

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**  
із дисципліни «*ТПЗ*»  
Тема: Основи проектування.

**Виконав:**

Студент групи ІА-34  
Ковальчук Станіслав

**Перевірив:**

Мягкий М.Ю.

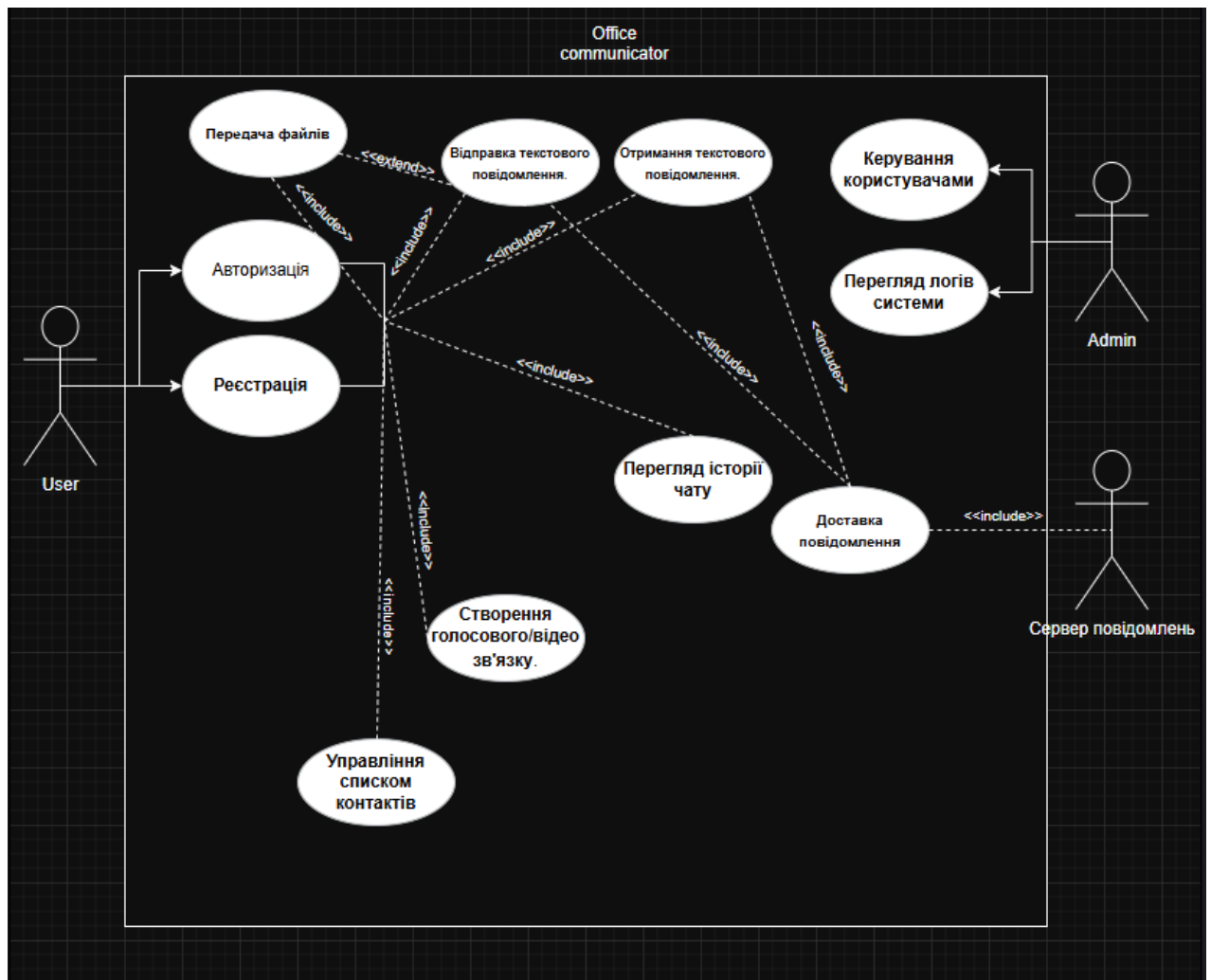
**Тема:** Основи проектування.

**Мета:** Обрати зручну систему побудови UML-діаграм та навчитися будувати діаграми варіантів використання для системи що проєктується, розробляти сценарії варіантів використання та будувати діаграми класів предметної області.

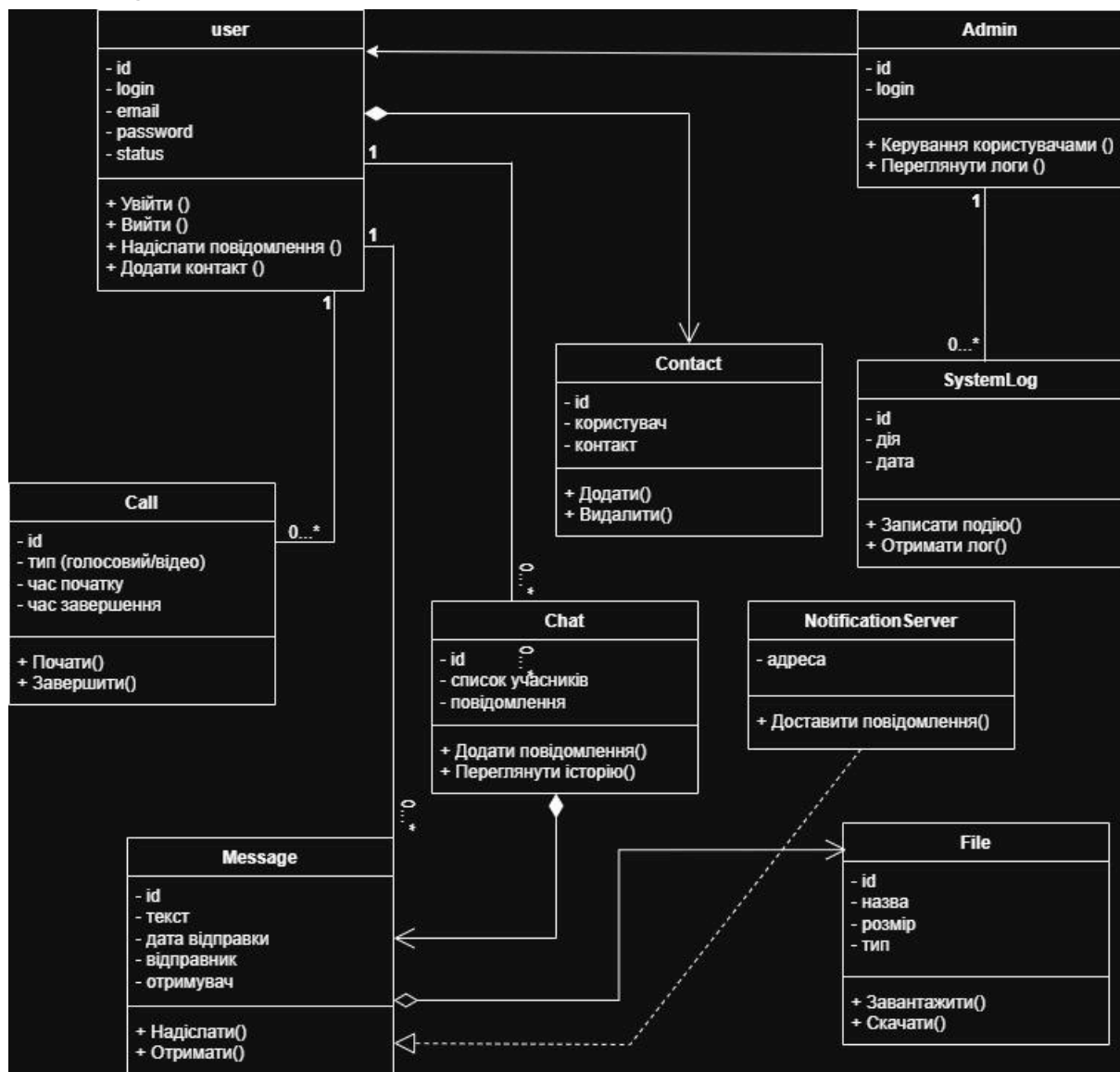
### **Завдання**

- Ознайомитись з короткими теоретичними відомостями.
- Проаналізувати тему та спроектувати діаграму варіантів використання відповідно до обраної теми лабораторного циклу.
- Спроектувати діаграму класів предметної області.
- Вибрати 3 варіанти використання та написати за ними сценарії використання.
- На основі спроектованої діаграми класів предметної області розробити основні класи та структуру бази даних системи. Класи даних повинні реалізувати шаблон Repository для взаємодії з базою даних.
- Нарисувати діаграму класів для реалізованої частини системи.
- Підготувати звіт щодо виконання лабораторної роботи. Поданий звіт повинен містити: діаграму варіантів використання відповідно, діаграму класів системи, вихідні коди класів системи, а також зображення структури бази даних.

Діаграма Use-case:



Class diagram:



Сценарії використання:

### Сценарій 1: Авторизація користувача

- **Взаємодіючі сторони:** Користувач, Система Office Communicator.
- **Передумови:** Користувач має бути зареєстрований обліковий запис.
- **Короткий опис:** Дозволяє користувачу отримати доступ до функцій месенджера.
- **Основний потік подій:**
  - Система відображає вікно входу та запитує логін і пароль.
  - Користувач вводить свої облікові дані.
  - Система перевіряє дані у базі даних через репозиторій.
  - Система надає доступ до головного вікна програми.

- **Винятки:**
  - **Виняток №1:** Неправильні дані. Система виводить повідомлення про помилку, користувач повертається до кроку 2.
- **Постумови:** Користувач авторизований, статус змінено на "Online".

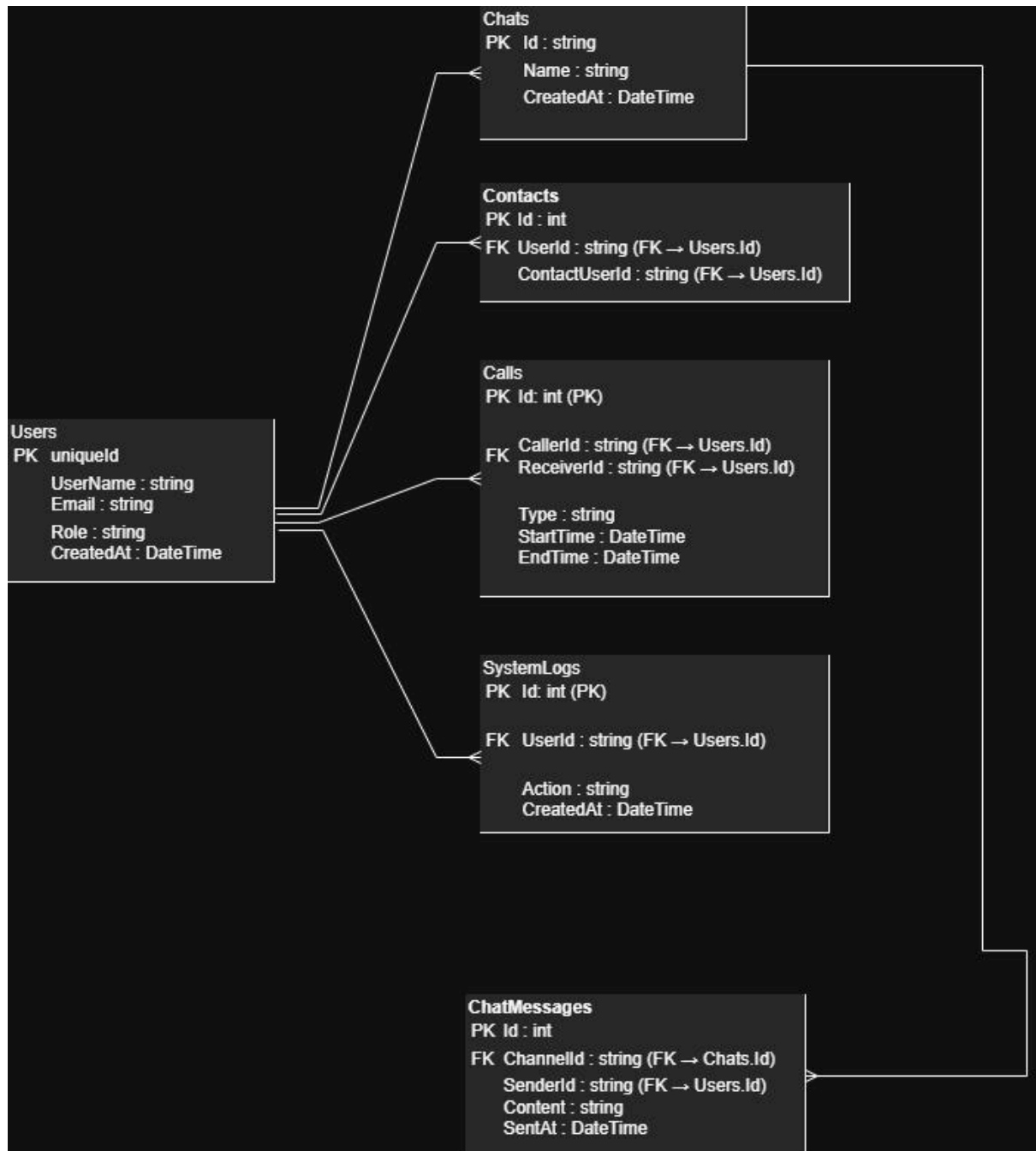
## Сценарій 2: Надсилання текстового повідомлення

- **Взаємодіючі сторони:** Користувач-відправник, Система, Користувач-отримувач.
- **Передумови:** Користувач авторизований та вибрав контакт зі списку.
- **Короткий опис:** Процес створення та передачі текстового повідомлення в чаті.
- **Основний потік подій:**
  - Користувач вводить текст повідомлення у відповідне поле.
  - Користувач натискає кнопку "Надіслати".
  - Система фіксує час відправки та викликає метод `Add()` у репозиторії повідомлень.
  - Система відображає повідомлення у вікні діалогу.
- **Винятки:**
  - **Виняток №1:** Відсутність з'єднання з мережею. Система повідомляє про неможливість доставки.
- **Постумови:** Повідомлення збережене в базі даних та доступне для перегляду отримувачу.

## Сценарій 3: Пошук контакту та додавання до списку

- **Взаємодіючі сторони:** Користувач, База даних (через репозиторій).
- **Передумови:** Користувач знаходиться в розділі контактів.
- **Короткий опис:** Дозволяє знайти співробітника для початку спілкування.
- **Основний потік подій:**
  - Користувач вводить ім'я або прізвище колеги в поле пошуку.
  - Система виконує запит до бази даних.
  - Система відображає список знайдених користувачів.
  - Користувач обирає потрібну особу та додає її до свого списку швидкого доступу.
- **Винятки:**

- **Виняток №1:** Користувача не знайдено. Система відображає повідомлення "Нікого не знайдено".
- **Постумови:** Обраний контакт з'являється у списку контактів користувача.



База даних Office Communicator має реляційну структуру та складається з шести основних таблиць. Центральною сутністю є таблиця Users, з якою пов'язані таблиці повідомлень, контактів, дзвінків та системних логів. Таблиця ChatMessages пов'язана зовнішніми ключами з таблицями Users та Chats, що забезпечує зберігання історії повідомлень у межах окремих чатів. Така структура бази даних забезпечує цілісність даних, масштабованість та зручність подальшого розширення системи.

**Висновок:** У ході виконання лабораторної роботи №2 було розроблено проектну документацію для системи «Office Communicator». Мною було побудовано діаграму варіантів використання, що описує функціонал месенджера, та діаграму класів, яка відображає архітектуру додатка.

Особливу увагу приділено рівню доступу до даних: реалізовано патерн Repository (EfMessageRepository), що дозволяє абстрагувати бізнес-логіку від конкретної реалізації бази даних. Для роботи з даними використано сучасний підхід Code First за допомогою Entity Framework Core. Проектування відповідає принципам SOLID, зокрема принципу інверсії залежностей (Dependency Inversion).

### **Контрольні питання:**

#### **1. Що таке UML?**

UML (Unified Modeling Language) — загальноцільова мова візуального моделювання, призначена для специфікації, візуалізації, проектування та документування компонентів ПЗ, бізнес-процесів і систем.

#### **2. Що таке діаграма класів UML?**

Діаграма класів UML — це логічна (статична) модель системи, що відображає класи, їх атрибути, операції (методи) та зв'язки між класами.

#### **3. Які діаграми UML називають канонічними?**

Канонічними (у вашому конспекті) є такі діаграми UML:

- діаграма варіантів використання (use case),

- діаграма класів (class),
- діаграма кооперації (collaboration),
- діаграма послідовності (sequence),
- діаграма станів (statechart),
- діаграма діяльності (activity),
- діаграма компонентів (component),
- діаграма розгортання (deployment).

#### **4. Що таке діаграма варіантів використання?**

Діаграма варіантів використання (Use Case Diagram) — найбільш загальна концептуальна модель системи, яка показує акторів, варіанти використання і зв'язки між ними, тобто “що система має робити” з точки зору користувачів/зовнішніх систем.

#### **5. Що таке варіант використання?**

Варіант використання (use case) — це опис цілі користувача та функції системи, що реалізує цю ціль, у вигляді узагальненого сценарію взаємодії “актор ↔ система”.

#### **6. Які відношення можуть бути відображені на діаграмі використання?**

Типово на діаграмі use case відображають:

- Асоціацію (association) між актором і варіантом використання;
- «include» (включення) — обов'язкове підключення спільного підсценарію;
- «extend» (розширення) — додатковий/умовний сценарій;
- Узагальнення (generalization) для акторів і/або для варіантів використання.

#### **7. Що таке сценарій?**

Сценарій — це конкретна послідовність кроків (повідомлень/дій) у межах варіанта використання, яка описує, як актор і система взаємодіють для досягнення результату (основний потік та альтернативи).



## 8. Що таке діаграма класів?

Діаграма класів — діаграма, що відображає статичну структуру: класи, їх атрибути/операції та відношення між ними; використовується як основа для проєктування структури коду та даних.

## 9. Які зв'язки між класами ви знаєте?

Основні типи зв'язків між класами:

- Асоціація (association) — загальний зв'язок “має/використовує”;
- Агрегація (aggregation) — “ціле–частина” зі слабким володінням;
- Композиція (composition) — “ціле–частина” із сильним володінням;
- Узагальнення/наслідування (generalization/inheritance) — “є різновидом”;
- Залежність (dependency) — тимчасове використання (через параметри, виклики, імпорти);
- Реалізація (realization) — клас реалізує інтерфейс (якщо інтерфейси моделюються).

## 10. Чим відрізняється композиція від агрегації?

- Агрегація: частина може існувати незалежно від цілого (життєвий цикл не прив'язаний жорстко).
- Композиція: частина не має сенсу/не існує без цілого; при знищенні цілого знищуються і частини (сильне володіння).

## 11. Чим відрізняється зв'язки типу агрегації від зв'язків композиції на діаграмах класів?

На нотації UML:

- Агрегація позначається порожнім (білим) ромбом біля класу-цілого.
- Композиція позначається заповненим (чорним) ромбом біля класу-цілого.  
Семантично це відповідає різній “силі” зв'язку (слабке vs сильне володіння).

## 12. Що являють собою нормальні форми баз даних?

Нормальні форми — це правила нормалізації, які зменшують дублювання даних і аномалії вставки/оновлення/видалення:

- 1НФ: атомарні значення, немає повторюваних груп/масивів у полі.
- 2НФ: 1НФ + відсутність часткових залежностей неключових атрибутів від частини складного ключа.
- 3НФ: 2НФ + відсутність транзитивних залежностей неключових атрибутів від ключа.
- (Додатково: BCNF, 4НФ, 5НФ — жорсткіші умови для специфічних типів залежностей.)

### **13. Що таке фізична модель бази даних? Логічна?**

- Логічна модель БД описує сутності, атрибути та зв'язки незалежно від конкретної СУБД (концептуальна структура даних).
- Фізична модель БД описує реалізацію в конкретній СУБД: таблиці, типи даних, первинні/зовнішні ключі, індекси, обмеження, схеми зберігання тощо.

### **14. Який взаємозв'язок між таблицями БД та програмними класами?**

Найпоширеніший підхід — відповідність рівня “дані ↔ об’єкти” (ORM/відображення):

- Таблиця відповідає класу сутності (Entity/Model);
- Рядок таблиці відповідає об’єкту (екземпляру класу);
- Стовець відповідає полю/властивості класу;
- Зв'язки між таблицями (FK) відповідають асоціаціям між класами (one-to-one, one-to-many, many-to-many).