Міністерство освіти і науки України
Чернівецький національний університет імені Юрія Федьковича

Інститут фізико-технічних та комп'ютерних наук
Кафедра програмного забезпечення комп'ютерних систем

ЗВІТ

про виконання лабораторної роботи №7
з курсу "Безпека програм та даних"

Тема: Підсистема керування доступом

Виконали: Неголюк О.О., Ратушняк М.А.
Перевірив: Остапов С.Е.

Чернівці – 2025

ЗМІСТ

МЕТА РОБОТИ

Метою лабораторної роботи є розроблення та впровадження підсистеми керування доступом у веб-застосунку HearMyPaper на основі моделі Bell-LaPadula з додатковими обмеженнями цілісності. Робота спрямована на забезпечення конфіденційності даних користувачів системи шляхом впровадження багаторівневої політики безпеки з використанням рівнів конфіденційності та цілісності для суб’єктів та об’єктів доступу.

ОПИС ПРОЕКТУ ТА ПРОГРАМИ

2.1. Загальна характеристика системи

HearMyPaper – це веб-застосунок для безпечної роботи з курсовими проектами студентів, що забезпечує конфіденційність та контрольований доступ до робіт. Система дозволяє студентам завантажувати зашифровані файли курсових проектів, а викладачам – отримувати доступ до них лише за наявності відповідних прав доступу. Особливістю системи є можливість автоматичного перетворення текстових робіт в аудіоформат для зручного прослуховування.

2.2. Архітектура системи

Проект складається з двох основних компонентів:

Серверна частина реалізована на мові Python з використанням фреймворку FastAPI. Використовуються наступні технології та бібліотеки: SQLAlchemy для роботи з базою даних PostgreSQL, Pydantic для валідації даних, PyJWT для роботи з токенами автентифікації, Cryptography для криптографічних операцій.

Клієнтська частина розробляється на мові Kotlin з використанням фреймворку BeeWare для створення кросплатформного застосунку, що працює на настільних та мобільних системах.

Система використовує протокол HTTPS для захищеної передачі даних, а також реалізує механізм автентифікації на основі асиметричної криптографії з використанням публічних та приватних ключів.

2.3. Модель безпеки

У системі реалізовано модель безпеки Bell-LaPadula з розширеннями для забезпечення цілісності. Модель включає чотири рівні конфіденційності:

- UNCLASSIFIED (незасекречений) – рівень 1
- CONTROLLED (контрольований) – рівень 2
- RESTRICTED (обмежений) – рівень 3
- CONFIDENTIAL (конфіденційний) – рівень 4

Кожен користувач (суб’єкт) має рівень конфіденційності confidentiality_level та список рівнів цілісності integrity_levels, які визначають його права доступу до об’єктів системи (проекти, файли). Система забезпечує виконання правил “no read up” (заборона читання об’єктів вищого рівня конфіденційності) та “no write down” (заборона запису об’єктів нижчого рівня конфіденційності).

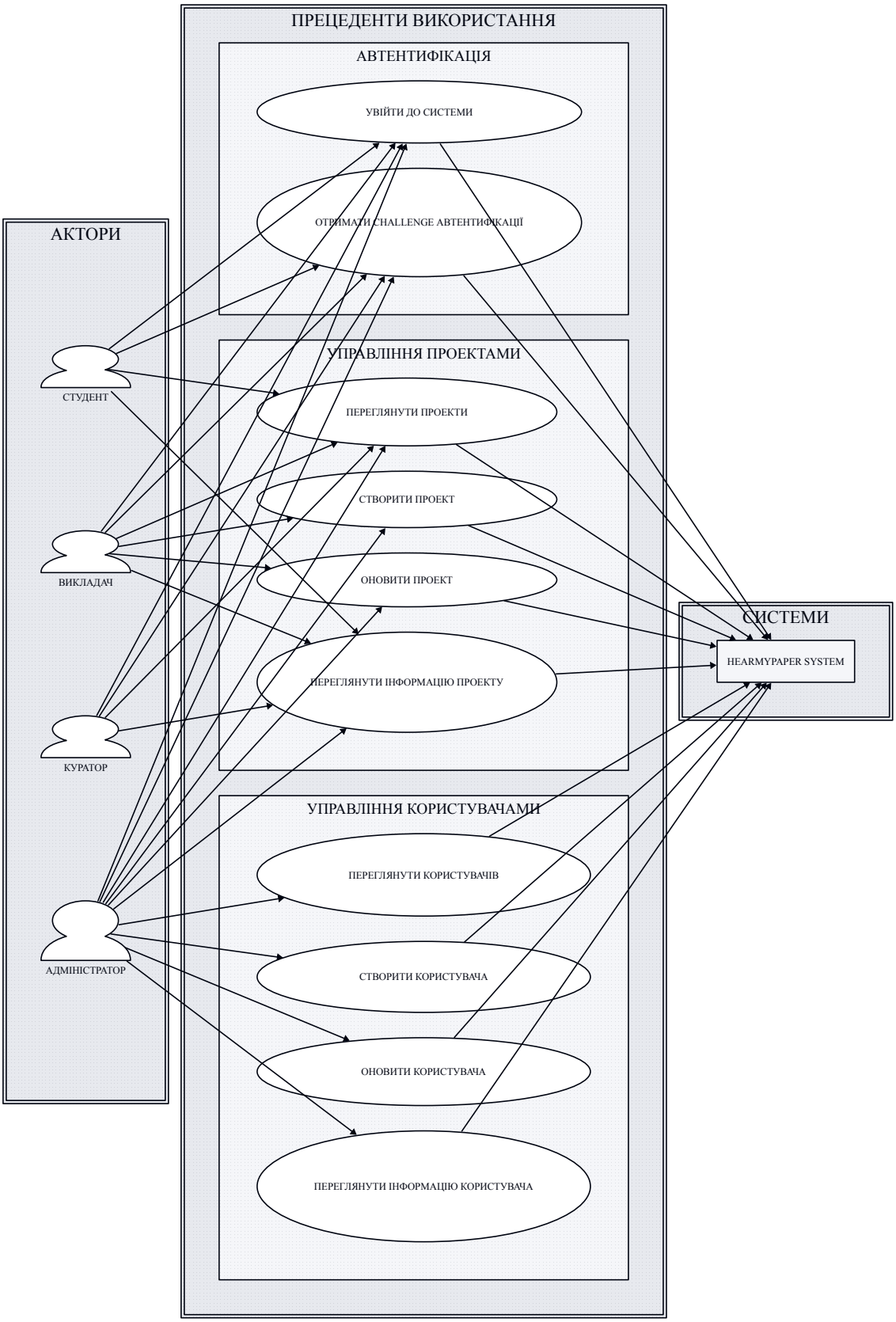

Рисунок 2.1 – Діаграма прецедентів системи HearMyPaper

На діаграмі прецедентів (Рисунок 2.1) показано основні випадки використання системи: студенти можуть завантажувати зашифровані роботи, викладачі – отримувати доступ до них, читати або конвертувати у аудіоформат, а також переглядати журнал аудиту дій у системі.

ПРОТОКОЛ РОБОТИ

3.1. Підготовка середовища розробки

Початковим етапом виконання лабораторної роботи було налаштування середовища розробки. Проект використовує систему контролю версій Git та розміщений у репозиторії на GitHub. Для роботи з Python використовувалася версія 3.13 з віртуальним оточенням, керованим інструментом uv. Серверна частина запускається у Docker-контейнерах з використанням docker-compose для оркестрації сервісів PostgreSQL, pgAdmin та основного застосунку FastAPI.

Структура проекту організована наступним чином: кореневий каталог містить підкаталоги server та client для серверної та клієнтської частин відповідно.

3.2. Проектування моделі безпеки

На етапі проектування було розроблено модель безпеки на основі Bell-LaPadula. Спочатку було визначено атрибути безпеки, де створено перелічення AccessLevel з чотирма рівнями конфіденційності та AccessType з прапорцями READ і WRITE для типів доступу.

Далі було створено базову структуру системи безпеки, що включала створення модулів auth для автентифікації, shared для загальних компонентів, та user для керування користувачами. Тоді ж було впроваджено базові утиліти для роботи з базою даних та залежностями FastAPI.

3.3. Реалізація автентифікації користувачів

Система автентифікації була реалізована у кілька етапів. Спочатку створено таблицю users у базі даних з полями для зберігання інформації про користувачів. Потім додано підтримку публічних ключів користувачів для асиметричної автентифікації.

Було реалізовано механізм challenge-response автентифікації. Процес автентифікації відбувається наступним чином: клієнт запитує challenge від сервера, підписує його своїм приватним ключем, сервер перевіряє підпис за допомогою збереженого публічного ключа користувача, і у разі успіху видає JWT токен з інформацією про суб’єкта.

У модулі auth/service.py реалізовано функції create_login_challenge та login_user, які обробляють процес автентифікації. Функція перевіряє, чи не закінчився термін дії облікового запису користувача (поле expires_at), валідує підпис та генерує токен з інформацією про рівні доступу користувача.

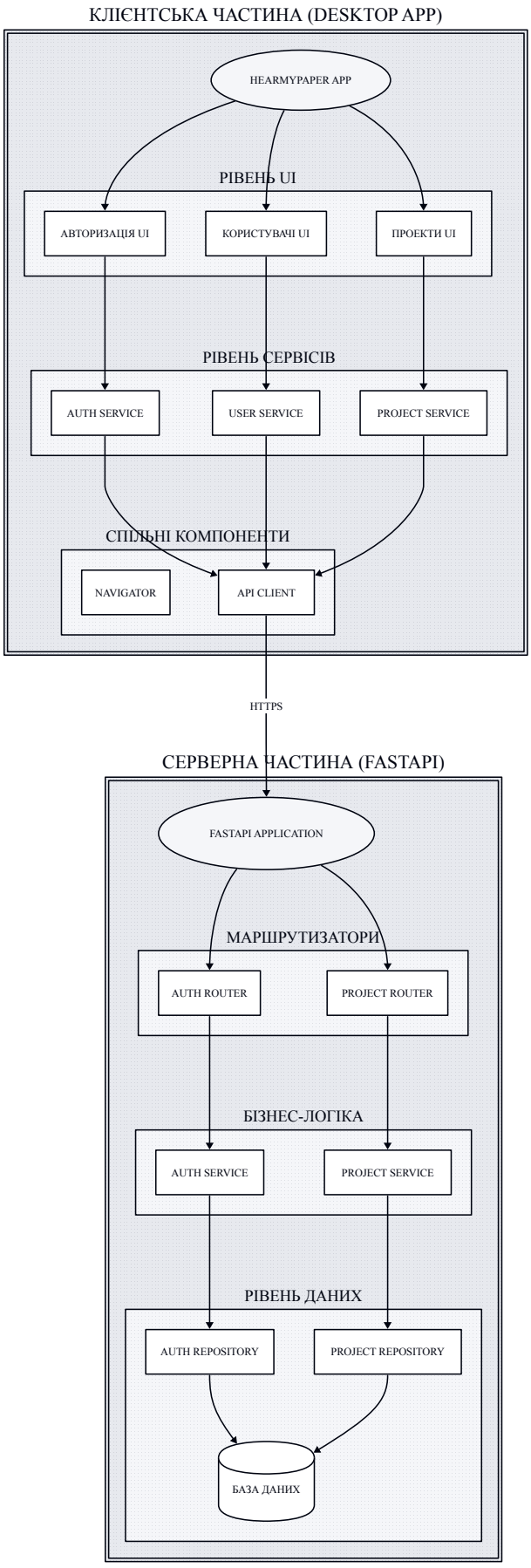

Рисунок 3.1 – Data Flow діаграма системи HearMyPaper

На діаграмі потоків даних (Рисунок 3.1) показано взаємодію компонентів системи під час автентифікації та роботи з проектами. Клієнтський застосунок взаємодіє з API сервером через HTTPS, сервер звертається до бази даних PostgreSQL та використовує JWT для передачі інформації про автентифікованого користувача.

3.4. Впровадження моделі Bell-LaPadula

Ключовим етапом роботи було впровадження моделі Bell-LaPadula з обмеженнями цілісності. Було проведено міграцію бази даних, яка перейменувала поле access_level на confidentiality_level та замінила access_rules на integrity_levels для зберігання масиву рівнів цілісності.

У модулі auth/service.py реалізовано функцію authorize_subject, яка перевіряє права доступу суб'єкта до об'єкта. Функція реалізує три основні правила:

Правило "no read up" забороняє суб'єкту читати об'єкти з вищим рівнем конфіденційності. Якщо рівень конфіденційності суб'єкта менший за рівень об'єкта, доступ на читання відхиляється з помилкою HTTP 403.

Правило "no write down" забороняє суб'єкту записувати об'єкти з нижчим рівнем конфіденційності. Якщо рівень конфіденційності суб'єкта вищий за рівень об'єкта, доступ на запис відхиляється.

Додаткова перевірка цілісності дозволяє запис лише на ті рівні, які явно вказані у списку integrity_levels суб'єкта. Це забезпечує додатковий контроль над тим, які об'єкти може створювати або модифікувати користувач.

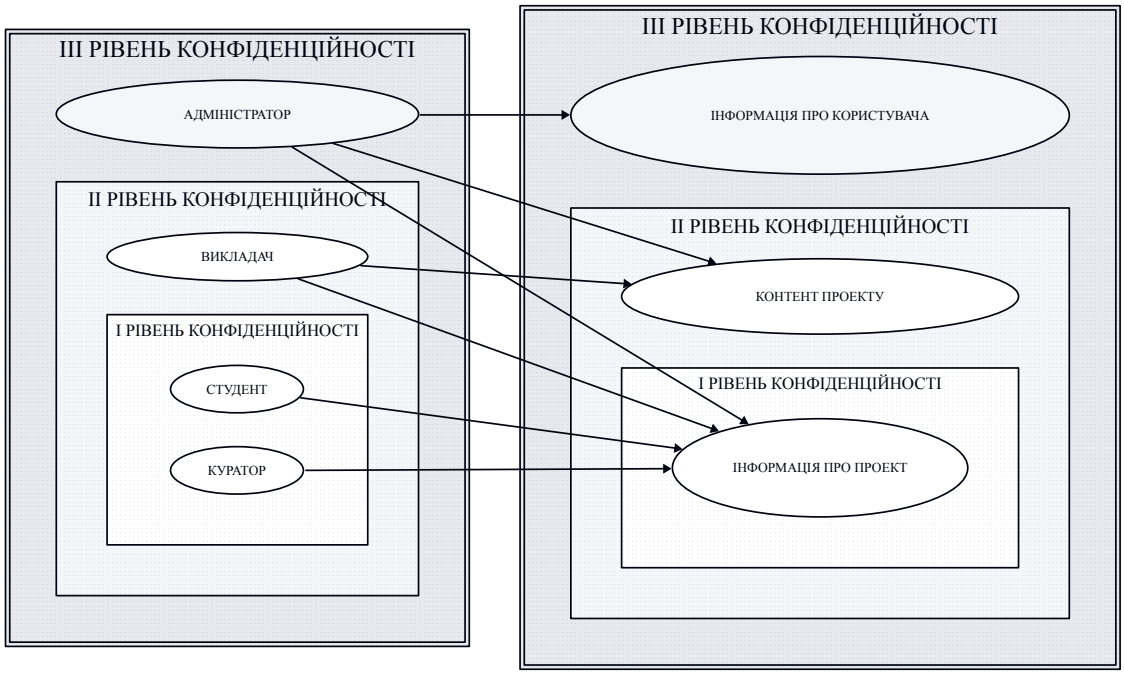

Рисунок 3.2 – Діаграма політики доступу для “читання” об’єктів

Діаграма на Рисунок 3.2 ілюструє правила доступу до об’єктів для операції читання. Суб’єкт може читати об’єкти свого рівня або нижчих рівнів конфіденційності, але не може читати об’єкти вищих рівнів. Наприклад, користувач з рівнем RESTRICTED може читати об’єкти RESTRICTED, CONTROLLED та UNCLASSIFIED, але не може читати CONFIDENTIAL.

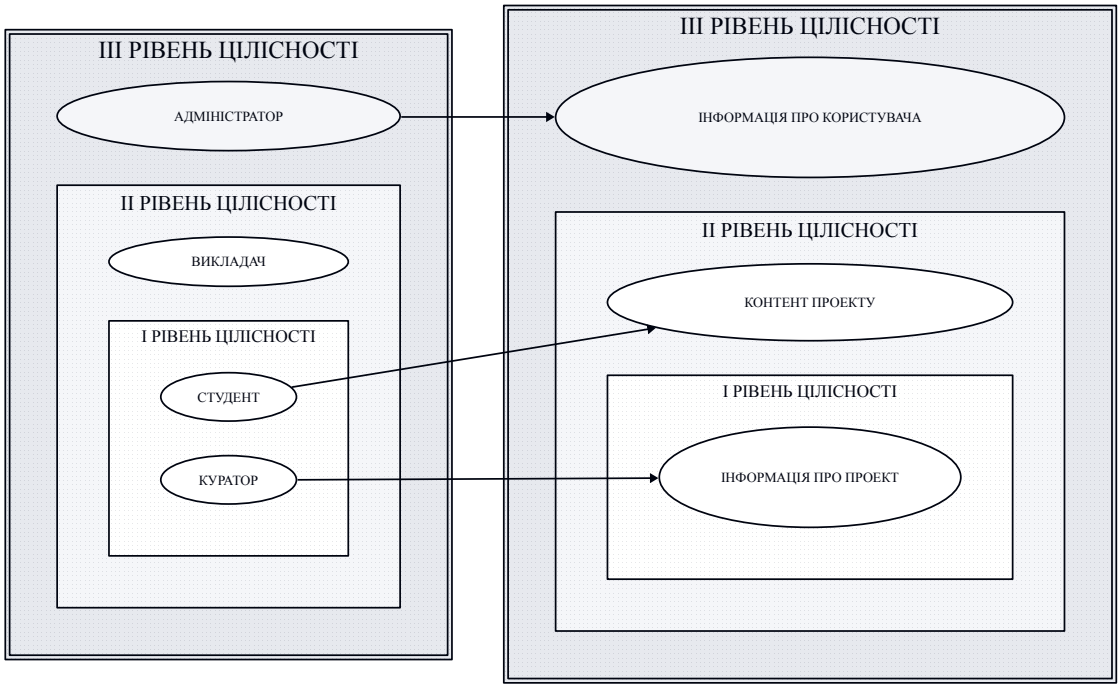

Рисунок 3.3 – Діаграма політики доступу для “створення” об’єктів

Діаграма на Рисунок 3.3 демонструє правила доступу для операції запису. Суб’єкт може записувати об’єкти лише свого рівня конфіденційності або вищих рівнів, за умови що ці рівні присутні у його списку integrity_levels. Це запобігає витоку інформації з вищих рівнів на нижчі.

3.5. Створення декораторів авторизації

Для спрощення використання підсистеми керування доступом було розроблено декоратори. У модулі auth/decorators.py реалізовано декоратор authorize, який автоматично перевіряє права доступу суб’єкта перед виконанням функції.

Декоратор інтегрується з FastAPI та використовує dependency injection для отримання інформації про автентифікованого користувача.

3.6. Розширення системи сутністю проектів

На завершальному етапі було додано підтримку проектів студентів. Створено міграцію бази даних, яка визначає таблицю projects для зберігання інформації про проекти.

Кожен проект має instructor_id – ідентифікатор викладача, який створив завдання, та deadline – термін здачі роботи.

Реалізовано API endpoints у модулі project/router.py для створення, читання та оновлення проектів з використанням декораторів авторизації. При доступі до проекту система автоматично перевіряє, чи має користувач достатній рівень доступу для виконання операції.

3.7. Налаштування адміністративного доступу

Для керування системою було створено механізм створення адміністраторів. У каталозі server/scripts додано скрипт для створення адміністративних облікових записів з найвищими рівнями доступу.

Міграція бази даних оновила модель користувача, додавши поля для адміністративного доступу. Це дозволяє окремим користувачам мати розширені права для керування іншими користувачами та налаштуваннями системи.

3.8. Тестування та валідація

Протягом розробки проводилося тестування функціональності системи безпеки. Використовувалися інструменти mypy для статичної перевірки типів

Python коду, ruff для лінтингу та форматування, pre-commit для автоматизації перевірок перед комітами.

Було перевірено коректність роботи правил Bell-LaPadula шляхом спроб доступу з різними рівнями конфіденційності. Підтверджено, що система правильно блокує несанкціоновані спроби читання та запису, повертаючи відповідні коди помилок HTTP 403 з детальними повідомленнями про причину відмови в доступі.

3.9. Документування системи

Паралельно з розробкою велася робота над документацією проекту. Створено діаграми у форматі D2, які візуалізують архітектуру системи: use-case.d2 для діаграми прецедентів, data-flow.d2 для потоків даних, read-access.d2 та write-access.d2 для правил доступу.

Діаграми були експортовані у формат SVG та інтегровані у звіт, написаний на мові розмітки Typst. Структура документації організована у каталозі docs/lab-reports з окремими підкаталогами для кожної лабораторної роботи.

ВІДПОВІДІ НА КОНТРОЛЬНІ ЗАПИТАННЯ

1. Проаналізуйте недоліки довірчого управління доступом.

Довірче управління доступом (Discretionary Access Control, DAC) має суттєві недоліки, особливо у контексті систем з високими вимогами до конфіденційності. Основною проблемою є можливість власника об’єкта самостійно змінювати права доступу, що може призвести до неконтрольованого розповсюдження конфіденційної інформації. Користувач з високим рівнем доступу може створити об’єкт з нижчим рівнем конфіденційності та передати до нього доступ іншим користувачам, що призведе до витоку інформації.

2. Оцініть позитивні та негативні сторони реалізованої політики безпеки.

Позитивні сторони реалізованої політики безпеки на основі Bell-LaPadula включають строге гарантування конфіденційності інформації через правила “no read up” та “no write down”, що унеможливлює несанкціонований доступ до інформації вищих рівнів та запобігає витоку на нижчі рівні. Модель є математично обгрунтованою та перевіреною у багатьох системах критичної інфраструктури. Додаткові обмеження цілісності через integrity_levels забезпечують додатковий контроль над тим, які об’єкти може створювати користувач.

Однак існують і негативні сторони. Модель може бути надто жорсткою для деяких сценаріїв використання, оскільки викладач з високим рівнем доступу не може створювати об’єкти для студентів з нижчим рівнем без додаткових механізмів. Відсутність підтримки зміни рівня конфіденційності об’єктів

після створення може ускладнювати деякі робочі процеси. Система потребує ретельного планування ієрархії рівнів доступу на етапі проектування системи.

Також одною із проблем є відсутність поділу користувачів на ролі, як от у рольових моделях контролю доступу. У мандатній ж системі контролю доступу кожен об’єкт визначається тільки рівнем доступу. Як наслідок у розробленій системі зв’язки між сутностями A та B предметної області у вигляді $A \rightarrow B$, при чому A виступає суб’єктом, а B – об’єктом, не можуть обмежити множину B за певним типом. Тож відповідальність за забезпечень даних обмежень лежатиме на користувачеві, що створює подібний зв’язок. До прикладу, викладач нестиме відповідальність за те, що “викладачем” створюваного курсу буде дійсно акаунт викладача, ане студент чи адміністратор.

3. Опишіть на рівні структур даних, як у Вашій роботі реалізовано матрицю доступу.

Матриця доступу реалізована неявним чином через атрибути безпеки суб’єктів та об’єктів. Кожен суб’єкт (користувач) представлений класом Subject у модулі auth/models.py з атрибутами: id (ідентифікатор), confidentiality_level (рівень конфіденційності типу AccessLevel), integrity_levels (список рівнів цілісності типу list[AccessLevel]). Ці атрибути зберігаються у таблиці users бази даних PostgreSQL, де confidentiality_level зберігається як ціле число, а integrity_levels як масив INTEGER[].

4. Охарактеризуйте рівень контролю доступу в реалізованій системі згідно НД ТЗІ 2.5-004-99. Що можна зробити, щоб його підвищити?

Оскільки реалізована система використовує модель Белла-Ла Падула, яка зосереджена на забезпеченні конфіденційності, то в якості критеріїв оцінки братимемо саме критерії конфіденційності зі стандарту.

За критеріями *довірчої кофіденційності* наша система підлягає усім вимогам щонайменш рівня **КД-2** (базової довірчої конфіденційності):

- Політика довірчої конфіденційності, що реалізується КЗЗ, повинна відноситись до всіх об’єктів КС (вимога рівнів КД-3, КД-4);
- КЗЗ повинен здійснювати розмежування доступу на підставі атрибутів доступу;
- Запити на зміну прав доступу до об’єкта повинні оброблятися КЗЗ на підставі атрибутів доступу користувача, що ініціює запит, і об’єкта;
- КЗЗ повинен надавати користувачу можливість для кожного захищеного об’єкта, що належить його домену, визначити конкретних користувачів і/або групи користувачів, які мають право одержувати інформацію від об’єкта;
- КЗЗ повинен надавати користувачу можливість для кожного процесу, що належить його домену, визначити конкретних користувачів і/або групи користувачів, які мають право ініціювати процес;
- Права доступу до кожного захищеного об’єкта повинні встановлюватись в момент його створення або ініціалізації. Як частина політики довірчої конфіденційності повинні бути представлені правила збереження атрибутів доступу об’єктів під час їх експорту та імпорту.

За критеріями *адміністративної кофіденційності* наша система підлягає усім щонайменш вимогам рівня **КА-2** (базової адміністративної кофіденційності):

- Політика адміністративної конфіденційності, що реалізується КЗЗ, повинна відноситись до всіх об’єктів КС (вимога рівнів КА-3, КА-4);
- КЗЗ повинен здійснювати розмежування доступу на підставі атрибутів доступу користувача і захищеного об’єкта;

- Запити на зміну прав доступу повинні оброблятися КЗЗ тільки в тому випадку, якщо вони надходять від адміністраторів або від користувачів, яким надані відповідні повноваження;
- КЗЗ повинен надавати можливість адміністратору або користувачу, що має відповідні повноваження, для кожного захищеного об’єкта шляхом керування належністю користувачів, процесів і об’єктів до відповідних доменів визначити конкретних користувачів і/або групи користувачів, які мають право одержувати інформацію від об’єкта;
- КЗЗ повинен надавати можливість адміністратору або користувачу, що має відповідні повноваження, для кожного процесу через керування належністю користувачів і процесів до відповідних доменів визначити конкретних користувачів і/або групи користувачів, які мають право ініціювати процес;
- Права доступу до кожного захищеного об’єкта повинні встановлюватися в момент його створення або ініціалізації. Як частина політики адміністративної конфіденційності мають бути представлені правила збереження атрибутів доступу об’єктів під час їх експорту та імпорту.

За критеріями *кофіденційності при обміні* наша система підлягає усім вимогам рівня **КВ-1** (мінімальної кофіденційності при обміні):

- Політика конфіденційності при обміні, що реалізується КЗЗ, повинна визначати множину об’єктів і інтерфейсних процесів, до яких вона відноситься;
- Політика конфіденційності при обміні, що реалізується КЗЗ, повинна визначати рівень захищеності, який забезпечується механізмами, що використовуються, і спроможність користувачів і/або процесів керувати рівнем захищеності;

• КЗЗ повинен забезпечувати захист від безпосереднього ознайомлення з інформацією, що міститься в об’єкті, який передається.

Для покрашення рівня конфіденційності розробленої системи, можна надати адміністратору можливість вибору конкретних рівнів доступу (за станадартом: “груп користувачів”), суб’єкт із яким **не матиме** права доступу до об’єкту.

5. Проаналізуйте взаємодію підсистеми управління доступом автентифікації в розробленій системі

Підсистеми управління доступом та автентифікації в розробленій системі тісно інтегровані та забезпечують комплексний захист. Процес взаємодії відбувається наступним чином.

Автентифікація є першим етапом та відбувається через challenge-response механізм з асиметричною криптографією. Клієнт відправляє запит на отримання challenge з вказанням user_id, сервер генерує випадкове значення та відправляє клієнту, клієнт підписує challenge своїм приватним ключем та відправляє підпис разом з challenge назад серверу. Сервер перевіряє підпис, використовуючи публічний ключ користувача, збережений у базі даних. Якщо підпис валідний та обліковий запис не прострочений (перевіряється поле expires_at), сервер генерує JWT токен.

Токен містить інформацію про суб’єкта: ідентифікатор користувача, рівень конфіденційності confidentiality_level, список рівнів цілісності integrity_levels. Ця інформація кодується у токені та підписується секретним ключем сервера.

Управління доступом використовує інформацію з токена для прийняття рішень про надання доступу. При кожному запиті до захищеного ресурсу сервер декодує JWT токен та відновлює об’єкт Subject. Декоратор authorize або функція

authorize_subject отримує інформацію про суб’єкта з токена та об’єкт доступу з бази даних, виконує перевірку правил Bell-LaPadula, приймає рішення про дозвіл або відмову в доступі.

Така архітектура забезпечує безстановий механізм авторизації, оскільки вся необхідна інформація міститься у токені, та підвищує безпеку завдяки відокремленню процесів автентифікації та авторизації. Токен має обмежений час життя, що зменшує ризики у разі його компрометації.

ВИСНОВКИ

У ході виконання лабораторної роботи було успішно розроблено та впроваджено підсистему керування доступом для застосунку HearMyPaper на основі мандатної моделі безпеки Bell-LaPadula з додатковими обмеженнями цілісності.

Реалізована система забезпечує чотирирівневу ієрархію конфіденційності (UNCLASSIFIED, CONTROLLED, RESTRICTED, CONFIDENTIAL) та строге дотримання правил “no read up” і “no write down”, що гарантує захист від несанкціонованого доступу до інформації та запобігає витоку конфіденційних даних на нижчі рівні безпеки. Додатково впроваджено механізм рівнів цілісності integrity_levels, який обмежує можливості користувачів щодо створення та модифікації об’єктів лише тими рівнями, на які вони мають явний дозвіл.

Автентифікація користувачів реалізована на основі асиметричної криптографії з використанням механізму challenge-response, що забезпечує високий рівень безпеки без необхідності передачі паролів по мережі. Інформація про рівні доступу користувача передається через JWT токени, що дозволяє створити безстановий механізм авторизації та спрощує подальше масштабування системи.

Архітектура підсистеми керування доступом спроектована з урахуванням принципів модульності та розширюваності. Використання декораторів авторизації дозволяє легко застосовувати перевірки прав доступу до API endpoints, а централізована функція authorize_subject забезпечує єдину точку контролю для всіх рішень про доступ.

У процесі роботи набуто практичних навичок проектування та реалізації систем безпеки для веб-застосунків, роботи з криптографічними бібліотеками Python, проектування баз даних з урахуванням вимог безпеки, створення RESTful API з використанням FastAPI та впровадження автоматизованих перевірок коду за допомогою інструментів статичного аналізу.

Розроблена підсистема керування доступом може бути використана як основа для подальшого розвитку системи HearMyPaper, зокрема для впровадження функціональності аудиту дій користувачів.