

Міністерство освіти і науки України
Державний університет «Житомирська політехніка»
Факультет інформаційно-комп'ютерних технологій
Кафедра комп'ютерної інженерії та кібербезпеки

ПОЯСНЮВАЛЬНА ЗАПИСКА

до випускної кваліфікаційної роботи бакалавра
на тему: «Проект рекомендацій для підвищення рівня
захисту веб-сайту»

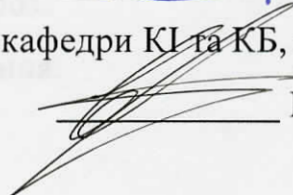
Виконала студентка 4-го курсу групи КІ-20-1
спеціальності 123 «Комп'ютерна інженерія»

 Тетяна ШМИРИНА

Керівник: доцент кафедри КІ та КБ, к.пед.н., доцент

 Олена ГОЛОВНЯ

Рецензент: доцент кафедри КІ та КБ, к.т.н., доцент

 Юрій БРОДСЬКИЙ

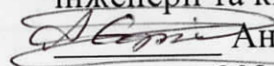
Житомир – 2024

Державний університет «Житомирська політехніка»
Факультет інформаційно-комп'ютерних технологій
Кафедра комп'ютерної інженерії та кібербезпеки
Спеціальність 125 «Кібербезпека»
Освітня програма «Кібербезпека»

ЗАТВЕРДЖУЮ

Завідувач

кафедри комп'ютерної
інженерії та кібербезпеки

 Андрій Єфіменко
5 березня 2024 р.

ЗАВДАННЯ

на випускну кваліфікаційну роботу бакалавра

Студентка: Тетяна ШМИРІНА

Тема роботи: «Проект рекомендацій для підвищення рівня захисту веб-сайту»
затверджена Наказом університету від 5 березня 2024 р., №97 /с

Термін здачі студентом закінченої роботи 9 червня 2024 р.

Вихідні дані роботи (зазначається предмет і об'єкт дослідження):
Об'єктом дослідження є процес розробки захищених інтерфейсів веб-сайтів, який спрямований на розробку інтерфейсу який буде не лише інтуїтивно зрозумілим, а й захищеним. Предметом дослідження є методи та інструменти UX/UI, зокрема обмеження на дані, типи файлів, інші параметри взаємодії, які впливають на безпеку веб-сайтів. Дослідження спрямоване на вивчення технологій та методів, що дозволять розробити проєкт інтерфейсу, який врахує стандарти безпеки та захисту від кіберзагроз.

Обладнання та програмне забезпечення:

1. Windows 10 (64-bit)
2. Ubuntu 22.04
3. phpMyAdmin
4. VS Code
5. PhpStorm

Консультанти з випускної кваліфікаційної роботи із зазначенням розділів, що їх стосуються

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
1			
2			
3			

Студент Тетяна ШМИРІНА
Керівник Олена ГОЛОВНЯ

Календарний план

№ з/п	Назви етапів випускної кваліфікаційної роботи	Термін виконання етапів роботи	Примітка
1	Постановка задачі	05.03.2024	Виконано
2	Пошук, огляд та аналіз аналогічних розробок	15.03.2024	Виконано
3	Формулювання технічного завдання	01.04.2024	Виконано
4	Опрацювання літературних джерел	10.04.2024	Виконано
5	Проектування структури	05.05.2024	Виконано
6	Написання програмного коду	15.05.2024	Виконано
7	Відлагодження	25.05.2024	Виконано
8	Написання пояснювальної записки	05.06.2024	Виконано
9	Захист	27.06.2024	Виконано

Студент Тетяна ШМИРІНА
Керівник Олена ГОЛОВНЯ

АНОТАЦІЯ

Випускна кваліфікаційна робота бакалавра складається з веб-сайту та пояснювальної записки. Пояснювальна записка до випускної роботи містить 97 сторінок та 38 ілюстрацій.

Метою даної роботи є дослідження впливу дизайну UX/UI на безпеку веб-сайтів та розробка рекомендацій щодо підвищення безпеки веб-сайту з метою запобігання можливих кіберзагроз.

Методи, використані під час створення рекомендацій, включають в себе: аналіз наявних систем захисту, програмування на мовах, орієнтованих на безпеку веб-додатків, роботу з інструментами для тестування безпеки, дизайн та впровадження заходів безпеки, аналіз потреб майбутніх користувачів та адміністраторів систем.

Результати кваліфікаційної роботи полягають у створенні інтерактивних та ефективних рекомендацій для підвищення рівня захисту веб-сайту, які включають зручний доступ до інструментів безпеки та планування заходів захисту. Згідно з виробленими рекомендаціями розроблено проєкт інтерфейсу веб-сайту музею науки та технологій «Політехнік».

КЛЮЧОВІ СЛОВА: SECURITY, ВЕБ-САЙТ, HTTPS, SSL, TLS, XSS, CSRF, WAF, IDS, IPS, MFA, БЕЗПЕКА, КІБЕРЗАГРОЗИ, РЕКОМЕНДАЦІЇ, USER EXPERIENCE, USER INTERFACE.

					ККІК.КР.Б – 123 – 24 – ПЗ			
Змн.	Арк.	№ докум.	Підпис	Дата	Проект рекомендацій для підвищення рівня захисту веб-сайту	Літ.	Арк.	Аркуші
Розроб.		Шмиріна Т.С.		26.06.24				
Керівник		Головня О.С.		26.06.24			4	97
Реценз.		Бродський Ю.Б.		26.06.24				
Н. контр.		Головня О.С.		26.06.24				
Затверд.		Сфіменко А.А.		26.06.24				
						ФІКТ, Гр. КІ-20-1		

ANNOTATION

The bachelor's thesis consists of a website and an explanatory note. The explanatory note to the final thesis contains 97 pages and 38 illustrations.

The purpose of this work is to study the impact of UX/UI design on website security and develop recommendations for improving website security in order to prevent possible cyber threats.

The methods used to create the recommendations include: analysis of existing security systems, programming in languages focused on the security of web applications, working with security testing tools, design and implementation of security measures, analysis of the needs of future users and system administrators.

The results of the development are the creation of interactive and effective recommendations for improving the level of website security, including convenient access to security tools and planning of security measures.

KEYWORDS: SECURITY, WEBSITE, HTTPS, SSL, TLS, XSS, CSRF, WAF, IDS, IPS, MFA, SAFETY, CYBERTHREATS, RECOMMENDATIONS, USER EXPERIENCE, USER INTERFACE.

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				5
Змн.	Арк.	№ докум.	Підпис	Дата		

ЗМІСТ

ЗМІСТ	6
ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ	8
ВСТУП	9
РОЗДІЛ 1 РОЛЬ UX/UI В БЕЗПЕЦІ КОРИСТУВАЧА САЙТУ	11
1.1 Основні аспекти UX/UI та їх вплив на безпеку веб-сайтів	11
1.2 Функціональні обмеження інтерфейсу та управління доступом	14
1.3 Управління типами дозволених файлів як аспект безпеки	17
Висновки з розділу 1	19
РОЗДІЛ 2 ПРОЄКТУВАННЯ САЙТУ МУЗЕЮ НАУКИ І ТЕХНОЛОГІЙ «ПОЛІТЕХНІК»	21
2.1 Проєктування фронтенду для сайту університету	21
2.1.1 Заходи безпеки під час проєктування дизайну інтерфейсу	21
2.1.2 Дослідження методів UX/UI для проєктування безпечного сайту	22
2.1.3 Проєктування та реалізація візуальної складової проєкту	25
2.2 Проєктування бекенду веб-сайту музею	33
2.2.1 Постановка завдань	33
2.2.2 Обґрунтування вибору інструментів та вимог до апаратного забезпечення	35
2.2.3 Проєктування архітектури	38
2.2.4 Проєктування бази даних	42
Висновки з розділу 2	46
РОЗДІЛ 3 РОЗРОБЛЕННЯ, НАЛАШТУВАННЯ ТА ТЕСТУВАННЯ САЙТУ МУЗЕЮ НАУКИ І ТЕХНОЛОГІЙ «ПОЛІТЕХНІК»	48
3.1 Заходи безпеки на рівні додатку	48
3.2 Валідація та доступ до endpoints	52
3.3 Заходи безпеки на рівні Docker в конфігурації	53
3.4 Адміністрування бази даних	59
3.5 Рекомендації для підвищення рівня захисту веб-сайту	59
Висновки з розділу 3	63

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		6

ВИСНОВКИ	65
СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ	67
ДОДАТКИ	69

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				7
Змн.	Арк.	№ докум.	Підпис	Дата		

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

UX — User Experience.
UI — User Interface.
Usability — Зручність використання.
Wireframe — Каркас.
Prototype — Прототип.
HTTP — HyperText Transfer Protocol.
HTTPS — HyperText Transfer Protocol Secure.
SSL — Secure Sockets Layer.
TLS — Transport Layer Security.
XSS — Cross-Site Scripting.
IP — Internet Protocol.
URL — Uniform Resource Locator.
DNS — Domain Name System.
API — Application Programming Interface.
JSON — JavaScript Object Notation.
XML — eXtensible Markup Language.
User Profile — Профіль користувача.

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				8
Змн.	Арк.	№ докум.	Підпис	Дата		

ВСТУП

Актуальність теми. В наші дні щодня спостерігається постійне зростання кількості кіберзагроз, що підкреслює важливість забезпечення надійності захищеності веб-сайтів. Таким чином, розвиток сучасних технологій вимагає постійного оновлення заходів безпеки, оскільки нові можливості часто вносять нові виклики. У цьому контексті, вивчення та впровадження інноваційних методів UX/UI стає ключовим фактором адаптації веб-сайтів до сучасних стандартів безпеки.

Підвищення захищеності є стратегічно важливим з погляду публічності та репутації. Вразливості веб-сайтів можуть призвести до витоку конфіденційної інформації та завдати серйозної шкоди репутації організації. Розробка безпечних та зручних рішень є важливою для уникнення таких ризиків.

Законодавчий тиск і вимоги до захисту даних роблять необхідним розробку та впровадження ефективних заходів безпеки, що враховують інтереси користувачів і відповідають вимогам закону. Отже, дослідження та розробка нових методів та інструментів UX/UI для поліпшення захищеності веб-сайтів стає **актуальною** задачею, яка може мати значний вплив на розвиток інтернет-простору, забезпечуючи ефективну захищеність без втрати користувацького комфорту.

Метою даної роботи є дослідження впливу дизайну UX/UI на безпеку веб-сайтів та розробка рекомендацій щодо підвищення безпеки веб-сайту для запобігання можливих кіберзагроз.

Сформульована мета визначає **наступні завдання**:

- Аналіз наявних методів UX/UI, які впливають на безпеку веб-сайтів.
- Розробка інтерфейсних обмежень для користувачів з метою уникнення потенційних кіберзагроз.
- Проєктування та розробка веб-сайту, інтерфейс якого відповідає безпековим обмеженням.
- Розробка рекомендацій для підвищення безпеки веб-сайту.

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				9
Змн.	Арк.	№ докум.	Підпис	Дата		

Об’єктом дослідження є процес розробки захищених інтерфейсів веб-сайтів, який спрямований на розробку інтерфейсу, що буде не лише інтуїтивно зрозумілим, а й захищеним.

Предметом дослідження є методи та інструменти UX/UI, зокрема обмеження на дані, типи файлів, інші параметри взаємодії, які впливають на безпеку веб-сайтів.

У процесі виконання кваліфікаційної роботи було розроблено проєкт інтерфейсу веб-сайту, який враховує стандарти безпеки та включає в себе механізми обмеження введення даних, фільтрації файлів, та інші заходи для захисту від кіберзагроз.

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				10
Змн.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 1 РОЛЬ UX/UI В БЕЗПЕЦІ КОРИСТУВАЧА САЙТУ

1.1 Основні аспекти UX/UI та їх вплив на безпеку веб-сайтів

Веб-сайти на сьогоднішній день стали не лише інформаційними платформами, а й інструментами для різноманітних видів взаємодії між користувачами та компаніями. При цьому, важливою складовою їх успішності є висока якість користувацького досвіду (UX) та інтерфейсу користувача (UI). Однак, нерідко у процесі покращення цих аспектів безпека веб-сайтів може бути недооціненою.

Користувацький досвід (user experience, UX) - це враження, яке користувач отримує, взаємодіючи з веб-сайтом. Він включає в себе такі аспекти, як зручність навігації, швидкість завантаження сторінок, зрозумілість інформації та використання різних функцій. Навігація в ньому має бути проста та зрозуміла, що дозволить користувачам швидко знаходити потрібну інформацію, а оптимізована структура меню та використання інтуїтивно зрозумілих елементів полегшуватимуть процес пошуку.

Інтерфейс користувача (user interface, UI) - це та частина веб-сайту, через яку користувач взаємодіє з його функціоналом. Правильно розроблений UI спрощує користування веб-сайтом та забезпечує приємний користувацький досвід. Функціональність включає в себе не лише привабливість UI, але й ефективність. Кожен елемент інтерфейсу повинен мати чітко визначену функцію та легко виконувати її. Також UI має бути спрощеним та легким для розуміння. Інтуїтивно зрозумілі кнопки, посилання та інші елементи взаємодії дозволяють користувачам без ускладнення здійснювати потрібні дії.

До поширених методів UX/UI дизайну належать наступні ^[1].

- Дослідження користувачів (User Research)

Проведення спостережень, опитувань, інтерв'ю та інших дослідницьких методів для збору інформації про потреби, поведінку та вподобання користувачів.

- Побудова карти шляху користувача (User Flow)

Створення візуальної моделі, що показує послідовність кроків, які користувач може виконати для досягнення певної мети на сайті чи додатку.

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				11
Змн.	Арк.	№ докум.	Підпис	Дата		

- Структурування інтерфейсу користувача (Wireframing)

Створення простого варіанту макету, який показує розміщення основних елементів і контенту на сторінці чи екрані, без детальності дизайну.

- Аналіз конкурентів (Competitive Analysis)

Вивчення і аналіз інших продуктів або послуг на ринку для з'ясування їхніх сильних і слабких сторін, щоб виявити можливості для вдосконалення власного продукту.

Покращення UX та UI може мати як позитивний, так і негативний вплив на безпеку веб-сайтів. Зручний UX може підвищити довіру користувачів до веб-сайту, що сприяє збільшенню трафіку та покращенню репутації. Ефективний UI допомагає уникнути помилок користувачів, що можуть призвести до потенційних проблем безпеки, таких як випадкове видалення або редагування важливих даних.

Проте, надмірна зосередженість на дизайні та зручності використання може призвести до нехтування аспектами безпеки. Наприклад, недостатня увага до валідації введених даних може відкрити двері для атак на веб-сайт. Складний UI може збентежити користувачів і змусити їх виходити за межі безпечних практик, таких як використання слабких паролів або натискання на підозрілі посилання.

Існує п'ять рівнів на які можна розділити елементи UX/UI [2].

1. Поверхневий рівень - включає в себе веб-сторінки з картинками та текстом, з якими можна взаємодіяти за допомогою миші.
2. Рівень компоновки - визначає розташування елементів інтерфейсу, таких як кнопки, вкладки, фотографії та текстові блоки, для максимальної зручності та ефективності.
3. Рівень структури - визначає потоки користувачів та розташування елементів, наприклад, розміщення елементів на сторінці оплати товару та шляхи, якими можуть піти користувачі.
4. Рівень набору можливостей - організує функціонал сайту в єдине ціле, наприклад, збереження адреси користувача для зручності під час наступних покупок.
5. Рівень стратегії - визначає цілі власників та користувачів сайту.

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				12
Змн.	Арк.	№ докум.	Підпис	Дата		

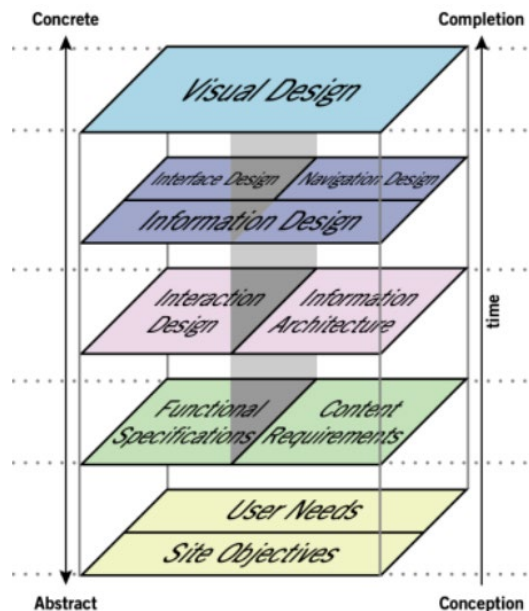


Рис. 1.1 – Елементи досвіду користувача

Вони складають концептуальну основу для обговорення проблем та засобів їх вирішення. Ці п'ять рівнів утворюють основу для обговорення проблем та їх вирішення в контексті взаємодії з користувачем. Кожен рівень залежить від рівнів, що знаходяться нижче, і непослідовність рішень може призвести до проблем.

Ця взаємозалежність значить, що рішення на стратегічному рівні викликають «хвильовий ефект» від нижніх рівнів до верхніх уздовж всього ланцюжка. З іншого боку, діапазон вибору рішень на кожному рівні обмежений рішеннями, що прийняті на нижчих рівнях. «Хвильовий ефект» означає, що виходження за межі прийнятого на верхньому рівні вимагає перегляду рішень, прийнятих на нижніх рівнях.

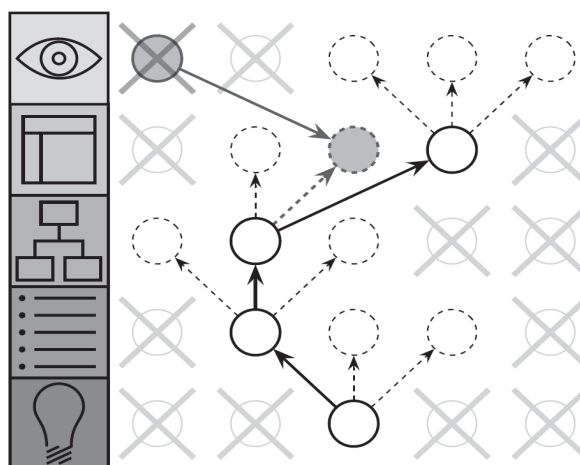


Рис. 1.2 – Демонстрація «Хвильового ефекту»

На кожному з цих рівнів є можливість впроваджувати якісь засоби безпеки. Поверхневий рівень UX/UI може включати в себе захист від кросс-сайтового скриптування (XSS) для уникнення впровадження шкідливого коду через текстові поля, а також захист від SQL-ін'єкцій для забезпечення безпеки баз даних від несанкціонованого доступу, а на рівні компоновки UX/UI важливо забезпечити коректне відображення елементів інтерфейсу на різних пристроях та розмірах екранів, а також використовувати захищені заголовки фреймів (X-Frame-Options) для запобігання clickjacking.

На рівні структури UX/UI може бути впроваджена валідація введених даних користувачів перед збереженням або обробкою, що гратиме ключову роль у запобіганні можливих атак. Також важливо використовувати механізми автентифікації та авторизації для забезпечення доступу тільки авторизованим користувачам. На рівні набору можливостей UX/UI важливо впроваджувати заходи захисту від фішингу, такі як механізми перевірки достовірності посилань та електронних листів, а також використання двофакторної аутентифікації для додаткового забезпечення безпеки облікових записів користувачів. А на рівні стратегії - визначити та реалізувати стратегії резервного копіювання та відновлення даних для запобігання втрат важливої інформації. Проведення аудиту безпеки та підтримка актуальності системи безпеки допомагає виявляти та усувати потенційні вразливості.

1.2 Функціональні обмеження інтерфейсу та управління доступом

Функціональні обмеження інтерфейсу включають в себе набір заходів, спрямованих на обмеження функціональності веб-сайту для певних користувачів або груп користувачів. Це може бути досягнуто шляхом різних методів, таких як обмеження доступу до певних функцій, приховування конфіденційної інформації користувачів або встановлення обмежень на кількість спроб введення паролів.

Наприклад, функціональні обмеження інтерфейсу можуть включати в себе обмеження прав доступу для різних рівнів користувачів, таких як адміністратори, модератори та звичайні користувачі. Це дозволяє забезпечити, що кожен

		Шмирина Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				14
Змн.	Арк.	№ докум.	Підпис	Дата		

користувач отримує лише ту функціональність, яка необхідна для виконання його завдань, та що конфіденційна інформація залишається захищеною від несанкціонованого доступу.

Дизайнер сайту може забезпечити безпеку користувачів шляхом реалізації різних заходів обмеження. Це включає в себе:

- вимогу сильних паролів та використання двофакторної аутентифікації;
- обмеження кількості спроб входу для уникнення атак перебору паролів;
- параметризовані запити до бази даних та перевірка введених даних;
- захист з'єднання з використанням протоколу HTTPS;
- обмеження типів файлів, які можуть бути завантажені на сервер;
- уникнення використання користувацького коду;
- регулярне створення резервних копій даних і моніторинг системи.

Деякі з завдань вимагає співпраці з іншими фахівцями, такими як системні адміністратори та інженери з безпеки, що забезпечують виконання цих процесів на рівні сервера та інфраструктури. Однак дизайнер може створити візуал інтерфейсу, що спрощує процес розробки. Щоб успішно реалізувати такі інтерфейси, необхідно приділяти увагу потребам та можливостям кінцевих користувачів. Ключовим в аспекті забезпечення безпеки та ефективності користувачів є використання принципів Human-Centered Design. Активне залучення користувачів до тестування прототипів дозволяє належним чином враховувати їхні потреби та вимоги. Також важливо враховувати аспекти безпеки в самому дизайні інтерфейсу, щоб уникнути вразливостей до потенційних атак. Такий підхід дозволяє забезпечити не лише функціональність, а й безпеку продукту на кожному етапі його розробки та використання.

В нормативних документах «Ergonomics of human-system interaction — Part 210: Human-centred design for interactive systems», прийнятих міжнародною організацією із стандартизації ISO, вказані такі принципи [3].

1. Дизайн має базуватися на явному розумінні користувачів, завдань та середовищ.
2. Користувачі мають бути залучені до всіх етапів дизайну та розробки.

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				15
Змн.	Арк.	№ докум.	Підпис	Дата		

3. Дизайн має здійснюватися та вдосконалюватися на основі оцінки, орієнтованої на користувача.
4. Процес має бути ітеративним і включати повторення усіх кроків розробки, з метою вдосконалення продукту.
5. Дизайн продукту або послуги повинен враховувати всі аспекти користувацького досвіду, включаючи взаємодію, зручність використання, емоційний відгук та якість обслуговування.
6. Команда дизайну включає різноманітні навички та перспективи.

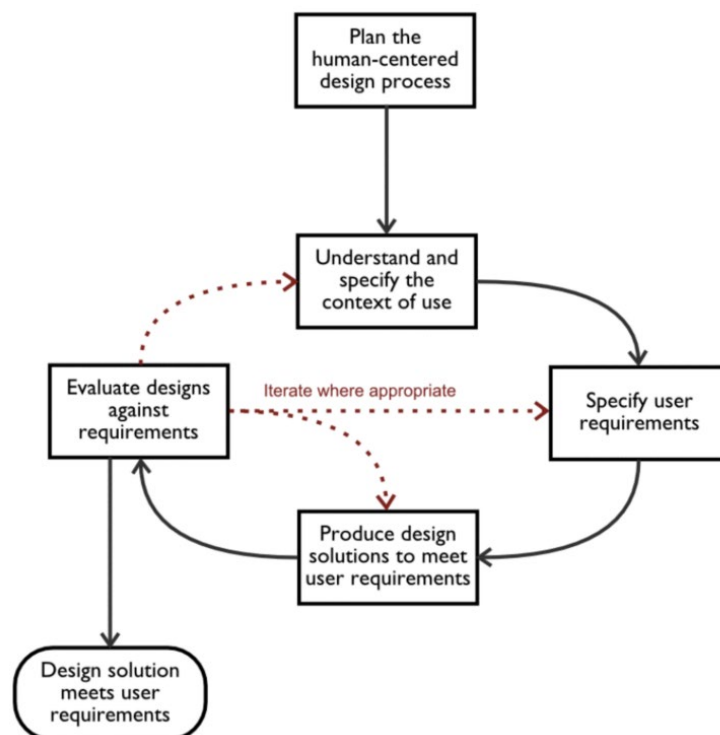


Рис. 1.3 – Схема послідовності дій при плануванні розробки людиноцентрованого проєкту дизайн-системи

Людиноцентрований дизайн сприяє захисту даних користувача, враховуючи їхні потреби та бажання щодо приватності. Це означає, що дизайн продукту чи послуги створює інтерфейси, які дозволяють користувачам контролювати свої персональні дані та забезпечують прозорість їх обробки. Це може включати реалізацію налаштувань конфіденційності, можливість вибору рівня доступу та використання шифрування для захисту інформації. Залучення користувачів до процесу розробки та навчання правил безпеки також є ключовим.

Управління доступом до різних частин веб-сайту або його функцій забезпечує контроль над доступом, що гарантує тільки авторизованим користувачам доступ до конфіденційної інформації та функціональності, яка впливає на безпеку системи. Наприклад, використання системи аутентифікації з двофакторним підтвердженням на веб-сайтах забезпечує додатковий рівень захисту, вимагаючи від користувачів підтвердження своєї особи через додатковий фактор, такий як код або відбиток пальця. Крім того, важливо забезпечити стале оновлення системи безпеки та надавати користувачам інформацію про потенційні загрози та правильні методи захисту їхніх даних.

1.3 Управління типами дозволених файлів як аспект безпеки

Управління типами дозволених файлів дозволяє зменшити ризик вразливостей і зловживань, забезпечуючи контроль над тим, які типи файлів можуть бути завантажені на сервер та як вони обробляються. Недостатньо або неправильно налаштовані обмеження щодо типів дозволених файлів можуть відкрити шлях до атак, таких як впровадження шкідливого коду, перехоплення інформації або навіть взлом системи сайту чи додатку.

Натомість, правильне управління типами дозволених файлів передбачає встановлення та дотримання заходів безпеки як на клієнтській, так і на серверній стороні. Це включає в себе обмеження типів файлів, які можуть бути завантажені користувачами, а також валідацію цих файлів на відповідність прийнятним стандартам безпеки.

Запобігання вразливостям в процесі завантаження файлів є важливим завданням для розробників веб-додатків, оскільки це дозволяє забезпечити надійність, конфіденційність та цілісність даних, які обробляються системою. Враховуючи швидкі темпи розвитку технологій та поширення інтернету, важливо постійно оновлювати та вдосконалювати підходи до управління безпекою файлів^[5] у веб-додатках.

Детальніше розглянемо деякі ключові стратегії та практики управління типами дозволених файлів у веб-інтерфейсах.

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				17
Змн.	Арк.	№ докум.	Підпис	Дата		

- Використання HTML атрибуту асепт у тегу <input type="file">, щоб обмежити типи файлів, які може вибрати користувач при завантаженні.
- Перевірка типів файлів на сервері після завантаження та використання бібліотеки чи фреймворку, які забезпечать перевірку MIME-типу файлу.
- Встановлення допустимого розміру файлу для завантаження, щоб запобігти завантаженню великих файлів, які можуть спричинити перевищення обсягу пам'яті або завантаження сервера.
- Перевірка імен файлів на наявність недопустимих символів або шаблонів, які можуть використовуватися для атак на веб-додатки
- Заборона завантаженим файлам виконувати будь-які дії на сервері без необхідної перевірки та санітизації, включаючи виконання JavaScript-коду з завантажених HTML або JavaScript файлів.
- Проведення регулярних аудитів безпеки веб-додатку для виявлення потенційних вразливостей.

Управління типами дозволених файлів у веб-інтерфейсах є критичним аспектом забезпечення захисту веб-додатків. Використання відповідних стратегій та практик може запобігти вразливостям і зловживанням, пов'язаним із завантаженням файлів користувачами. Обмеження типів файлів на стороні клієнта та перевірка їх на стороні сервера, разом з іншими заходами, такими як обмеження розміру файлу та валідація імен файлів, допомагають забезпечити захист та цілісність веб-додатків.

Регулярне оновлення та аудит безпеки важливі для виявлення та виправлення нових вразливостей у веб-інтерфейсах. До цього слід додати постійне вдосконалення підходів до управління безпекою файлів у відповідь на постійну появу нових технологічних та кібербезпекових викликів. Запровадження цих практик допоможе забезпечити надійність та захищеність ваших веб-додатків від різноманітних загроз та атак, пов'язаних із завантаженням файлів користувачами.

		Шмирина Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				18
Змн.	Арк.	№ докум.	Підпис	Дата		

Таким чином, вироблення та впровадження належних стратегій управління типами дозволених файлів варто розглядати як одну з важливих складових безпеки веб-додатків.

Існують також деякі типи даних, які можуть становити загрозу безпеці програмного забезпечення та даних на сервері, якщо їх дозволено завантажувати користувачами на сайт. Такі типи даних включають [4]:

- .exe, .dll, .bat, .sh, .jar та інші, що можуть містити виконуваний код, представляючи потенційну загрозу безпеці.
- .php, .asp, .cgi, .py, .rb тощо, можуть містити скриптовий код, який виконається на сервері і може бути використаний для атак типу "ін'єкція коду" (code injection);
- .exe або скомпільовані файли .js можуть містити шкідливий код або виконувати небезпечні дії;
- .zip, .rar, .tar.gz тощо можуть містити виконуваний код та бути небезпечними, якщо цей архів розпакується та виконається на сервері;
- .swf можуть містити в собі шкідливий код, який використовує вразливості в плеєрах або веб-браузерах;
- .exe, .bat, .cmd (для Windows), .app (для MacOS) не повинні завантажуватися на сервер, оскільки вони можуть становити загрозу для безпеки системи.

Заборона завантаження цих типів файлів допоможе запобігти можливим атакам, а також захистить сайт та дані на сервері від неправомірного доступу або пошкодження. Шляхом встановлення чітких правил щодо типів дозволених файлів та використання відповідних механізмів перевірки та валідації, можна забезпечити надійність та безпеку сайту.

Висновки з розділу 1

Веб-сайти сьогодні є не лише способом надання інформації, вони стали інтерактивними платформами, де користувачі можуть взаємодіяти з компаніями через різноманітні інтерфейси. Важливою складовою успішного веб-сайту є висока

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				19
Змн.	Арк.	№ докум.	Підпис	Дата		

якість користувацького досвіду (UX) та інтерфейсу користувача (UI), однак, покращення цих аспектів часто супроводжується зниженням рівня безпеки.

Важливість UX полягає в тому, що він включає в себе зручність навігації, швидкість завантаження сторінок, зрозумілість інформації та використання різних функцій. Інтерфейс користувача (UI), зі свого боку, забезпечує інтуїтивно зрозуміле та привабливе взаємодію користувачів з веб-сайтом. Правильно розроблений UI спрощує користування веб-сайтом та забезпечує приємний користувацький досвід, однак, надмірна зосередженість на дизайні може призвести до нехтування безпекою.

На основі проведеного аналізу можна зробити висновок, що необхідно дотримуватися балансу між зручністю користування та безпекою. Використання методів UX/UI дизайну, таких як дослідження користувачів, побудова карти шляху користувача, структурування інтерфейсу та аналіз конкурентів, допомагають створити ефективні та безпечні веб-сайти. Важливо також враховувати п'ять рівнів UX/UI, які дозволяють детально аналізувати та впроваджувати безпечні рішення на всіх етапах розробки.

Забезпечення безпеки веб-сайтів включає в себе низку заходів, від захисту від кросс-сайтового скриптування (XSS) та SQL-ін'єкцій до впровадження механізмів аутентифікації та авторизації. Впровадження функціональних обмежень інтерфейсу, таких як вимога сильних паролів, обмеження спроб входу та перевірка введених даних, також грають ключову роль у забезпеченні безпеки.

Ключовим аспектом є використання принципів людиноцентрованого дизайну, що дозволяє враховувати потреби користувачів та забезпечувати безпеку їх даних. Управління типами дозволених файлів є ще одним важливим аспектом безпеки, який дозволяє запобігти вразливостям, пов'язаним із завантаженням шкідливих файлів.

Отже, для створення безпечних і зручних у користуванні веб-сайтів необхідно комплексно підходити до розробки, враховуючи як UX/UI, так і аспекти безпеки. Такий підхід дозволяє створювати продукти, що не лише задовольняють потреби користувачів, але й забезпечують надійний захист їх даних.

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				20
Змн.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 2 ПРОЄКТУВАННЯ САЙТУ МУЗЕЮ НАУКИ І ТЕХНОЛОГІЙ «ПОЛІТЕХНІК»

2.1 Проєктування фронтенду для сайту університету

2.1.1 Заходи безпеки під час проєктування дизайну інтерфейсу

Впровадження принципів кібербезпеки на етапі проєктування UX/UI дизайну є ключовим фактором для створення стійкого та надійного веб-сайту. Цей розділ описує основні заходи безпеки, які слід врахувати при проєктуванні інтерфейсу користувача. Такі заходи можна умовно поділити на наступні три групи.

1. Ідентифікація та аналіз загроз

Першим кроком до забезпечення кібербезпеки веб-сайту є ідентифікація та аналіз можливих загроз^[6]. Це передбачає:

- вивчення поширених типів кібератак, таких як фішинг, DDoS-атаки, ін'єкції SQL, XSS-атаки тощо;
- оцінювання ймовірності та впливу кожної загрози, щоб сфокусувати свої зусилля на найсуттєвіших ризиках;
- визначення цінних даних, які потребують захисту, наприклад, особиста інформація користувачів, інформація про продукти або послуги, фінансові дані або інтелектуальна власність.

2. Принципи дизайну, орієнтовані на безпеку

При проєктуванні інтерфейсу користувача важливо використовувати принципи дизайну, орієнтовані на безпеку, зокрема наступні.

- надання користувачам лише тієї інформації, яка їм дійсно потрібна для виконання завдання;
- полегшення розпізнавання потенційно небезпечних елементів, таких як посилання або кнопки, які можуть призвести до небажаних дій;
- створення кількох рівнів захисту, щоб навіть у разі успішної атаки на один рівень інші залишалися неушкодженими;
- шифрування даних, що передаються та зберігаються, щоб захистити їх від несанкціонованого доступу.

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		21

3. Інтеграція з інструментами кібербезпеки

Окрім застосування принципів безпечного дизайну, важливо також інтегрувати з веб-сайтом інструменти кібербезпеки, зокрема наступні:

- брандмауери та системи запобігання вторгненням (IDS/IPS) допоможуть блокувати несанкціонований доступ та шкідливий трафік;
- система керування доступом IAM дозволяє контролювати, хто має доступ до веб-сайту та які дії вони можуть виконувати;
- застосовувати протоколи HTTPS та SSL/TLS для захисту даних, що передаються між користувачами та вашим веб-сайтом;
- додатковий рівень безпеки 2FA вимагає від користувачів ввести не лише пароль, але й інший фактор аутентифікації, наприклад, код, який надсилається на їхній телефон.

2.1.2 Дослідження методів UX/UI для проєктування безпечного сайту

Для проєктування сайту важливо розуміти, хто буде користувачем сайту музею науки і технологій «Політехнік», для цього також можемо використати метод UX досліджень User Profile^[7]. У межах даного дослідження було здійснено інтерв'ювання працівників музею «Політехнік», а також враховано власний досвід взаємодії з музеєм. Наведемо три приклади User Profile.

1. Юлія, 12 років:

- **Опис** — учениця 7 класу, яка цікавиться наукою та технологіями, любить читати про нові винаходи та відвідувати наукові виставки, мріє стати інженеркою.
- **Потреби** — інформація про експонати музею та історію науки.
- **Мотивація** — дізнатися більше про технології, розважитися та отримати нові знання.
- **Користування сайтом** — знайти інформацію про експонати музею, дізнатися про історію науки та технологій, подивитися відео та фотографії експонатів.

2. Олег, 30 років:

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				22
Змн.	Арк.	№ докум.	Підпис	Дата		

- **Опис** — викладач фізики старших класів, який пристрасно цікавиться наукою та технологіями. Він прагне зацікавити своїх 11-класників у вступі до закладів вищої освіти, що пропонують освітні програми технологічного спрямування.
- **Потреби** — інформація про інноваційні експонати музею, щоб продемонструвати його учням реальні застосування науки та технологій.
- **Мотивація** — надихнути своїх учнів на вивчення STEM-дисциплін та підготувати їх до успішного життя в 21 столітті.
- **Користування сайтом** — знайти інформацію про експонати музею, для підготовки нових навчальних матеріалів, які він зможе використовувати на своїх уроках, та контакти представників музею.

3. Олена, 65 років:

- **Опис** — пенсіонерка, яка любить відвідувати музеї з онуками, цікавиться історією науки та технологій.
- **Потреби** — знайти на сайті інформацію про експонати музею, які їй цікаві, дізнатися більше про історію науки та технологій.
- **Мотивація** — провести час з сім'єю та поділитися з ними своїми знаннями.
- **Користування сайтом** — знайти інформацію про місцезнаходження музею, переглянути розклад подій, подивитися відео та фотографії експонатів.

Кожен з користувачів цікавиться експонатами, фото та відео. Виходячи з даної інформації можемо зробити висновок, що сайт повинен мати наступні розділи.

Головна сторінка:

- Вітальний банер з описом музею.

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				23
Змн.	Арк.	№ докум.	Підпис	Дата		

- Коротка інформація про музей
- Посилання на розділи «Про музей», «Експозиції», «Події» та «Запис».
- Кнопка «Записатися на екскурсію».

Про музей:

- Історія музею.
- Місія та цінності.
- Команда музею.
- Партнери.

Експозиції:

- Перелік експозицій з коротким описом.
- Інформація про експонати.

Події:

- Анонси та новини про попередні заходи.
- Тематичні лекції та семінари.
- Екскурсії та майстер-класи.

Екскурсії:

- Правила відвідування.
- Графік роботи.
- Контактні телефони.
- Анкета для запису на екскурсію.

Запис:

- Анкета.
- Вільний час для екскурсій.

Щоб краще проілюструвати принципи безпечного дизайну та інтеграції інструментів кібербезпеки, розглянемо User Flow^[8] на прикладі веб-сайту Музею науки і технологій «Політехнік».

User Flow описує, як користувач взаємодіє з продуктом, щоб досягти певної мети та допомагають візуалізувати:

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		24

- логічний маршрут користувачів;
- взаємозв'язки між функціями, діями та наслідками;
- проблемні місця в UX.

Шлях користувача веб-сайту інтерактивного музею науки, базуючись на потребах користувачів та визначених розділах може мати ось такий вигляд:

1. Введення в пошуку браузера «Інтерактивний музей науки і техніки «Політехнік».
2. Відвідування головної сторінки веб-сайту, перегляд банеру та основного меню.
3. Перехід до сторінок «Про музей», «Експонати», «Екскурсії» або «Події».
4. Дослідження сторінок із детальною інформацією про музей.
5. Запис за телефоном або на сайті, що передбачує майбутнє фізичне відвідування музею.

Візуальне подання послідовності дій користувача під час дослідження сайту музею наведено на рис. 2.1.

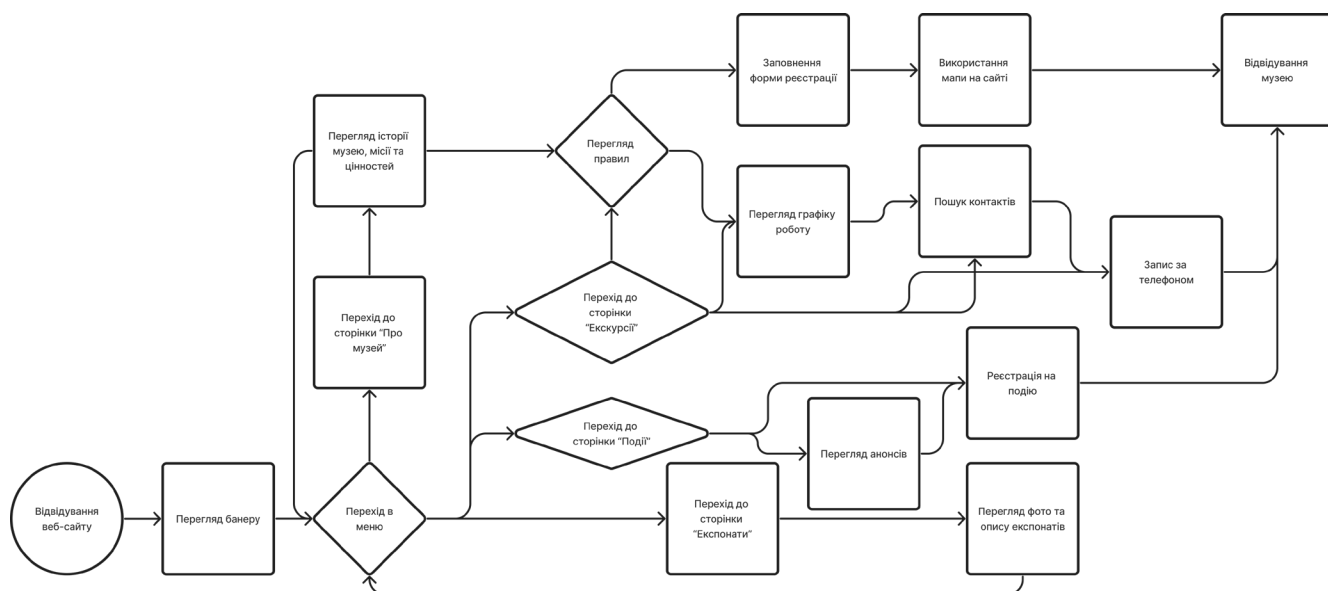


Рис. 2.1 – Блок-схема User Flow веб-сайту інтерактивного музею

2.1.3 Проектування та реалізація візуальної складової проєкту

Після реалізації User Flow веб-сайту ми можемо чітко зрозуміти, які сторінки та елементи інтерфейсу необхідно створити для комфортного сценарію

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				25
Змн.	Арк.	№ докум.	Підпис	Дата		

користування сайту користувачем. Це дозволяє перейти до створення Wireframes, які відображатимуть структуру та компоненти кожної сторінки, а також надають можливість детально опрацювати розташування елементів інтерфейсу, забезпечуючи зручність і логічність навігації для користувачів. Вони є першим кроком у візуалізації майбутнього дизайну і служать основою для подальшої роботи над UI.

Wireframes^[9] — це ескізи сторінок, що відображають структуру веб-сторінки або мобільного додатку. Вони використовуються для планування розташування елементів інтерфейсу користувача (UI), таких як заголовки, текстові блоки, кнопки, форми, зображення та навігаційні панелі. Метою wireframes є візуалізація дизайну на ранніх етапах проєктування без відволікання на кольори, стилі чи точні деталі. Це дозволяє зосередитися на функціональності та взаємодії користувачів з системою.

Wireframes є важливим етапом у процесі UX-дизайну, оскільки вони:

- допомагають зрозуміти, як буде виглядати та функціонувати кінцевий продукт;
- сприяють обговоренню та узгодженню між дизайнерами, розробниками та замовниками;
- дають змогу виявити потенційні проблеми та внести зміни на ранніх стадіях проєктування;
- економлять час і ресурси, адже зміни в макеті легше внести на етапі Wireframe, ніж у завершеному дизайні.

Головна сторінка

У верхній частині сторінки розміщений вітальний банер для привітання відвідувачів і надана логотип музею. Нижче від банеру знаходиться секція з основною інформацією. Посилання на розділи «Про музей», «Експозиції», «Події» та «Запис» є у вигляді кнопок або вкладок для швидкого доступу. Кнопка «Записатися на екскурсію» розташована видимо і доступно для відвідувачів (рис. 2.2).

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		26

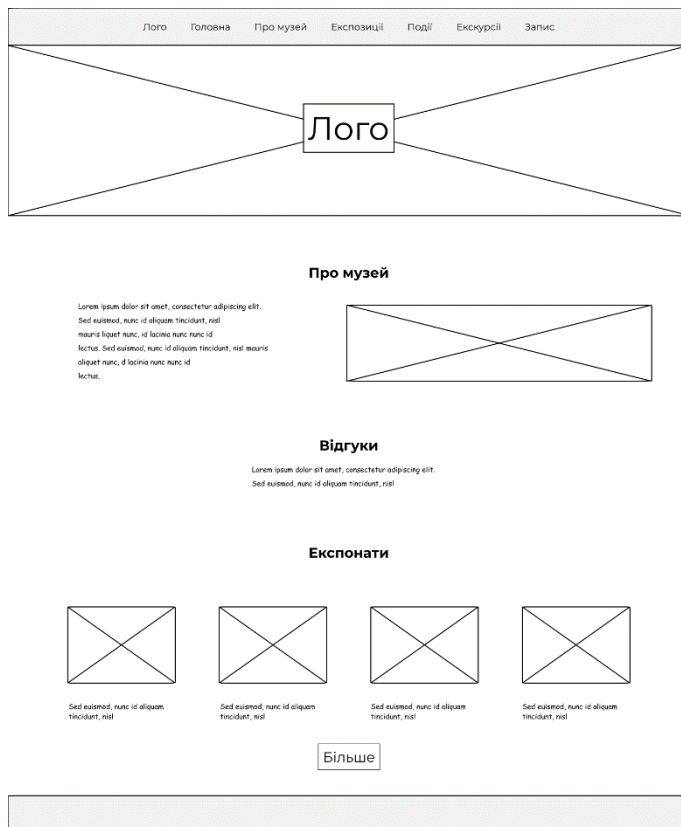


Рис. 2.2. – Wireframe головної сторінки

Про музей

Розділ з текстом про історію створення та розвитку музею. Опис місії та цінностей, яких дотримується музей. Інформація про ключових співробітників музею. Перелік партнерів (рис. 2.3).

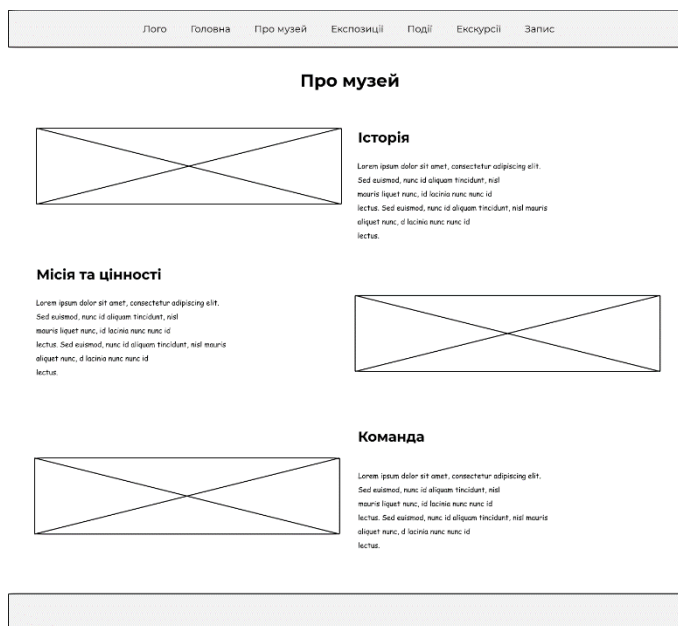


Рис. 2.3. – Wireframe сторінки про музей

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головна О.С.				27
Змн.	Арк.	№ докум.	Підпис	Дата		

Експозиції

Відображає назви та короткий опис кожної експозиції. Детальна інформація про окремі експонати (рис. 2.4).

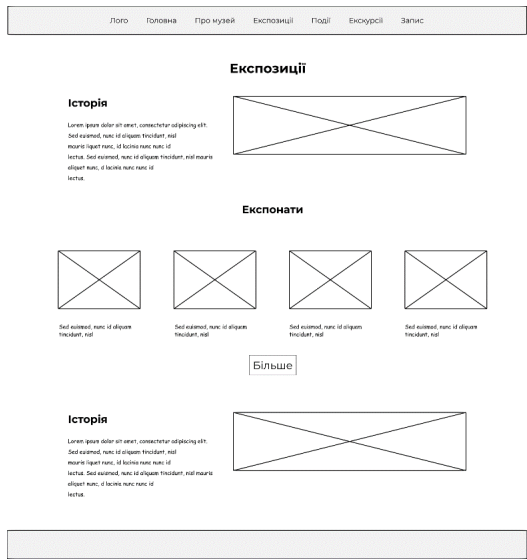


Рис. 2.4. – Wireframe сторінки експозицій

Події

Список майбутніх заходів з можливістю перегляду минулих подій. Опис запланованих тематичних лекцій та семінарів. Інформація про екскурсії та майстер-класи (рис. 2.5).

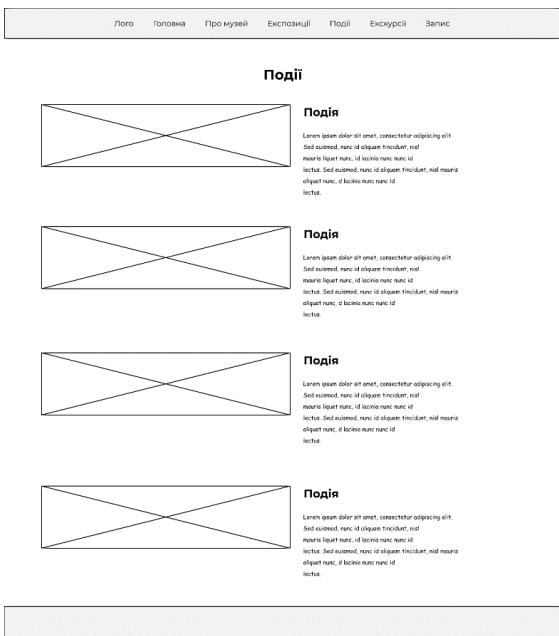


Рис. 2.5. – Wireframe сторінки подій

Екскурсії

Розділ з правилами поведінки та відвідування музею. Інформація про години роботи музею. Номери телефонів для зв'язку. Форма для заповнення відвідувачами (рис. 2.6).

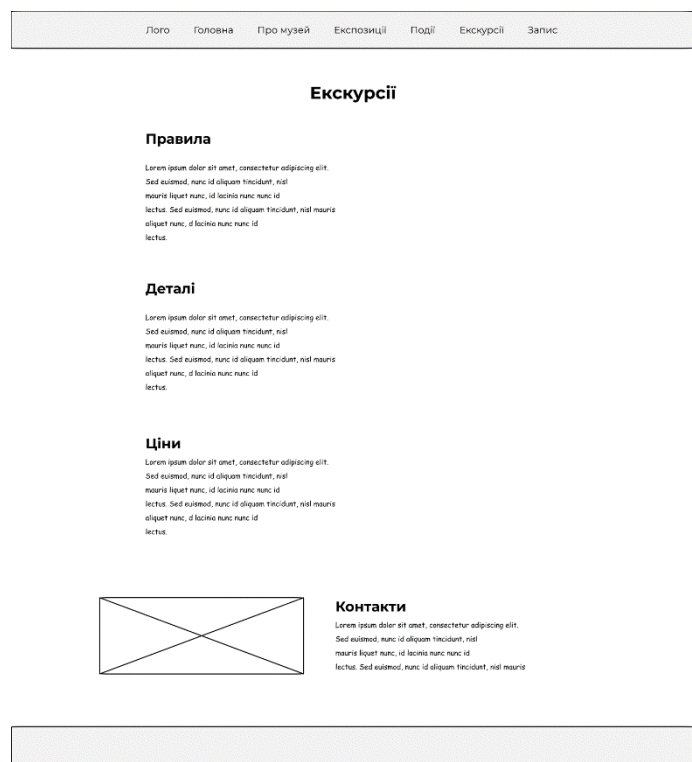


Рис. 2.6. – Wireframe сторінки екскурсій

Запис

Форма для заповнення інформації про відвідувача. Список доступного часу для екскурсій (рис. 2.7).

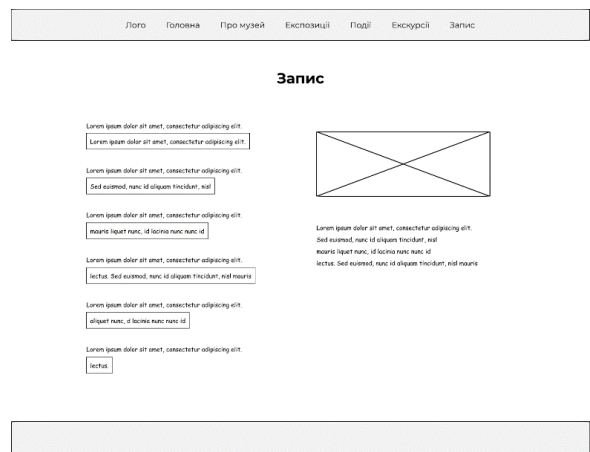


Рис. 2.7. – Wireframe сторінки запису

Тепер, коли ми маємо чітке уявлення про структуру і компоненти кожної сторінки, наступним кроком є перетворення цих базових ескізів у повноцінні дизайн-макети. Цей процес включає вибір кольорової палітри, типографіки, стилів

кнопок і форм, а також інших візуальних елементів, що зроблять сайт естетично привабливим і зручним для користувачів.

На сайті бренд-системи університету^[10] кольорами університету вказані HTML коди — #224A98 (синій), #E53E23 (червоний), #EE781C (помаранчевий), і #4CAD3B (зелений). Головним є відтінок синього, його ми і візьмемо за основний колір сайту. Шрифтом, що використовується в айдентиці, є Montserrat Regular для звичайного тексту і Bold для заголовків.

Дизайн-макети дозволяють більш детально опрацювати візуальну складову, з урахуванням брендингу та загального стилю музею. Вони допомагають побачити, як буде виглядати кінцевий продукт, і внесення коригувань на цьому етапі набагато простіше, ніж на стадії розробки.

Вигляд ескізів пропрацьованим дизайн макетом продемонстровано на рис. 2.8-2.11.

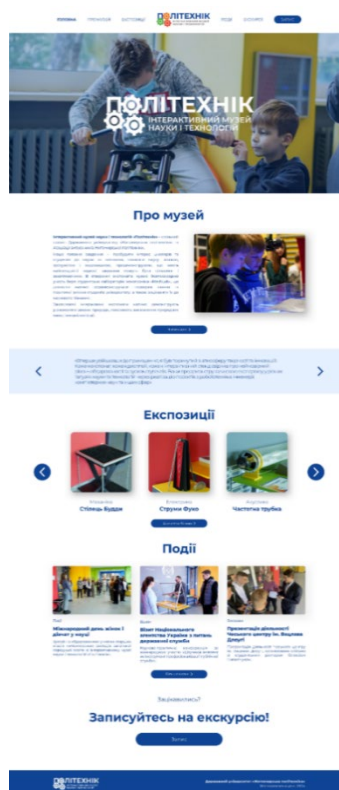


Рис. 2.8. – Дизайн сторінки запису

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				30
Змн.	Арк.	№ докум.	Підпис	Дата		



Рис. 2.9. – Дизайн сторінки про музеї



Рис. 2.10. – Дизайн сторінки подій

		Шмиріна Т.С.			Арк.
		Головня О.С.			31
Змн.	Арк.	№ докум.	Підпис	Дата	



Рис. 2.11. – Дизайн сторінки екскурсій

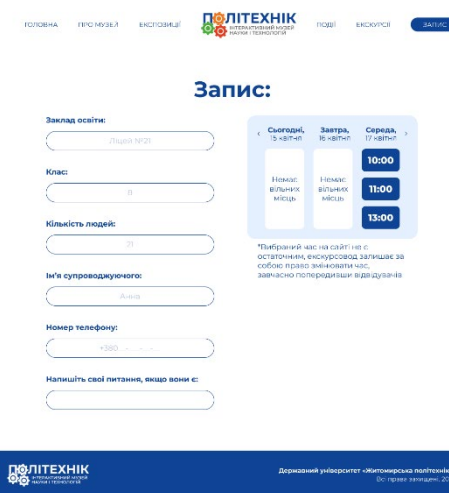


Рис. 2.11. – Дизайн сторінки запису на екскурсію

Методи UX досліджень перед розробкою веб-сайту є критично важливим етапом, що забезпечує створення логічної та зручної структури сайту, ефективну взаємодію між усіма учасниками проєкту та виявлення потенційних проблем на ранніх стадіях. Це дозволяє економити час і ресурси, гарантуючи, що кінцевий дизайн буде естетично привабливим, функціональним і безпечним для користувачів. Ретельне проєктування, включаючи визначення User Flow, створення wireframes і розробку дизайн-макетів, забезпечує високий рівень відповідності потребам користувачів і цілям організації.

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		32

2.2 Проєктування бекенду веб-сайту музею

2.2.1 Постановка завдань

Задачею даного підрозділу є проєктування бекенду для сайту музею університету, яка забезпечить ефективне управління експонатами, інформаційними ресурсами та взаємодію з відвідувачами. Основні вимоги до бекенду сайту – це висока швидкість обробки даних та безпечність використання.

Ефективне проєктування системи бекенду сайту є високопріоритетним завданням, особливо в контексті управління даними, де безпека, надійність та швидкодія грають ключову роль. Забезпечення швидкої реакції системи на запити користувачів є важливим аспектом, оскільки система повинна демонструвати миттєвий відгук на дії адміністратора та відвідувачів, уникнувши затримок або сповільнення, що можуть виникати під час виконання алгоритмів.

У цьому контексті важливо використати сучасні інструменти для управління даними. Для цієї системи доцільно застосувати фреймворк Laravel, який надає зручні засоби для моделювання взаємодії з базою даних та забезпечує високу продуктивність і безпеку. Використання Laravel допоможе оптимізувати виконання операцій і забезпечить ефективне управління даними, підвищуючи загальну продуктивність системи.

Встановлення задач:

1. Визначення основних функцій комп'ютерної системи.

2. Аналіз вимог:

2.1 Система адміністрування музею університету.

2.2 Музей надає можливість управління експонатами, створення каталогів та описів.

2.3 Роль “Адміністратор” має можливість додавати нові експонати, редагувати наявні, створювати та змінювати описи експонатів, а також керувати доступом до інформації.

2.4 Як відвідувач маю можливість переглядати каталоги експонатів, шукати інформацію про конкретні експонати та брати участь у віртуальних турах.

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		33

2.5 Адміністратор музею може додавати інших користувачів з різними рівнями доступу, наприклад, наукових співробітників, які можуть доповнювати інформацію про експонати.

2.6 Кожен експонат має свої метадані, включаючи назву, опис, дату додавання, автора тощо.

3. Проаналізувати способи реалізації вимог.

4. Спроектувати загальний алгоритм взаємодії класів між собою та елементами інтерфейсу.

Конфігурування AWS S3 бакету

Конфігурування AWS S3 бакету:

1. Створити S3 бакет на Amazon Web Services для зберігання зображень експонатів.
2. Налаштувати права доступу для безпечного збереження та отримання зображень.
3. Інтегрувати S3 бакет з системою адміністрування, використовуючи SDK AWS для PHP.
4. Реалізувати функції завантаження, зберігання та отримання зображень з AWS S3.

Розгортання мікросервісної архітектури за допомогою Docker та Docker Compose

Розгортання мікросервісної архітектури:

1. Визначити необхідні мікросервіси для системи адміністрування музею (наприклад, сервіс управління експонатами, сервіс користувачів, сервіс зображень).
2. Написати Dockerfile для кожного мікросервісу, включаючи необхідні залежності та налаштування середовища.
3. Налаштувати Docker Compose файл для запуску та координації всіх мікросервісів.
4. Створити Docker мережі (network) для забезпечення безпечної комунікації між мікросервісами.

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				34
Змн.	Арк.	№ докум.	Підпис	Дата		

5. Визначити та налаштувати Docker томи (volume) для зберігання даних бази даних та інших необхідних файлів.
6. Забезпечити масштабованість та високу доступність системи шляхом налаштування реплікації сервісів та балансування навантаження.

Реалізація:

Реалізація:

1. Реалізувати класи для управління експонатами, користувачами та іншими сутностями системи.
2. Забезпечити захист даних і контроль доступу.
3. Оптимізувати швидкість виконання запитів до бази даних.
4. Реалізувати інтерфейс користувача для адміністратора та відвідувачів.
5. Провести тестування системи на різних етапах розробки, включаючи модульні, інтеграційні та функціональні тести.

Створення комп'ютерної системи для адміністрування музею університету включає багатоетапний процес планування, проєктування, реалізації та тестування, який включає інтеграцію сучасних технологій, таких як AWS S3 для зберігання зображень та Docker для мікросервісної архітектури, забезпечуючи високу продуктивність, надійність та безпеку системи.

2.2.2 Обґрунтування вибору інструментів та вимог до апаратного забезпечення

Для розробки бекенду сайту музею університету було обрано наступні інструментальні засоби: Symfony, Nginx, AWS S3, Docker, MySQL та phpMyAdmin. Кожен з цих інструментів відіграє важливу роль у забезпеченні функціональності, надійності та ефективності системи. Нижче наведено обґрунтування вибору цих засобів.

Symfony

Symfony є одним із найпотужніших та найбільш гнучких PHP фреймворків. Він надає широкий набір компонентів та бібліотек для швидкої розробки веб-додатків. Основні причини вибору Symfony включають:

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		35

1. Модульність та повторне використання коду: Symfony забезпечує високу модульність, дозволяючи використовувати окремі компоненти в різних частинах додатка, що спрощує підтримку та розвиток проєкту.

2. Безпека: Фреймворк надає вбудовані механізми для захисту від поширених веб-уразливостей, таких як SQL-ін'єкції та XSS-атаки.

3. Продуктивність: Symfony оптимізований для швидкого виконання, що є важливим при роботі з великою кількістю даних.

Nginx

Nginx обраний в якості веб-сервера завдяки його високій продуктивності та надійності. Основні причини вибору включають:

1. Висока продуктивність: Nginx здатний обробляти велику кількість одночасних з'єднань з низькими витратами на апаратні ресурси.

2. Гнучкість налаштувань: Веб-сервер підтримує широкий спектр конфігурацій для обробки статичних файлів, реверсного проксування та балансування навантаження.

3. Безпека: Nginx надає потужні механізми для захисту від DDoS-атак та інших загроз.

AWS S3

Amazon S3 використовується для зберігання зображень експонатів музею. Основні причини вибору включають:

1. Масштабованість: S3 дозволяє зберігати необмежену кількість даних, забезпечуючи високу доступність і масштабованість.

2. Безпека: AWS надає потужні засоби для контролю доступу та шифрування даних.

3. Інтеграція: Легкість інтеграції з іншими сервісами AWS та можливість використання SDK для роботи з S3 в середовищі PHP.

Docker

Docker обраний для розгортання мікросервісної архітектури системи. Основні причини вибору включають:

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		36

1. Контейнеризація: Docker дозволяє ізолювати додатки та їх залежності в контейнерах, що спрощує управління середовищем виконання.

2. Портативність: Контейнери Docker можуть працювати на будь-якій платформі, що підтримує Docker, забезпечуючи однакову поведінку на різних середовищах.

3. Масштабованість та керованість: Використання Docker Compose дозволяє легко керувати багатоконтейнерними додатками, визначаючи їх взаємодію та залежності.

MySQL

MySQL обраний в якості системи керування базами даних завдяки його надійності та продуктивності. Основні причини вибору включають:

1. Продуктивність: MySQL забезпечує високу швидкість обробки запитів, що є важливим для забезпечення швидкодії системи.

2. Масштабованість: MySQL підтримує горизонтальне та вертикальне масштабування для обробки великої кількості даних.

3. Безпека: Вбудовані механізми безпеки дозволяють захистити дані від несанкціонованого доступу.

phpMyAdmin

phpMyAdmin використовується для адміністрування бази даних MySQL. Основні причини вибору включають:

1. Зручність: phpMyAdmin надає зручний веб-інтерфейс для управління базами даних, виконання запитів та аналізу даних.

2. Функціональність: Інструмент підтримує широкий набір функцій для роботи з базами даних, включаючи резервне копіювання та відновлення даних.

3. Інтеграція: Легко інтегрується з MySQL і дозволяє адміністраторам ефективно керувати базою даних.

Вимоги до апаратного забезпечення

Для забезпечення стабільної роботи системи обрано наступну конфігурацію апаратного забезпечення (як локального, так і у хмарі):

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		37

1. Сервер:

- Процесор: багатоядерний процесор (мінімум 4 ядра)
- Оперативна пам'ять: мінімум 16 ГБ
- Накопичувач: SSD-накопичувач з мінімум 500 ГБ місця
- Мережа: високошвидкісне з'єднання з інтернетом

2. Розподілене середовище:

- Сервери для контейнерів Docker з достатньою кількістю ресурсів для розгортання мікросервісів
- Об'єктне сховище AWS S3 для зберігання зображень

3. Резервне копіювання:

- Відокремлений сервер або хмарне сховище для резервного копіювання баз даних та важливих файлів

Обрані інструментальні засоби та відповідне апаратне забезпечення дозволять створити ефективну та надійну систему адміністрування музею університету, що відповідатиме всім вимогам безпеки, швидкодії та зручності використання.

2.2.3 Проектування архітектури

Архітектура додатку для сайту музею університету базується на мікросервісній моделі, яка забезпечує високий рівень масштабованості, гнучкості та ізоляції компонентів. Ось основні елементи архітектури та їх взаємодія:

Сервіси

У проєкті використовується кілька контейнерів Docker, кожен з яких виконує окрему функцію:

1. API - Цей контейнер відповідає за обробку запитів до серверної частини додатку.

- Технології: PHP (Symfony), Xdebug для відладки.
- Конфігурація: Включає конфігурацію Xdebug для налагодження.
- Мережа: Підключений до мережі `courselab` з статичною IP-адресою.

2. Frontend - Цей контейнер відповідає за обробку запитів до клієнтської частини додатку.

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		38

- Технології: Node.js, React.

- Конфігурація: Використовує pm2 для запуску сервера.

- Мережа: Підключений до мережі `courselab` з статичною IP-адресою.

3. Nginx - Цей контейнер працює як зворотний проксі-сервер для API та Frontend.

- Технології: Nginx.

- Конфігурація: Серверні сертифікати для SSL, налаштування для проксі-форвардингу.

- Мережа: Підключений до мережі `courselab` з статичною IP-адресою.

4. MySQL - СУБД для збереження даних додатку.

- Технології: MySQL.

- Конфігурація: Використовує об'єм для зберігання даних.

- Мережа: Підключений до мережі `courselab` з статичною IP-адресою.

5. PhpMyAdmin - Інтерфейс для адміністрування бази даних MySQL.

- Технології: PhpMyAdmin.

- Конфігурація: Підключається до MySQL контейнера.

- Мережа: Підключений до мережі `courselab` з статичною IP-адресою.

Мережа та Volumes

1. Мережа (courselab):

- Всі сервіси підключені до однієї мережі `courselab`, що дозволяє їм взаємодіяти між собою.

- Використовується статична IP-адресація для зручності налаштування та підключення між сервісами.

2. Volumes:

- mysql: Для зберігання даних MySQL.

- api: Для зберігання файлів та конфігурацій API.

- nginx: Для зберігання конфігурацій Nginx та сертифікатів SSL.

- data: Використовується для зберігання загальних даних, якщо потрібно.

Конфігурація Nginx

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		39

- Nginx працює як зворотний проксі для перенаправлення запитів до відповідних сервісів.
- Використовується SSL для забезпечення безпеки з'єднань.
- Налаштування кешування та обробка статичних файлів для підвищення продуктивності.

Встановлення та запуск

1. Docker Compose Використовується для керування всіма сервісами через файл `docker-compose.yml`.
2. Build: Кожен сервіс має свій Dockerfile для налаштування середовища виконання.
3. Запуск: Команда docker-compose up піднімає всі сервіси, забезпечуючи їх взаємодію та належну роботу.

Ось детальний опис мережі Docker universityMuseum:

Основні властивості мережі universityMuseum

- Name: universityMuseum
- Id: 581e822bf67a786e925ecbdd0d4edd43e93f6bea832a97d54d401bc4adb7eea2
- Created 2024-05-21T20:45:43.951197339Z
- Scope: local
- Driver: bridge
- EnableIPv6: false

IPAM (IP Address Management)

- Driver: default
- Config:
- Subnet: 172.22.75.0/16
- Gateway: 172.22.0.1

Властивості

- Internal: false
- Attachable: false
- Ingress: false
- ConfigFrom:

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				40
Змн.	Арк.	№ докум.	Підпис	Дата		

- ConfigOnly: false

Контейнери, підключені до мережі **universityMuseum**

1. courselab-nginx

-

EndpointID:

5ba74f2ef7122957f2b94589a2f63ed52294c2075ef76f4d131787e5e5d1d234

- MacAddress: 02:42:ac:16:4b:02

- IPv4Address: 172.22.75.2/16

2. courselab-phpmyadmin

-

EndpointID:

40104eab4a747212cf842e0ad5fc5823e596f1a37e88c7eb81d5d97c9d06677a

- MacAddress: 02:42:ac:16:4b:0a

- IPv4Address: 172.22.75.10/16

4. courselab-mysql

-

EndpointID:

37fc4c9c5a74abee74183cb601c09902a8784b25af75c999dc38aeb4012daa94

- MacAddress: 02:42:ac:16:4b:08

- IPv4Address: 172.22.75.8/16

5. courselab-frontend

-

EndpointID:

cea3f21995dab408f4949aeac2e306482220c6d5ad6553f693c92d523d0ce9e1

- MacAddress: 02:42:ac:16:4b:04

- IPv4Address: 172.22.75.4/16

6. courselab-api

-

EndpointID:

f8f827f1a598bd4bcee8d0edd2560bd9bb92ee946ee5bb28742c82e5d9f123f5

- MacAddress: 02:42:ac:16:4b:03

- IPv4Address: 172.22.75.3/16

Опції та Лейбли

- com.docker.compose.network: courselab

- com.docker.compose.project: diploma-social-photo-service

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				41
Змн.	Арк.	№ докум.	Підпис	Дата		

- com.docker.compose.version: 2.20.2

Перейменування

Мережа "Docker universityMuseum" є прикладом локальної мережі, створеної за допомогою платформи Docker. Вона має назву "universityMuseum" та використовує драйвер "bridge", не підтримує IPv6. Ця мережа призначена для забезпечення комунікації між контейнерами у межах одного хоста.

При перезапуску проєкту створюється нова мережа з іншими ідентифікаторами. Це означає, що кожен раз при перезапуску проєкту створюється нова ізольована мережа, яка забезпечує безпечну та незалежну взаємодію між контейнерами.

У мережі "Docker universityMuseum" використовується система управління IP-адресами (IPAM) зі стандартним драйвером та параметрами конфігурації, такими як підсітка та шлюз. Це дозволяє ефективно керувати IP-адресами контейнерів у межах мережі та забезпечує їхню взаємодію з іншими мережами або зовнішніми ресурсами через маршрутизацію.

2.2.4 Проєктування бази даних

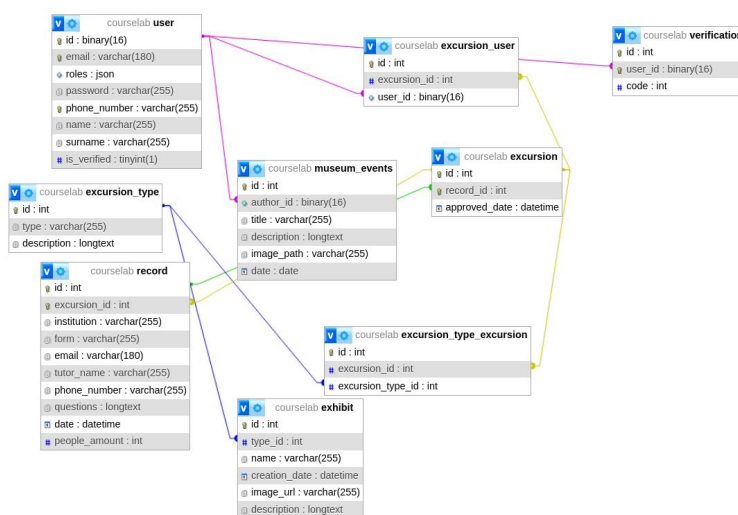


Рис. 2.2. – Структура бази даних.

База даних (рис. 2.2) розроблена для керування туристичною та культурною діяльністю музею. Вона включає сутності для опису різних типів екскурсій, виставок у музеях, подій, користувачів та їх дій. Така структура дозволяє

		Шмирина Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				42
Змн.	Арк.	№ докум.	Підпис	Дата		

відслідковувати участь користувачів у подіях, оцінювати популярність екскурсій та виставок, а також забезпечує безпеку даних та можливість підтвердження дій.

За допомогою взаємозв'язків між сутностями, такими як типи екскурсій та конкретні екскурсії, або користувачі та їх участь у подіях, база даних створює ефективну систему управління та аналізу даних. Це допомагає удосконалити сервіси для користувачів та забезпечити надійність інформації, що зберігається в системі.

В даному проєкті використовується MySQL Server, як основний кошик зберігання даних додатку. MySQL Server - це популярна система управління базами даних (СУБД), яка забезпечує зберігання та обробку структурованої інформації. Робота MySQL Server базується на клієнт-серверній моделі, де клієнтські додатки взаємодіють з сервером баз даних через мережу.

Під час роботи з MySQL Server, клієнтські додатки надсилають запити на обробку даних через мову запитів SQL (Structured Query Language). Сервер MySQL виконує ці запити, здійснюючи різні операції з даними, такі як вибірка, вставка, оновлення та видалення, залежно від вимог додатків.

Основні функції MySQL Server включають зберігання та управління базами даних, забезпечення безпеки даних через механізми автентифікації та авторизації, підтримку транзакцій для забезпечення консистентності даних та оптимізацію запитів для підвищення продуктивності обробки даних.

MySQL Server також підтримує різні типи зберігання даних, реплікацію для створення дубльованих копій баз даних та інші функціональні можливості, що роблять його однією з найпопулярніших СУБД для веб-додатків, бізнес-застосунків та інших сценаріїв роботи з даними.

Сутності (таблиці):

Excursion

Сутність Excursion відображає екскурсії в системі. Вона має відносини з іншими сутностями для організації та зберігання пов'язаної інформації.

Ідентифікатор (ID): Це поле відповідає за унікальний ідентифікатор кожної конкретної екскурсії.

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		43

Record: Це поле вказує на зв'язок з сутністю Record, яка, ймовірно, містить інформацію про участь користувача в екскурсії або інших записах.

Approved Date: Це поле містить дату затвердження екскурсії. Excursion

Type Excursions: Це колекція, яка вказує на зв'язок з сутністю

ExcursionTypeExcursion. Ймовірно, ця колекція містить інформацію про типи екскурсій, які доступні для конкретної екскурсії.

Excursion Users: Це колекція, яка вказує на зв'язок з сутністю ExcursionUser. Вона містить інформацію про користувачів, які приймають участь в цій екскурсії.

Крім того, в сутності **Excursion** визначені методи для додавання, видалення та отримання даних про типи екскурсій та користувачів, які беруть участь в екскурсіях. Такий підхід дозволяє ефективно управляти та модифікувати інформацію про екскурсії у системі.

User

Сутність **User** відображає користувачів системи і включає в себе різноманітні аспекти їхнього профілю та функціональності.

1. Ідентифікатор (ID): Це поле є унікальним ідентифікатором кожного користувача в системі. Використовується Uuid для забезпечення унікальності.

2. Email та Phone Number: Поля для зберігання електронної пошти та номеру телефону користувача. Мають валідаційні обмеження, такі як перевірка на унікальність та формат.

3. Roles: Поле, яке вказує на роль користувача в системі (наприклад, USER, ADMIN, GUEST).

4. Password: Захешований пароль користувача з відповідними валідаційними обмеженнями.

5. Name та Surname: Поля для зберігання імені та прізвища користувача.

6. Verification та Is Verified: Відображення інформації про підтвердження користувача (через сутність Verification) та прапорець, що вказує на підтвердження акаунта.

		Шмирина Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		44

7. Excursion Users та Museum Events: Відносини з іншими сутностями, які вказують на участь користувача у екскурсіях (ExcursionUsers) та створення подій у музеях (MuseumEvents).

8. Методи: Сутність містить методи для отримання, встановлення та модифікації даних користувача, такі як електронна пошта, пароль, ім'я, прізвище тощо, а також для управління ролями та підтвердженням акаунта.

В цілому, ця сутність добре структурована та має валідаційні обмеження для забезпечення правильності та безпеки даних користувачів у системі.

Exhibit

Сутність Exhibit відображає виставки, які можуть бути представлені в системі. Вона має різноманітні атрибути та відносини для забезпечення правильного функціонування системи.

1. Ідентифікатор (ID): Це поле є унікальним ідентифікатором кожної конкретної виставки в системі.

2. Назва (Name): Поле для зберігання назви виставки. Його можна використовувати для ідентифікації та відображення вмісту користувачам.

3. Дата створення (Creation Date): Це поле містить дату та час створення запису про виставку.

4. Тип (Type): Поле, яке вказує на тип виставки. Ймовірно, зв'язане з сутністю ExcursionType, яка містить інформацію про різні типи екскурсій.

5. Посилання на зображення (Image URL): Поле для зберігання посилання на зображення, що представляє виставку. Це може бути використано для візуального відображення вмісту користувачам.

6. Опис (Description): Текстове поле для зберігання опису виставки. Використовується для надання додаткової інформації про виставку.

Ця сутність також має конструктор, який ініціалізує деякі поля, методи для отримання та встановлення даних, що використовуються для доступу до атрибутів виставки, і відносини з іншими сутностями, такими як ExcursionType. Такий підхід дозволяє ефективно управляти та модифікувати інформацію про виставки в системі.

		Шмирина Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		45

Висновки з розділу 2

Ми дослідили ключові аспекти впровадження принципів кібербезпеки при проектуванні UX/UI дизайну веб-сайту музею науки і технологій «Політехнік». Основна увага була приділена трьом групам заходів безпеки: ідентифікації та аналізу загроз, принципам дизайну, орієнтованим на безпеку, та інтеграції з інструментами кібербезпеки.

Під час ідентифікації та аналізу загроз ми розглянули основні типи кібератак, оцінювали їх ймовірність і вплив, а також визначили цінні дані, які потребують захисту. Це дозволило сфокусувати увагу на ризиках і забезпечити захист особистої інформації користувачів.

Застосування принципів дизайну, орієнтованих на безпеку, дозволило створити кілька рівнів захисту, забезпечити шифрування даних та спростити користувачам розпізнавання потенційно небезпечних елементів. Це значно підвищило загальний рівень безпеки веб-сайту.

Інтеграція з інструментами кібербезпеки, такими як брандмауери, системи запобігання вторгненням, система керування доступом та двофакторна аутентифікація, забезпечила додатковий рівень захисту даних і системи в цілому.

Ми також дослідили методи UX/UI для проектування безпечного сайту, зокрема використання User Profile та User Flow. Це дозволило краще зрозуміти потреби різних категорій користувачів, що сприяло створенню логічної та зручної структури сайту, ефективної взаємодії та виявленню потенційних проблем на ранніх стадіях проектування.

На основі отриманих даних ми створили wireframes, які відображають структуру і компоненти кожної сторінки сайту, а також дизайн-макети, що враховують брендинг та загальний стиль музею. Це забезпечило високий рівень відповідності потребам користувачів та цілям організації, гарантуючи, що кінцевий дизайн буде естетично привабливим, функціональним і безпечним.

Проектування бекенду для сайту музею, включаючи використання сучасних інструментів, таких як Laravel, AWS S3 та Docker, дозволило створити комп'ютерну систему для ефективного управління експонатами, інформаційними

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		46

ресурсами та взаємодії з відвідувачами. Використання мікросервісної архітектури та забезпечення високої продуктивності, надійності та безпеки системи стало основою для успішного впровадження та експлуатації веб-сайту.

Таким чином, комплексний підхід до проєктування UX/UI дизайну з врахуванням кібербезпеки, детальне опрацювання потреб користувачів та використання сучасних технологій для розробки бекенду забезпечило створення безпечного, зручного та ефективного веб-сайту для музею науки і технологій «Політехнік».

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		47

РОЗДІЛ 3 РОЗРОБЛЕННЯ, НАЛАШТУВАННЯ ТА ТЕСТУВАННЯ САЙТУ МУЗЕЮ НАУКИ І ТЕХНОЛОГІЙ «ПОЛІТЕХНІК»

3.1 Заходи безпеки на рівні додатку

В основному тексті розділу 3 увагу зосереджено переважно на безпекових заходах, втілених на різних рівнях, передусім - на рівнях, де відбувається взаємодія з користувачів. Решту розгортань та налаштувань описано у додатках. Процедuru розгортання веб-сервера розглянуто у додатку В, налаштування AWS S3 продемонстровано у додатку Г.

На рівні серверної частини було використано JSON web token. За допомогою цієї технології в додатку є можливість надавати користувачеві права на доступ до певних ендпоінтів. Ендпоінти забезпечують взаємодію між клієнтом та сервером через визначені URL-адреси та дозволяють здійснювати специфічні операції над ресурсами на сервері, наприклад, створення нового запису (POST /records) та видалення існуючого запису (DELETE /records/{id}).

JSON Web Token (JWT) є стандартом для передачі інформації між сервісами у форматі JSON.

Ключові компоненти JWT є наступними.

Header: Містить інформацію про тип токена і алгоритм підпису (наприклад, HMAC, SHA256).

Payload: Містить твердження (claims), які представляють закодовану інформацію про користувача та інші метадані.

Signature: Підписується з використанням секретного ключа або пари ключів RSA/ECDSA, забезпечуючи цілісність і автентичність токена.

```
lexik_jwt_authentication:  
  secret_key: '%env(resolve:JWT_SECRET_KEY) %'  
  public_key: '%env(resolve:JWT_PUBLIC_KEY) %'  
  pass_phrase: '%env(JWT_PASSPHRASE) %'  
  token_ttl: 10000
```

Послідовність роботи JWT

- 1) **Аутентифікація користувача:** Користувач відправляє свої облікові дані на ендпоінт /api/login.

		Шмирина Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				48
Змн.	Арк.	№ докум.	Підпис	Дата		

- Цей процес забезпечує безпеку передачі даних і доступу до ресурсів, гарантує, що тільки авторизовані користувачі можуть отримати доступ до захищених ендпоінтів.



```
import { getExpirationDate, isExpired } from "../checkExpiredToken";
import eventBus from "../eventBus";
import { storageGetItem, TOKEN } from "../storage/storage";

const userAuthenticationConfig = (jsonld = true, multipart = false)
=> {

  let type = jsonld ? "application/json+ld" : "application/json";
  let contentType = multipart ? "multipart/form-data" : type;

  if (localStorage.getItem("token") !== null) {
    if (isExpired(getExpirationDate(localStorage.getItem("token"))))
    {
      eventBus.dispatch("logout", { expired: true });
      return;
    }
  }
}
```

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				49
Змн.	Арк.	№ докум.	Підпис	Дата		

```

return {
  headers: {
    "Content-Type": contentType,
    "Authorization": "Bearer " + localStorage.getItem("token"),
    "Accept": type
  }
};

return {
  headers: {
    "Content-Type": contentType,
    "Accept": type
  }
};
};

export default userAuthenticationConfig;

```

Цей метод повертає Headers запиту на сервер, де і передається токен. Серверна частина читає його, за допомогою security bundle перевіряє запит, після чого на сервері обробляється інформація, і вже залежно від логіки програми віддається Response.

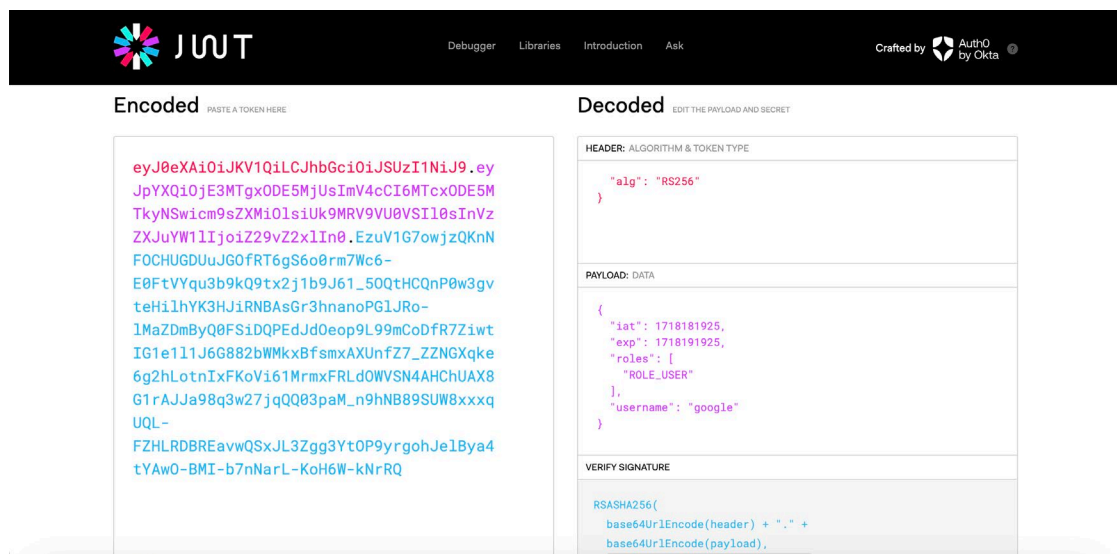


Рис. 3.2 – JWT decode на сайті jwt.io.

Файл Security.yaml виглядає наступним чином:

```

security:
  enable_authenticator_manager: true
  # https://symfony.com/doc/current/security.html#registering-the-user-hashing-passwords
  password_hashers:
    Symfony\Component\Security\Core\User\PasswordAuthenticatedUserInterface: 'auto'
  # https://symfony.com/doc/current/security.html#loading-the-

```

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				50
Змн.	Арк.	№ докум.	Підпис	Дата		

```

user-the-user-provider
  providers:
    # used to reload user from session & other features (e.g.
    switch_user)
    app_user_provider:
      entity:
        class: App\Entity\User
        property: username
  firewalls:
    login:
      pattern: ^/api/login
      stateless: true
      json_login:
        check_path: /api/login-check
        success_handler: lexik_jwt_authentication.handler.authentication_success
        failure_handler: lexik_jwt_authentication.handler.authentication_failure

    api:
      pattern: ^/api
      stateless: true
      jwt: ~

  access_control:
    - { path: ^/api/registration, roles: PUBLIC_ACCESS }
    - { path: ^/api/login, roles: PUBLIC_ACCESS }
    - { path: ^/api, roles: PUBLIC_ACCESS }

```

В цьому файлі описується робота security bundle – це пакет, який допомагає налаштовувати Firewalls, доступ до певних ендпоінтів, та конфігурує тип автентифікації.

```

firewalls:
  login:
    pattern: ^/api/login

```

Цей маршрут **/login**, визначений у класі **LoginController**, забезпечує базову функціональність для перевірки автентифікації користувача в системі. Використовуючи Symfony Security Component, він дозволяє нам визначити, чи є поточний користувач автентифікованим, і відповідно реагувати на цей запит.

Під час виклику маршруту /login, метод index спочатку намагається отримати поточного автентифікованого користувача за допомогою методу **getUser**. Якщо

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				51
Змн.	Арк.	№ докум.	Підпис	Дата		

користувач не аутентифікований, метод повертає **JSON**-відповідь з повідомленням **missing credentials** і статусом **401 Unauthorized**. Цей підхід гарантує, що неавторизовані користувачі не отримають доступ до захищених ресурсів і будуть проінформовані про необхідність надання облікових даних.

У разі успішної аутентифікації, метод повертає JSON-відповідь, яка містить ідентифікатор користувача, отриманий через **getUserIdentifier()**. Цей ідентифікатор може бути використаний для подальшої персоналізації взаємодії з користувачем, надання йому доступу до специфічних даних або функцій системи. Загалом, цей маршрут забезпечує важливу частину функціональності аутентифікації у додатку, покращуючи безпеку та зручність користування.

3.2 Валідація та доступ до endpoints

За допомогою Symfony анотацій дуже легко налаштовувати валідацію, та права доступу до контролерів.

Endpoint в REST API – це конкретна URL-адреса, за якою клієнт може отримати доступ до ресурсу або виконати певну дію на сервері. Кожен endpoint асоціюється з певним HTTP-методом (GET, POST, PUT, DELETE тощо) і визначає, яку операцію можна виконати над ресурсом. Наприклад, у REST API для управління користувачами endpoint <https://api.example.com/users> з методом GET може повернути список усіх користувачів, тоді як той самий endpoint з методом POST може використовуватися для створення нового користувача. Правильна організація та документування endpoint'ів є критично важливими для забезпечення ефективної взаємодії між клієнтом та сервером у рамках REST API.

Валідація та контроль доступу до точок входу на сервер (endpoints) є ключовими аспектами безпеки та надійності веб-додатків, розроблених на Symfony. Використовуючи анотації, Symfony надає потужний і зручний спосіб для налаштування цих аспектів. Валідація даних та контроль доступу до ресурсів не тільки підвищують рівень безпеки, але й забезпечують коректну роботу додатка, гарантуючи, що тільки авторизовані користувачі можуть виконувати певні дії або отримувати доступ до конкретних контролерів.

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		52

Валідація в Symfony реалізується через використання анотацій безпосередньо в класах моделей. За допомогою таких анотацій, як **@Assert**, можна легко визначити правила валідації для кожного атрибуту класу. Наприклад, для перевірки електронної пошти можна використовувати анотацію **@Assert\Email**, а для перевірки обов'язкових полів - **@Assert\NotBlank**. Це дозволяє розробникам швидко налаштовувати валідацію даних без необхідності писати додатковий код для кожної перевірки. Крім того, Symfony надає механізм кастомних валідаторів, які можна використовувати для реалізації специфічних правил валідації, що робить цей інструмент надзвичайно гнучким.

Для контролю доступу до endpoints у Symfony використовується анотація **@IsGranted**. Ця анотація дозволяє легко налаштувати права доступу до певних методів контролерів на основі ролей або прав користувачів. Наприклад, використання **@IsGranted("ROLE_ADMIN")** у методі контролера гарантує, що тільки користувачі з роллю адміністратора зможуть викликати цей метод. Це забезпечує високий рівень безпеки, оскільки дозволяє встановити доступ до критичних частин додатка лише для авторизованих користувачів з відповідними правами.

Разом з можливістю налаштування контролю доступу за допомогою анотацій, Symfony також підтримує складніші логіки авторизації. Використовуючи сервіс безпеки, як це було показано вище, можна налаштувати перевірки доступу на основі складних умов, таких як права доступу до конкретних ресурсів чи прив'язка до певних параметрів запиту. Таким чином, Symfony забезпечує не тільки прості та зрозумілі механізми для основних потреб валідації та авторизації, але й достатню гнучкість для вирішення більш складних завдань, пов'язаних з безпекою додатка.

3.3 Заходи безпеки на рівні Docker в конфігурації

У цій роботі розглядаються заходи безпеки, реалізовані на рівні Docker, які застосовуються для захисту контейнеризованих застосунків. Основну увагу приділено конфігураційному файлу `docker-compose.yml`, де описано налаштування

		Шмирина Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		53

для трьох основних сервісів: `api`, `frontend` та `nginx`. Кожен з цих сервісів ізольований один від одного та підключений до приватної мережі, що мінімізує ризики безпеки та забезпечує стабільність роботи системи.

Ізоляція контейнерів

Ізоляція контейнерів — це ключовий аспект сучасних контейнеризованих систем, який гарантує стабільність та безпеку роботи додатків. Перш за все, кожен контейнер має своє власне ізольоване середовище, що включає в себе файли, бібліотеки та залежності, не залежні від інших контейнерів. Це дозволяє уникнути конфліктів між версіями програм та забезпечує незалежність їх роботи.

Друга важлива перевага ізоляції полягає у тому, що кожен контейнер має свою власну мережу, що відділяє його від інших контейнерів і визначає доступність для зовнішніх систем. Це дозволяє контролювати мережевий трафік та забезпечує безпеку від несанкціонованого доступу.

Крім того, ізольованість контейнерів забезпечує простоту та надійність розгортання додатків. Кожен контейнер можна розглядати як окремий віртуальний сервер, що спрощує процес управління та розширення інфраструктури.

Мережеві налаштування

У даній конфігурації використовується спеціально налаштована мережа **courselab** з визначеною підмережею (172.22.75.0/16). Це забезпечує підключення всіх контейнерів до приватної мережі, що значно зменшує ризик зовнішніх атак. Приватні мережі Docker дозволяють ізолювати трафік між контейнерами, роблячи його недоступним зовні.

У сучасних додатках, що базуються на контейнеризації, мережеві налаштування відіграють важливу роль у забезпеченні стабільності, безпеки та ефективності роботи програмного забезпечення. Правильно сконфігуровані мережеві з'єднання дозволяють забезпечити безперебійну роботу додатків, зменшуючи ризики зовнішніх атак та оптимізуючи ресурсне споживання.

Правильно налаштовані мережеві з'єднання сприяють ефективному використанню ресурсів. Контейнери, підключені до відповідних мереж, можуть спільно використовувати ресурси, такі як CPU, пам'ять та мережевий пропуск,

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		54

відповідно до заданих обмежень. Це дозволяє оптимізувати роботу системи та забезпечити стабільну продуктивність додатків навіть при великому обсязі роботи.

Крім того, мережеві налаштування сприяють легкому масштабуванню та розширенню додатків. Завдяки правильно сконфігурованим мережам і залежностям, нові компоненти можуть бути додані до системи без перебоїв у роботі і безперервної доступності сервісів. Це робить мережеві налаштування не лише інструментом безпеки та ефективності, але й зручним засобом для розвитку та удосконалення програмних забезпечень у масштабі. Таким чином, мережеві налаштування сприяють підвищенню безпеки даних та зменшенню ймовірності несанкціонованого доступу.

Контрольовані зв'язки між контейнерами

Контрольовані зв'язки між контейнерами є важливою частиною конфігурації Docker, що дозволяє забезпечити коректну послідовність запуску та взаємодії між різними сервісами. У цьому контексті ключову роль відіграє директива `depends_on`, яка використовується для визначення залежностей між контейнерами. Наприклад, у нашій конфігурації контейнер `nginx` має залежність від контейнера `api`, що гарантує запуск `api` контейнера перед запуском `nginx`. Це забезпечує необхідну послідовність завантаження сервісів, знижуючи ймовірність помилок, пов'язаних із тим, що один з контейнерів ще не запущений.

Це налаштування особливо важливе для забезпечення стабільної роботи системи. Уявімо, що контейнер `nginx`, який відповідає за обробку запитів клієнтів, намагається підключитися до `api`, який ще не запущений. У такому випадку, клієнти отримають помилку, і система не зможе виконувати свої функції належним чином. Використовуючи `depends_on`, ми можемо гарантувати, що спочатку буде запущений контейнер `api`, який обробляє логіку додатку та відповідає на API-запити, і тільки потім буде запущений контейнер `nginx`, який обробляє зовнішні HTTP-запити.

Крім того, контрольовані зв'язки між контейнерами сприяють спрощенню процесу відладки та технічного обслуговування системи. У разі виникнення проблеми легко визначити, чи були всі залежні сервіси запущені в правильному

		Шмирина Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		55

порядку. Це зменшує час, необхідний для пошуку та виправлення помилок, а також підвищує загальну надійність та доступність системи. Такі налаштування допомагають створювати масштабовані та стійкі до помилок додатки, що є важливим фактором для забезпечення високого рівня сервісу та задоволення потреб користувачів.

Мапінг volumes

У конфігурації використовується мапінг volumes, який дозволяє зберігати дані контейнерів на хості. Це забезпечує збереження даних у разі перезапуску контейнерів. Крім того, доступ до volumes може бути налаштований на рівні файлової системи хоста для обмеження прав доступу. Таким чином, volumes мапінг підвищує надійність зберігання даних та захищає їх від несанкціонованого доступу.

Змінні середовища

Використання змінних середовища (.env) дозволяє налаштовувати контейнери без необхідності жорстко кодувати конфіденційну інформацію в конфігураційних файлах. Це дозволяє легко змінювати налаштування безпеки, не змінюючи самих файлів конфігурації. Змінні середовища надають гнучкість та зручність в управлінні конфігурацією системи.

Використання SSL-сертифікатів

SSL-сертифікати є важливим компонентом безпеки в конфігурації Docker. Вони забезпечують шифрування даних, що передаються між клієнтом і сервером, захищаючи їх від перехоплення злоумисниками. У цій конфігурації, SSL-сертифікати використані разом з контейнером nginx для забезпечення захищеного доступу до сервісів. Контейнер nginx налаштований на використання SSL-сертифікатів, які зберігаються у безпечних томах Docker. Це забезпечує захищену передачу даних та автентифікацію сервера, що значно підвищує загальний рівень безпеки системи.

nginx:
build:
dockerfile: Dockerfile.local
restart: unless-stopped
volumes:

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		56

```

- ./docker/certs/local:/etc/nginx/certs
ports:
- 80:80
- 443:443
- 8443:8443
environment:
NODE_ENV: dev
NGINX_HOST: ${NGINX_HOST_LOCAL}
NGINX_PORT: ${NGINX_PORT_LOCAL}
NGINX_CERT_CRT: ${NGINX_CERT_CRT_LOCAL}
NGINX_CERT_KEY: ${NGINX_CERT_KEY_LOCAL}

```

В налаштуваннях контейнеру nginx (веб-сервер), окремим volume підключаються ssl сертифікати.

Volume в Docker – це спеціальний механізм для зберігання даних, який використовується контейнерами Docker. Він дозволяє зберігати дані поза межами життєвого циклу контейнера, забезпечуючи збереження та доступність даних навіть після видалення або перезапуску контейнера. Volume створюється та керується самим Docker'ом, і може бути змонтований до одного або кількох контейнерів одночасно.

volumes:
- ./docker/certs/local:/etc/nginx/certs
SSL сертифікати зберігаються в директорії docker/certs/local, а після запуску контейнеру веб-серверу вони потрапляють у директорію /etc/nginx/certs.

```

/ # cd etc/nginx/certs/
/etc/nginx/certs # ls -la
total 16
drwxr-xr-x  4 root  root    128 Jan 31 12:51 .
drwxr-xr-x  1 root  root   4096 May 21 20:45 ..
-rwxr-xr-x  1 root  root   1935 Jan 31 12:51 cert.crt
-rwxr-xr-x  1 root  root   3272 Jan 31 12:51 cert.key
/etc/nginx/certs #

```

Рис. 3.3 – Збережені ssl сертифікати на веб-сервері.

```
/etc/nginx/certs # cat cert.crt
-----BEGIN CERTIFICATE-----
MIIFZzCCA0+gAwIBAgIUWBw0DABWtET/xAedmCGLj85GviUwDQYJKoZIhvcNAQEL
BQAwQzELMAkGA1UEBhMCVUExETAPBgNVBAgMCFpoeXRvbXlyMREwDwYDVQQHDAha
aHl0b215cjE0MAwGA1UECgwFTG9jYWwwHhcNMjEwMTQxMzI1WhcNMzEwMTMx
MTQxMzI1WjBDMQswCQYDVQQGEwJVQTERMA8GA1UECAwIWmh5dG9teXlxEtAPBgNV
BAcMCFpoeXRvbXlyMQ4wDAYDVQQKDAVMb2NhbDCCAiIwDQYJKoZIhvcNAQEBBQAD
```

Рис. 3.4 – Частина ssl сертифікату

У моєму випадку, веб-сервер має два файли: **cert.crt** і **cert.key**. Файл **cert.crt** (Certificate) містить сам SSL сертифікат, який включає публічний ключ і дані про організацію, для якої виданий сертифікат, а також інформацію про сертифікаційний центр (Certificate Authority, CA), який підписав цей сертифікат. Цей сертифікат підтверджує автентичність сервера для користувачів і забезпечує шифрування даних, що передаються. Коли користувач підключається до сайту, браузер перевіряє цей сертифікат, щоб переконатися, що він дійсний і виданий довіреним сертифікаційним центром.

Файл **cert.key** (Private Key) містить приватний ключ, який використовується для дешифрування інформації, зашифрованої публічним ключем з сертифікату **cert.crt**. Цей ключ ніколи не повинен передаватися третім особам. Під час SSL/TLS рукописання сервер використовує цей приватний ключ для створення безпечного з'єднання з клієнтом. Коли клієнт надсилає зашифровані дані на сервер, сервер використовує приватний ключ для дешифрування цих даних. Співпраця сертифікату і приватного ключа забезпечує безпечну передачу даних, захищаючи їх від перехоплення і модифікації.

Таким чином, ці два файли працюють разом, щоб забезпечити надійне шифрування і автентифікацію вашого веб-сайту. Сертифікат (**cert.crt**) використовується для підтвердження автентичності вашого сервера і шифрування даних, тоді як приватний ключ (**cert.key**) використовується для розшифрування цих даних і забезпечення безпечного зв'язку.

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		58

3.4 Адміністрування бази даних

В проєкті використовується MySQL для зберігання і управління інформацією. Для адміністрування бази даних з допомогою інтерфейсу користувача, використовується інструмент phpMyAdmin, який доступний через веб-браузер за адресою <http://localhost:8443>.

Після запуску проєкту та всіх необхідних контейнерів (включаючи MySQL та phpMyAdmin), можна увійти в адміністративний інтерфейс phpMyAdmin для управління базою даних. Там буде доступний весь функціонал phpMyAdmin, включаючи створення, видалення та редагування таблиць, виконання SQL-запитів, імпорт та експорт даних, налаштування прав доступу користувачів і багато іншого.

При вході до phpMyAdmin необхідно буде вказати наступні дані для підключення до вашої MySQL бази даних:

- Хост (PMA_HOST)
- Порт ('PMA_PORT'): 3306 (за умовчанням для MySQL)
- Ім'я користувача (MYSQL_USER)
- Пароль (MYSQL_PASSWORD)
- Ім'я бази даних (MYSQL_DATABASE)

Всі ці змінні зберігаються в env-файлі.

Після введення цих даних можна увійти до phpMyAdmin та почати адміністрування бази даних, здійснюючи різноманітні операції для управління та обробки даних у проєкті.

3.5 Рекомендації для підвищення рівня захисту веб-сайту

У результаті аналізу теоретичних аспектів підвищення захисту веб-сайтів та набуття досвіду впровадження обраних заходів на практиці було сформульовано наступні рекомендації щодо підвищення рівня захисту веб-сайту. Частина цих рекомендацій тісно пов'язані з UX/UI, частина також залучає конфігурацію бекенду сайту.

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		59

1. **Захист від крос-сайтового скриптування (XSS).** Використовувати відповідні методи очищення даних, щоб уникнути впровадження шкідливого коду через текстові поля.
2. **Захист від SQL-ін'єкцій.** Забезпечити бази даних від несанкціонованого доступу шляхом валідації та параметризації запитів.
3. **Коректне відображення елементів інтерфейсу.** Забезпечити правильне відображення елементів інтерфейсу на різних пристроях та розмірах екранів.
4. **Використання захищених заголовків фреймів (X-Frame-Options).** Встановити правильні заголовки фреймів для захисту від clickjacking.
5. **Валідація введених даних.** Перевіряти дані користувачів перед збереженням або обробкою для запобігання можливих атак.
6. **Механізми автентифікації та авторизації.** Забезпечити доступ тільки авторизованим користувачам через надійні механізми автентифікації та авторизації.
7. **Захист від фішингу.** Впровадити механізми перевірки достовірності посилань та електронних листів та використовувати двофакторну аутентифікацію для додаткового захисту облікових записів користувачів.
8. **Резервне копіювання та відновлення даних.** Визначити та реалізувати стратегії резервного копіювання та відновлення даних для запобігання втрат важливої інформації.
9. **Аудит безпеки.** Проводити регулярні аудити безпеки та підтримувати актуальність системи безпеки для виявлення та усунення потенційних вразливостей.
10. **Обмеження прав доступу для різних рівнів користувачів.** Забезпечити, щоб адміністратори, модератори та звичайні користувачі мали доступ лише до необхідних функцій.
11. **Вимога сильних паролів та двофакторна аутентифікація.** Зобов'язати користувачів створювати складні паролі. Впровадити двофакторну аутентифікацію для підвищення рівня безпеки.

		Шмирина Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				60
Змн.	Арк.	№ докум.	Підпис	Дата		

12. **Обмеження кількості спроб входу.** Встановити обмеження на кількість спроб введення паролів для уникнення атак перебору.
13. **Параметризовані запити до бази даних та перевірка введених даних.** Використовувати параметризовані запити для запобігання SQL-ін'єкціям. Валідувати введені користувачами дані перед їх збереженням чи обробкою.
14. **Захист з'єднання з використанням протоколу HTTPS.** Встановити HTTPS для забезпечення захищеного з'єднання між користувачами та сервером.
15. **Обмеження типів файлів, що можуть бути завантажені на сервер.** Заборонити завантаження потенційно небезпечних файлів. Використовувати атрибут ассерт у тегу `<input type="file">` для обмеження типів файлів, які може вибрати користувач. Перевіряти типи файлів після завантаження за допомогою бібліотек або фреймворків. Обмежити розмір файлу, який може бути завантажений, щоб уникнути перевантаження сервера. Перевіряти імена файлів на наявність недопустимих символів або шаблонів. Заборонити завантаженим файлам виконувати будь-які дії на сервері без необхідної перевірки та санітизації.
16. **Проведення регулярних аудитів безпеки.** Проводити регулярний аудит безпеки веб-додатку для виявлення потенційних вразливостей.
17. **Управління типами дозволених файлів.** Встановити чіткі правила щодо типів дозволених файлів та використовувати відповідні механізми перевірки та валідації.
18. **Людиноцентрований дизайн.** Залучати користувачів до всіх етапів дизайну та розробки. Враховувати потреби користувачів щодо приватності та забезпечувати їм можливість контролю над своїми персональними даними. Впроваджувати налаштування конфіденційності та використання шифрування для захисту інформації.
19. **Управління доступом до різних частин веб-сайту.** Використовувати системи аутентифікації з двофакторним підтвердженням для забезпечення

		Шмирина Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				61
Змн.	Арк.	№ докум.	Підпис	Дата		

додаткового рівня захисту. Оновлювати системи безпеки та надавати користувачам інформацію про потенційні загрози та правильні методи захисту їхніх даних.

- 20. Ідентифікація та аналіз загроз.** Вивчати поширені типи кібератак, такі як фішинг, DDoS-атаки, ін'єкції SQL, XSS-атаки. Оцінювати ймовірність та вплив кожної загрози, щоб сфокусувати зусилля на найсуттєвіших ризиках. Визначити цінні дані, які потребують захисту (особиста інформація користувачів, інформація про продукти чи послуги, фінансові дані, інтелектуальна власність).
- 21. Принципи дизайну, орієнтовані на безпеку.** Надавати користувачам лише ту інформацію, яка їм дійсно потрібна для виконання завдання. Розпізнавання потенційно небезпечних елементів, таких як посилання або кнопки, що можуть призвести до небажаних дій. Створювати кілька рівнів захисту, щоб навіть у разі успішної атаки на один рівень інші залишалися неушкодженими. Шифрувати дані, що передаються та зберігаються, для захисту від несанкціонованого доступу.
- 22. Інтеграція з інструментами кібербезпеки.** Використовувати брандмауери та системи запобігання вторгненням (IDS/IPS) для блокування несанкціонованого доступу та шкідливого трафіку. Використовувати IAM для контролю, хто має доступ до веб-сайту та які дії вони можуть виконувати. Застосовувати HTTPS та SSL/TLS для захисту даних, що передаються між користувачами та веб-сайтом. Впроваджувати 2FA, яка вимагає від користувачів введення не лише пароля, а й іншого фактору аутентифікації (наприклад, коду, що надсилається на телефон).
- 23. Інші ключові заходи.** Використовувати атрибут HTML асерт у тегу `<input type="file">` для обмеження типів файлів, які може вибрати користувач при завантаженні. Перевіряти типи файлів на сервері після завантаження за допомогою бібліотек або фреймворків для забезпечення перевірки MIME-типу файлу. Встановлювати допустимий розмір файлу

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				62
Змн.	Арк.	№ докум.	Підпис	Дата		

для завантаження, щоб запобігти завантаженню великих файлів, що можуть спричинити перевантаження сервера. Перевіряти імена файлів на наявність недопустимих символів або шаблонів, які можуть використовуватися для атак. Забороняти виконання дій завантаженими файлами без необхідної перевірки та санітизації, включаючи виконання JavaScript-коду з HTML або JavaScript файлів. Проводити регулярні аудити безпеки веб-додатку для виявлення потенційних вразливостей.

Ці заходи допоможуть забезпечити захист веб-сайту від різноманітних загроз та атак, а також гарантують надійність, конфіденційність та цілісність даних, які обробляються системою.

Висновки з розділу 3

Було докладно розглянуто заходи безпеки, впроваджені на різних рівнях системи, особливо на рівнях взаємодії з користувачем. Описано використання JSON Web Token (JWT) для автентифікації та авторизації користувачів, що забезпечує безпечну передачу даних та доступ до ресурсів лише для авторизованих користувачів. Процес генерації та перевірки токенів, а також їх зберігання в localStorage детально пояснено, що підкреслює важливість захисту даних на стороні клієнта та сервера.

Також використано Docker для ізоляції контейнерів, налаштування мережі та контрольовані зв'язки між контейнерами. Ці заходи забезпечують стабільність, безпеку роботи системи та ефективне управління ресурсами. Ізоляція контейнерів дозволяє уникнути конфліктів між версіями програмного забезпечення та забезпечує незалежність їх роботи, а правильно налаштовані мережеві з'єднання зменшують ризики зовнішніх атак.

Додатково було впроваджено адміністрування бази даних за допомогою phpMyAdmin, що надає зручний інтерфейс для управління даними, а також різні рекомендації для підвищення рівня захисту веб-сайту. Ці рекомендації включають захист від XSS та SQL-ін'єкцій, використання SSL-сертифікатів, налаштування

		Шмирина Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				63
Змн.	Арк.	№ докум.	Підпис	Дата		

доступу до ресурсів через анотації Symfony, валідацію даних та інші важливі аспекти, що сприяють загальному підвищенню безпеки веб-додатку.

Отже, реалізовані заходи безпеки значно підвищують захищеність системи від різноманітних атак та забезпечують стабільну роботу додатка. Комплексний підхід до безпеки, що включає автентифікацію, ізоляцію контейнерів, адміністрування баз даних та різноманітні механізми захисту, є ключовим фактором для створення надійного та безпечного веб-додатка.

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		64

ВИСНОВКИ

Розвиток сучасних технологій вимагає постійного оновлення заходів безпеки, оскільки нові можливості часто вносять нові виклики. У цьому контексті, вивчення та впровадження інноваційних методів UX/UI стає ключовим фактором адаптації веб-сайтів до сучасних стандартів безпеки.

Розробка безпечних та зручних рішень є важливою для уникнення ризиків витоку конфіденційної інформації та шкоди репутації організацій. Законодавчий тиск і вимоги до захисту даних роблять необхідним розробку та впровадження ефективних заходів безпеки, що враховують інтереси користувачів і відповідають вимогам закону. Мета цієї роботи полягає у дослідженні впливу дизайну UX/UI на безпеку веб-сайтів та розробка рекомендацій щодо підвищення безпеки веб-сайту з метою запобігання можливих кіберзагроз.

У процесі дослідження було вирішено наступні завдання:

- Проаналізовано наявні методи UX/UI, які впливають на безпеку веб-сайтів.
- Розроблено інтерфейсні обмеження для користувачів з метою уникнення потенційних кіберзагроз.
- Спроектовано та розроблено веб-сайт, інтерфейс якого відповідає безпековим обмеженням.
- Розробка рекомендацій для підвищення безпеки веб-сайту.

Об'єктом дослідження є процес розробки захищених інтерфейсів веб-сайтів, а предметом - методи та інструменти UX/UI, зокрема обмеження на дані, типи файлів, інші параметри взаємодії, які впливають на безпеку веб-сайтів.

У першому розділі роботи було підкреслено важливість забезпечення балансу між зручністю використання веб-сайту та його безпекою. Було розглянуто вплив UX/UI на безпеку, рівні, на яких можна впроваджувати заходи безпеки, функціональні обмеження інтерфейсу, управління типами дозволених файлів та захист даних користувачів. Встановлено, що безпека веб-сайту - це постійний процес, який потребує оновлення та вдосконалення.

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				65
Змн.	Арк.	№ докум.	Підпис	Дата		

Другий розділ присвячений розробці веб-сайту для науково-технічного музею, де було враховано необхідність поєднання принципів безпеки та зручності користування. Було розглянуто типи кібератак, оцінено їх ймовірність та вплив, розроблено інтуїтивно зрозумілий дизайн інтерфейсу, використано інструменти для захисту сайту від атак та впроваджено методи дослідження потреб користувачів для створення безпечного та зручного дизайну.

Третій розділ детально описує впроваджені заходи безпеки на різних рівнях системи, включаючи використання JSON Web Token (JWT) для автентифікації та авторизації користувачів, Docker для ізоляції контейнерів та адміністрування бази даних за допомогою phpMyAdmin. Було також розроблено рекомендації для захисту від XSS та SQL-ін'єкцій, використання SSL-сертифікатів, налаштування доступу до ресурсів через анотації Symfony, валідацію даних та інші заходи для підвищення рівня захисту веб-сайту.

Загалом, кваліфікаційна робота демонструє комплексний підхід до забезпечення безпеки веб-сайтів, поєднуючи інноваційні методи UX/UI з ефективними заходами кібербезпеки. Реалізовані заходи значно підвищують захищеність системи від різноманітних атак та забезпечують стабільну роботу додатка. Таким чином, робота робить вагомий внесок у розвиток інтернет-простору, забезпечуючи ефективну захищеність без втрати користувацького комфорту.

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		66

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. While W. P. UI/UX ДИЗАЙН: ПРИНЦИПИ ТА МЕТОДИ СТВОРЕННЯ ЗРУЧНОГО ІНТЕРФЕЙСУ КОРИСТУВАЧА [Електронний ресурс] / Web Production While. – 2023. – Режим доступу до ресурсу: <https://whileweb.com/uk/blog/uiux-dizajn-principi-ta-metodi-stvorennya-zruchnogo-interfejsu-koristuvacha/>
2. Jesse J. G. The Elements of User Experience User-Centered Design for the Web [Електронний ресурс] / James Garrett Jesse // American Institute of Graphic Arts. – 2002. – Режим доступу до ресурсу: <https://bootcamp.uxdesign.cc/the-elements-of-user-experience-bb98f4e385e3>
3. Ergonomics of human-system interaction — Part 210: Human-centred design for interactive systems [Електронний ресурс] // INTERNATIONAL STANDARD ISO 9241-210. – 2019. – Режим доступу до ресурсу: <https://cdn.standards.iteh.ai/samples/77520/8cac787a9e1549e1a7ffa0171dfa33e0/ISO-9241-210-2019.pdf>.
4. Як відрізнити потенційно небезпечні файли [Електронний ресурс] // Держспецзв’язок. – 2022. – Режим доступу до ресурсу: <https://www.village.com.ua/village/knowledge/edu-news/331709-yak-vidrizniti-potentsiyno-nebezpechni-fayli-derzhspetszv-yazku>.
5. File Upload Cheat Sheet [Електронний ресурс] // OWASP Cheat Sheets Series Team. – 2024. – Режим доступу до ресурсу: https://cheatsheetseries.owasp.org/cheatsheets/File_Upload_Cheat_Sheet.html.
6. The NIST Cybersecurity Framework (CSF) 2.0 [Електронний ресурс] // National Institute of Standards and Technology. – 2024. – Режим доступу до ресурсу: <https://nvlpubs.nist.gov/nistpubs/CSWP/NIST.CSWP.29.pdf>
7. Zhmykhov Y. Користувацькі персони [Електронний ресурс] / Yaroslav Zhmykhov // UX Pub. – 2022. – Режим доступу до ресурсу: <https://ux.pub/zhmikhov/koristuvatski-piersoni-533p>.
8. Muzyka B. User Flows. Як ця техніка допомагає в роботі над проєктами [Електронний ресурс] / Bohdana Muzyka // DOU.UA Спільнота

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				67
Змн.	Арк.	№ докум.	Підпис	Дата		

програмістів. – 2020. – Режим доступу до ресурсу: <https://dou.ua/lenta/columns/user-flows/>.

9. Guilizzoni P. What are wireframes and why are they used? [Електронний ресурс] / Peldi Guilizzoni // Balsamiq Wireframing Academy – Режим доступу до ресурсу: <https://balsamiq.com/learn/articles/what-are-wireframes/>.
10. Бренд-система Державного університету «Житомирська політехніка» [Електронний ресурс] – Режим доступу до ресурсу: <https://media.ztu.edu.ua/>.

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		68

ДОДАТКИ

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				69
Змн.	Арк.	№ докум.	Підпис	Дата		

Вигляд сторінок веб-сайту

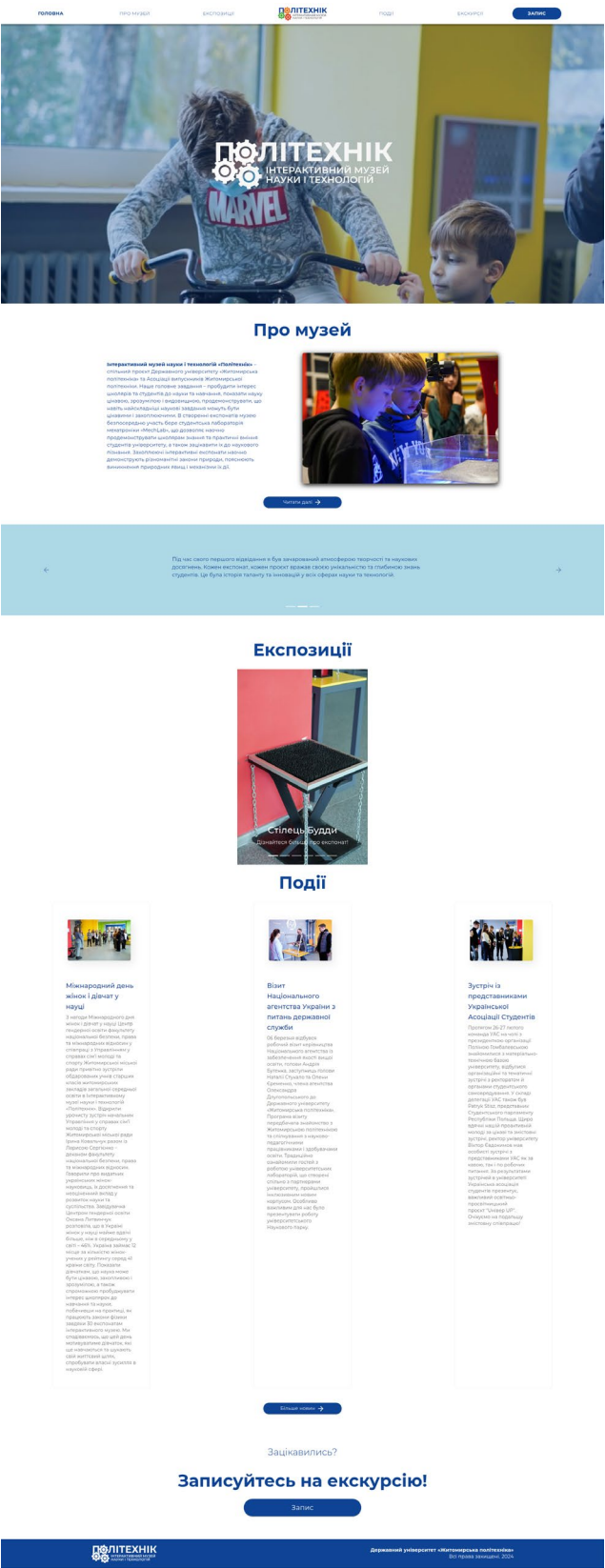


Рис. А.1 – Головна сторінка

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				70
Змн.	Арк.	№ докум.	Підпис	Дата		

Про музей

Інтерактивний музей науки і технологій «Політехнік» – спільний проєкт Державного університету «Житомирська політехніка» та Асоціації випускників Житомирської політехніки. Наше головне завдання – пробудити інтерес школярів та студентів до науки та навчання, показати науку цікавою, зрозумілою і видовищною, продемонструвати, що навіть найскладніші наукові завдання можуть бути цікавими і захоплюючими. В створенні експонатів музею безпосередню участь бере студентська лабораторія мехатроніки «MechLab», що дозволяє наочно продемонструвати школярам знання та практичні вміння студентів університету, а також зацікавити їх до наукового пізнання. Захоплюючі інтерактивні експонати наочно демонструють різноманітні закони природи, пояснюють виникнення природних явищ і механізми їх дії.



Місія музею – популяризація науки та техніки, стимулювання інтересу до наукових досліджень та виховання нового покоління інженерів та винахідників.

Цінності музею:

- **Доступність:** Ми прагнемо зробити науку доступною для людей будь-якого віку та рівня освіти.
- **Інтерактивність:** Наші експонати не просто розповідають про науку, вони дозволяють відвідувачам досліджувати та експериментувати.
- **Захопливість:** Ми прагнемо зробити вивчення науки цікавим та захоплюючим.
- **Інноваційність:** Ми постійно оновлюємо нашу експозицію, щоб вона відповідала найсучаснішим науковим досягненням.



Команда музею складається з ентузіастів науки та освіти, які прагнуть поділитися своїми знаннями та любов'ю до науки з іншими. До складу входять — екскурсоводи, фотографи, SMM-спеціалісти, навіть дизайнер та працівники лабораторії мехатроніки та робототехніки, аби всі експонати зберігалися в цілісності та збереженості. Важливо розуміти, що **кожен із експонатів унікальний, адже створювався безпосередньо руками студентів та випускників нашого університету.** Жоден із них не був куплений чи зроблений на замовлення поза стінами навчального закладу. До всього, що ви бачите можна доторкнутися і повзаємодіяти власноруч, проте обережно, щоб не нашкодити собі і не зламати прилади.



Музей співпрацює з партнерами:

- Асоціація випускників Житомирської політехніки.
- Лабораторія мехатроніки та робототехніки MechLab.
- Відділ маркетингу та зв'язів з громадськістю.
- Науковий парк.
- Інноваційний хаб.

Це дозволяє нам пропонувати нашим відвідувачам цікаві заходи, підтримувати активність в соціальних мережах, а також постійно оновлювати наші експозиції.



Асоціація випускників
Державного університету
«Житомирська політехніка»



Інноваційний хаб Житомирської
політехніки



Державний університет «Житомирська політехніка»
Всі права захищені. 2024

Рис. А.2 – Сторінка Про Музей

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				71
Змн.	Арк.	№ докум.	Підпис	Дата		

Експозиції

Відкрийте для себе дивовижний світ науки

Музей науки пропонує захоплюючу подорож у світ фізики, хімії, механіки та інших наук.

Тут ви зможете:

- Почути своє ехо в ехофоні, дослідити різні частоти звуку за допомогою частотної трубки, і насолодитися звучанням органу з труб.
- Побачити, як гальмується алюмінієвий диск магнітом.
- Дослідити принцип роботи котушки Тесли
- Дізнатися, чому "вічний двигун" неможливий.
- Перетворити механічну енергію в електричну за допомогою динамо-машини
- Кинути виклик своєму логічному мисленню та просторовому уявленню, розв'язуючи головоломки.



100+

експурсій

50+

експонатів

2000+

відвідувачів



- Дослідити принцип дії магнітного поля, побачити, як магніти притягуються або відштовхуються, і дізнатися про полюси магніту.
- Спробувати сісти на **стілець Будди**, який має лише одну точку опори, дослідити поширення хвиль у механічних системах, і побачити, як кулі Ньютона стикаються одна з одною.
- Подивитися, як рухаються платформи без двигуна, дослідити принцип роботи маятника **Максвелла**, і дізнатися про **фігури Хладні**.
- Дослідити оптичні явища за допомогою **цифрового мікроскопа**, побачити магію дзеркал, і дізнатися про поляризацію світла.

І багато іншого!
Музей науки - це місце, де ви зможете не лише дізнатися багато нового про світ навколо вас, але й отримати незабутні враження.
Приходьте та досліджуйте чудеса науки!



Рис. А.3 – Сторінка Експозиції

		Шмирина Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		72

Події

Міжнародний день жінок і дівчат у науці

З нагоди Міжнародного дня жінок і дівчат у науці Центр гендерної освіти факультету національної безпеки, права та міжнародних відносин у співпраці з Управлінням у справах сім'ї молоді та спорту Житомирської міської ради привітно зустріли обдарованих учнів старших класів житомирських закладів загальної середньої освіти в Інтерактивному музеї науки і технологій «Політехнік». Відкрили урочисту зустріч начальник Управління у справах сім'ї молоді та спорту Житомирської міської ради Ірина Ковальчук р...

Читати далі...

23.06.2024 03:00

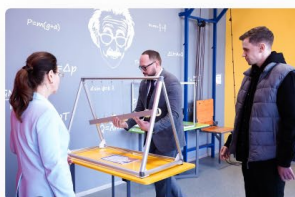


Візит Національного агентства України з питань державної служби

06 березня відбувся робочий візит керівництва Національного агентства із забезпечення якості вищої освіти, голови Андрія Бутенка, заступниць голови Наталії Стукало та Олени Єременко, члена агентства Олександра Длугопольського до Державного університету «Житомирська політехніка». Програма візиту передбачала знайомство з Житомирською політехнікою та спілкування з науково-педагогічними працівниками і здобувачами освіти. Традиційно ознайомили гостей з роботою університетських лабораторій, що створ...

Читати далі...

23.06.2024 03:00



Зустріч із представниками Української Асоціації Студентів

Протягом 26-27 лютого команда УАС на чолі з президенткою організації Поліною Гомбалецькою знайомилися з матеріально-технічною базою університету, відбулися організаційні та тематичні зустрічі з ректором й органами студентського самоврядування. У складі делегації УАС також був Patryk Stisz, представник Студентського парламенту Республіки Польща. Щиро вдячні нашій проактивній молоді за цікаві та змістовні зустрічі, ректор університету Віктор Євдокимов мав особисті зустрічі з представниками УА...

Читати далі...

23.06.2024 03:00



Рис. А.4 – Сторінка Події

		Шмиріна Т.С.		
		Головня О.С.		
Змн.	Арк.	№ докум.	Підпис	Дата



Вартість квитків

- Для класів або груп закладів освіти:
- З дитиною: 35 грн.
 - До 2-х супроводжуючих: безкоштовно

Контакти

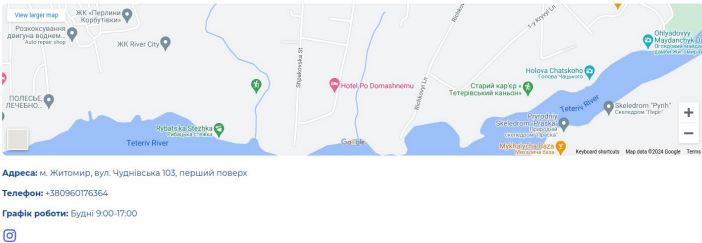


Рис. А.5 – Сторінка Екскурсії

Запис

Заклад освіти: Ліцей 28

Клас: 7-Б

Кількість людей: 55

Ім'я супроводжуючого: Галина Миколаївна

Номер телефону: +380 50 424 4478

Налишіть свої питання, якщо вони є: Яка вартість квитків для дорослих та дітей?

Email: lyceum28@ukr.net

Дата: 25.06.2023

Час: 04:00 PM

ВІДПРАВИТИ

Рис. А.6 – Сторінка Форми запису на екскурсію

		Шмирина Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				74
Змн.	Арк.	№ докум.	Підпис	Дата		

Список екскурсій

Список записів на екскурсію

Список експонатів

Список подій

Список типів

Список експонатів

Створити експонат



Стілець Будди

Більше 2-х тисяч цвяхів, розташовані один біля одного. Якщо доторкнутися пальцем, то боляче. Але якщо ми сядемо на цей стілець то ми нічого не відчуваємо. Це відбувається завдяки тому, що маса тіла розподіляється на всю площу поверхні. Приклад інженерного рішення, який дозволяє зменшити тиск на кожен окремий цвях

Редагувати

Видалити



Струми Фуко

Струми Фуко, які протікають у пластині, викликають власне магнітне поле, яке протистоїть зовнішньому магнітному полю магнітів у центрі. Ця взаємодія створює силу опору, яка сповільнює рух пластини, - чим менша площа заповненості поверхні, тим швидше буде їхати пластина.

Редагувати

Видалити

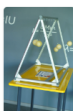


Маленький тесла

Інтерактивні схеми для пояснення принципів дії електричних законів. Інструкція до конструктора налічує 1000 схем, які дозволяють вивчати закони електрики в різноманітних поєднаннях елементів.

Редагувати

Видалити



Хвильовий маятник

Амплітуда коливань визначає максимальне відхилення кульки від її рівноважного положення. Чим більша амплітуда, тим далі кулька буде рухатися вгору і вниз під час коливань. Якщо всі кульки мають однакову амплітуду, то візерунки, створені ними під час коливань, будуть симетричними. Для того, щоб побачити візерунки, важливо підтримувати конструкцію, яка тримає кульки.

Редагувати

Видалити



Кулі ньютонa

У кулях Ньютона є декілька кульок з однаковою масою, які розміщені поруч одна з одною. Коли одна куля вдаряється в крайню кулю, яка знаходиться у спокої, енергія та імпульс передаються від однієї кулі до іншої. Закон збереження енергії стверджує, що сума енергій у системі залишається постійною, якщо на неї не діє зовнішня сила або не втрачається енергія через опір або інші фактори. У випадку куль Ньютона, коли одна куля вдаряється в крайню кулю, кінетична енергія переходить від першої кулі до останньої.

Редагувати

Видалити



Крісло Мюнхаузена

Дозволяє підняти самого себе. Ваша вага буде у 8 разів менша завдяки системі блоків, розміщеної позаду крісла.

Редагувати

Видалити

Рис. А.10 – Сторінка Експонатів

		Шмирина Т.С.		
		Головня О.С.		
Змн.	Арк.	№ докум.	Підпис	Дата

ККІКБ.КР.Б – 123 – 24 – ПЗ

Арк.

76

Список екскурсій

Список записів на екскурсію

Список експонатів

Список подій

Список типів

Список подій

Міжнародний день жінок і дівчат у науці



З нагоди Міжнародного дня жінок і дівчат у науці Центр гендерної освіти факультету національної безпеки, права та міжнародних відносин у співпраці з Управлінням у справах сім'ї молоді та спорту Житомирської міської ради

Розгорнути

Розгорнути

Відкрити

Візит Національного агентства України з питань державної служби



06 березня відбувся робочий візит керівництва Національного агентства із забезпечення якості вищої освіти, голови Андрія Бутенка, заступниці голови Наталії Стукало та Олени Єременко, члена агентства Олександра

Розгорнути

Розгорнути

Відкрити

Зустріч із представниками Української Асоціації Студентів



Протягом 26-27 лютого команда УАС на чолі з президенткою організації Поліною Гомбальською знайомилися з матеріально-технічною базою університету, відбулися організаційні та тематичні зустрічі з ректором й органами

Розгорнути

Розгорнути

Відкрити

Створити подію

Рис. А.11 – Сторінка створення Подій

Список екскурсій

Список записів на екскурсію

Список експонатів

Список подій

Список типів

Створення типу екскурсії

Назва*

Опис*

СТВОРИТИ ТИП ЕКСКУРСІЇ

Рис. А.12 – Сторінка Типів

		Шмиріна Т.С.		
		Головня О.С.		
Змн.	Арк.	№ докум.	Підпис	Дата

ККІКБ.КР.Б – 123 – 24 – ПЗ

Арк.

77

API PLATFORM

Default Project API1.0.1OAS3

API description

Servers

Authorize

ExcursionTypeExcursion

GET

/api/excursion-type-excursions

Retrieves the collection of ExcursionTypeExcursion resources.

POST

/api/excursion-type-excursions

Creates a ExcursionTypeExcursion resource.

GET

/api/excursion-type-excursions/{id}

Retrieves a ExcursionTypeExcursion resource.

PUT

/api/excursion-type-excursions/{id}

Replaces the ExcursionTypeExcursion resource.

ExcursionType

GET

/api/excursion-types

Retrieves the collection of ExcursionType resources.

POST

/api/excursion-types

Creates a ExcursionType resource.

GET

/api/excursion-types/{id}

Retrieves a ExcursionType resource.

PUT

/api/excursion-types/{id}

Replaces the ExcursionType resource.

DELETE

/api/excursion-types/{id}

Removes the ExcursionType resource.

ExcursionUser

GET

/api/excursion-users

Retrieves the collection of ExcursionUser resources.

POST

/api/excursion-users

Creates a ExcursionUser resource.

GET

/api/excursion-users/{id}

Retrieves a ExcursionUser resource.

PUT

/api/excursion-users/{id}

Replaces the ExcursionUser resource.

Excursion

GET

/api/excursions

Retrieves the collection of Excursion resources.

POST

/api/excursions

Creates a Excursion resource.

GET

/api/excursions/{id}

Retrieves a Excursion resource.

PUT

/api/excursions/{id}

Replaces the Excursion resource.

Exhibit

GET

/api/exhibits

Retrieves the collection of Exhibit resources.

POST

/api/exhibits

Creates a Exhibit resource.

GET

/api/exhibits/{id}

Retrieves a Exhibit resource.

PUT

/api/exhibits/{id}

Replaces the Exhibit resource.

DELETE

/api/exhibits/{id}

Removes the Exhibit resource.

MuseumEvents

GET

/api/museum-events

Retrieves the collection of MuseumEvents resources.

POST

/api/museum-events

Creates a MuseumEvents resource.

GET

/api/museum-events/{id}

Retrieves a MuseumEvents resource.

PUT

/api/museum-events/{id}

Replaces the MuseumEvents resource.

DELETE

/api/museum-events/{id}

Removes the MuseumEvents resource.

Record

GET

/api/records

Retrieves the collection of Record resources.

POST

/api/records

Creates a Record resource.

GET

/api/records/{id}

Retrieves a Record resource.

PUT

/api/records/{id}

Replaces the Record resource.

DELETE

/api/records/{id}

Removes the Record resource.

User

GET

/api/users

Retrieves the collection of User resources.

POST

/api/users

Creates a User resource.

GET

/api/users/{id}

Retrieves a User resource.

PUT

/api/users/{id}

Replaces the User resource.

Schemas

Рис. А.13.1 – Сторінка API

Schemas
Excursion-empty >
Excursion-get.collection.excursion >
Excursion-get.item.excursion >
Excursion-post.collection.excursion >
Excursion-put.item.excursion >
Excursion.jsonld-empty >
Excursion.jsonld-get.collection.excursion >
Excursion.jsonld-get.item.excursion >
Excursion.jsonld-post.collection.excursion >
Excursion.jsonld-put.item.excursion >
ExcursionType >
ExcursionType-empty >
ExcursionType-get.collection.excursion >
ExcursionType-get.collection.excursionType >
ExcursionType-get.collection.excursionTypeExcursion >
ExcursionType-get.collection.exhibit >
ExcursionType-get.item.excursionType >
ExcursionType-get.item.excursionTypeExcursion >
ExcursionType-get.item.exhibit >
ExcursionType-post.collection.excursionType >
ExcursionType-put.item.excursionType >
ExcursionType.jsonld >
ExcursionType.jsonld-empty >
ExcursionType.jsonld-get.collection.excursion >
ExcursionType.jsonld-get.collection.excursionType >
ExcursionType.jsonld-get.collection.excursionTypeExcursion >
ExcursionType.jsonld-get.collection.exhibit >
ExcursionType.jsonld-get.item.excursionType >
ExcursionType.jsonld-get.item.excursionTypeExcursion >
ExcursionType.jsonld-get.item.exhibit >
ExcursionType.jsonld-post.collection.excursionType >
ExcursionType.jsonld-put.item.excursionType >
ExcursionTypeExcursion-empty >
ExcursionTypeExcursion-get.collection.excursion >
ExcursionTypeExcursion-get.collection.excursionTypeExcursion >
ExcursionTypeExcursion-get.item.excursionTypeExcursion >
ExcursionTypeExcursion-post.collection.excursionTypeExcursion >
ExcursionTypeExcursion-put.item.excursionTypeExcursion >
ExcursionTypeExcursion.jsonld-empty >
ExcursionTypeExcursion.jsonld-get.collection.excursion >
ExcursionTypeExcursion.jsonld-get.collection.excursionTypeExcursion >
ExcursionTypeExcursion.jsonld-get.item.excursionTypeExcursion >
ExcursionTypeExcursion.jsonld-post.collection.excursionTypeExcursion >
ExcursionTypeExcursion.jsonld-put.item.excursionTypeExcursion >
ExcursionUser-empty >
ExcursionUser-get.collection.excursion >
ExcursionUser-get.collection.excursionUser >
ExcursionUser-get.item.excursionUser >
ExcursionUser-post.collection.excursionUser >
ExcursionUser-put.item.excursionUser >
ExcursionUser.jsonld-empty >
ExcursionUser.jsonld-get.collection.excursion >
ExcursionUser.jsonld-get.collection.excursionUser >
ExcursionUser.jsonld-get.item.excursionUser >
ExcursionUser.jsonld-post.collection.excursionUser >
ExcursionUser.jsonld-put.item.excursionUser >

Рис. А.13.2 – Продовження сторінки API

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				79
Змн.	Арк.	№ докум.	Підпис	Дата		

Exhibit >
Exhibit-empty >
Exhibit-get.collection.excursion >
Exhibit-get.collection.excursionType >
Exhibit-get.collection.exhibit >
Exhibit-get.item.exhibit >
Exhibit-post.collection.exhibit >
Exhibit-put.item.exhibit >
Exhibit.jsonld >
Exhibit.jsonld-empty >
Exhibit.jsonld-get.collection.excursion >
Exhibit.jsonld-get.collection.excursionType >
Exhibit.jsonld-get.collection.exhibit >
Exhibit.jsonld-get.item.exhibit >
Exhibit.jsonld-post.collection.exhibit >
Exhibit.jsonld-put.item.exhibit >
MuseumEvents >
MuseumEvents-empty >
MuseumEvents-get.collection.events >
MuseumEvents-get.item.events >
MuseumEvents-post.collection.events >
MuseumEvents-put.item.events >
MuseumEvents.jsonld >
MuseumEvents.jsonld-empty >
MuseumEvents.jsonld-get.collection.events >
MuseumEvents.jsonld-get.item.events >
MuseumEvents.jsonld-post.collection.events >
MuseumEvents.jsonld-put.item.events >
Record >
Record-empty >
Record-get.collection.record >
Record-get.item.record >
Record-post.collection.record >
Record-put.item.record >
Record.jsonld >
Record.jsonld-empty >
Record.jsonld-get.collection.record >
Record.jsonld-get.item.record >
Record.jsonld-post.collection.record >
Record.jsonld-put.item.record >
User-empty >
User-get.collection.excursion >
User-get.collection.excursionUser >
User-get.collection.user >
User-get.item.excursionUser >
User-get.item.user >
User-post.collection.user >
User-put.item.user >
User.jsonld-empty >
User.jsonld-get.collection.excursion >
User.jsonld-get.collection.excursionUser >
User.jsonld-get.collection.user >
User.jsonld-get.item.excursionUser >
User.jsonld-get.item.user >
User.jsonld-post.collection.user >
User.jsonld-put.item.user >

Available formats: [JSON](#), [JSON-LD](#), [XML](#)
Other API docs: [API Explorer](#), [API Reference](#)



Рис. А.13.3 – Продовження сторінки API

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				80
Змн.	Арк.	№ докум.	Підпис	Дата		

Лістинг HomePageGuest.jsx

```

import React, { useEffect, useState } from "react";
import { Carousel } from 'react-bootstrap';
import { Button, CardContent, CardMedia, Link, Typography } from "@mui/material";
import 'bootstrap/dist/css/bootstrap.min.css';
import mainImage from "../../assets/images/MainMavImage.png"
import secondMainImage from "../../assets/images/secondMainImage.png"
import mainLogo from "../../assets/icons/mainLogo.png"
import axios from "axios";
import PostModal from "../../components/elements/postModal/PostModal";
import Box from "@mui/material/Box";
import ArrowForwardIcon from '@mui/icons-material/ArrowForward';
import {useNavigate} from "react-router-dom";
import {ArrowBack, ArrowForward} from "@mui/icons-material";
import {Card} from "antd";
import "../../assets/css/main.css"

const HomePageGuest = () => {
  const [exposures, setExposures] = useState([]);
  const [showPopup, setShowPopup] = useState(false);
  const [selectedExposure, setSelectedExposure] = useState(null);
  const navigate = useNavigate();
  useEffect(() => {
    axios.get("https://courselab.com/api/get-exhibits")
      .then(response => {
        setExposures(response.data);
      })
      .catch(error => {
        console.error("Error fetching exposures:", error);
      });
  }, []);
  const [events, setEvents] = useState([]);
  useEffect(() => {
    axios.get('https://courselab.com/api/get-exhibits')
      .then(response => {
        const latestEvents = response.data.slice(0, 3);
        setEvents(latestEvents);
      })
      .catch(error => {
        console.error('Помилка при отриманні подій:', error);
      });
  }, []);
  const [openModal, setOpenModal] = useState(false);

```

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				81
Змн.	Арк.	№ докум.	Підпис	Дата		

```

const handleOpenModal = (exposure) => {
  setSelectedExposure(exposure);
  setOpenModal(true);
};

const handleCloseModal = () => {
  setSelectedExposure(null);
  setOpenModal(false);
};

const textArray = [
  "«Вперше увійшовши до приміщення, я був поринутий в атмосферу творчості та інновацій. Кожен експонат, кожен дисплей, кожен інтерактивний стенд свідчив про неймовірний рівень обдарованості та зусиль студентів. Вони продемонстрували свою експертизу у різних галузях науки та технологій через реалізацію проєктів з робототехніки, інженерії, комп'ютерних наук та інших сфер»",
  "Під час свого першого відвідання я був зачарований атмосферою творчості та наукових досягнень. Кожен експонат, кожен проєкт вражав своєю унікальністю та глибиною знань студентів. Це була історія таланту та інновацій у всіх сферах науки та технологій.",
  "Мій перший візит у цей приміщення залишив незабутнє враження. Кожен експонат, кожен дисплей, кожен проєкт переповідав історію про вражаючий рівень зусиль студентів та їхню пристрасть до знань. Це був справжній океан ідей та таланту, який вразив своєю розмаїтістю та глибиною."
];
return (
  <div style={{position: "relative", width: "100%", overflow: "hidden"}}>
    <div className="image-container" style={{position: "relative", display: "flex", justifyContent: "center", alignItems: "center"}}>
      <img
        src={mainImage}
        alt="Фото"
        style={{
          width: "100%",
          height: "auto",
          objectFit: "cover",
          maxHeight: "calc(100vh - 65px)",
          objectPosition: "center top"
        }}
      />

      <img src={mainLogo} alt="Логотип" style={{
        position: "absolute",
        maxWidth: "50%",
        width: "30%",
        height: "auto",
        zIndex: 1,
        top: "50%",
        left: "50%",
        transform: "translate(-50%, -50%)"
      }}/>
    </div>
  </div>
);

```

		Шмирина Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				82
Змн.	Арк.	№ докум.	Підпис	Дата		

```

</div>
<Box sx={{textAlign: "center", margin: "20px", marginTop: "50px"}}>
  <strong>
    <h1 style={{fontWeight: 700, fontSize:"52px",
color:"#0F4495"}}>Про музей</h1>
  </strong>
  <Box sx={{
    display: "flex",
    justifyContent: "space-between",
    alignItems: "center",
    mt: 3,
    mx: "auto",
    marginTop: "40px",
    marginBottom: "40px",
    maxWidth:"1200px"
  }}>
    <div style={{flexGrow: 1, mr: 3, fontSize:"15px" , maxWidth:
"40%", maxHeight: "400px", overflowY: "auto", textAlign: "left"}}>
      <p style={{fontWeight:"400"}}><strong>Інтерактивний музей
науки і технологій «Політехнік»</strong> – спільний проєкт
Державного університету «Житомирська політехніка» та
Асоціації випускників Житомирської
політехніки.
Наше головне завдання – пробудити інтерес школярів та
студентів до науки та навчання,
показати науку цікавою, зрозумілою і видовищною,
продемонструвати, що навіть найскладніші
наукові завдання можуть бути цікавими і захоплюючими. В
створенні експонатів музею
безпосередню участь бере студентська лабораторія
мехатроніки «MechLab», що дозволяє наочно
продемонструвати школярам знання та практичні вміння
студентів університету, а також
зацікавити їх до наукового пізнання.
Захоплюючі інтерактивні експонати наочно демонструють
різноманітні закони природи, пояснюють
виникнення природних явищ і механізми їх дії.</p>
    </div>
    <div style={{maxWidth: "50%"}}>
      <img src={secondMainImage} alt="Фото" style={{maxWidth:
"100%", height: "auto", borderRadius:"5px", boxShadow: "0px 4px 16px black"}}/>
    </div>
  </Box>
  <Box sx={{mt: 3, display: "flex", justifyContent: "center"}}>
    <Button
      variant="contained"
      style={{
        backgroundColor: "#0F4495",
        borderRadius: "20px",
        display: "flex",

```

		Шмирина Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		83

```

        alignItems: "center",
        justifyContent: "center",
        width: "240px",
        height: "35px",
        textTransform: "none"
      }}
      onClick={() => navigate('/about')}
    >
      <p style={{fontWeight: "400", paddingTop: "16px"}}>Читати
далі</p>
      <ArrowForwardIcon sx={{marginLeft: "5px"}}/>
    </Button>
  </Box>
</Box>
<div>
  <Carousel
    style={{backgroundColor: 'lightblue', padding: "40px", marginTop:
"50px", marginBottom: "50px"}}
    nextIcon={<strong><ArrowForward style={{color:
'#0F4495'}}/></strong>}
    prevIcon={<strong><ArrowBack style={{color:
'#0F4495'}}/></strong>}
  >
    {textArray.map((text, index) => (
      <Carousel.Item key={index} style={{color: "#0F4495"}}>
        <div
          style={{
            width: "100%",
            height: "200px",
            display: "flex",
            justifyContent: "center",
            alignItems: "center",
          }}
        >
          <div style={{display: "flex", justifyContent: "center",
alignItems: "center", textAlign: "left", maxWidth: 800, }}>
            <p style={{fontWeight: "400"}}>{text}</p>
          </div>
        </div>
      </Carousel.Item>
    ))}
  </Carousel>
</div>
<div className="container" style={{textAlign: "center"}}>
  <strong>
    <h1 style={{fontWeight: 700, fontSize: "52px",
paddingBottom: "20px", color: "#0F4495"}}>Експозиції</h1>
  </strong>
  <Carousel>
    {exposures.map((exposure) => (
      <Carousel.Item key={exposure.id}>

```

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		84

```

        <div onClick={() => handleOpenModal(exposure)}
style={{display:"flex", justifyContent:"center"}}>
        <img
            src={exposure.imageUrl}
            alt={exposure.name}
            style={{
                Width: "100%",
                height: "600px",
                objectFit: "cover"
            }}
        />
        <Carousel.Caption>
            <h3>{exposure.name}</h3>
            <p>Дізнайтеся більше про експонат!</p>
        </Carousel.Caption>
    </div>
</Carousel.Item>
    )}
</Carousel>
</div>
<div>
    <strong>
        <h1 align="center" style={{fontWeight: 700, fontSize:"52px",
color:"#0F4495"}}>Події</h1>
    </strong>
    <div style={{display: 'flex', justifyContent: 'space-around'}}>
        {events.map((event, index) => (
            <Card key={index} style={{maxWidth: 300, margin: '20px'}}>
                <CardMedia
                    component="img"
                    height="200"
                    image={event.imageUrl}
                    alt={`Event ${index + 1}`}
                    style={{objectFit:"contain"}}
                />
                <CardContent>
                    <Typography variant="h6" gutterBottom>
                        {event.title}
                    </Typography>
                    <Typography variant="body2" color="textSecondary">
                        {event.description}
                    </Typography>
                </CardContent>
            </Card>
        ))}
    </div>
    <Box sx={{mt: 3, display: "flex", justifyContent: "center"}}>
        <Button
            variant="contained"
            style={{
                backgroundColor: "#0F4495",

```

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				85
Змн.	Арк.	№ докум.	Підпис	Дата		

```

borderRadius: "20px",
display: "flex",
alignItems: "center",
justifyContent: "center",
width: "240px",
height: "35px",
textTransform: "none"
}}
onClick={() => navigate('/events')}
>
<p style={{fontWeight: "400", paddingTop: "15px"}}>Більше
новин</p>
<ArrowForwardIcon sx={{marginLeft: "5px"}}/>
</Button>
</Box>
</div>
<div style={{textAlign: 'center', marginTop: "100px"}}>
<h3 style={{marginBottom: "40px", fontWeight: "300",
color: "#0F4495"}}>Зацікавилися?</h3>
<strong style={{marginBottom: "40px"}}>
<h1 style={{fontWeight: "700", fontSize: "52px",
color: "#0F4495"}}>Записуйтеся на екскурсію!</h1>
</strong>
<Box sx={{mt: 3, display: "flex", justifyContent: "center"}}>
<Button
variant="contained"
style={{
backgroundColor: "#0F4495",
padding: "15px",
paddingLeft: "100px",
paddingRight: "100px",
borderRadius: "40px",
display: "flex",
alignItems: "center",
height: "54px",
width: "360px",
textTransform: "none"
}}
onClick={() => navigate('/record')}
>
<strong style={{fontWeight: "400",
fontSize: "21px"}}>Запис</strong>
</Button>
</Box>
</div>
{selectedExposure && (
<PostModal open={openModal} exposure={selectedExposure}
onClose={handleCloseModal}/>
)}
<Box
component="footer"

```

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		86

```

        sx={{
            py: 2,
            px: 3,
            backgroundColor: "#0F4495",
            color: "white",
            display: "flex",
            alignItems: "center",
            justifyContent: "space-between",
            paddingRight:"15%",
            paddingLeft:"15%",
            marginTop:"70px"
        }}
    >
    <Box sx={{ paddingRight: 2 }}>
        <img src={mainLogo} alt="Логотип" style={{ maxWidth: "200px" }} />
    </Box>
    <Box sx={{ textAlign: "right" }}>
        <h4 style={{ margin: 0, fontWeight:"700", fontSize:"15px" }}>Державний університет «Житомирська політехніка»</h4>
        <p style={{ margin: 0, fontSize:"15px" }}>Всі права захищені.
        2024</p>
    </Box>
</Box>
</div>
);
};

```

```
export default HomePageGuest;
```

RegistrationController.php

```

<?php

namespace App\Controller;

use ApiPlatform\Validator\ValidatorInterface;
use App\Entity/Card;
use App\Entity\User;
use App\Entity\Verification;
use Doctrine\ORM\EntityManagerInterface;
use Exception;
use Sensio\Bundle\FrameworkExtraBundle\Configuration\IsGranted;
use Sensio\Bundle\FrameworkExtraBundle\Configuration\Security;
use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;
use Symfony\Component\HttpFoundation\JsonResponse;
use Symfony\Component\HttpFoundation\Request;
use Symfony\Component\HttpFoundation\Response;
use Symfony\Component\HttpKernel\Exception\AccessDeniedHttpException;
use Symfony\Component\HttpKernel\Exception\UnprocessableEntityHttpException;
use Symfony\Component>PasswordHasher\Hasher\UserPasswordHasherInterface;
use Symfony\Component\Routing\Annotation\Route;

```

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				87
Змн.	Арк.	№ докум.	Підпис	Дата		

```

use Symfony\Component\Security\Http\Attribute\CurrentUser;
use Symfony\Component\Serializer\Exception\ExceptionInterface;
use Symfony\Component\Serializer\Normalizer\DenormalizerInterface;

class RegistrationController extends AbstractController
{
    /**
     * @param UserPasswordHasherInterface $passwordHasher
     * @param EntityManagerInterface $entityManager
     * @param ValidatorInterface $validator
     * @param DenormalizerInterface $denormalizer
     */
    public function __construct(
        private UserPasswordHasherInterface $passwordHasher,
        private EntityManagerInterface $entityManager,
        private ValidatorInterface $validator,
        private DenormalizerInterface $denormalizer
    ){}

    /**
     * @throws ExceptionInterface
     */
    #[Route('/register', name: 'app_register', methods: ['POST'])]
    public function registration(Request $request): JsonResponse
    {
        $user = $this->createUser($request);
        $this->entityManager->persist($user);
        $this->entityManager->flush();

        return new JsonResponse([], Response::HTTP_CREATED);
    }

    /**
     * @param Request $request
     * @return User
     * @throws ExceptionInterface
     * @throws Exception
     */
    private function createUser(Request $request): User
    {
        $content = $request->getContent();

        if (!$content) {
            throw new UnprocessableEntityHttpException("Data required");
        }

        $requestData = json_decode($request->getContent(), true);

        /** @var User $user */
        $user = $this->denormalizer->denormalize($requestData, User::class, "array");
    }
}

```

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		88

```

        $this->validator->validate($user);

        $hashedPassword = $this->passwordHasher->hashPassword($user, $user-
>getPassword());
        $user->setPassword($hashedPassword);

        $user->setRoles([User::ROLE_GUEST]);

        return $user;
    }
}

```

VerificationController.php

```

<?php

namespace App\Controller;

use ApiPlatform\Validator\ValidatorInterface;
use App\Entity\User;
use App\Entity\Verification;
use App\Services\MailerService;
use DateTime;
use Doctrine\ORM\EntityManagerInterface;
use Symfony\Bundle\FrameworkBundle\Controller\AbstractController;
use Symfony\Component\HttpFoundation\JsonResponse;
use Symfony\Component\HttpFoundation\Request;
use Symfony\Component\HttpFoundation\Response;
use Symfony\Component\HttpKernel\Exception\UnprocessableEntityHttpException;
use Symfony\Component\Routing\Annotation\Route;
use Symfony\Component\Security\Core\Security;
use Symfony\Component\Serializer\Normalizer\DenormalizerInterface;

class VerificationController extends AbstractController
{
    /**
     * @var EntityManagerInterface
     */
    private $entityManager;
    /**
     * @var ValidatorInterface
     */
    private ValidatorInterface $validator;
    /**
     * @var DenormalizerInterface
     */
    private DenormalizerInterface $denormalizer;
    /**
     * @var MailerService
     */
}

```

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				89
Змн.	Арк.	№ докум.	Підпис	Дата		

```

private MailerService $mailerService;

/**
 * @param EntityManagerInterface $entityManager
 * @param ValidatorInterface $validator
 * @param DenormalizerInterface $denormalizer
 * @param MailerService $mailerService
 */
public function __construct(EntityManagerInterface $entityManager,
ValidatorInterface $validator, DenormalizerInterface $denormalizer, MailerService
$mailerService)
{
    $this->entityManager = $entityManager;
    $this->validator = $validator;
    $this->denormalizer = $denormalizer;
    $this->mailerService=$mailerService;
}

/**
 * @param Request $request
 * @return JsonResponse
 */
#[Route('/verify-user', name: 'verify-user', methods: ['PUT'])]
public function verifyUser(Request $request): JsonResponse
{
    $user = $this->getUser();

    $currentUserRepository = $this->entityManager->getRepository(User::class);
    /** @var User|null $currentUser */
    $currentUser = $currentUserRepository->findOneBy(['email' => $user-
>getUserIdentifier()]);

    $content = json_decode($request->getContent(), true);
    $code = $content['code'];

    $currentVerifyRepository= $this->entityManager-
>getRepository(Verification::class);

    /** @var Verification|null $currentVerify */
    $currentVerify = $currentVerifyRepository->findOneBy(['user' =>
$currentUser->getId()]);

    if(!($currentVerify && $currentVerify->getCode()==$code)){
        return new JsonResponse("Невірний код!", Response::HTTP_BAD_REQUEST);
    }

    $currentUser->setRoles(["ROLE_USER"]);

    $this->entityManager->persist($currentUser);
    $this->entityManager->flush();
}

```

		Шмирина Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арх.
		Головня О.С.				
Змн.	Арх.	№ докум.	Підпис	Дата		90

```

        return new JsonResponse([], Response::HTTP_OK);
    }

    /**
     * @param Request $request
     * @return JsonResponse
     */
    #[Route('/generate-code', name: 'generate_code', methods: ['POST'])]
    public function generateVerificationCode(Request $request): JsonResponse
    {
        $user = $this->getUser();

        $currentUserRepository = $this->entityManager->getRepository(User::class);
        /** @var User|null $currentUser */
        $currentUser = $currentUserRepository->findOneBy(['email' => $user->getUserIdentifier()]);

        $currentVerificateRepository= $this->entityManager->getRepository(Verification::class);

        $newVerification = new Verification();
        $newVerification->setUser($currentUser);

        $randomCode = rand(1000,9999);
        $newVerification->setCode($randomCode);

        $userReceiver = [
            'userEmail' => $currentUser->getEmail(),
        ];

        $verificationInfo = [
            'code' => $randomCode,
        ];

        $this->mailerService->SendVerificationMail($userReceiver, $verificationInfo);

        $this->entityManager->persist($newVerification);
        $this->entityManager->flush();

        return new JsonResponse([], Response::HTTP_CREATED);
    }
}

```

		Шмирина Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				91
Змн.	Арк.	№ докум.	Підпис	Дата		

Розвертання веб-серверу

Nginx - це веб-сервер з відкритим вихідним кодом та серверне програмне забезпечення для проксі-сервера. Однією з його ключових особливостей є високий рівень продуктивності та масштабованості, що робить його популярним вибором для веб-додатків з великим обсягом трафіку.

Nginx відомий своєю здатністю ефективно обробляти багато одночасних з'єднань, використовуючи мінімальні ресурси сервера. Це досягається завдяки асинхронній та подієвій архітектурі, яка дозволяє обробляти тисячі з'єднань одночасно без перенавантаження сервера.

Окрім того, Nginx має вбудовані засоби безпеки, такі як обмеження доступу до ресурсів, фільтрація HTTP-запитів та захист від DDoS-атак. Також він часто використовується як засіб балансування навантаження, що дозволяє розподіляти трафік між різними серверами для покращення продуктивності та надійності веб-додатків.

Налаштування Nginx складається з трьох серверних блоків, які визначають різні веб-сервери та їх функціональність:

1. Перший серверний блок працює на порті 80 і відповідає за перенаправлення HTTP-трафіку на HTTPS. Використовується запис `return 301 https://${NGINX_HOST}`; для перенаправлення на HTTPS-версію сайту.

2. Другий серверний блок слухає порт `${NGINX_PORT}` для HTTPS-з'єднань і має налаштовані SSL-сертифікати для безпечного з'єднання. Він також використовує проксі-пас-записи для передачі трафіку на веб-сервер frontend та API серверу, використовуючи префікси `/` та `/api` відповідно.

3. Третій серверний блок слухає порт 8443 і використовується для проксі-переадресації трафіку на веб-інтерфейс phpMyAdmin. Він також має налаштовані SSL-сертифікати для безпечного з'єднання.

У цих блоках також присутні налаштування `access_log` та `error_log` для логування подій доступу та помилок. Кожен блок має свої налаштування, такі як

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		92

кореневі директорії, проксі-пас-записи, налаштування кешування та обробки PHP-файлів.

```
2024-06-05 22:13:42 172.22.0.1 - - [05/Jun/2024:19:13:42 +0000] "GET /api/newestuserPosts?offset=0&limit=10 HTTP/2.0" 200 15587 "https://courselab.com/" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/125.0.0.0 Safari/537.36" "-"
2024-06-05 23:01:32 172.22.0.1 - - [05/Jun/2024:20:01:32 +0000] "GET /ws HTTP/1.1" 101 7478 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/125.0.0.0 Safari/537.36" "-"
2024-06-06 11:23:31 172.22.0.1 - - [06/Jun/2024:08:23:31 +0000] "GET /ws HTTP/1.1" 101 14618 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/125.0.0.0 Safari/537.36" "-"
2024-06-06 11:46:46 172.22.0.1 - - [06/Jun/2024:08:46:46 +0000] "GET /ws HTTP/1.1" 101 3616 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/125.0.0.0 Safari/537.36" "-"
2024-06-06 12:52:27 172.22.0.1 - - [06/Jun/2024:09:52:27 +0000] "GET /ws HTTP/1.1" 101 4660 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/125.0.0.0 Safari/537.36" "-"
2024-06-06 16:42:58 172.22.0.1 - - [06/Jun/2024:13:42:57 +0000] "GET /ws HTTP/1.1" 101 27416 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/125.0.0.0 Safari/537.36" "-"
2024-06-06 16:43:03 172.22.0.1 - - [06/Jun/2024:13:43:02 +0000] "GET /ws HTTP/1.1" 101 1754 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/125.0.0.0 Safari/537.36" "-"
2024-06-07 10:41:40 172.22.0.1 - - [07/Jun/2024:07:41:40 +0000] "GET /ws HTTP/1.1" 101 17298 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/125.0.0.0 Safari/537.36" "-"
2024-06-07 12:18:00 172.22.0.1 - - [07/Jun/2024:09:18:00 +0000] "GET /ws HTTP/1.1" 101 8498 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/125.0.0.0 Safari/537.36" "-"
2024-06-07 13:50:40 172.22.0.1 - - [07/Jun/2024:10:50:40 +0000] "GET /ws HTTP/1.1" 101 11042 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/125.0.0.0 Safari/537.36" "-"
2024-06-09 23:07:32 2024/06/09 20:07:32 [notice] 1#1: using the "epoll" event method
2024-06-09 23:07:32 2024/06/09 20:07:32 [notice] 1#1: nginx/1.20.1
2024-06-09 23:07:32 2024/06/09 20:07:32 [notice] 1#1: built by gcc 10.2.1 20201203 (Alpine 10.2.1_pre1)
2024-06-09 23:07:32 2024/06/09 20:07:32 [notice] 1#1: OS: Linux 5.15.49-linuxkit-pr
2024-06-09 23:07:32 2024/06/09 20:07:32 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2024-06-09 23:07:32 2024/06/09 20:07:32 [notice] 1#1: start worker processes
2024-06-09 23:07:32 2024/06/09 20:07:32 [notice] 1#1: start worker process 11
2024-06-09 23:07:32 2024/06/09 20:07:32 [notice] 1#1: start worker process 12
2024-06-09 23:07:32 2024/06/09 20:07:32 [notice] 1#1: start worker process 13
2024-06-09 23:07:32 2024/06/09 20:07:32 [notice] 1#1: start worker process 14
```

Рис. 2.1. –Логування подій на NGINX контейнері.

Налаштування AWS S3

Моя робота з S3 уособлює надійність та ефективність управління файлами в хмарному сховищі. За допомогою AWS SDK та налаштованого середовища AWS можливо виконувати операції з файлами, такі як завантаження, оновлення та отримання. Кожен файл, що завантажується, проходить перевірку та зберігається у відповідному бакеті, гарантуючи його безпеку та доступність.

Сервіс розроблений для забезпечення зручного та безпечного використання S3. Він дозволяє користувачам завантажувати файли, включаючи аватари та пости, та отримувати доступ до них з будь-якого пристрою. Розробка сервісу з урахуванням принципів безпеки та швидкості дозволяє максимально використовувати можливості S3 для зберігання та обробки файлів.

Взаємодія з S3 включає в себе створення та керування теками та файлами у бакетах. Це дозволяє організувати зберігання та доступ до файлів у логічні групи, що полегшує навігацію та управління файловою структурою. Безперервна підтримка та моніторинг роботи сервісу забезпечують надійну та безперервну роботу з файлами.

Робота з AWS SDK реалізована з урахуванням потреб користувачів у швидкому та безпечному доступі до файлів. Всі завантажені файли проходять валідацію та зберігаються з дотриманням стандартів безпеки. Це дозволяє користувачам максимально ефективно використовувати сервіс для зберігання та обміну файлами.

Ключовим елементом в системі є зберігання та обробки файлів на базі Amazon S3. Це важливий компонент, що допомагає забезпечити високу доступність, швидкість та безпеку управління файлами у хмарному середовищі.

Підписні посилання, які використовуються для передачі файлів у системі - це механізм, який дозволяє створювати безпечні та тимчасові посилання на вміст або

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		94

ресурси, зазвичай в хмарних сховищах або веб-серверах. Ось як вони зазвичай працюють:

1. Створення посилання: Підписні посилання створюються на основі вмісту (файлу, сторінки, документа тощо) або ресурсу (URL, API endpoint). Це посилання може бути створене за допомогою спеціальних бібліотек чи сервісів, а також непрямо шляхом підписування частин URL або параметрів запиту.

2. Підписання посилання: Після створення посилання, воно підписується за допомогою алгоритму хешування або криптографічного підпису. Це додає до посилання спеціальний підпис або токен, який підтверджує його автентичність та непорушеність.

3. Надання доступу: Коли користувач переходить за підписним посиланням, запит містить підпис або токен, який перевіряється на сервері. Сервер перевіряє цей підпис, порівнюючи його з тим, який має він сам. Якщо підпис співпадає та токен ще не прострочений, сервер надає доступ до вмісту чи ресурсу.

4. Обмеження доступу: Підписні посилання часто мають обмеження в часі дії або кількості використань. Наприклад, посилання може бути дійсним лише протягом певного періоду, після чого воно автоматично стає недійсним. Також можна обмежити кількість відкриттів посилання або його використання на певні дії.

5. Автоматична збірка інформації: Крім надання доступу, система підписних посилань може збирати інформацію про те, як часто вони використовуються, хто за ними переходить, та інші аналітичні дані. Це може бути корисно для відстеження ефективності посилань та аналізу їхнього використання.

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				95
Змн.	Арк.	№ докум.	Підпис	Дата		

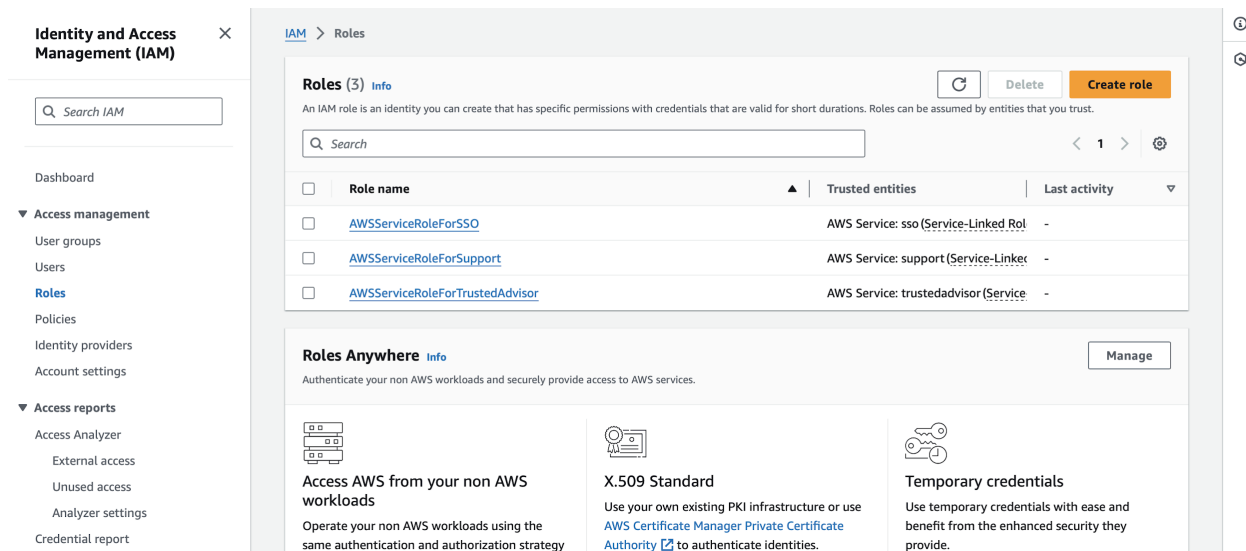


Рис. 2.3. – IAM користувачікі ролі.

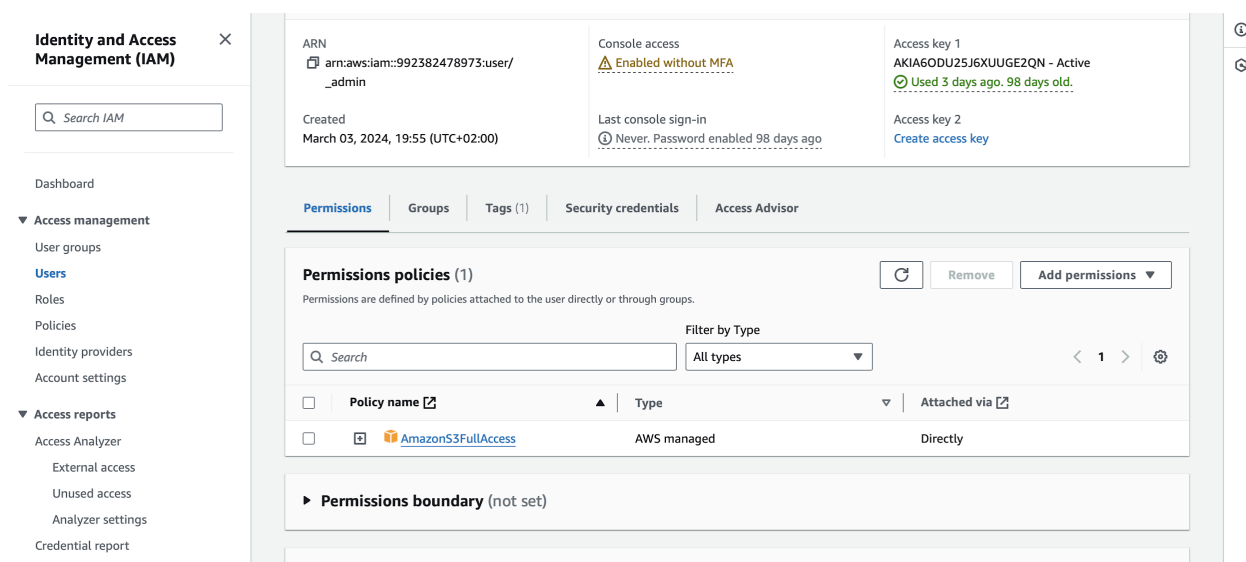


Рис. 2.3. – Інформація про IAM користувача.

IAM User (Identity and Access Management User) - це сутність в Amazon Web Services (AWS), яка представляє користувача системи і надає можливість керувати доступом до різних ресурсів та послуг AWS. Ось як працює IAM User і для чого його використовують:

1. Авторизація і аутентифікація: IAM User використовується для аутентифікації користувача в AWS. Кожен IAM User має унікальний ідентифікатор і пароль, які використовуються для входження в обліковий запис. Це забезпечує безпеку та визначає, хто має доступ до AWS-ресурсів.

2. Керування правами доступу: Однією з основних функцій IAM User є керування правами доступу. Адміністратор може надати або обмежити права

		Шмирина Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				96
Змн.	Арк.	№ докум.	Підпис	Дата		

користувачам на використання конкретних AWS-ресурсів, таких як S3, EC2, RDS та інші. Це дозволяє розподіляти доступ та забезпечувати принцип найменшого доступу.

3. Створення і управління інфраструктурою: IAM User дозволяє керувати різними аспектами інфраструктури AWS, такими як створення та управління інстансами EC2, базами даних, бакетами S3 та іншими сервісами. Це дає можливість налаштовувати інфраструктуру згідно з потребами проєкту або організації.

4. Моніторинг і аудит: Кожна дія, виконана під IAM User, реєструється в журналі AWS CloudTrail. Це дозволяє відстежувати та аналізувати дії користувачів для виявлення неправомірних або небажаних активностей. Такий моніторинг допомагає забезпечити безпеку та дотримання внутрішніх політик безпеки.

5. Розподіл завдань: IAM User дозволяє розподіляти завдання та керувати доступом до AWS-сервісів між членами команди або внутрішніми підрозділами організації. Це спрощує співпрацю та дозволяє кожному користувачеві працювати з необхідними ресурсами.

В цілому, IAM User є важливим елементом управління безпекою та доступом до AWS-сервісів. Використання IAM User допомагає забезпечити безпеку, гнучкість та ефективність управління обліковими записами та ресурсами AWS.

		Шмиріна Т.С.			ККІКБ.КР.Б – 123 – 24 – ПЗ	Арк.
		Головня О.С.				
Змн.	Арк.	№ докум.	Підпис	Дата		97