

Київський національний університет імені Тараса Шевченка  
Факультет інформаційних технологій

Кафедра програмних систем і технологій

ЗВІТ

з лабораторної роботи № 2

Тема: “Автоматизація бізнес-процесів архітектурного 3Д-дизайну”

Дисципліна «Об’єктно-орієнтоване конструювання програм»

Підготував:

студент гр. ІПЗ-23(1)

Шевченко Андрій Віталійович

Дата : 19.05.22

Перевірила:

Зубик Л.В

Київ – 2022

**Тема:** MVC.

**Мета роботи:** ознайомитися з паттерном MVC.

**Умова**

Створити MVC ASP-застосунок. Для теми проекту з попереднього семестру (або іншої самостійно обраної теми, яка буде закріплена за студентом до кінця поточного семестру після її погодження з викладачами) розробити його модель, подання і контролер(-и).

**Тема проекту:** “Автоматизація бізнес-процесів архітектурного 3Д-дизайну”.

**Хід роботи:**

В основі архітектурного шаблону MVC, лежать Model, View, Controller. Для початку реалізації цього шаблону потрібно створити моделі даних, які будуть взаємодіяти з контролерами, та зберігатися в базі даних.

Я створив 3 моделі даних:

- Order - замовлення клієнта

```
public class Order
{
    public int Id { get; set; }
    public string Theme { get; set; }
    public string Description { get; set; }
}
```

- Artist - 3д дизайнер

```

Ссылка: 0
public class Artist
{
    Ссылка: 0
    public int Id { get; set; }
    Ссылка: 0
    public string Name { get; set; }
    Ссылка: 0
    public string LastName { get; set; }
    Ссылка: 0
    public DateTime DateOfBirth { get; set; }

    Ссылка: 0
    public byte Experience { get; set; }
}

```

- FinalOrder - співставлення замовлення та 3д дизайнера на виконання

```

Ссылка: 0
public class FinalOrder
{
    Ссылка: 0
    public int Id { get; set; }
    Ссылка: 0
    public int OrderId { get; set; }
    Ссылка: 0
    public int ArtistId { get; set; }
}

```

Шар доступу до даних (Dal) який взаємодіє з базою даних, в моєму випадку це MSSQL, реалізований на основі Ef core. Я буду використовувати метод Code First.

```

Ссылка: 2
public class ArchitectureDesignDbContext : DbContext
{
    Ссылка: 0
    public ArchitectureDesignDbContext(DbContextOptions<ArchitectureDesignDbContext> options) : base(options)
    {
    }

    Ссылка: 0
    public DbSet<Order> Orders { get; set; }
    Ссылка: 0
    public DbSet<Artist> Artists { get; set; }
    Ссылка: 0
    public DbSet<FinalOrder> FinalOrders { get; set; }

    Ссылка: 0
    protected override void OnModelCreating(ModelBuilder builder)
    {
        base.OnModelCreating(builder);
    }
}

```

Після цього можна приступати до створення логіки застосунку, я буду використовувати паттерн “Strategy”. Я створив інтерфейс з сигнатурою методів та сервіс який буде реалізовувати даний інтерфейс.

```
Ссылка: 0
public interface IArchitectureRepository
{
    Ссылка: 0
    IEnumerable<Order> GetOrders();
    Ссылка: 0
    IEnumerable<Artist> GetArtists();
    Ссылка: 0
    IEnumerable<FinalOrder> GetAOrder();

    Ссылка: 0
    void AddOrders(Order order);
    Ссылка: 0
    void AddArtists(Artist artist);
    Ссылка: 0
    void AddOrder(FinalOrder order);
}
```

```
Ссылка: 1
public class ArchitectureService : IArchitectureRepository
{
    private readonly ArchitectureDesignDbContext _dbContext;

    Ссылка: 0
    public ArchitectureService(ArchitectureDesignDbContext dbContext)
    {
        _dbContext = dbContext;
    }
}
```

```

Ссылка: 1
public void AddArtists(Artist artist)
{
    var artistDto = new Artist
    {
        Id = artist.Id,
        Name = artist.Name,
        LastName = artist.LastName,
        DateOfBirth = artist.DateOfBirth,
        Experience = artist.Experience
    };

    _dbContext.Artists.Add(artistDto);
    _dbContext.SaveChanges();
}

```

```

Ссылка: 1
public void AddOrder(FinalOrder order)
{
    var orderDto = new FinalOrder
    {
        Id = order.Id,
        OrderId = order.OrderId,
        ArtistId = order.ArtistId
    };

    _dbContext.FinalOrders.Add(orderDto);
    _dbContext.SaveChanges();
}

```

```

Ссылка: 1
public void AddOrders(Order order)
{
    var orderDto = new Order
    {
        Id = order.Id,
        Theme = order.Theme,
        Description = order.Description
    };

    _dbContext.Orders.Add(orderDto);
    _dbContext.SaveChanges();
}

Ссылка: 1
public IEnumerable<FinalOrder> GetOrder()
{
    return _dbContext.FinalOrders.ToList();
}

Ссылка: 1
public IEnumerable<Artist> GetArtists()
{
    return _dbContext.Artists.ToList();
}

Ссылка: 1
public IEnumerable<Order> GetOrders()
{
    return _dbContext.Orders.ToList();
}

```

Після виконання попередніх пунктів, я створив 3 контролера для кожної з сутностей, відповідно кожна сутність має методи додавання та перегляду.

```

public class BaseController : Controller
{

    public IActionResult Index()
    {
        return View();
    }
}

```

```

Ссылка: 1
public class OrderController : Controller
{
    private readonly IArchitectureRepository _repository;

    Ссылка: 0
    public OrderController(IArchitectureRepository repository)
    {
        _repository = repository;
    }

    Ссылка: 0
    public IActionResult Index()
    {
        return View();
    }

    [HttpGet]
    Ссылка: 0
    public IActionResult GetAllOrders()
    {
        var result = _repository.GetOrders();
        return View(result);
    }

    [HttpGet]
    Ссылка: 0
    public IActionResult AddOrder(int Id)
    {
        ViewBag.Id = Id;
        return View();
    }

    [HttpPost]
    Ссылка: 0
    public string AddOrder(Order order)
    {
        _repository.AddOrders(order);
        return "Order," + order.Id + ",Added";
    }
}

```

```

public class ArtistController : Controller
{
    private readonly IArchitectureRepository _repository;

    public ArtistController(IArchitectureRepository repository)
    {
        _repository = repository;
    }

    public IActionResult Index()
    {
        return View();
    }

    [HttpGet]
    public IActionResult GetAllArtist()
    {
        var result = _repository.GetArtists();
        return View(result);
    }

    [HttpGet]
    public IActionResult AddArtist(int Id)
    {
        ViewBag.Id = Id;
        return View();
    }

    [HttpPost]
    public string AddArtist(Artist artist)
    {
        _repository.AddArtists(artist);
        return "Artist," + artist.Id + ",Added";
    }
}

```

```

public class FinalOrderController : Controller
{
    private readonly IArchitectureRepository _repository;

    public FinalOrderController(IArchitectureRepository repository)
    {
        _repository = repository;
    }

    public IActionResult Index()
    {
        return View();
    }

    [HttpGet]
    public IActionResult GetAllOrders()
    {
        var result = _repository.GetOrder();
        return View(result);
    }

    [HttpGet]
    public IActionResult AddArtist(int Id)
    {
        ViewBag.Id = Id;
        return View();
    }

    [HttpPost]
    public string AddOrder(FinalOrder order)
    {
        _repository.AddOrder(order);
        return "FinalOrder," + order.Id + ",Added";
    }
}

```

Відповідно код для відображення контролерів на прикладі Order:

```
1 <body>
2 <div>
3 <table>
4 <tr>
5 <td>GetAllOrders</td>
6 <td>AddOrder</td>
7 </tr>
8 <tr>
9 <td>
10 <form action="@Url.Action("GetAllOrders", "FinalOrder")" method="get">
11 <input type="hidden"/>
12 <button type="submit">GetAllOrders</button>
13 </form>
14 </td>
15 <td>
16 <form action="@Url.Action("AddOrder", "FinalOrder")" method="get">
17 <input type="hidden" name="Id"/>
18 <button type="submit">AddOrder</button>
19 </form>
20 </td>
21 </tr>
22 </table>
23 </div>
24 </body>
```

```
1 <body>
2 <div>
3 <table>
4 <tr>
5 <td>Id</td>
6 <td>>OrderId</td>
7 <td>ArtistId</td>
8 </tr>
9 @foreach (var value in Model)
10 {
11 <tr>
12 <td>@value.Id</td>
13 <td>@value.OrderId</td>
14 <td>@value.ArtistId</td>
15 </tr>
16 }
17 </table>
18 </div>
19 </body>
```



```

1  <form method="post" action="">
2      <input type="hidden" value="@ViewBag.Id" name="Id"/>
3      <table>
4          <tr>
5              <td><p>Enter Id</p></td>
6              <td><input type="text" name="Id"/></td>
7          </tr>
8          <tr>
9              <td><p>Enter OrderId: </p></td>
10             <td><input type="text" name="OrderId"/></td>
11          </tr>
12          <tr>
13              <td><p>Enter ArtistId: </p></td>
14              <td><input type="text" name="ArtistId"/></td>
15          </tr>
16          <tr>
17              <td><input type="submit" value="Add Order"/></td>
18          </tr>
19      </table>
20  </form>

```

### Скріншот результатів:

Lab2   Order   Artist   FinalOrder

Lab2   Order   Artist   FinalOrder

GetAllArtist   AddArtist

GetAllArtist   AddArtist

Lab2   Order   Artist   FinalOrder

Enter Id  

Enter Name:  

Enter LastName:  

Enter Experience:  

Add Artist

Artist,0,Added

Id	Name	LastName	DateOfBirth	Experience
1	Andrey	Shevchenko	01.01.0001 00:00:00	2
2	Andrey	Shevchenko	01.01.0001 00:00:00	3
3	Andrey	Shevchenko	01.01.0001 00:00:00	3

## Перевірка БД:

/\*\*\*\*\*\* Скрипт для команды SelectTopNRows из среды SSMS \*\*\*\*\*/

```
SELECT TOP (1000) [Id]
      ,[Name]
      ,[LastName]
      ,[DateOfBirth]
      ,[Experience]
FROM [ArchitectureDesign].[dbo].[Artists]
```

85 %

Результаты    Сообщения

	Id	Name	LastName	DateOfBirth	Experience
1	1	Andrey	Shevchenko	0001-01-01 00:00:00.0000000	2
2	2	Andrey	Shevchenko	0001-01-01 00:00:00.0000000	3
3	3	Andrey	Shevchenko	0001-01-01 00:00:00.0000000	3

## Висновки:

Під час виконання цієї лабораторної роботи я ознайомився з паттерном MVC. Розробив веб-застосунок згідно своєї теми. Застосував ORM Ef core для роботи з БД, та паттерн Strategy.