ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ

«ДОНЕЦКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет ИСП

Кафедра ПИ им. Л. П. Фельдмана

Лабораторная работа №6

по курсу: «Архитектура и проектирование ПО»

по теме: «Паттерны объектно-ориентированного проектирования»

Выполнил

ст. гр. ПИ-19а

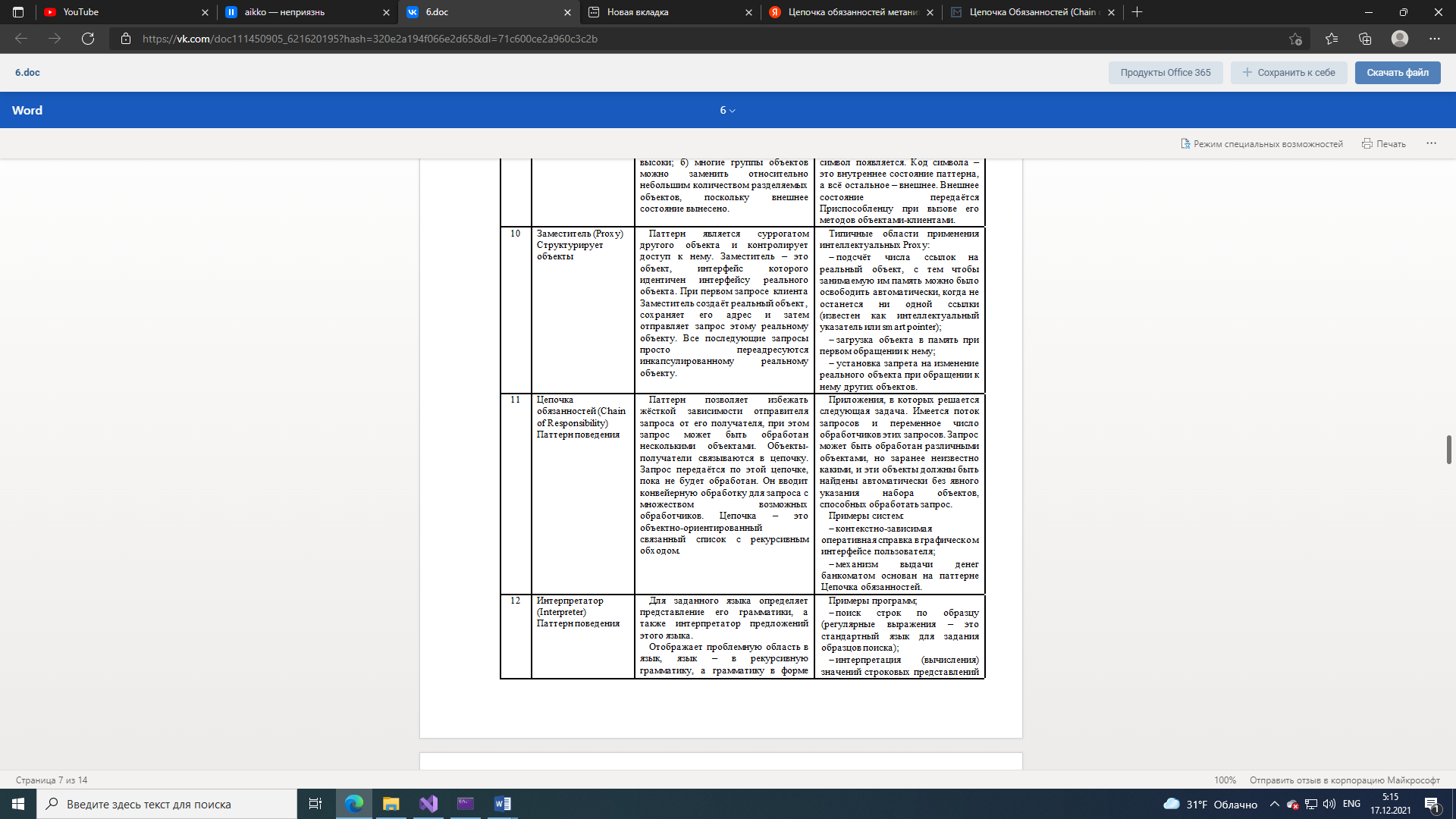
Саевский Олег

Проверила:

Серёженко О.А.

Донецк – 2021

Задание



2. Назначение и условие применимости паттерна проектирование.

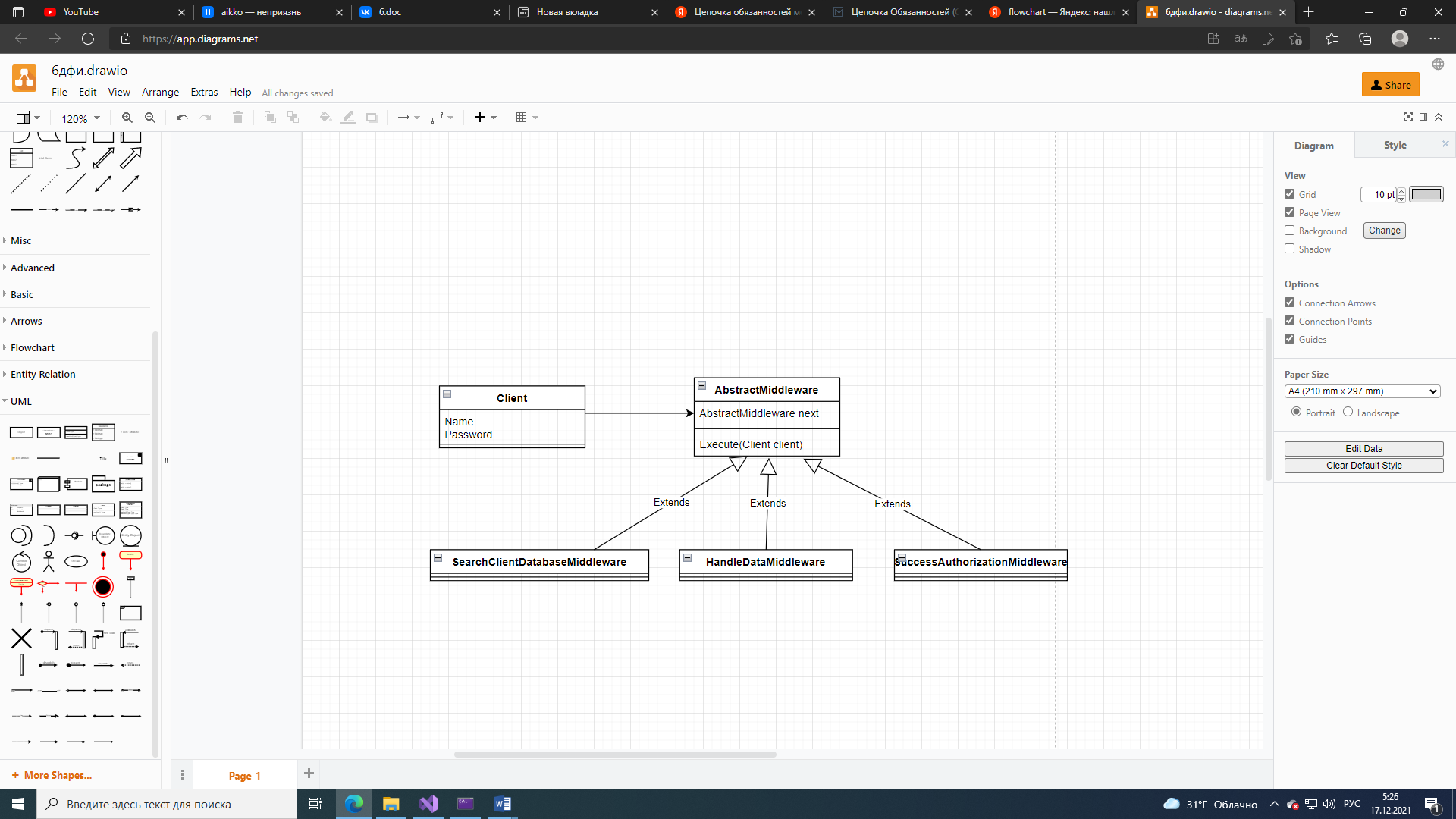
Данный паттерн применяется, когда:

* Когда имеется более одного объекта, который может обработать определенный запрос
* Когда надо передать запрос на выполнение одному из нескольких объект, точно не определяя, какому именно объекту
* Когда набор объектов задается динамически

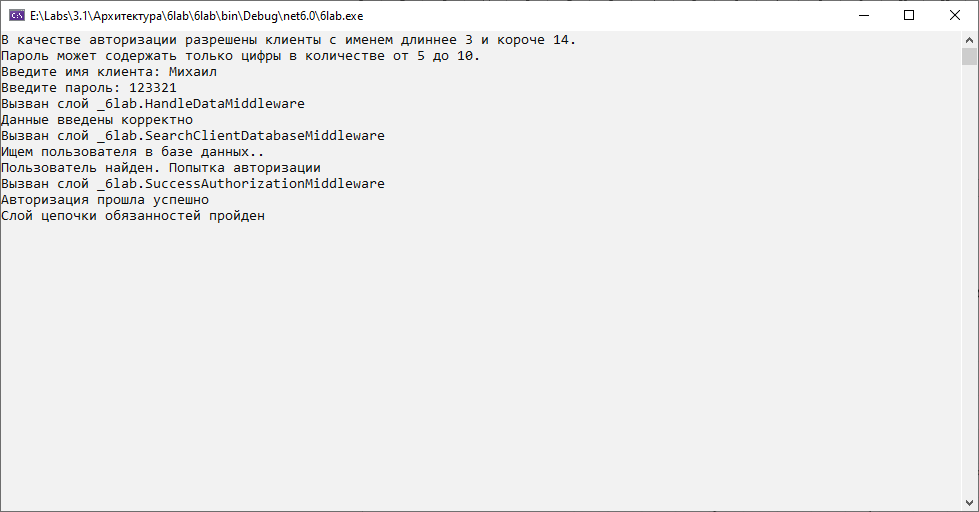
3. Предметная ориентация приложения, на которую настроен используемый паттерн. Описание предметной задачи.

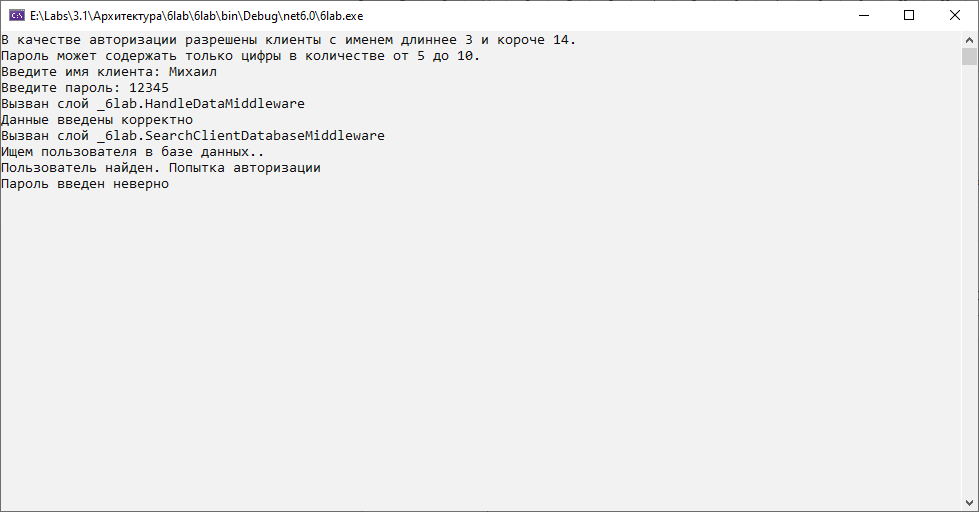
Приложение авторизации пользователя к приложение. Пользователь должен ввести свои данные для входа в учетную запись. Цепочка состоит из трех объектов. Первый – проверяет данные на корректный ввод, второй – ищет пользователя в базе данных и третий – выдает ответ в случае успешной авторизации.

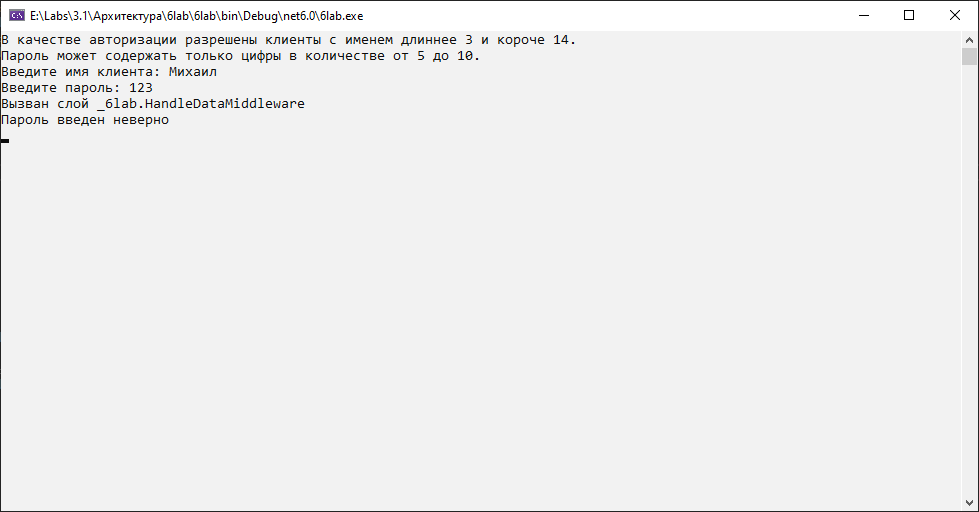
Структура паттерна в виде UML - диаграмма классов и её описание



6-7. Результаты применения паттерна. Реализация паттерна по шагам







Исходный код

Program.cs

using \_6lab;

var middleware = new HandleDataMiddleware(new SearchClientDatabaseMiddleware(new SuccessAuthorizationMiddleware()));

while (true)

{

Console.Clear();

Console.WriteLine("В качестве авторизации разрешены клиенты с именем длиннее 3 и короче 14.");

Console.WriteLine("Пароль может содержать только цифры в количестве от 5 до 10.");

Console.Write("Введите имя клиента: ");

var name = Console.ReadLine() ?? string.Empty;

Console.Write("Введите пароль: ");

var password = Console.ReadLine() ?? string.Empty;

var client = new Client(name, password);

try { middleware.Execute(client); }catch (Exception ex){ Console.WriteLine(ex.Message); }

Console.ReadKey();

}

Middleware.cs

namespace \_6lab

{

internal record class Client(string Name, string Password);

internal abstract class AbstractMiddleware

{

protected AbstractMiddleware? \_next;

protected AbstractMiddleware(AbstractMiddleware next)

{

\_next = next; ;

}

protected AbstractMiddleware() { }

public abstract void Execute(Client client);

}

internal class HandleDataMiddleware : AbstractMiddleware

{

public HandleDataMiddleware(AbstractMiddleware next) : base(next) { }

public override void Execute(Client client)

{

Console.WriteLine($"Вызван слой {GetType()}");

if (client.Name.Length < 3 || client.Name.Length > 14)

throw new Exception("Имя пользователя указано неверно");

if(client.Password.Length < 5 || client.Password.Length > 10)

throw new Exception("Пароль введен неверно");

try { int.Parse(client.Password); } catch { throw new Exception("Пароль должен состоят из цифр"); }

Console.WriteLine("Данные введены корректно");

Thread.Sleep(1500);

\_next?.Execute(client);

}

}

internal class Middleware : AbstractMiddleware

{

public override void Execute(Client client)

{

Console.WriteLine($"Вызван слой {GetType()}");

Thread.Sleep(1000);

Console.WriteLine("Авторизация прошла успешно");

Console.WriteLine("Слой цепочки обязанностей пройден");

}

}

internal class SearchClientDatabaseMiddleware : AbstractMiddleware

{

private readonly List<Client> clients = new() { new Client("Михаил", "123321"), new Client("Сергей", "321123"), new Client("Артем", "66666") };

public SearchClientDatabaseMiddleware(AbstractMiddleware next) : base(next) { }

public override void Execute(Client client)

{

Console.WriteLine($"Вызван слой {GetType()}");

Console.WriteLine("Ищем пользователя в базе данных..");

Thread.Sleep(2000);

var c = clients.FirstOrDefault(x => x.Name == client.Name);

if (c != null)

{

Console.WriteLine("Пользователь найден. Попытка авторизации");

if(c.Password == client.Password)

\_next?.Execute(client);

else throw new Exception("Пароль введен неверно");

}

else throw new Exception("Пользователя с таким именем не найдено в базе данных");

}

}

}