 75
Equals, 73
Make, 75
operator!=, 74
operator==, 74
Price, 75
Product, 73
Reference, 75
Stock, 75
ToString, 74
Warranty, 76
trabalhoPOO_27967.Products, 81
Add, 83
Exist, 83
Prods, 85
Products, 82
Remove, 83
SearchProduct, 84
ToString, 84
TotalPrice, 84
ValueInPosition, 84
WarratyExpirationDateForProduct, 85
trabalhoPOO_27967.Sale, 91
Campaigns, 96
Client, 96
Equals, 93
ExistProductOnSale, 93
Id, 96
InsertProductOnSale, 93
operator!=, 94
operator==, 94
Products, 96
RemoveProductFromSale, 94
Sale, 92
SaleDate, 96
ToString, 95
TotalPrice, 95
TotPrice, 96
WarrantyExpirationDate, 95
trabalhoPOO_27967.Sales, 99
Add, 101
Exist, 101
GetSale, 101
Remove, 102
Sales, 100
SalesStored, 102
trabalhoPOO_27967.Store, 14
trabalhoPOO_27967.Store.Store, 111
CatList, 112
ClientLIst, 112
GetMakeNameFromID, 112
MakeList, 113
ProdList, 113

SaleList, 113
Store, 111
WarrantList, 113
trabalhoPOO_27967.Warranties, 116
Add, 117
Exist, 118
Remove, 118
Warranties, 117
Warrants, 119
trabalhoPOO_27967.Warranty, 123
Conditions, 126
DurationInYears, 126
Equals, 124
ExpirationDate, 125
operator!=, 125
operator==, 125
ProdID, 126
ToString, 126
Warranty, 124
trabalhoPOO_27967/BestSale.cs, 128
trabalhoPOO_27967/Campaign/Campaign.cs, 138
trabalhoPOO_27967/Campaign/Campaigns.cs, 141
trabalhoPOO_27967/Category/Categories.cs, 144
trabalhoPOO_27967/Category/Category.cs, 147
trabalhoPOO_27967/Client/Client.cs, 150
trabalhoPOO_27967/Client/Clients.cs, 153
trabalhoPOO_27967/Interface/IListManagement.cs, 154
trabalhoPOO_27967/Make/Make.cs, 156
trabalhoPOO_27967/Make/Makes.cs, 159
trabalhoPOO_27967/obj/Debug/.NETFramework,Version=v4.7.2.AssemblyAttributes.cs, 129
trabalhoPOO_27967/Product/Product.cs, 163
trabalhoPOO_27967/Product/Products.cs, 166
trabalhoPOO_27967/Properties/AssemblyInfo.cs, 132
trabalhoPOO_27967/Sale/Sale.cs, 170
trabalhoPOO_27967/Sale/Sales.cs, 174
trabalhoPOO_27967/Store/Store.cs, 178
trabalhoPOO_27967/Warranty/Warranties.cs, 181
trabalhoPOO_27967/Warranty/Warranty.cs, 183

ValueInPosition
Data_BestSale.Products, 80
trabalhoPOO_27967.Products, 84
VerifyApplicability
Data_BestSale.Campaign, 18
trabalhoPOO_27967.Campaign, 22

Warranties
Data_BestSale.Warranties, 114
trabalhoPOO_27967.Warranties, 117
WarrantList
trabalhoPOO_27967.Store.Store, 113
Warrants
Data_BestSale.Warranties, 116
trabalhoPOO_27967.Warranties, 119
Warranty
Data_BestSale.Product, 72
Data_BestSale.Warranty, 120
trabalhoPOO_27967.Product, 76
trabalhoPOO_27967.Warranty, 124
WarrantyExpirationDate
Data_BestSale.Sale, 89
trabalhoPOO_27967.Sale, 95
WarratyExpirationDateForProduct
Data_BestSale.Products, 81
trabalhoPOO_27967.Products, 85