РІВНЕНСЬКИЙ ДЕРЖАВНИЙ ГУМАНІТАРНИЙ УНІВЕРСИТЕТ
Факультет математики та інформатики
Кафедра цифрових технологій та методики навчання інформатики

«До захисту допущено»
Завідувач кафедри
_____ Наталія ПАВЛОВА
«_____»_____2025р.
протокол № __

АНТОНЮК АНТОН

КВАЛІФІКАЦІЙНА РОБОТА
на тему:

РОЗРОБКА ТА ВПРОВАДЖЕННЯ МЕТОДИКИ ВИКОРИСТАННЯ АВТОМАТИЗОВАНИХ СИСТЕМ ПЕРЕВІРКИ ЗАВДАНЬ З ІНФОРМАЦІЙНОЇ БЕЗПЕКИ У НАВЧАЛЬНОМУ ПРОЦЕСІ ЗАКЛАДІВ ФАХОВОЇ ПЕРЕДВИЩОЇ ОСВІТИ

Виконав:
здобувач другого (магістерського) рівня вищої освіти спеціальності 015.39 Професійна освіта (Цифрові технології)
Антон АНТОНЮК

Науковий керівник:
кандидат юридичних наук, доцент
Павло КІНДРАТ

Рівне – 2025

АНОТАЦІЯ

АНТОНЮК Антон. Розробка засобів автоматизованої перевірки якості виконання завдань з інформаційної безпеки – Кваліфікаційна робота на здобуття другого (магістерського) рівня вищої освіти за спеціальністю 015.39 Професійна освіта (Цифрові технології). Рівненський державний гуманітарний університет. Рівне, 2025. 63 с.

Кваліфікаційна робота присвячена вирішенню актуальної науково-практичної проблеми забезпечення автоматизованої, систематичної та об’єктивної оцінки якості виконання практичних завдань з інформаційної безпеки в закладах передвищої освіти.

У роботі проаналізовано предметну область, доведено невідповідність традиційних методів контролю знань потребам ринку та обґрунтовано необхідність застосування кіберполігонів – ізольованих, реалістичних, віртуальних тренувальних середовищ, як єдиного легального способу формування практичних навичок. Визначено, що існуючі системи управління навчанням створюють технологічний бар’єр через примітивність перевірки, а спеціалізовані платформи не надають деталізованої, педагогічно ефективної оцінки методології роботи студента.

Обґрунтовано вибір технологічного стеку для розробки системи. Ключовим елементом визначено інтеграцію зі штучним інтелектом для виконання якісного аналізу методології виконання завдань. Спроєктовано архітектуру програмного забезпечення на основі трирівневої клієнт-серверної моделі. Сформовано детальні функціональні та нефункціональні вимоги до системи, що включають вимоги до ізоляції середовищ, швидкості розгортання та надання миттєвого зворотного зв’язку.

Розроблена система забезпечує систематичне та об’єктивне оцінювання, надаючи викладачам інструмент для ефективного управління навчальним процесом та підготовки висококваліфікованих фахівців з кібербезпеки.

Ключові слова: інформаційна безпека, навчальна платформа, кіберполігон, практична освіта.

ANNOTATION

ANTONIUK Anton. Development and implementation of methodology of using automated information security task verification systems in the educational process of higher vocational education institutions – Qualification work for obtaining the second (Master's) level of higher education in specialty 015.39 Professional Education (Digital Technologies). Rivne State Humanitarian University. Rivne, 2025. 63 p.

The qualification work is dedicated to solving the urgent scientific and practical problem of ensuring an automated, systematic, and objective assessment of the quality of practical information security tasks execution in higher education institutions.

It is determined that existing LMS (Learning Management Systems) create a technological barrier due to primitive verification, and specialized platforms do not provide a detailed, pedagogically effective assessment of the student's work methodology.

The choice of the technology stack for system development is justified. Integration with artificial intelligence is defined as a key element for performing a qualitative analysis of the task execution methodology.

The software architecture is designed based on a three-tier client-server model. Detailed functional and non-functional requirements for the system are formulated, including requirements for environment isolation, deployment speed, and providing immediate feedback.

The developed system ensures systematic and objective evaluation, providing instructors with a tool for effective management of the educational process and the training of highly qualified cybersecurity specialists.

Keywords: Information security, educational platform, cyber range, practical education.

ЗМІСТ

ВСТУП

В епоху глобалізації та стрімкої цифровізації захищеність інформаційних ресурсів набуває статусу критично важливої складової успішного розвитку суспільства. Зростання кількості та складності інформації неминуче призводить до збільшення кількості та витонченості загроз. Для України, яка перебуває в умовах війни та протистоїть складним і комплексним деструктивним інформаційним впливам, пріоритетним завданням є не лише захист власної критичної інфраструктури, але й системна підготовка висококваліфікованих фахівців з кібербезпеки, здатних ефективно протистояти цим загрозам.

Європейська Комісія відзначає загальносвітовий дефіцит кібербезпекових кадрів (European Union Agency for Cybersecurity, 2022). Успішна інтеграція України до європейського ринку праці є стратегічним напрямком, що прямо залежить від якості та відповідності європейським стандартам підготовки громадян.

Роль закладів вищої освіти є визначальною, адже саме вони формують фундаментальні знання, професійні компетентності та світогляд майбутніх фахівців у сфері кібербезпеки. Сучасна система вищої освіти має не лише відповідати актуальним вимогам ринку праці, але й випереджати їх, готуючи студентів до роботи в умовах постійно змінюваного цифрового середовища. Тож варто звернути особливу увагу на оновлення освітніх програм, впровадження практикоорієнтованих методів навчання, використання сучасних технологій симуляції кібератак, систем моніторингу та аналітики.

Забезпечення національної стійкості в цифровому просторі вимагає переосмислення підходів до професійної підготовки. Якщо раніше акцент в освіті з кібербезпеки міг зміщуватися на теоретичні основи та нормативні документи, то сьогодні домінуючою вимогою є здатність до практичного протистояння та

проактивного захисту. Критична інфраструктура, державні реєстри та комерційні активи потребують фахівців, які вміють оперативно виявляти, аналізувати та експлуатувати вразливості, застосовувати підходи DevSecOps та керувати процесами відновлення систем.

Це підвищення вимог висуває фундаментальний виклик перед освітніх закладів: як об’єктивно оцінити ці комплексні, практико-орієнтовані навички? Успіх освітньої програми у сфері кібербезпеки визначається її здатністю подолати розрив між теорією та реальністю кіберпростору. Відповіддю на цей виклик є створення ізольованих, реалістичних, віртуальних тренувальних середовищ – кіберполігонів. Однак, впровадження самого полігону є лише першим кроком. Нагальною проблемою залишається розробка механізмів систематичної, масштабованої та якісної перевірки результатів роботи студентів у цих середовищах.

Метою дослідження є розробка та імплементація системи автоматизованої перевірки якості виконання практичних завдань з інформаційної безпеки для застосування в освітньому процесі закладів передвищої освіти.

Завдання дослідження:

– провести огляд та аналіз предметної області, визначити ключові вимоги до системи оцінювання практичних навичок фахівців з кібербезпеки;

– здійснити аналіз існуючих освітніх та спеціалізованих платформ (LMS, кіберполігони) для виявлення їхніх обмежень у контексті об’єктивного оцінювання процесу виконання завдань;

– обрати оптимальний стек технологій для розробки системи, що забезпечить масштабованість, гнучкість та інтеграцію функціоналу штучного інтелекту;

– спроектувати архітектуру програмного забезпечення, що забезпечить ізольоване середовище для кожного студента та механізм деталізованого контролю;

– розробити ключові програмні модулі системи, включаючи механізми взаємодії з кіберполігоном та модуль автоматизованої перевірки якості на основі API штучного інтелекту;

– запропонувати методику використання розробленої системи у навчальному процесі.

Об’єктом дослідження є процес автоматизації оцінювання практичних навичок та знань здобувачів освіти з інформаційної безпеки.

Предметом дослідження є методи, засоби та технології автоматизованої перевірки якості виконання практичних завдань з інформаційної безпеки.

Методами дослідження є комплекс теоретичних та емпіричних методів. Теоретичні методи використовувались для вивчення предметної області, виявлення обмежень існуючих платформ та обґрунтування вибору технологічного стеку. Емпіричні методи використовувались для розробки архітектурних рішень, формування вимог та специфікації логіки взаємодії ключових модулів системи.

Наукова новизна отриманих результатів отримала розвитку концепція застосування систем на основі штучного інтелекту для автоматизації перевірки та оцінювання рівня виконання завдань з інформаційної безпеки.

Практичне значення дослідження полягає у розробці технологічного рішення, яке забезпечує пряме впровадження новітніх методів оцінювання в освітній процес, критично важливий для підготовки фахівців з інформаційної безпеки.

Апробація результатів роботи. Основні теоретичні та практичні результати дослідження доповідалися та обговорювалися на IV Всеукраїнській науково-практичній конференції «Підготовка педагогів до професійної діяльності в умовах

змішаного навчання» (м. Рівне. 14-15 травня 2025) та XVIII Всеукраїнській науково-практичній конференції «Інформаційні технології в професійній діяльності» (м. Рівне, 10 листопада 2025 р.), а також були опубліковані у науковому журналі «Актуальні питання у сучасній науці» (№12(42), 2025, 10 с.).

Розроблена система автоматизації перевірки знань та методика її застосування були впроваджені у навчальний процес Голосіївського економіко-правового фахового коледжу (Акт впровадження №272 від 18.12.2025).

Структура роботи. Дипломна робота складається зі вступу, трьох розділів, висновків, переліку використаних джерел та 2 додатків. Загальний обсяг роботи становить 62 сторінки. Вона містить 6 рисунків та 2 таблиці. Список використаних джерел включає 50 найменувань. Обсяг додатків – 6 сторінок.

РОЗДІЛ 1.
ОГЛЯД ПРЕДМЕТНОЇ ОБЛАСТІ ДОСЛІДЖЕННЯ

1.1 Засади підготовки фахівців у закладах фахової передвищої освіти.

Розвиток інформаційних технологій та їх все глибше проникнення у повсякденне життя супроводжується постійним зростанням кількості кіберінцедентів як технічного так і антропогенного характеру. Це, в свою чергу, призводить до зростання ролі спеціалістів з кібербезпеки у забезпеченні коректного та безперервного функціонування інформаційних систем які слугують підґрунтям для функціонування не лише для підприємств чи організацій, а й суспільства в цілому. Для забезпечення стрімко зростаючого попиту на спеціалістів у сфері інформаційної безпеки збільшується і кількість навчальних закладів які забезпечують підготовку фахівців відповідного профілю.

На сьогодні в Україні підготовка спеціалістів зі спеціальності F5 «Кібербезпека та захист інформації» здійснюється на 3 рівнях: фахового молодшого бакалавра, бакалавра та магістра. Кожен з цих рівнів передбачає досягнення визначених рівнів кваліфікації спеціаліста та, відповідно виховання певних якостей та напрацювання умінь і навичок. Специфіка вивчення зазначеної спеціальності у закладах фахової передвищої освіти суттєво відрізняється від університетської програми. Головний акцент тут робиться не на глибокій теорії чи проектуванні складних архітектур, а на практичних навичках адміністрування, моніторингу та технічної підтримки інформаційних систем.

Для університетської підготовки притаманним є приділення значної уваги теоретичним засадам функціонування систем захисту інформації та методикам розробки інноваційних технологій. Натомість заклади фахової передвищої освіти передбачають, в першу чергу, прикладне, інструментально орієнтоване навчання. Зокрема це обумовлює фокус на роботі з конкретним програмним забезпеченням, підбором та налаштуванням апаратних компонент обчислювальних систем,

прокладанням та налаштуванням мереж, управлінням інформаційною безпекою об’єктів що підлягають захисту тощо. По суті, навчання покликане відповісти на питання: «що слід робити якщо виникла певна ситуація?».

Зважаючи, що більшість студентів закладів фахової передвищої освіти вступають до них після дев’ятого класу, це створює на початковому етапі додаткове навантаження. Адже за перші два роки студенти мають як виконувати шкільну програму для отримання сертифікату про завершену повну середню освіту, так і здійснювати перші кроки по вступу у свою майбутню спеціальність. Водночас, рання спеціалізація та професійна орієнтація дозволяє вже на початкових етапах розпочати формувати специфічний тип мислення та критичний погляд на інформаційні системи. Що робить студентів більш сприйнятними для вивчення профільних дисциплін та застосування отриманих знань у практичній площині.

Програми закладів фахової передвищої освіти передбачають підготовку фахівця на рівні технік-адміністратор, та забезпечують досягнення ним спеціальних фахових компетентностей. При цьому б’єктом вивчення можуть виступати автоматизовані, телекомунікаційні, інформаційні, інформаційно-аналітичні, інформаційно-телекомунікаційні системи, інформаційні ресурси і технології, технології забезпечення безпеки інформації, процеси управління інформаційною та/або кібербезпекою об’єктів, що підлягають захисту (ОПП «Кібербезпека та захист інформації», 2025)

Зважаючи на вимоги спеціальності, специфіку роботи закладів фахової передвищої освіти, а також вікові особливості їх студентів, в них часто застосовуються специфічні методи викладання, які покликані підвищити зацікавленість студентів до навчання із застосуванням обмежених ресурсів. До них відносяться:

1. віртуалізація навчального середовища – полягає у широкому використанні віртуальних машин для безпечного проведення лабораторних робіт та практичних

занять по взаємодії зі шкідливим програмним забезпеченням та протидії атакам без ризику для основної мережі навчального закладу.

2. гейміфікація навчання – полягає у залученні студентів до змагань у форматі Capture The Flag на початковому рівні, що суттєво підвищує мотивацію та розуміння принципів функціонування інформаційних систем та необхідності дотримання вимог щодо їх захисту.

3. проектна робота – полягає у залучені студентів до проектування захищених мереж та систем під час виконання курсових та дипломних робіт. Це дозволяє залучати студентів до розвитку навчальної та мережевої інфраструктури навчального закладу шляхом подальшої інтеграції в неї найкращих проектів.

4. етично-правововове виховання студентів – воно полягає у необхідності формування чіткого морально-етичного кодексу поведінки, розумінні різниці між правомірним і неправомірним використанням отриманих знань, усвідомлені професійної відповідальності.

В сучасних реаліях розвиток спеціальності F5 «Кібербезпека та захист інформації» у закладах фахової передвищої освіти зіштовхується з багатьма викликами, починаючи від застарілості освітньо-професійних програм і закінчуючи застарілістю наявного програмно-апаратного забезпечення. І якщо перший чинник не можливо усунути у зв’язку з динамічністю розвитку технологій та методологій у сфері інформаційної безпеки, в той час як освітні програми приймаються з розрахунку на 3-4 роки навчання. То на другий можна вплинути не лише шляхом закупівлі дорого вартісного обладнання, яке теж має тенденцію д швидкого застарівання, а й шляхом створення, впровадження та подальшого розвитку інноваційних систем та програмних комплексів які покликані посприяти підвищенню якості та зручності управління освітнім процесом і перевірці отриманих знань та навичок.

1.2 Особливості оцінювання знань з інформаційної безпеки

Організація освітнього процесу в закладах фахової передвищої освіти мало відрізняється від такої в інших навчальних закладах, хоч і володіє притаманними їм особливостями. Ключовим завданням залишається необхідність належної верифікації досягнення студентом цільових компетентностей по своїй спеціальності. Це породжує потребу у забезпеченні належної перевірки рівня засвоєння отриманих знань та оцінюванні якості виконання завдань які притаманні для інформаційної безпеки.

Традиційні методи навчання та контролю знань (усні відповіді, тестові перевірки) у закладах освіти не можуть повноцінно оцінити здатність студента до комплексних дій у реальних, стресових умовах кіберзагрози. Сучасний фахівець повинен мати практичні навички у таких сферах, як безпека застосунків, аудит вихідного коду, DevSecOps, тестування на проникнення та управління доступом. Фахівець має вміти здійснювати оцінювання можливості несанкціонованого доступу та вирішувати задачі управління процесами відновлення штатного функціонування систем (Puchko O., & Uvarkina O., 2023).

Ключова проблема традиційного оцінювання полягає у його статичності та відірваності від контексту. Тести перевіряють знання фактів, а не здатність до застосування цих знань у агресивній мережі. У реальному світі кібератаки ніколи не обмежуються одним питанням із чотирма варіантами відповіді; вони є ланцюгом послідовних дій (Kill Chain), що вимагають креативного мислення та не шаблонних рішень (Сус В., 2024). Отже, компетентність фахівця з кібербезпеки не може бути адекватно виміряна без:

1) стресового фактора: здатність працювати під тиском часу, коли критична інфраструктура перебуває під загрозою.

2) комплексності системи: взаємодія з багатокомпонентними мережами, які імітують корпоративну чи промислову інфраструктуру, а не лише ізольовані віртуальні машини.

3) етичної безпеки: потреба відпрацьовувати методи експлуатації вразливостей, які є незаконними на живих мережах.

Оцінити здатність до виконання «пентестів» або до застосування теорій захисту в умовах реалізації загроз різних класів можливо лише у спеціалізованих симуляційних середовищах. Оскільки освітні заклади, як публічні заклади, не можуть використовувати власні робочі мережі для відпрацювання атак та експлуатації вразливостей , виникає абсолютна необхідність застосування кіберполігонів (Cyber Ranges) – ізольованих, реалістичних, віртуальних тренувальних середовищ (Василенко В., Грицкевич Г., & Кузнецов І., 2025). кіберполігони є єдиним легальним та безпечним способом сформувати та оцінити ці практичні навички.

Для забезпечення ефективності та об’єктивності такого практико-орієнтованого навчання викладачам необхідні зручні та ефективні механізми перевірки. Контроль має бути максимально індивідуалізованим та систематичним, з чіткими та зрозумілими критеріями оцінювання. Чим досконалішим та об’єктивнішим є контроль, тим успішніше буде здійснюватися управління навчальним процесом, та в цілому позитивно впливатиме на ефективність навчання. Це вимагає від викладачів використання нових форм і методів практичної перевірки знань, зокрема ситуаційних вправ та ділових ігор, які дозволяють оцінювати готовність до застосування знань у критичних ситуаціях.

У контексті стратегічної необхідності системної та якісної підготовки кібербезпекових кадрів, виникає назріла потреба в автоматизації процесів оцінювання. Автоматизація перевірки якості завдань з інформаційної безпеки перетворюється з бажаного інструменту на критично важливий елемент сучасної освітньої інфраструктури.

В результаті можна винести кілька ключових проблем:

1) масштабованість та систематичність: забезпечення можливості одночасної перевірки практичних навичок великої кількості студентів та інтеграції практичних занять у модульну структуру курсую.

2) педагогічна ефективність: надання студентам миттєвого та конструктивного зворотного зв’язку про процес виконання завдання, а не лише про кінцевий результат.

1.3 Огляд систем управління навчальним процесом

Аналіз предметної області виявив критичну необхідність застосування кіберполігонів для формування та об’єктивної оцінки практичних навичок фахівців з кібербезпеки. Однак, назріла потреба в автоматизації процесів оцінювання вимагає критичного огляду наявних інструментів, що можуть бути інтегровані або використані як основа для розробки нової системи. На цьому етапі дослідження проводиться порівняльний аналіз існуючих на ринку рішень, щоб чітко визначити їхні переваги, обмеження та, головне, технологічний розрив, який необхідно подолати.

1.2.1. Огляд Google Classroom

Google Classroom – це безкоштовний вебсервіс, розроблений компанією Google спеціально для навчальних закладів. Його основна мета – це спростити організацію навчального процесу, створення, поширення та оцінювання завдань у безпаперовий спосіб.

Серед основного функціональну даної програми можна виділити наступні можливості:

1) створення віртуальних класів та запрошення студентів за кодом або електронною поштою.

2) публікації лекцій, презентацій, відео, посилань та інших ресурсів для студентів.

3) створення завдань, встановлення термінів здачі, отримання виконаних робіт від учнів в електронному вигляді.

4) перевірка робіт, виставлення оцінок та моніторинг успішності кожного учня в електронному журналі.

5) обмін повідомленнями, організація обговорень, надання індивідуальних коментарів та спілкування між викладачем і учнями.

6) інтеграція з Google Meet, що дозволяє проводити відеозустрічі та живі уроки.

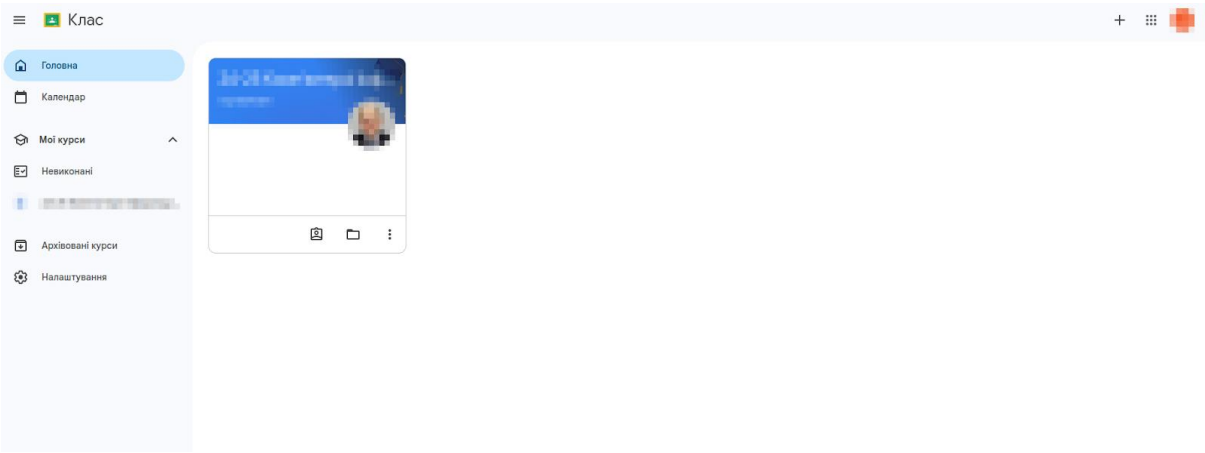


Рис. 1.1. Графічний вигляд веб-застосунку «Google Classroom»

1.2.2. Огляд Moodle

Moodle (Modular Object-Oriented Dynamic Learning Environment) – це лідер серед Систем Управління Навчанням (LMS), відомий своєю гнучкістю та масштабованістю. На відміну від Google Classroom, який є простішим вебсервісом, Moodle – це повноцінна платформа з відкритим вихідним кодом.

Серед переваг Moodle можна виділити наступний функціонал:

1) Створення повноцінного освітнього простору для студентів, школярів, корпоративних клієнтів (підвищення кваліфікації).

2) Розробка навчальних курсів із чіткою структурою, розділами, уроками та різноманітними інтерактивними елементами.

3) Централізоване розміщення та організація всіх навчальних матеріалів (тексти, відео, презентації, посилання).

Системи управління навчанням (LMS), такі як Moodle та Google Classroom, є універсальними інструментами, призначеними для часткової або повноцінної організації освітнього процесу в середніх і вищих навчальних закладах. Їхня основна функція полягає у забезпеченні створення, управління та розповсюдження навчальних матеріалів, відстеження прогресу студентів, автоматизації адміністративних процесів та організації дистанційного навчання.

LMS ефективно підтримують загальні форми контролю:

1) Тестування: зручні механізми для створення тестів з множинним вибором, відповідністю, які оцінюються автоматично.

2) Письмові роботи та реферати: можливість завантаження документів з подальшою ручною перевіркою викладачем.

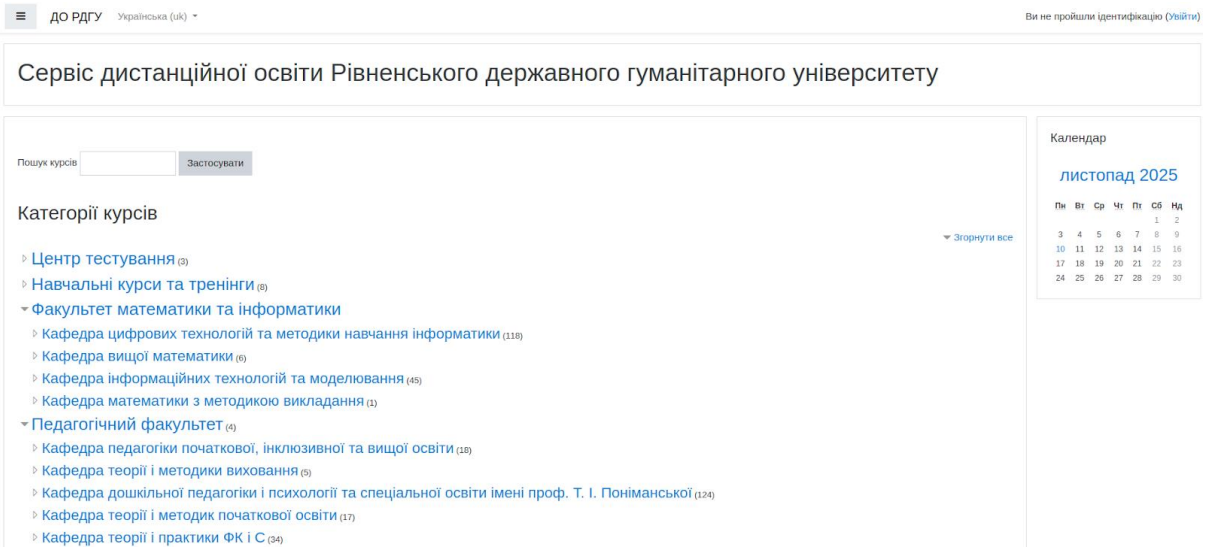


Рис. 1.2. Графічний вигляд веб-застосунку «Moodle»

1.2.3. Обмеження загальних систем управління навчанням

Ключова проблема вищезгаданих платформ – це відсутність автоматизованої перевірки якості виконання будь яких завдань, зокрема з інформаційної безпеки.

Ці платформи є технологічно-агностичними до завдань з інформаційної безпеки. Їхній функціонал не прив’язаний до специфічних вимог практичної ІБ-освіти, тому вони створюють технологічний бар’єр для об’єктивного оцінювання комплексних практичних навичок.

Їхні обмеження полягають у наступному:

1) Емуляція реалістичного середовища: LMS не можуть створювати ізольовані віртуальні мережі, які імітують корпоративну інфраструктуру, що є необхідним для практичного відпрацювання Pentest, Blue Teaming або реагування на інциденти.

2) Автоматизована перевірка якості виконання завдань: у загальних LMS автоматизована перевірка обмежується лише бінарним результатом (правильно/неправильно) або відповіддю типу «прапор» (Flag).

Існуючі LMS, такі як Moodle, є необхідними для адміністративного управління курсами, але вони створюють технологічний бар’єр для об’єктивного оцінювання комплексних практичних навичок з кібербезпеки (Поліщук Н. В., Бугаєнко Т. І., & Лемешева Н. В., 2024). Це унеможливлює повноцінне впровадження принципу контролю та корекції знань, оскільки викладач не отримує якісної інформації про методологію роботи студента.

Висновки до розділу 1

Проведено аналіз предметної області, який засвідчує невідповідність традиційних методів навчання та контролю знань потребам сучасної кібербезпеки. Сучасний фахівець має володіти не лише теоретичними знаннями, але й

практичними навичками у DevSecOps, тестуванні на проникнення (PenTest) та аудиті вихідного коду.

Проведений огляд існуючих освітніх та практичних платформ виявляє суттєвий розрив між адміністративними потребами закладів фахової передвищої освіти та специфічними вимогами практичної підготовки кібербезпекових кадрів.

Системи управління навчанням (LMS), такі як Moodle та Google Classroom, є незамінними для адміністративного управління курсами та підтримки загальних форм контролю (тести, письмові роботи). Однак, вони створюють технологічний бар’єр для якісної практичної ІБ-освіти через відсутність симуляції реального середовища та примітивність автоматизованої перевірки, яка обмежується лише бінарним результатом (правильно/неправильно) або введенням «прапора» (Flag). Вони не можуть оцінити методологію роботи студента, що унеможливлює повноцінний, об’єктивний контроль та корекцію знань.

РОЗДІЛ 2.
Вимоги до системи контролю знань з інформаційної безпеки

2.1 Спеціалізовані платформи для навчання інформаційної безпеки

Жодна з розглянутих в розділі 1 платформ управління навчальним процесом сама по собі не задовольняє комплексну потребу закладів освіти які готують спеціалістів напрямку «Кібербезпека ті захисті інформації». Це обумовлює необхідність створення або адаптації спеціалізованого рішення для кіберполігонів з потужним автоматизованим механізмом перевірки якості практичних завдань.

2.1.1. Огляд HackThebox

Hack The Box (HTB) – це онлайн-платформа та віртуальна лабораторія, розроблена для навчання та вдосконалення практичних навичок у галузі кібербезпеки (Information Security), зокрема, пентестингу (penetration testing) – тестування на проникнення.

На відміну від навчальних LMS (Moodle) чи платформ для управління завданнями (Google Classroom), HTB є ігровим, практично-орієнтованим середовищем для «етичних хакерів» та фахівців з інформаційної безпеки.

Hack The Box надає користувачам доступ до великої кількості ізольованих віртуальних машин, мережевих вузлів і завдань, що імітують реальні системи та вразливості, які можна знайти у світі.

Будучи платформою для навчання інформаційної безпеки, HTB надає студентам наступні можливості:

1) Створення безпечного, законного середовища для відпрацювання всіх етапів пентестингу: розвідка, сканування, експлуатація вразливостей, підвищення привілеїв та закріплення в системі.

2) Постійне вдосконалення своїх знань та навичок з кіберзахисту та кібернападу.

3) Об’єднання професіоналів, дослідників, студентів і ентузіастів у світову спільноту, де вони можуть обмінюватися знаннями та брати участь у змаганнях (CTF – Capture The Flag).

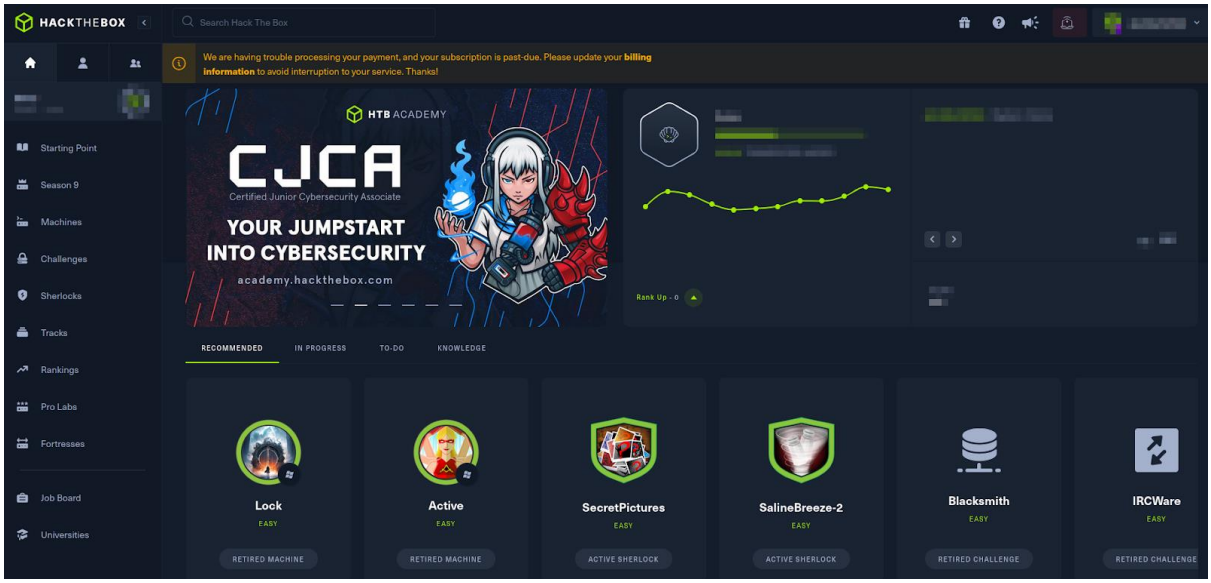


Рис. 2.1. Графічний вигляд веб-застосунку «HackTheBox»

2.1.2. Огляд TryHackMe

TryHackMe (THM) – це інтерактивна онлайн-платформа, спеціально розроблена для навчання кібербезпеці та пентестингу. Її основна відмінність від Hack The Box (HTB) полягає в тому, що THM орієнтована не лише на досвідчених професіоналів, але й, в першу чергу, на абсолютних новачків, пропонуючи структурований, покроковий навчальний процес.

TryHackMe використовує методику гейміфікації (Gamification), перетворюючи складне навчання на захоплюючу гру. Платформа пропонує модулі (так звані «кімнати» або «paths»), де навчання відбувається через читання теорії та негайне виконання практичних завдань у віртуальній машині, вбудованій прямо у браузер.

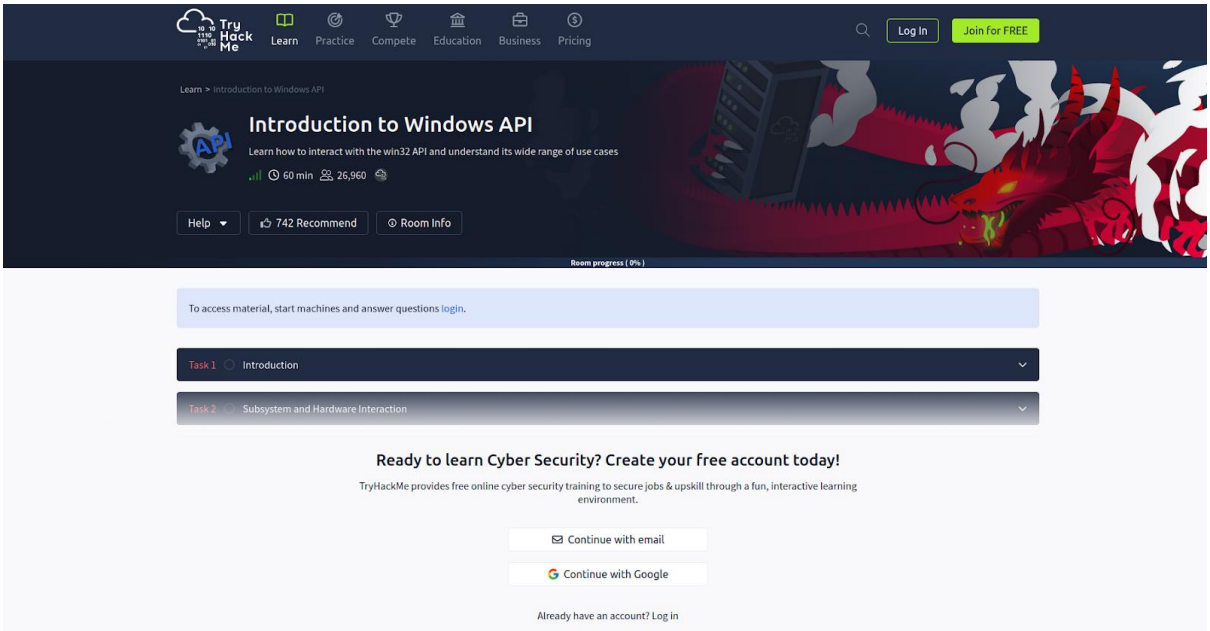


Рис. 2.2. Графічний вигляд веб-застосунку «TryHackMe»

2.2 Методичні передумови для впровадження системи автоматичної перевірки знань з інформаційної безпеки

Ґрунтуючись на проведеному вище аналізі існуючих систем управління навчальним середовищем та контролю знань студентів, а також з урахуванням наведених міркувань можна стверджувати що система перевірки знань з інформаційної безпеки вимагатиме балансу між перевіркою теоретичних знань та здатністю студентів виконати практичні завдання. При цьому, зважаючи що фахова передвища освіта орієнтована, в першу чергу, на підготовку технічних спеціалістів (практиків), система оцінювання їх знань має бути зміщена в бік демонстрації практичних навичок.

Система перевірки знань має базуватися на принципі «навчання через дію». Тож контроль не повинен бути відокремленим від навчального середовища. Він має поєднувати автоматизовані тести та практичних кейсів виконання яких відбувається у віртуальному середовищі.

Будь-яка перевірка практичних навичок має відбуватись виключно в ізольованому віртуальному середовищі, щоб не зашкодити реальному мережевому середовищі навчального закладу. Самі завдання мають моделювати типові робочі ситуації.

Для забезпечення об’єктивності та повноти перевірки знань доцільно використовувати трирівневу архітектуру системи оцінювання:

- модуль теоретичного контролю – для нього можуть бути використані існуючі платформи оцінювання знань чи супроводу навчального середовища (Moodle, Google Classroom, Canvas тощо) з різнотиповими питаннями. В цьому модулі не варто обмежуватись виключно питаннями типу «одна вірна відповідь». Водночас не забуваючи про градієнт складності питань і вагу кожного питання;

- модуль практичного тренінгу – для нього можуть бути використані як віртуальні машини так і власні розробки для автоматизованої перевірки якості виконання практичних завдань. Останні можуть застосовувати скрипти для перевірки налаштовуваної конфігурації системи цільовим показникам;

- змагальний модуль – для нього може застосовуватись концепція CTF (Capture The Flag). Цей модуль можна застосовувати для проведення підсумкового контролю або атестації, адже він дозволяє найкраще оцінити нешаблонне мислення, стресостійкість та відповідність ключовим компетентностям спеціальності.

В якості форми подачі матеріалу для перевірки та оцінювання доцільно використовувати звіти, які за структурою відповідають звітам про інциденти та їх вирішення. Для забезпечення достовірності в звіт має бути включена вимога доказовості, яка може буде реалізована як шляхом прикріплення відповідного скріншоту з часовою міткою, так і шляхом додаванням скрипта який використовувався для вирішення завдання.

Сама ж система перевірки має включати різного роду запобіжники. До таких можуть бути віднесені: унікальні завдання (кожен студент отримує свій варіант IP-

адресації або унікальний «ключ» (прапор), який генерується динамічно), захист звіту (скріншоти мають містити унікальні ідентифікатори).

Система автоматизованої перевірки також має включати в себе підсистему зворотнього зв’язку. Вона повинна видавати коментар при помилці який описуватиме суть помилки. Це потрібно щоб контроль також виконував навчальну функцію.

2.3 Обгрунтування технологій для реалізації застосунку

Для розроблення системи автоматизованої перевірки якості виконання завдань з інформаційної безпеки було використано наступні технології python, sqlite3, gemini, terraform, flask, docker. Далі розглянуто технічні характеристики перерахованих технологій.

2.3.1. Мова програмування Python

Python є однією з найбільш популярних мов програмування для завдань машинного навчання та аналізу даних, завдяки своїм численним перевагам, які роблять його ідеальним вибором для розробки складних систем, які потребують інтеграції багатьох компонентів несхожої структури (Кобильник Т., Когут У., & Жидик В., 2021).

Python має надзвичайно багату екосистему бібліотек та інструментів для роботи зі штучним інтелектом і машинним навчанням.

Важливою перевагою Python є його простота і зручність у використанні. Синтаксис мови є інтуїтивно зрозумілим і легко читається, що сприяє швидкій розробці та прототипуванню. Це дозволяє інженерам і дослідникам швидко створювати і тестувати систему.

Python дозволяє почати з невеликого проекту і поступово розширювати його функціональність та обсяги обробки даних без потреби зміни мови програмування або архітектури системи. Така гнучкість є ключовою для проектів, які можуть з часом змінюватися і розвиватися.

Підсумовуючи, використання Python для побудови застосунку для автоматизації перевірки якості вирішення завдань з інформаційної безпеки є виправданим і оптимальним вибором.

2.3.2. База даних SQLite3

SQLite3 є програмною бібліотекою, що реалізує безсерверну систему управління реляційними базами даних з нульовою конфігурацією.

Ключова особливість SQLite полягає в її безсерверній архітектурі: вона не вимагає окремого процесу сервера для роботи, на відміну від традиційних СУБД, таких як PostgreSQL або MySQL (Піскунов О., Єсик Л. В., & Томащук О. П., 2022). Уся база даних зберігається як єдиний стандартний файл на диску. Це значно спрощує розгортання та адміністрування системи, оскільки зникає необхідність у встановленні, налаштуванні прав доступу чи запуску будь-яких мережевих служб. Фактично, бібліотека SQLite3 просто компонується із застосунком, що забезпечує нульову конфігурацію для розробника.

Для розробки на Python особливо важливо, що модуль sqlite3 входить до стандартної бібліотеки мови. Це означає, що для роботи з базою даних не потрібна установка додаткових сторонніх пакетів, що спрощує керування залежностями та значно підвищує переносимість проекту.

SQLite є ідеальним вибором для локальних, однокористувацьких додатків, мобільних пристроїв, вбудованих систем, а також може використовуватися як сховище конфігурацій або кешу в настільних чи веб-додатках.

2.3.3. Веб-фреймворк Flask

Flask – це популярний мікрофреймворк для веб-розробки, написаний на Python. Він реалізує принцип мінімалізму, надаючи лише ключові функції, необхідні для створення веб-застосунків: механізми маршрутизації (обробка URL-адрес та спрямування запитів до відповідного коду), керування HTTP-запитами та рендеринг шаблонів.

Вибір Flask для розробки системи ґрунтується на його легкості та високій гнучкості. Фреймворк не диктує розробнику жорстких правил щодо архітектури проекту, вибору системи баз даних, інструментів для роботи з даними (ORM) чи механізмів автентифікації. Ця особливість дозволяє розробнику вільно інтегрувати будь-які сторонні бібліотеки та компоненти, що найкраще відповідають унікальним вимогам конкретного проекту. Така архітектурна свобода особливо цінується при створенні вузькоспеціалізованих RESTful API або невеликих, незалежних мікросервісів, що є основою багатьох сучасних розподілених систем.

Завдяки своїй простоті та низькому порогу входження, Flask дозволяє дуже швидко розпочати розробку; працездатний веб-застосунок можна запустити мінімальною кількістю коду. Це прискорює процес розробки прототипів та основної функціональності. Хоча Flask є мінімалістичним за своєю суттю, його функціональність легко розширюється через велику екосистему додаткових розширень (Extensions), які додають підтримку форм, автентифікації, управління базами даних та інші необхідні можливості, забезпечуючи при цьому контроль над кожним компонентом системи.

У підсумку, Flask є потужним, гнучким і простим у використанні веб-фреймворком, який ідеально підходить для побудови застосунку lдля перевірки якості вирішення завдань з інформаційної безпеки.

2.3.4. Бібліотека Gemini

Бібліотека Google Gemini (або її клієнтська бібліотека для Python) є основним програмним інтерфейсом для взаємодії з передовими генеративними моделями штучного інтелекту від Google, зокрема сімейством моделей Gemini (Yanev M., 2025). Її використання є критично важливим для даного проєкту, оскільки саме за допомогою можливостей цього штучного інтелекту буде виконуватись автоматизована, якісна перевірка виконання практичних завдань з інформаційної безпеки. На відміну від традиційних систем оцінювання, які обмежуються лише бінарним результатом або введенням «прапора» (Flag),

інтеграція моделі Gemini дозволяє аналізувати методологію роботи студента та надавати деталізований, якісний зворотний зв’язок.

Ця бібліотека дає змогу системі, розробленій на Flask та Python, програмно надсилати структуровані запити (наприклад, логи роботи студента, контекст завдання та очікувані кроки) до віддаленого API Gemini. У відповідь система отримує згенерований, аналітичний текст із порівнянням виконаних дій студента з оптимальним алгоритмом, оцінкою якості та рекомендаціями. Інтеграція є простою завдяки стандартним інструментам Python. Використання офіційного клієнта гарантує стабільну та актуальну взаємодію з новітніми функціями API.

Таким чином, бібліотека Gemini є технологічним містком, що забезпечує педагогічну ефективність системи, дозволяючи об’єктивно оцінювати не лише кінцевий результат, але й процес досягнення цього результату, що є необхідним для корекції знань у закладі освіти.

2.3.5. Програмний інструмент Terraform

Terraform – це потужний інструмент з відкритим вихідним кодом від компанії HashiCorp, що реалізує парадигму Infrastructure as Code (IaC), тобто «Інфраструктура як код». Його основне завдання полягає в тому, щоб дозволити розробникам та адміністраторам описувати, створювати та керувати хмарними ресурсами та іншою інфраструктурою за допомогою декларативних конфігураційних файлів, а не за допомогою ручних операцій через веб-інтерфейс або скрипти (Обозний Д. М., 2020). Цей підхід є абсолютно критичним для реалізації концепції кіберполігону у контексті даного проєкту. Terraform підтримує широкий спектр провайдерів, включаючи AWS, Google Cloud, Azure, а також спеціалізовані сервіси, як-от Docker або Kubernetes, що забезпечує високу універсальність розгорнутої системи.

Ключова перевага Terraform, що обумовлює його вибір, полягає у його унікальній здатності до швидкої та багаторазової реплікації ідентично налаштованих середовищ. Для системи автоматизованої перевірки, де кожен

студент повинен працювати в персоналізованому та ізольованому інстансі цільової мережі, Terraform гарантує, що це середовище буде створене за єдиним, перевіреним шаблоном. Це забезпечує відтворюваність та об'єктивність оцінювання, оскільки фактори, пов'язані з конфігураційними відмінностями, виключаються. Terraform автоматично керує життєвим циклом інфраструктури: від її створення (provisioning) до безпечного видалення (destroy), коли завдання завершено.

Таким чином, він вирішує проблему масштабованості для освітнього закладу, дозволяючи викладачам швидко розгортати необхідну кількість віртуальних лабораторій, які використовують обчислювальні потужності AWS, що робить практичну підготовку доступною для великих груп студентів.

2.3.6. Інструментарій Docker

Docker є провідною платформою для контейнеризації, що дозволяє розробляти, доставляти та запускати додатки в стандартизованих, ізольованих і повністю автономних пакетах, які включають увесь необхідний код, середовище виконання та конфігурацію (Щербина І. С., & Явор Д. В., 2023). Застосування Docker є критично важливим для реалізації концепції кіберполігону та забезпечення масштабованості системи перевірки. Він дозволяє створювати легковагові та ізольовані віртуальні середовища (цільові машини з вразливостями) для кожного практичного завдання, що гарантує, що дії одного студента не вплинуть на роботу інших або на хостову інфраструктуру закладу.

Завдяки контейнеризації досягається абсолютна відтворюваність тестового середовища на будь-якій платформі, що є запорукою об'єктивності оцінювання, оскільки всі студенти працюють з ідентичною конфігурацією.

Крім того, Docker значно спрощує процес розгортання як самої системи автоматизованої перевірки (забезпечуючи її портативність), так і цільових машин, ідеально доповнюючи функціонал Terraform. Контейнери є ефективнішими за повноцінні віртуальні машини з точки зору використання ресурсів, що дозволяє

одночасно запускати більшу кількість ізольованих інстансів, забезпечуючи високу масштабованість навчального процесу.

2.3.7. Мова стилів CSS

CSS (Cascading Style Sheets) – це мова стилів, яка використовується для опису зовнішнього вигляду, представлення та форматування документа, написаного мовою розмітки, такою як HTML. Якщо HTML створює структуру веб-застосунку, то CSS надає йому візуальну привабливість, читабельність та інтуїтивність (Січко Т. В., & Яцковська Р. О., 2020). У розробці системи автоматизованої перевірки CSS відіграє вирішальну роль у забезпеченні педагогічної ефективності та зручності користування. Він використовується для визначення шрифтів, кольорів, макету сторінки та адаптивності, гарантуючи, що деталізовані звіти про перевірку якості від AI будуть представлені в структурованому, зрозумілому для викладача та студента вигляді. Наприклад, CSS дозволяє виділяти кольором критичні помилки, індивідуальні коментарі та загальну оцінку, що є ключовим для швидкого та ефективного зворотного зв’язку.

Його застосування, часто у поєднанні з бібліотекою Bootstrap забезпечує, що інтерфейс системи буде адаптивним і коректно відображатиметься на різних пристроях, від настільних комп’ютерів до планшетів, підтримуючи таким чином масштабованість навчального процесу.

2.3.8. Мова розмітки HTML

HTML (HyperText Markup Language) є фундаментальною та невід’ємною основою для створення будь-якого веб-застосунку, виступаючи у ролі архітектурного каркасу, який визначає логічну структуру та зміст веб-сторінок (Павленко Ю. С., 2022). У контексті розробки системи автоматизованої перевірки якості завдань з інформаційної безпеки, HTML виконує функцію базового інтерфейсу користувача, який забезпечує візуалізацію та взаємодію для обох ключових груп користувачів: студентів та викладачів. Саме за допомогою HTML створюється базовий скелет для всіх ключових елементів системи. Це включає в

себе формування форм подачі завдань, де студенти можуть вводити знайдені результати, наприклад, «прапори» (Flags), або завантажувати файли з логами та звітами про виконання пентесту. Кожен елемент введення, кнопки та поля позначення мають бути чітко структуровані мовою HTML, щоб забезпечити коректну обробку даних бекендом на Flask.

Крім того, HTML є вирішальним для організації інтерфейсу зворотного зв’язку. Система, що використовує AI (Gemini) для деталізованої оцінки якості роботи студента, генеруватиме великі обсяги структурованого аналітичного тексту. HTML-розмітка відповідає за те, щоб цей згенерований звіт був не просто суцільним текстом, а мав чітко визначені розділи, заголовки, абзаци та таблиці. Наприклад, HTML використовується для створення елементів, які відображають загальну оцінку, деталізований аналіз методології роботи студента, список виявлених помилок, а також індивідуальні рекомендації для корекції знань, що є критично важливим для підвищення педагогічної ефективності системи.

Структурна роль HTML також поширюється на навігацію та організацію контенту. Він визначає ієрархію інформації, створюючи навігаційні панелі, посилання для переходу між різними розділами системи (наприклад, «Мої курси», «Календар подій», «Результати перевірки», «Налаштування профілю») та модульні блоки для зручного відображення великої кількості навчальних матеріалів чи активних інстансів кіберполігону. Таким чином, HTML не лише надає контент, але й встановлює його логічні зв’язки, що є основою для гіпертекстової природи Інтернету. На рівні взаємодії з бекендом, мова HTML використовується Flask-ом (через механізми рендерингу шаблонів, наприклад, Jinja), що дозволяє динамічно генерувати вміст сторінки, вставляючи дані з бази SQLite3 або результати обробки від Gemini безпосередньо в структуру розмітки перед відправкою до браузера користувача.

Вибір HTML як основного засобу структури є універсальним та гарантує максимальну сумісність і доступність застосунку на всіх сучасних платформах і браузерах.

2.3.9. Шаблонізатор Bootstrap

Bootstrap є найпопулярнішим у світі, безкоштовним фронтенд-фреймворком з відкритим вихідним кодом, який є потужною колекцією попередньо розроблених шаблонів HTML, CSS та JavaScript. Він призначений для прискореної розробки адаптивних (responsive) та мобільно-орієнтованих веб-сайтів і застосунків. Його інтеграція у систему автоматизованої перевірки є стратегічним рішенням, що безпосередньо впливає на педагогічну ефективність та загальну масштабованість проєкту. Замість того, щоб витрачати значний час на ручне написання CSS-коду для базових елементів, розробники, які використовують Flask та Python, можуть застосовувати готові компоненти Bootstrap. Це включає стандартизовані стилі для навігаційних панелей, кнопок, модальних вікон, типографіки, форм та, що найважливіше, системи сітки (Grid System), яка забезпечує коректне та привабливе розташування елементів на сторінці (Шакуров Є. О., & Клокова К. В., 2022).

Використання Bootstrap гарантує, що інтерфейс системи є професійним, інтуїтивно зрозумілим і зручним для користувачів. Це критично важливо для складних інтерфейсів, які мають відображати деталізовані звіти про перевірку якості, згенеровані штучним інтелектом. Bootstrap дозволяє легко структурувати ці звіти у вигляді карток, панелей або акуратних таблиць, підвищуючи їхню читабельність і спрощуючи для викладача процес аналізу методології роботи студента.

Крім того, завдяки своїй філософії mobile-first, Bootstrap забезпечує ідеальну адаптивність: інтерфейс системи автоматично підлаштовується під розмір екрана користувача, чи то великий монітор, чи планшет, чи смартфон. Ця властивість напряму підтримує модульну структуру курсу та можливість систематичної перевірки практичних навичок, дозволяючи студентам і викладачам отримувати

доступ до функціоналу кіберполігону та результатів оцінювання з будь-якого пристрою, що є необхідною умовою для сучасної вищої освіти.

2.3.10. Хмарне рішення AWS

AWS (Amazon Web Services) є найбільш комплексною та широко використовуваною у світі хмарною платформою, що пропонує широкий спектр повнофункціональних сервісів, які охоплюють обчислювальні потужності, сховища, бази даних, мережі, аналітику та інструменти для розробки та машинного навчання (Шелестов А. Ю., & Колотій А. В., 2022). Вибір AWS як цільового хмарного провайдера для розгортання системи автоматизованої перевірки та кіберполігону є обґрунтованим стратегічним рішенням, що забезпечує необхідну масштабованість, надійність та ізоляцію. AWS надає потужну інфраструктуру, яка є необхідною для розгортання та оркестрації складних віртуальних тренувальних середовищ, що імітують реальні корпоративні мережі.

Зокрема, сервіси Amazon EC2 (Elastic Compute Cloud) використовуються для швидкого створення та ізоляції віртуальних машин для кожного студента. Це дозволяє реалізувати концепцію персоналізованого, ізольованого інстансу, визначеного єдиним конфігураційним кодом Terraform, що є ключовим для об’єктивного оцінювання практичних навичок. Крім того, такі сервіси, як AWS Lambda або Amazon S3, можуть бути задіяні для масштабованого розміщення функціоналу Flask та безпечного зберігання даних SQLite3 або звітів про виконання завдань відповідно. Глибока інтеграція AWS з інструментом Infrastructure as Code, Terraform, дозволяє описувати, створювати та керувати цими хмарними ресурсами за допомогою декларативних конфігураційних файлів, забезпечуючи швидку та багаторазову реплікацію ідентично налаштованих середовищ, що критично важливо для підтримки навчального процесу великої кількості студентів у закладах фахової передвищої освіти.

AWS також забезпечує високий рівень безпеки та доступності, що є необхідною умовою для розміщення критичної освітньої інфраструктури, яка використовує моделі штучного інтелекту для перевірки.

Висновки до розділу 2

Спеціалізовані платформи, орієнтовані на кібербезпеку (Hack The Box, TryHackMe), ефективно вирішують проблему практичних навичок, надаючи ізольовані, легальні віртуальні середовища для відпрацювання всіх етапів PenTest. Вони використовують гейміфікацію, що підвищує мотивацію студентів. TryHackMe є особливо цінним для новачків завдяки структурованим, покроковим навчальним «шляхам».

Ключова невирішена проблема полягає у відсутності системи, яка поєднує практичний простір (кіберполігон, як HTB/THM) із педагогічною ефективністю, а саме – автоматизованою, систематичною, деталізованою оцінкою якості виконання завдань. Ця система має надавати індивідуалізований та миттєвий зворотний зв’язок про процес роботи студента, а не лише про кінцевий результат, що є критично важливим для викладачів освітніх закладів.

Проведений аналіз та обґрунтування вибору технологічного стеку переконливо демонструють, що обрані засоби є оптимальними та взаємодоповнюючими для ефективної розробки системи автоматизованої перевірки якості виконання завдань з інформаційної безпеки. Цей комплексний вибір технологій безпосередньо спрямований на вирішення ключових проблем, визначених у першому розділі, а саме: забезпечення педагогічної ефективності, абсолютної ізоляції тренувальних середовищ, масштабованості навчального процесу та об’єктивності оцінювання.

Фундаментальне ядро системи побудоване на Python та мікрофреймворку Flask. Python забезпечує потужну, гнучку та аналітичну базу для обробки даних, необхідну для роботи зі складними алгоритмами, тоді як Flask надає необхідну

архітектурну гнучкість для створення вузькоспеціалізованого, легко розширюваного веб-застосунку та RESTful API, що є ідеальною структурою для інтеграції численних зовнішніх сервісів. Найбільш інноваційним та критично важливим елементом є інтеграція бібліотеки Gemini, яка перетворює систему з простого засобу перевірки бінарних відповідей типу «Flag» на інструмент деталізованої та якісної оцінки. Штучний інтелект дозволяє аналізувати методологію роботи студента та оперативно надавати індивідуалізований і конструктивний зворотний зв’язок, що є головною невирішеною проблемою існуючих систем управління навчанням.

Для реалізації практичного середовища використано потужну комбінацію AWS, Terraform та Docker. AWS виступає як надійна хмарна платформа, а Terraform використовується для опису, керування та швидкої реплікації ідентично налаштованих, повністю ізольованих інстансів кіберполігону. Це є фундаментальним для легального та безпечного відпрацювання практичних навичок. У свою чергу, Docker доповнює це, забезпечуючи легковагові та ізольовані контейнеризовані середовища для самих цільових вразливих машин, гарантуючи відтворюваність і ефективне використання ресурсів.

Нарешті, фронтенд, побудований на HTML, CSS та фреймворку Bootstrap, забезпечує високу адаптивність інтерфейсу. SQLite3 спрощує розгортання, оскільки не потребує окремого сервера баз даних, що ідеально підходить для мікросервісної архітектури. У сукупності цей технологічний вибір формує цілісну та потужну платформу, здатну подолати технологічний бар’єр у сучасній кібербезпековій освіті, створивши середовище, що відповідає стратегічним потребам закладу освіти.

РОЗДІЛ 3.
РОЗРОБКА ТА АНАЛІЗ ПРОГРАМНОГО ПРОДУКТУ

3.1 Вимоги до програмного продукту

3.1.1. Опис бізнес моделі

Метою розробки даного програмного продукту є забезпечення якісної, систематичної та об’єктивної оцінки практичних знань і навичок у галузі кібербезпеки здобувачів фахової передвищої освіти з ІТ спеціальностей шляхом надання ізольованих тренувальних середовищ та автоматизованого, деталізованого контролю виконання завдань. Система є цінною для освітнього та корпоративного секторів завдяки високому рівню автоматизації процесів розгортання інфраструктури, інтеграції зі штучним інтелектом для аналізу методології роботи , а також наданню викладачам чітких, зрозумілих критеріїв оцінювання.

Функціональні можливості системи передбачають автоматичне розгортання та керування інфраструктурою кіберполігону за допомогою Terraform на хмарній платформі AWS, що забезпечує ізольовані та реалістичні віртуальні тренувальні середовища для кожного студента. Для оцінки якості виконання завдань з інформаційної безпеки, включаючи PenTest та аудит коду , розроблена система, що використовує бібліотеку Gemini для програмного надсилання запитів до ШІ-моделей та отримання згенерованих відповідей. Це дозволяє здійснити деталізовану, якісну оцінку методології роботи студента, а не обмежуватися бінарним результатом («правильно/неправильно») або відповіддю типу «прапор» (Flag). Звіти на основі ШІ надають викладачам критично важливу інформацію про процес виконання завдання , що унеможливлює повноцінне впровадження принципу контролю та корекції знань.

Беручи до уваги усі очікування від методів використання системи у контексті закладу фахової передвищої освіти, ключовою бізнес-моделлю є SaaS (Software as a Service), що дозволяє надавати доступ до інфраструктури та сервісу автоматизованої перевірки за підпискою, що базується на кількості місць або інтенсивності використання інфраструктурних ресурсів (Бріжак В. Г., б. д.).

3.1.2. Опис архітектури програмного забезпечення

Програмне забезпечення системи автоматизованої перевірки якості виконання завдань з інформаційної безпеки реалізовано на основі трирівневої (триланкової) клієнт-серверної архітектури, що дозволяє досягти високої масштабованості, надійності та незалежності компонентів. Ця архітектура чітко розділяє логіку представлення, логіку застосунку та логіку даних і зовнішніх сервісів, що є найкращою практикою для розподілених систем, які використовують хмарні обчислення та штучний інтелект.

Перший рівень, рівень клієнта, відповідає за безпосередню взаємодію з користувачем, забезпечуючи інтуїтивно зрозумілий інтерфейс для Студентів та Викладачів. Цей рівень побудований на технологіях HTML та CSS, з використанням фреймворку Bootstrap для забезпечення повної адаптивності та професійного вигляду на всіх пристроях. Клієнт відповідає за відображення навчальних матеріалів, форм для подачі результатів роботи з кіберполігону, а також за структурну візуалізацію деталізованих звітів, згенерованих ШІ, підкреслюючи критичні оцінки та рекомендації.

Другий рівень, рівень сервера застосунку, є центральним хабом, що реалізує основну бізнес-логіку системи та координує роботу всіх зовнішніх сервісів. Він реалізований як мікросервіс на основі Python та фреймворку Flask. Сервер містить Менеджер завдань, який обробляє запити від клієнта та ініціює процеси оцінювання. Ключовими тут є два модулі: Модуль Інфраструктурного Керування, який взаємодіє з Terraform для ініціації розгортання та знищення ізольованих інстансів кіберполігону на хмарній платформі AWS, і Модуль ШІ-аналізу, який

використовує бібліотеку Gemini. Цей модуль формує структурований запит на основі наданих Студентом логів чи результатів, надсилає його до API ШІ, отримує якісну оцінку методології та обробляє її для подальшого зберігання.

Третій рівень, рівень даних та зовнішніх сервісів, включає всі постійні сховища та високоспеціалізовані сервіси. Тут розташована база даних, яка для гнучкості та простоти розгортання може бути реалізована на SQLite3, зберігаючи інформацію про користувачів, курси та деталізовані результати оцінювання. Фізично інфраструктура розгортається на AWS, яка надає обчислювальні потужності (EC2) для роботи самого сервера та кіберполігону. Управління життєвим циклом цих ресурсів здійснюється за допомогою Terraform, а ізольовані, відтворювані цільові машини створюються за допомогою Docker. Нарешті, Зовнішній ШІ-провайдер (Gemini) є ключовим сервісом на цьому рівні, надаючи моделі машинного навчання для виконання якісного аналізу методології виконання завдань, що є основною цінністю продукту. Ця чітка архітектура забезпечує гнучке, незалежне та масштабоване функціонування всієї системи ().

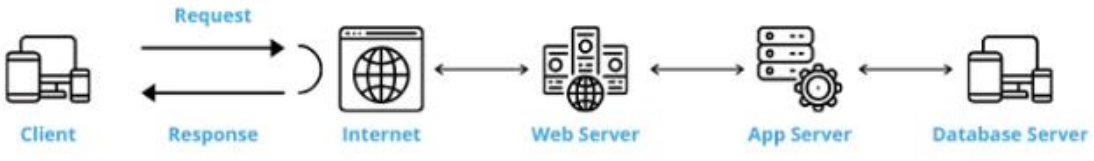


Рис. 3.1. Ілюстрація клієнт-серверної архітектури

У даній архітектурі можна виділити наступні особливості:

1) централізація (сервери керують даними та ресурсами, що полегшує адміністрування та підтримку системи).

2) розподіл завдань (клієнти виконують інтерфейсні обчислення, а сервери виконують складніші завдання, такі як управління базами даних та обробка бізнес логіки).

3) масштабованість (сервери можна додавати та оновлювати без значних змін для клієнтів).

4) безпека (сервери можуть забезпечувати централізовану безпеку, керуючи доступ до ресурсів, а також автентифікацією та авторизацією користувачів).

3.1.3. Визначення вимог до розробки програмного забезпечення

Розробка вимог до програмного забезпечення є критично важливим етапом, що забезпечує чітке розуміння функціоналу та якісних характеристик кінцевого продукту. Вимоги до системи автоматизованої перевірки якості виконання завдань з інформаційної безпеки формуються на основі аналізу потреб цільових користувачів – викладачів та студентів, та необхідності інтеграції передових інфраструктурних рішень (кіберполігони) і технологій аналізу (Штучний Інтелект). Вони поділяються на функціональні, які описують поведінку системи та її взаємодію, та нефункціональні, які описують якісні характеристики роботи системи.

Функціональні вимоги визначають, що саме система повинна виконувати для задоволення потреб користувачів у контексті управління навчанням та оцінювання:

1) Управління доступом та ролями: Система має забезпечувати надійну автентифікацію та авторизацію користувачів, чітко розмежовуючи права доступу для ролей «Студент» та «Викладач».

2) Управління навчальним контентом: викладач повинен мати функціонал для створення, редагування та публікації практичних завдань з кібербезпеки, включаючи встановлення термінів здачі та критеріїв оцінювання.

3) Розгортання інфраструктури кіберполігону: система повинна інтегруватися з Terraform та AWS для автоматичного та швидкого розгортання персоналізованого, повністю ізольованого інстансу тренувального середовища для кожного студента за єдиним шаблоном.

4) Надсилання результатів: система повинна надавати студенту зручні механізми для надсилання результатів виконання завдання, які можуть включати логі роботи, скрипти чи знайдені ідентифікатори («прапори»/Flags).

5) Автоматизована ШІ-перевірка якості: система повинна використовувати бібліотеку Gemini для програмного надсилання даних роботи студента до моделі штучного інтелекту з метою деталізованої оцінки методології виконання завдання.

Нефункціональні вимоги описують якісні атрибути, що визначають, наскільки ефективно, надійно та безпечно функціонує система в умовах її практичного застосування в освітньому процесі.

Вимоги до продуктивності та масштабованості є критичними для підтримки навчального процесу великих груп студентів. Зокрема, процес розгортання ізольованого інстансу кіберполігону для одного студента має бути максимально швидким, що забезпечується використанням Terraform та AWS, щоб не створювати затримок і гарантувати систематичність практичних занять. Аналогічно, час генерації деталізованого звіту про перевірку якості від ШІ має бути мінімальним, що забезпечує миттєвий зворотний зв’язок. Архітектура системи повинна підтримувати високе навантаження та забезпечувати одночасне функціонування великої кількості ізольованих тренувальних середовищ, що є необхідною умовою для масштабованості.

Жорсткі вимоги до безпеки та ізоляції є фундаментальними для системи, що працює з кібератаками. Інстанси кіберполігону, що контролюються Docker та Terraform, повинні бути абсолютно ізольовані один від одного та від внутрішньої мережі закладу освіти, забезпечуючи законність та безпеку відпрацювання атак. Крім того, всі дані, що передаються між Клієнтом та Сервером, включаючи логи роботи студентів та результати ШІ-аналізу, мають передаватися за захищеним протоколом (HTTPS), а доступ до API Gemini та Terraform повинен бути суворо обмежений.

Вимоги до надійності та юзабіліті забезпечують стабільність та зручність користування системою. Інфраструктурний код Terraform має бути стійким до помилок при розгортанні, забезпечуючи багаторазову реплікацію ідентичних середовищ. Інтерфейс, розроблений на базі Bootstrap, повинен бути інтуїтивно зрозумілим для швидкого освоєння Викладачами та Студентами, забезпечуючи зручний перегляд складних аналітичних звітів. Система також повинна коректно функціонувати у всіх поширених сучасних веб-браузерах та на різних типах пристроїв, підтримуючи таким чином доступність освітнього процесу (Посвістак В. С., & Демківська Т. І., 2020).

3.2 Програмна реалізація

3.2.1. Опис реалізації сервера

Для розгортання та запуску освітньої платформи необхідне попередньо налаштоване середовище, що включає інтерпретатор Python та систему управління пакетами pip. Первинне налаштування вимагає встановлення ключових залежностей, зокрема мікрофреймворку Flask, бібліотеки requests для взаємодії з API, markdown2 для рендерингу, та werkzeug для безпеки.

Вся конфігурація серверної частини інкапсульована у головному файлі додатку (app.py). При ініціалізації екземпляру Flask визначаються фундаментальні параметри, що керують поведінкою системи:

1) DATABASE: визначає шлях до файлу бази даних SQLite 3. Шлях є динамічним і вказує на спеціалізовану instance директорію, що ізолює дані від кодової бази.

2) UPLOAD_FOLDER: вказує на захищену директорію (instance/writeups), призначену для зберігання файлів, завантажених користувачами (еталонні рішення та райтапи).

3) ALLOWED_EXTENSIONS: встановлює «білий список» дозволених розширень файлів (наприклад, .md, .txt) для запобігання завантаженню потенційно небезпечного контенту.

4) SECRET_KEY: критично важливий параметр, що використовується для криптографічного підпису клієнтських сесій (cookies), забезпечуючи механізм автентифікації.

5) TEACHER_REGISTRATION_CODE: спеціальний статичний пароль, який система використовує для верифікації користувачів, що намагаються зареєструватися з привілейованою роллю «викладач».

Перед першим запуском серверу необхідно створити структуру бази даних. Для цього реалізовано спеціалізовану команду командного рядка (CLI). Виконання flask init-db у терміналі активує функцію, яка відкриває та виконує SQL-скрипт, визначений у файлі schema.sql. Цей скрипт створює всі необхідні таблиці (для користувачів, класів, завдань, виконань) та визначає зв’язки (foreign keys) між ними, гарантуючи цілісність даних.

Після ініціалізації бази даних, додаток готовий до запуску. Сервер активується виконанням стандартної команди flask run. Ця команда запускає вбудований у Flask веб-сервер у режимі розробки, який починає прослуховувати HTTP-запити на локальній адресі (127.0.0.1, порт 5000), роблячи платформу доступною для використання через веб-браузер.

Для цільової системи автоматизацію оцінювання завдань з інформаційної безпеки було розроблено наступні кінцеві точки:

Таблиця 3.1.

Список кінцевих точок серверної частини програми

Кінцева точка	Метод	Дія	Вхідні параметри	Тип відповіді
/	GET	Головна сторінка. Перенаправляє на /login або /dashboard залежно від статусу сесії.	-	Redirect
/login	GET	Показує сторінку входу.	-	HTML
/login	POST	Обробляє дані форми, автентифікує користувача, створює сесію.	username, password (з форми)	Redirect
/register	GET	Показує сторінку реєстрації.	-	HTML
/register	POST	Обробляє дані форми, валідує код викладача, створює нового користувача.	username, full_name, password, role, teacher_code (з форми)	Redirect
/logout	GET	Завершує сесію користувача.	Сесія	Redirect

/dashboard	GET	Головна панель. Перенаправляє на панель викладача або студента залежно від ролі у сесії.	Сесія	Redirect
/uploads/<filename>	GET	Безпечна роздача файлів (райтапів, рішень). Перевіряє права доступу.	filename (з URL), Сесія	File / Abort (403/404)
/teacher/dashboard	GET	(Викладач) Показує головну панель зі списком класів.	Сесія	HTML
/teacher/dashboard	POST	(Викладач) Створює новий клас.	Сесія, class_name (з форми)	Redirect
/teacher/class/<int:class_id>	GET	(Викладач) Показує детальну сторінку класу (завдання, студенти, активність).	class_id (з URL), Сесія	HTML
/teacher/class/<int:class_id>/add_task	POST	(Викладач) Створює нове завдання для класу, обробляє завантаження файлу-рішення.	class_id (з URL), Сесія, Дані форми (title, description, flag...)	Redirect

/teacher/task/<int:task_id>/submissions	GET	(Викладач) Показує список усіх виконань для конкретного завдання.	task_id (з URL), Сесія	HTML
/teacher/review_submission/<int:submission_id>	GET	(Викладач) Показує сторінку порівняння райтапів та висновок ШІ.	submission_id (з URL), Сесія	HTML
/student/dashboard	GET	(Студент) Показує панель зі списком завдань.	Сесія	HTML
/student/dashboard	POST	(Студент) Обробляє приєднання до класу за кодом.	Сесія, join_code (з форми)	Redirect
/student/profile	GET	(Студент) Показує профіль студента, загальний рахунок та лідерборди.	Сесія	HTML
/student/task/<int:task_id>	GET	(Студент) Показує сторінку конкретного завдання.	task_id (з URL), Сесія	HTML
/student/task/<int:task_id>	POST	(Студент) Обробляє здачу прапора та райтапу. Викликає Gemini API.	task_id (з URL), Сесія, flag, writeup_file (з форми)	Redirect

Клієнтська частина (front-end) веб-застосунку реалізована за допомогою HTML, CSS та JavaScript, з використанням фреймворку Bootstrap 5 для забезпечення адаптивного та сучасного дизайну. На відміну від односторінкових застосунків (SPA), даний проєкт використовує архітектуру рендерингу на боці сервера (SSR).

У цій моделі серверний фреймворк Flask відповідає за обробку кінцевих точок (URL-адрес) та динамічну генерацію HTML-сторінок за допомогою шаблонізатора Jinja2. Кожна «сторінка» або «компонент», який бачить користувач, є окремим HTML-файлом у директорії templates/, що успадковує базовий шаблон layout.html.

Таблиця 3.2.

Список кінцевих точок клієнтської частини програми

Кінцева точка (Маршрут)	Шаблон (Компонент)	Дія (Призначення сторінки)
/login	login.html	Надає користувачу форму для входу в систему.
/register	register.html	Надає форму для реєстрації, включаючи динамічну логіку (на JavaScript) для показу/приховування поля «Код викладача».
/teacher/dashboard	teacher_dashboard.html	(Викладач) Головна панель, що відображає список класів та надає форму для створення нового класу.

/teacher/class/<id>	view_class.html	(Викладач) Детальна сторінка класу з навігацією по вкладках (Bootstrap Tabs) для перегляду завдань, студентів, активності та додавання нових завдань.
/teacher/task/<id>/submissions	view_all_submissions.html	(Викладач) Відображає таблицю усіх виконань (правильних та неправильних) для конкретного завдання.
/teacher/review_submission/<id>	review_submission.html	(Викладач) Сторінка оцінювання, що відображає пліч–о–пліч еталонне рішення та райтап студента (конвертовані з Markdown), а також висновок ШІ.
/student/dashboard	student_dashboard.html	(Студент) Головна панель, що відображає список доступних завдань та надає форму для приєднання до класу за кодом.
/student/profile	student_profile.html	(Студент) Сторінка профілю, що показує загальну кількість балів та лідерборди по кожному класу.
/student/task/<id>	view_task.html	(Студент) Сторінка завдання. Відображає опис, URL/IP цілі (якщо є) та надає форму для відправки прапора і завантаження файлу-райтапу.

3.2.2 Опис бази даних

Для зберігання даних освітньої платформи було обрано реляційну систему управління базами даних SQLite 3. Цей вибір обумовлений відсутністю необхідності у розгортанні окремого серверного процесу, оскільки SQLite є вбудованою, файловою СУБД, що ідеально інтегрується з Python та відповідає масштабам і вимогам застосунку.

Схема даних спроєктована з дотриманням принципів нормалізації для забезпечення цілісності даних та уникнення їх дублювання. Архітектура складається з п’яти ключових таблиць.

Центральною сутністю є таблиця users, що зберігає автентифікаційні дані користувачів. Для гарантування безпеки, паролі зберігаються не у відкритому вигляді, а як криптографічний хеш (password_hash). Авторизація реалізована через колонку role, яка чітко розмежовує права («student» або «teacher»).

Таблиця classes представляє навчальні класи. Вона пов’язана з таблицею users відношенням «один-до-багатьох» через зовнішній ключ teacher_id, що вказує на викладача-власника класу. Кожен клас має унікальний join_code для контрольованого приєднання студентів.

Для реалізації відношення «багато-до-багатьох» між студентами та класами введено проміжну (асоціативну) таблицю enrollments. Вона містить пари ключів student_id та class_id, ефективно реєструючи студентів у відповідних класах.

Таблиця tasks містить опис завдань. Вона пов’язана з таблицею classes відношенням «один-до-багатьох». Ця таблиця зберігає всю інформацію про завдання: опис, статичний прапор (flag), кількість балів за прапор (points), бали за райтап (writeup_points), а також опціональні посилання на зовнішній ресурс (instance_url) та на файл з еталонним рішенням (solution_path).

Нарешті, таблиця submissions фіксує кожну спробу студента вирішити завдання. Вона є центральною для відстеження прогресу і пов’язана одночасно з таблицями users (через student_id) та tasks (через task_id). У цій таблиці

зберігається результат спроби (is_correct), шлях до завантаженого студентом файлу-райтапу (writeup_path), нараховані бали за прапор та райтап, а також текстовий висновок, згенерований моделлю ШІ (ai_analysis).

В результаті, структуру бази даних було відтворено у вигляді ER-діаграми:

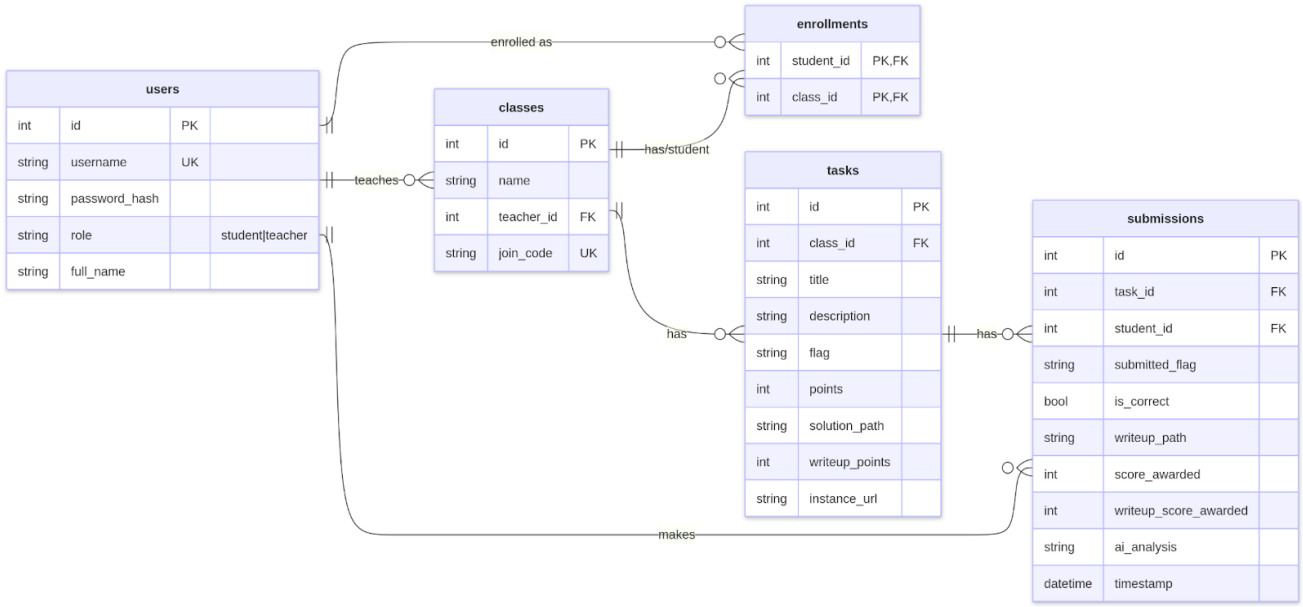


Рис. 3.2. ER-діаграма структури бази даних

3.3 Особливості використання автоматизованої системи перевірки завдань

Впровадження розробленої системи автоматизованої перевірки завдань з інформаційної безпеки змінює сприйняття процесу виконання лабораторних та практичних робіт. Воно накладає на викладача системно іншого підходу до формування завдань: більш ретельно пропрацьованого та структурно орієнтованого. Як наслідок вона надає змогу перевіряти не те, що студент запам’ятав, а те, чи зрозумів він завдання, чи зміг належно налаштувати конфігурацію обладнання, чи коректно написати скрипт. Методичні рекомендації щодо застосування такої системи представлено в Додатку Б.

Самі завдання які можна реалізовувати в системі поділяються на:

1. налаштування конфігурації – вони полягають у перевірці коректності налаштування параметрів операційної системи та окремих програмних за стосунків. Для перевірки коректності виконання передбачається використання скриптів;

2. пентест – результатом є знаходження відповідних «прапорців» під час вдало організованої «атаки»;

3. скриптування – написання програмних модулів для автоматизації процесів інформаційної безпеки. Для перевірки можна скористатись відповідними unit-тестами.

Самі завдання мають відповідати критеріям:

– детермінованості – результат виконання перевірки кожного елементу завдання має бути однозначним та не передбачати оціночних суджень;

– ізольованості середовища – кожен студент має виконувати завдання у власному середовищі, щоб никнути впливу на результати інших;

– незмінності стану – результат перевірки виконання завдання не має впливати на стан самої системи, а отже повторна перевірка має показати такий самий результат.

Для гнучкого оцінювання застосовується система штрафів та винагород, які можуть визначати кінцеву оцінку та диференціювати рівень оволодіння матеріалом навіть серед студентів які показали однаковий кінцевий результат. Для цього, за необхідності, встановлюються ліміти на кількість спроб та час виконання, що дозволяє запобігти використанню прямого перебору можливих комбінацій для досягнення потрібного результату. Також слід застосовувати динамічне оцінювання на результат якого можуть впливати: швидкість виконання, кількість спроб, частка коректно виконаних етапів завдання тощо.

При підготовці завдань та налаштуванні системи автоматичної перевірки їх виконання викладачеві слід уважно поставитись до налаштування системи

зворотного зв’язку яка не має бути бінарною («завдання виконано», або «завдання не виконано»), а містити більш розлогий коментар. Попри те що в розробленій системі ця функція покладається на ШІ, все ж щоразу перед публікацією завдань доцільно переконатись, щодо коректності відповіді на основні помилки які можуть допустити студенти.

Важливо пам’ятати, що попри автоматизацію процесу перевірки та оцінювання якості виконання завдань з інформаційної безпеки, остаточне рішення про зарахування результату перевірки приймає викладач. Система ж має на меті лише спростити процес оцінювання та надати викладачу зручний інструмент для збору даних про результати перевірки комплексних та об’ємних рішень.

Висновки до розділу 3

У даному розділі було детально описано процес проєктування та програмної реалізації серверної та клієнтської частин освітньої CTF-платформи. Архітектуру системи було побудовано на базі мікрофреймворку Flask, що забезпечило гнучку монолітну структуру, відповідальну за всю бізнес-логіку, маршрутизацію та взаємодію з даними. Клієнтська частина реалізована з використанням технології рендерингу на боці сервера (SSR) за допомогою шаблонізатора Jinja2 та фреймворку Bootstrap 5, що дозволило створити чіткий та адаптивний інтерфейс для двох ключових ролей: викладача та студента. Для зберігання даних було спроєктовано та імплементовано реляційну базу даних SQLite 3, схема якої ефективно моделює користувачів, навчальні класи, завдання та їх виконання. Безпека системи гарантується механізмами автентифікації на основі сесій та авторизації, що базується на ролях. Ключовою інновацією розробленої платформи є інтеграція з хмарною моделлю штучного інтелекту Gemini. Ця інтеграція дозволила автоматизувати процес оцінювання студентських райтапів шляхом їх порівняння з еталонними рішеннями викладача, що значно знижує навантаження на викладачів та надає миттєвий зворотний зв’язок. Вся взаємодія з системою

відбувається через чітко визначений набір кінцевих точок, що забезпечує логічну та безпечну обробку всіх запитів.

ВИСНОВКИ

У рамках виконаної дипломної роботи було успішно вирішено актуальну науково-практичну проблему, яка виникає на перетині динамічного розвитку кіберзагроз та вимог сучасної вищої освіти, а саме: забезпечення автоматизованої, систематичної та об’єктивної оцінки якості виконання практичних завдань з інформаційної безпеки для здобувачів вищої освіти. Цей підхід є критично важливим для України, яка перебуває в умовах гібридної війни та потребує системної підготовки висококваліфікованих фахівців, здатних ефективно протистояти складним деструктивним інформаційним впливам.

На початковому етапі дослідження було проаналізовано предметну область та обґрунтовано необхідність радикальної зміни підходів до контролю знань. Встановлено, що традиційні методи навчання та контролю (усні відповіді, тестові перевірки) у закладах вищої освіти не можуть повноцінно оцінити здатність студента до комплексних дій у реальних, стресових умовах кіберзагрози. Сучасний фахівець повинен мати практичні навички у таких сферах, як тестування на проникнення (PenTest), аудит вихідного коду та DevSecOps. Була доведена абсолютна необхідність застосування кіберполігонів (Cyber Ranges) – ізольованих, реалістичних, віртуальних тренувальних середовищ – як єдиного легального та безпечного способу формування та оцінки цих практичних навичок.

Аналіз існуючих рішень виявив суттєвий розрив між адміністративними потребами освітнього закладу та специфічними вимогами практичної підготовки. Системи управління навчанням (LMS), як-от Moodle та Google Classroom , є незамінними для адміністративного управління курсами, проте створюють технологічний бар’єр для якісної ІБ-освіти через відсутність симуляції реального середовища та примітивність автоматизованої перевірки, яка обмежується бінарним результатом або введенням «прапора» (Flag). Це унеможливлює

об’єктивну оцінку методології роботи студента. У свою чергу, спеціалізовані платформи, як Hack The Box та TryHackMe, ефективно вирішують проблему практичних навичок, але не задовольняють педагогічну потребу закладів фахової передвищої освіти в автоматизованій, систематичній, деталізованій оцінці якості виконання завдань. Таким чином, було сформовано ключову невирішену проблему: відсутність системи, що поєднує практичний простір кіберполігону з педагогічною ефективністю.

Для вирішення поставленої проблеми була розроблена та обґрунтована трирівнева клієнт-серверна архітектура, що базується на принципах мікросервісів та Infrastructure as Code (IaC).

Ключову функціональну цінність та інноваційну складову системи забезпечує інтеграція зі штучним інтелектом (бібліотека Google Gemini), яка використовується для якісного аналізу наданих студентом логів та результатів. Ця інтеграція дозволяє перейти від бінарної перевірки результату до деталізованої оцінки методології виконання завдання, надаючи викладачам критично важливу інформацію для корекції знань.

Була успішно розроблена архітектурна модель та сформовані детальні функціональні і нефункціональні вимоги до програмного забезпечення, здатного подолати технологічний розрив у кібербезпековій освіті. Створена система забезпечує систематичне, масштабоване та об’єктивне оцінювання, надаючи викладачам необхідний інструмент для ефективного управління навчальним процесом, що є прямою інвестицією у національну безпеку та євроінтеграцію.

Запропонована методика використання розробленої системи дозволила обґрунтувати напрямки та способи її використання в навчальному процесі, а також роль і завдання викладачів і студентів.

Подальші роботи будуть спрямовані на удосконалення практичної імплементації спроектованих модулів та розширення тестування системи у навчальних закладах.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Бріжак В. Г. (б. д.). Використання моделі SaaS у різних сферах життя. *Матеріали 77-ї студентської науково-технічної конференції «Тиждень студентської науки»* (с. 335–336). НТУ «Дніпровська політехніка».
2. Вараксіна, Н. В. (2021). Використання віртуальних лабораторій в освіті. *Аналітичний вісник у сфері освіти й науки: довід. бюл.*, (14), 3-11.
3. Василенко В., Грицкевич Г., & Кузнецов І. (2025). Роль змагань Capture-The-Flag (CTF) у дослідженнях та навчанні з кібербезпеки: *Аналіз машини «Editorial». Кібербезпека: освіта, наука, техніка*, 4(28), 75–84. https://doi.org/10.28925/2663-4023.2025.28.762
4. Глибовець, А. М., & Бабич, Т. А. (2025). Implementation of a Practice-Oriented Cyber Range System with Individualized Environment Deployment. *Наукові записки НаУКМА. Комп'ютерні науки*, 8, 174-179.
5. Глибовець, А., & Бабич, Т. (2024). Інтеграція кіберполігонів в освітній процес. *Теоретичні та прикладні аспекти побудови програмних систем : праці 15 міжнародної науково-практичної конференції*. 107-110
6. Горбач Т.В, Славгородський В.Ю., Шубін І.Ю., & Ковалевська А.В. (2018) Моделі електронного навчання та вимоги до програмного забезпечення. *Проблеми інформаційних технологій*, 23, 205-218
7. Гуревич, Р., Коношевський, Л., Коношевський, О., Воєвода, А., & Люльчак, С. (2024). Інтеграція штучного інтелекту в сферу освіти: проблеми, виклики, загрози, перспективи. *Modern Information Technologies and Innovation Methodologies of Education in Professional Training Methodology Theory Experience Problems*, (72), 170-186.
8. Гурська, О., & Лучицький, А. (2024). Штучний інтелект у забезпеченні кібербезпеки: сучасні виклики та перспективи. *Кібербезпека в транскордонному співробітництві. Міжнародна науково-практична конференція*. 135
9. Делембовський, М. М., Пристайло, М. О., & Маркевич, М. О. (2025). Проблеми та виклики підготовки фахівців з кібербезпеки. *Актуальні проблеми освітнього процесу в контексті європейського вибору України : зб. матер. VII Міжнародної конференції*, 131 – 135
10. Дутчак, М. С. (2020). Методи та програмні засоби автоматизованої побудови адаптивної траєкторії навчання. *Вісник Вінницького політехнічного інституту*, 2, 58-66.

11. Жданова, Ю. Д., Спасітєлєва, С. О., & Шевченко, С. М. (2019). Формування практичних навичок студентів спеціальності 125 Кібербезпека за допомогою віртуальних лабораторій. *VII Міжнар. наук.-практ. конф. «Математика в сучасному технічному університеті»*. 253-255

12. Кобильник Т., Когут У., & Жидик В. (2021). Методичні аспекти вивчення основ алгоритмізації і програмування мовою Python у шкільному курсі інформатики у старших класах. *Physical and Mathematical Education*, 31(5), 36–44. https://doi.org/10.31110/2413-1571-2021-031-5-006

13. Колеснікова, В. В. (2022). Практичні аспекти діджиталізації дуальної форми освіти у закладах фахової передвищої освіти. *Електронне наукове фахове видання «Відкрите освітнє е-середовище сучасного університету»*, (12), 69-79.

14. Комарницький, К. (2025). Методи та засоби формування компетентності з кібернетичної безпеки в процесі професійної підготовки майбутніх інженерів-програмістів. *Науковий вісник Мелітопольського державного педагогічного університету. Серія: Педагогіка*, 1(34), 191-199.

15. Кузнецова, О & Фазан, В & Штефан, Л. (2024). Впровадження програмного забезпечення з використанням штучного інтелекту у навчальний процес закладів освіти. *Теорія та методика навчання та виховання*. 68-79. 10.34142/23128046.2024.57.06.

16. Листопад, О. А., & Листопад, Н. Л. (2025). Організація дистанційного навчання на платформі Moodle: теорія та практика. *Науковий вісник Ізмаїльського державного гуманітарного університету: збірник наукових праць. Педагогічні науки*, 70, 145-155.

17. Обозний Д. М. (2020). Спосіб прискорення реконфігурації мережі в середовищах хмарних обчислень [Магістерська дисертація, не публікована]. Київ.

18. Освітньо-професійна програма «Кібербезпека та захист інформації» (2025) *Рівненський фаховий коледж Національного університету біоресурсів і природокористування України* URL: https://rfc.nubip.edu.ua/wp-content/uploads/2025/11/proyekt_opp_f5_125_fmb_2025_compresspdf.pdf

19. Павленко Ю. С. (2022). Верстка веб-сторінок з допомогою HTML та CSS: Методичні рекомендації до першого модуля нормативної дисципліни «Програмування та підтримка веб-застосувань». *Волинський національний університет імені Лесі Українки*.

20. Піскунов О., Єсик Л. В., & Томащук О. П. (2022). Розробка клієнт-серверних додатків мовою C# та SQLite. *ResearchGate*. https://doi.org/10.13140/RG.2.2.18178.90561/1

21. Поліщук Н. В., Бугаєнко Т. І., & Лемешева Н. В. (2024). Підвищення якості вищої освіти за допомогою цифрових технологій та дистанційного навчання для здобувачів вищої освіти в Україні. *Академічні візії*, 38, 1–7. https://doi.org/10.5281/zenodo.14537287

22. Посвістак В. С., & Демківська Т. І. (2020). Клієнт-серверна архітектура та її використання при розробці програмного забезпечення. Інформаційні технології в науці, виробництві та підприємництві: *Збірник наукових праць молодих вчених, аспірантів, магістрів кафедри комп'ютерних наук та технологій* (с. 78–81). ФОП Маслаков.

23. Синявіна, Ю. Інтеграція штучного інтелекту в підготовку спеціалістів з кібербезпеки та захисту інформації. *Інтеграція штучного інтелекту в освіту – виклики та можливості : збірник тез науково-методичних доповідей Всеукраїнського науково-педагогічного підвищення кваліфікації*, 720-723. DOI: https://doi.org/10.36059/978-966-397-477-4-189

24. Січко Т. В., & Яцковська Р. О. (2020). Web-програмування: Методичні вказівки для виконання лабораторних та самостійних робіт здобувачами освітньо-кваліфікаційного рівня «Бакалавр» напряму підготовки 6.030502 «Економічна кібернетика» денної та заочної форм навчання. *Вінницький національний аграрний університет*.

25. Скляр, Р. В., & Скляр, О. Г. (2025). Цифрові платформи моніторингу якості освіти: від Moodle до аналітики на базі Power BI. *Удосконалення освітньо-виховного процесу в закладі вищої освіти: збірник*, 283.

26. Соколов, В., & Складанний, П. (2023). Порівняльний аналіз стратегій побудови другого та третього рівня освітніх програм зі спеціальності 125 «кібербезпека». *Електронне фахове наукове видання «Кібербезпека: освіта, наука, техніка»*, 4(20), 183–204. https://doi.org/10.28925/2663-4023.2023.20.183204

27. Стасюк, О. (2025). Цифрова трансформація закладів вищої освіти. *Проблеми і перспективи економіки та управління*, 3 (43), 232-242.

28. Суков, М. В. (2025). Комплексний підхід до розробки курсу з кібербезпеки для професійних навчальних закладів. *Всеукраїнська науково-практична Інтернет-конференція «Автоматизація та комп'ютерно-інтегровані технології у виробництві та освіті: стан, досягнення, перспективи розвитку» Секція 6. Інформаційні технології в навчанні та управлінні освітнім процесом*. 151-153

29. Сус В. (2024). Аналіз моделі прогнозування кіберзагроз «Cyber Kill Chain». *Modeling, control & information technologies*, (1), 63. https://doi.org/10.31713/MCIT.2024.063

30. Шакуров Є. О., & Клокова К. В. (2022). Розробка веб-сайту з використанням фреймворку Bootstrap. *Наумовські читання: Збірник тез доповідей учасників XX Всеукраїнської науково-методичної конференції здобувачів вищої освіти та молодих вчених, присвяченої 300-річчю з дня народження Г. С. Сковороди* (с. 185–187). Харківський національний педагогічний університет імені Г. С. Сковороди.

31. Шарун, П. В. (2025) Огляд інтерактивних засобів навчання основам кібербезпеки. *Матеріали XVI-ої Міжнародної науково-практичної конференції «Free and Open Source Software»*, 92.

32. Шевчук, О. Ф., Яровий, А. А., Паночишин, Ю. М., Петришин, С. І., & Козловський, О. А. (2025). Моделювання адаптивного тестування знань: поріг ефективності, рівень складності та час виконання завдань. *Вісник Вінницького політехнічного інституту*, 1: 104–112.

33. Шелестов А. Ю., & Колотій А. В. (2022). Хмарні технології обробки даних: Лабораторний практикум [Навчальний посібник]. *КПІ ім. Ігоря Сікорського*.

34. Шишенко, І. (2022). Деякі аспекти впливу цифрових технологій на освітній процес закладів вищої освіти: огляд проблем та викликів. *Освіта. Інноватика. Практика*, 10(5), 42-47.

35. Щербина І. С., & Явор Д. В. (2023). Вплив технології Docker на архітектуру мікросервісів. *Наукові записки Державного університету інформаційно-комунікаційних технологій*, (1). https://doi.org/10.31673/2786-8362.2023.010707

36. (б. д.). Google for Education. https://edu.google.com/workspace-for-education/products/classroom/

37. Cyber Mastery: Community Inspired. Enterprise Trusted. (б. д.). *Hack The Box*. https://www.hackthebox.com/

38. Danyk, Y. (2019). Особливості створення кіберполігонів для дослідження комплексних кібердій та підготовки фахівців з кібербезпеки. *Сучасні інформаційні технології у сфері безпеки та оборони*, 34(1), 95-102.

39. European Union Agency for Cybersecurity (ENISA). (2021). Addressing the EU cybersecurity skills shortage and gap through higher education. *Publications Office of the European Union, Luxembourg*.

40. Furfaro, A., Piccolo, A., Parise, A., Argento, L., & Saccà, D. (2018). A cloud-based platform for the emulation of complex cybersecurity scenarios. *Future Generation Computer Systems*, 89, 791-803.

41. Home | Moodle.org. (б. д.). *Home | Moodle.org*. https://moodle.org

42. Katsantonis, M. N., Manikas, A., Mavridis, I., & Gritzalis, D. (2023). Cyber range design framework for cyber security education and training. *International Journal of Information Security*, 22(4), 1005-1027.

43. Kumar, R. P., Muthukumaran, N., Hanisha, C., Rithvik, P., & Goud, M. S. (2023) Examination system automation using natural language processing. *In 2023 International Conference on Self Sustainable Artificial Intelligence Systems (ICSSAS)*, 1002-1008

44. Puchkov O., & Uvarkina O. (2023). Sustainable development of the system of formal cyber education: reflection of modern concepts. *Collection «Information Technology and Security»*, 11(1), 60–68. https://doi.org/10.20535/2411-1031.2023.11.1.283635

45. Shevchenko, S., Zhdanova, Y., Spasiteleva, S., & Skladannyi, P. (2020). Проведення swot-аналізу оцінювання інформаційних ризиків як засіб формування практичних навичок студентів спеціальності 125 Кібербезпека. *Електронне фахове наукове видання «Кібербезпека: освіта, наука, техніка»*, 2(10), 158-168.

46. TryHackMe | Cyber Security Training. (б. д.). *TryHackMe | Cyber Security Training*. https://tryhackme.com/

47. Tymoshchuk, D., Yatskiv, V., Tymoshchuk, V., & Yatskiv, N. (2024). Interactive cybersecurity training system based on simulation environments. *Measuring and Computing Devices in Technological Processes*, (4), 215–220.. https://doi.org/10.31891/2219-9365-2024-80-26

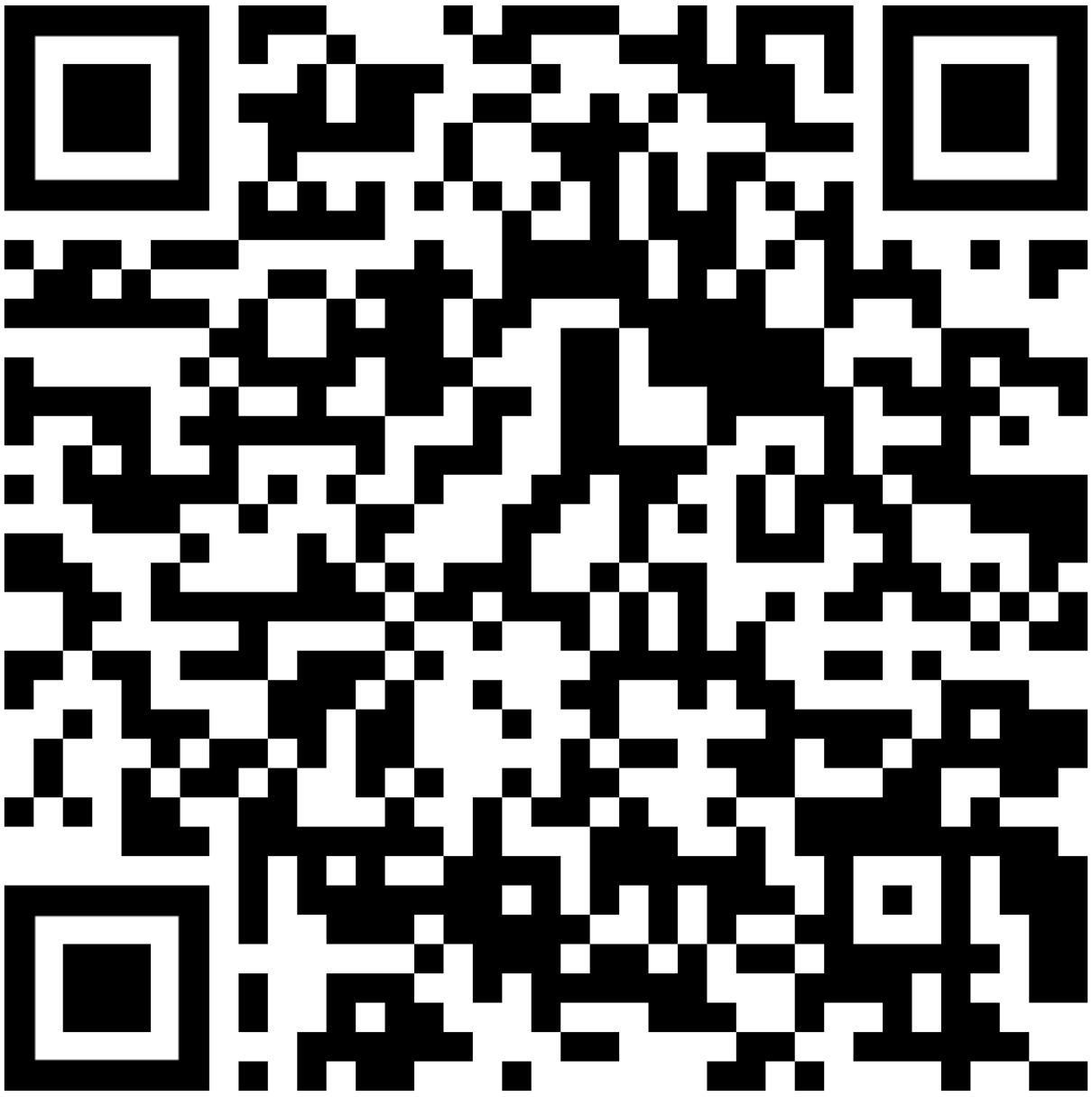
48. Vetrivel, S. C., Vidhyapriya, P., & Arun, V. P. (2025). The role of AI in transforming assessment practices in education. *AI Applications and Strategies in Teacher Education*, 43-70.

49. Welcome to Flask – Flask Documentation (3.1.x). (б. д.). *Welcome to Flask – Flask Documentation (3.1.x).* https://flask.palletsprojects.com/en/stable/

50. Yanev M. (2025). Improving OpenAI and Gemini AI API Responses via Schema Validation for Python Developers (Preprint). *ResearchGate*. https://doi.org/10.21203/rs.3.rs-6247664/v1

Додаток А.

ВИХІДНИЙ КОД СТВОРЕНОГО ДОДАТКУ



QR-код з посиланням на архів з вихідним кодом

Додаток Б

МЕТОДИЧНІ РЕКОМЕНДАЦІЇ ЩОДО ЗАСТОСУВАННЯ АВТОМАТИЗОВАНОЇ СИСТЕМИ ПЕРЕВІРКИ ЗАВДАНЬ З ІНФОРМАЦІЙНОЇ БЕЗПЕКИ

1. Загальні положення

Мета методики: забезпечення об’єктивності оцінювання виконання практичних завдань з інформаційної безпеки та надання зворотного зв’язку здобувачам освіти шляхом впровадження засобу автоматизованого контролю.

Сфера застосування: Навчальні дисципліни, що передбачають вивчення параметрів інформаційної безпеки та дозволяють визначити чіткі критерії оцінювання рівня оволодіння практичними навичками.

2. Підготовчий етап

2.1. Налаштування програмного забезпечення

а) засовлення та налаштування інфраструктури кіберполігону за допомогою Terraform на хмарній платформі AWS здійснюється за допомогою підготовлених програмних інструментів та також open-source компонентів в залежності від потреб освітньої програми. Результуюча інфраструктура має забезпечувати функціонал повністю ізольованого інстансу тренувального середовища для кожного студента за єдиним шаблоном;

б) налаштування інтерфейсу взаємодії з кіберполігоном за принципом Software as a Service за допомогою розроблених програмних компонент. Як результат інтерфейс має бути інтуїтивно зрозумілим для швидкого освоєння Студентами, забезпечуючи зручний перегляд складних аналітичних звітів. Система також повинна коректно функціонувати у всіх поширених сучасних веб-браузерах та на різних типах пристроїв;

в) реєстрація користувачів платформи та присвоєння їм відповідних ролей що забезпечують розмежування прав доступу та обсяг повноважень.

Рекомендовано використовувати мінімум дві ролі: «Викладач» та «Студент». У разі якщо робота з системою передбачається кількома викладачами, то доцільно також створення ролі «Адміністратор» на яку буде покладено завдання управління системою, в той час як Викладач лише викладач лише матиме функціонал для створення, редагування та публікації практичних завдань з кібербезпеки, включаючи встановлення термінів здачі та критеріїв оцінювання.

Примітка: за наявності відповідних обчислювальних ресурсів система може бути розгорнута на локальному сервері.

2.2 Ознайомлення викладача з системою:

а) після реєстрації Викладача йому необхідно зайти в систему під своїми реєстраційними даними та пересвідчитись у коректному присвоєнню ролі;

б) рекомендується попередньо здійснити тестове підключення до системи з метою перевірки як налаштувань роботи мережі, так і сумісності програмного середовища з технічним обладнанням що буде застосовуватись Викладачем під час підготовки до заняття.

в) провести практичне тренування роботи з системою шляхом підготовки та розміщення тестового завдання створеного згідно цих рекомендацій.

3. Підготовка завдань для автоматизованої перевірки

3.1 Зміст завдань

Система передбачає можливість автоматизованої перевірки виконання типізованих завдань:

а) налаштування конфігурації – вони полягають у перевірці коректності налаштування параметрів операційної системи та окремих програмних за стосунків. Для перевірки коректності виконання передбачається використання скриптів;

б) пентест – результатом є знаходження відповідних «прапорців» під час вдало організованої «атаки»;

в) скриптування – написання програмних модулів для автоматизації процесів інформаційної безпеки. Для перевірки можна скористатись відповідними unit-тестами.

Вони мають відповідати функціональним критеріям:

а) детермінованості – результат виконання перевірки кожного елементу завдання має бути однозначним та не передбачати оціночних суджень;

б) ізольованості середовища – кожен студент має виконувати завдання у власному середовищі, щоб никнути впливу на результати інших;

в) незмінності стану – результат перевірки виконання завдання не має впливати на стан самої системи, а отже повторна перевірка має показати такий самий результат.

Для уникнення академічної не доброчесності допускається використання параметризованих завдань, де кожен студент отримує унікальні вхідні дані при збереженні загальної логіки виконання завдань.

3.2 Формалізація критеріїв оцінювання

Диференціювання рівня оволодіння матеріалом при перевірці знань може бути здійснено шляхом застосовування динамічного оцінювання яке включає в себе:

- систему штрафів та винагород;
- встановлення лімітів на кількість спроб;
- обмеження часу виконання.

Примітка: Остаточне рішення про зарахування результату перевірки приймає викладач. Система ж має на меті лише спростити процес оцінювання та надати викладачу зручний інструмент для збору даних про результати перевірки комплексних та об’ємних рішень.

3.3 Зворотній зв’язок

Система зворотного зв’язку передбачає небінарність реакції на надану відповідь щодо виконання завдання і міститиме більш розлогий коментар.

Оскільки в розробленій системі ця функція покладається на ШІ, то щоразу перед публікацією завдань доцільно переконатись, щодо коректності відповіді на основні помилки які можуть допустити студенти. Для цього, перед тим як надавати завдання студентам для виконання, доцільно кілька разів виконати його самостійно із застосуванням автоматизованої системи оцінювання для верифікації коректності наданих нею результатів перевірки.

4. Впровадження в навчальний процес

Застосування системи в навчальному процесі передбачає ідентифікацію студента в системі з подальшою циклічністю виконання окремих етапів:

4.1 Інструктаж студентів

Перед початком виконання завдань студенти повинні отримати чітку інструкцію, яка включає:

- мету завдання;
- чітке окреслення інструментарію виконання завдання;
- вимоги до подачі звіту (опис алгоритму виконання завдання, скрипт тощо);
- інформацію про обмеження (кількість спроб, часові обмеження тощо);

4.2 Виконання завдання

Система автоматично фіксує певну подію-тригер, яка ідентифікує початок виконання студентом завдань та запускає алгоритми нарахування штрафів та винагород.

Після завершення виконання завдання студент готує та передає в систему звіт про виконане завдання.

4.3 Форма подачі звіту про виконання завдання

В якості форми подачі матеріалу для перевірки та оцінювання використовуються звіти, які за структурою відповідають звітам про інциденти та їх вирішення. Для забезпечення достовірності в звіт має бути включена вимога

доказовості (наприклад, шляхом прикріплення відповідного скріншоту з часовою міткою чи додаванням скрипта який використовувався для вирішення завдання).

4.4 Валідація та зворотній зв’язок

Система автоматично здійснює перевірку отриманого звіту із застосуванням інструментів ШІ шляхом порівняння наданого студентом звіту та опису сформованого викладачем.

Результати перевірки повідомляються студенту.

Примітка: студент може оскаржити рішення автоматизованої перевірки результатів виконання завдання звернувшись безпосередньо до викладача та надавши звіт та доказові матеріали зазначені в п. 4.3.

Для уникнення несанкціонованого доступу, після завершення роботи, студент має вийти з системи.

5. Аналіз результатів

Після завершення виконання завдань викладачу рекомендовано здійснити аналізу статистики результатів та оптимізувати завдання для подальшого використання. Зокрема:

- якщо частка студентів що не змогли виконати певне завдання надто висока, то слід перевірити чіткість формулювань в них;
- якщо середній час виконання завдання низький, слід його ускладнити, або переоцінити.

Впровадження даної методики дозволяє усунути необхідність рутинної перевірки викладачем стану виконання завдань з інформаційної безпеки, забезпечивши прозорість оцінювання та скоротити час отримання зворотного зв’язку студентами.