Міністерство освіти і науки України
Національний технічний університет
України
“Київський політехнічний інститут імені Ігоря
Сікорського” Факультет інформатики та обчислювальної
техніки
Кафедра інформаційних систем та технологій

Лабораторна робота №5

з дисципліни «Технології розроблення програмного забезпечення»

Тема: «Патерни проектування»

Виконав:
студент групи ІА-з31
Курбатов Кіріл Андрійович

Перевірив:
Амонс Олександр Анатолійович

Київ 2025

Мета: Вивчити структуру шаблонів «Adapter», «Builder», «Command», «Chain of responsibility», «Prototype» та навчитися застосовувати їх в реалізації програмної системи.

Завдання

• Ознайомитись з короткими теоретичними відомостями.
• Реалізувати частину функціоналу робочої програми у вигляді класів та їхньої взаємодії для досягнення конкретних функціональних можливостей.
• Реалізувати один з розглянутих шаблонів за обраною темою.
• Реалізувати не менше 3-х класів відповідно до обраної теми.
• Підготувати звіт щодо виконання лабораторної роботи. Поданий звіт повинен містити: діаграму класів, яка представляє використання шаблону в реалізації системи, навести фрагменти коду по реалізації цього шаблону.

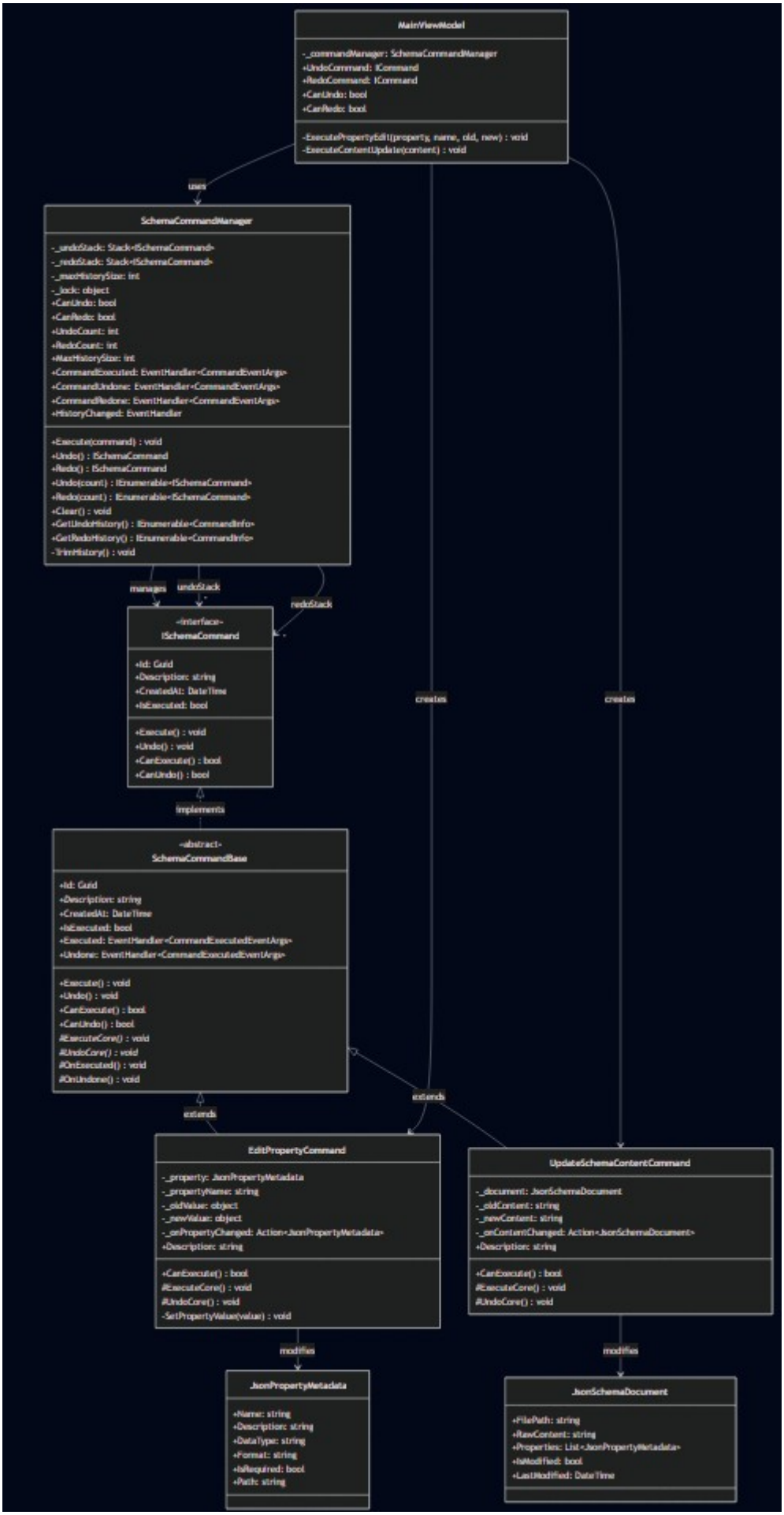

Рис. 1 — Діаграма класів патерну Command для системи Undo/Redo

Реалізація патерну Command

```
namespace JsonTool.Core.Command;

public abstract class SchemaCommandBase : ISchemaCommand
{
    public Guid Id { get; } = Guid.NewGuid();
    public abstract string Description { get; }
    public DateTime CreatedAt { get; } = DateTime.Now;
    public bool IsExecuted { get; protected set; }

    public event EventHandler<CommandExecutedEventArgs>? Executed;
    public event EventHandler<CommandExecutedEventArgs>? Undone;

    public void Execute()
    {
        if (!CanExecute())
        {
            throw new InvalidOperationException($"Cannot execute command: {Description}");
        }

        ExecuteCore();
        IsExecuted = true;
        OnExecuted();
    }

    public void Undo()
    {
        if (!CanUndo())
        {
            throw new InvalidOperationException($"Cannot undo command: {Description}");
        }

        UndoCore();
        IsExecuted = false;
        OnUndone();
    }

    public virtual bool CanExecute() => !IsExecuted;
    public virtual bool CanUndo() => IsExecuted;

    protected abstract void ExecuteCore();
    protected abstract void UndoCore();

    protected virtual void OnExecuted()
    {
        Executed?.Invoke(this, new CommandExecutedEventArgs(this, CommandAction.Execute));
    }

    protected virtual void OnUndone()
    {
        Undone?.Invoke(this, new CommandExecutedEventArgs(this, CommandAction.Undo));
    }
}
 }

 public class CommandExecutedEventArgs : EventArgs
 {
     public ISchemaCommand Command { get; }
     public CommandAction Action { get; }
     public DateTime Timestamp { get; } = DateTime.Now;

     public CommandExecutedEventArgs(ISchemaCommand command, CommandAction action)
     {
         Command = command;
         Action = action;
     }
 }
```