y

Львівський національний університет імені Івана Франка

Факультет прикладної математики та інформатики

Звіт до курсової роботи з дисципліни "Програмне забезпечення"

Виконав: Середович Володимир
Група: ПМП-31

2021.12.05

Зміст

Розділ 1

Постановка задачі

1.1 Завдання

Роль Фінансовий менеджер – логується, попадає на стартову сторінку з можливістю перейти на сторінку своєї ролі за допомогою меню. Доступний перехід на сторінку з відображенням списку всіх товарів, які були замовлені в постачальників або продані. Може виконувати пошук та сортування. Може сформувати звіт за вибраний період та відобразити сумарний прибуток. Може розлогуватися. Реалізувати названий функціонал.

1.2 Поставлені задачі

Реалізувати завдання на мові C# з використанням Microsoft SQL Server, трьома різними способами:

1 Console Application

2 Windows Forms Application

3 WPF Application

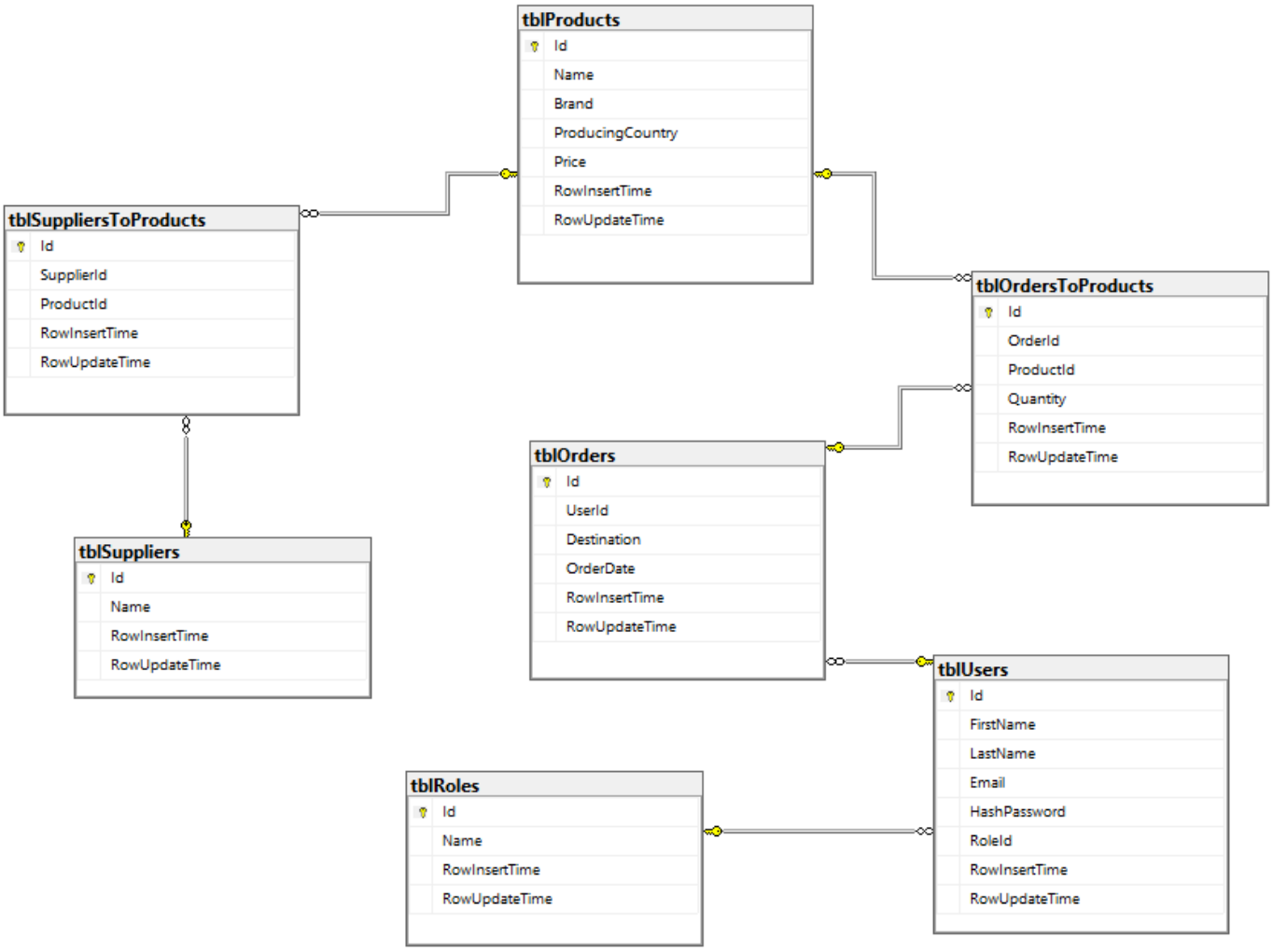
Розділ 2

База даних

2.1 Опис бази даних

Для виконання завдання була створена MSSQL база даних яка складлється з таблиць для ролей, користовачів, постачальників, замовлень, продуктів та ще декілька для встановлення відношень між продуктами та замовленнями, та між постачальниками та продуктами які вони надають.

2.2 Таблиці та відношення між ними



2.3 Опис таблиць

DDL

```
CREATE TABLE tblRoles(
Id bigint IDENTITY(1,1) PRIMARY KEY not null ,
Name nvarchar(50) not null
RowInsertTime datetime2(7) default getutcdate(),
RowUpdateTime datetime2(7) default getutcdate()
);

CREATE TABLE tblProducts(
Id bigint IDENTITY(1,1) PRIMARY KEY not null,
Name nvarchar(50) not null,
Brand nvarchar(50) not null,
ProducingCountry nvarchar(50) not null,
Price int not null
RowInsertTime datetime2(7) default getutcdate(),
RowUpdateTime datetime2(7) default getutcdate()
```

```
);

CREATE TABLE tblSuppliers(
Id bigint IDENTITY(1,1) PRIMARY KEY not null,
Name nvarchar(50) not null
RowInsertTime datetime2(7) default getutcdate(),
RowUpdateTime datetime2(7) default getutcdate()
);

CREATE TABLE tblUsers(
Id bigint not null IDENTITY(1,1) PRIMARY KEY,
FirstName nvarchar(50) not null,
LastName nvarchar(50) not null,
Email nvarchar(50) not null,
HashPassword varchar(256) not null,
RoleId bigint not null
RowInsertTime datetime2(7) default getutcdate(),
RowUpdateTime datetime2(7) default getutcdate()
);

ALTER TABLE tblUsers
ADD CONSTRAINT FK_Users_Roles FOREIGN KEY (RoleId)
REFERENCES tblRoles (Id)
ON DELETE CASCADE
ON UPDATE CASCADE
;

CREATE TABLE tblOrders(
Id bigint not null IDENTITY(1,1) PRIMARY KEY,
UserId bigint not null,
Destination nvarchar(50) not null,
OrderDate DateTime not null
RowInsertTime datetime2(7) default getutcdate(),
RowUpdateTime datetime2(7) default getutcdate()
);

ALTER TABLE tblOrders
ADD CONSTRAINT FK_Orders_Users FOREIGN KEY (UserId)
REFERENCES tblUsers (Id)
ON DELETE CASCADE
ON UPDATE CASCADE
;

CREATE TABLE tblOrdersToProducts(
OrderId bigint not null,
ProductId bigint not null,
Quantity int not null
RowInsertTime datetime2(7) default getutcdate(),
RowUpdateTime datetime2(7) default getutcdate()
);

ALTER TABLE tblOrdersToProducts
```

```
ADD CONSTRAINT FK_OrdersToProducts_Orders FOREIGN KEY (OrderId)
REFERENCES tblOrders (Id)
ON DELETE CASCADE
ON UPDATE CASCADE
;

ALTER TABLE tblOrdersToProducts
ADD CONSTRAINT FK_OrdersToProducts_Products FOREIGN KEY (ProductId)
REFERENCES tblProducts (Id)
ON DELETE CASCADE
ON UPDATE CASCADE
;

CREATE TABLE tblSuppliersToProducts(
Id bigint not null IDENTITY(1,1) PRIMARY KEY,
SupplierId bigint not null,
ProductId bigint not null
RowInsertTime datetime2(7) default getutcdate(),
RowUpdateTime datetime2(7) default getutcdate()
);

ALTER TABLE tblSuppliersToProducts
ADD CONSTRAINT FK_SuppliersToProducts_Suppliers FOREIGN KEY (SupplierId)
REFERENCES tblSuppliers (Id)
ON DELETE CASCADE
ON UPDATE CASCADE
;

ALTER TABLE tblSuppliersToProducts
ADD CONSTRAINT FK_SuppliersToProducts_Products FOREIGN KEY (ProductId)
REFERENCES tblProducts (Id)
ON DELETE CASCADE
ON UPDATE CASCADE
;
```

Розділ 3

Трирівнева архітектура програми

3.1 Опис

Вся архітектура програми складається з трьох шарів: Data Access Layer, Business Logic та UI. DAL відповідає за передачу даних із бази даних до програми або навпаки. BLL обробляє дані отримані через DAL та готує їх для подальшого представлення на UI. Також BLL приймає запити від інтерфейса користовача та виконує їх. UI відповідає за роботу користовача із програмою. В мене UI написана трьома різними способами: WindowsForms application, WPF application та Console application. Але вони всі використовують однакові рівні доступа до даних та бізнес логіки.

3.2 DAL

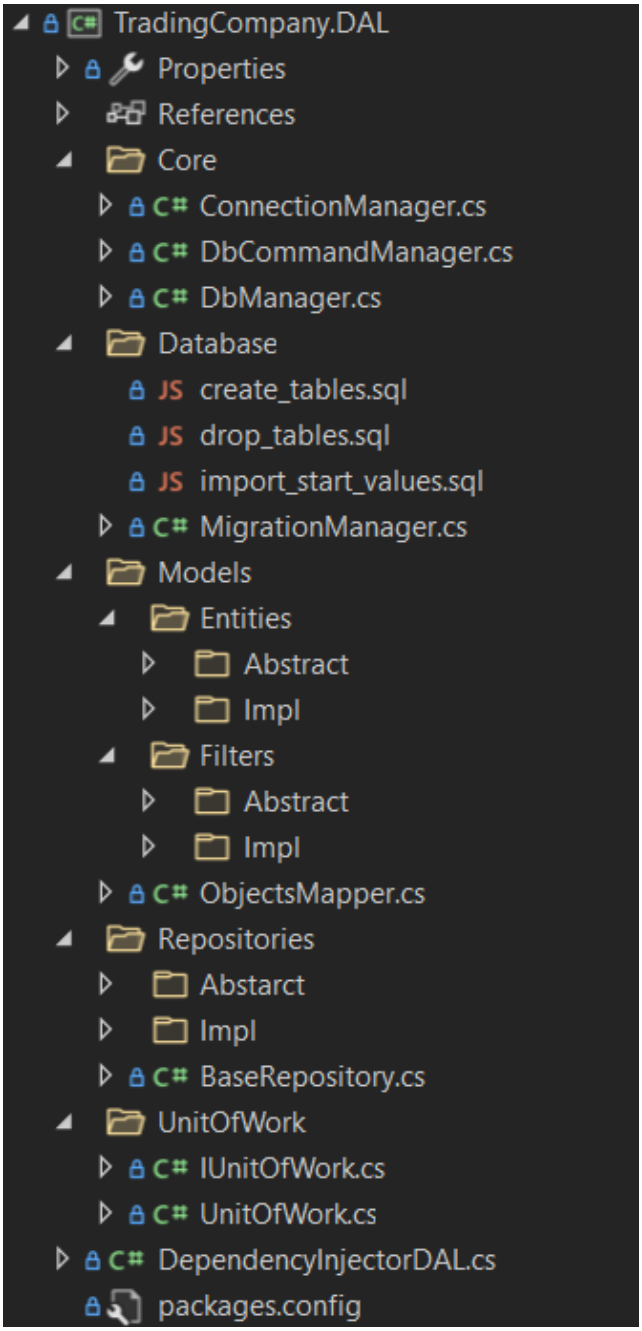
Рівень доступу до бази був написаний використовуючи ADO.NET. Під час написання були вокористані модулі AutoMapper та Unity Container для реаліщації патерна Dependency Injection. Для уникнення дублювання коду був написаний базовий клас репозиторіїв від якого наслідували всі інші. Для побудови SQL запитів до бази використовувались SQL параметри для захисту від SQL ін'єкцій.

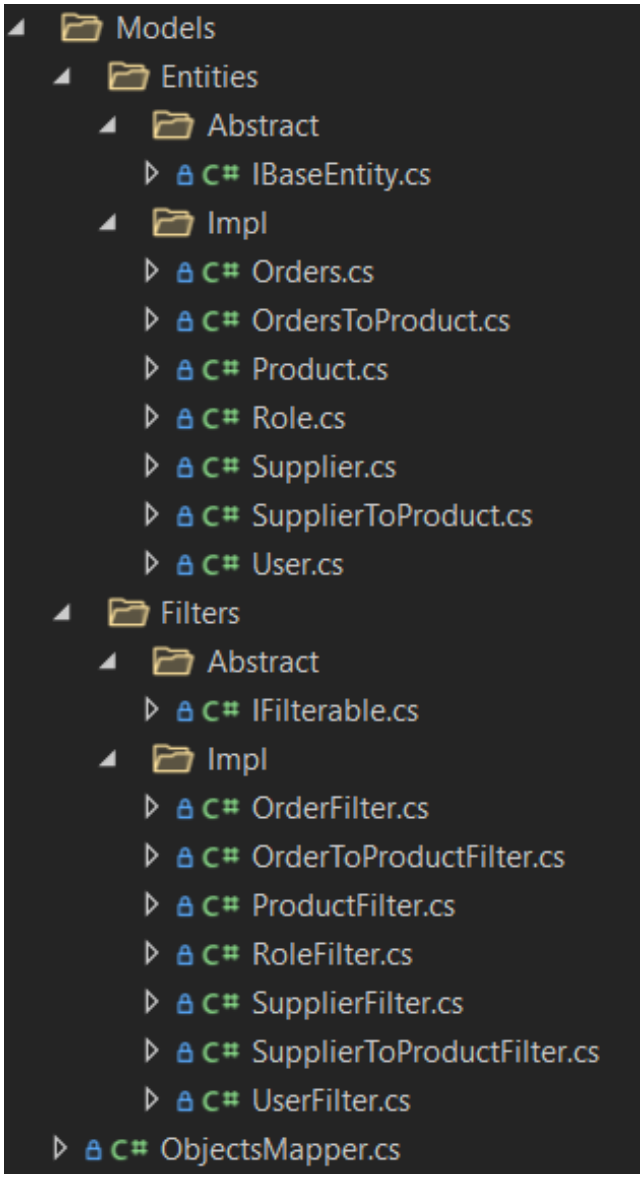
3.2.1 Entity

Entity - це сутність, яка відповідає формату даних із таблиці бази. Приклад:

```
public class Order : IBaseEntity
{
public ulong Id { get; set; }

public ulong UserId { get; set; }

public string Destination { get; set; }

public DateTime OrderDate { get; set; }

public override string ToString()
{
return string.Format("Id: {0} \nUserId: {1} \nDestination: {2} \nOrderDate: {3}\n",
Id, UserId, Destination, OrderDate);
}
}
```

3.2.2 Структура DAL

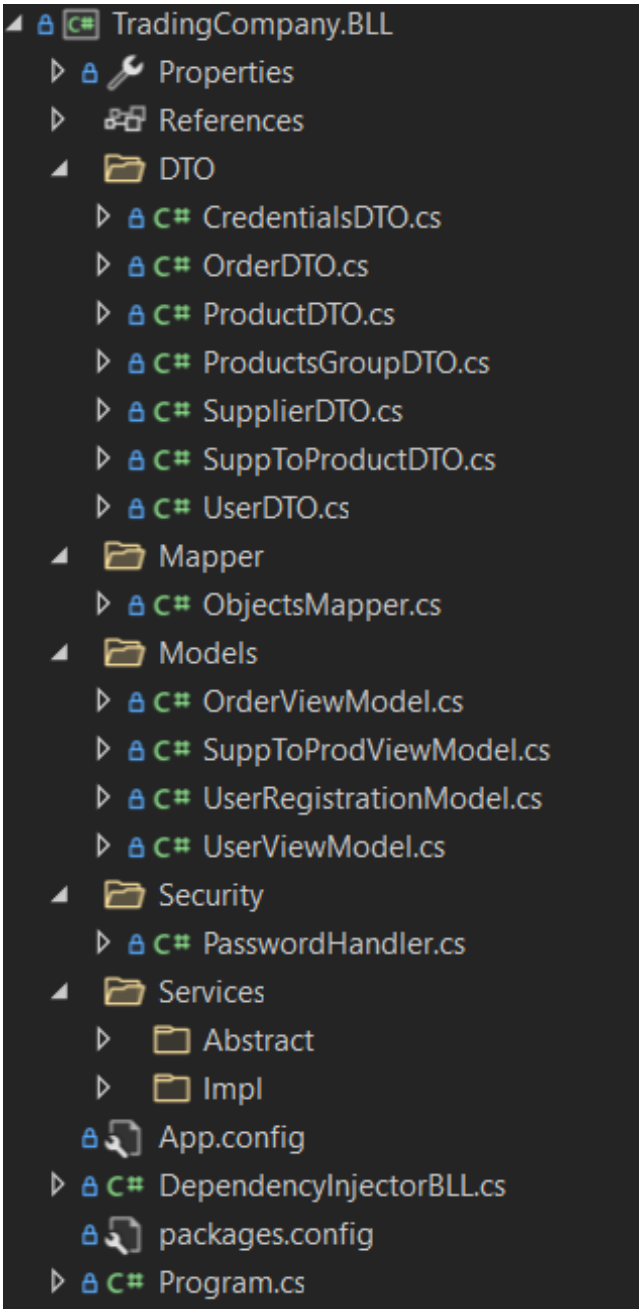




3.3 BLL

У бізнес логці були написані сервіси які відповідають за додавання/редагування/видалення даних з бази шляхом доступу до dal рівня. Також існуюсть Veiw Model які виконують роль представлення даних з бази на інтерфейс користовача. Для хешування пароля із сіллю написаний клас який виконує керування та перевірку пароля.

3.3.1 Структура BLL



3.4 UI

Для інтерфейсу користовача були написані три проекти: Console, WindowsForms та WPF User Interface. Останній з них (WPF) був написаний з використанням MVVM патерну.

Розділ 4

Console Application

4.1 Menu

У реалізації консольного меню заради уникнення повторень був написаний базовий загальний класс меню від якого наслідують всі інші.

```
public abstract class BaseMenu<TRepository, TEntity, TFilter> : IMenu<TRepository, TEnti
where TEntity : IBaseEntity, new()
where TFilter : IFilterable, new()
where TRepository : IRepository<TEntity, TFilter>, new()
public void Add()
{
TRepository repository;
public BaseMenu()
{
repository = new TRepository();
}
{
TEntity entity = InputValues();
repository.Create(entity);
}
public void Delete(ulong id)
{
repository.Delete(new TFilter() { Id = id});
}
public void Update(ulong id)
{
TEntity entity = InputValues();
repository.Update(entity, new TFilter() { Id = id });
}
public IEnumerable<TEntity> GetAll()
{
return repository.GetAll();
}
public void OutputValues()
{
foreach (var entity in GetAll())
{
Console.WriteLine(entity);
}
```

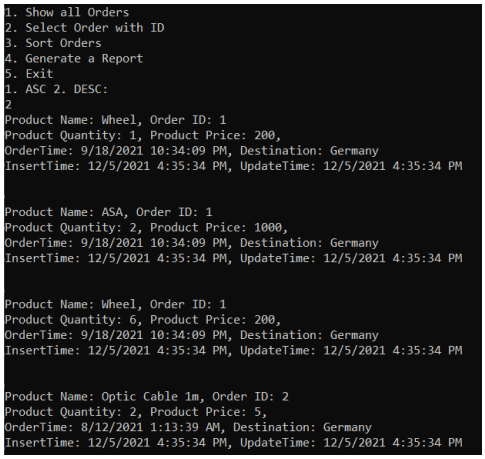
```
}
protected virtual TEntity InputValues()
{
return new TEntity();
}
}
```

Приклад меню для замовленюь OrderMenu

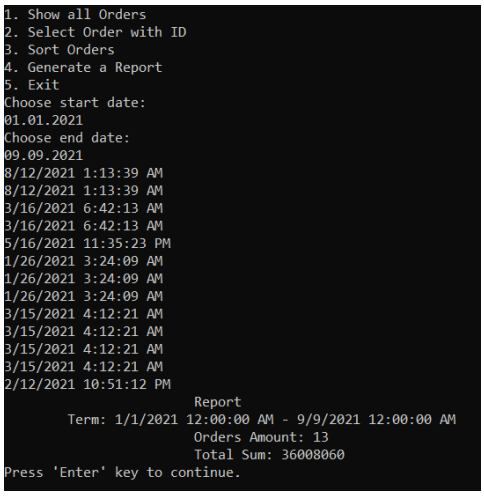
```
class OrderMenu : BaseMenu<OrdersRepository, Order, OrderFilter>
{
protected override Order InputValues()
{
Order order = new Order();
Console.Write("Destination: ");
order.UserId = Convert.ToUInt64(Console.ReadLine().ToString());
Console.Write("OrderDate: ");
order.OrderDate = Convert.ToDateTime(Console.ReadLine().ToString());

return order;
}
}
```

4.2 Вигляд

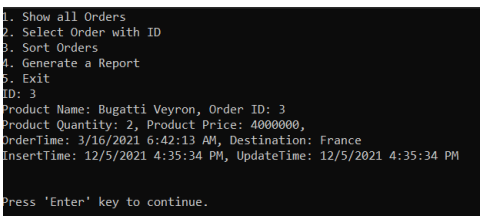
Приклади роботи:



(а) Sort orders



(б) Make a report



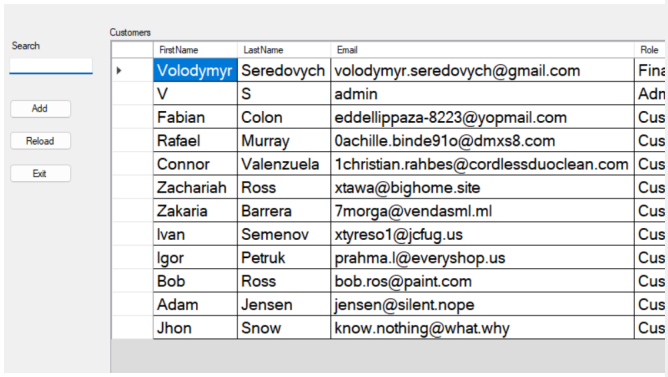
Report by ID

Розділ 5

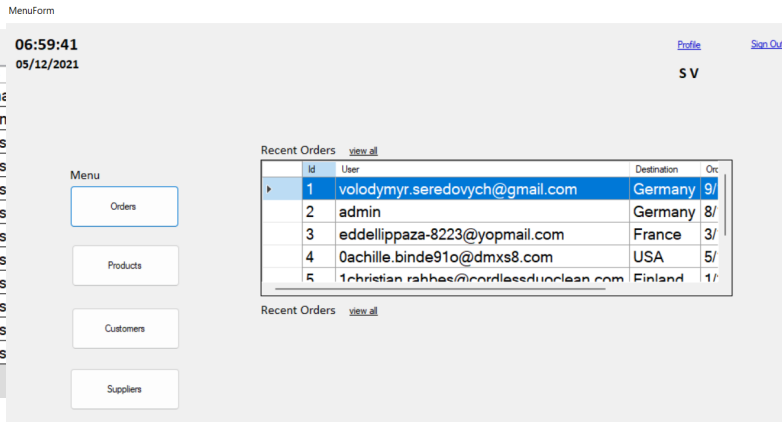
Windows Forms Application

5.1 Вигляд та опис програми

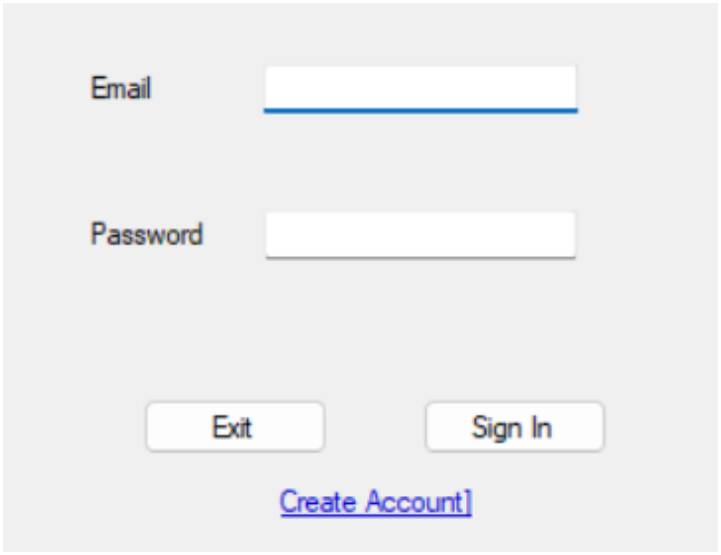
WindowsForms проекті була реалізована CRUD логіка для таблиць бази даних. Є можливість залогінитись під існуючим акаунтом або створити новий. Для зручної роботи з інформацією був створений пошук по полям таблиць. Приклади роботи:



(а) Customers



(б) Orders



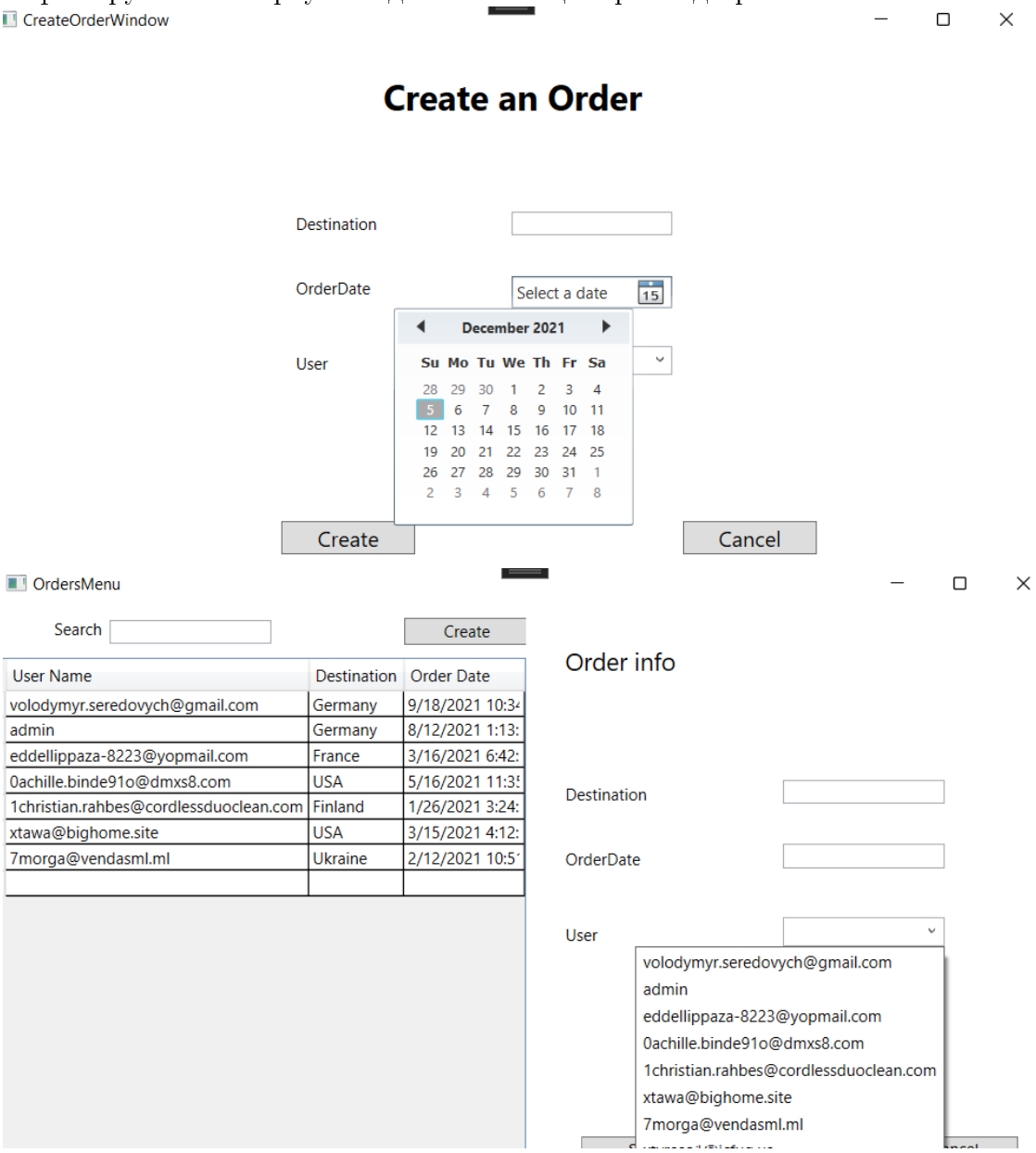
Login

Розділ 6

WPF Application

6.1 Вигляд та опис програми

WPF проекті було реалізовано логування та керування таблицею замовлень. Для роботи з таблицею був використаний патерн MVVM для динамічного оновлення таблиці. Є можливість фільтрувати та сортувати дані з таблиці. Приклади роботи:



Розділ 7

Висновок

7.1 Висновок

За час написання програми я покращив свої знання в роботі з MsSql базою даних, навчився будувати трирівневу архітектуру. Дізнався як хешувати інформацію та працювати з нею. Також навчився працювати х модулями AutoMapper та UnityContainer і з патернами DependencyInjection та MVVM. Окрім цього отривав багато досвіду у роботі з WPF, WindowsForms.