КВАЛІФІКАЦІЙНА РОБОТА

Група ІПЗс-20-1

Костенко М.І

2024

ЗВО УНІВЕРСИТЕТ КОРОЛЯ ДАНИЛА

Факультет суспільних та прикладних наук

Кафедра інформаційних технологій

на правах рукопису

Костенко Михайло Іванович

УДК 004.378

Розробка веб-платформи для анонімного спілкування студентів

Спеціальність 121 – «Інженерія програмного забезпечення»

Кваліфікаційна робота на здобуття кваліфікації бакалавра

Нормоконтроль

_____Стисло О.В.
(підпис, дата, розшифрування підпису)

Студент

_____Костенко М.І
(підпис, дата, розшифрування підпису)

Допускається до захисту

Завідувач кафедри

_____к.т.н., доц. Ващишак С.П.
(підпис, дата, розшифрування підпису)

Керівник роботи

асистент каф. ІТ

_____ Витвицький Р. І.
(підпис, дата, розшифрування підпису)

Івано-Франківськ – 2024

ЗВО УНІВЕРСИТЕТ КОРОЛЯ ДАНИЛА

Факультет суспільних та прикладних наук

Кафедра інформаційних технологій

Освітній ступінь: «бакалавр»

Спеціальність: 121 «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____

« ___ » _____ 2024 року

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Костенко Михайло Іванович

(прізвище, ім’я, по батькові)

1. Тема кваліфікаційної роботи

Розробка веб-платформи для анонімного спілкування студентів

керівник роботи:

Витвицький Роман Ігорович, асистент каф. ІТ

затверджена наказом вищого навчального закладу від «12» березня 2024 року № 19/1

2. Термін подання студентом роботи 05.06.2024

3. Вихідні дані роботи: Мова програмування JavaScript, HTML, CSS

4. Зміст кваліфікаційної роботи (перелік питань, які потрібно розробити)

1. Опис та аналіз існуючих аналогів

2. Проектування веб-палтформи

3. Розробка та дизайн веб-платформи для конфіденційного обміну інформацією серед студенті

5. Дата видачі завдання 14.03.2024

КОНСУЛЬТАНТИ РОЗДІЛІВ КВАЛІФІКАЦІЙНОЇ РОБОТИ

Розділ	Консультант (прізвище, ініціали та посада)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Термін виконання етапів роботи	Примітка
1.		12.01.2024	Виконано
2.		18.01.2024	Виконано
3.		15.02.2024	Виконано
4.		13.03.2024	Виконано
5.		19.04.2024	Виконано
6.		27.04.2024	Виконано

Студент _____ Костенко М.І.

(підпис) (прізвище та ініціали)

Керівник роботи _____ Витвицький Р.І.

(підпис) (прізвище та ініціали)

Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

Сторінка	Опис графічного матеріалу	Сторінка	Опис графічного матеріалу
	Процес авторизації		Асиметричне шифрування даних
	Google Drive сервіс		Симетричне шифрування
	Dropbox сервіс		Use Case Diagram діаграма
	OneDrive сервіс		Макет веб-платформи
	Dropbox сервіс		Макет веб-платформи
	OneDrive сервіс		Макет месенджеру
	Signal сервіс		Макет календарю
	Tresorit сервіс		Макет веб-платформи
	SpiderOak сервіс		Двофакторна автентифікація

АНОТАЦІЯ

У сучасному світі інформаційних технологій та цифрових комунікацій безпека та конфіденційність даних стають все більш важливими аспектами. Особливо це стосується освітнього середовища, де студенти часто обмінюються особистою та академічною інформацією, яка потребує захисту. Ця кваліфікаційна робота присвячена розробці веб-платформи для конфіденційного обміну інформацією серед студентів.

Основною метою роботи є створення безпечної та зручної веб-платформи, яка забезпечить студентам можливість обміну інформацією без ризику витоку або несанкціонованого доступу. У ході дослідження були розглянуті сучасні методи шифрування даних, механізми автентифікації та авторизації користувачів, а також технології, що забезпечують цілісність та конфіденційність переданої інформації.

Робота включає аналіз існуючих рішень та визначення вимог до системи, проектування архітектури веб-платформи з акцентом на безпеку та масштабованість. Розробка функціоналу для реєстрації та автентифікації користувачів із застосуванням двофакторної авторизації, впровадження шифрування даних на стороні клієнта та сервера, тестування та оцінка ефективності та безпеки розробленої платформи стали важливими етапами цього дослідження.

Результатом роботи є функціональна веб-платформа, яка забезпечує конфіденційний обмін повідомленнями та файлами серед студентів, надаючи при цьому високий рівень захисту даних. Платформа має потенціал для подальшого розширення та інтеграції з іншими освітніми сервісами.

КЛЮЧОВІ СЛОВА: ВЕБ-ПЛАТФОРМА, КОНФІДЕНЦІЙНІСТЬ ДАНИХ, БЕЗПЕКА ІНФОРМАЦІЇ, ШИФРУВАННЯ, АВТЕНТИФІКАЦІЯ, АВТОРИЗАЦІЯ, ОБМІН ІНФОРМАЦІЄЮ, ДВОФАКТОРНА АВТОРИЗАЦІЯ, ЗАХИСТ ДАНИХ.

SUMMARY

In today's world of information technology and digital communications, data security and confidentiality are becoming increasingly important aspects. This is especially true in the educational environment, where students frequently exchange personal and academic information that requires protection. This qualification work is dedicated to the development of a web platform for confidential information exchange among students.

The primary goal of this work is to create a secure and user-friendly web platform that provides students with the ability to exchange information without the risk of leakage or unauthorized access. The study examines modern methods of data encryption, user authentication and authorization mechanisms, as well as technologies that ensure the integrity and confidentiality of transmitted information.

The work includes the analysis of existing solutions and the determination of system requirements, the design of the web platform architecture with an emphasis on security and scalability. The development of functionality for user registration and authentication using two-factor authorization, the implementation of data encryption on both the client and server sides, testing, and the evaluation of the platform's effectiveness and security were key stages of this research.

The result of this work is a functional web platform that ensures confidential exchange of messages and files among students, providing a high level of data protection. The platform has the potential for further expansion and integration with other educational services.

KEYWORDS: WEB PLATFORM, DATA CONFIDENTIALITY, INFORMATION SECURITY, ENCRYPTION, AUTHENTICATION, AUTHORIZATION, INFORMATION EXCHANGE, TWO-FACTOR AUTHORIZATION, DATA PROTECTION.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

Html – мова гіпертекстової розмітки

UI – інтерфейс користувача

Css – каскадні таблиці стилів

Scss – препроцесор мови Css

Js – JavaScript

React – бібліотека для розробки користувацьких інтерфейсів

Node.js – середовище виконання JavaScript на сервері

SPA – односторінковий додаток (Single Page Application)

DOM – об'єктна модель документа (Document Object Model)

ВСТУП

Актуальність теми. У сучасних умовах стрімкого розвитку інформаційних технологій та цифрових комунікацій питання безпеки та конфіденційності даних стають надзвичайно актуальними. Зокрема, в освітньому середовищі студенти активно обмінюються особистою та академічною інформацією через різноманітні онлайн-платформи, що підвищує ризик несанкціонованого доступу до цієї інформації. Забезпечення безпеки даних та конфіденційності інформації є ключовим завданням для розробників програмного забезпечення, особливо у сфері освіти. Розробка веб-платформи, яка б гарантувала захист даних студентів та забезпечувала безпечний обмін інформацією, є важливим кроком до підвищення рівня безпеки в освітніх установах.

Мета і завдання дослідження. Метою даного дослідження є створення безпечної та зручної веб-платформи для конфіденційного обміну інформацією серед студентів. Для досягнення цієї мети необхідно вирішити декілька важливих завдань. По-перше, потрібно проаналізувати існуючі рішення та визначити вимоги до системи, що дозволить зрозуміти поточний стан справ і виявити можливі недоліки та переваги різних підходів. По-друге, необхідно розробити архітектуру веб-платформи з акцентом на безпеку та масштабованість, що забезпечить надійність та гнучкість системи. По-третє, слід створити функціонал для реєстрації та автентифікації користувачів із застосуванням двофакторної авторизації, що підвищить рівень захисту облікових записів користувачів. По-четверте, потрібно впровадити шифрування даних на стороні клієнта та сервера, щоб забезпечити конфіденційність інформації під час її передачі. Нарешті, необхідно провести тестування та оцінку ефективності та безпеки розробленої платформи, що дозволить перевірити її працездатність та надійність у реальних умовах.

Об’єкт дослідження. Процеси обміну інформацією серед студентів у цифровому середовищі.

Предмет дослідження. Методи та технології забезпечення безпеки та

конфіденційності даних під час обміну інформацією на веб-платформі.

Методи дослідження. Для досягнення поставлених завдань у даному дослідженні застосовуються різні методи. Аналіз та синтез використовуються для вивчення існуючих рішень та визначення вимог до системи. Ці методи дозволяють систематизувати інформацію про поточні технології та підходи, а також сформулювати чіткі вимоги до розроблюваної платформи. Проектування застосовується для створення архітектури та функціоналу веб-платформи, що включає розробку структурних та функціональних компонентів системи. Моделювання використовується для розробки бази даних та користувацького інтерфейсу, що дозволяє створити ефективну та зручну структуру даних і інтуїтивно зрозумілий інтерфейс для користувачів. Тестування необхідне для перевірки ефективності та безпеки платформи, що включає проведення різних видів тестів, таких як функціональне тестування, навантажувальне тестування та тестування безпеки. Емпіричний метод застосовується для оцінки користувацького досвіду та задоволеності від використання платформи, що дозволяє отримати зворотний зв'язок від користувачів та вдосконалити систему на основі їхніх відгуків.

Практичне значення одержаних результатів. Розроблена веб-платформа надає студентам безпечний засіб для обміну конфіденційною інформацією, що сприяє підвищенню рівня безпеки в освітньому середовищі. Платформа може бути інтегрована з іншими освітніми сервісами, що розширить її функціональні можливості та збільшить зручність користування. Запропоновані рішення можуть бути використані не тільки в освітніх установах, але й в інших сферах, де важлива конфіденційність інформації, що обмінюється. Практичне значення роботи полягає в підвищенні загального рівня захисту даних у цифрових комунікаціях, що є важливим аспектом сучасного інформаційного суспільства.

Структура. Розділи – 3. Обсяг основної частини – 51 сторінок. Список використаних джерел – 20.

РОЗДІЛ 1. ОПИС ТА АНАЛІЗ ІСНУЮЧИХ АНАЛОГІВ

1.1 Огляд сучасних технологій для захисту даних.

У сучасному світі захист даних є критично важливим аспектом будь-якої інформаційної системи, особливо у контексті веб-платформ, що використовуються для обміну конфіденційною інформацією. Сучасні технології захисту даних спрямовані на забезпечення цілісності, конфіденційності та доступності інформації. До основних технологій захисту даних належать шифрування, методи автентифікації та авторизації, а також механізми забезпечення цілісності даних.

Шифрування є одним із найефективніших способів захисту інформації. Існують різні типи шифрування, такі як симетричне та асиметричне шифрування. Симетричне шифрування використовує один ключ для шифрування та дешифрування даних. Найпопулярнішими алгоритмами симетричного шифрування є AES (Advanced Encryption Standard) та DES (Data Encryption Standard). Асиметричне шифрування, на відміну від симетричного, використовує два ключі: відкритий для шифрування та закритий для дешифрування. Найвідомішими алгоритмами асиметричного шифрування є RSA (Rivest–Shamir–Adleman) та ECC (Elliptic Curve Cryptography).

Методи автентифікації відіграють ключову роль у забезпеченні безпеки доступу до інформаційних систем. Автентифікація зазвичай базується на чотирьох основних принципах: щось, що користувач знає (пароль), щось, що користувач має (токен або смарт-карта), щось, чим користувач є (біометричні дані, такі як відбитки пальців або розпізнавання обличчя), та динамічні одноразові паролі (OTP). Двофакторна автентифікація (2FA) комбінує два з цих принципів, значно підвищуючи рівень безпеки (рис. 1.1).

Методи авторизації визначають рівень доступу користувачів до ресурсів системи після успішної автентифікації. Найпоширенішими методами авторизації є ACL (Access Control List), RBAC (Role-Based Access Control) та ABAC (Attribute-Based Access Control). ACL надає або забороняє доступ до ресурсів на

основі списків доступу, тоді як RBAC надає доступ на основі ролей користувачів. ABAC, у свою чергу, надає доступ на основі атрибутів користувачів, ресурсів та контексту доступу (рис. 1.1).

Рисунок 1.1 – Процес авторизації

Механізми забезпечення цілісності даних включають такі технології, як хешування та цифрові підписи. Хешування використовує алгоритми, такі як SHA-256 (Secure Hash Algorithm), для створення унікальних відбитків даних, що дозволяє перевірити їхню цілісність. Цифрові підписи забезпечують автентичність та цілісність даних шляхом використання криптографічних алгоритмів, які дозволяють підтвердити, що дані не були змінені після підписання.

Крім того, сучасні технології захисту даних включають засоби захисту від атак типу DDoS (Distributed Denial of Service), технології захисту від зловмисного програмного забезпечення, а також системи виявлення та запобігання вторгнень (IDS/IPS). Використання цих технологій у комплексі дозволяє створити надійні та безпечні веб-платформи для обміну конфіденційною інформацією.

1.2 Аналіз існуючих платформ для обміну інформацією

В сучасному освітньому середовищі обмін інформацією між студентами є невід'ємною складовою навчального процесу. Важливо забезпечити безпеку і

конфіденційність при обміні даними, особливо коли мова йде про академічні роботи, особисту інформацію та інші важливі документи. Для розробки веб-платформи, яка задовольнятиме ці вимоги, необхідно проаналізувати існуючі рішення, виявити їх сильні та слабкі сторони і визначити, які технології та методи можуть бути використані для створення нової платформи.

Популярні платформи для обміну інформацією:

1. Google Drive є одним з найпопулярніших інструментів для зберігання та обміну файлами. Він дозволяє користувачам створювати, зберігати та спільно використовувати документи, таблиці, презентації та інші файли. До переваг цієї платформи можна віднести простоту використання, інтеграцію з іншими сервісами Google, підтримку різних форматів файлів та безкоштовний обсяг зберігання до 15 ГБ. Проте, серед недоліків варто зазначити обмежену конфіденційність даних, оскільки Google може аналізувати вміст файлів для покращення своїх сервісів, а також залежність від підключення до інтернету (рис.2.2).

Рисунок 1.2 – Google Drive сервіс

2. Dropbox є ще одним популярним сервісом для зберігання та обміну файлами. Він пропонує зручний інтерфейс та підтримку різних типів файлів. Серед переваг Dropbox можна виділити легку інтеграцію з різними додатками та сервісами, надійну синхронізацію файлів між пристроями та можливість відновлення видалених файлів. Однак, обмежений безкоштовний обсяг зберігання (до 2 ГБ) та відсутність розширених функцій безпеки в базовій версії є недоліками цієї платформи (рис.2.3).

Рисунок 2.3 – Dropbox сервіс

3. OneDrive – це сервіс від Microsoft, який пропонує користувачам можливість зберігати та ділитися файлами. Він тісно інтегрований з продуктами Microsoft Office. Серед переваг OneDrive можна відзначити глибоку інтеграцію з Microsoft Office, можливість спільного редагування документів в реальному часі та значний обсяг безкоштовного зберігання до 5 ГБ. Проте, обмежена конфіденційність даних, не завжди стабільна синхронізація файлів та залежність від Microsoft екосистеми є недоліками цього сервісу (рис.2.4).

Рисунок 2.4 – OneDrive сервіс

Платформи для захищеного обміну даними:

1. Signal – це месенджер, відомий своєю високою безпекою та конфіденційністю. Він використовує наскрізне шифрування для захисту повідомлень та файлів. Серед переваг Signal варто виділити високий рівень шифрування, відсутність збору даних про користувачів та відкритий вихідний код, що дозволяє незалежну перевірку безпеки. Однак, обмежена функціональність для обміну великими файлами та обмежений обсяг сховища є недоліками цієї платформи (рис. 2.5).

Рисунок 2.5 – Signal сервіс

2. Tresorit – це сервіс для зберігання файлів з акцентом на безпеку та конфіденційність. Він пропонує наскрізне шифрування та контроль доступу до файлів. Серед переваг Tresorit можна виділити високий рівень безпеки, можливість керування правами доступу до файлів та підтримку різних платформ. Проте, висока вартість у порівнянні з іншими сервісами та обмежений безкоштовний обсяг зберігання є його недоліками (рис. 2.6).

Рисунок 2.6 – Tresorit сервіс

3. SpiderOak пропонує безпечне зберігання та обмін файлами з використанням наскрізного шифрування. Сервіс орієнтований на конфіденційність користувачів. Серед переваг SpiderOak варто зазначити високий рівень шифрування, прозору політику конфіденційності та можливість синхронізації файлів між різними пристроями. Однак, відносно складний інтерфейс та обмежений безкоштовний обсяг зберігання є його недоліками (рис. 2.7).

Рисунок 2.7 – SpiderOak сервіс

Аналіз існуючих платформ для обміну інформацією показує, що жодна з них не задовольняє повністю всі вимоги щодо конфіденційного обміну інформацією серед студентів. Більшість популярних платформ мають обмежену конфіденційність та безпеку даних, тоді як спеціалізовані сервіси часто мають обмежену функціональність або високу вартість. Тому для створення нової веб-платформи важливо врахувати досвід існуючих рішень, використати їх сильні сторони та уникнути їхніх недоліків, забезпечивши при цьому високий рівень конфіденційності та безпеки для користувачів.

1.3 Порівняльний аналіз методів шифрування та автентифікації

Шифрування та автентифікація є ключовими компонентами будь-якої системи безпеки, особливо для веб-платформ, що обмінюються конфіденційною інформацією. Вибір відповідних методів шифрування та автентифікації має вирішальне значення для забезпечення надійного захисту даних користувачів. У цьому підпункті ми розглянемо різні методи шифрування та автентифікації, їхні переваги та недоліки, а також порівняємо їх з метою вибору найбільш підходящих

для розробки веб-платформи для конфіденційного обміну інформацією серед студентів.

Шифрування є процесом перетворення інформації в форму, яку можуть прочитати лише авторизовані користувачі. Існують різні методи шифрування, кожен з яких має свої особливості, переваги та недоліки. Симетричне шифрування використовує один і той самий ключ для шифрування та дешифрування даних. Найбільш відомими алгоритмами симетричного шифрування є AES (Advanced Encryption Standard) та DES (Data Encryption Standard). Перевагами симетричного шифрування є висока швидкість шифрування та дешифрування і простота реалізації. Проте, цей метод потребує безпечного обміну ключем між сторонами, а якщо ключ скомпрометовано, вся система стає вразливою.

Асиметричне шифрування використовує пару ключів: відкритий ключ для шифрування та закритий ключ для дешифрування. Найвідоміші алгоритми асиметричного шифрування – RSA (Rivest–Shamir–Adleman) та ECC (Elliptic Curve Cryptography). Перевагами асиметричного шифрування є необхідність передачі лише відкритого ключа, що підвищує безпеку, і можливість використання для цифрових підписів. Недоліками є нижча швидкість шифрування порівняно з симетричним шифруванням та складність реалізації (рис. 2.8).

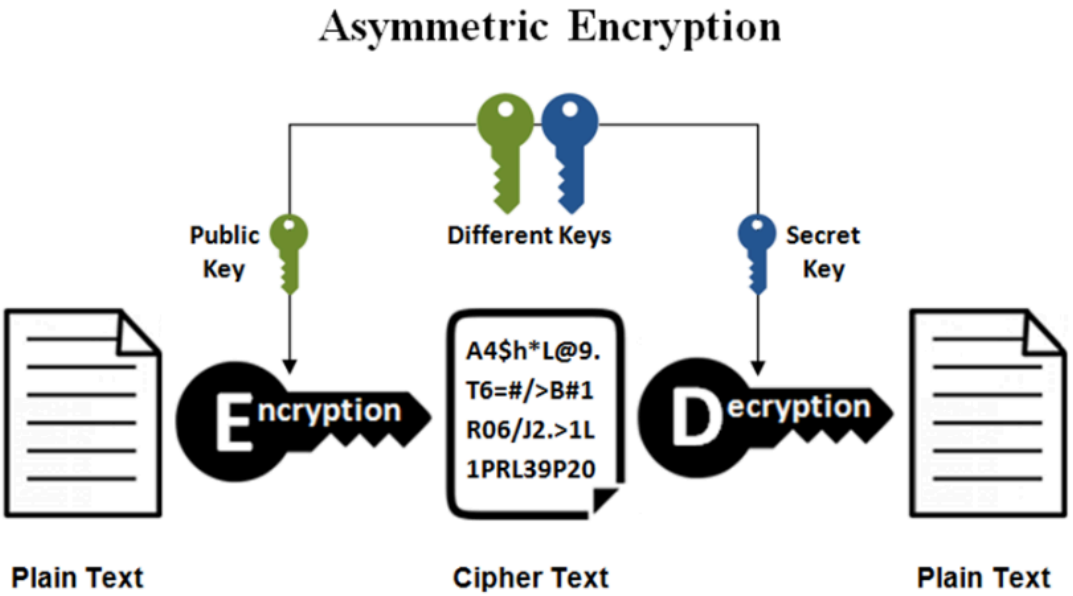


Рисунок 2.8 – Асиметричне шифрування даних

Автентифікація є процесом перевірки особи користувача. Існують різні методи автентифікації, кожен з яких має свої особливості. Парольна автентифікація є найбільш поширеним методом, де користувач вводить свій унікальний пароль для доступу до системи. Перевагами є простота реалізації та використання і широке поширення та звичність для користувачів. Проте, вона вразлива до атак грубої сили та фішингу, і необхідно зберігати паролі у захищеному вигляді.

Двофакторна автентифікація додає додатковий рівень безпеки, вимагаючи від користувача надання двох видів доказів для підтвердження своєї особи, наприклад, пароль та код з мобільного додатку. Це значно підвищує рівень безпеки та ускладнює несанкціонований доступ. Однак, реалізація такого методу є складнішою, і існує можливість втрати доступу до другого фактора, наприклад, у випадку втрати телефону (рис. 2.9).

Рисунок 2.9 – Двофакторна автентифікація

Біометрична автентифікація використовує фізичні характеристики користувача, такі як відбитки пальців, розпізнавання обличчя або голосу, для підтвердження особи. Вона забезпечує високий рівень безпеки і зручність використання, оскільки не потребує запам'ятовування паролів. Проте, її впровадження є більш складним та дорогим, і виникають питання конфіденційності та збереження біометричних даних.

Симетричне шифрування, як AES, забезпечує високу швидкість та ефективність, але потребує надійного методу передачі ключів. Асиметричне шифрування, як RSA та ECC, надає можливість безпечного обміну ключами, але є

менш ефективним за швидкістю. В ідеалі, комбінація обох методів, така як гібридне шифрування, забезпечує оптимальний баланс безпеки та ефективності. Парольна автентифікація є найпростішим методом, але має багато вразливостей. Двофакторна автентифікація значно підвищує безпеку, додаючи додатковий рівень захисту. Біометрична автентифікація пропонує найвищий рівень безпеки, але її впровадження є більш складним та дорогим.

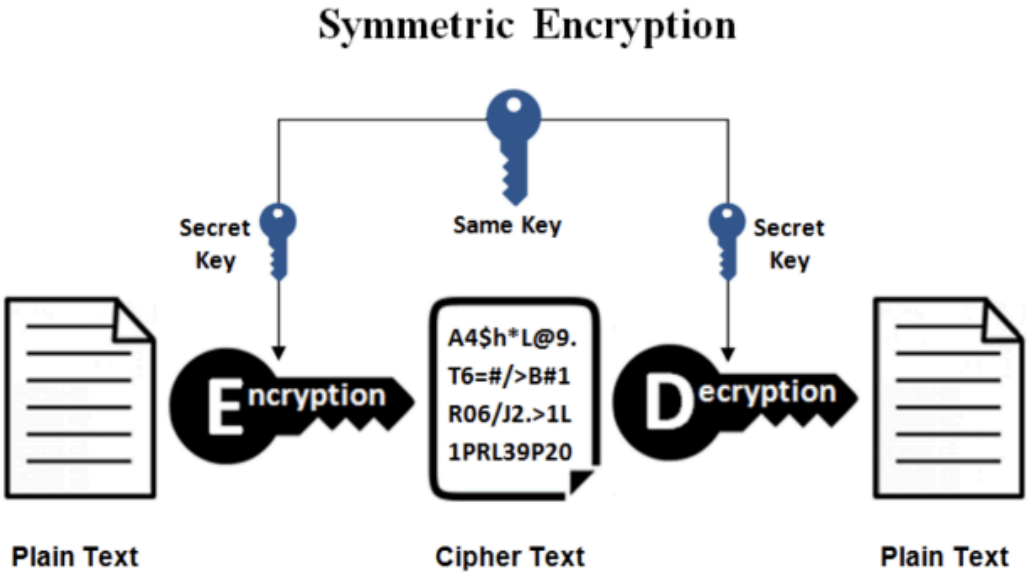


Рисунок 2.10 – Симетричне шифрування

Для розробки веб-платформи для конфіденційного обміну інформацією серед студентів рекомендується використовувати комбінацію симетричного та асиметричного шифрування для забезпечення надійного захисту даних. Для автентифікації користувачів оптимальним варіантом є двофакторна автентифікація, яка забезпечить високий рівень безпеки при помірній складності впровадження. Комбінуючи ці методи, можна створити веб-платформу, яка захищатиме конфіденційну інформацію студентів та забезпечить зручний і безпечний доступ.

1.4 Проблеми та обмеження існуючих рішень

Існуючі рішення для обміну інформацією серед студентів мають різні проблеми та обмеження, які заважають їм повністю задовольнити потреби

користувачів. У цьому підпункті ми розглянемо основні проблеми та обмеження популярних платформ для обміну даними, що використовуються в освітньому середовищі, а також спеціалізованих рішень для захищеного обміну інформацією. Це дозволить визначити, які аспекти необхідно врахувати при розробці нової веб-платформи для конфіденційного обміну інформацією серед студентів.

Google Drive, Dropbox та OneDrive є популярними платформами для зберігання та обміну файлами. Незважаючи на їхню зручність і широке поширення, вони мають низку проблем. Одна з основних проблем полягає в обмеженій конфіденційності даних.

Ці платформи можуть аналізувати вміст файлів для покращення своїх сервісів або навіть передавати дані третім сторонам. Це створює ризик витоку конфіденційної інформації. Крім того, вони залежать від підключення до інтернету, що може бути проблематичним у випадку низької якості з'єднання або відсутності доступу до мережі. Ще одним обмеженням є обмежений обсяг безкоштовного зберігання, що може бути недостатнім для студентів з великим обсягом даних.

Спеціалізовані рішення для захищеного обміну інформацією, такі як Signal, Tresorit та SpiderOak, також мають свої обмеження. Signal відомий своєю високою безпекою та конфіденційністю, але він має обмежену функціональність для обміну великими файлами та обмежений обсяг сховища.

Tresorit пропонує високий рівень безпеки, але його вартість є значно вищою порівняно з іншими сервісами, що може бути проблемою для студентів з обмеженим бюджетом. SpiderOak забезпечує високий рівень шифрування та конфіденційності, але його інтерфейс є відносно складним для користувачів, а обсяг безкоштовного зберігання обмежений.

Існуючі рішення часто мають технічні проблеми, які впливають на зручність використання та безпеку. Наприклад, багато платформ не підтримують наскрізне шифрування за замовчуванням, що створює ризики для безпеки даних. Деякі платформи мають складні інтерфейси, що робить їх важкими для використання менш досвідченими користувачами. Крім того, не всі платформи пропонують можливість гнучкого керування правами доступу, що може бути критично важливим для конфіденційного обміну інформацією.

Автентифікація та управління доступом також можуть бути проблематичними в існуючих рішеннях. Багато платформ використовують лише парольну автентифікацію, яка є вразливою до атак грубої сили та фішингу. Відсутність підтримки двофакторної автентифікації або біометричних методів автентифікації знижує загальний рівень безпеки. Крім того, відсутність можливості гнучкого налаштування прав доступу може призвести до небажаного доступу до конфіденційної інформації.

Інтеграція з іншими сервісами та додатками також може бути проблемою для існуючих рішень. Наприклад, деякі платформи мають обмежену інтеграцію з іншими інструментами, що використовуються в освітньому середовищі, такими як системи управління навчанням (LMS) або інструменти для спільної роботи. Це може створювати труднощі для користувачів, які хочуть використовувати єдину платформу для всіх своїх потреб. Аналіз проблем та обмежень існуючих рішень для обміну інформацією показує, що жодна з них не задовольняє повністю всі потреби студентів щодо конфіденційного обміну інформацією. Популярні платформи мають обмежену конфіденційність даних та залежність від підключення до інтернету, тоді як спеціалізовані рішення часто мають високу вартість або обмежену функціональність.

Технічні проблеми, обмеження в автентифікації та управлінні доступом, а також проблеми інтеграції та сумісності також впливають на зручність використання та безпеку. Тому, при розробці нової веб-платформи для конфіденційного обміну інформацією серед студентів, важливо врахувати всі ці аспекти, щоб створити рішення, яке буде максимально відповідати потребам користувачів.

1.4 Постановка задачі

Розробка веб-платформи для конфіденційного обміну інформацією серед студентів вимагає ретельного аналізу існуючих аналогів, щоб виявити їхні сильні та слабкі сторони. Це дозволить створити ефективне, безпечне та зручне рішення, яке задовольнятиме потреби користувачів. У цьому розділі ми опишемо та

проаналізуємо існуючі платформи для обміну інформацією, зокрема їх функціональні можливості, рівень безпеки та користувацький досвід. Основною метою аналізу є визначення вимог до нової веб-платформи, яка забезпечить високий рівень конфіденційності та безпеки обміну інформацією серед студентів.

Для досягнення цієї мети необхідно:

1. Оцінити функціональні можливості існуючих рішень, таких як Google Drive, Dropbox, OneDrive, Signal, Tresorit та SpiderOak.

2. Визначити рівень безпеки та конфіденційності, що надається цими платформами.

3. Проаналізувати користувацький досвід, зокрема зручність використання, інтерфейс та інтеграцію з іншими сервісами.

4. Виявити основні проблеми та обмеження існуючих рішень, які заважають їм повністю задовольнити потреби студентів щодо конфіденційного обміну інформацією.

Для досягнення поставленої мети необхідно виконати такі завдання:

1. Провести детальний опис існуючих платформ для обміну інформацією, їх функціональних можливостей та особливостей.

2. Визначити методи шифрування та автентифікації, що використовуються цими платформами, та оцінити їхню ефективність у забезпеченні безпеки даних.

3. Оцінити зручність використання та користувацький досвід кожної платформи, включаючи інтерфейс, швидкість роботи та можливості інтеграції з іншими сервісами.

4. Виявити основні проблеми та обмеження існуючих рішень, зокрема технічні проблеми, обмеження в автентифікації та управлінні доступом, проблеми інтеграції та сумісності.

5. Розробити рекомендації щодо поліпшення функціональних можливостей, рівня безпеки та користувацького досвіду для нової веб-платформи.

В результаті виконання поставлених завдань буде отримано всебічний аналіз існуючих рішень для обміну інформацією серед студентів. Це дозволить виявити їхні сильні та слабкі сторони, визначити основні проблеми та обмеження, а також розробити рекомендації для створення нової веб-платформи, яка забезпечить

високий рівень конфіденційності, безпеки та зручності використання. Це, в свою чергу, сприятиме ефективному та безпечному обміну інформацією серед студентів.

Висновки до розділу 1

У розділі розглянуто та оцінено різні платформи для обміну інформацією, що використовуються в освітньому середовищі, а також спеціалізовані рішення для захищеного обміну даними. Наш аналіз показав, що популярні сервіси, такі як Google Drive, Dropbox та OneDrive, пропонують широкий спектр функціональних можливостей, але мають обмеження щодо конфіденційності та безпеки. Вони також залежать від стабільного підключення до інтернету та мають обмежений безкоштовний обсяг зберігання.

Спеціалізовані рішення, такі як Signal, Tresorit та SpiderOak, забезпечують високий рівень безпеки та конфіденційності, але можуть бути менш зручними для користувачів через складний інтерфейс, високу вартість або обмежену функціональність. Технічні проблеми, обмеження в автентифікації та управлінні доступом, а також проблеми інтеграції з іншими сервісами є значними недоліками існуючих рішень. На основі проведеного аналізу можна зробити висновок, що жодне з існуючих рішень не задовольняє повністю потреби студентів щодо конфіденційного обміну інформацією. Для створення нової веб-платформи необхідно врахувати виявлені проблеми та обмеження. Це дозволить розробити рішення, яке забезпечить оптимальний баланс між функціональністю, безпекою та зручністю використання.

Основними рекомендаціями для нової веб-платформи є використання гібридного шифрування для забезпечення високого рівня безпеки, впровадження двофакторної автентифікації для підвищення захисту користувачів, а також розробка інтуїтивно зрозумілого інтерфейсу з можливістю інтеграції з іншими освітніми інструментами. Врахування цих аспектів сприятиме створенню платформи, яка надійно захищатиме конфіденційну інформацію студентів та забезпечить зручний і безпечний обмін даними в освітньому середовищі.

РОЗДІЛ 2. ПРОЕКТУВАННЯ ВЕБПЛАТФОРМИ

3.1 Визначення вимог до системи

Процес розробки веб-платформи для конфіденційного обміну інформацією серед студентів починається з визначення вимог до системи. Це є критичним етапом, який забезпечує відповідність кінцевого продукту потребам користувачів та вимогам безпеки. Вимоги до системи можна розділити на функціональні та нефункціональні.

Функціональні вимоги:

1. Реєстрація користувачів - система повинна забезпечувати можливість реєстрації нових користувачів із валідацією введених даних та підтвердженням електронної пошти. Процес реєстрації має бути простим та інтуїтивно зрозумілим, включати перевірку унікальності імені користувача та електронної пошти. Після введення необхідних даних, користувач повинен отримати електронний лист із посиланням для підтвердження облікового запису. Це запобігає створенню фальшивих акаунтів і забезпечує достовірність інформації.

2. Автентифікація та авторизація - система повинна підтримувати надійні методи автентифікації, включаючи двофакторну автентифікацію (2FA), для захисту облікових записів користувачів. Користувачі повинні вводити свої логін та пароль для доступу до системи, а також мати можливість налаштувати додатковий рівень захисту через OTP (одноразовий пароль), який надсилається на мобільний пристрій або електронну пошту. Це значно зменшує ризик несанкціонованого доступу до облікових записів.

3. Шифрування даних - всі дані, що передаються між клієнтом та сервером, повинні бути шифровані. Це стосується як повідомлень, так і файлів. Використання SSL/TLS для шифрування переданих даних гарантує, що інформація не може бути перехоплена і розшифрована третіми особами. Крім того, дані, що зберігаються на сервері, повинні бути зашифровані для захисту від можливих витоків у випадку злому бази даних.

4. Обмін повідомленнями - користувачі повинні мати можливість надсилати один одному текстові повідомлення, які будуть зберігатися в зашифрованому вигляді. Інтерфейс для обміну повідомленнями повинен бути зручним і підтримувати функції групових чатів, пересилання повідомлень та пошуку по історії повідомлень. Шифрування повідомлень повинно здійснюватися як на стороні клієнта, так і на сервері, забезпечуючи повну конфіденційність.

5. Обмін файлами - система повинна дозволяти користувачам обмінюватися файлами з шифруванням під час передачі та зберігання. Інтерфейс для завантаження та завантаження файлів повинен підтримувати різні типи файлів, забезпечуючи при цьому автоматичне шифрування файлів при завантаженні та дешифрування при завантаженні. Це забезпечить захист конфіденційної інформації, що міститься у файлах.

6. Управління контактами - користувачі повинні мати можливість додавати контакти, створювати групи та управляти своїми контактами. Інтерфейс управління контактами повинен бути зручним, дозволяючи легко додавати або видаляти контакти, створювати списки контактів та групи для групового обміну інформацією. Це сприятиме ефективній комунікації між користувачами.

7. Сповіщення - система повинна надсилати сповіщення користувачам про нові повідомлення або файли. Сповіщення можуть бути у вигляді електронних листів, SMS або push-повідомлень. Це забезпечить своєчасне інформування користувачів про нові події та сприятиме більш ефективному використанню платформи.

Нефункціональні вимоги:

1. Безпека - система повинна бути захищена від зовнішніх атак, включаючи DDoS-атаки, SQL-ін'єкції та інші загрози. Це включає регулярне оновлення програмного забезпечення, використання фаєрволів, систем виявлення та запобігання вторгнень (IDS/IPS), а також застосування інших сучасних засобів кібербезпеки.

2. Продуктивність - платформа повинна обробляти запити користувачів швидко, забезпечуючи низький час відгуку навіть при високому навантаженні. Це передбачає оптимізацію коду, ефективне управління базами даних та використання

CDN (Content Delivery Network) для зменшення затримок.

3. Масштабованість - система повинна бути здатною до масштабування, щоб підтримувати зростаючу кількість користувачів та обсяг даних. Це може включати використання хмарних технологій, контейнеризації та мікросервісної архітектури для забезпечення гнучкості та адаптивності системи.

4. Надійність - система повинна забезпечувати безперервну роботу без збоїв, із мінімальним часом простою. Це включає резервне копіювання даних, відновлення після збоїв, а також моніторинг стану системи для своєчасного виявлення та усунення проблем.

5. Юзабіліті - інтерфейс користувача повинен бути інтуїтивно зрозумілим та зручним для використання, з урахуванням потреб цільової аудиторії. Це включає продуманий дизайн, логічну навігацію та доступність функцій. Регулярне тестування з участю користувачів допоможе вдосконалити інтерфейс.

6. Сумісність - система повинна бути сумісною з різними веб-браузерами та мобільними пристроями, забезпечуючи кросплатформеність. Це включає підтримку основних браузерів (Chrome, Firefox, Safari, Edge) та операційних систем (Windows, macOS, Linux, iOS, Android).

7. Конфіденційність - дані користувачів повинні бути захищені відповідно до міжнародних стандартів та законодавчих вимог щодо конфіденційності. Це передбачає дотримання вимог GDPR, HIPAA та інших відповідних регуляцій, а також впровадження політик конфіденційності та процедур обробки даних.

2.2 Архітектура веб-платформи

Архітектура веб-платформи для конфіденційного обміну інформацією серед студентів визначає структуру та компоненти системи, їх взаємодію, а також принципи, на яких базується її функціонування. Архітектура повинна забезпечити високий рівень безпеки, масштабованість, надійність та зручність використання. У цьому підпункті розглянемо ключові елементи архітектури веб-платформи.

Основні компоненти архітектури:

1. Клієнтська частина (Front-end) - клієнтська частина веб-платформи

відповідає за взаємодію користувача з системою. Основні функції включають відображення користувацького інтерфейсу, обробку введення користувача, та забезпечення комунікації з сервером. Для розробки клієнтської частини зазвичай використовуються сучасні веб-технології, такі як HTML, CSS та JavaScript. Фреймворки, як-от React, Angular або Vue.js, можуть бути використані для створення динамічних та інтерактивних інтерфейсів.

2. Серверна частина (Back-end) - серверна частина відповідає за обробку запитів від клієнтів, управління даними та забезпечення бізнес-логіки платформи. Серверна частина також забезпечує автентифікацію та авторизацію користувачів, шифрування даних, та інтеграцію з базами даних. Популярними технологіями для розробки серверної частини є Node.js, Django, Ruby on Rails та Spring Boot.

3. База даних - база даних зберігає інформацію про користувачів, повідомлення, файли та інші дані, необхідні для функціонування платформи. Використовуються як реляційні (SQL) бази даних, такі як PostgreSQL та MySQL, так і нереляційні (NoSQL) бази даних, такі як MongoDB. Вибір конкретної бази даних залежить від вимог до продуктивності, масштабованості та структури даних.

4. Сервери автентифікації - ці сервери спеціалізуються на обробці запитів автентифікації та авторизації користувачів. Вони забезпечують управління сесіями користувачів, підтримку двофакторної автентифікації та інтеграцію з зовнішніми службами автентифікації, такими як OAuth та OpenID Connect.

5. Сервери шифрування - сервери шифрування відповідають за шифрування та дешифрування даних, що передаються між клієнтами та сервером, а також за шифрування даних, що зберігаються. Це забезпечує конфіденційність та цілісність даних. Використовуються протоколи SSL/TLS для захисту переданих даних, а також сучасні алгоритми шифрування, такі як AES та RSA.

6. Сервери зберігання файлів - ці сервери зберігають файли, які користувачі завантажують на платформу, забезпечуючи їх шифрування та захист від несанкціонованого доступу. Файли можуть зберігатися в хмарних сховищах, таких як AWS S3, або на власних серверах компанії.

7. API шлюзи - API шлюзи забезпечують єдиний вхідний пункт для всіх API запитів до серверної частини платформи. Вони відповідають за маршрутизацію

запитів, управління навантаженням, а також забезпечують безпеку та аутентифікацію на рівні API.

8. Системи моніторингу та логування - ці системи забезпечують моніторинг працездатності платформи, збір та аналіз логів, а також сповіщення про можливі проблеми. Це дозволяє вчасно виявляти та усувати збої, забезпечуючи стабільну роботу системи. Популярні інструменти для моніторингу включають Prometheus, Grafana, ELK Stack (Elasticsearch, Logstash, Kibana).

Принципи архітектури:

1. Безпека - безпека є пріоритетом в архітектурі веб-платформи. Всі дані повинні бути шифровані як під час передачі, так і при зберіганні. Система повинна мати механізми захисту від поширених атак, таких як SQL-ін'єкції, XSS (Cross-Site Scripting), CSRF (Cross-Site Request Forgery) та DDoS атаки. Регулярні перевірки на вразливості та оновлення безпекових патчів також є важливими аспектами.

2. Масштабованість - система повинна бути здатною до масштабування для підтримки зростаючої кількості користувачів та обсягів даних. Це досягається за рахунок використання мікросервісної архітектури, де кожен сервіс може бути масштабований незалежно. Використання контейнерів (Docker) та оркестраційних платформ (Kubernetes) також сприяє ефективному масштабуванню.

3. Надійність - система повинна забезпечувати високу доступність та надійність, мінімізуючи час простою. Це включає використання резервного копіювання даних, механізмів відновлення після збоїв, а також реалізацію кластеризації та реплікації для баз даних.

4. Юзабіліті - інтерфейс користувача повинен бути інтуїтивно зрозумілим та зручним для використання. Важливо проводити регулярні тестування з участю користувачів для збору зворотного зв'язку та вдосконалення інтерфейсу. Це допомагає забезпечити позитивний користувацький досвід та сприяє більш активному використанню платформи.

5. Модульність - архітектура повинна бути модульною, що дозволяє легко додавати нові функції та компоненти без впливу на роботу інших частин системи. Це досягається за рахунок використання мікросервісної архітектури та дотримання принципів SOLID в розробці.

Загалом, архітектура веб-платформи повинна забезпечити безпеку, надійність, масштабованість та зручність використання, що є ключовими вимогами для успішної реалізації проекту з конфіденційного обміну інформацією серед студентів.

2.3 Вибір технологій для розробки платформи

Вибір технологій для розробки веб-платформи є ключовим аспектом, що впливає на її функціональність, безпеку, масштабованість та зручність використання. У цьому підпункті розглянемо основні технології, що можуть бути використані для створення веб-платформи для конфіденційного обміну інформацією серед студентів.

Клієнтська частина (Front-end):

1. HTML, CSS, JavaScript: HTML (HyperText Markup Language) є основою для створення веб-сторінок, визначаючи структуру контенту. CSS (Cascading Style Sheets) використовується для стилізації та розташування елементів на сторінці, забезпечуючи привабливий і функціональний дизайн. JavaScript дозволяє створювати інтерактивні елементи на сторінці, такі як форми, спливаючі вікна та анімації.

2. React: React — це популярна бібліотека для розробки користувацьких інтерфейсів, розроблена компанією Facebook. Вона забезпечує ефективне оновлення DOM (Document Object Model) за рахунок використання віртуального DOM, що значно покращує продуктивність. React також підтримує компонентний підхід, що дозволяє розробляти повторно використовувані компоненти інтерфейсу.

3. Vue.js: Vue.js — це прогресивний фреймворк для створення користувацьких інтерфейсів. Він відомий своєю гнучкістю, простотою інтеграції з існуючими проектами та низьким порогом входу для новачків. Vue.js забезпечує високий рівень продуктивності та легкість у навчанні.

4. Angular: Angular — це фреймворк від Google для створення складних Landing page додатків (SPA). Він включає безліч вбудованих інструментів для управління станом додатка, маршрутизації, та двостороннього зв'язку даних, що

робить його потужним інструментом для розробки великих і масштабованих додатків.

Серверна частина (Back-end):

1. Node.js: Node.js — це середовище виконання JavaScript на сервері, що дозволяє розробляти швидкі та масштабовані мережеві додатки. Використовуючи подієву архітектуру та неблокуючий ввід/вивід, Node.js забезпечує високу продуктивність. Популярний фреймворк Express.js може бути використаний для створення веб-додатків на базі Node.js.

2. Django: Django — це високорівневий фреймворк для розробки веб-додатків на мові Python, який надає багато готових компонентів для швидкого створення безпечних та підтримуваних веб-додатків. Він дотримується принципу "батареї в комплекті", що означає, що більшість необхідних функцій вже вбудовані.

3. Ruby on Rails: Ruby on Rails — це веб-фреймворк, написаний на мові програмування Ruby, який використовує парадигму MVC (Model-View-Controller). Rails відомий своєю продуктивністю та швидкістю розробки завдяки багатій екосистемі гемів (бібліотек).

4. Spring Boot: Spring Boot — це фреймворк для створення додатків на мові Java, що значно спрощує конфігурацію та розгортання додатків. Spring Boot підходить для створення високопродуктивних та масштабованих веб-додатків з можливістю інтеграції з великою кількістю сторонніх бібліотек та сервісів.

База даних:

1. PostgreSQL: PostgreSQL — це потужна реляційна база даних з відкритим вихідним кодом, яка підтримує різноманітні типи даних та забезпечує високу продуктивність і надійність. PostgreSQL добре підходить для обробки складних запитів та масштабування.

2. MySQL: MySQL — це ще одна популярна реляційна база даних з відкритим вихідним кодом, яка відома своєю швидкістю та надійністю. Вона широко використовується у веб-розробці завдяки своїй простоті та ефективності.

3. MongoDB: MongoDB — це нереляційна база даних (NoSQL), що зберігає дані у форматі BSON (розширення JSON). Вона добре підходить для додатків, що потребують гнучкого управління даними, таких як обробка великих обсягів

неструктурованої інформації.

Сервери автентифікації:

1. OAuth: OAuth — це стандартний протокол для авторизації, який дозволяє стороннім сервісам доступ до ресурсів користувача без передачі пароля. OAuth 2.0 забезпечує безпечну передачу токенів доступу між клієнтом та сервером.

2. OpenID Connect: OpenID Connect — це рівень автентифікації, побудований поверх OAuth 2.0, який забезпечує додатковий рівень безпеки для автентифікації користувачів. Він дозволяє користувачам входити в різні додатки, використовуючи один обліковий запис.

Сервери шифрування:

1. SSL/TLS: SSL (Secure Sockets Layer) та TLS (Transport Layer Security) — це протоколи шифрування, які забезпечують захист даних під час їх передачі між клієнтом та сервером. Вони забезпечують цілісність даних, конфіденційність та аутентифікацію.

2. AES (Advanced Encryption Standard): AES — це симетричний алгоритм шифрування, який широко використовується для захисту даних завдяки своїй ефективності та високому рівню безпеки. Він підтримує ключі розміром 128, 192 та 256 біт.

3. RSA (Rivest-Shamir-Adleman): RSA — це асиметричний алгоритм шифрування, який використовується для безпечної передачі ключів симетричного шифрування. Він забезпечує високий рівень безпеки за рахунок використання публічних та приватних ключів.

Сервери зберігання файлів:

1. AWS S3: Amazon Web Services S3 (Simple Storage Service) — це хмарне сховище даних, яке забезпечує високу доступність, надійність та безпеку. Воно підтримує шифрування даних як під час передачі, так і при зберіганні.

2. Google Cloud Storage: Google Cloud Storage — це сервіс для зберігання великих обсягів даних у хмарі. Він забезпечує надійне зберігання з можливістю шифрування та глобального доступу.

API шлюзи

1. Kong: Kong — це популярний API шлюз з відкритим вихідним кодом, який

забезпечує маршрутизацію запитів, управління навантаженням, а також захист і аутентифікацію на рівні API.

2. Apigee: Apigee від Google Cloud — це комерційний API шлюз, що надає розширені можливості для управління API, включаючи аналітику, моніторинг та безпеку.

Системи моніторингу та логування:

1. Prometheus: Prometheus — це система моніторингу з відкритим вихідним кодом, яка забезпечує збір та збереження метрик, а також налаштування алертів для сповіщення про проблеми в реальному часі.

2. Grafana: Grafana — це інструмент для візуалізації даних та метрик, зібраних Prometheus та іншими системами моніторингу. Він дозволяє створювати дашборди для наочного відображення стану системи.

3. ELK Stack (Elasticsearch, Logstash, Kibana): ELK Stack — це набір інструментів для збору, аналізу та візуалізації логів. Elasticsearch забезпечує пошук і аналітику, Logstash обробляє і передає дані, а Kibana надає інтерфейс для візуалізації.

Вибір технологій повинен базуватися на вимогах до системи, наявних ресурсах та досвіді команди розробників. Правильний вибір забезпечить успішну реалізацію веб-платформи, яка відповідатиме всім вимогам конфіденційного обміну інформацією серед студентів.

2.4 Моделювання бази даних

Моделювання бази даних є одним із ключових етапів розробки веб-платформи для конфіденційного обміну інформацією серед студентів. Правильно спроектована база даних забезпечує ефективне зберігання, доступ та обробку даних, а також підтримує вимоги до безпеки та цілісності інформації. У цьому підпункті розглянемо основні аспекти моделювання бази даних для даної платформи.

Вимоги до бази даних:

1. Безпека та конфіденційність - база даних повинна забезпечувати захист від

несанкціонованого доступу та витоку даних. Це досягається за рахунок використання шифрування даних як при зберіганні, так і при передачі, а також за допомогою механізмів контролю доступу.

2. Цілісність даних - база даних повинна підтримувати цілісність даних, забезпечуючи правильність та узгодженість збереженої інформації. Це включає використання ключів, обмежень та тригерів.

3. Масштабованість - база даних повинна бути здатною до масштабування для обробки зростаючих обсягів даних та кількості користувачів. Це включає підтримку розподілених систем зберігання та обробки даних.

4. Продуктивність - база даних повинна забезпечувати високу продуктивність для швидкої обробки запитів та доступу до даних, особливо у випадках великого навантаження.

Для даної веб-платформи доцільно використовувати реляційну базу даних, таку як PostgreSQL або MySQL, через їх підтримку складних запитів, високий рівень безпеки та узгодженість даних. Можливо також розглянути використання нереляційної бази даних (NoSQL), такої як MongoDB, для зберігання неструктурованих або напівструктурованих даних.

Основні таблиці та їх взаємозв'язки:

1. Користувачі (Users) - ця таблиця зберігає інформацію про користувачів платформи, включаючи унікальний ідентифікатор, ім'я користувача, електронну пошту, хеш пароля, роль (студент, адміністратор) та дату реєстрації.

2. Повідомлення (Messages) - таблиця для зберігання повідомлень, що обмінюються між користувачами. Вона включає унікальний ідентифікатор повідомлення, ідентифікатори відправника та отримувача, текст повідомлення, час відправлення та статус (прочитано/непрочитано).

3. Файли (Files) - таблиця для зберігання інформації про файли, завантажені користувачами. Вона включає унікальний ідентифікатор файлу, ім'я файлу, тип файлу, шлях до файлу на сервері, ідентифікатор користувача, який завантажив файл, та час завантаження.

4. Сесії (Sessions) - таблиця для зберігання інформації про сесії користувачів. Вона включає унікальний ідентифікатор сесії, ідентифікатор користувача, токен

сесії, час початку та завершення сесії, а також IP-адресу користувача.

5. Логи (Logs) - таблиця для зберігання логів системних подій, таких як входи в систему, помилки, зміни налаштувань та інші дії користувачів. Вона включає унікальний ідентифікатор логу, тип події, ідентифікатор користувача, час події та опис події.

Моделювання ER-діаграми.

ER-діаграма (діаграма сутність-зв'язок) є важливим інструментом для моделювання бази даних. Вона допомагає візуалізувати структуру бази даних, визначити сутності (таблиці) та зв'язки між ними.

1. Сутності:

- користувачі (Users);
- повідомлення (Messages);
- файли (Files);
- сесії (Sessions);
- логи (Logs).

2. Зв'язки:

- кожен користувач може відправляти та отримувати багато повідомлень (зв'язок "один-до-багатьох" між користувачами та повідомленнями);
- кожен користувач може завантажити багато файлів (зв'язок "один-до-багатьох" між користувачами та файлами);
- кожен користувач може мати багато сесій (зв'язок "один-до-багатьох" між користувачами та сесіями);
- кожен лог може бути пов'язаний з одним користувачем (зв'язок "багато-до-одного" між логами та користувачами).

Забезпечення цілісності та безпеки даних.

1. Ключі та обмеження:

- первинні ключі (Primary keys) для унікальної ідентифікації записів у таблицях;
- зовнішні ключі (Foreign keys) для забезпечення зв'язків між таблицями;
- обмеження цілісності (Integrity constraints) для забезпечення правильності даних (наприклад, обов'язковість заповнення полів).

2. Шифрування:

– використання шифрування для захисту конфіденційних даних, таких як паролі користувачів (хешування паролів з використанням алгоритмів, таких як bcrypt);

– шифрування даних при зберіганні та передачі з використанням AES для симетричного шифрування та SSL/TLS для захисту переданих даних;

3. Контроль доступу:

– Використання ролей та дозволів для обмеження доступу до даних та функцій платформи;

– Впровадження механізмів автентифікації та авторизації користувачів.

Підтримка масштабованості та продуктивності.

1. Індексація:

– використання індексів для прискорення пошуку та доступу до даних;

– оптимізація індексів для часто використовуваних запитів.

2. Шардінг (Sharding):

– розподіл даних на кілька баз даних для підвищення продуктивності та забезпечення горизонтального масштабування.

3. Кешування:

– використання механізмів кешування для зменшення навантаження на базу даних та прискорення доступу до часто запитуваних даних (наприклад, Redis або Memcached).

Моделювання бази даних є критичним етапом розробки веб-платформи, оскільки забезпечує ефективне управління даними та підтримку всіх функцій системи. Правильне проектування структури бази даних, визначення сутностей та зв'язків, а також впровадження механізмів безпеки та масштабованості забезпечать надійну та продуктивну роботу платформи для конфіденційного обміну інформацією серед студентів.

2.6 Опис користувацького інтерфейсу

Опис користувацького інтерфейсу (UI) є важливим етапом розробки веб-платформи, оскільки від нього залежить зручність використання та ефективність взаємодії користувачів з системою. У цьому підпункті розглянемо основні компоненти та принципи дизайну інтерфейсу веб-платформи для конфіденційного обміну інформацією серед студентів.

Основні принципи дизайну

1. Простота та інтуїтивність: Інтерфейс повинен бути простим і зрозумілим для користувачів, незалежно від їх технічного рівня. Інтуїтивна навігація та логічна структура допоможуть користувачам швидко орієнтуватися на платформі.

2. Консистентність: Дизайн інтерфейсу повинен бути послідовним на всіх сторінках платформи. Використання однакових елементів, стилів та шаблонів допоможе уникнути плутанини та забезпечить узгодженість.

3. Відповідність завданням: Інтерфейс має бути спроектований з урахуванням завдань, які вирішує платформа. Кожен елемент інтерфейсу повинен відповідати конкретним потребам користувачів.

4. Доступність: Інтерфейс повинен бути доступним для всіх користувачів, включаючи людей з обмеженими можливостями. Це включає підтримку екранних читачів, можливість зміни розміру тексту та кольорових схем.

Основні компоненти користувацького інтерфейсу

1. Головна сторінка: Головна сторінка є першим екраном, який бачить користувач після входу на платформу. Вона містить привітання, коротку інформацію про платформу, останні новини та посилання на основні розділи сайту. На головній сторінці можуть бути розміщені інформаційні банери, оголошення та рекомендації для користувачів.

2. Реєстрація та автентифікація: Сторінки реєстрації та входу дозволяють користувачам створювати нові облікові записи та входити в систему. Форма реєстрації включає поля для введення основних даних користувача (ім'я, електронна пошта, пароль) та можливість підтвердження електронної пошти. Сторінка входу дозволяє користувачам вводити свій логін і пароль, а також

передбачає можливість відновлення пароля.

3. Панель керування: Панель керування (Dashboard) є центральним елементом інтерфейсу, де користувачі можуть бачити огляд своєї активності, отримувати сповіщення та керувати своїм обліковим записом. Вона містить посилання на основні функціональні розділи, такі як профіль користувача, повідомлення, файли та налаштування.

4. Профіль користувача: Сторінка профілю дозволяє користувачам переглядати та редагувати свою особисту інформацію, змінювати пароль, налаштовувати параметри конфіденційності та управління обліковим записом. Тут користувачі можуть також бачити свої контакти та налаштовувати сповіщення.

5. Повідомлення: Розділ повідомлень надає користувачам можливість обміну текстовими повідомленнями з іншими користувачами платформи. Інтерфейс повідомлень включає список контактів, панель для створення нового повідомлення та вікно чату. Користувачі можуть переглядати історію повідомлень, відповідати на отримані повідомлення та додавати вкладення.

6. Управління файлами: Цей розділ дозволяє користувачам завантажувати, переглядати, зберігати та обмінюватися файлами. Інтерфейс включає список завантажених файлів, панель для завантаження нових файлів та можливість створення папок для організації файлів. Функції пошуку та сортування допомагають користувачам швидко знаходити потрібні файли.

7. Налаштування: Розділ налаштувань дозволяє користувачам змінювати параметри свого облікового запису та налаштовувати інтерфейс платформи відповідно до своїх потреб. Це включає налаштування мови, теми оформлення, параметрів конфіденційності та безпеки.

8. Допомога та підтримка: Розділ допомоги надає користувачам доступ до документації, поширених запитань (FAQ) та контактної інформації служби підтримки. Інтерфейс включає пошук по базі знань, інструкції з використання платформи та можливість звернення до технічної підтримки.

Прототипування та тестування інтерфейсу.

Для створення ефективного користувацького інтерфейсу необхідно провести прототипування та тестування з користувачами. Прототипування дозволяє

візуалізувати та оцінити інтерфейс до його повної реалізації, виявити можливі проблеми та внести корективи на ранніх етапах розробки. Тестування з реальними користувачами допомагає отримати зворотний зв'язок та забезпечити високу якість та зручність використання платформи.

Розробка користувацького інтерфейсу є важливим етапом створення веб-платформи для конфіденційного обміну інформацією серед студентів. Приділяючи увагу принципам дизайну, основним компонентам інтерфейсу та процесу прототипування і тестування, можна створити зручний та ефективний інтерфейс, що забезпечить позитивний досвід користувачів та підвищить продуктивність роботи з платформою.

Висновки до розділу 2

Проектування веб-платформи для конфіденційного обміну інформацією серед студентів є критичним етапом, що визначає успіх та ефективність кінцевого продукту. У цьому розділі було детально розглянуто основні аспекти, які включають огляд сучасних технологій для захисту даних, визначення вимог до системи, проектування архітектури веб-платформи, вибір технологій для розробки, моделювання бази даних та опис користувацького інтерфейсу.

Зокрема, аналіз сучасних технологій для захисту даних дозволив вибрати найефективніші методи шифрування, автентифікації та авторизації, які забезпечують високий рівень безпеки та конфіденційності. Визначення вимог до системи допомогло сформувати чітке уявлення про необхідні функціональні та нефункціональні характеристики платформи, що є основою для подальшого проектування та розробки.

Проектування архітектури веб-платформи, з акцентом на безпеку, масштабованість та продуктивність, заклало міцний фундамент для створення стабільного та надійного продукту. Вибір відповідних технологій для розробки, таких як мови програмування, фреймворки та інструменти, забезпечив основу для реалізації всіх необхідних функцій та інтеграції з іншими системами.

Моделювання бази даних гарантувало ефективне управління даними та

забезпечення їх цілісності та безпеки. Опис користувацького інтерфейсу забезпечив зручність та інтуїтивність використання платформи, що є важливим для залучення та утримання користувачів.

Усі ці аспекти разом створюють цілісну та ефективну систему, яка відповідає потребам користувачів та забезпечує високий рівень захисту інформації. Завдяки ретельному підходу до проектування веб-платформи, вона має потенціал стати надійним інструментом для конфіденційного обміну інформацією серед студентів та може бути успішно використана в інших сферах, де важлива безпека даних.

РОЗДІЛ 3. РОЗРОБКА ТА ДИЗАЙН ВЕБ-ПЛАТФОРМИ ДЛЯ КОНФІДЕНЦІЙНОГО ОБМІНУ ІНФОРМАЦІЄЮ СЕРЕД СТУДЕНТІВ

3.1 Написання Use Case Diagram для веб-платформи

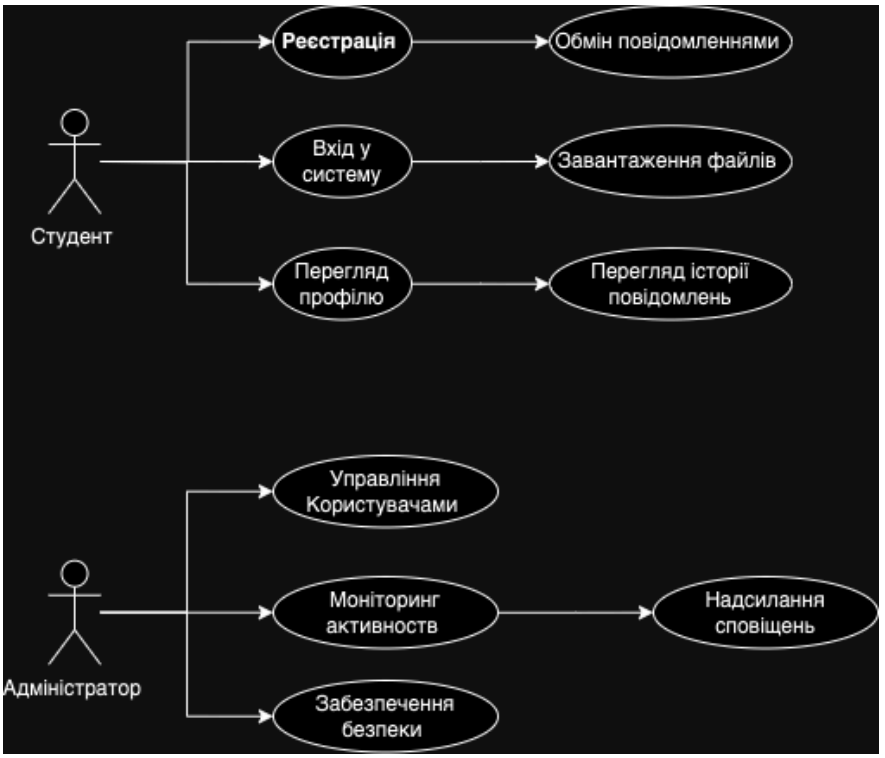


Рисунок 3.1 – Use Case Diagram діаграма

Use Case Diagram (діаграма прецедентів) є візуальним представленням взаємодій між користувачами (акторами) та функціями (прецедентами) системи. В UML (Unified Modeling Language) ця діаграма відображає, як різні ролі взаємодіють із системою, як показано на рисунку 3.1. Давайте розглянемо її докладніше.

Актори: На діаграмі прецедентів акторами є суб'єкти, які взаємодіють із системою. В нашій системі основними акторами є “Студент” та “Адміністратор”.

Прецеденти (варіанти використання):Прецеденти описують функціональні можливості, які система надає користувачам. Основні варіанти використання включають:

1. Реєстрація: Студент створює новий обліковий запис на платформі.

2. Вхід у систему: Студент проходить процедуру авторизації.

3. Обмін повідомленнями: Студент надсилає та отримує приватні повідомлення.

4. Перегляд профілю: Студент переглядає та редагує свій профіль.

5. Завантаження файлів: Студент обмінюється файлами через платформу.

6. Перегляд історії повідомлень: Студент переглядає історію повідомлень.

7. Управління користувачами: Адміністратор керує обліковими записами користувачів.

8. Моніторинг активності: Адміністратор контролює активність на платформі.

9. Забезпечення безпеки: Адміністратор налаштовує параметри безпеки системи.

10. Надсилання сповіщень: Адміністратор надсилає сповіщення користувачам про оновлення або важливі події.

Ця діаграма дозволяє зрозуміти, як система взаємодіє з користувачами і які функціональні можливості вона їм надає. Вона є важливим інструментом для проектування та аналізу системи.

Детальний опис актора та прецедентів:

Актори:

– студент: користувач платформи, який використовує її для обміну конфіденційною інформацією;

– адміністратор: користувач з правами адміністрування, відповідальний за управління платформою та її безпекою.

Прецеденти:

1. Реєстрація:

– опис: процес створення нового користувача на платформі;

– актор: студент.

2. Вхід у систему:

– опис: процедура входу в систему після введення облікових даних;

– актор: студент.

3. Обмін повідомленнями:

- опис: надсилання та отримання приватних повідомлень між студентами;
- актор: студент.

4. Перегляд профілю:

- опис: огляд та редагування особистої інформації в профілі користувача;
- актор: студент.

5. Завантаження файлів:

- опис: обмін файлами між користувачами платформи;
- актор: студент.

6. Перегляд історії повідомлень:

- опис: огляд історії обміну повідомленнями;
- актор: студент.

7. Управління користувачами:

- опис: додавання, редагування та видалення користувачів з платформи;
- актор: адміністратор.

8. Моніторинг активності:

- опис: відстеження активності користувачів на платформі;
- актор: адміністратор.

9. Забезпечення безпеки:

- опис: налаштування та забезпечення захисту даних на платформі;
- актор: адміністратор.

10. Надсилання сповіщень:

- опис: надсилання сповіщень користувачам про оновлення або важливі події;
- актор: адміністратор.

Ця структура надає чітке розуміння того, як веб-платформа для конфіденційного обміну інформацією серед студентів взаємодіє з користувачами та які функціональні можливості вона забезпечує.

3.2 Створення макету веб-платформи

У сучасній розробці веб-додатків важливе значення мають UI/UX дизайн.

Інтерфейс користувача (UI) створює перше враження про продукт і визначає, наскільки його використання буде зручним та привабливим. Досвід користувача (UX) визначає, наскільки задоволеним і ефективним буде користувач під час взаємодії з продуктом.

UI дизайн допомагає привернути увагу користувачів і зберегти її на веб-платформі. Якщо інтерфейс привабливий, зрозумілий і ергономічний, користувачі будуть більш зацікавлені у його використанні.

UX дизайн забезпечує інтуїтивну взаємодію користувача з продуктом. Він враховує потреби користувачів і дозволяє їм легко знаходити інформацію, виконувати завдання та отримувати задоволення від використання продукту. Розробка якісного UI/UX є критично важливою для успіху проекту та задоволення потреб користувачів. Веб-розробка на основі React дозволяє поділити інтерфейс на невеликі незалежні компоненти, які можуть бути багаторазово використані. Це спрощує управління станом додатка та полегшує розробку великих проектів.

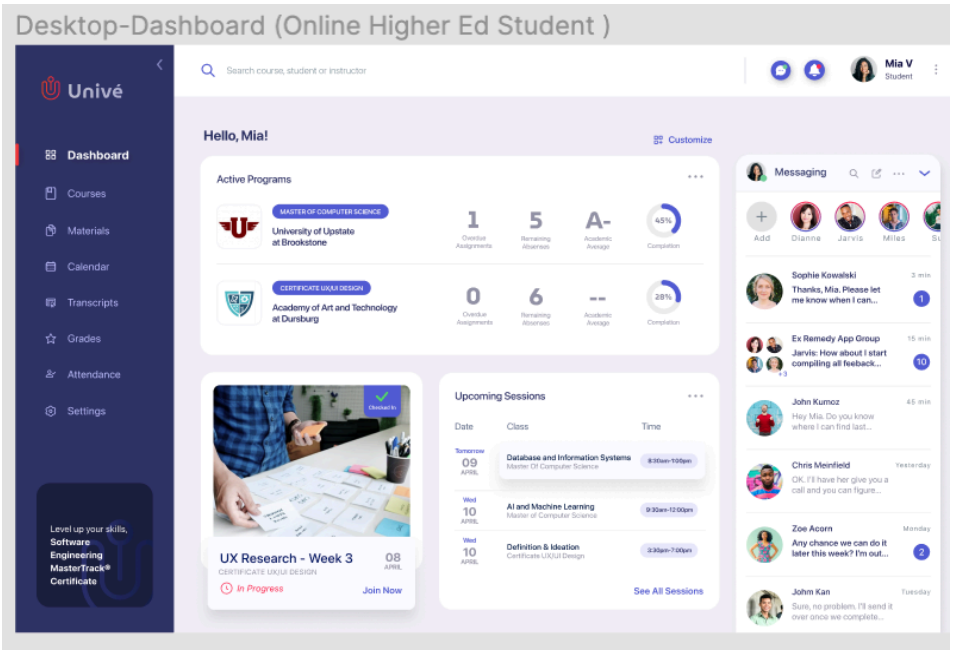


Рисунок 3.2 – Макет веб-платформи

На головній сторінці веб-платформи для конфіденційного обміну інформацією серед студентів будуть доступні основні функції користувача. Процес створення макету включає кілька етапів, що дозволяють візуалізувати та

спланувати вигляд і функціональність сайту до початку його реалізації. Ця фаза є критично важливою для успіху проекту та задоволення потреб користувачів. Основні кроки цього процесу включають:

1. Збір вимог: Визначення детальних вимог до веб-платформи, таких як функціональні можливості, цільова аудиторія та особливості взаємодії.

2. Дослідження та конкурентний аналіз: Аналіз існуючих платформ для обміну інформацією, щоб визначити найкращі дизайнерські рішення та функціональні можливості для власного макету.

3. Розробка структури сайту: Встановлення логіки та ієрархії сторінок сайту, включаючи навігаційні можливості.

4. Створення скетчів та візуалізація: Створення початкової візуальної концепції головної сторінки за допомогою ручних скетчів або програм для дизайну.

5. Прототипування та тестування: Створення інтерактивних прототипів для перевірки функціональності та зручності використання, а також проведення тестування з користувачами для отримання зворотного зв'язку і внесення коректив.

Дизайн елементи

Кольори:

Для дизайну платформи було обрано сучасну кольорову палітру, яка створює приємне та комфортне враження. Основні кольори включають:

– основний колір: синій (#4A90E2), що символізує довіру та професіоналізм;

– акцентний колір: помаранчевий (#F5A623), що додає динаміки та привертає увагу до важливих елементів;

– фон: світло-сірий (#F0F0F0), що забезпечує нейтральне тло для всіх елементів інтерфейсу.

Шрифти:

Для текстових елементів використано два основні шрифти:

– основний шрифт: Roboto, який добре читається та підходить для заголовків та основного тексту;

– додатковий шрифт: Open Sans, що використовується для додаткових пояснень та менш важливих текстів.

Дизайн елементів:

1. Головна сторінка:

– шапка: включає логотип, навігаційне меню та іконку профілю користувача;

– основний контент: відображає останні повідомлення, активні сесії та нагадування про майбутні події;

– бічна панель: містить меню з посиланнями на основні розділи платформи, такі як курси, календар, матеріали, оцінки, відвідування та налаштування.

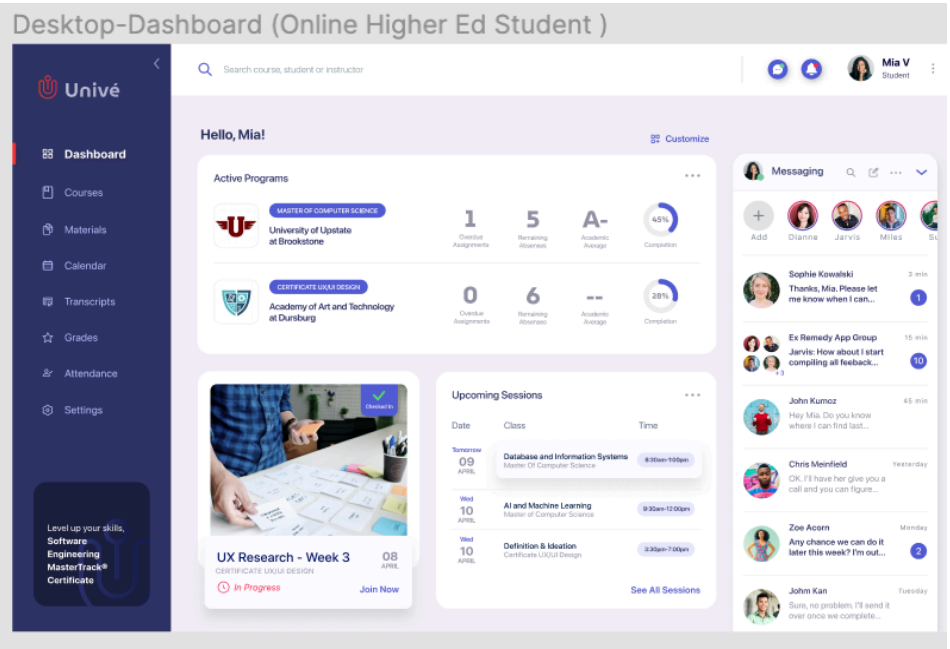


Рисунок 3.3 – Макет веб-платформи

2. Профіль користувача:

– інформація про користувача: включає фото профілю, ім'я, статус та коротку біографію;

– налаштування: дозволяє користувачу редагувати особисті дані та налаштування безпеки.

3. Месенджер:

– список контактів: відображає список контактів з можливістю пошуку та фільтрації.

– чат: відображає історію повідомлень з можливістю надсилання текстових повідомлень та файлів.

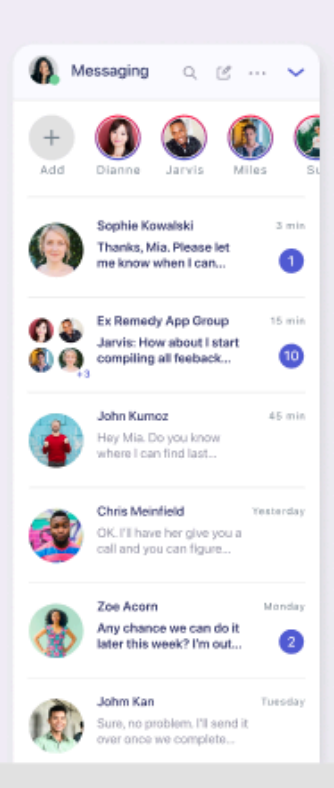


Рисунок 3.4 – Макет месенджеру

4. Календар:

– події: відображає майбутні події, заняття та дедлайни для завдань;

– синхронізація: можливість синхронізації з особистими календарями користувачів.

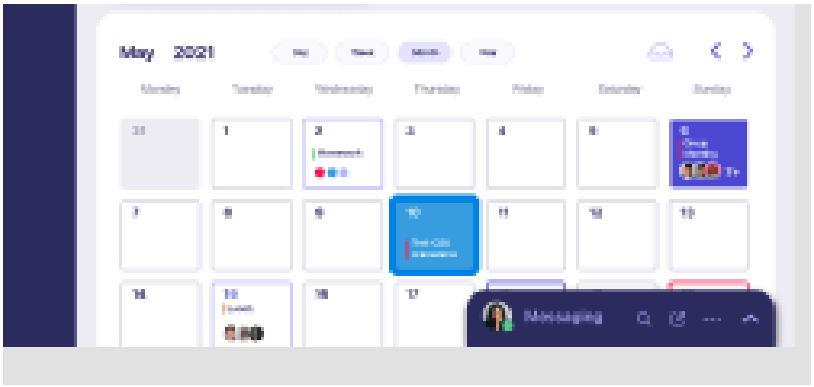


Рисунок 3.5 – Макет календарю

5. Оцінки та завдання:

– оцінки: відображення поточних оцінок та статистики по курсам;

– завдання: відображення завдань з можливістю завантаження виконаних робіт та перегляду коментарів викладачів.

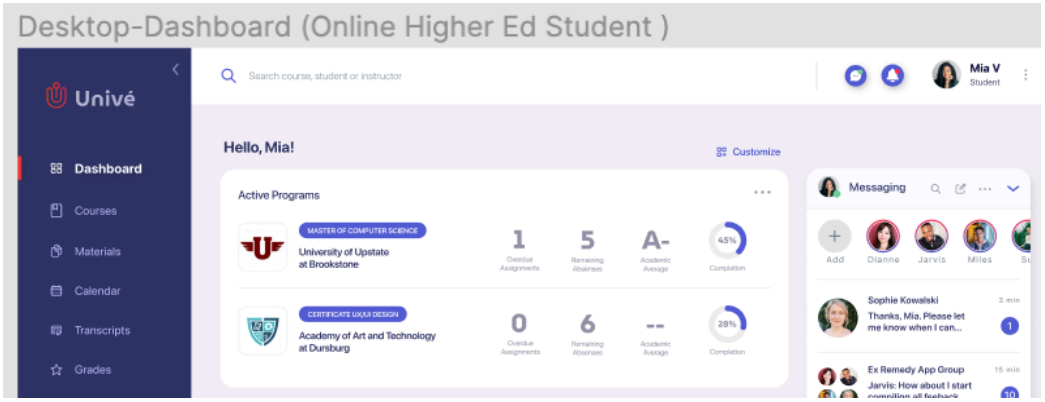


Рисунок 3.6 – Макет веб-платформи

Створення макету веб-платформи для конфіденційного обміну інформацією серед студентів є важливим етапом у процесі розробки. Макет забезпечує основу для подальшої реалізації функціонального та ефективного сайту, орієнтованого на задоволення потреб користувачів і привернення їхньої уваги. Інтерфейс платформи розроблений з урахуванням сучасних вимог до UI/UX дизайну, що сприяє зручній та інтуїтивний взаємодії користувачів з системою.

3.3 Реалізація основного функціоналу процесу реєстрації

Процес реєстрації на веб-платформі для конфіденційного обміну інформацією серед студентів є ключовим етапом, який забезпечує доступ користувачів до системи. Цей процес повинен бути простим, інтуїтивно зрозумілим та безпечним, щоб забезпечити зручність та захист даних користувачів.

Кроки реалізації процесу реєстрації:

1. Створення реєстраційної форми:

– поля форми: реєстраційна форма включає поля для введення імені, прізвища, адреси електронної пошти, пароля та підтвердження пароля;

– валідація на стороні клієнта: перевірка коректності введених даних, включаючи формат електронної пошти, мінімальну довжину пароля та відповідність паролів.

2. Передача даних на сервер:

– API-запит: використання методу POST для передачі даних форми на сервер у форматі JSON;

– асинхронність: забезпечення асинхронної передачі даних для покращення користувацького досвіду.

3. Валідація даних на сервері:

– перевірка унікальності: сервер перевіряє, чи є введена адреса електронної пошти унікальною і не використовується іншими користувачами;

– додаткова валідація: сервер виконує додаткову перевірку коректності введених даних;

4. Збереження даних в базі даних:

– хешування паролів: використання бібліотеки bcrypt для хешування паролів перед їх збереженням у базі даних;

– збереження запису: створення нового запису користувача в базі даних.

5. Відправка підтвердження реєстрації:

– електронний лист: надсилання листа з посиланням для підтвердження адреси електронної пошти користувача;

– активація облікового запису: користувач активує свій обліковий запис, натискаючи на посилання в листі.

6. Відображення повідомлення користувачу:

– підтвердження: після успішного завершення реєстрації користувач отримує повідомлення про необхідність підтвердження електронної пошти.

7. Інструменти та технології:

– Front-end: реєстраційна форма створена з використанням React для забезпечення інтерактивної валідації на стороні клієнта;

– Back-end: серверна логіка реалізована на Node.js з використанням фреймворку Express.js;

– база даних: використання реляційної бази даних PostgreSQL для

збереження даних користувачів;

– безпека: використання bcrypt для хешування паролів і HTTPS для захищеної передачі даних.

Приклад реалізації.

Front-end 1 (React):

```
import React, { useState } from 'react';
import axios from 'axios';
    const [formData, setFormData] = useState({
        firstName: '',
        lastName: '',
        email: '',
        password: '',
        confirmPassword: ''
    });
    const handleChange = (e) => {
        setFormData({
            ...formData,
            [e.target.name]: e.target.value
        });
    };
    const handleSubmit = async (e) => {
        e.preventDefault();
        try {
                  const response = await axios.post('/api/register',
formData);
            console.log('Registration successful:', response.data);
        } catch (error) {
            console.error('Error during registration:', error);
        }
    };
```

Front-end 2 (React):

```
    return (
Back-end (Node.js/Express):
    const { firstName, lastName, email, password } = req.body;
```

```
          const userCheck = await pool.query('SELECT * FROM users WHERE
email = $1', [email]);
          if (userCheck.rows.length > 0) {
               return res.status(400).json({ error: 'Email already in use'
});
          }
          const hashedPassword = await bcrypt.hash(password, 10);
          const newUser = await pool.query(
               'INSERT INTO users (first_name, last_name, email, password)
VALUES ($1, $2, $3, $4) RETURNING *',
              [firstName, lastName, email, hashedPassword]
          );
          res.status(201).json(newUser.rows[0]);
     });
     app.listen(5000, () => {
          console.log('Server is running on port 5000');
     });
```

Процес реєстрації на веб-платформі для конфіденційного обміну інформацією серед студентів забезпечує безпеку та зручність використання. Важливо забезпечити надійну валідацію даних, захист паролів та інтуїтивний інтерфейс для користувачів. Реалізація описаних кроків допоможе створити ефективну та безпечну систему реєстрації.

3.4 Процес створення головної сторінки

Головна сторінка веб-платформи для конфіденційного обміну інформацією серед студентів є ключовим елементом, який забезпечує перше враження про сайт та навігацію користувачів. Важливо, щоб головна сторінка була інтуїтивно зрозумілою, візуально привабливою та функціональною.

Основні кроки створення головної сторінки:

1. Аналіз вимог та планування:

– визначення цілей: головна сторінка повинна забезпечувати доступ до основних функцій платформи, таких як реєстрація, вхід, обмін повідомленнями, перегляд профілю та інші;

– цільова аудиторія: розуміння потреб та очікувань цільової аудиторії - студентів.

2. Розробка макету головної сторінки:

– візуальна концепція: створення ескізів та макетів головної сторінки, включаючи розташування елементів, кольорову палітру та типографіку;

– UI/UX дизайн: використання сучасних принципів UI/UX дизайну для забезпечення зручності користування.

3. Вибір кольорової палітри та типографіки:

– кольори: вибір кольорів, які створюють приємну атмосферу та підвищують читабельність. Основні кольори: синій для спокійного та професійного вигляду, білий для чистоти та простоти, зелений для акцентів;

– шрифти: використання сучасних та легких для читання шрифтів, таких як Roboto або Open Sans. Основний шрифт для заголовків - жирний та виразний, для тексту - звичайний або світлий.

4. Створення HTML/CSS структури:

– HTML: розмітка основних елементів сторінки, таких як заголовки, текстові блоки, кнопки, навігаційне меню;

– CSS: стилізація елементів за допомогою CSS для забезпечення привабливого та узгодженого вигляду.

5. Інтеграція динамічних елементів:

– JavaScript: використання JavaScript для реалізації інтерактивних елементів, таких як слайдери, випадаючі меню, анімації;

– React: застосування React для створення динамічних компонентів та забезпечення швидкого відгуку інтерфейсу.

Візуалізація головної сторінки та ескіз головної сторінки:

Головна сторінка веб-платформи має наступні ключові елементи:

– логотип та навігаційне меню: розташовані у верхній частині сторінки для легкого доступу до основних розділів сайту;

– вступне повідомлення: теплий та привітний текст, який знайомить користувачів з платформою;

– кнопки реєстрації та входу: розташовані на видному місці для швидкого

доступу;

– основні функції та переваги платформи: блоки з короткими описами основних можливостей та переваг;

– контактна інформація та посилання на соціальні мережі: розташовані внизу сторінки (футер).

Код реалізації головної сторінки:

```
            <section id="welcome">
                        <h1>Welcome to the Student Information Exchange Platform</h1>
                        <p>Your secure space for sharing and exchanging information confidentially.</p>
                <a href="#register" class="btn">Register Now</a>
                <a href="#login" class="btn">Login</a>
            </section>
            <section id="features">
                <h2>Our Features</h2>
                <div class="feature">
                    <h3>Confidential Messaging</h3>
                    <p>Communicate securely with fellow students.</p>
                </div>
                <div class="feature">
                    <h3>Profile Management</h3>
                    <p>Manage your profile and privacy settings.</p>
                </div>
                <div class="feature">
                    <h3>Resource Sharing</h3>
                                <p>Share and access study materials and resources.</p>
                </div>
            </section>
        </main>
```

Процес створення головної сторінки веб-платформи для конфіденційного обміну інформацією серед студентів включає в себе аналіз вимог, планування, розробку макету, вибір кольорової палітри та типографіки, створення HTML/CSS структури та інтеграцію динамічних елементів. Важливо забезпечити зручність

користування, привабливий дизайн та доступ до основних функцій платформи.

3.5 Процес створення корзини

Створення корзини для веб-платформи конфіденційного обміну інформацією серед студентів є важливим етапом, оскільки цей функціонал дозволяє користувачам зберігати та керувати документами та іншими елементами, які вони планують обміняти. У цьому розділі розглянемо основні етапи та технології, використані для створення цього функціоналу.

Основні етапи реалізації:

1. Аналіз вимог та визначення функціональності:

– основні функції корзини: додавання елементів до корзини, видалення елементів, перегляд вмісту корзини, оформлення обміну або відправки;

– безпека даних: забезпечення конфіденційності документів та іншої інформації в корзині.

2. Розробка серверної частини:

– API для управління корзиною: створення RESTful API для обробки запитів на додавання, видалення та перегляд елементів у корзині;

– база даних: використання реляційної бази даних для зберігання інформації про корзину кожного користувача.

Приклад реалізації на Node.js з використанням Express та MySQL:

```
// server.js
const express = require('express');
const bodyParser = require('body-parser');
const mysql = require('mysql');
const app = express();
app.use(bodyParser.json());
const db = mysql.createConnection({
    host: 'localhost',
    user: 'root',
    password: 'password',
    database: 'student_exchange'
```

```
});
db.connect(err => {
    if (err) throw err;
    console.log('Database connected!');
});
// Перегляд вмісту корзини
app.get('/cart/:userId', (req, res) => {
    const { userId } = req.params;
    const query = 'SELECT * FROM cart WHERE user_id = ?';
    db.query(query, [userId], (err, results) => {
        if (err) throw err;
        res.send(results);
    });
});
app.listen(3000, () => {
    console.log('Server running on port 3000');
});
```

3. Розробка клієнтської частини:

– інтерфейс для корзини: створення зручного інтерфейсу, який дозволяє користувачам легко додавати, видаляти та переглядати елементи у корзині;

– оформлення обміну: реалізація функціоналу для оформлення обміну або відправки документів з корзини.

Приклад реалізації інтерфейсу на React:

```
    const removeItem = async (itemId) => {
        try {
            await axios.post('/cart/remove', { userId, itemId });
            fetchCartItems();
        } catch (error) {
            console.error('Error removing item', error);
        }
    };
    return (
        <div>
            <h2>Cart</h2>
            <ul>
                {cartItems.map(item => (
```

```
                        <li key={item.item_id}>
                            {item.item_name}
    );
};
export default Cart;
```

4. Тестування та перевірка безпеки

– тестування функціоналу: проведення тестування для перевірки коректної роботи всіх компонентів системи;

– перевірка безпеки: виконання тестів на проникнення та верифікація безпеки даних у корзині.

Процес створення корзини для веб-платформи конфіденційного обміну інформацією серед студентів включає в себе аналіз вимог, розробку серверної та клієнтської частини, а також тестування та перевірку безпеки. Використання сучасних технологій, таких як Node.js, Express, MySQL та React, дозволяє створити ефективну та безпечну систему управління елементами корзини, яка забезпечує зручність та конфіденційність користувачів.

Висновок до розділу 3

У цьому розділі було розглянуто процес розробки та дизайну веб-платформи для конфіденційного обміну інформацією серед студентів. Основні аспекти включали:

– написання Use Case Diagram: представлено взаємодії між акторами та системою, що допомагає зрозуміти функціональні вимоги;

– створення макету веб-платформи: розроблено візуальні компоненти та структуру інтерфейсу, що забезпечує зручність користування;

– реалізація процесу реєстрації: детально описано етапи створення функціоналу реєстрації користувачів, що гарантує безпеку і зручність входу до системи;

– процес створення головної сторінки: показано етапи створення головної сторінки, яка є основним елементом взаємодії користувачів із системою;

– функціонал конфіденційного обміну інформацією: описано реалізацію механізмів, що забезпечують безпечний обмін даними між студентами.

Завдяки комплексному підходу до розробки та впровадження веб-платформи вдалося створити зручний, безпечний та функціональний інструмент для обміну інформацією, що задовольняє потреби студентів.

ВИСНОВОК

У своїй кваліфікаційна роботі я розглянув питання розробки веб-платформи для конфіденційного обміну інформацією серед студентів, приділивши особливу увагу безпеці даних, архітектурі системи та користувацькому досвіду.

Перший розділ аналізував існуючі платформи для обміну інформацією в освітньому середовищі та їх обмеження щодо конфіденційності та безпеки. Було виявлено, що популярні сервіси, такі як Google Drive, Dropbox та OneDrive, мають обмеження у цих аспектах. Спеціалізовані рішення, такі як Signal, Tresorit та SpiderOak, забезпечують вищий рівень безпеки, але мають інші недоліки, такі як складний інтерфейс та висока вартість. Висновок цього розділу вказує на необхідність створення нової платформи, яка б враховувала ці проблеми та забезпечувала високий рівень конфіденційності та безпеки.

Другий розділ присвячений проектуванню веб-платформи, включаючи визначення вимог до системи, архітектуру, вибір технологій, моделювання бази даних та опис користувацького інтерфейсу. Було підкреслено важливість вибору сучасних технологій для захисту даних, таких як методи шифрування, автентифікації та авторизації. Проектування архітектури веб-платформи з акцентом на безпеку, продуктивність створило міцний фундамент для подальшого розроблення продукту, що забезпечить надійність та стабільність системи.

Третій розділ охоплював розробку та дизайн веб-платформи, включаючи створення макету, реалізацію функціоналу реєстрації та конфіденційного обміну інформацією. Було здійснено опис процесу створення веб-платформи, зокрема, розробку діаграм використання (Use Case Diagram) та реалізацію основних функцій системи. Висновки цього розділу підкреслюють важливість забезпечення зручності користування та високого рівня безпеки для кінцевих користувачів, що є ключовим аспектом для успішної реалізації платформи. Таким чином, моя кваліфікаційна робота комплексно розглянула проблему створення безпечної веб-платформи для студентів, забезпечивши її архітектурною стабільністю та зручним інтерфейсом, що сприяє ефективному та безпечному обміну інформацією.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Design smart grid hybrid in faculty of engineering universitas negeri yogyakarta / M. Ali та ін. Journal of physics: conference series. 2021. Т. 2111, № 1. С. 012003. URL: https://doi.org/10.1088/1742-6596/2111/1/012003 (дата звернення: 27.03.2024).

2. Green T., Labrecque J. The UI design process. A guide to UX design and development. Berkeley, CA, 2023. С. 131–156. URL: https://doi.org/10.1007/978-1-4842-9576-2_7 (дата звернення: 27.03.2024).

3. Groothuis D. Who designed the designer?: a dialogue on richard dawkins's the god delusion. Think. 2009. Т. 8, № 21. С. 71–81. URL: https://doi.org/10.1017/s1477175608000407 (дата звернення: 27.03.2024).

4. Лопай С. А., Шипілов А. В. Тестова оболонка для автоматизованого контролю навчальних досягнень. Theory and methods of e-learning. 2014. Т. 3. С. 167–173. URL: https://doi.org/10.55056/e-learn.v3i1.335 (дата звернення: 27.03.2024).

5. Сергеєва Н. Дизайн і філософія: у пошуках точок дотику : thesis. 2021. URL: https://er.knutd.edu.ua/handle/123456789/17916 (дата звернення: 27.03.2024).

6. Чемерис Г., Брянцева Г., Брянцев О. Шляхи вдосконалення дизайн-освіти у контексті стратегії цифрової трансформації освіти і науки україни. Physical and mathematical education. 2022. Т. 32, № 6. С. 49–56. URL: https://doi.org/10.31110/2413-1571-2021-032-6-008 (дата звернення: 27.03.2024).

7. Green T., Labrecque J. The UI design process. A guide to UX design and development. Berkeley, CA, 2023. С. 131–156. URL: https://doi.org/10.1007/978-1-4842-9576-2_7 (дата звернення: 27.03.2024).

8. Sketchbooks F. U. D. UI/UX design sketchbook: dot grid UI and UX notebook, 8. 5 X 11 inches, 110 pages. Independently Published, 2021. 110 с.

9. HTML developer.mozilla.org: вебсайт. URL: https://developer.mozilla.org/ru/docs/Web/HTML. (Дата звертання 23.03.2024).

10. CSS developer.mozilla.org: вебсайт. URL:

https://developer.mozilla.org/ru/docs/Web/CSS. (Дата звертання 27.03.2024).

11. JavaScript developer.mozilla.org: вебсайт. URL: https://developer.mozilla.org/ru/docs/Web/JavaScript. (Дата звертання 27.03.2024).

12. Doherty M. HTML. Computers and Composition. 1996. Т. 13, № 3. С. 340–341. URL: https://doi.org/10.1016/s8755-4615(96)90022-5 (дата звернення: 29.03.2024).

13. Jennifer Niederst Robbins - "Learning Web Design: A Beginner's Guide to HTML, CSS, JavaScript, and Web Graphics" 2018. 808 с. URL:https://www.amazon.com/Learning-Web-Design-Beginners-JavaScript/dp/1449319270. (Дата звернення: 22.04.2024).

14. Douglas Crockford - "JavaScript: The Good Parts" 2008. 176 с. URL:https://www.amazon.com/JavaScript-Good-Parts-Douglas-Crockford/dp/059651774 2. (Дата звернення: 23.04.2024).

15. David Flanagan - "JavaScript: The Definitive Guide: Activate Your Web Pages" 2020. 706 с. URL:https://www.amazon.com/JavaScript-Definitive-Guide-Activate-Guides/dp/0596805527. (Дата звернення: 24.04.2024).

16. Doherty M. HTML. Computers and Composition. 1996. Т. 13, № 3. С. 340–341. URL: https://doi.org/10.1016/s8755-4615(96)90022-5 (дата звернення: 29.04.2024).

17. Balasubramanian V., Ashman H. HTML. ACM SIGWEB Newsletter. 1994. Т. 3, № 3. С. 27. URL: https://doi.org/10.1145/195477.195492 (дата звернення: 03.05.2024).

18. Bello C. The Genital Discharges - At a Glance. Nigerian Medical Practitioner. 2005. Т. 46, № 4. URL: https://doi.org/10.4314/nmp.v46i4.28731 (Дата звернення: 09.05.2024).

19. Jon Duckett - "JavaScript and JQuery: Interactive Front-End Web Development" 2014. 640 с. URL: https://www.amazon.com/JavaScript-JQuery-Interactive-Front-End-Development/dp/1118531647. (Дата звернення: 27.04.2024).

20. Jon Duckett - "HTML and CSS: Design and Build Websites" 2011. 490 с.

URL: https://www.amazon.com/HTML-CSS-Design-Build-Websites/dp/1118008189. (Дата звернення: 10.05.2024).

метадані

Заголовок

Розробка веб-платформи для конфіденційного обміну інформацією серед студентів

Автор

Костенко М. І. Науковий керівник / Експерт

підрозділ

King Danylo University

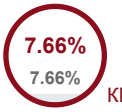
Тривога

У цьому розділі ви знайдете інформацію щодо текстових спотворень. Ці спотворення в тексті можуть говорити про МОЖЛИВІ маніпуляції в тексті. Спотворення в тексті можуть мати навмисний характер, але частіше характер технічних помилок при конвертації документа та його збереженні, тому ми рекомендуємо вам підходити до аналізу цього модуля відповідально. У разі виникнення запитань, просимо звертатися до нашої служби підтримки.

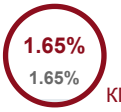
Заміна букв	ß	0
Інтервали	A→	0
Мікропробіли	·	0
Білі знаки	ß	0
Парафрази (SmartMarks)	a	52

Обсяг знайдених подібностей

Коефіцієнт подібності визначає, який відсоток тексту по відношенню до загального обсягу тексту було знайдено в різних джерелах. Зверніть увагу, що високі значення коефіцієнта не автоматично означають плагіат. Звіт має аналізувати компетентна / уповноважена особа.



КП 1



КП 2

25
Довжина фрази для коефіцієнта подібності 2

10249
Кількість слів

83851
Кількість символів

Подібності за списком джерел

Нижче наведений список джерел. В цьому списку є джерела із різних баз даних. Колір тексту означає в якому джерелі він був знайдений. Ці джерела і значення Коефіцієнту Подібності не відображають прямого плагіату. Необхідно відкрити кожне джерело і проаналізувати зміст і правильність оформлення джерела.

10 найдовших фраз — Колір тексту

ПОРЯДКОВИЙ НОМЕР	НАЗВА ТА АДРЕСА ДЖЕРЕЛА URL (НАЗВА БАЗИ)	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)	
1	Розробка інтернет-магазину з реалізації книжкового асортименту [redacted] 5/21/2024 King Danylo University (King Danylo University)	49	0.48 %
2	Розробка інтернет-магазину з реалізації книжкового асортименту [redacted] 5/21/2024 King Danylo University (King Danylo University)	48	0.47 %
3	http://repository.ukd.edu.ua/bitstream/handle/123456789/397/%D0%94%D0%B8%D0%BF%D0%BB%D0%BE%D0%BC%D0%BD%D0%B0_%D0%A2%D0%B0%D1%82%D0%B0%D1%80%D1%87%D1%83%D0%BA.pdf?sequence=1	36	0.35 %