Home work 24-11-2023

Lập Trình c# 3

Lê Hữu Hoàn

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

CODE:

|  |
| --- |
| using System;  using System.Collections.Generic;  namespace AutomobileLibrary.DataAccess  {  public partial class Car  {  public int CarId { get; set; }  public string CarName { get; set; } = null!;  public string Manufacturer { get; set; } = null!;  public decimal Price { get; set; }  public int ReleasedYear { get; set; }  }  } |
| using Microsoft.EntityFrameworkCore;  using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace AutomobileLibrary.DataAccess  {  public class CarManagement  {  private static CarManagement instance = null;  private static readonly object instanceLock = new object();  private CarManagement() { }  public static CarManagement Instance  {  get  {  lock (instanceLock)  {  if (instance == null)  {  instance = new CarManagement();  }  return instance;  }  }  }  public IEnumerable<Car> GetCarList()  {  List<Car> cars;  try  {  var myStockDB = new MyStockContext();  cars = myStockDB.Cars.ToList();  }  catch (Exception ex)  {  throw new Exception(ex.Message);  }  return cars;  }  public Car GetCarByID(int carID)  {  Car car = null;  try  {  var myStockDB = new MyStockContext();  car = myStockDB.Cars.SingleOrDefault(car => car.CarId == carID);  }  catch (Exception ex)  {  throw new Exception(ex.Message);  }  return car;  }  public void AddNew(Car car)  {  try  {  Car \_car = GetCarByID(car.CarId);  if (\_car == null)  {  var myStockDB = new MyStockContext();  myStockDB.Cars.Add(car);  myStockDB.SaveChanges();  }  else  {  throw new Exception("The car is already exist.");  }  }  catch (Exception ex)  {  throw new Exception(ex.Message);  }  }  public void Update(Car car)  {  try  {  Car c = GetCarByID(car.CarId);  if (c != null)  {  var myStockDB = new MyStockContext();  myStockDB.Entry<Car>(car).State = EntityState.Modified;  myStockDB.SaveChanges();  }  else  {  throw new Exception("The car does not already exist.");  }  }  catch (Exception ex)  {  throw new Exception(ex.Message);  }  }  public void Remove(Car car)  {  try  {  Car \_car = GetCarByID(car.CarId);  if (\_car != null)  {  var myStockDB = new MyStockContext();  myStockDB.Cars.Remove(car);  myStockDB.SaveChanges();  }  else  {  throw new Exception("The car does not already exist.");  }  }  catch (Exception ex)  {  throw new Exception(ex.Message);  }  }  }  } |
| using System;  using System.Collections.Generic;  using Microsoft.EntityFrameworkCore;  using Microsoft.EntityFrameworkCore.Metadata;  namespace AutomobileLibrary.DataAccess  {  public partial class MyStockContext : DbContext  {  public MyStockContext()  {  }  public MyStockContext(DbContextOptions<MyStockContext> options)  : base(options)  {  }  public virtual DbSet<Car> Cars { get; set; } = null!;  protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)  {  if (!optionsBuilder.IsConfigured)  {  optionsBuilder.UseSqlServer("Server=localhost;Database=MyStock;User ID=sa;Password=123456;Trusted\_Connection=True;TrustServerCertificate=True;");  }  }  protected override void OnModelCreating(ModelBuilder modelBuilder)  {  modelBuilder.Entity<Car>(entity =>  {  entity.Property(e => e.CarId)  .ValueGeneratedNever()  .HasColumnName("CarID");  entity.Property(e => e.CarName)  .HasMaxLength(50)  .IsUnicode(false);  entity.Property(e => e.Manufacturer)  .HasMaxLength(50)  .IsUnicode(false);  entity.Property(e => e.Price).HasColumnType("money");  });  OnModelCreatingPartial(modelBuilder);  }  partial void OnModelCreatingPartial(ModelBuilder modelBuilder);  }  } |
| using AutomobileLibrary.DataAccess;  using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace AutomobileLibrary.Repository  {  public class CarRepository : ICarRepository  {  public Car GetCarByID(int carId) => CarManagement.Instance.GetCarByID(carId);  public IEnumerable<Car> GetCars() => CarManagement.Instance.GetCarList();  public void InsertCar(Car car) => CarManagement.Instance.AddNew(car);  public void DeleteCar(Car car) => CarManagement.Instance.Remove(car);  public void UpdateCar(Car car) => CarManagement.Instance.Update(car);  }  } |
| using AutomobileLibrary.DataAccess;  using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace AutomobileLibrary.Repository  {  public interface ICarRepository  {  IEnumerable<Car> GetCars();  Car GetCarByID(int carId);  void InsertCar(Car car);  void DeleteCar(Car car);  void UpdateCar(Car car);  }  } |
| using AutomobileLibrary.Repository;  using Microsoft.Extensions.DependencyInjection;  using System;  using System.Collections.Generic;  using System.Configuration;  using System.Data;  using System.Linq;  using System.Threading.Tasks;  using System.Windows;  namespace AutomobileWPFApp  {  /// <summary>  /// Interaction logic for App.xaml  /// </summary>  public partial class App : Application  {  private ServiceProvider serviceProvider;  public App()  {  ServiceCollection services = new ServiceCollection();  ConfigureServices(services);  serviceProvider = services.BuildServiceProvider();  }  private void ConfigureServices(ServiceCollection services)  {  services.AddSingleton(typeof(ICarRepository), typeof(CarRepository));  services.AddSingleton<WindowCarManagement>();  }  private void OnStartup(object sender, StartupEventArgs e)  {  var windowCarManagement = serviceProvider.GetService<WindowCarManagement>();  windowCarManagement.Show();  }  }  } |
| <Window x:Class="AutomobileWPFApp.WindowCarManagement"  xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"  xmlns:d="http://schemas.microsoft.com/expression/blend/2008"  xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"  xmlns:local="clr-namespace:AutomobileWPFApp"  mc:Ignorable="d"  Title="Car Management" Width="800"  WindowStartupLocation="CenterScreen" ResizeMode="NoResize">  <Grid>  <DockPanel VerticalAlignment="Top" Margin="10">  <Grid>  <Grid.RowDefinitions>  <RowDefinition Height="Auto"/>  <RowDefinition Height="\*"/>  <RowDefinition Height="4\*"/>  <RowDefinition Height="\*"/>  </Grid.RowDefinitions>  <!--StackPanel for Label and TextBox controls-->  <StackPanel Background="LightBlue" Orientation ="Vertical"  HorizontalAlignment="Left" Width="400">  <Label Name="lbTitle" Foreground="Red" FontWeight="DemiBold"  FontSize="20" Content="Car Information" />  <Label Name="lbCarId" Content="Car Id"/>  <TextBox Name="txtCarId" HorizontalAlignment="Stretch"  Height="25"  Text="{Binding Path=CarId, Mode=OneWay}"  DataContext="{Binding ElementName=lvCars,  Path=SelectedItem}" />  <Label Name="lbCarName" Content="Car Name" />  <TextBox Name="txtCarName" HorizontalAlignment="Stretch"  Height="25"  Text="{Binding Path=CarName, Mode=OneWay}"  DataContext="{Binding ElementName=lvCars,  Path=SelectedItem}" />  <Label Name="lbManufacturer" Content="Manufacturer" />  <TextBox Name="txtManufacturer" HorizontalAlignment="Stretch"  Height="25"  Text="{Binding Path=Manufacturer, Mode=OneWay}"  DataContext="{Binding ElementName=lvCars,  Path=SelectedItem}" />  <Label Name="lbPrice" Content="Price" />  <TextBox Name="txtPrice" HorizontalAlignment="Stretch"  Height="25"  Text="{Binding  Path=Price,StringFormat={}{0:N3}, Mode=OneWay}"  DataContext="{Binding ElementName=lvCars,  Path=SelectedItem}" />  <Label Name="lbReleasedYear" Content="ReleasedYear" />  <TextBox Name="txtReleasedYear" HorizontalAlignment="Stretch"  Height="25"  Text="{Binding Path=ReleasedYear, Mode=OneWay}"  DataContext="{Binding ElementName=lvCars,  Path=SelectedItem}" />  </StackPanel>  <!--StackPanel for Button controls-->  <StackPanel Grid.Row="1" Orientation="Horizontal"  HorizontalAlignment="Left">  <Button x:Name="btnLoad" Margin="10" Width="80" Content="Load"  Click="btnLoad\_Click"/>  <Button x:Name="btnInsert" Margin="10" Width="80" Content="Insert"  Click="btnInsert\_Click"/>  <Button x:Name="btnUpdate" Margin="10" Width="80" Content="Update"  Click="btnUpdate\_Click"/>  <Button x:Name="btnDelete" Margin="10" Width="80" Content="Delete"  Click="btnDelete\_Click"/>  </StackPanel>  <!--ListView control-->  <ListView Grid.Row="2" Name="lvCars" Width="Auto" Height="Auto" >  <ListView.View>  <GridView>  <GridViewColumn Header="Car ID" Width="100"  DisplayMemberBinding="{Binding Path=CarId }"/>  <GridViewColumn Header="Car Name" Width="200"  DisplayMemberBinding="{Binding Path=CarName}"/>  <GridViewColumn Header="Manufacturer" Width="200"  DisplayMemberBinding="{Binding Path=Manufacturer }"/>  <GridViewColumn Header="Price" Width="100"  DisplayMemberBinding="{Binding Path=Price,  StringFormat={}{0:N3}}"/>  <GridViewColumn Header="ReleasedYear" Width="100"  DisplayMemberBinding="{Binding Path=ReleasedYear}"/>  </GridView>  </ListView.View>  </ListView>  <!--Button control-->  <Button Grid.Row="3" x:Name="btnClose" Margin="10"  HorizontalAlignment="Right" VerticalAlignment="Bottom"  Width="80" Content="Close" Click="btnClose\_Click" />  </Grid>  </DockPanel>  </Grid>  </Window> |
| using AutomobileLibrary.DataAccess;  using AutomobileLibrary.Repository;  using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  using System.Windows;  using System.Windows.Controls;  using System.Windows.Data;  using System.Windows.Documents;  using System.Windows.Input;  using System.Windows.Media;  using System.Windows.Media.Imaging;  using System.Windows.Navigation;  using System.Windows.Shapes;  namespace AutomobileWPFApp  {  /// <summary>  /// Interaction logic for MainWindow.xaml  /// </summary>  public partial class WindowCarManagement : Window  {  ICarRepository carRepository;  public WindowCarManagement(ICarRepository repository)  {  InitializeComponent();  carRepository = repository;  }  private Car GetCarObject()  {  Car car = null;  try  {  car = new Car  {  CarId = int.Parse(txtCarId.Text),  CarName = txtCarName.Text,  Manufacturer = txtManufacturer.Text,  Price = decimal.Parse(txtPrice.Text),  ReleasedYear = int.Parse(txtReleasedYear.Text)  };  }  catch(Exception ex)  {  MessageBox.Show(ex.Message, "Get car");  }  return car;  }  public void LoadCarList()  {  lvCars.ItemsSource = carRepository.GetCars();  }  private void btnLoad\_Click(object sender, RoutedEventArgs e)  {  try  {  LoadCarList();  }  catch(Exception ex)  {  MessageBox.Show(ex.Message, "Load car list");  }  }  private void btnInsert\_Click(object sender, RoutedEventArgs e)  {  try  {  Car car = GetCarObject();  carRepository.InsertCar(car);  LoadCarList();  MessageBox.Show($"{car.CarName} inserted successfully", "Insert car");  }  catch(Exception ex)  {  MessageBox.Show(ex.Message, "Insert car");  }  }  private void btnUpdate\_Click(object sender, RoutedEventArgs e)  {  try  {  Car car = GetCarObject();  carRepository.UpdateCar(car);  LoadCarList();  MessageBox.Show($"{car.CarName} updated successfully", "Update car");  }  catch (Exception ex)  {  MessageBox.Show(ex.Message, "Update car");  }  }  private void btnDelete\_Click(object sender, RoutedEventArgs e)  {  try  {  Car car = GetCarObject();  carRepository.DeleteCar(car);  LoadCarList();  MessageBox.Show($"{car.CarName} deleted successfully", "Delete car");  }  catch (Exception ex)  {  MessageBox.Show(ex.Message, "Delete car");  }  }  private void btnClose\_Click(object sender, RoutedEventArgs e) => Close();  }  } |