



Національний технічний університет України  
«Київський політехнічний інститут імені Ігоря Сікорського»  
Факультет інформатики та обчислювальної техніки  
Кафедра інформаційних систем та технологій

**Лабораторна робота №2**

з дисципліни «Технології розроблення програмного забезпечення»

Тема: «Основи проектування»

Варіант: E-mail клієнт

**Виконав:**

студент групи ІА-32

Карповець Роман

Київ - 2025

**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.

**Тема проєкту:** E-mail клієнт

**Опис:** Поштовий клієнт повинен нагадувати функціонал поштових програм Mozilla Thunderbird, The Bat і т.д. Він повинен сприймати і коректно обробляти pop3/smtp/imap протоколи, мати функції автонастройки основних поштових провайдерів для України (gmail, ukr.net, i.ua), розділяти повідомлення на папки/категорії/важливість, зберігати чернетки незавершених повідомлень, прикріплювати і обробляти прикріплені файли.

### Зміст

Теоретичні відомості . . . . .	3
Завдання. . . . .	4
Хід роботи. . . . .	4
Висновок. . . . .	10

## Теоретичні відомості

**Мова UML** – загальноцільова мова візуального моделювання, яка розроблена для специфікації, візуалізації, проектування та документування компонентів програмного забезпечення, бізнес-процесів та інших систем. Мова UML є досить строгим та потужним засобом моделювання, який може бути ефективно використаний для побудови концептуальних, логічних та графічних моделей складних систем різного цільового призначення. Ця мова увібрала в себе найкращі якості та досвід методів програмної інженерії, які з успіхом використовувалися протягом останніх років при моделюванні великих та складних систем.

**Діаграма (diagram)** – графічне уявлення сукупності елементів моделі у формі зв'язкового графа, вершинам і ребрам (дугам) якого приписується певна семантика. Нотація канонічних діаграм є основним засобом розробки моделей мовою UML.

**Діаграма варіантів використання (Use-Cases Diagram)** – це UML діаграма за допомогою якої у графічному вигляді можна зобразити вимоги до системи, що розробляється. Діаграма варіантів використання – це вихідна концептуальна модель проєктованої системи, вона не описує внутрішню побудову системи.

**Сценарії використання** – це текстові уявлення тих процесів, які відбуваються при взаємодії користувачів системи та самої системи. Вони є чітко формалізованими, покроковими інструкціями, що описують той чи інший процес у термінах кроків досягнення мети. Сценарії використання однозначно визначають кінцевий результат.

**Діаграми класів** використовуються при моделюванні програмних систем найчастіше. Вони є однією із форм статичного опису системи з погляду її проєктування, показуючи її структуру. Діаграма класів не відображає динамічної поведінки об'єктів зображених на ній класів. На діаграмах класів показуються класи, інтерфейси та відносини між ними.

**Клас** – це основний будівельний блок програмної системи. Це поняття є і в

мовах програмування, тобто між класами UML та програмними класами є відповідність, що є основою для автоматичної генерації програмних кодів або для виконання реінжинірингу. Кожен клас має назву, атрибути та операції. Клас на діаграмі показується як прямокутник, розділений на 3 області. У верхній міститься назва класу, у середній – опис атрибутів (властивостей), у нижній – назви операцій – послуг, що надаються об'єктами цього класу.

**Логічна модель** бази даних є структурою таблиць, уявлень, індексів та інших логічних елементів бази даних, що дозволяють власне програмування та використання бази даних.

### **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.

### **Хід роботи**

#### **Діаграма варіантів використання**

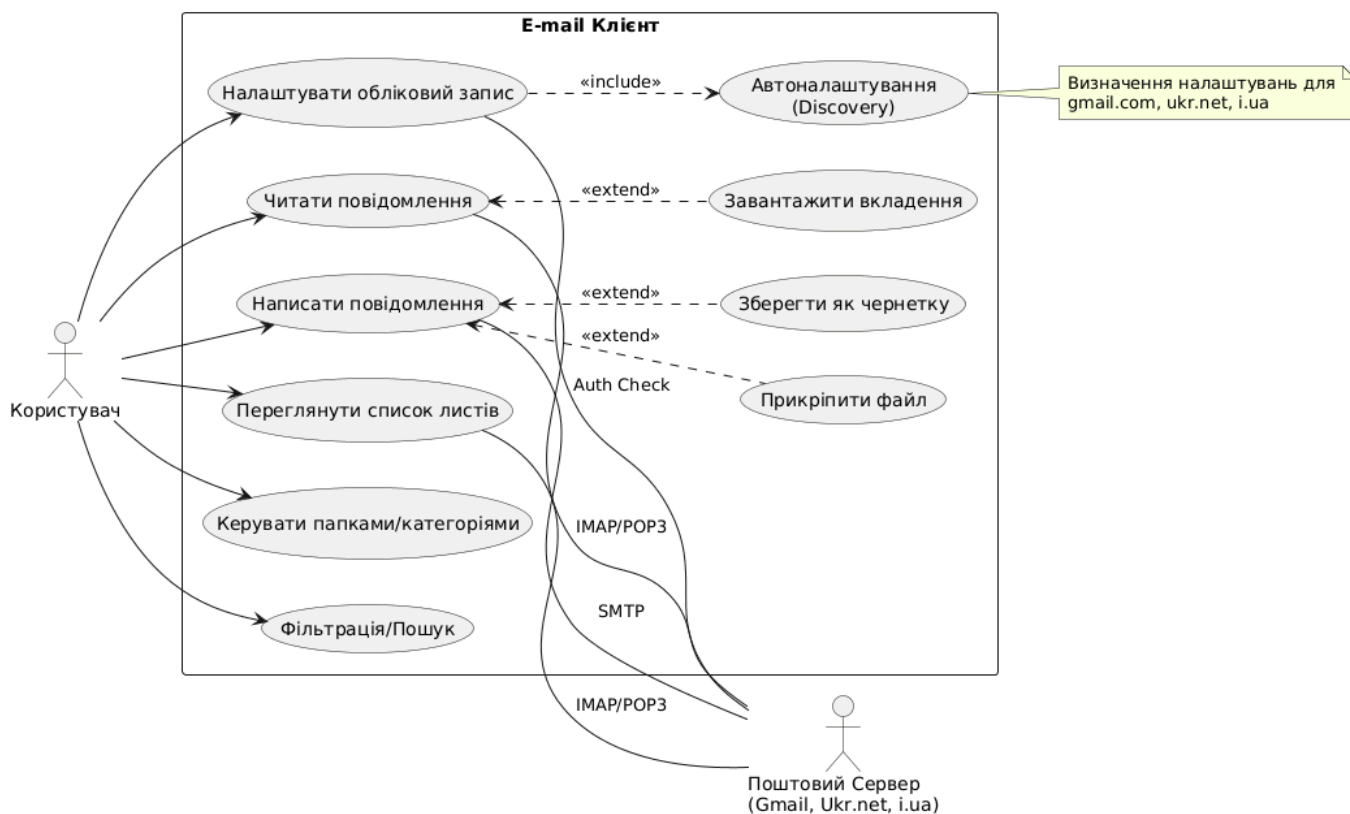


рис. 1. Діаграма варіантів використання

### Сценарії використання

#### Сценарій 1: Налаштування облікового запису (Автоналаштування)

**Передумови:** Програма запущена, з'єднання з Інтернетом активне.

**Постумови:** У разі успішного виконання, профіль користувача збережено, завантажено список папок із сервера. У протилежному випадку обліковий запис не створено.

**Взаємодіючі сторони:** Користувач, Поштовий сервер (IMAP/SMTP).

**Короткий опис:** Цей варіант використання визначає процес підключення пошти з автоматичним визначенням налаштувань провайдера (Gmail, Ukr.net, i.ua).

#### Основний потік подій:

1. Система відображає форму входу.
2. Користувач вводить E-mail адресу та пароль.
3. Система аналізує домен адреси та підставляє відповідні порти/хости (для gmail.com, ukr.net тощо).

4. Система намагається встановити з'єднання з сервером для перевірки даних. Якщо з'єднання не вдалося або дані неправильні, Виняток №1.
5. Система зберігає налаштування та завантажує список листів.

**Винятки: Помилка аутентифікації або з'єднання.** Якщо під час виконання кроку 4 сервер повертає помилку, система виводить повідомлення («Невірний логін/пароль» або «Сервер недоступний»). Користувач може виправити дані та повернутися до кроку 2, ввести налаштування вручну або скасувати дію.

## **Сценарій 2: Відправлення листа з вкладенням**

**Передумови:** Користувач успішно увійшов до системи.

**Постумови:** У разі успішного виконання лист передано на SMTP-сервер, копію збережено у папці «Відправлені». У протилежному випадку лист залишається у статусі «Чернетка» або «Вихідні».

**Взаємодіючі сторони:** Користувач, Поштовий сервер (SMTP).

**Короткий опис:** Користувач створює новий лист, додає файл та відправляє його адресату.

**Основний потік подій:** Система відкриває вікно редактора повідомлень.

1. Користувач заповнює поля «Кому», «Тема» та вводить текст листа.
2. Користувач обирає опцію прикріплення файлу та вказує шлях до нього.
3. Система перевіряє розмір файлу та прикріплює його. Якщо розмір перевищує ліміт, Виняток №1.
4. Користувач натискає кнопку «Відправити».
5. Система перевіряє формат адреси отримувача. Якщо формат неправильний, Виняток №2.
6. Система з'єднується з SMTP-сервером та передає дані.

**Винятки: Виняток №1: Перевищення розміру вкладення.** Якщо файл занадто великий (наприклад, >25 Мб), система виводить попередження і не прикріплює файл. Користувач повертається до кроку 3. **Виняток №2: Некоректна адреса.** Якщо поле «Кому» містить помилки, система підсвічує його червоним. Користувач повертається до кроку 2.

**Примітки:** Відсутні.

### **Сценарій 3: Отримання пошти (Синхронізація)**

**Передумови:** Користувач авторизований.

**Постумови:** Список листів у клієнті відповідає стану на сервері.

**Взаємодіючі сторони:** Користувач, Поштовий сервер (IMAP/POP3).

**Короткий опис:** Система завантажує заголовки нових повідомлень та сортує їх.

#### **Основний потік подій:**

1. Система встановлює з'єднання з сервером вхідної пошти. Якщо Інтернет відсутній, Виняток №1.
2. Система запитує перелік нових повідомлень.
3. Система завантажує заголовки (тема, дата, відправник) та відображає їх у списку жирним шрифтом (як непрочитані).
4. Система застосовує локальні фільтри (розподіляє листи по категоріях/папках).

**Винятки: Виняток №1: Відсутнє з'єднання.** Система відображає стан скриньки на момент останнього оновлення та виводить індикатор «Офлайн».

**Примітки:** При використанні протоколу POP3 листи можуть бути видалені з сервера після завантаження (залежно від налаштувань).

#### **Діаграма класів**

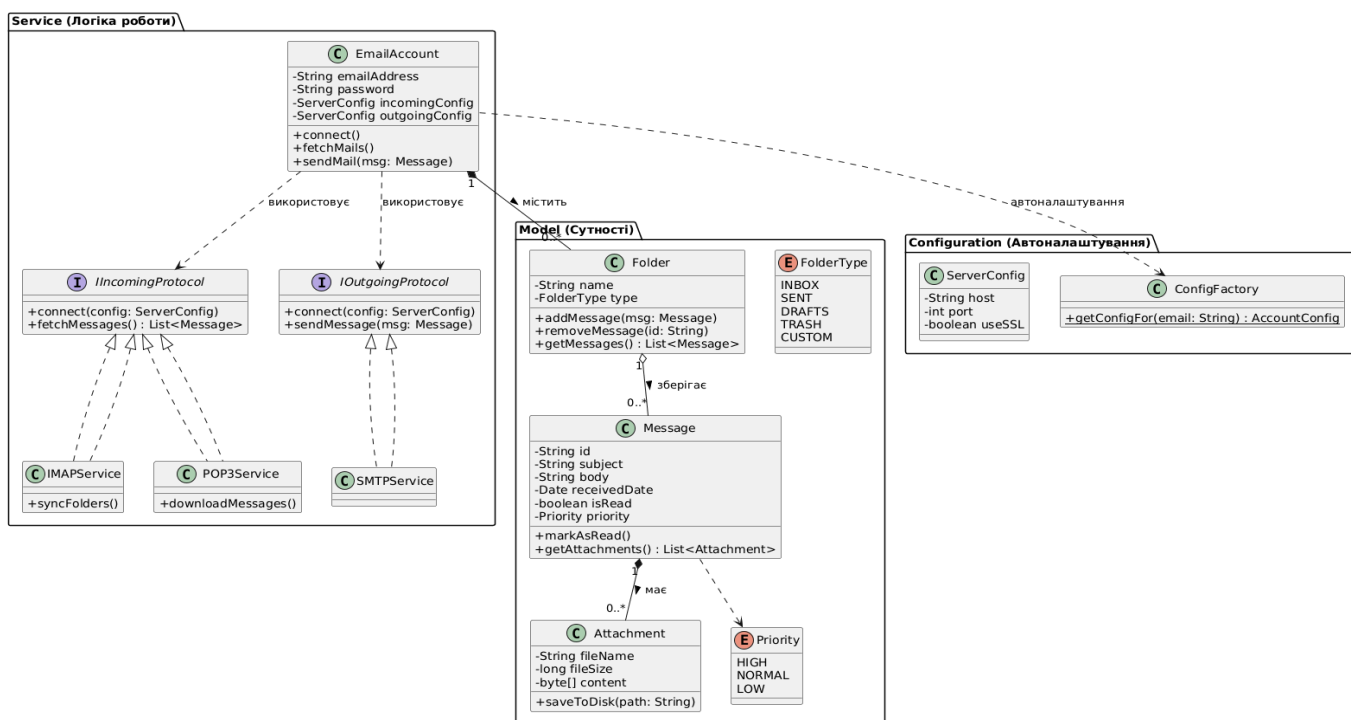


рис 2. Діаграма класів

### Зв'язки між класами

**EmailAccount – Folder Композиція.** EmailAccount володіє об'єктами Folder. При видаленні облікового запису всі пов'язані з ним папки знищуються.

**Folder – Message Агрегація.** Folder містить об'єкти Message, але Message є самостійною сутністю (лист може бути переміщений до іншої папки, не зникаючи з системи).

**Message – Attachment Композиція.** Message володіє об'єктами Attachment. Вкладення є частиною структури листа і видаляються разом із ним (якщо не були попередньо збережені на диск як окремі файли).

**Message – Priority Асоціація.** Message посилається на перелік Priority (enum) для визначення атрибуту важливості листа.

**EmailAccount – IncomingProtocol Асоціація (Стратегія).** EmailAccount посилається на абстракцію IncomingProtocol, що дозволяє динамічно змінювати спосіб отримання пошти (IMAP або POP3).

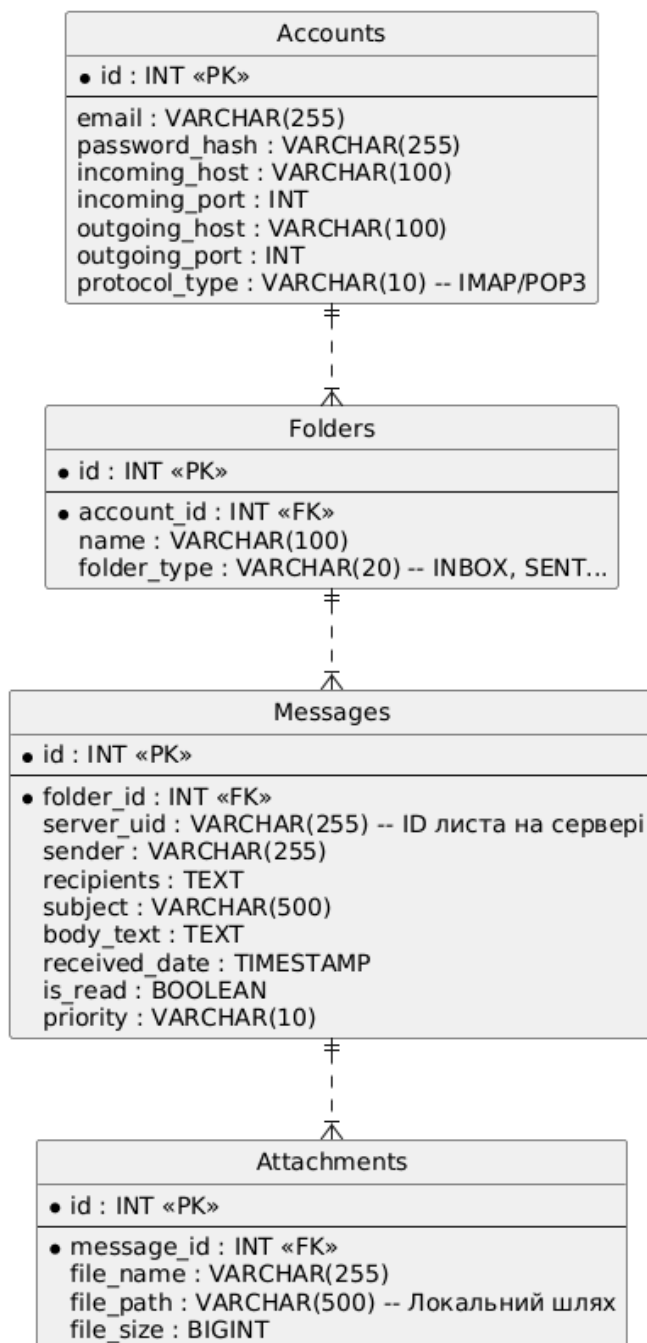
**IMAPService – IncomingProtocol Реалізація.** Клас IMAPService реалізує контракт (методи), визначений інтерфейсом IncomingProtocol.



**EmailAccount – ConfigFactory Залежність.** EmailAccount тимчасово звертається до ConfigFactory для отримання об'єкта конфігурації під час ініціалізації.

**ConfigFactory – ServerConfig Залежність (Створення).** ConfigFactory створює та повертає нові екземпляри класу ServerConfig.

### Реалізація БД



**Висновок:** я навчився будувати діаграми варіантів використання для системи що проектується, розробляти сценарії варіантів використання та будувати діаграми класів предметної області.