МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет кібербезпеки та програмної інженерії
Кафедра інженерії програмного забезпечення

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач кафедри
Катерина НЕСТЕРЕНКО

“____”_____2024 р.

КВАЛІФІКАЦІЙНА РОБОТА
(ПОЯСНЮВАЛЬНА ЗАПИСКА)

ВИПУСНИКА ОСВІТНЬОГО СТУПЕНЯ
“БАКАЛАВР”

Тема: “Web-застосунок сповіщення студентів про ключові події навчального процесу”

Виконавець: Бежнар Валерій Ігорович

Керівник: к.т.н. Талалаєв Володимир Опанасович

Нормоконтролер: Варнавський В’ячеслав Володимирович

Київ 2024

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет кібербезпеки та програмної інженерії
Кафедра інженерії програмного забезпечення
Освітній ступінь бакалавр
Спеціальність 121 «Інженерія програмного забезпечення»
Освітньо-професійна програма «Програмне забезпечення систем»

ЗАТВЕРДЖУЮ
Завідувач кафедри
Катерина НЕСТЕРЕНКО
"__" _____ 2024 р.

ЗАВДАННЯ
на виконання кваліфікаційної роботи студента
Бежнара Валерія Ігоровича

1. Тема проекту: «Web-застосунок сповіщення студентів про ключові події навчального процесу»
затверджена наказом ректора від 8.12.2023 р. № 2483/ст

2. Термін виконання роботи: з 3.01.2024 р. до 29.02.2024 р.

3. Вихідні данні до проекту: програмний продукт розроблений у вигляді веб-застосунку, який сповіщатиме студентів про ключові події навчального процесу.

4. Зміст пояснювальної записки:
1. Загальні відомості про веб-застосунки.
2. Аналіз існуючих програмних продуктів.
3. Функціональні та нефункціональні вимоги до програмного забезпечення, вибір засобів реалізації.
4. Реалізація веб-застосунку.

5. Перелік обов'язкових слайдів презентації:
1. Актуальність веб-застосунків в сучасному світі.
2. Аналіз існуючих програмних продуктів для керування навчального процесу.
3. Вибір мови програмування для створення програмного застосунку.
4. Вибір бази даних для реалізації веб-застосунку.
5. Реалізація Web-застосунку сповіщення студентів про ключові події навчального процесу.

6. Календарний план-графік

№ пор	Завдання	Термін виконання	Відмітка про виконання
1.	Складання та затвердження графіку роботи дипломного проектування Написання 1 розділу, представлення керівнику	03.01.24 - 09.01.24	
2.	Попередній друк 1 розділу та допоміжних сторінок (чорновик) - титульної, завдання, графіка, реферат, список скорочень, зміст, вступ, список джерел, 1-й нормо-контроль.	10.01.24 - 17.01.24	
3.	Написання 2 розділу, представлення керівнику	18.01.24 – 22.01.24	
4.	Написання 3 розділу, представлення керівнику	23.01.24 – 26.01.24	
5.	Написання 4 розділу, представлення керівнику	27.01.24 – 31.01.24	
6.	Загальне редагування та друк пояснювальної записки, графічного матеріалу	01.02.24 – 04.02.24	
7.	Проходження нормо-контролю, перевірка на антиплагіат, перепліт пояснювальної записки.	05.02.24 – 11.02.24	
8.	Розробка тексту доповіді. Оформлення графічного матеріалу для презентації	12.02.24 – 15.02.24	
9.	Отримання відгуку керівника, рецензії.	16.02.24 – 17.02.24	
10.	Підготовка матеріалів для передачі секретарю ДЕК (ПЗ, CD-R з електронними копіями ПЗ, презентації, відгук керівника, рецензія) в папці	18.02.24 – 29.02.24	

7. Дата видачі завдання 3.01.2024

Керівник: к.т.н. Володимир ТАЛАЛАЄВ

Завдання прийняв до виконання: Валерій БЕЖНАР

Дата

РЕФЕРАТ

Пояснювальна записка до кваліфікаційної роботи «Web-застосунок сповіщення студентів про ключові події навчального процесу»: 53 с., 22 рис., 10 інформаційних джерел.

СПОВІЩЕННЯ СТУДЕНТІВ, НАВЧАЛЬНИЙ ПРОЦЕС

Об’єкт розробки – веб-застосунок, що допомагає студентам отримувати інформацію про події, які відбуваються в університеті, розклад, і непередбачувані зміни у навчальному процесі.

Мета роботи – проаналізувати існуючі аналоги веб-застосунків та додатків про сповіщення студентів про ключові події навчального процесу, та створити веб-застосунок для покращення комунікації між викладачем та студентом, своєчасним отриманням інформації та для ефективної організації навчання.

ABSTRACT

Explanatory note to the thesis " Web application for communication of students about key events of the educational process": 53 p., 24 Fig., 10 informations sources.

NOTICE TO STUDENTS, LEARNING PROCESS

Property development – web application that helps students get information about events that take place at the university, the schedule, and unexpected changes in the educational process.

Purpose – to analyze existing analogues of web applications and applications for notifying students of key events of the educational process, and to create a web application to improve communication between the teacher and the student, timely receipt of information and effective organization of training.

ЗМІСТ

ПЕРЕЛІК ПРИЙНЯТИХ СКОРОЧЕНЬ

ІС – Інформаційні системи

ПЗ – Програмне забезпечення

БД – Бази даних

JS – JavaScript

HTML - HyperText Markup Language

CSS - Cascading Style Sheets

SM - SchoolManagement

ВСТУП

У сучасному світі, де технології та інновації стрімко трансформують усі сфери життя, освіта не відстає від цього процесу. Зростаюча залежність від інформаційних технологій вимагає постійного вдосконалення систем управління навчанням та комунікації між студентами та адміністрацією закладу.

У цьому контексті великого значення набула розробка та впровадження веб-додатку, який має на меті інформувати студентів про важливі події в освітньому процесі. Цей проект є відповіддю на зростаючу потребу в покращенні комунікації та забезпеченні легкого доступу до інформації, важливої для студентської спільноти.

Метою цієї дипломної роботи є детальний аналіз, розробка та впровадження Web-застосунку для інформування студентів про важливі події в навчальному процесі. У навчальному середовищі, де доступність та своєчасність інформації відіграє ключову роль у формуванні траєкторії успішного навчання, цей проект слугує інноваційним інструментом, який має на меті полегшити навчання та надати студентам необхідну інформацію у найбільш зручний та доступний спосіб.

У цьому контексті в даній роботі розглядаються основні аспекти розробки веб-додатків, визначаються його основні функціональні, архітектурні та технічні вимоги, а також оцінюється його вплив на ефективність навчального процесу та взаємодію між студентами та навчальним закладом.

Важливими складовими цієї роботи є основні етапи розробки та впровадження веб-додатків та їх вплив на підвищення якості навчального процесу та забезпечення зручності для студентів. Результати цього дослідження можуть мати значний вплив на організацію навчального процесу та підвищити ефективність взаємодії між студентами та навчальними закладами.

Дана робота відкриває нові перспективи для розвитку та застосування інформаційних технологій у сфері освіти та вказує на важливість їхнього внеску у підвищення якості освіти.

РОЗДІЛ 1.

ЗАГАЛЬНІ ВІДОМОСТІ ПРО ВЕБ-ЗАСТОСУНКИ

1.1. Історія створення веб-застосунків

Історія веб-додатків сягає корінням у перші дні Всесвітньої павутини. На початку 1990-х років Тім Бернерс-Лі винайшов Всесвітню павутину, спочатку створюючи статичні веб-сторінки, які переважно складалися з тексту та гіперпосилань, використовуючи HTML (HyperText Markup Language). Однак із впровадженням таких технологій, як сценарії CGI (Common Gateway Interface), веб-сторінки почали ставати більш інтерактивними, дозволяючи генерувати динамічний вміст.

З наближенням кінця 1990-х і початку 2000-х років веб-розробка стрімко розвивалася. Такі технології, як JavaScript і CSS (каскадні таблиці стилів), набули популярності, дозволяючи розробникам створювати більш інтерактивні та візуально привабливі веб-сайти. Крім того, з'явилися серверні мови сценаріїв, такі як PHP, ASP (Active Server Pages) і JSP (JavaServer Pages), які сприяли створенню динамічного вмісту та взаємодії з базами даних. У середині 2000-х відбулися значні зміни з появою AJAX (асинхронний JavaScript і XML), що зробило революцію в веб-розробці, забезпечивши асинхронний обмін даними між браузером і сервером без перезавантаження всієї сторінки.

Ця епоха також ознаменувала розвиток Web 2.0, який характеризується контентом, створеним користувачами, співпрацею та інтерактивністю. Платформи соціальних мереж, такі як Facebook, YouTube і Twitter, набули величезної популярності в цей період.

З поширенням смартфонів і планшетів наприкінці 2000-х і на початку 2010-х років фокус змістився в бік мобільних веб-додатків. Розробники почали використовувати технології адаптивного дизайну, щоб забезпечити оптимальну взаємодію з користувачами на різних пристроях і розмірах екрану. Такі фреймворки, як jQuery Mobile і адаптивні CSS-фреймворки, стали важливими для створення мобільних веб-додатків.

В останні роки односторінкові програми (SPA) стали популярною архітектурою для веб-програм. SPA динамічно оновлюють одну HTML-сторінку, коли користувачі взаємодіють із програмою, пропонуючи більш плавну та бездоганну роботу користувача. Фреймворки/бібліотеки Frontend JavaScript, такі як AngularJS, React і Vue.js, отримали широке поширення, надаючи розробникам потужні інструменти для створення інтерактивних і динамічних інтерфейсів користувача. Крім того, такі серверні технології, як Node.js, Python Django, Ruby on Rails і ASP.NET Core, стають все більш популярними для створення масштабованих і ефективних серверних програм веб-додатків.

Загалом історія веб-додатків відображає безперервну еволюцію, спричинену прогресом технологій, зміною очікувань користувачів і зростанням складності онлайн-сервісів.

1.2. Рентабельність веб-застосунків

Рентабельність веб-додатків залежить від їх здатності генерувати прибуток порівняно з інвестиціями, зробленими в їх розробку, підтримку та маркетинг. Кілька факторів сприяють цій прибутковості.

Впершу чергу це стратегія монетизації, такі як реклама, підписка або прямі продажі, відіграють ключову роль у визначенні потоку доходів веб-

програми. Ефективність цих стратегій залежить від таких факторів, як залучення користувачів, демографічні показники цільової аудиторії та ринковий попит.

По-друге, це – розмір і рівень залученості бази користувачів, які значно впливають на отримання прибутку. Збільшення кількості користувачів і збільшення залученості означає більше можливостей для монетизації, зокрема через рекламу та продажі.

Важливим пунктом є також витрати на рекламу та конкуренція на рекламному ринку, які впливають на рентабельність, оскільки вищі витрати можуть знизити прибуток. Отже, веб-програми повинні знайти баланс між залученням рекламодавців і збереженням прибутку.

Витрати на розробку та обслуговування є критично важливими міркуваннями. Зменшення цих витрат може прискорити шлях до прибутковості, дозволяючи веб-програмі швидше окупати інвестиції.

Задоволення потреб цільової аудиторії та ефективне задоволення потреб ринку є важливими для стабільної прибутковості. Веб-програма, яка резонує з користувачами та заповнює прогалини на ринку, має більше шансів на успіх.

Вкрай важливу роль відіграють ефективні маркетингові та рекламні заходи у залученні користувачів і збільшенні прибутку. Добре реалізована маркетингова стратегія може збільшити залучення користувачів і впізнаваність бренду, сприяючи прибутковості веб-додатку.

Таким чином, прибутковість веб-додатків залежить від комбінації факторів, включаючи стратегії монетизації, залучення користувачів, витрати на розробку та обслуговування, конкуренцію, ринковий попит і ефективність маркетингу. Ретельно орієнтуючись на ці фактори, веб-програми можуть підвищити свою прибутковість і довгострокову життєздатність.

1.3. Аналіз предметної області веб-застосунків

Аналіз предметної області веб-застосунків є ключовим етапом для розуміння контексту, в якому буде функціонувати веб-застосунок. Цей аналіз допомагає визначити вимоги, цільову аудиторію, конкурентні переваги та потенційні ризики. Далі йде опис кількох основних аспектів аналізу предметної області веб-застосунків:

1. Цільова аудиторія: Розуміння, хто буде користуватися веб-застосунком. Це включає в себе визначення демографічних характеристик, інтересів, потреб та проблем, які вирішує веб-застосунок для своїх користувачів.

2. Конкурентний аналіз: Дослідження існуючих веб-застосунків у цій предметній області. Це допомагає з'ясувати, які функції та можливості вже присутні на ринку, а також визначити прогалини, які можна заповнити.

3. Технологічні вимоги: Аналіз технічних аспектів, таких як мови програмування, фреймворки, бази даних тощо, щоб визначити оптимальні технології для реалізації веб-застосунку.

4. Бізнес-модель: Визначення того, яким чином веб-застосунок буде генерувати прибуток. Це може бути реклама, платні підписки, продаж продуктів або послуг, спонсорські внески тощо.

5. Потенційні ризики та відповідність: Аналіз потенційних загроз безпеці, приватності та регуляторних вимог, що можуть вплинути на веб-застосунок.

6. Потреби користувачів: Збір відгуків та зворотного зв'язку від користувачів щодо їхніх потреб і вимог до веб-застосунку. Це допомагає вирішити, які функції та покращення можуть бути включені в майбутні версії веб-застосунку.

7. Тестування і зворотний зв'язок: Проведення тестування веб-застосунку для перевірки функціональності, якості та ефективності. Збір зворотного зв'язку від користувачів після випуску для подальшого вдосконалення.

Цей аналіз допомагає створити базу для успішного розвитку веб-застосунку, забезпечуючи врахування потреб користувачів, конкурентного середовища та технічних обмежень.

1.4. Положення про веб-застосунки та принципи їх роботи

Положення про веб-застосунки та принципи їх роботи визначають основні принципи створення та функціонування веб-додатків. Ось деякі основні положення та принципи:

1. Клієнт-серверна архітектура: Веб-застосунки базуються на клієнт-серверній моделі, де клієнти (браузери користувачів) взаємодіють з сервером через HTTP або HTTPS протоколи.

2. HTTP/HTTPS протоколи: Взаємодія між клієнтом і сервером відбувається за допомогою HTTP або HTTPS протоколів, які забезпечують передачу даних через мережу Інтернет.

3. Веб-браузери: Клієнти веб-застосунків — це веб-браузери, які відображають вміст веб-застосунку та взаємодіють з ним за допомогою HTML, CSS та JavaScript.

4. HTML, CSS та JavaScript: HTML (Hypertext Markup Language), CSS (Cascading Style Sheets) та JavaScript використовуються для створення структури, оформлення та функціональності веб-застосунків у браузері.

5. Сесії та кукі: Для збереження стану користувача та ідентифікації веб-сесій використовуються куки (cookies) або сеанси (sessions), що зберігаються на боці клієнта або сервера відповідно.

6. Відкритість і доступність: Веб-застосунки можуть бути доступні з будь-якого пристрою, що має доступ до Інтернету, без необхідності встановлення спеціального програмного забезпечення.

7. Масштабованість: Веб-застосунки повинні бути здатні масштабуватися для обробки великої кількості запитів від користувачів, забезпечуючи при цьому високу продуктивність та доступність.

8. Безпека: Забезпечення захисту веб-застосунків від різноманітних загроз, таких як атаки злому, витоки даних та інші види кіберзлочинності, є однією з найважливіших задач при їх розробці та експлуатації.

Ці принципи визначають основу для розробки та експлуатації веб-застосунків, забезпечуючи їхню ефективність, надійність та безпеку.

1.4.1. Архітектура веб-застосунків

Архітектура веб-застосунків - це організація компонентів та взаємозв'язків між ними для створення веб-застосунку. Вона визначає, як будуть виконуватися різні функції, ролі та взаємодія між складовими частинами системи. Ось деякі з найпоширеніших архітектур веб-застосунків:

1.Однорівнева (монолітна) архітектура:

a) У цій архітектурі всі компоненти, включаючи інтерфейс, бізнес-логіку та базу даних, розташовані в одному застосунку.

b) Це простий підхід для розробки та використання, але може бути складним для масштабування та розвитку великих проектів.

2. Клієнт-серверна архітектура:

a) У цій архітектурі клієнт (веб-браузер) та сервер (веб-сервер) взаємодіють за допомогою HTTP або HTTPS протоколів.

b) Веб-сервер відповідає за обробку запитів від клієнтів та надання відповідей, які можуть бути створені з використанням даних з бази даних або інших джерел.

3. Клієнт-серверна архітектура з розділенням на рівні (RRIA):

a) Ця архітектура розширює клієнт-серверну модель, розділяючи її на рівні (presentation, application, data).

b) Presentation layer (рівень презентації) відповідає за відображення інформації для користувача.

c) Application layer (рівень додатку) містить бізнес-логіку та логіку взаємодії з базою даних.

d) Data layer (рівень даних) забезпечує доступ до даних та взаємодію з базою даних.

4. Мікросервісна архітектура:

a) У цій архітектурі великі додатки розбиваються на невеликі незалежні мікросервіси, які взаємодіють один з одним за допомогою мережі.

b) Кожен мікросервіс відповідає за виконання конкретного функціоналу та може бути розвинутий, масштабований та розгорнутий незалежно.

5. Серверна архітектура з використанням безстатевих API (stateless):

a) У цій архітектурі кожен запит від клієнта до сервера повністю незалежний, а сервер не зберігає стан клієнта між запитами.

b) Це полегшує масштабування та розділення завдань між серверами, але може потребувати додаткової роботи для збереження стану клієнта, якщо це необхідно.

Зрозуміння принципів та особливостей різних архітектурних підходів дозволяє розробникам створювати надійні, швидкі та функціональні веб-застосунки, які відповідають потребам сучасного інтернет-середовища. Враховуючи важливість архітектури, команди розробників можуть забезпечити успіх своїх проектів та задоволення користувачів.

1.4.2. Порівняння різних архітектур веб-застосунків

Порівняння різних архітектур веб-застосунків допомагає краще зрозуміти їх переваги та недоліки у відповідності до конкретних вимог проекту. Ось деякі аспекти порівняння різних архітектур:

1. Монолітна архітектура:

Переваги:

a) Простота розробки та впровадження, оскільки всі компоненти розташовані в одному застосунку.

b) Можливість швидко реагувати на зміни, оскільки все знаходиться в одному місці.

Недоліки:

a) Складність масштабування, оскільки потрібно масштабувати весь монолітний застосунок.

b) Високий ризик виникнення точки відмови, оскільки помилка в одному компоненті може вплинути на весь застосунок.

2. Клієнт-серверна архітектура:

Переваги:

a) Чітке розділення обов'язків між клієнтом та сервером.

b) Можливість масштабування, оскільки клієнти можуть бути розділені від сервера.

Недоліки:

a) Збільшена складність розробки через необхідність підтримувати взаємодію між клієнтом та сервером.

b) Можливість виникнення проблем з безпекою через відкритий обмін даними через мережу.

3. Мікросервісна архітектура:

Переваги:

a) Гнучкість та швидкість розробки, оскільки різні функціональність можуть бути розроблені та підтримані незалежно.

b) Легка масштабованість, оскільки кожен мікросервіс може бути масштабований окремо.

Недоліки:

a) Підвищена складність управління та моніторингу через велику кількість окремих сервісів.

b) Додаткові витрати на управління та координацію між мікросервісами.

4. Серверна архітектура з використанням безстатевих API (stateless):

Переваги:

a) Простота масштабування через відсутність потреби зберігати стан на сервері між запитами.

b) Покращена надійність через можливість розподілення навантаження між безстатевими серверами.

Недоліки:

a) Потреба у додатковій роботі для збереження стану клієнта, якщо це необхідно.

b) Обмеження в розвитку додаткових функціональностей, які потребують збереження стану.

При виборі архітектури веб-застосунку, важливо враховувати специфіку проекту, його масштаби, потреби у масштабованості, надійності, швидкості розробки та інші фактори.

1.4.3. Переваги та недоліки веб-застосунків

Веб-програми пропонують численні переваги, які сприяють їх широкій популярності та прийняттю. Ось деякі основні переваги:

1. Доступність: веб-програми доступні з будь-якого пристрою з підключенням до Інтернету та веб-браузером. Користувачі можуть отримати до них доступ із настільних ПК, ноутбуків, смартфонів і планшетів, забезпечуючи гнучкість і зручність.

2. Сумісність із різними платформами: на відміну від власних програм, веб-програми не прив’язані до певної операційної системи чи платформи. Вони бездоганно працюють на різних платформах, включаючи Windows, macOS, iOS, Android та інші.

3. Простота оновлення: веб-програми можна оновлювати централізовано на сервері, не вимагаючи від користувачів завантажувати чи встановлювати оновлення вручну. Це гарантує, що всі користувачі мають доступ до останньої версії програми одночасно.

4. Економічна ефективність: розробка та підтримка веб-додатків часто є економічно ефективнішою, ніж створення власних додатків для кількох платформ. Завдяки єдиній кодовій базі розробники можуть охопити ширшу аудиторію без необхідності розробки для конкретної платформи.

5. Масштабованість: веб-програми можна легко масштабувати відповідно до зростаючої кількості користувачів і зростаючого попиту. Хмарна інфраструктура дозволяє безперешкодно масштабувати без значних початкових інвестицій в апаратне забезпечення чи інфраструктуру.

6. Встановлення не потрібне: користувачі можуть негайно почати використовувати веб-програми без необхідності завантажувати чи інсталювати щось локально. Це знижує бар'єр входу та усуває проблеми сумісності, пов'язані з різними пристроями та операційними системами.

7. Миттєві оновлення та розгортання: зміни та оновлення веб-додатків можна розгортати миттєво, забезпечуючи швидку ітерацію та постійне вдосконалення. Така гнучкість дозволяє розробникам швидко реагувати на відгуки користувачів і вимоги ринку.

8. Видимість у пошуковій системі: веб-програми за своєю суттю легше виявляються пошуковими системами, що полегшує користувачам пошук і доступ до них. Це може призвести до збільшення трафіку та можливостей залучення користувачів.

9. Інтеграція зі сторонніми службами: веб-програми можна легко інтегрувати з різними сторонніми службами та API, розширюючи їх функціональні можливості та надаючи користувачам більш багатий досвід.

10. Аналітика та відстеження: веб-додатки можуть використовувати інструменти аналітики для відстеження поведінки користувачів, моніторингу показників продуктивності та збору інформації для оптимізації та прийняття рішень.

Загалом веб-додатки пропонують універсальне та ефективне рішення для надання програмних послуг користувачам на різних пристроях і платформах. Їхня доступність, економічність, масштабованість і легкість оновлення роблять їх привабливим вибором як для компаній, так і для розробників.

Недоліки веб-застосунків:

Хоча веб-програми пропонують багато переваг, вони також мають деякі недоліки, які важливо враховувати. Ось деякі з недоліків веб-додатків:

1. Залежність від підключення до Інтернету: для належної роботи веб-додатків потрібне стабільне підключення до Інтернету. Користувачі в регіонах зі слабким підключенням до Інтернету можуть сповільнювати роботу або мати обмежений доступ до функцій.

2. Обмежені функціональні можливості в автономному режимі: на відміну від власних програм, веб-програми часто мають обмежену функціональність або взагалі не працюють у режимі офлайн. Користувачі можуть не мати доступу до певних функцій або вмісту без підключення до Інтернету.

3. Обмеження продуктивності: веб-програми можуть страждати від проблем із продуктивністю, особливо під час роботи з великими обсягами даних або складною функціональністю. Такі фактори, як затримка мережі, сумісність браузера та час відповіді сервера, можуть впливати на продуктивність.

4. Проблеми безпеки: веб-додатки більш уразливі до загроз безпеці, таких як міжсайтовий сценарій (XSS), впровадження SQL і витік даних. Забезпечення надійних заходів безпеки, таких як шифрування, автентифікація та регулярні перевірки безпеки, має важливе значення для пом’якшення цих ризиків.

5. Проблеми сумісності веб-переглядача: веб-програми можуть не працювати однаково в різних веб-браузерах і версіях. Розробникам часто доводиться проводити ретельне тестування та впроваджувати обхідні шляхи, щоб забезпечити сумісність із такими популярними браузерами, як Chrome, Firefox, Safari та Edge.

6. Обмежений доступ до апаратного забезпечення пристрою: на відміну від власних програм, веб-програми мають обмежений доступ до апаратного забезпечення пристрою, наприклад камер, датчиків і пам’яті. Це може

обмежити функціональність і роботу певних функцій, наприклад доступ до камери пристрою або GPS.

7. Менш інтерактивний досвід користувача: хоча сучасні веб-технології покращили інтерактивність веб-додатків, вони все ще можуть відставати від рідних додатків з точки зору взаємодії з користувачем і швидкості реагування. Складні анімації, жести та переходи можуть працювати не так гладко, як у рідних середовищах.

8. Занепокоєння конфіденційністю даних. Зберігання конфіденційних даних на віддалених серверах викликає занепокоєння щодо конфіденційності, особливо щодо відповідності нормам захисту даних, таким як GDPR. Користувачі можуть вагатися щодо надання особистої інформації чи конфіденційних даних веб-додаткам через ризики конфіденційності та безпеки.

9. Обмежений доступ до функцій пристрою: веб-додатки можуть мати обмежений доступ до специфічних для пристрою функцій і API, що обмежує їх здатність використовувати розширені можливості, такі як push-повідомлення, фонова обробка та апаратне прискорення.

10. Залежність від сторонніх служб: веб-програми часто покладаються на сторонні служби та API для таких функцій, як автентифікація, обробка платежів і зберігання даних. Залежність від цих служб створює потенційні точки збою та може ускладнити роботу з розробки та обслуговування.

Незважаючи на ці недоліки, веб-програми залишаються популярним вибором для надання програмних рішень завдяки їх доступності, крос-платформній сумісності та економічній ефективності. Пом’якшення цих недоліків вимагає ретельного розгляду найкращих практик проектування, розробки та безпеки.

Висновок

Отож, предметна область веб-застосунків охоплює широкий спектр технічних, дизайнерських, інформаційних та аналітичних аспектів. Для успішної розробки, впровадження та управління веб-застосунками необхідно ретельно вивчити і аналізувати технології веб-розробки, дизайн інтерфейсів, безпеку, оптимізацію для пошукових систем та мобільних пристроїв, аналітику користувачів, соціальну інтеграцію, електронну комерцію та адаптивний дизайн. Ретельний аналіз цих аспектів допомагає забезпечити якість, ефективність та конкурентоспроможність веб-застосунків на ринку.

РОЗДІЛ 2.
АНАЛІЗ ІСНУЮЧИХ ПРОГРАМНИХ ПРОДУКТІВ

2.1. Аналіз програмного продукту «Moodle»

Опис продукту: Moodle - це система управління навчанням з відкритим вихідним кодом, призначена для того, щоб допомагати педагогам створювати онлайн-курси та керувати віртуальними навчальними середовищами. Розроблена Мартіном Дугіамасом, Moodle означає "Modular Object-Oriented Dynamic Learning Environment" (Модульне об'єктно-орієнтоване динамічне навчальне середовище). Ця платформа дозволяє педагогам створювати та поширювати вміст, спілкуватися зі студентами та керувати різними аспектами навчального процесу.

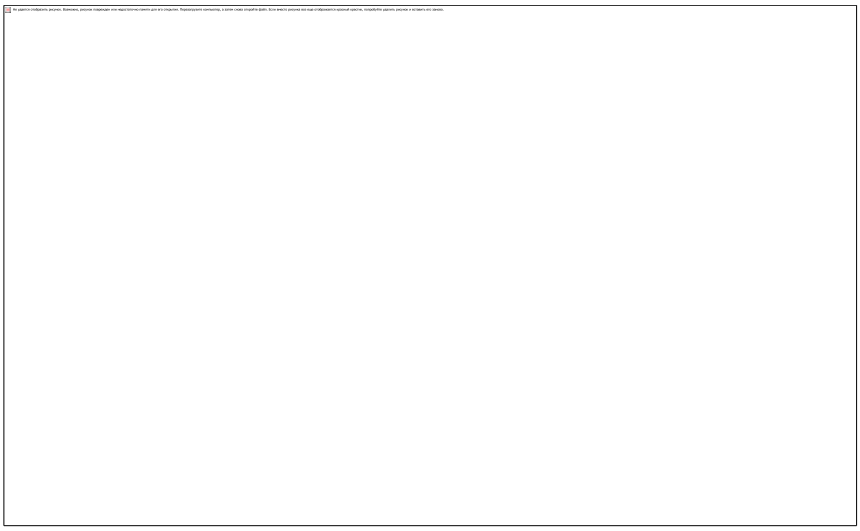

Рис. 2.1. Moodle

Переваги:

1. Moodle базується на відкритому вихідному коді, що означає велику гнучкість та можливість модифікації під потреби конкретної установи чи вчителя.

2. Moodle дозволяє додавати різноманітні плагіни та розширення для розширення функціональності відповідно до конкретних вимог користувача.

3. Moodle надає різні інструменти для співпраці та взаємодії, такі як форуми, чати та інші соціальні елементи.

4. Платформа дозволяє викладачам створювати різноманітні завдання, тести та інші інструменти для оцінювання знань студентів.

5. Moodle підтримує багатомовність, що робить його придатним для використання в різномовному середовищі.

6. Moodle надає вчителям інструменти для організації та управління навчальними курсами.

7. Деякі функції Moodle можуть бути використані оф-лайн, що важливо в умовах обмеженого Інтернет-з'єднання.

Недоліки:

1. Для новачків використання та конфігурування Moodle може бути складним завданням.

2. Для повноцінної роботи системи може знадобитися технічна підтримка або спеціалізовані навчання.

3. Зовнішній вигляд та інтерфейс Moodle можуть виглядати менш сучасно порівняно з іншими платформами.

4. Moodle може вимагати значних ресурсів для ефективної роботи, що може становити проблему для менших установ або серверів.

5. Неправильне або несвоєчасне проведення апдейтів може впливати на безпеку та функціональність.

6. Вартість розгортання та підтримки може бути значною, особливо для

користувачів, які не мають власних IT-ресурсів.

7. Деякі аспекти відкритого вихідного коду можуть бути обмежені для користувачів, які не мають досвіду у роботі з програмним забезпеченням з відкритим вихідним кодом.

2.2. Аналіз програмного продукту «Google Classroom»

Опис продукту: Google Classroom є популярним та ефективним інструментом для організації навчального процесу. Він приваблює простотою використання та інтеграцією з іншими Google-сервісами. Однак, для більш складних завдань або великих установ, може знадобитися розгляд інших Learning Management Systems з розширеним функціоналом.

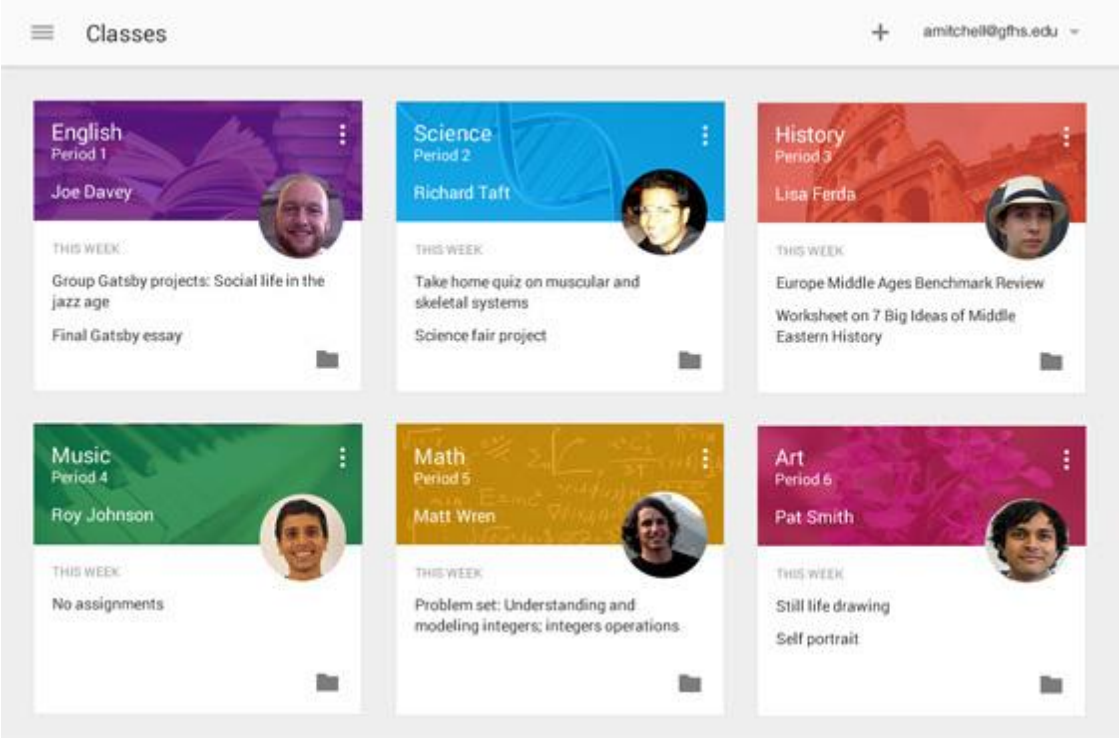

Рис. 2.2. Google Classroom

Переваги:

1. Інтеграція з Google Workspace: Classroom пов'язаний з іншими інструментами Google, такими як Gmail, Google Drive і Google Calendar, що спрощує роботу з документами та комунікацію.

2. Простий та зрозумілий інтерфейс: Інтуїтивно зрозумілий для вчителів і студентів, що робить його дружелюбним для користувача.

3. Створення завдань та матеріалів: Вчителі можуть легко створювати завдання, надсилати матеріали для читання і взаємодіяти зі студентами в одному місці.

4. Гнучкість для завдань: Дозволяє створювати різні типи завдань, використовуючи текст, фотографії, відео тощо.

5. Чат і сповіщення: Можливість комунікації через чат та надсилання сповіщень, що спрощує обмін інформацією.

6. Групи: Дозволяє створювати групи для спільної роботи або обговорення.

Недоліки:

1. Обмежена функціональність: Деякі користувачі можуть зазначати обмежені можливості для складніших завдань порівняно з іншими LMS.

2. Не достатньо варіативності: Інтерфейс та можливості можуть здатися занадто стандартними для тих, хто шукає більші можливості налаштувань.

3. Відсутність розширених інструментів для оцінювання: Деякі вчителі можуть знайти обмежені функції оцінювання, зокрема в області зворотного зв'язку або аналітики.

4. Відсутність реального часу: Комунікація може не бути в реальному часі, що робить його менш ефективним для ситуацій, де потрібна негайна відповідь.

5. Система прав доступу: Для більшої безпеки може бути бажано розширення системи управління правами для уникнення неправомірного доступу.

2.3. Аналіз програмного продукту «Microsoft Teams»

Опис продукту: Microsoft Teams - це комплексний програмний продукт для

комунікації та співпраці в робочих групах та командах. Він є частиною екосистеми Microsoft 365 та призначений для організації онлайн-зустрічей, спільної роботи над проектами, обміну файлами та спілкування в командному середовищі.

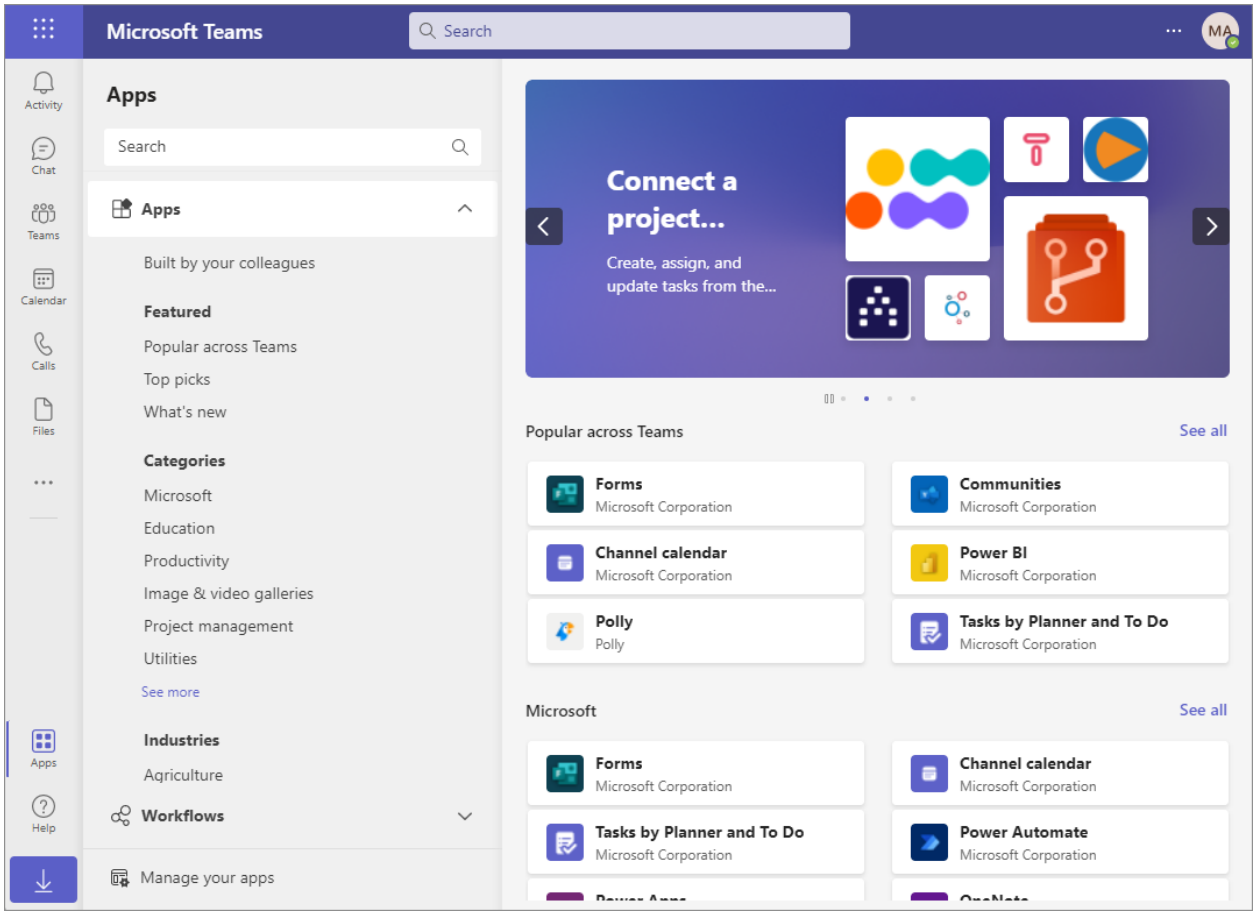

Рис. 2.3. Microsoft Teams

Переваги:

1. Teams надає можливість проводити відеоконференції та аудіозв'язок з учасниками в режимі реального часу.

2. Інтеграція з іншими сервісами Microsoft, такими як Outlook, для планування зустрічей та іншими опціями.

3. Зручні інструменти для спільної роботи над документами, таблицями, та презентаціями, включаючи інтеграцію з Microsoft Office.

4. Всі зміни вносяться в реальному часі, що полегшує колективну

роботу.

5. Можливість створювати приватні та групові чати для обговорення завдань та обміну інформацією.

6. Збереження історії чатів для зручного відстеження.

7. Широкий вибір додатків та сервісів, які можна інтегрувати в Teams, таких як плагіни для задач, календарі, CRM-системи тощо.

8. Розширена інтеграція з Microsoft 365 та іншими інструментами Microsoft.

Недоліки:

1. Інтерфейс Teams може здаватися складним для новачків, особливо якщо вони не мають попереднього досвіду з продуктами Microsoft. Велика кількість функцій та вкладок може призвести до плутанини.

2. Teams може вимагати значних ресурсів комп'ютера та мережі, особливо при великій кількості одночасних користувачів або використанні відеоконференцій високої якості.

3. Незважаючи на наявність засобів безпеки, Teams має обмежені можливості в порівнянні з іншими продуктами. Для підприємств із високими вимогами до безпеки це може бути недостатнім.

4. Інтеграція Teams з іншими платформами часто потребує додаткових налаштувань, і у деяких випадках вона може бути менш зручною порівняно з іншими інструментами.

5. Деякі розширені функції, такі як аналітика та розширена система управління, доступні тільки у платних планах Microsoft 365.

6. Використання Teams повністю залежить від доступу до Інтернету, що може становити проблему в умовах обмеженої або невпевненої мережі.

7. Користувачі можуть відчувати надмірний обсяг нотифікацій та повідомлень, що може впливати на продуктивність та концентрацію.

2.4. Аналіз програмного продукту «Remind»

Опис продукту: "Remind" - це платформа для комунікації між вчителями, учнями і їхніми батьками. Вона спрямована на полегшення обміну інформацією та сповіщень між освітніми установами і учасниками навчального процесу.

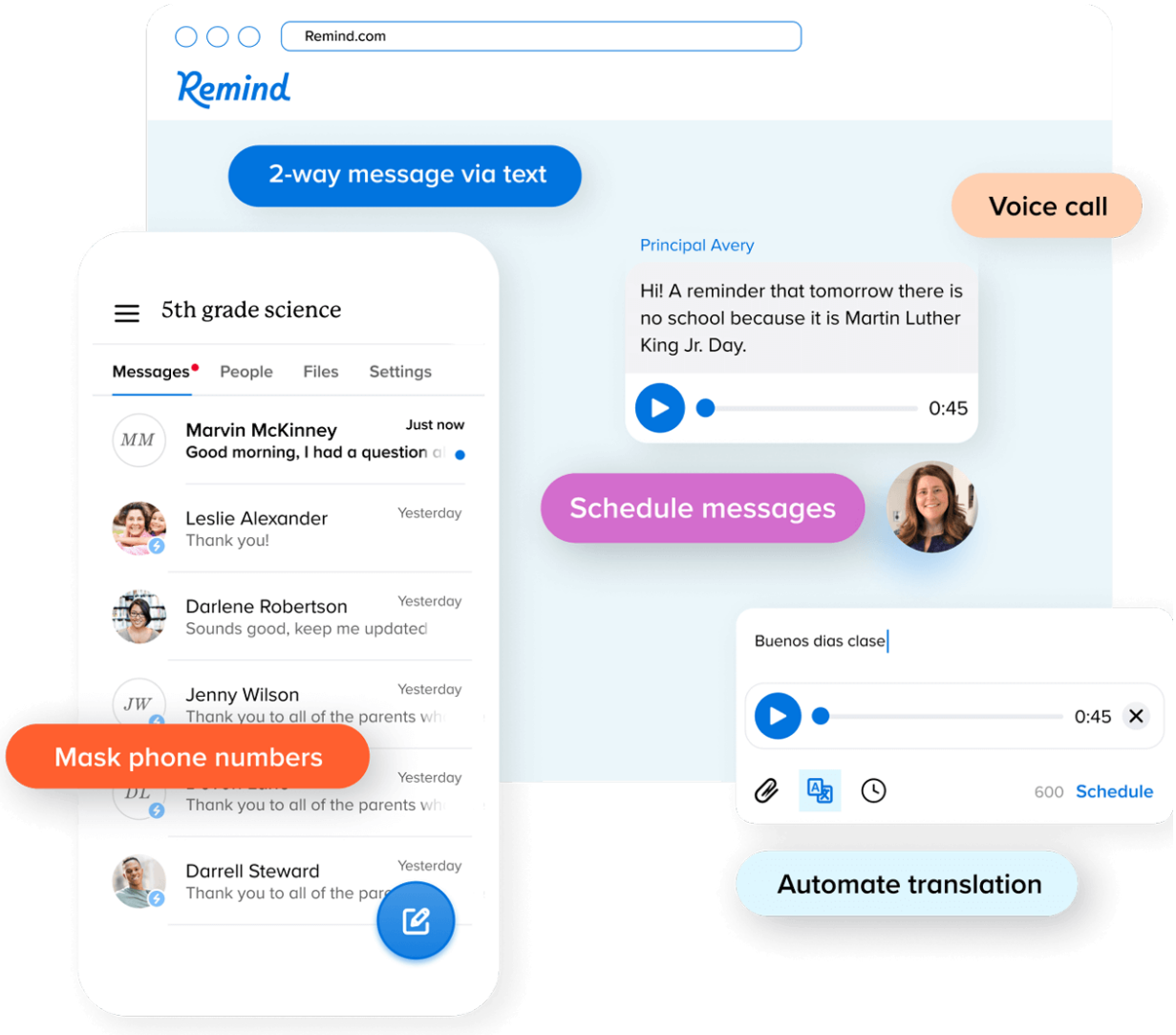

Рис. 2.4. Remind

Переваги:

1. Remind надає зручний та простий інтерфейс для обміну повідомленнями та сповіщеннями між вчителями, учнями і батьками.

2. Можливість використовувати текстові повідомлення, фотографії та аудіозаписи для комунікації.

3. Вчителі можуть створювати групові чати для роботи з учнями та їхніми батьками.

4. Можливість розміщення оголошень та нагадувань для всієї групи.

5. Вчителі можуть надсилати завдання, графіки та інші матеріали для вивчення безпосередньо через платформу.

6. Система нагадувань сприяє вчасному виконанню завдань та плануванню подій.

7. Наявність мобільних додатків для платформ iOS та Android для зручного доступу до Remind з будь-якого пристрою.

8. Основний функціонал Remind доступний безкоштовно, що робить його доступним для широкого кола користувачів.

9. Remind підтримує декілька мов, що робить його придатним для використання в різномовному середовищі.

Недоліки:

1. Remind може бути сприйнятий деякими користувачами як менш розширений у порівнянні з іншими платформами для спілкування в освітньому середовищі.

2. Платформа потребує доступу до Інтернету для ефективного функціонування. В умовах обмеженого Інтернет-з'єднання це може бути проблемою.

3. Деякі розширені функції та інструменти доступні тільки у платних версіях Remind, що може обмежити можливості користувачів без платної підписки.

4. У порівнянні з іншими платформами, Remind не має вбудованих інструментів для відеоконференцій, що може бути недоліком у контексті дистанційного навчання.

5. Деякі користувачі можуть відчувати обмеження в розміщенні медіа-файлів та обміні зображеннями або відео через Remind.

6. Хоча Remind надає інструменти для збереження конфіденційності

даних, важливо враховувати можливі ризики щодо приватності у зв'язку з обробкою особистої інформації.

7. Деякі користувачі можуть виявити обмежені можливості інтеграції Remind з іншими сервісами або платформами, які вони використовують.

Висновок

Аналіз програмних продуктів веб-застосунків в освіті виявляє широкий спектр інструментів та ресурсів, що сприяють покращенню навчального процесу. Від платформ для дистанційного навчання до інтерактивних навчальних засобів та відеоконференційних платформ, ці програмні продукти надають вчителям та учням можливість ефективно взаємодіяти та здобувати знання у сучасному цифровому середовищі.

Важливо також враховувати аспекти безпеки даних та підтримки технічної інфраструктури для забезпечення надійності та зручності використання цих засобів. Аналіз дозволяє вибрати оптимальні програмні продукти, які відповідають потребам та цілям освітнього процесу, забезпечуючи ефективну та продуктивну навчальну діяльність.

РОЗДІЛ 3.

ФУНКЦІОНАЛЬНІ ТА НЕФУНКЦІОНАЛЬНІ ВИМОГИ ДО ПЗ, ВИБІР ЗАСОБІВ РЕАЛІЗАЦІЇ

3.1. Функціональні вимоги

Функціональні вимоги для web-застосунку сповіщення студентів про ключові події навчального процесу можуть включати такі пункти:

1. Реєстрація користувачів:

    a) Можливість реєстрації як студента з обов'язковим введенням особистої інформації (ім'я, прізвище, електронна пошта, номер студентського квитка тощо).

    b) Авторизація студентів з використанням електронної пошти та пароля.

2. Профіль користувача:

    a) Змога зміни особистої інформації в профілі.

    b) Перегляд інформації про навчання, такої як розклад занять, групи, розклад іспитів тощо.

3.Сповіщення про події:

    a) Автоматичне надсилання сповіщень про ключові події навчального процесу (наприклад, зміни у розкладі, наближення дат іспитів тощо).

b) Змога вибору студентом типу сповіщень та часу отримання (наприклад, SMS, електронна пошта, push-сповіщення).

4. Календар подій:

a) Перегляд календаря з навчальними подіями та іншими важливими датами.

b) Можливість додавання власних подій до календаря.

5. Інтеграція з системами управління навчальним процесом:

a) Взаємодія з базами даних університету для отримання актуальної інформації про події.

6. Пошук і фільтрація:

a) Зручний пошук і фільтрація подій за різними критеріями (наприклад, за датою, типом події, назвою предмету тощо).

7. Підтримка мобільних пристроїв:

a) Адаптація web-застосунку до різних типів пристроїв (смартфони, планшети, комп'ютери) для зручного використання на різних пристроях та в різний час.

8. Аналітика та звітність:

a) Збір та аналіз даних про використання застосунку (наприклад, кількість користувачів, популярність функцій тощо).

b) Генерація звітів для адміністраторів.

9. Захист і безпека:

a) Захист особистої інформації користувачів.

b) Забезпечення безпеки передачі даних між користувачем та сервером.

3.2. Нефункціональні вимоги

Нефункціональні вимоги для web-застосунку сповіщення студентів про ключові події навчального процесу описують якість, ефективність та інші характеристики системи.

Веб-додаток повинен відповідати наступним нефункціональним вимогам:

1. Безпека:
    a) Застосунок повинен мати механізми автентифікації та авторизації для захисту особистої інформації студентів.
    b) Повинна бути застосована шифрування для передачі та зберігання конфіденційної інформації.
2. Доступність:
    a) Система повинна бути доступною для використання відповідно до вимог часу, наприклад, працювати безперервно або мати мінімальний час відновлення після відмови.
3. Відмовостійкість:
    a) Застосунок повинен відновлюватися після відмови системи або мережі, а також забезпечувати цілісність даних.
4. Швидкодія:
    a) Застосунок повинен відповідати на запити користувачів швидко та ефективно, мінімізуючи затримки.
5. Масштабованість:
    a) Система повинна бути здатною масштабуватися для обробки збільшеного обсягу користувацьких запитів та даних.
6. Сумісність:
    a) Застосунок повинен бути сумісним з різними веб-браузерами та мобільними пристроями.
7. Удосконалення користувацького досвіду:
    a) Забезпечення інтуїтивного та зручного інтерфейсу користувача.

b) Оптимізація швидкості завантаження сторінок та відповідей сервера.

8. Локалізація та міжнароднізація:

a) Здатність адаптуватися до мовних та культурних варіацій для задоволення потреб користувачів з різних регіонів.

9. Документація та підтримка:

a) Наявність докладної документації щодо використання та налаштування системи.

b) Надання технічної підтримки для вирішення проблем користувачів.

3.3. Засоби для реалізації програмного забезпечення

3.3.1. HTML

HTML (Hypertext Markup Language) є стандартизованим мовою розмітки, яка використовується для створення структурированих документів на веб-сайтах. Вона є основою для будь-якої веб-сторінки і визначає її структуру, зовнішній вигляд та спосіб відображення контенту для користувачів.

Основними елементами HTML є теги, які визначають тип контенту та його оформлення. Теги включають назву елемента, угорі з обмежуючими символами "<" та ">", які показують початок та кінець елемента. Наприклад, тег "<p>" використовується для визначення абзаців тексту, тег "<img>" - для вставки зображень, а тег "<a>" - для створення посилань.

HTML також використовує атрибути, які додають додаткову інформацію до елементів. Наприклад, атрибут "src" вказує шлях до зображення для тега "<img>", атрибут "href" вказує адресу посилання для тега "<a>".

Окрім основних тегів, HTML також має спеціальні елементи для розмітки заголовків, таблиць, списків, форм та інших складних структур. Ці

елементи дозволяють розміщати і організовувати контент на сторінці так, щоб він був зручним для сприйняття користувачами.

Крім того, HTML є основою для взаємодії з іншими технологіями веб-розробки, такими як CSS (Cascading Style Sheets) для оформлення сторінок та JavaScript для створення динамічного контенту та інтерактивності.

У дипломній роботі ви зможете детально побачити історію роботи з HTML, її основні концепції та синтаксис, а також практичний приклад створення веб-сторінок з використанням HTML.

3.3.2. CSS

CSS (Cascading Style Sheets) - це мова стилів, яка використовується для оформлення веб-сторінок і надання їм зовнішнього вигляду. CSS дозволяє розробникам визначати стилізацію елементів HTML, таких як кольори, шрифти, розміри, відступи, вирівнювання та багато іншого.

Основна ідея CSS полягає в тому, щоб визначити стилі окремо від вмісту HTML, що дозволяє розділити структуру сторінки від її представлення. Це робить код більш організованим, легким для змін та підтримки.

CSS застосовується за допомогою правил, які складаються з селекторів та оголошень стилів. Селектор визначає, які елементи HTML будуть застосовувати стилі, а оголошення стилів встановлюють конкретні властивості та їх значення.

CSS також підтримує концепцію каскаду, що означає, що стилі можуть успадковуватися від батьківських елементів, а також перевизначатися чи змінюватися в залежності від порядку, в якому вони вказані в коді.

За допомогою CSS веб-розробники можуть створювати привабливий та професійний дизайн для веб-сторінок, що полегшує їх використання та покращує користувацький досвід.

3.3.3. JavaScript

JavaScript є високорівневою, інтерпретованою мовою програмування, яка використовується для створення динамічного контенту та інтерактивності на веб-сайтах. Вона дозволяє розробникам створювати скрипти, які можуть контролювати поведінку веб-сторінок, реагувати на події користувача, маніпулювати DOM (Document Object Model) та взаємодіяти з сервером.

Основні характеристики JavaScript включають:

Динамічність: JavaScript дозволяє додавати та змінювати елементи та властивості на сторінці під час її завантаження або після нього. Це дозволяє створювати динамічні ефекти, анімацію та змінювати вміст сторінок без перезавантаження.

Події: JavaScript може реагувати на події, такі як натискання кнопок, наведення миші, введення тексту тощо. Це дозволяє створювати інтерактивні елементи, такі як форми, слайдери та меню.

Маніпуляція DOM: JavaScript дає можливість змінювати структуру, стиль та вміст веб-сторінок, використовуючи DOM API. Це дозволяє створювати динамічний контент та взаємодіяти з елементами на сторінці.

Асинхронність: JavaScript підтримує асинхронне програмування, що дозволяє виконувати операції, які можуть займати час (наприклад, завантаження даних з сервера), без блокування виконання інших операцій.

Крос-платформенність: JavaScript може виконуватися на будь-якому веб-браузері, що підтримує стандарт ECMAScript, що робить його універсальним інструментом для розробки веб-додатків.

JavaScript зазвичай використовується в поєднанні з HTML та CSS для створення повноцінних веб-додатків та інтерактивних інтерфейсів користувача. Вона є однією з ключових технологій для розробки веб-сайтів та додатків і забезпечує широкі можливості для реалізації різноманітних функціональностей.

3.3.4. Python

Python - це високорівнева мова програмування загального призначення, яка відома своєю простотою та читабельністю синтаксису. Це інтерпретована мова, що означає, що програми Python виконуються рядок за рядком, без попередньої компіляції в машинний код. Розроблена в 1991 році Гвідо ван Россумом, Python став однією з найпопулярніших мов програмування завдяки своїй простоті в освоєнні та використанні.

Python має широкий спектр застосувань, від веб-розробки до аналізу даних, наукових обчислень, штучного інтелекту та інших областей. Він підтримує різноманітні парадигми програмування, включаючи процедурне, об'єктно-орієнтоване та функціональне програмування.

Однією з ключових особливостей Python є його екосистема бібліотек та фреймворків. Наприклад, бібліотеки, такі як NumPy, Pandas, та Matplotlib, забезпечують потужні засоби для роботи з даними та візуалізації. Django та Flask - це популярні фреймворки для веб-розробки, а TensorFlow та PyTorch - це фреймворки для роботи з машинним навчанням та глибоким навчанням.

Ще однією перевагою Python є його велика та активна спільнота розробників. Ця спільнота підтримує відкритий доступ до безлічі корисних ресурсів, включаючи документацію, уроки, та форуми обговорень. Крім того, Python має велику кількість сторонніх пакетів та інструментів, які розширюють його можливості та полегшують роботу розробників.

3.3.5. Django

Django - це відкритий веб-фреймворк, написаний на мові програмування Python, який дозволяє швидко створювати потужні та масштабовані веб-додатки. Розроблений в 2003 році компанією Lawrence Journal-World, Django набув широкої популярності завдяки своїй простоті, ефективності та розширюваності.

Основні принципи Django включають модульність, повторне використання коду та швидкість розробки. Фреймворк надає

стандартизовану структуру проекту, вбудовану адміністративну панель, систему автентифікації та авторизації, механізми маршрутизації URL, підтримку баз даних та багато інших корисних функцій.

Однією з ключових особливостей Django є його "принцип батарейок включено", що означає, що багато функціональності, яка часто використовується в веб-розробці, вже вбудовано у фреймворк або доступно через сторонні розширення (пакети). Це спрощує розробку, оскільки розробникам не потрібно постійно винаходити велосипеди і можуть сконцентруватися на розвитку своїх додатків.

Крім того, Django має активну спільноту розробників, що підтримує фреймворк, надаючи документацію, пакети, уроки та підтримку через форуми та спеціалізовані конференції. Це робить Django одним із переважних виборів для розробників, які шукають ефективний та потужний інструмент для створення веб-додатків.

Висновок

У цьому розділі було визначено потрібні функціональні та нефункціональні вимоги, а також, проаналізовано та обрано прикладні інструменти для розробки продуктивного веб-додатку для сповіщення студентів про ключові події навчального процесу. Обрані інструменти налічують відповідну функціональність для виконання поставленого завдання та враховує специфікацію проекту. Вони також забезпечують функціональність і надійність системи, необхідні для ефективного виконання завдання.

РОЗДІЛ 4.

РЕАЛІЗАЦІЯ WEB-ЗАСТОСУНКУ

4.1. Формування БД

Таблиця 1 – «Реєстрація»(school_management_app_customuser) містить унікальний ідентифікатор(id), пароль(password), останній час входу(last_login), перевірку чи це адміністратор (is_superuser), користувацького імені(username), імені користувача(first_name), прізвища(last_name), пошти користувача(email), перевірку чи це працівник університету(is_staff), дати створення(date_joined) та типу користувача(user_type).

Рис. 4.1. Таблиця «Реєстрація»

Таблиця 2 – «Студент»(school_management_app_students) складається з наступних рядків – унікальний ідентифікатор(id), стать(gender), фотографія профілю(profile_pic), адреси(address), з дати створення і зміни профілю(created_at, updated_at), з даними про курс(course_id_id) та з сесійного року(session_year_id_id).

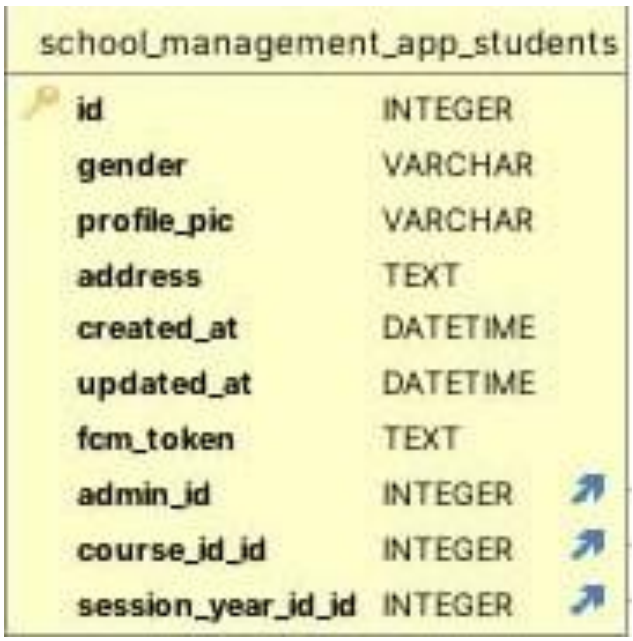

Рис. 4.2. Таблиця «Студент»

Таблиця 3 – «Онлайн клас»(school_management_app_onlineclassroom) складається з наступних пунктів – унікальний ідентифікатор(id), назви класу(room_name), паролю класу(room_pwd), чи є він активним і чи відбуваються заняття(is_active), дати створення(created_on), сесійного

року(session_years_id), ким був створений(started_by_id), користувачів приєднаних до цього класу(subject_id).

Рис. 4.3. Таблиця «Онлайн клас»

Таблиця 4. – «Курс»(school_management_app_courses) включає в себе такі рядки – унікальний ідентифікатор(id), назва курсу(course_name), дати створення(created_at) та його оновлень(updated_at).

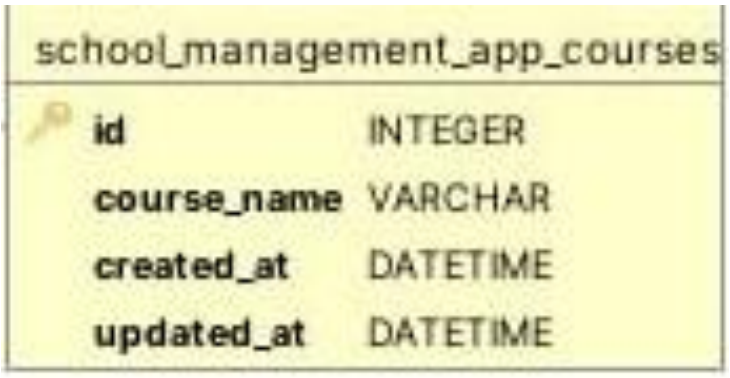

Рис. 4.4. Таблиця «Курс»

Таблиця 5. – «Оцінка студента»(school_management_app_studentresult) у цій таблиці наявні наступні пункти – унікальний ідентифікатор(id), оцінка за

екзамен(subject_exam_marks), оцінка за завдання(subject_assignment_marks), дата оцінювання(created_at), зміна оцінки(updated_at), ідентифікатор студента(student_id_id), ідентифікатор викладача(subject_id_id).

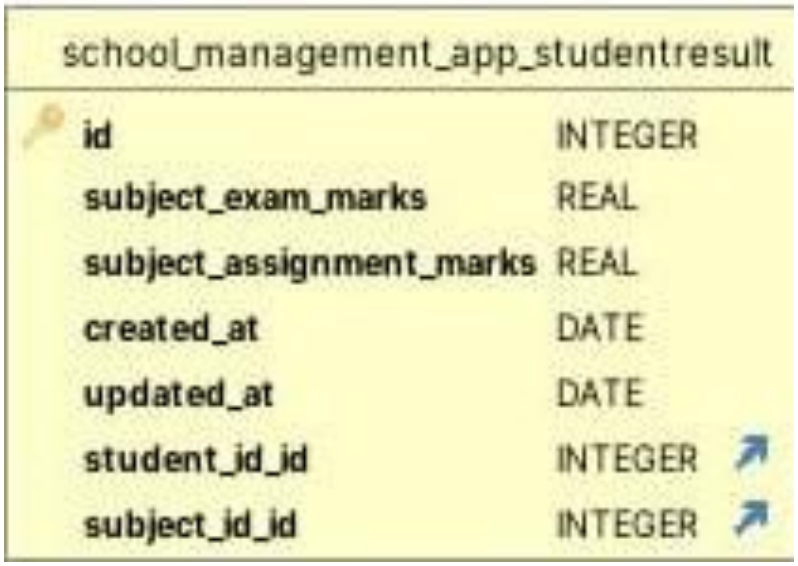

Рис. 4.5. Таблиця «Оцінка студента»

Таблиця 6. – «Сповіщення для студентів від викладача»(school_management_app_tnews) складається з таких полів, як – унікальний ідентифікатор(id), текст сповіщення(ntext), заголовок сповіщення(ntitle), зображення для сповіщення(pic), час та дата створення сповіщення(ndate).

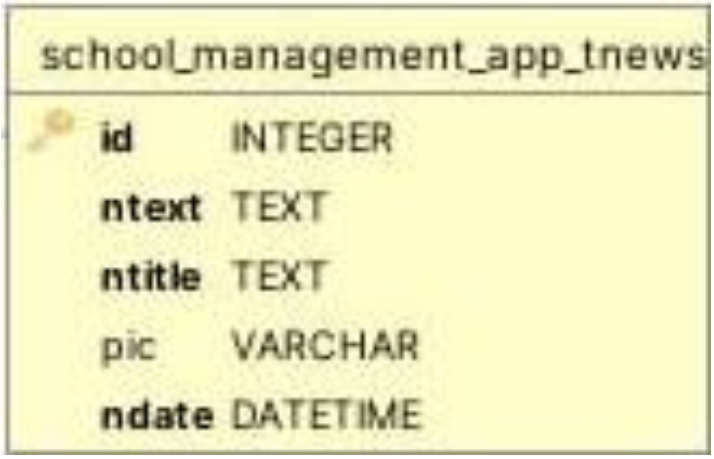

Рис. 4.6. Таблиця «Сповіщення для студентів від викладача»

Таблиця 7. – «Коментарі від викладача під сповіщенням»(school_management_app_tcomment) включає в себе – унікальний ідентифікатор(id), коментар(body), кількість символів в коментарів(count), ідентифікатор викладача(staff_id_id), сповіщення під яким був поставлений коментар(TNews_id), час коментування(created_on), ідентифікатор коментаря, написаного до даного коментаря(reply_id).

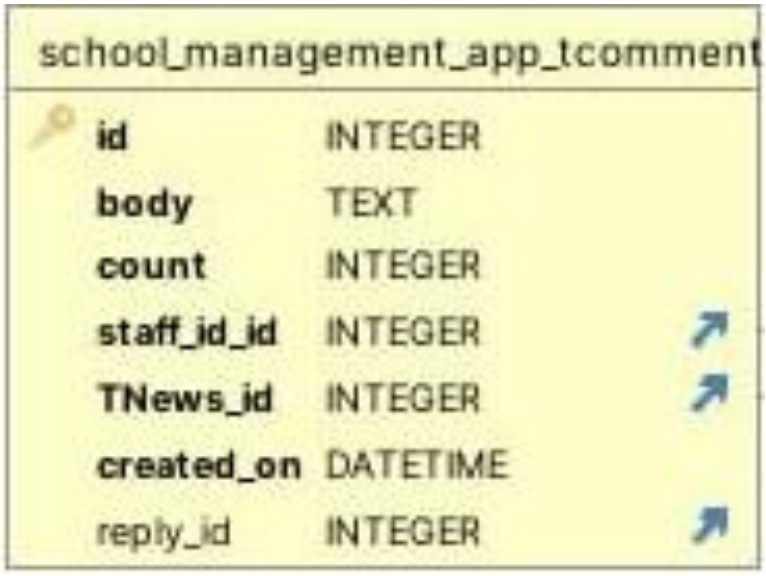

Рис. 4.7. Таблиця «Коментарі викладача під сповіщенням»

Таблиця 8. – «Сповіщення від студента для викладача»(school_management_app_snews) містить в собі унікальний ідентифікатор(id), текст сповіщення(ntext), заголовок сповіщення(ntitle), час створення сповіщення(ndate), зображення для сповіщення(pic).

Рис. 4.8. Таблиця «Сповіщення від студента для викладача»

Таблиця 9. – «Коментарі студентів під сповіщеннями»(school_management_app_scomment) включає в себе унікальний ідентифікатор(id), текст коментаря(body), час та дату коментування(created_on), кількість символів(count), сповіщення під яким було поставлено коментар(PNews_id), здатність відповісти на інший коментар(reply_id).

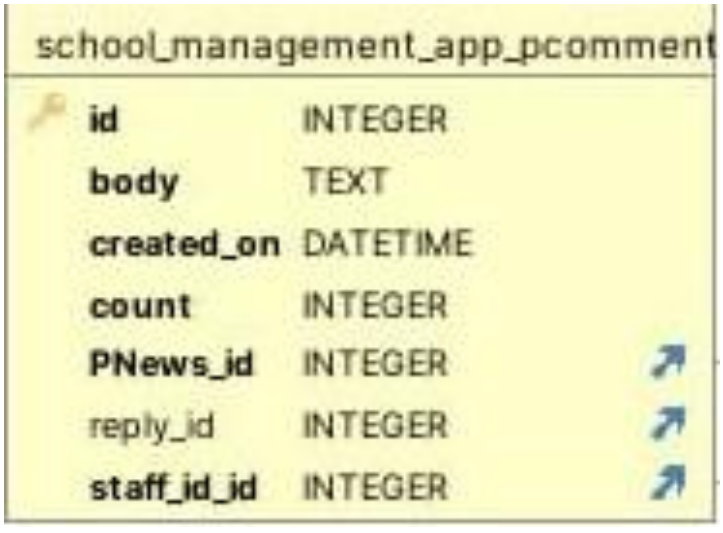

Рис. 4.9. Таблиця «Коментарі студентів під сповіщеннями»

Для більшої читабельності ER-моделі, її зображення було поділене на 3 частини, які представлені нижче:

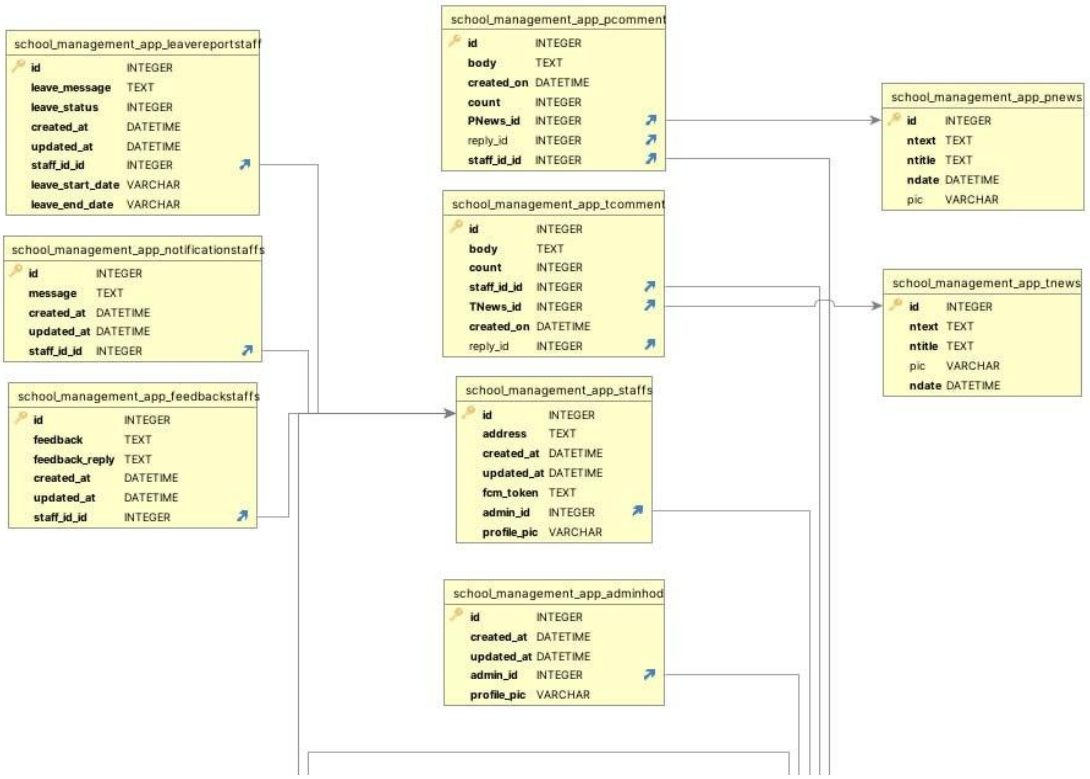

Рис. 4.10. ER-модель(частина 1)

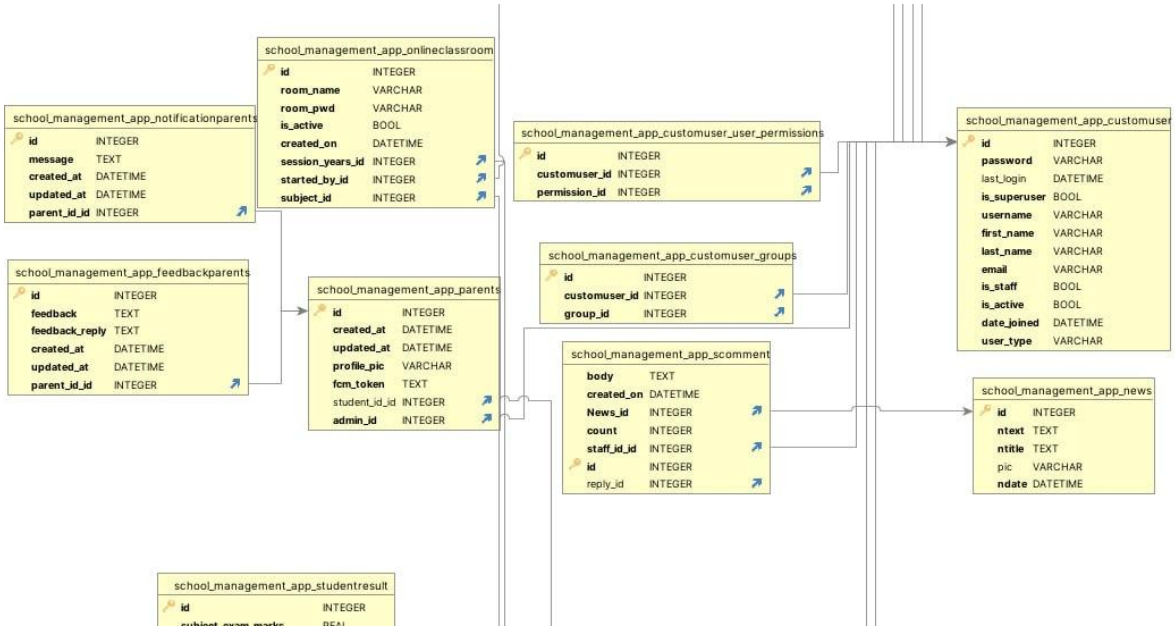

Рис. 4.11. ER-модель(частина 2)

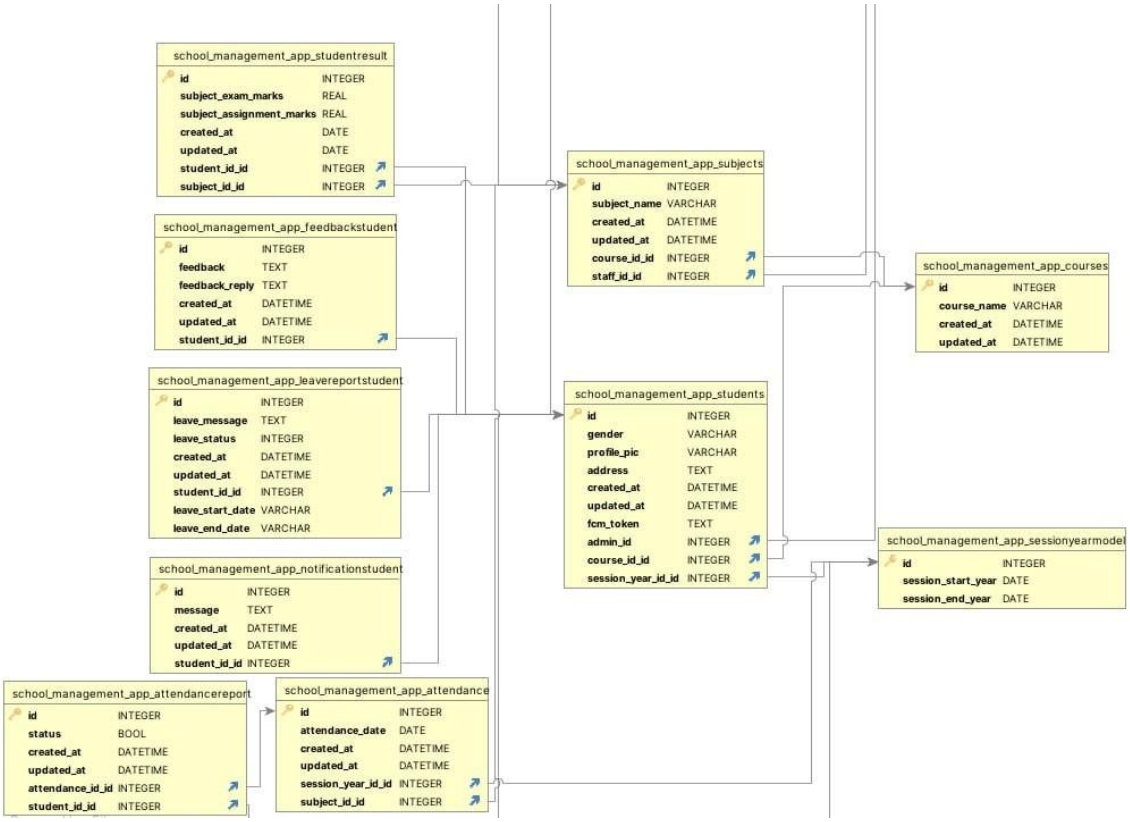

Рис. 4.12. ER-модель(частина 3)

4.2 Представлення Web-застосунку

Веб-застосунок сповіщення студентів про ключові події навчального процесу отримав назву SM(SchoolManagement), дальше буде показано вхід у застосунок, який складається з адреси електронної пошти та паролю, а також можливості відновлення паролю, у випадку, якщо користувач його втратив. Також є функція «Темний режим», який дозволяє краще працювати з веб-застосунком у темний період часу, тим самим менше подразнюючи людське око.

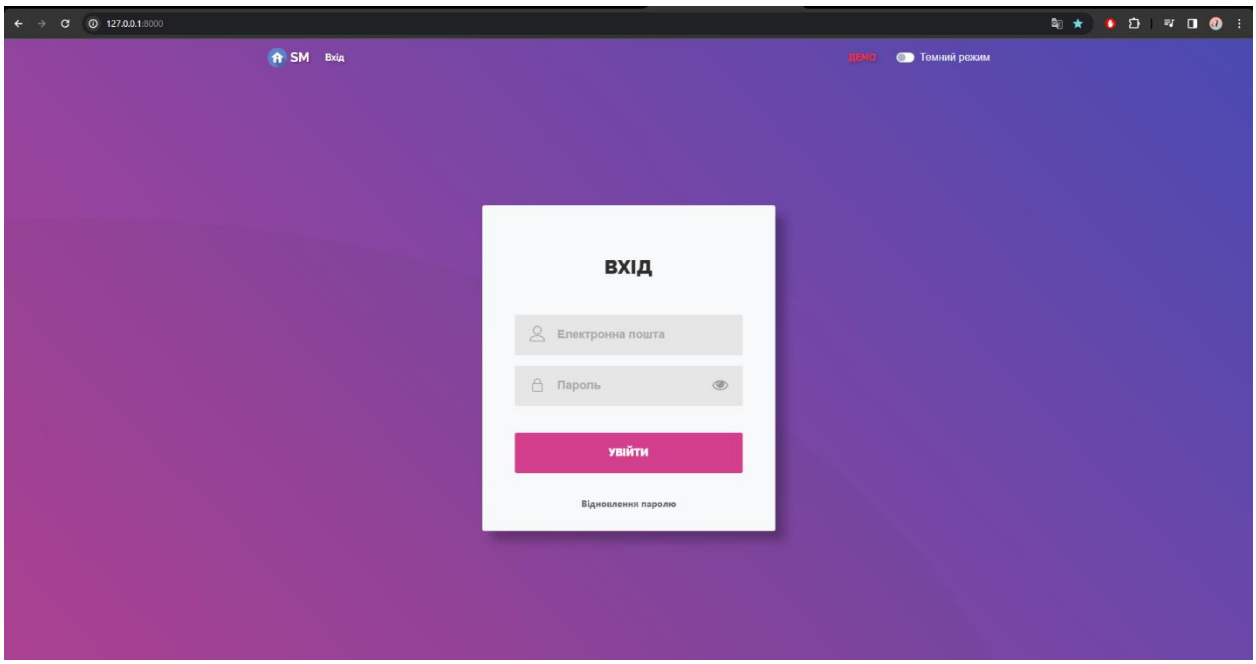

Рис. 4.13. Авторизація у веб-застосунок

Тепер ми можемо побачити головну сторінку даного застосунку з сторони викладача, де у нього зображено «Статистику», про кількість студентів, загальну кількість відвідувань, скільки уроків він повинен провести, дані про навчальну діяльність, а також графіки та діаграми відвідувань.

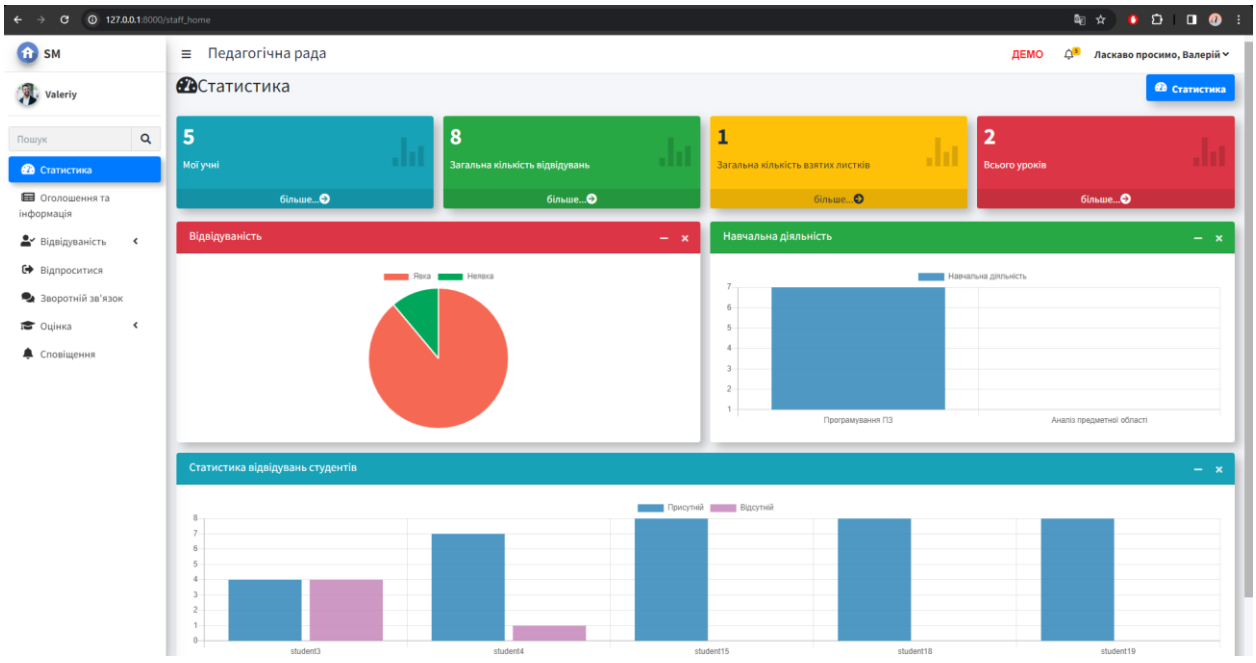

Рис. 4.14. Головна сторінка для викладача

А зараз ми підійшли до основної частини моєї дипломної роботи та цього веб-застосунку, це сповіщення студентів про ключові події та новини. Для прикладу було взято декілька статей з інтернету, щоб продемонструвати роботу цього функціоналу.

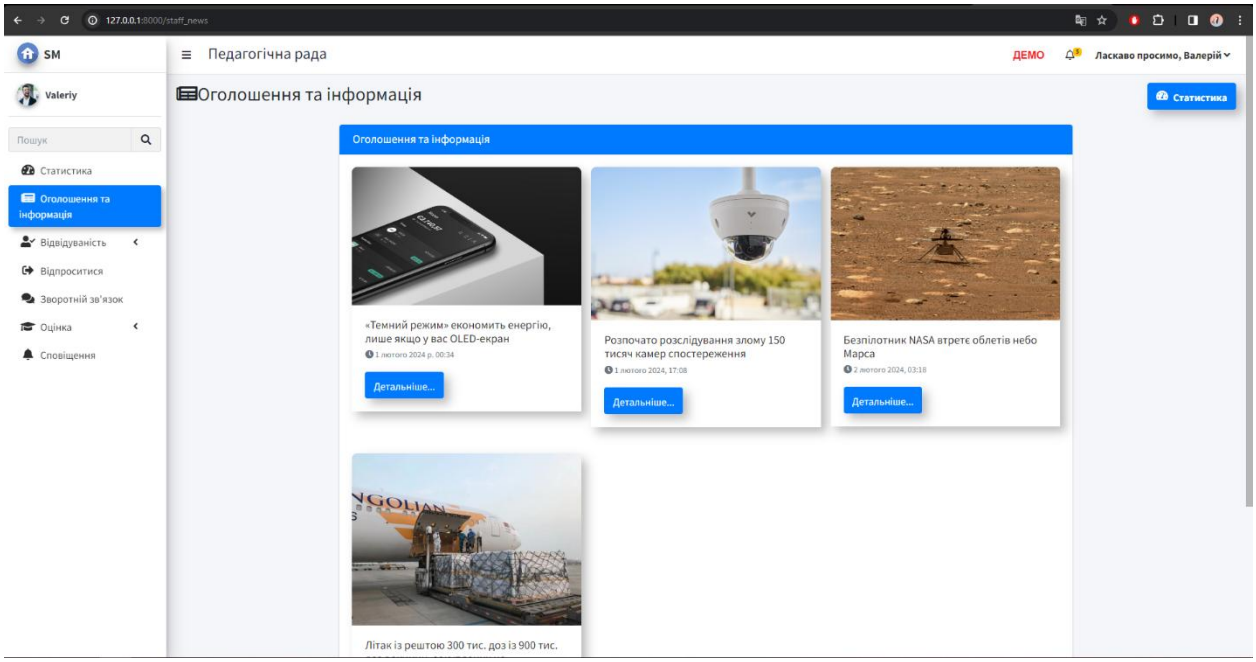

Рис. 4.15. Сповіщення студентів про події

Якщо ми натиснемо кнопку «Детальніше», ми матимемо змогу побачити всю інформацію про цю новину чи сповіщення. Також студенти мають можливість коментувати ці новини, відповідати на повідомлення інших користувачів і взаємодіяти з ними.

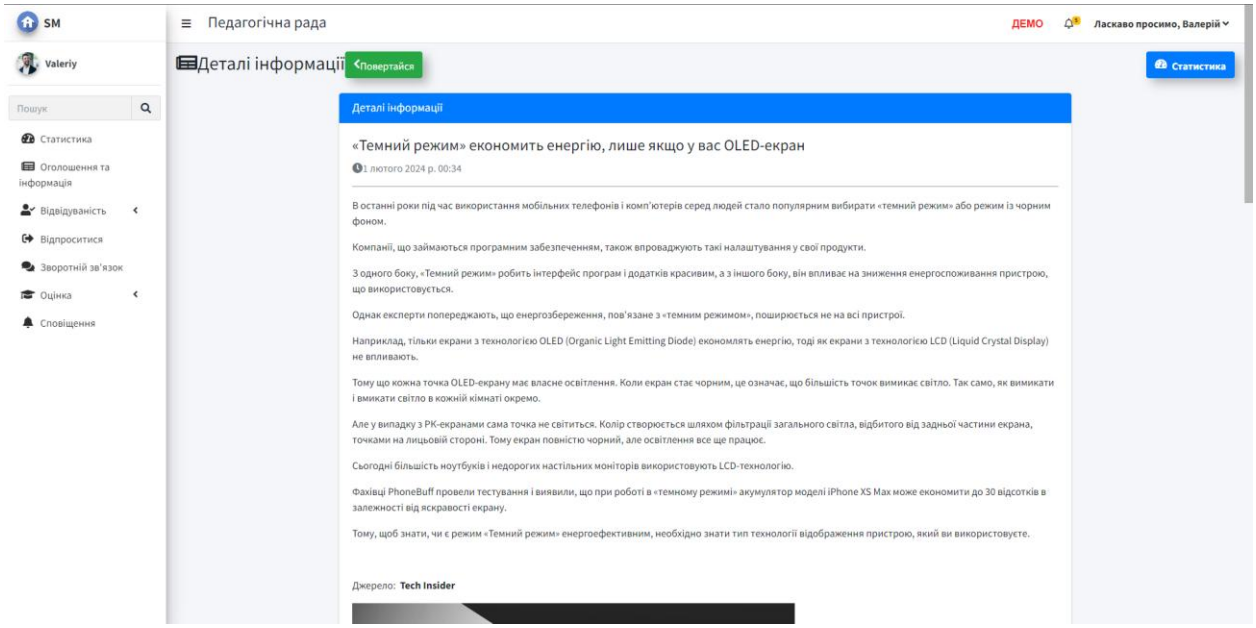

Рис. 4.16. Сповіщення студентів про події

Тепер буде представлено використання цього застосунку, з точки студента. Студент має менш доступний функціонал, оскільки він є працівником чи адміністратором даного веб-застосунку. Але у нього є можливість відслідковувати свою відвідуваність, кількість пропусків і кількість курсів до яких він приєднаний.

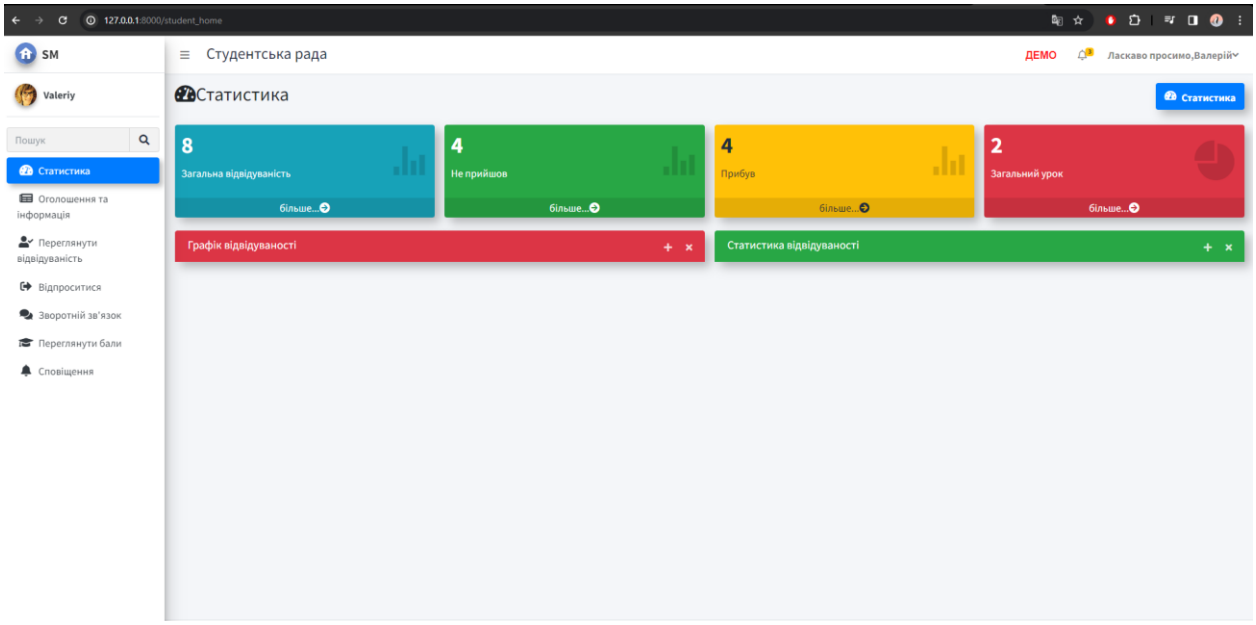

Рис. 4.17. Головна сторінка для студента

А зараз наведено приклад отримання особистих повідомлень для студента, які він може отримати від викладача чи інших працівників університету.

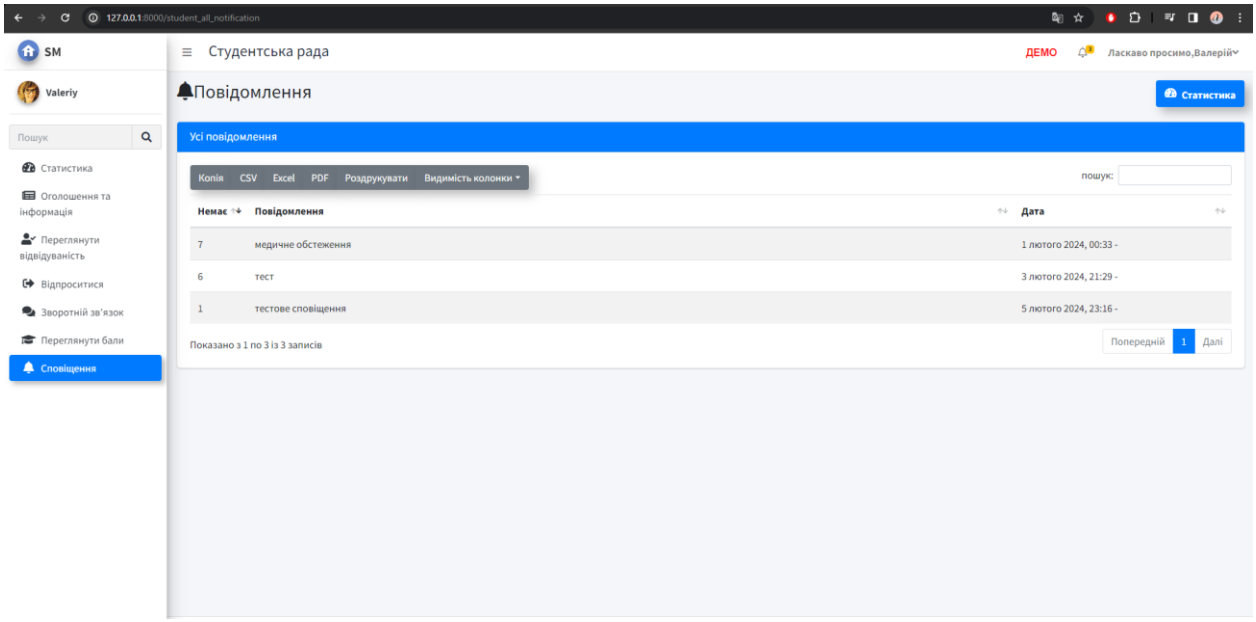

Рис. 4.18. Сповіщення для студента

Висновок

У цьому розділі мною було описано класи БД, продемонстровано повну ER-модель, а також показано зовнішній вигляд веб-застосунку сповіщення студентів про ключові події навчального процесу.

ВИСНОВКИ

В рамках даної дипломної роботи було проведено дослідження та розроблено веб-застосунок, спрямований на сповіщення студентів про ключові події навчального процесу. В процесі дослідження було визначено рентабельність веб-застосунків, проаналізовано предметну область веб-застосунків та їхні архітектури, а також було визначено потреби користувачів та сучасні підходи до розробки веб-додатків для підтримки навчального процесу.

Результатом роботи став розроблений прототип веб-застосунку, який дозволяє студентам отримувати оперативні сповіщення про події, такі як зміни в розкладі занять, нагадування про здачу завдань, оголошення важливих подій тощо. Веб-застосунок був реалізований з використанням сучасних технологій веб-розробки та дотриманням принципів дизайну інтерфейсів користувача.

Під час тестування прототипу було виявлено, що веб-застосунок відповідає функціональним та нефункціональним вимогам ПЗ, що підтверджує його потенціал для подальшого використання у навчальних закладах. Також, у майбутньому додаток можна розширити і добавити можливість батькам отримувати інформацію про студентів і отримувати зворотній зв’язок з викладачами. Однак, для досягнення повноцінного успіху впровадження, необхідно провести додаткове тестування серед різних груп користувачів та врахувати їхні фідбеки для подальшого вдосконалення продукту.

У цілому, розроблений веб-застосунок може значно полегшити спілкування між викладачами та студентами, сприяти своєчасному отриманню інформації про навчальні події та покращити загальний досвід навчання.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. "Web Application Development: Concepts, Methodologies, Tools, and Applications" - Редагована колективна робота, IGI Global, 2019.

2. "Designing Web APIs: Building APIs That Developers Love" - Бренда Фрізер, O'Reilly Media, 2018.

3. "Web Development with Node and Express: Leveraging the JavaScript Stack" - Етан Браун, O'Reilly Media, 2019.

4. "Learning React: A Hands-On Guide to Building Web Applications Using React and Redux" - Кірк Браун, Addison-Wesley Professional, 2017.

5. "User Interface Design and Evaluation" - Дебора Мей, Elsevier, 2014.

6. "Mobile Design and Development: Practical concepts and techniques for creating mobile sites and web apps" - Брайан Флінн, O'Reilly Media, 2009.

7. "Designing Web Interfaces: Principles and Patterns for Rich Interactions" - Білл Скотт, Тереза Нілссон, O'Reilly Media, 2009.

8. "Modern Web Development with ASP.NET Core 3: An end-to-end guide covering the latest features of Visual Studio 2019, Blazor, Entity Framework Core, and more" - Андерс Хофман, O'Reilly Media, 2020.

9. "The Elements of User Experience: User-Centered Design for the Web and Beyond" - Джессі Джеймс Гарретт, New Riders, 2010.

10. "Building Web Applications with Visual Studio 2017: Using .NET Core and Modern JavaScript Frameworks" - Philip Japikse, Kevin Grossnicklaus, O'Reilly Media, 2017.