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 №2
Технологія розроблення програмного забезпечення
«*Діаграми прецедентів, класів, структура системи бази даних»*
Варіант 19

Виконав
студент групи ІА-13
Павлюк Оскар Ігорович

Перевірив:
Мягкий М.Ю.

Київ 2023р.

Мета: Навчитися розробляти діаграму прецедентів, діаграму класів, структуру системи бази даних

Хід роботи

..19 IRC client (singleton, builder, abstract factory, template method, composite, client-server)

Клієнт для IRC-чатів з можливістю вказівки порту і адреси з'єднання, підтримка базових команд (підключення до чату, створення чату, установка імені, реєстрація, допомога і т.д.), отримання метаданих про канал.

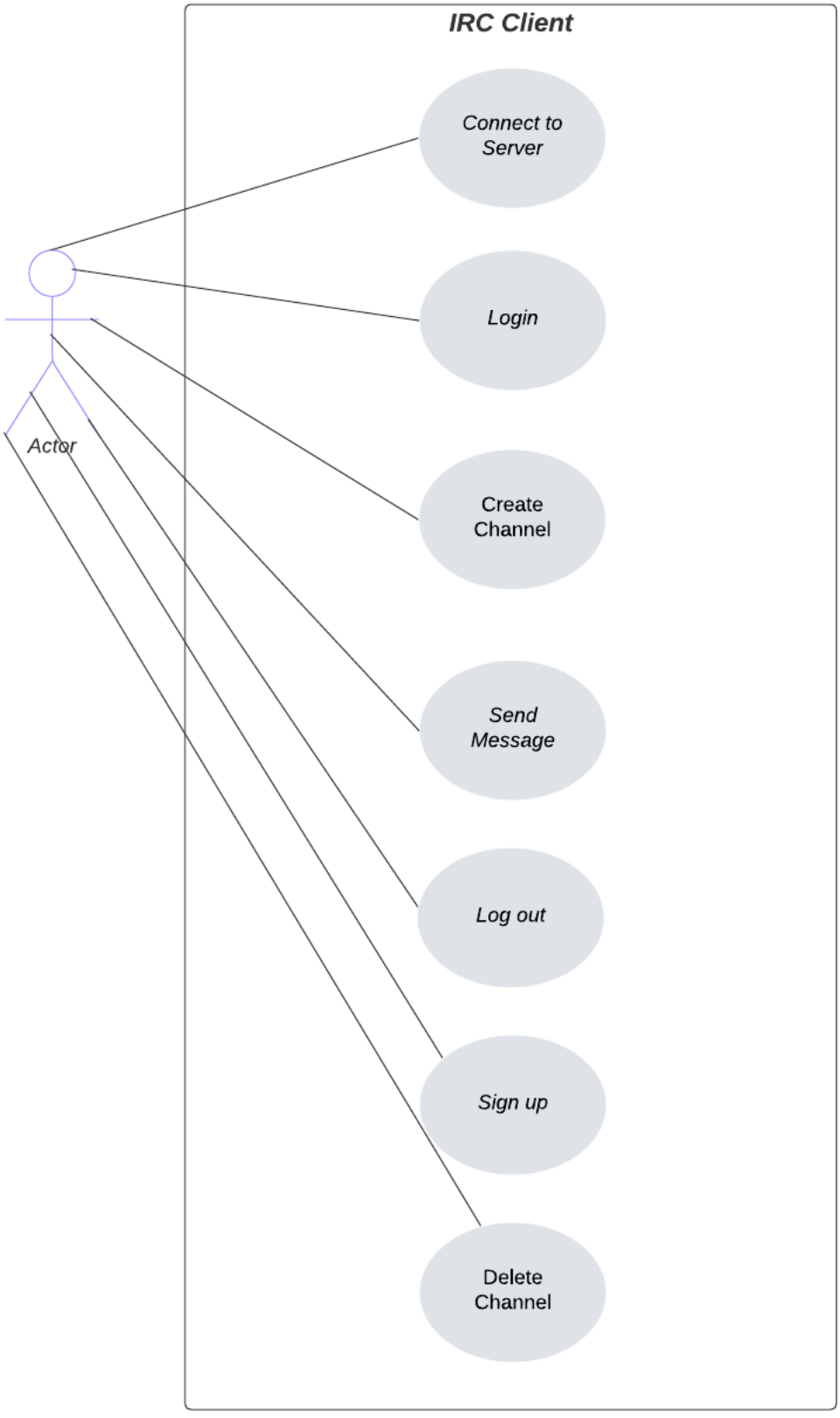
Теорія

Діаграма класів UML - це вид діаграми, який використовується для відображення класів системи, їх атрибутів, методів і взаємозв'язків між ними. Діаграма класів відображає структуру системи і служить основою для подальших деталей дизайну системи.

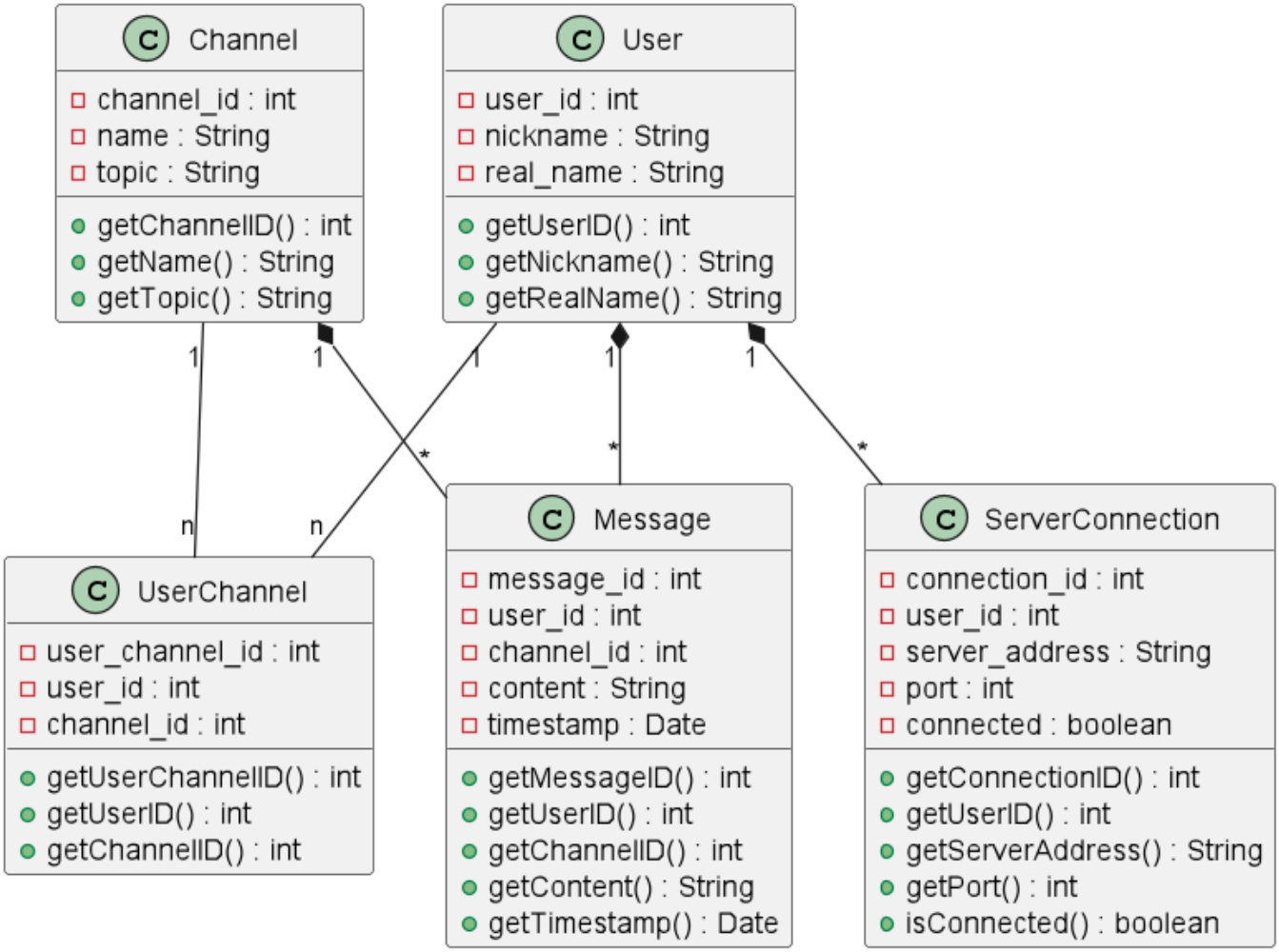
Діаграма прецедентів - це одна з типів діаграм у мові моделювання UML (Unified Modeling Language), яка використовується для візуалізації функціонального поведінки системи з точки зору зовнішніх користувачів (акторів) і їх взаємодії з системою.

Структура системи бази даних (DBMS Structure) - це організація та конфігурація компонентів, які входять до складу системи управління базами даних (СУБД) для зберігання та обробки даних. Ця структура визначає, як інформація зберігається, організована, доступна та обробляється в базі даних.

Діаграма Прецедентів



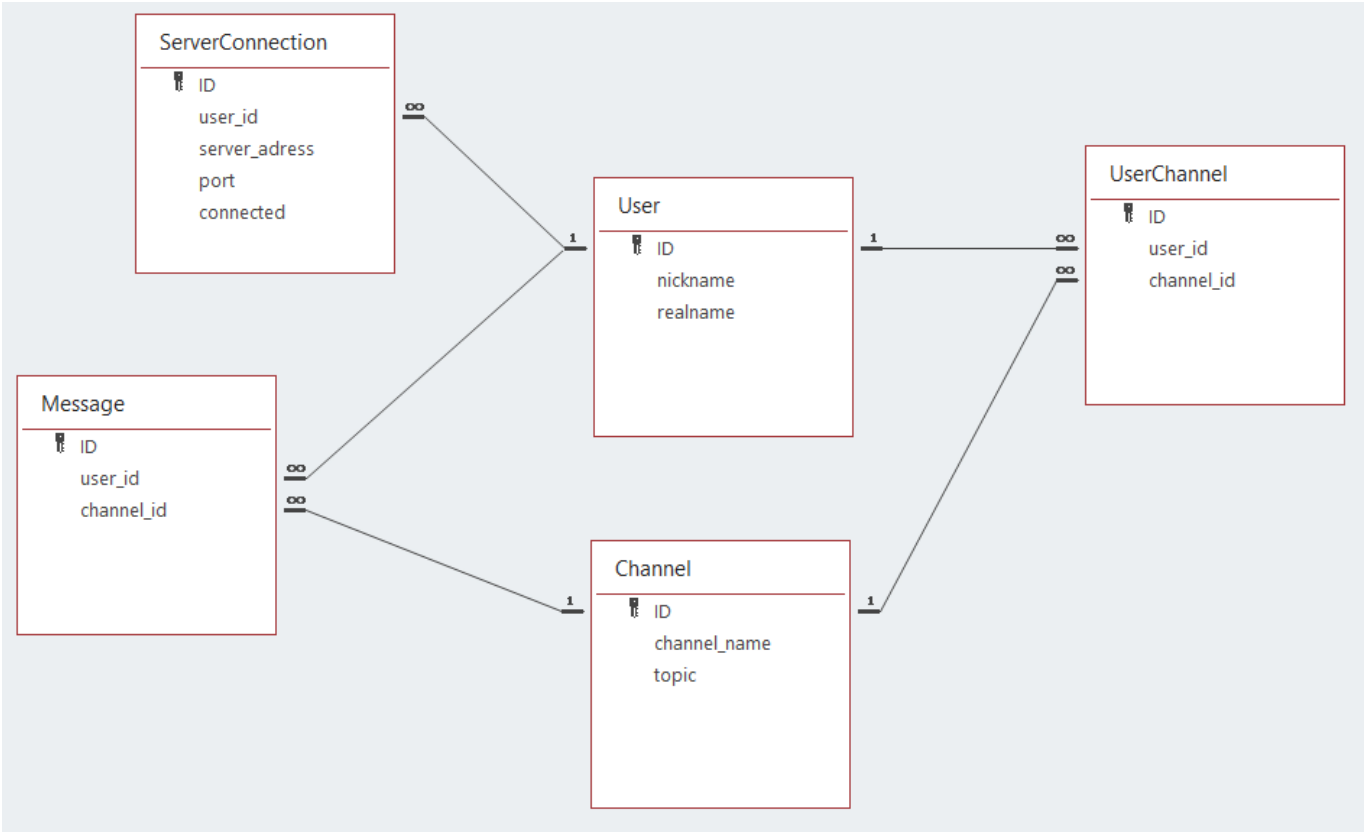
Діаграма класів



File Diagram.puml (код, який відображає діаграму класів) середовище - IntelliJ IDEA 2021.2.2

```
@startuml
class User {
  - user_id : int
  - nickname : String
  - real_name : String
  + getUserID() : int
  + getNickname() : String
  + getRealName() : String
}

class Channel {
  - channel_id : int
  - name : String
  - topic : String
  + getChannelID() : int
  + getName() : String
  + getTopic() : String
}

class Message {
  - message_id : int
  - user_id : int
  - channel_id : int
  - content : String
  - timestamp : Date
  + getMessageID() : int
  + getUserID() : int
  + getChannelID() : int
  + getContent() : String
  + getTimestamp() : Date
}

class ServerConnection {
  - connection_id : int
  - user_id : int
  - server_address : String
  - port : int
  - connected : boolean
  + getConnectionID() : int
  + getUserID() : int
  + getServerAddress() : String
  + getPort() : int
  + isConnected() : boolean
}

class UserChannel {
  - user_channel_id : int
  - user_id : int
  - channel_id : int
  + getUserChannelID() : int
  + getUserID() : int
  + getChannelID() : int
}

User -- UserChannel : "1" -- "n" : UserChannel
Channel -- UserChannel : "1" -- "n" : UserChannel
User *-- Message : "1" -- "*" : Message
Channel *-- Message : "1" -- "*" : Message
User *-- ServerConnection : "1" -- "*" : ServerConnection

@enduml
```

Діаграма Бази даних



Опис трьох прецидентів (функціональність)

1. *Користувач входить в систему (User Login):*

Актор: Користувач

Опис: Користувач вводить своє ім'я користувача та пароль для входу в систему IRC.

Результат: Система перевіряє правильність інформації і надає доступ до чату.

2. *Створення нового чату (Create New Channel):*

Актор: Користувач

Опис: Користувач може створити новий чат, вказавши ім'я та, можливо, тему для каналу.

Результат: Система створює новий чат і дозволяє користувачеві вступити до нього.

3. *Надсилання повідомлення в чат (Send Message):*

Актор: Користувач

Опис: Користувач може надсилати текстові повідомлення в активний чат або особисто іншому користувачеві.

Результат: Повідомлення відображаються у чаті або в особистому діалозі з іншим користувачем.

Зв’язки між класами:

1. Відношення між User та UserChannel вказує, що один користувач може бути пов'язаний з багатьма записами UserChannel, і навпаки. Тобто, один користувач може брати участь в багатьох каналах, а кожен канал може мати багатьох користувачів.

2. Відношення між Channel та UserChannel вказує на зв'язок багато-до-багатьох. Один канал може мати багатьох користувачів, і кожен користувач може бути пов'язаним з багатьма каналами.

3. Відношення між User та Message показує, що один користувач може відправляти багато повідомлень, а кожне повідомлення пов'язане з одним користувачем. Тобто, один користувач може мати багато повідомлень, але кожне з них відправлено лише одним користувачем.

4. Відношення між User та ServerConnection показує, що один користувач може мати багато з'єднань з сервером, а кожне з'єднання пов'язане з одним користувачем. Тобто, користувач може мати багато з'єднань, але кожне з них відноситься до одного користувача.

"1 -- n" вказує на точну кількість (один до багатьох)

"1 -- *" означає, що кількість об'єктів може бути будь-якою (один до багатьох або нуль до багатьох)

Висновок: Під час цієї лабораторної роботи я навчився працювати із різними типами діаграм, та закріпив знання як теоритично так і практично.