МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

„КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ім. І. Сікорського”

Факультет прикладної математики

Кафедра програмного забезпечення комп’ютерних систем

## КУРСОВА РОБОТА

з дисципліни «Об’єктно-орієнтоване програмування»   
на тему

## Шаблони проектування в ООП.

## Програмне забезпечення організації конференцій

Виконала студентка

ІІ курсу групи КП-02   
Красношапка Анастасія Андріївна залікова книжка КП-0206

Керівник роботи

доцент, к.т.н. Заболотня Т.М. Оцінка

(дата, підпис)

КИЇВ 2022

## ЗМІСТ

**ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ ТА ТЕРМІНІВ** 3

## ВСТУП 4

## СТРУКТУРНО-АЛГОРИТМІЧНА ОРГАНІЗАЦІЯ ПРОГРАМНОЇ СИСТЕМИ ВЕБФРЕЙМВОРКУ 6

* 1. Модульна організація програми 6
  2. Функціональні характеристики 8

## ПРОГРАМНА РЕАЛІЗАЦІЯ СИСТЕМИ ЗА ДОПОМОГОЮ ШАБЛОНІВ ПРОЕКТУВАННЯ 28

* 1. Обґрунтування вибору та опис шаблонів проектування для

програмного забезпечення організації конференцій 28

* 1. Діаграма класів 45
  2. Опис результатів роботи програми 47

## ВИСНОВКИ 52

## СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ 53

## ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ ТА ТЕРМІНІВ

|  |  |
| --- | --- |
| ПЗ - | (Програмне забезпечення) сукупність програм системи оброблення інформації та програмних документів, необхідних для експлуатації цих програм. Це набір інструкцій, які розповідають комп'ютеру, як працювати. |
| C# - | об'єктно-орієнтована мова програмування з безпечною системою типізації для платформи .NET. |

## ВСТУП

Дана курсова робота присвячена розробці програмного забезпечення організації конференцій за допомогою використання шаблонів проектування. Так як сьогодні відбувається автоматизація усіх можливих процесів, люди звикають до того, що необхідність самостійно виконувати однотипні повторювані задачі, як от відсилати запрошення на конференцію десяткам людей, стає все меншою. Саме завдяки використанню розробленого програмного забезпечення можна буде автоматизувати процес створення, редагування конференцій, відправлення запрошень учасникам, а також зручно переглядати всі створені конференції та конференції на які людина була запрошена. Тому розроблене програмне забезпечення є дійсно актуальним. Дана тематика обрана для виконання курсової роботи тому, що результати абстрагування об’єктів у цій предметній галузі дозволяють застосувати вивчені принципи та методи об’єктно‑орієнтованого програмування для створення програмного забезпечення, зокрема шаблони проектування.

*Об’єктом* дослідження є процес організації конференцій.

*Метою роботи* є розроблення програмного забезпечення організації конференцій з використанням шаблонів проектування.

Для досягнення визначеної мети необхідно виконати такі *завдання*:

* абстрагувати об’єкти предметної галузі;
* розробити структурну організацію ПЗ за допомогою застосування основних принципів ООП та шаблонів проектування;
* визначити та описати функціональні характеристики програми;
* обґрунтувати вибір шаблонів проектування, використаних для побудови програми;
* виконати реалізацію програмного забезпечення відповідно до вимог технічного завдання;
* виконати тестування розробленої програми;
* оформити документацію з курсової роботи.

Реалізовані шаблони проектування: Міст, Будівельник, Спостерігач, Стратегія та Стан.

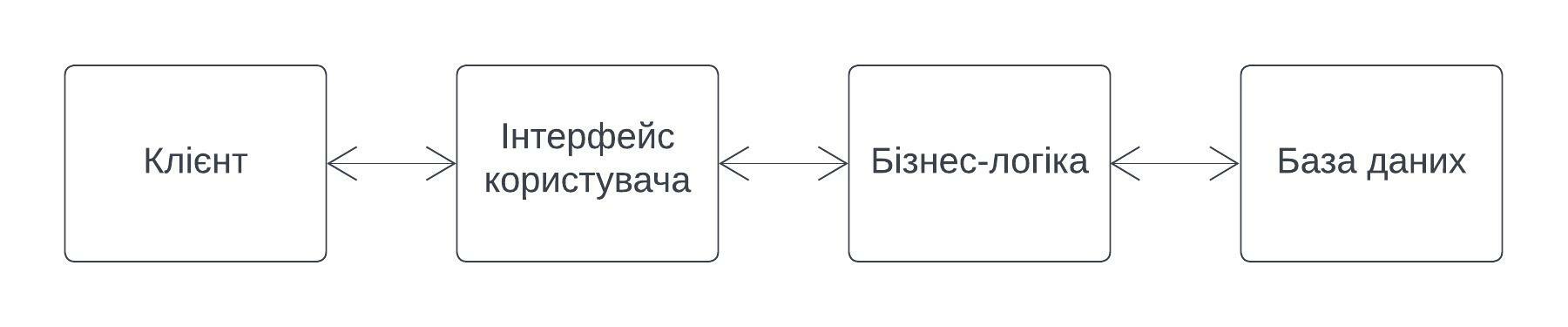
До функціональних можливостей програми належать: створення та редагування конференцій, відправка запрошень учасникам конференцій, можливість авторизуватися, перегляд списку конференцій, на які користувач був запрошений, а також списку конференцій, що були створені даним користувачем.

Для функціонування розробленої програми необхідно забезпечити наявність на комп’ютері 25 Мб вільного дискового простору та встановленого .Net Framework 4.0.

Розроблене програмне забезпечення може бути використане будь-якою людиною, у якої є потреба запланувати конференцію та запросити на неї можливих учасників без необхідності відправляти кожне запрошення окремо.

Пояснювальна записка складається зі вступу, трьох розділів, загальних висновків, списку використаних джерел. Робота містить 26 рисунків. Загальний обсяг роботи – 24 друкованих сторінки, з них 23 сторінки основного тексту та 1 сторінка списку використаних джерел.

1. **ОПИС СТРУКТУРНО-ЛОГІЧНОЇ СХЕМИ ПРОГРАМИ**
   1. **Модульна організація програми**



*Рис. 1.1.1. Структурна схема системи*

Структура системи побудована по шаблону багаторівневої архітектури та складається з рівнів абстракцій, що є взаємоізольовані. Система складається з трьох рівнів: користувацький інтерфейс, бізнес-логіка (або серверна частина) та база даних.

Шаблон багаторівневої архітектури був обраний, спираючись на потребу в побудові ієрархічної взаємодії рівнів абстракції та простоту реалізації.

Програмні засоби реалізовані у вигляді десктопного додатку з консольним інтерфейсом користувача.

* 1. **Функціональні характеристики**

При запуску програми користувач побачить повідомлення про необхідність авторезуватися. Це можна здійснити шляхом входу в існуючий акаунт або зареєструвавшись.

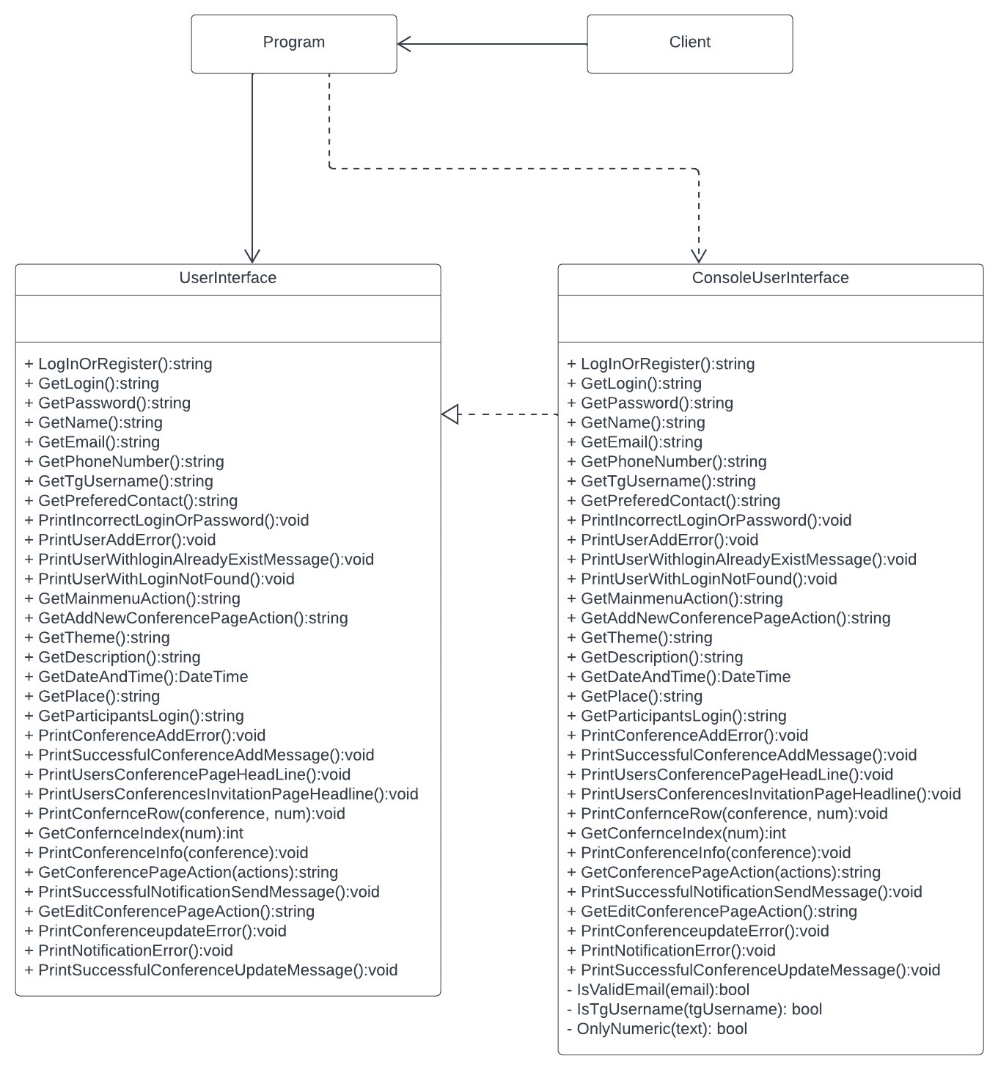
Після авторизації відкривається головне меню, яке дозволяє користувачу вибрати наступну дію: створити нову конференцію, подивитися список створених користувачем конференцій, подивитися список конференцій, на які користувача запросили, вийти з акаунту або вийти з додатку.

Під час створення конференції користувач може в довільній послідовності додавати або змінювати характеристики конференції, додавати довільну кількість учасників. Після створення конференції усім її учасникам відправляється запрошення на електронну пошту, телеграм або номер телефону в залежності від вподобань кожного учасника.

Під час перегляду списку конференцій користувач має змогу вибрати якусь з конференцій та переглянути більш детальну інформацію по ній. Якщо конференція ще не була завершена, її можна відредагувати, завершити та відправити повторне запрошення усім її учасникам. Якщо конференція була завершена, її можна тільки відредагувати.

1. **ПРОГРАМНА РЕАЛІЗАЦІЯ СИСТЕМИ ЗА ДОПОМОГОЮ ШАБЛОНІВ ПРОЕКТУВАННЯ** 
   1. **Обґрунтування вибору та опис шаблонів проектування** **для програмної реалізації мікрофреймворку для web-додатків**
2. **Міст**

Це структурний шаблон проектування, що дозволяє розділяти абстракцію і реалізацію таким чином, щоб вони могли змінюватися незалежно. Шаблон «Міст» використовує інкапсуляцію, агрегування та успадкування для того, щоб розділити відповідальність між класами. Діаграма класів представлена на рис. 2.1.1.



*Рис. 2.1.1. Діаграма класів, які входять до шаблону «Міст»*

*Учасники шаблону:*

* **Abstraction** (Program) визначає інтерфейс взаємодії і зберігає посилання на об'єкт типу Implementor (UserInterface);
* **Implementor** (UserInterface) визначає інтерфейс для класів реалізації клієнтської частини розроблюваного програмного забезпечення;
* **ConcreteImplementor** (ConsoleUserInterface) - один з конкретних виконавців UserInterface.

*Результати використання конструкції:*

* Серверна та клієнтська частини ПЗ взаємодіють між собою через конкретний інтерфейс, а тому при необхідності змінити консольний інтерфейс користувача на будь-який інший, цей процес не потребуватиме внесення змін в серверну частину.

*Обґрунтування використання шаблону:*

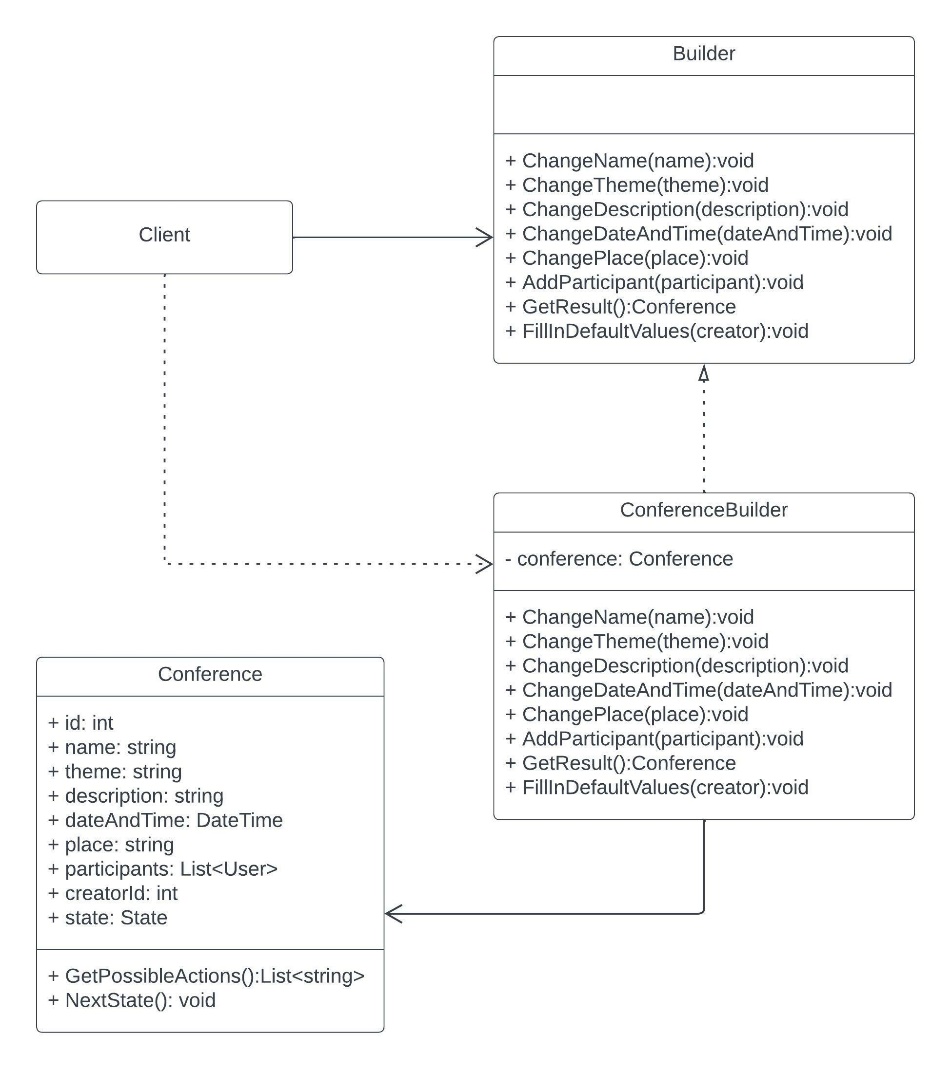
Даний шаблон проектування було використано для того, щоб без значних труднощів можна було змінювати конкретну реалізацію інтерфейсу користувача в залежності від тогочасних потреб. Окрім того, зміни серверної частини не створюють необхідності переписуати клієнтську частину. Таким чином вдалося уникнути жорсткої зв’язки абстракції та імплементації. Замість того, щоб зберігати стан та поведінку кожної реалізації всередині одного класу, що призвело би до ускладнення підтримки такого коду, було виокремлено певний інтерфейс та збережено всередині класу лише посилання на конкретний об’єкт.

1. **Будівельник**

Будівельник – породжуючий шаблон, який відокремлює конструювання складного об’єкта від його подання (моделі), тому в результаті одного і того ж процесу конструювання можуть утворюватися різні подання.

*Учасники шаблону*:

* **Builder** (Builder) – містить перелік всіх можливих абстрактних методів для створення конференції;
* **ConcreteBuilder** (ConferenceBuilder) – конкретний будівельник, який займається побудовою конференції за допомогою реалізації абстрактних методів інтерфейсу Builder;
* **Product** (Conference) – об’єкт, що створюється.



*Рис. 2.1.2. Діаграма класів, які входять до шаблону «Будівельник»*

*Результати використання конструкції:*

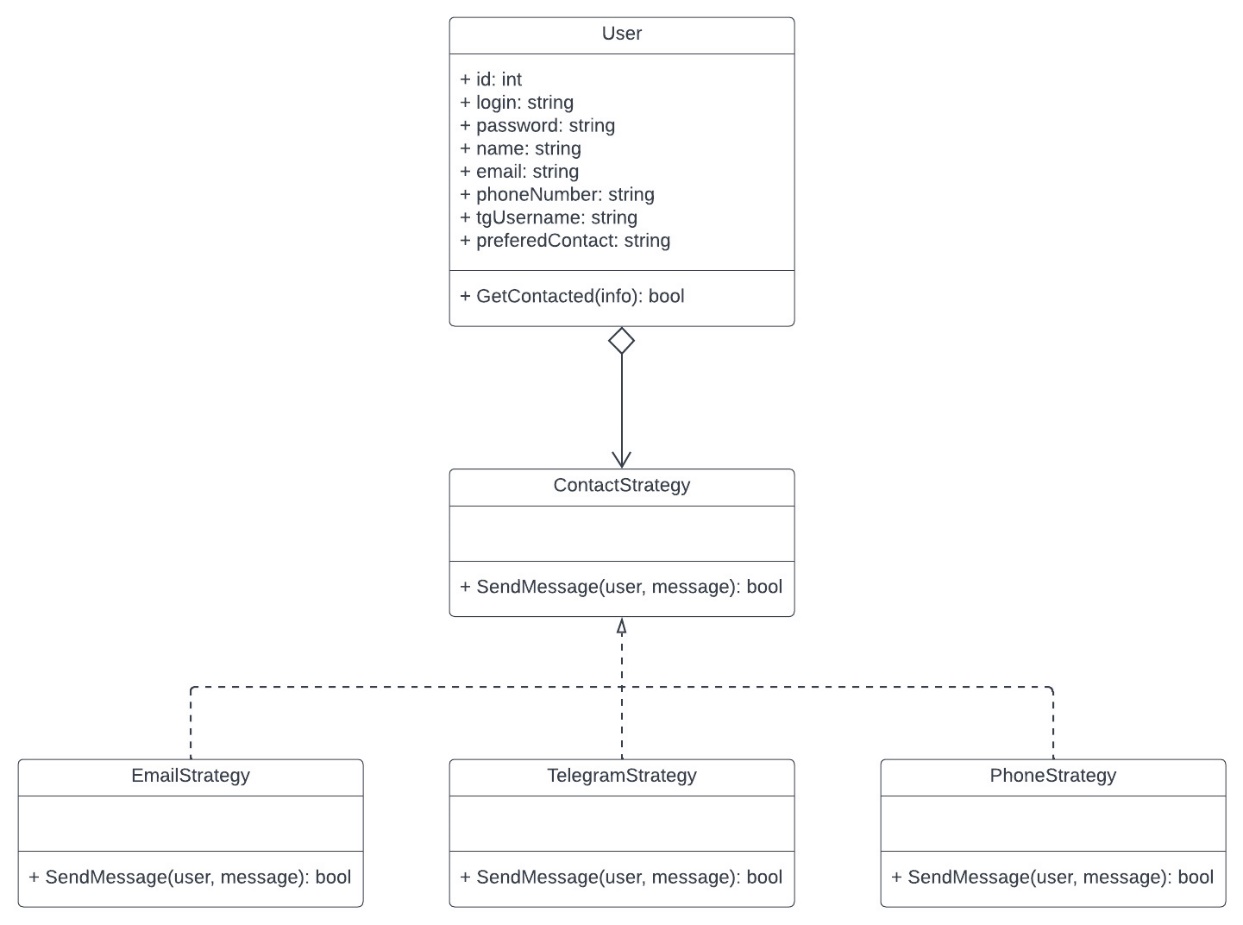
* При створенні конференції додавання полів відбувається у довільній послідовності в залежності від потреб користувача;
* Є можливість додавати довільну кількість учасників до конференції у будь-який момент.

*Обґрунтування використання шаблону:*

Шаблон проектування «Будівельник» було обрано, оскільки процес створення об’єктів можна розділити на окремі етапи, які потрібно було виконувати в різній послідовності, різній кількості, а також деякі не виконувати взагалі в залежності від бажання користувача. Таким чином, завдяки використанню цього шаблону вдалося спростити процес побудови об’єктів конференцій.

1. **Стратегія**

Поведінковий шаблон, який визначає сімейство алгоритмів, інкапсулює кожен з них та робить їх взаємозамінними. Дозволяє змінювати алгоритми незалежно від коду клієнтів. Якщо в системі є алгоритми, які часто можуть використовуватися повторно в різних частинах програми, зручно їх виділити в окрему сутність, параметризувати та запускати там, де це потрібно, не дублюючи сам код.



*Рис. 2.1.3. Діаграма класів, які входять до шаблону «Стратегія»*

*Учасники шаблону*:

* **Strategy** (ContactStrategy) – оголошує загальний інтерфейс для всіх підтримуваних алгоритмів відправки повідомлення користувачу. Клас User користується цим інтерфейсом для виклику конкретного алгоритму, визначеного в класі ConcreteStrategy.
* **ConcreteStrategy** (EmailStrategy, TelegramStrategy, PhoneStrategy) - реалізує алгоритм відправки повідомлення користувачу, використовує інтерфейс, оголошений у класі Strategy.
* **Context** (User) **–** зберігає посилання на об'єкт класу Strategy; Обирає певну конкретну реалізацію алгоритму в залежності від обраного користувачем способу зв’язку з ним.

*Результати використання конструкції:*

* При розсилці запрошень учасникам конференції кожен учасник отримує запрошення на обраний месенджер;
* Є можливість додавати нові методи відправки запрошень, майже не вносячи зміни в існуючу програму.

*Обґрунтування використання шаблону:*

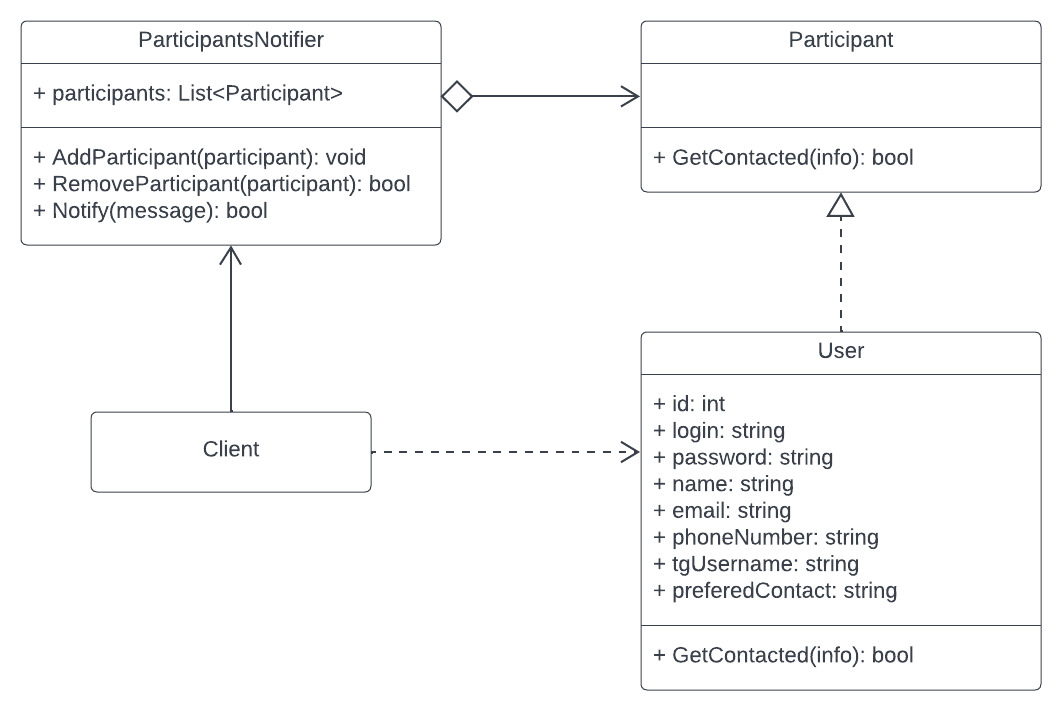
Даний шаблон проектування було обрано тому, що програма повинна забезпечувати різні варіанти алгоритму розсилки запрошень на конференцію. Окрім того, потрібно змінювати поведінку додатку на стадії оброблення кожного нового запиту. Ввівши інтерфейс у вигляді абстрактного класу, стало можливим інкапсулювати конкретні алгоритми. Завдяки цьому клас User став незалежним від даних поведінок, він не перенавантажується додатковим кодом та безліччю перевірок, а сам алгоритм стало легше підтримувати та відлагоджувати. Також, для додання нового алгоритму відправки запрошень достатньо всього лише розробити окремий клас та додати створену стратегію до даного розширення.

1. **Спостерігач**

Поведінковий шаблон проектування, який створює механізм підписки, що дає змогу одним об’єктам стежити й реагувати на події, які відбуваються в інших об’єктах.

*Учасники шаблону*:

* **Publisher** (ParticipantsNotifier) – створює та зберігає список підписників, яким при виклику метода Notify відправляється сповіщення про певні дії.
* **Subscriber** (Participant) - оголошує загальний інтерфейс, яким користується Publisher при надсиланні сповіщень.
* **ConcreteSubscribers** (User) **–** реалізує інтерфейс Subscriber, у відповідь на сповіщення виконує певну дію, задану в інтерфейсі.



*Рис. 2.1.4. Діаграма класів, які входять до шаблону «Спостерігач»*

*Результати використання конструкції:*

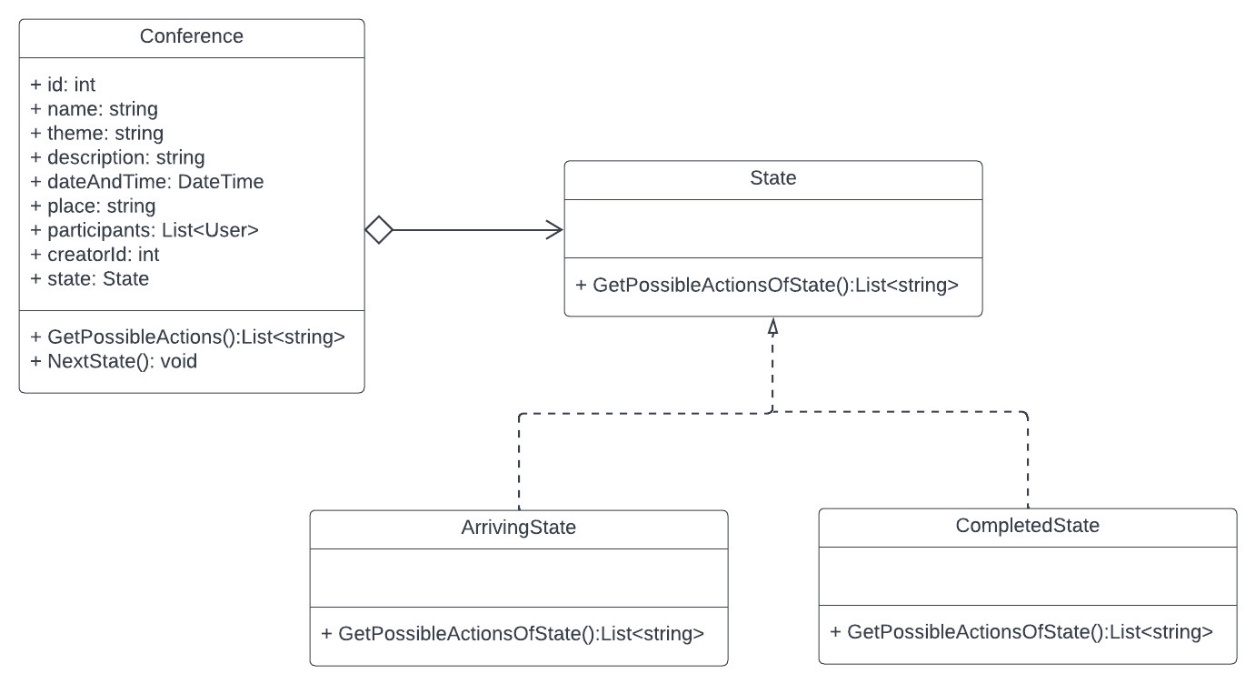
* Після створення конференції усі додані до неї учасники сповіщаються про те, що їх запросили до даної конференції.
* Є можливість в майбутньому додати сповіщення не тільки користувачів системи, а й інших учасників, не змінюючи існуючий код, а просто створивши клас нових об’єктів, який буде реалізувати інтерфейс Participant.

*Обґрунтування використання шаблону:*

Даний шаблон проектування було обрано тому, що є необхідність сповіщувати об’єкти одного класу (User) про створення іншого об’єкта (Conference). Також потрібен був шаблон, який би дозволив в майбутньому спростити процес сповіщення об’єктів різних класів, і це забезпечується завдяки інтерфейсу Participant, притаманному вибраному шаблону.

1. **Стан**

Поведінковий шаблон проектування, що дає змогу об’єктам змінювати поведінку в залежності від їхнього стану. Ззовні створюється враження, ніби змінився клас об’єкта.



*Рис. 2.1.5. Діаграма класів, які входять до шаблону «Стан»*

*Учасники шаблону*:

* **Context** (Conference) – зберігає посилання на об'єкт класу State; В залежності від стану, в якому знаходиться, відправляє різні списки дій, які користувачу дозволено робити над даною конференцією.
* **State** (State) - оголошує загальний інтерфейс, яким користуються об’єкти різних станів конференції.
* **ConcreteStates** (ArrivingState, CompletedState) **–** реалізує інтерфейс State.

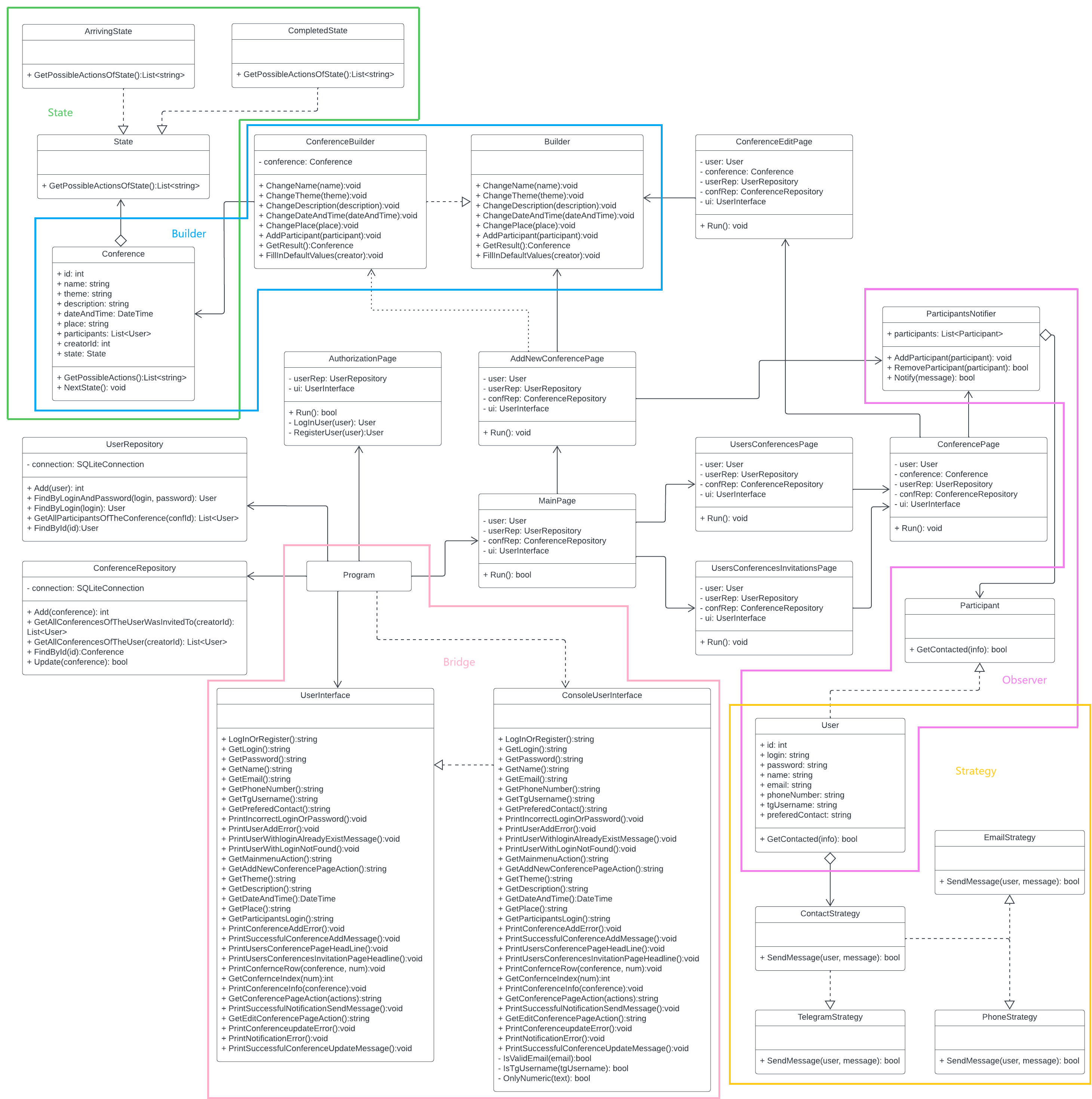
*Результати використання конструкції:*

* В залежності від стану конференції (завершена чи ні) користувач отримує різні варіанти дій, які він може виконати над даною конференцією.
* Є можливість створювати нові стани конференцій, не змінюючи існуючий код, а просто створивши клас нового стану, який буде реалізувати інтерфейс State.

*Обґрунтування використання шаблону:*

Даний шаблон проектування було обрано тому, що була потреба змінювати поведінку конференції в залежності від того, завершена вона чи ні. До того ж планувалось розширення можливих станів конференції (додання стану “Upgraded”), а також поведінки конференції, яка б змінювалася в залежності від її стану, і шаблон «Стан» дозволяє вносити такі зміни максимально легко, майже не змінюючи вже існуючого коду.

* 1. **Діаграма класів**



*Рис. 2.2. Діаграма класів*

* 1. **Опис результатів роботи програми**

|  |
| --- |
| Початок роботи програми |
|  |

|  |
| --- |
| Реєстрація нового користувача |
|  |

|  |
| --- |
| Авторизація користувача |
|  |

|  |
| --- |
| Головне меню |
|  |

|  |
| --- |
| Створення нової конференції |
|  |

|  |
| --- |
| Перегляд списку конференцій користувача |
|  |

|  |
| --- |
| Перегляд списку конференцій, на які користувач був запрошений |
|  |

|  |
| --- |
| Перегляд детальної інформації про конференцію (не завершену) |
|  |

|  |
| --- |
| Перегляд детальної інформації про конференцію (завершену) |
|  |

|  |
| --- |
| Редагування конференції |
|  |

|  |
| --- |
| Відправка запрошень учасникам конференції |
|  |
| В прикладі видно, що перший з учасників обрав Телеграм в якості методу зв’язку, а другий – електронну пошту. |

|  |
| --- |
| Перевірка на правильність введених даних |
|  |
|  |
|  |
|  |
|  |

|  |
| --- |
| Збереження результатів роботи програми до бази даних |
| Таблиця «conferences»: |
| Таблиця «users»: |
| Таблиця «conferences\_users»: |

## ВИСНОВКИ

Метою даної курсової роботи було розроблення програмного забезпечення організації конференцій з використанням шаблонів проектування. Підставою для розроблення стало завдання на виконання курсової роботи з дисципліни «Об’єктно-орієнтоване програмування» студентами ІІ курсу кафедри ПЗКС НТУУ «КПІ ім. І.Сікорського».

Для досягнення поставленої мети у повному обсязі виконано завдання, визначені у аркуші завдання на курсову роботу; розроблено графічні матеріали; реалізовано всі вимоги до програмного продукту, програмного та апаратного забезпечення, наведені у технічному завданні; створено відповідну документацію.

Розроблене програмне забезпечення дозволяє створювати та редагувати конференції відповідно до власних потреб, переглядати список створених вами конференцій і список конференцій, на які ви були запрошені. Також програма дозволяє відправляти запрошення на конференцію максимально зручним для отримувачів способом.

Програму створено на основі використання шаблонів проектування, зокрема до структури розробленого програмного забезпечення входить реалізація п’яти шаблонів, які належать до різних груп шаблонів проектування (структурних, породжувальних та поведінкових).

Для розроблення програмного забезпечення використано мову програмування C#. Платформа: Windows.

Перспективним напрямком подальшого дослідження даної тематики є розширення функціоналу роботи з конференціями.

**СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ**

1. Каталог патернів проектування [Електронний ресурс]: [Веб-сайт]. - Режим доступу: https://refactoring.guru/uk/design-patterns/catalog (дата звернення 28.06.2022) - Назва з екрана
2. Software Design Patterns [Електронний ресурс]: [Веб-сайт]. - Режим доступу: https://www.geeksforgeeks.org/software-design-patterns (дата звернення 28.06.2022) - Назва з екрана