КВАЛІФІКАЦІЙНА РОБОТА

Група ІПЗс-2019

Костишин Д. О.

2023

ЗВО УНІВЕРСИТЕТ КОРОЛЯ ДАНИЛА

Факультет суспільних та прикладних наук

Кафедра інформаційних технологій

на правах рукопису

Костишин Дмитро Олегович

УДК 004.378

Розробка системи підтримки навчального процесу з використанням функціоналу соціальних мереж.

Спеціальність 121 – «Інженерія програмного забезпечення»

Кваліфікаційна робота на здобуття кваліфікації бакалавра

Нормоконтроль

_____Стисло О.В.

(підпис, дата, розшифрування підпису)

Студент

_____Костишин Д.О.

(підпис, дата, розшифрування підпису)

Допускається до захисту

Завідувач кафедри

_____к.т.н., доц. Пашкевич О.П.

(підпис, дата, розшифрування підпису)

Керівник роботи

_____ к.т.н., доц. Пашкевич О.П

(підпис, дата, розшифрування підпису)

Івано-Франківськ – 2023

ЗВО УНІВЕРСИТЕТ КОРОЛЯ ДАНИЛА

Факультет суспільних та прикладних наук

Кафедра інформаційних технологій

Освітній ступінь: «бакалавр»

Спеціальність: 121 «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____

« ___ » _____ 2023 року

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Костишин Дмитро Олегович

(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи

Розробка системи підтримки навчального процесу з використанням функціоналу …соціальних мереж.

керівник роботи:

Пашкевич Олег Петрович, кандидат технічних наук

затверджена наказом вищого навчального закладу від « 11» листопада 2022 року № 155/1НВ

2. Термін подання студентом роботи 14.06.2023

3. Вихідні дані роботи: Мова програмування JavaScript, TypeScript, HTML, SCSS,

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

1. Огляд проблеми теперішньої системи

2. Реалізація прототипу та його недоліки

3. Реалізація системи проєктування навчального процесу UKD-NEXT

5. Дата видачі завдання 10.10.2022

КОНСУЛЬТАНТИ РОЗДІЛІВ КВАЛІФІКАЦІЙНОЇ РОБОТИ

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

КАЛЕНДАРНИЙ ПЛАН

No з/п	Назва етапів кваліфікаційної роботи	Термін виконання	Примітка
1.	Основні теоретичні відомості	15.02.2023	Виконано
2.	Створення Прототипу з дизайном	14.03.2023	Виконано
3.	Розробка архітектури клієнт-сервер	07.04.2023	Виконано
4.	Розробка програмного забезпечення мобільного додатка	28.04.2023	Виконано
5.	Інтеграція API сервера до мобільного додатка	05.05.2023	Виконано
6.	Оформлення пояснювальної записки	19.05.2023	Виконано
7.	Оформлення графічного матеріалу та підготовка до захисту роботи	03.06.2023	Виконано

Студент _____ Костишин Д.О.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Пашкевич О.П.
(підпис) (прізвище та ініціали)

Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень

Сторінка	Опис графічного матеріалу	Сторінка	Опис графічного матеріалу
14	Розклад в Деканат++	36	Профіль користувача в UKD-NEXT на iPhone 14 Pro

25	UKD Schedule запущений як Google Chrome додаток на Windows 11.	37	Новини в UKD-NEXT на iPhone 14 Pro
27	Схема роботи UKD Schedule	38	Схема бази даних UKD-NEXT
35	Розклад в UKD-NEXT на iPhone 14 Pro		

АНОТАЦІЯ

Дана робота присвячена розробці нової системи для студентів з метою поліпшення процесу отримання розкладу занять та інших повідомлень. У розділі 1 проведено аналіз проблеми існуючих систем та огляд розкладу занять від Деканат++. Зазначено вимоги до нової системи.

У розділі 2 надано опис прототипу нової системи та визначено вимоги з урахуванням недоліків попередніх систем. Проведено вибір інструментів розробки для реалізації системи.

Розділ 3 присвячений реалізації системи проєктування навчального процесу UKD NEXT. Розроблена архітектура системи, а також реалізовані розділи розкладу занять, новин та профілі студентів.

Отримані результати свідчать про успішну розробку нової системи, яка забезпечує студентам зручний доступ до розкладу занять та іншої інформації. Реалізація проєкту була здійснена з використанням відповідних інструментів розробки та архітектурних рішень.

КЛЮЧОВІ СЛОВА: РОЗКЛАД, MOBILE, WEB.

SUMMARY

This paper is dedicated to the development of a new system for students aimed at improving the process of accessing schedules and other notifications. In Section 1, an analysis of the problem with existing systems and an overview of the schedule from the Деканат++ are presented. The requirements for the new system are specified.

Section 2 provides a description of the prototype of the new system and defines the requirements, taking into account the shortcomings of previous systems. The selection of development tools for system implementation is also discussed.

Section 3 focuses on the implementation of the UKD NEXT educational process design system. The system architecture is developed, and sections for the schedule, news, and student profiles are implemented.

The obtained results demonstrate the successful development of a new system that provides students with convenient access to schedules and other information. The project implementation was carried out using appropriate development tools and architectural solutions.

KEYWORDS: SCHEDULE, MOBILE, WEB.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

БД - Бази даних;

UI - Інтерфейс користувача;

API - Прикладний програмний інтерфейс.

JSON API - це стандарт для структурування та обміну даними у форматі JSON в мережевих API.

PWA - (Progressive Web Application) це технологія, яка дозволяє створювати вебдодатки, які можуть працювати як нативні додатки на мобільних пристроях.

ВСТУП

Актуальність теми. У світі, де освіта визначає успіх, важливо мати доступ до актуальної інформації про навчальні заняття, а також зберігати її ефективно.

Одним з основних аспектів актуальності цієї проблеми є змінні графіки занять для студентів і школярів. Це може призводити до труднощів у запам'ятовуванні актуальної інформації та проблем з організацією часу. Ця непостійність може також впливати на стан студентів, викликаючи стрес.

У сучасному технологічному контексті, деякі засоби збереження графіків занять, такі як паперові записи або месенджери, стають недостатньо зручними та непрактичними. Введення програмного забезпечення для створення навчальних графіків може бути важливим кроком у розв'язанні цієї проблеми.

Проте, навіть з використанням програмного забезпечення, студентам все ще потрібно перевіряти графік щодня через його змінності. Деякі проблеми, як неадаптованість до мобільних пристроїв і відсутність історії пошуку, залишаються актуальними. Також важливо забезпечити повну інформацію про зміни у графіку занять для кращої прозорості і відповідальності управління навчальним процесом.

Отже, для досягнення більшої задоволеності користувачів і поліпшення організації навчання, необхідно удосконалювати існуючі рішення та зосередитися на усуненні недоліків, забезпечуючи більшу інформативність та доступність у системах керування графіком занять.

Мета роботи. Мета даної роботи полягає у розробці нової системи, спрямованої на поліпшення процесу отримання розкладу занять та інших повідомлень для студентів. Робота також має на меті визначення вимог до цієї системи та реалізацію її прототипу, з урахуванням недоліків попередніх систем.

Об’єкт роботи. Об'єктом даної роботи є розробка нової системи, спрямованої на поліпшення процесу отримання розкладу занять та інших

повідомлень для студентів. Система призначена для забезпечення зручного доступу студентів до необхідної інформації та полегшення їхнього навчального процесу. Об'єктом даної роботи є розробка нової системи, спрямованої на поліпшення процесу отримання розкладу занять та інших повідомлень для студентів. Система призначена для забезпечення зручного доступу студентів до необхідної інформації та полегшення їхнього навчального процесу.

Предмет роботи. Предметом даної роботи є розробка нової системи для студентів, яка спрямована на поліпшення отримання розкладу занять та інших повідомлень. Робота охоплює аналіз проблем існуючих систем, огляд розкладу занять, визначення вимог до нової системи, розробку прототипу та реалізацію проєкту. Предмет роботи також включає проєктування навчального процесу та забезпечення зручного доступу студентів до необхідної інформації.

Завдання роботи. Відповідно до вибраної теми в роботі покладені такі задачі:

- пошук існуючих на цей час аналогів для їхнього аналізу;
- вибір мови програмування та технологій розробки
- розроблення сучасного та зручного дизайну;
- розробка схеми бази даних;
- розробка серверної частини разом з мобільним додатком;
- проведення тестування продукту.

Методи роботи. Для вирішення поставленого завдання були використані мова програмування JavaScript, база даних PostgreSQL, фреймворк Nest.js, Node.js, React.js.

Результати роботи. Результати роботи свідчать про успішну розробку нової системи, яка забезпечує студентам доступ до розкладу та іншої інформації. Були проведені аналіз існуючих систем, визначені вимоги до нової системи, розроблений прототип та реалізовані розділи розкладу, новин та профілі студентів. Реалізація проєкту включала використання відповідних інструментів

та архітектурних рішень. Отримані результати свідчать про ефективність та функціональність нової системи, сприяючи поліпшенню навчального процесу.

Структура роботи. Розділи – 3. Загальний обсяг основної частини – 40 сторінок. Список використаних джерел – 20.

РОЗДІЛ 1. ОГЛЯД ПРОБЛЕМИ ТЕПЕРІШНЬОЇ СИСТЕМИ

1.1 Аналіз проблеми нотифікацій студентів

В сучасному освітньому середовищі, де гнучкість і зручність графіка занять відіграють важливу роль у раціональному управлінні часом, проблема нотифікацій студентів про зміни графіку предметів стає дедалі більш актуальною. Зміни в розкладах занять можуть виникати з різних причин, таких як перенесення лекцій, заміна викладачів або введення нових курсів. Нотифікація студентів про зміни є важливим для забезпечення їхнього своєчасного та належного інформування.

Одна з основних проблем полягає у тому, що існуючі системи нотифікацій не завжди ефективно функціонують. Багато університетів та шкіл використовують електронну пошту, платформи для навчання або спеціальні додатки для сповіщення студентів про зміни графіку. Однак, інформація може загубитися серед безлічі повідомлень або бути недостатньо помітною. Це призводить до ситуацій, коли студенти не отримують вчасно повідомлення про зміни та, як наслідок, пропускають заняття або неадекватно планують свій розклад.

Крім того, існуючі системи нотифікацій часто не дотримуються індивідуальних потреб студентів. У багатьох випадках, студентам надходять загальні повідомлення про зміни графіку, які можуть не враховувати їхніх особистих обставин або індивідуальних потреб. Наприклад, якщо студент має перекриття графіку між двома предметами, важливо мати можливість отримувати індивідуальні повідомлення, які надають альтернативні варіанти або можливість переглянути розклад занять.

Також варто зазначити, що існуючі системи нотифікацій можуть бути обмеженими в своїй функціональності та особностях адаптуватися до різних пристроїв. У сучасному світі студенти використовують різноманітні мобільні

пристрої, такі як смартфони та планшети, для доступу до інформації. Отже, системи нотифікацій повинні бути оптимізовані для різних платформ та мати зручний та простий інтерфейс.

Для розв'язання проблеми нотифікацій студентів про зміни графіку предметів необхідно впровадити більшість технологічні та інноваційні рішення. Перш за все, важливо розробити централізовану систему нотифікацій, яка поєднає усі елементи графіку занять та може надсилати індивідуальні повідомлення студентам залежно від їхніх потреб. Крім того, така система повинна бути легко доступною з будь-якого пристрою та мати зручний інтерфейс.

Додатково, можна використовувати сучасні технології сповіщень, такі як мобільні додатки або повідомлення у реальному часі, для забезпечення швидкого та ефективного інформування студентів про зміни графіку. Застосування таких технологій дозволить студентам отримувати миттєві сповіщення, а також мати можливість налаштовувати параметри сповіщень відповідно до своїх вимог.

Узагалі, аналіз проблеми нотифікацій студентів про зміни графіку предметів показує, що існуючі системи потребують поліпшення та модернізації. Впровадження централізованих систем нотифікацій з використанням сучасних технологій та індивідуальних налаштувань сповіщень може значно покращити інформування студентів про зміни графіку та забезпечити більш гнучке та зручне управління їхнім розкладом занять.

1.2 Огляд розклад занять від Деканат++

Університетське життя є непередбачуваним та завжди наповнене подіями. Студенти повинні вміти керувати своїм часом, забезпечувати баланс між навчанням, відпочинком та іншими справами. У цьому процесі ключову роль відіграє організація та доступність розкладу занять на (рис 1.1). Деканат++ – це

інноваційна система, яка забезпечує великий комфорт для студентів, допомагаючи їм впоратися з навчальними завданнями та іншими зобов'язаннями.

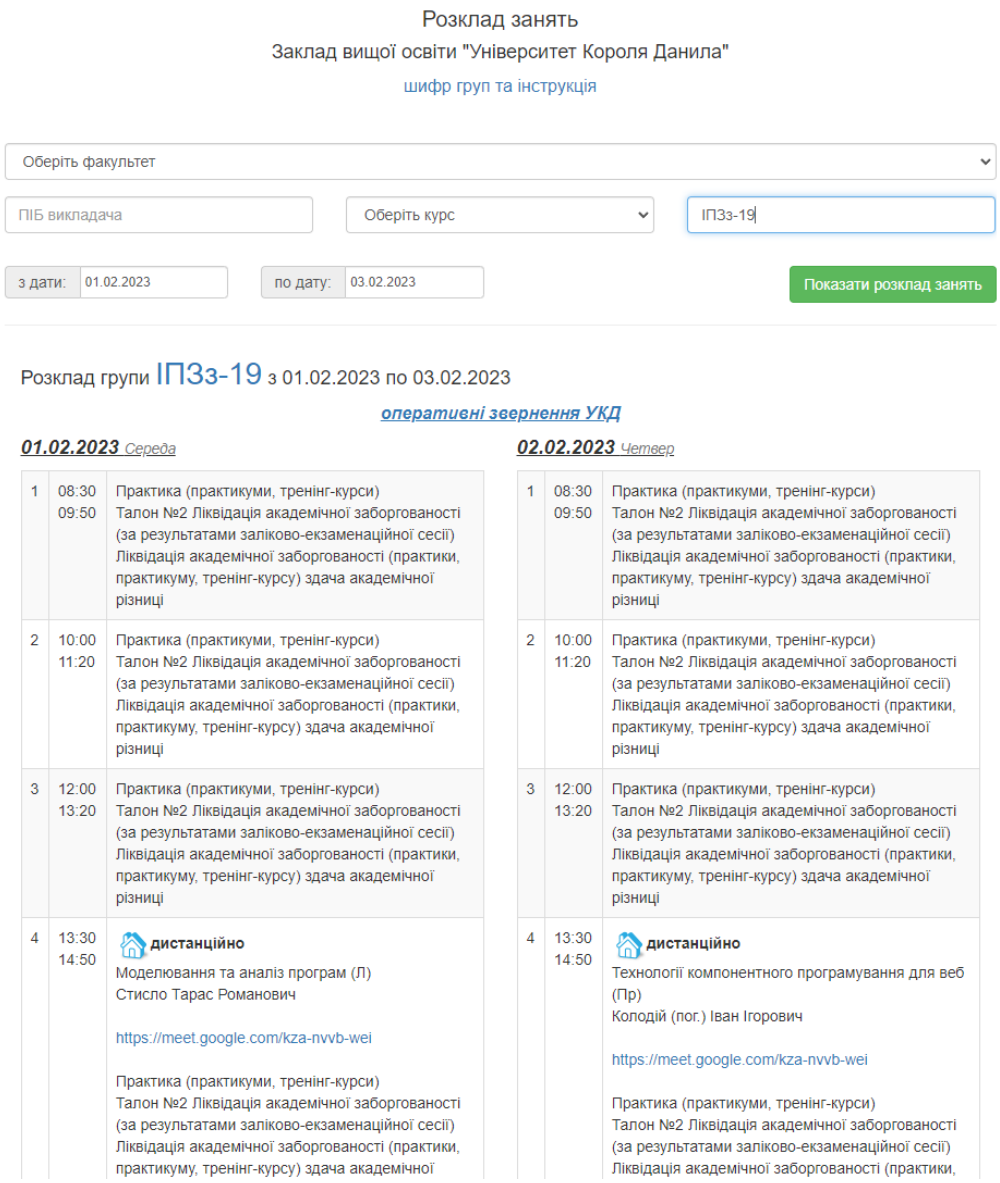

Рисунок 1.1 – Розклад в Деканат++

Переваги розкладу занять від Деканат++ є незаперечними, оскільки він забезпечує індивідуальний підхід до потреб студентів. Ця система враховує, що кожен студент має свої особисті пріоритети, інтереси та можливості, і дає змогу змінювати розклад занять відповідно до їхніх потреб.

Однією з великих переваг розкладу є гнучкість. Студентам надається можливість внести зміни у свій розклад, щоб відповідати їхнім особистим обставинам. Наприклад, якщо у студента виникла важлива зустріч чи подія, він може легко перенести заняття на інший час, щоб забезпечити собі вільний час. Це дозволяє уникнути перевантаження та більш ефективно організувати свій розклад.

Така гнучкість допомагає студентам зберегти баланс між навчанням, особистими зобов'язаннями та відпочинком. Вона дає можливість студентам активніше займатися додатковими активностями, розвивати свої інтереси та брати участь у заходах, які сприяють їхньому особистому й професійному росту.

Крім того, індивідуально налаштований розклад сприяє збереженню мотивації та підтримує позитивну навчальну атмосферу. Студенти відчувають більшу відповідальність за своє навчання, оскільки вони мають свій розклад. Це стимулює їх до більш активного й цілеспрямованого підходу до навчання, що сприяє покращенню академічних результатів.

Таким чином, розклад занять від Деканат++ не тільки враховує потреби студентів, але й надає їм гнучкість та можливість ефективно керувати своїм розкладом. Це дозволяє студентам бути більш організованими, зосередженими та успішними у своєму навчанні та особистому розвитку.

По-друге, розклад занять від Деканат++ забезпечує оптимальне використання часу, максимізуючи продуктивність студентів. Він побудований на основі принципів раціонального розподілу навчальних навантажень, уникнення заторів та мінімізації втрат часу між заняттями. Завдяки цьому підходу, студенти можуть ефективно планувати свій день, максимально використовувати кожну хвилину та зосереджуватися на навчанні.

Розклад занять ретельно розроблений з урахуванням часових і просторових обмежень, щоб уникнути перекриття занять та забезпечити достатній проміжок часу для підготовки між ними. Це дозволяє студентам не тільки присутність на

заняттях, але й зосередитися на освоєнні матеріалу, задавати питання та активно взаємодіяти з викладачами та однокурсниками.

Крім того, оптимально побудований розклад сприяє підвищенню ефективності навчання та розвитку студентського потенціалу. Він дозволяє студентам знайти баланс між навчанням та іншими активностями, такими як волонтерство, спорт або культурні заходи. Завдяки оптимізованому розкладу, студенти мають можливість гармонійно поєднувати свої навчальні зусилля з інтересами та розвитком особистості. Вони можуть займатися додатковими активностями, які поглиблюють їхні знання та навички, сприяють самореалізації та досягненню високих результатів.

Отже, розклад занять від Деканат++ створює сприятливі умови для розвитку студентського потенціалу та досягнення високих результатів. Він дозволяє студентам оптимально використовувати свій час, зосереджуватися на навчанні та одночасно розвивати свої інтереси та таланти. Цей підхід сприяє зростанню студентської самодисципліни, впевненості та здатності досягати успіху в усіх сферах свого життя.

По-третє, система Деканат++ надає можливість отримувати оновлення та інформацію про будь-які зміни в розкладі. Це дозволяє студентам завчасно планувати свої заняття та інші зобов'язання. Завдяки системі сповіщень, студенти отримують повідомлення про будь-які зміни в розкладі, такі як перенесення або скасування занять, зміна аудиторій тощо. Це дозволяє їм своєчасно організувати свій розклад і уникнути непередбачених ситуацій.

Крім того, розклад доступний в онлайн-форматі, що робить його доступним з будь-якого пристрою з Інтернет-підключенням. Студенти можуть легко отримати доступ до свого розкладу, навіть коли вони перебувають поза кампусом або в дорозі. Це дозволяє їм завчасно планувати свої дії та належним чином організовувати свій час. Наприклад, студенти можуть заздалегідь побачити, які

заняття їх очікують найближчими днями та згідно з цим планувати свої особисті зустрічі, заняття в спортзалі або відвідування заходів.

Наявність онлайн-розкладу занять також забезпечує зручність та гнучкість для студентів. Вони можуть звертатися до розкладу в будь-який час, коли виникає потреба або змінюються обставини. Наприклад, якщо студент забув про певне заняття або потребує додаткової інформації щодо групи, він може швидко відкрити розклад на своєму смартфоні та знайти необхідну інформацію. Така гнучкість дозволяє студентам ефективно управляти своїм часом та виконувати свої навчальні обов'язки без зайвих труднощів та затримок.

Загалом, розклад занять від Деканат++ забезпечує студентам зручність, доступність та гнучкість при плануванні свого навчального процесу. Він допомагає студентам завчасно організувати свій розклад, отримувати оновлення та інформацію про зміни та забезпечує доступ до розкладу з будь-якого пристрою. Такий підхід допомагає студентам ефективно використовувати свій час, планувати свої дії та забезпечує їхній зручний і успішний студентський досвід.

Розклад занять від Деканат++ також сприяє покращенню студентського досвіду. Він дозволяє студентам планувати свій час таким чином, щоб вони могли займатися іншими активностями, такими як спорт, культурні заходи, волонтерство тощо. Це сприяє їх загальному розвитку та допомагає зберегти баланс між навчанням та особистими інтересами.

Завдяки розкладу занять від Деканат++, студенти можуть ефективно використовувати свій час, знаючи точний графік своїх занять. Це дозволяє їм встановити пріоритети і забезпечити собі достатньо часу для навчання та інших занять, які вони цінують. Наприклад, студенти можуть запланувати тренування в спортивному залі, відвідування або участь у культурних заходах.

Крім того, розклад занять допомагає студентам визначати доступні проміжки часу, які вони можуть використовувати для здійснення волонтерської діяльності. Волонтерство є важливою складовою студентського життя, оскільки воно дозволяє

набути нових навичок, знайти прихильників та зробити позитивний внесок у суспільство. Розклад занять, який забезпечує достатньо вільного часу для волонтерської роботи, сприяє розвитку активної громадянської позиції студентів та формуванню їхнього особистісного росту.

Організація розкладу занять від Деканат++ з урахуванням інших активностей студентів допомагає їм уникнути перевантаження та вигорання, що можуть виникнути при неконтрольованому розподілі часу. Збереження балансу між навчанням та особистими інтересами дозволяє студентам ефективно використовувати свій потенціал та отримувати повноцінний досвід університетського життя.

Одним з недоліків розкладу занять від Деканат++ є його нездатність зберігати вибрану групу, по якій шукає користувач. Це може стати проблемою для студентів, які відвідують заняття разом зі своєю групою і хочуть бути впевненими, що отримують актуальну інформацію щодо змін в розкладі лише для своєї групи. Відсутність цієї функції може призвести до незручностей та плутанини, коли студенти втрачають час на пошук своєї групи у загальному розкладі. Крім того, це може призвести до неправильної інтерпретації інформації, якщо студенти помилково використовують зміни, призначені для інших груп.

Для вирішення цього недоліку важливо впровадити функціонал збереження вибраної групи в розкладі занять від Деканат++. Це може бути зроблено шляхом створення особистих облікових записів для студентів, де вони можуть вибрати свою групу та зберегти її в налаштуваннях. Після цього система буде автоматично відображати тільки розклад занять для вибраної групи студента, уникнувши змішування з розкладами інших груп.

Такий функціонал дозволить студентам швидко та легко отримувати актуальну інформацію щодо змін в розкладі занять лише для їх обраної групи. Це зробить процес пошуку та сприйняття інформації більш зручним та ефективним. Крім того, такий підхід забезпечить студентам впевненість у тому, що вони

отримують правильну та релевантну інформацію щодо розкладу занять для їх конкретної групи, сприяючи організації їх навчального процесу та уникненню непотрібних незручностей.

Ще одним недоліком розкладу занять від Деканат++ є його неадаптованість під мобільні пристрої. У сучасному світі, коли мобільність стала невід'ємною частиною нашого життя, важливо мати доступ до необхідної інформації в будь-який час та в будь-якому місці. Недоступність адаптованої версії розкладу для мобільних пристроїв обмежує студентів у їх можливостях отримувати актуальну інформацію та організовувати свій розклад занять в зручний спосіб. Ця проблема може створювати незручності для студентів, які часто пересуваються, відсутні в університеті або працюють з мобільних пристроїв.

Для вирішення цього недоліку необхідно розробити мобільну версію розкладу занять від Деканат++. мобільний додаток або вебверсія, оптимізовані для використання на різних типах мобільних пристроїв, дозволять студентам зручно переглядати свій розклад занять, отримувати оновлення про зміни та отримувати повідомлення про нагадування або повідомлення від викладачів.

Мобільна адаптація розкладу занять покращить доступність і зручність його використання для студентів, що дозволить їм організовувати свій час ефективніше та враховувати зміни в розкладі навчання навіть під час пересування або відсутності в університеті. Такий мобільний функціонал сприятиме зручності та задоволенню студентів, сприяючи їхній успішності та ефективності навчання.

Третім недоліком є відсутність повідомлень про зміни в розкладі занять. Це може стати серйозною перепоною для студентів, оскільки зміни в розкладі можуть виникати з невеликим попередженням або навіть нещадно. Відсутність повідомлень унеможливлює своєчасне сповіщення студентів про зміни та ускладнює їх можливість планувати свій час та займатися іншими діяльностями. Ця проблема може призвести до незручностей, непередбачуваності та навіть пропуску важливих занять або подій.

Недолік може бути вирішений шляхом впровадження системи повідомлень про зміни в розкладі занять. Така система може включати механізм надсилання повідомлень студентам через електронну пошту, мобільні додатки або повідомлення у вебінтерфейс. Це дозволить студентам своєчасно отримувати інформацію про будь-які зміни в розкладі, такі як скасування занять, зміна часу або місця проведення. Такі повідомлення дадуть студентам можливість організувати свій час, уникнути незручностей і запланувати свої діяльності.

Крім того, система повідомлень може мати інтерактивний характер, дозволяючи студентам підтверджувати отримання повідомлень та підтверджувати свою доступність на заняттях. Це полегшить комунікацію між викладачами та студентами, забезпечить точність та взаєморозуміння.

Впровадження системи повідомлень про зміни в розкладі занять буде великим кроком у поліпшенні організації університет

Четвертим недоліком є те, що система має вразливість до DDoS-атаки, яка може призвести до не роботоздатності, якщо використовувати цей скрипт, який робить 200 запитів на сторінку водночас. Така атака використовує можливість виконання багатьох паралельних запитів до сервера, перевантажуючи його ресурси та заважаючи нормальному функціонуванню. Ось код скрипта, який виконує цю DDoS-атаку:

```
const SITE_URL = "http://195.162.83.28";
const NUMBER_OF_REQUESTS = 200;

function request() {
return fetch(SITE_URL + "/cgi-bin/timetable.cgi?n=700", {
  body:
"faculty=0&teacher=&course=0&group=%B2%CF%C7%E7-19&sdate=01.01.2019&edate=31.12.2029
&n=700",
  method: "POST",
}).then((res) => res.text());
}
```

```
for (let i = 0; i <= NUMBER_OF_REQUESTS; i++) {
request();
}
```

Цей скрипт виконує 200 паралельних запитів до вказаної сторінки, використовуючи цикл **for**. Кожен запит виконується з використанням функції **request()**, яка виконує HTTP-запит до сервера. При великій кількості запитів система може перевантажитися і стати не роботоздатною.

Для захисту від таких атак необхідно приймати заходи, такі як використання фаєрволу, обмеження кількості одночасних запитів від одного джерела, моніторинг та виявлення підозрілих активностей, а також застосування технологій, що дозволяють розподілити навантаження між декількома серверами. Заходи безпеки повинні бути впроваджені для запобігання подібним атакам і забезпечення стабільності та надійності системи

У підсумку, розклад занять від Деканат++ є вагомим кроком до покращення студентського життя. Він допомагає студентам бути більш організованими, ефективно керувати своїм часом та досягати успіху як у навчанні, так і в особистому розвитку. Інноваційність та гнучкість цієї системи дозволяють студентам зосередитися на своїх цілях та мріях, не відчуваючи надмірного стресу та перевантаження. Деканат++ створює оптимальні умови для росту та розвитку студентського потенціалу, що робить його незамінним інструментом для сучасних студентів.

1.3 Постанова вимог

Список вимог до нової системи, яка враховуватиме недоліки системи Деканат++, міг би виглядати так:

1. Збереження вибраної групи:

- можливість студентів обрати конкретну групу і отримувати актуальну інформацію про зміни в розкладі лише для цієї групи;
- зручний доступ до розкладу занять для обраної групи без необхідності шукати її у загальному розкладі.

2. Адаптація для мобільних пристроїв:

- розробка мобільної версії системи, яка буде оптимізована для використання на смартфонах та інших мобільних пристроях;
- гнучкий та зручний інтерфейс для перегляду та організації розкладу занять на мобільних пристроях.

3. Повідомлення про зміни в розкладі:

- механізм сповіщення студентів про будь-які зміни в розкладі занять, включаючи перенесення, скасування або додаткові заняття;
- різні способи сповіщення (наприклад, повідомлення на мобільний пристрій або електронна пошта) з можливістю налаштування пріоритетності та наявності розкладу.

Для покращення функціональності та задоволення користувачів нової системи, необхідно врахувати ряд вимог, що забезпечать їм більш зручний та ефективний спосіб керування розкладом занять та організації студентського життя.

По-перше, важливо забезпечити інтуїтивно зрозумілий та простий інтерфейс системи. Користувачі повинні легко знаходити потрібну інформацію та мати можливість швидко змінювати свій розклад занять. Чітке та логічне розташування функцій і опцій допоможе уникнути заплутаності та збільшить задоволення користувачів від використання системи.

По-друге, важливо забезпечити мобільну доступність системи. У сучасному світі, коли мобільність стала невід'ємною частиною нашого життя, студентам потрібно мати можливість отримувати доступ до свого розкладу та вносити зміни у нього з будь-якого мобільного пристрою. Розробка спеціальної мобільної

програми або адаптивного вебінтерфейс дозволить користувачам зручно користуватися системою навіть поза межами кампусу.

По-третє, система повинна підтримувати сповіщення та нагадування. Користувачі мають мати можливість налаштовувати повідомлення про зміни в розкладі, нагадування про надходження занять або інших подій. Це допоможе студентам бути в курсі актуальної інформації та запобігати пропускам або непередбаченим змінам у їхньому розкладі.

По-четверте, система повинна підтримувати можливість інтеграції з іншими календарними додатками або сервісами. Багато студентів використовують різні календарні програми або онлайн-сервіси для організації свого часу. Забезпечення сумісності та можливості імпорту/експорту розкладу між системою Деканат++ та іншими популярними календарними додатками дозволить студентам зручно синхронізувати свої заняття з іншими планувальними інструментами.

По-п'яте, система повинна забезпечувати надійність та безпеку даних. Важливо мати механізми резервного копіювання даних, щоб уникнути втрати інформації, а також забезпечити захист персональних даних студентів. Використання шифрування та інших методів безпеки допоможе зберегти конфіденційність та запобігти несанкціонованому доступу до даних.

Загалом, врахування цих вимог допоможе покращити функціональність та задоволення користувачів нової системи, забезпечуючи їм більш зручний та ефективний спосіб керування своїм розкладом занять та організації студентського життя.

Висновок до розділу 1

У цьому розділі було розглянуто недоліки існуючої системи Деканат++ і визначено список вимог до нової системи. Виявлені недоліки, такі як обмежена доступність розкладу для студентів та неадаптованість системи до мобільних

пристроїв, свідчать про необхідність удосконалення та модернізації системи управління розкладом занять.

Перша вимога полягає у забезпеченні можливості студентам обирати конкретну групу та отримувати актуальну інформацію про зміни в розкладі лише для цієї групи. Це дозволить студентам ефективніше організовувати свій навчальний процес, отримувати необхідну інформацію без зайвих зусиль та зберігати персоналізовані налаштування.

Друга вимога стосується адаптації системи для мобільних пристроїв. Розробка мобільної версії системи, яка буде оптимізована для використання на смартфонах та інших мобільних пристроях, є важливим кроком у покращенні доступності та зручності користування. Гнучкий та зручний інтерфейс на мобільних пристроях дозволить студентам легко переглядати та організовувати свій розклад занять навіть в дорозі або поза навчальним закладом.

Третя вимога стосується повідомлень про зміни в розкладі. Механізм сповіщення студентів про будь-які зміни в розкладі занять, включаючи перенесення, скасування або додаткові заняття, є важливою складовою ефективного управління розкладом. Різні способи сповіщення, такі як повідомлення на мобільні пристрої або електронна пошта, з можливістю налаштування пріоритетності та наявності розкладу, допоможуть студентам завжди бути в курсі оновлень та змін.

Зазначені вимоги до нової системи мають на меті покращення функціональності та задоволення користувачів. Шляхом впровадження цих поліпшень, студентам буде надано більш зручний та ефективний спосіб керування своїм розкладом занять та організації студентського життя. Окрім того, впровадження нової системи сприятиме підвищенню продуктивності та ефективності роботи адміністрації навчального закладу, що позитивно вплине на всю навчальну спільноту.

РОЗДІЛ 2. РЕАЛІЗАЦІЯ ПРОТОТИПУ ТА ЙОГО НЕДОЛІКИ

2.1 Опис прототипу нової системи

З метою поліпшення організації навчального процесу для студентів, я та Владислав Лутчин розробили прототип додатку під назвою UKD Schedule на (рис 2.1). Цей додаток призначений для зручного перегляду актуального навчального графіка студентами. Він має зручний інтерфейс та оптимізований під різні мобільні пристрої.

Рисунок 2.1 – UKD Schedule запущений як Google Chrome додаток на Windows 11.

Основна мета UKD Schedule - спростити організацію навчання та забезпечити студентам зручний доступ до актуальної інформації про їх навчальний графік. Цей додаток допомагає студентам легко організувати свій навчальний розклад і мати доступ до актуальної інформації в будь-який час та місці.

Прототип UKD Schedule забезпечує наступні функціональні можливості:

1. Перегляд актуального навчального графіка: Студенти можуть легко переглядати свій навчальний графік зручним інтерфейсом додатку.

2. Зручний пошук: Додаток зберігає історію пошуку, що дозволяє студентам швидко знайти графік за певними параметрами, спрощуючи процес пошуку.

3. Адаптованість до різних пристроїв: UKD Schedule розроблений таким чином, щоб працювати на різних мобільних пристроях, забезпечуючи зручний доступ до навчального графіка незалежно від використовуваного пристрою.

4. Режим офлайн: Додаток можна використовувати навіть у відсутності Інтернет-з'єднання, завдяки його здатності працювати в офлайн-режимі. Це дозволяє студентам мати доступ до свого навчального розкладу в будь-який момент, що є зручно.

Прототип системи UKD Schedule базується на такій архітектурі (рис 2.2): існує серверна частина, що надає JSON API. При виклику цього API, сервер встановлює з'єднання з вебсайт розкладу університету та отримує відповідні дані в форматі JSON. Далі ці дані парситься і повертаються на сторону клієнта.

Така архітектура додатка забезпечує швидку та зручну передачу актуального розкладу на сторону клієнта. Студенти можуть оновлювати та отримувати актуальні дані про свої предмети на наступні сім днів. Крім того, завдяки можливості роботи в офлайн-режимі, вони можуть використовувати додаток навіть без доступу до Інтернету.

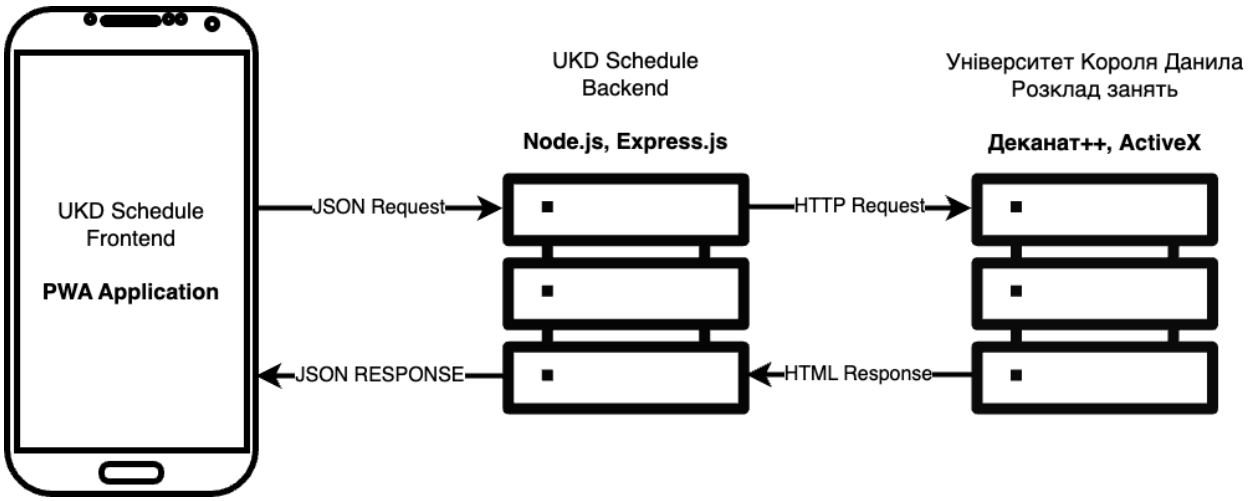

Рисунок 2.2 – Схема роботи UKD Schedule

Цей прототип системи UKD Schedule надає гнучкість та зручність використання, оскільки студенти можуть отримати доступ до свого навчального розкладу на будь-якій платформі, яка підтримує PWA (Progressive Web App). Завдяки цьому, користувачі можуть контролювати свій розклад університету у зручний для них спосіб і не залежати від пристрою чи операційної системи.

Прототип UKD Schedule дозволяє студентам отримувати доступ до актуальної інформації про навчальний графік в будь-який час і з будь-якого місця. Вони можуть переглядати свій розклад занять на своєму смартфоні, планшеті, ноутбуці або навіть на персональному комп'ютері через веббраузер. Додаток адаптується до екрана та можливостей кожного пристрою, забезпечуючи максимальний комфорт при користуванні.

Завдяки прогресивному вебдодатку, UKD Schedule не потребує установки з магазинів додатків і займає мінімальний обсяг пам'яті на пристрої. Користувачі можуть просто відкрити додаток у веббраузер і отримати доступ до свого розкладу занять без додаткових зусиль. Це робить навчання більш зручним та ефективним, оскільки студенти завжди мають під рукою свій навчальний графік.

Прототип UKD Schedule також надає різноманітні можливості для ведення навчального розкладу та планування навчального процесу. Крім перегляду

розкладу занять, студенти можуть встановлювати нагадування про певні події або зміни у розкладі, додавати особисті нотатки та мітки до конкретних занять, а також редагувати свої пріоритети та встановлювати періоди відпочинку. Все це сприяє більш ефективному плануванню та організації навчального процесу студентами.

Загалом, прототип UKD Schedule є надійним помічником студентів у веденні навчального розкладу та плануванні навчального процесу. Він забезпечує зручний доступ до актуальної інформації про навчальний графік, гнучкість у використанні на різних платформах та різних пристроях, а також розширені можливості для персоналізації та оптимізації навчального процесу студентами.

2.2 Вимоги до нової системи з врахуванням усіх недоліків попередніх систем

У зв'язку з недоліками, виявленими у попередній системі, нова система має задовольняти ряд вимог для поліпшення її функціональності та зручності використання. З метою врахування усіх попередніх недоліків і запобігання їх повторенню, наступні вимоги вважаються пріоритетними:

1. Можливість вибору групи: Нова система повинна надавати користувачам можливість вибору своєї групи при реєстрації або пізніше в особистому кабінеті. Це дозволить кожному конкретному користувачеві зручно планувати свої активності, мати доступ до необхідної інформації та спілкуватися з однодумцями у відповідній групі.

2. Можливість вибору конкретного дня в розкладі: Враховуючи, що користувачі можуть мати різний графік або особисті обставини, необхідно забезпечити можливість вибору конкретного дня для отримання розкладу занять або подій. Це дозволить користувачам індивідуалізувати свій графік і забезпечить їм більшу гнучкість у плануванні своїх активностей.

3. Система push-повідомлень: Враховуючи важливість оперативної

інформації про зміни в графіку або інших подіях, нова система повинна мати можливість надсилати push-повідомлення користувачам. Це дозволить сповіщати їх про будь-які зміни в реальному часі і уникнути пропуску важливої інформації.

4. Зручний інтерфейс та навігація: Нова система повинна мати простий і зрозумілий інтерфейс, який дозволяє користувачам швидко.

Зорієнтуватися і знайти потрібну інформацію без зайвих зусиль. Це забезпечить ефективне використання системи і зменшить час, витрачений на пошук потрібних функцій.

5. Гнучкість і сумісність: Нова система повинна бути гнучкою і сумісною з різними пристроями та операційними системами, щоб користувачі могли отримувати доступ до неї з будь-якого зручного пристрою без обмежень.

Враховуючи ці вимоги, нова система з врахуванням недоліків попередніх систем буде забезпечувати кращу функціональність, зручність використання та задоволення користувачів, пропонуючи їм більш гнучкі інструменти для планування своїх активностей та отримання необхідної інформації.

2.3 Вибір інструментів розробки

Розділ "Вибір інструментів розробки" включає набір інструментів, які можна використовувати для розробки вашого проєкту. Ось опис кожного інструменту з переліку:

1. React.js: це бібліотека JavaScript для побудови інтерфейсу користувача. Вона дозволяє розробникам побудувати ефективний інтерактивний інтерфейс для вашого додатка [1].

2. PWA (Progressive Web Application): це технологія, що дозволяє створювати вебдодатки з можливостями, подібними до нативних мобільних додатків. Вона дозволяє вашому додатку працювати офлайн та використовувати різні функції мобільного пристрою, джерело [2].

3. React-router-dom: це бібліотека для маршрутизації в React додатку. Вона дозволяє вам організувати навігацію між різними сторінками додатка [3].

4. SASS: це препроцесор CSS, який надає покращені можливості для створення стилів вашого додатка. Він дозволяє використовувати змінні, міксіни та багато інших корисних функцій для полегшення розробки та підтримки [4].

5. TypeScript: це мова програмування, що розширює JavaScript додатковими функціями типізації. Вона дозволяє вам визначати типи даних для змінних, функцій та об'єктів, що полегшує виявлення помилок та поліпшує інтеграцію з іншими інструментами розробки [5].

6. Ant Design: це набір готових компонентів для побудови інтерфейсу користувача в React додатках. Він надає різноманітні стильні компоненти, які можна використовувати для швидкої розробки користувацького інтерфейсу [6].

7. JavaScript: це мова програмування, яка використовується для реалізації клієнтської логіки. Ви можете використовувати JavaScript разом з React.js для створення динамічних функцій та взаємодії з користувачем [7].

8. Lodash: це бібліотека з корисними функціями для маніпулювання даними. Вона надає різноманітні утиліти, які допомагають спростити та прискорити операції з масивами, об'єктами, рядками та іншими типами даних [16].

9. Vite: це швидкий і мінімалістичний інструмент для розробки веб-додатків. Він надає швидку реакцію при розробці, швидку перезавантаження та оптимізацію залежностей для покращення продуктивності вашого проекту [9].

10. NX: це інструмент для розробки монорепозиторіїв з багатьма проектами. Він дозволяє організувати ваш проект у вигляді колекції пакетів та допомагає в управлінні залежностями, тестуванні та використанні спільного коду між різними проектами [10].

11. Ant Design Pro: це набір готових компонентів для побудови інтерфейсу користувача в React додатках для адміністраторів. Він надає різноманітні готові компоненти та шаблони для створення ефективних та панелей управління [19].

12. Node.js: це середовище виконання JavaScript, засноване на двигуні V8. Воно дозволяє вам виконувати JavaScript-код на сервері та створювати серверні додатки, API та інші бекенд-рішення [12].

13. Nest.js: це фреймворк для побудови масштабованих серверних додатків на Node.js. Він надає структуру та конвенції для розробки, що допомагають створювати ефективні та організовані бекенд-додатки [13].

14. PostgreSQL: це реляційна база даних, яка забезпечує надійне зберігання та керування даними. Вона використовує SQL для операцій з даними та може бути інтегрована з вашим серверним додатком [14].

15. TypeOrm: це ORM (Object-Relational Mapping), яке дозволяє зв'язувати базу даних з об'єктами вашої програми. Воно полегшує роботу з базою даних і дозволяє вам використовувати об'єктно-орієнтований підхід до роботи з даними.

16. pg: це модуль для зв'язку з PostgreSQL базою даних з використанням Node.js. Він надає простий та зручний спосіб взаємодії з базою даних з використанням JavaScript [17].

17. WebSocket: це протокол для двостороннього зв'язку між клієнтом та сервером через веб-сокети. Він дозволяє реалізувати взаємодію в режимі реального часу та передавати потокові дані між клієнтом та сервером [18].

18. Passport.js: це бібліотека для аутентифікації та авторизації користувачів. Вона надає різноманітні стратегії аутентифікації, такі як аутентифікація з використанням логіну/пароля, соціальних медіа тощо, і спрощує процес авторизації в вашому додатку.

19. Swagger: це інструмент для створення документації API. Він дозволяє описувати ваші API-шляхи, параметри, типи даних тощо, і автоматично генерує документацію, яка допомагає іншим розробникам використовувати ваше API.

20. Axios: це бібліотека для виконання HTTP-запитів з використанням JavaScript. Вона надає простий та зручний спосіб взаємодії з сервером, отримання та відправлення даних.

21. Class-transformer: це бібліотека для перетворення об'єктів. Вона дозволяє вам легко конвертувати об'єкти з одного формату в інший, зокрема для роботи з даними, що отримані з сервера.

22. Class-validator: це бібліотека для валідації даних. Вона дозволяє вам встановлювати правила валідації для об'єктів і перевіряти, чи задовольняють дані правила валідації [20].

Ці інструменти можна комбінувати та використовувати разом для створення потужних та ефективних веб-додатків з покращеними можливостями, швидкістю та розширюваністю. Вибір конкретних інструментів залежить від ваших потреб та вимог проекту.

Висновок до розділу 2

У розділі 2 було розглянуто прототип додатку під назвою UKD Schedule, який був розроблений з метою поліпшення організації навчального процесу для студентів. Цей додаток надає зручний доступ до актуального навчального графіка та спрощує організацію навчання. Він має зручний інтерфейс та може працювати на різних мобільних пристроях.

Прототип UKD Schedule забезпечує такі функціональні можливості, як перегляд актуального навчального графіка, зручний пошук, адаптованість до різних пристроїв та режим офлайн. Цей додаток базується на серверній архітектурі, яка дозволяє швидку передачу розкладу на сторону клієнта.

З урахуванням недоліків попередніх систем, в розділі було сформульовано вимоги до нової системи. Пріоритетними вимогами є можливість вибору групи та конкретного дня в розкладі, система push-повідомлень, зручний інтерфейс та навігація, а також сумісність з різними пристроями та операційними системами.

З урахуванням цих вимог, нова система з врахуванням недоліків попередніх систем буде забезпечувати кращу функціональність, зручність використання та задоволення користувачів.

У розділі також було зроблено вибір інструментів розробки, які будуть використовуватись для реалізації нової системи. Детальніші відомості про цей вибір можна знайти у відповідному розділі.

Загалом, розділ 2 надає важливу інформацію про прототип

РОЗДІЛ 3. РЕАЛІЗАЦІЯ СИСТЕМИ ПРОЄКТУВАННЯ НАВЧАЛЬНОГО ПРОЦЕСУ UKD NEXT

UKD NEXT - це потужна програмна система, розроблена в якості ключового компонента інформаційного сервісу університету, яка надає студентам широкий спектр зручних та корисних функціональних можливостей. Ця інноваційна система створена з метою повної автоматизації процесу складання розкладу занять, враховуючи безліч факторів, що впливають на оптимальне планування навчальних занять.

UKD NEXT використовує розумні алгоритми, які враховують доступні аудиторії та їх специфічні особливості, індивідуальні потреби груп студентів, часовий графік та важливу інформацію про кількість предметів, які необхідно викладати для конкретних груп на тиждень. Завдяки цим функціям, UKD NEXT забезпечує оптимальне розподілення навчальних занять, мінімізуючи перекриття та конфлікти в розкладі.

Крім того, система UKD NEXT має інтуїтивно зрозумілий і легкий у використанні інтерфейс, що дозволяє студентам швидко та зручно отримувати доступ до своїх розкладів занять.

UKD NEXT - це незамінний інструмент для університетського співтовариства, що допомагає оптимізувати процес планування навчальних занять та забезпечує ефективну організацію навчального процесу для всіх студентів. Завдяки цій системі, університет здатний забезпечити максимальний рівень задоволення студентів щодо їх навчального досвіду та покращити загальну продуктивність навчального процесу.

UKD NEXT має декілька особливостей, які роблять його корисним інструментом для студентів університету:

1. Гнучкість та оптимізація розкладу на (рис 3.1): UKD NEXT враховує різні фактори, такі як доступні аудиторії, їх особливості, групи студентів і часовий графік. Система маніпулює цими даними, щоб забезпечити оптимальне розподілення предметів і аудиторій, мінімізуючи конфлікти та забезпечуючи максимальну ефективність навчання.

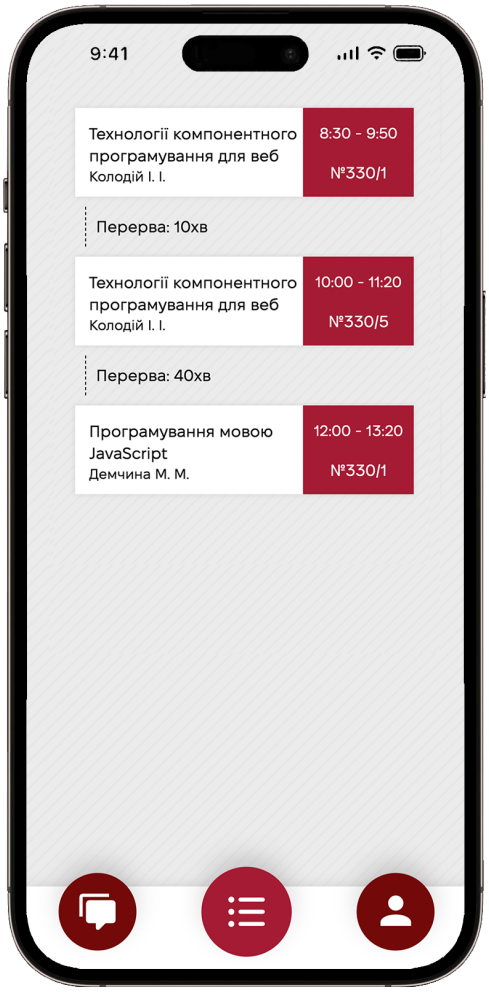

Рисунок 3.1 – Розклад в UKD-NEXT на iPhone 14 Pro

2. Особистий профіль учня на (рис 3.2): Кожен студент має свій особистий профіль у системі UKD NEXT. Цей профіль містить інформацію про навчальні досягнення, академічні рейтинги, інформацію про групу та інші

персональні дані. Студент може відстежувати свій прогрес, отримувати індивідуальні поради та рекомендації щодо свого навчання.

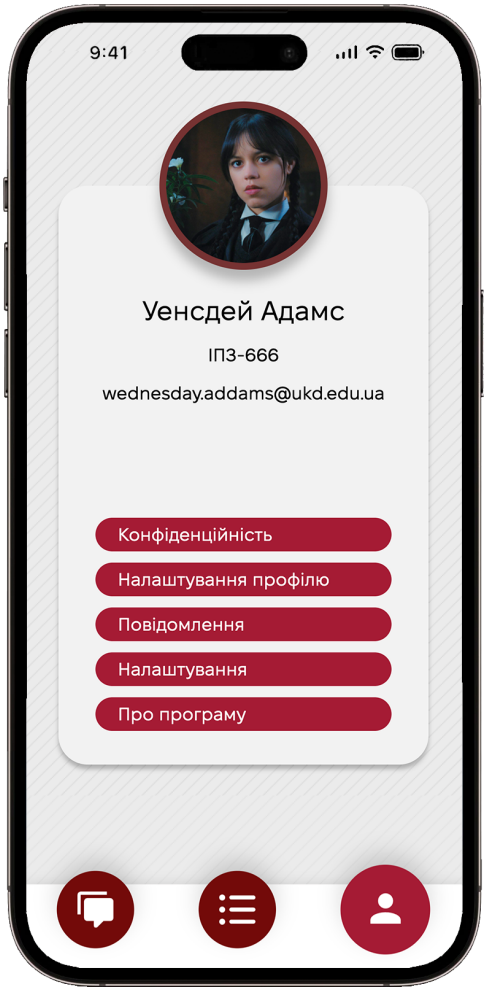

Рисунок 3.2 – Профіль користувача в UKD-NEXT на iPhone 14 Pro

3. Новини на (рис 3.3): Система UKD NEXT надає студентам доступ до свіжих новин та оголошень, пов'язаних з університетом. Студенти можуть бути в курсі подій, актуальної інформації та змін, які відбуваються в університетському середовищі. Це допомагає студентам бути орієнтованими та своєчасно отримувати важливу інформацію.

Рисунок 3.3 – Новини в UKD-NEXT на iPhone 14 Pro

4. Евенти: UKD NEXT надає інформацію про різноманітні заходи, які відбуваються в університеті, такі як академічні конференції, семінари, спортивні події тощо. Студенти можуть ознайомитися з цими заходами та приймати участь в них, що сприяє їхньому активному участю в університетському житті.

5. Рейтинг учнів: UKD NEXT може відображати рейтинг студентів на основі їхніх академічних досягнень та успішності. Це допомагає студентам відстежувати свої результати порівняно з іншими студентами і надихає до здобуття кращих результатів.

Загалом, UKD NEXT є потужним інструментом, який спрощує процес навчання та допомагає студентам бути організованими, інформованими та активними учасниками університетського життя.

3.1 Розробка архітектури системи

Розділ включає структурування бази даних UKD-NEXT, яка описана у (рис 3.4). База даних включає кілька таблиць, таких як "users", "departments", "groups", "lesson", "schedule", "classroom" та "news", які взаємодіють між собою.

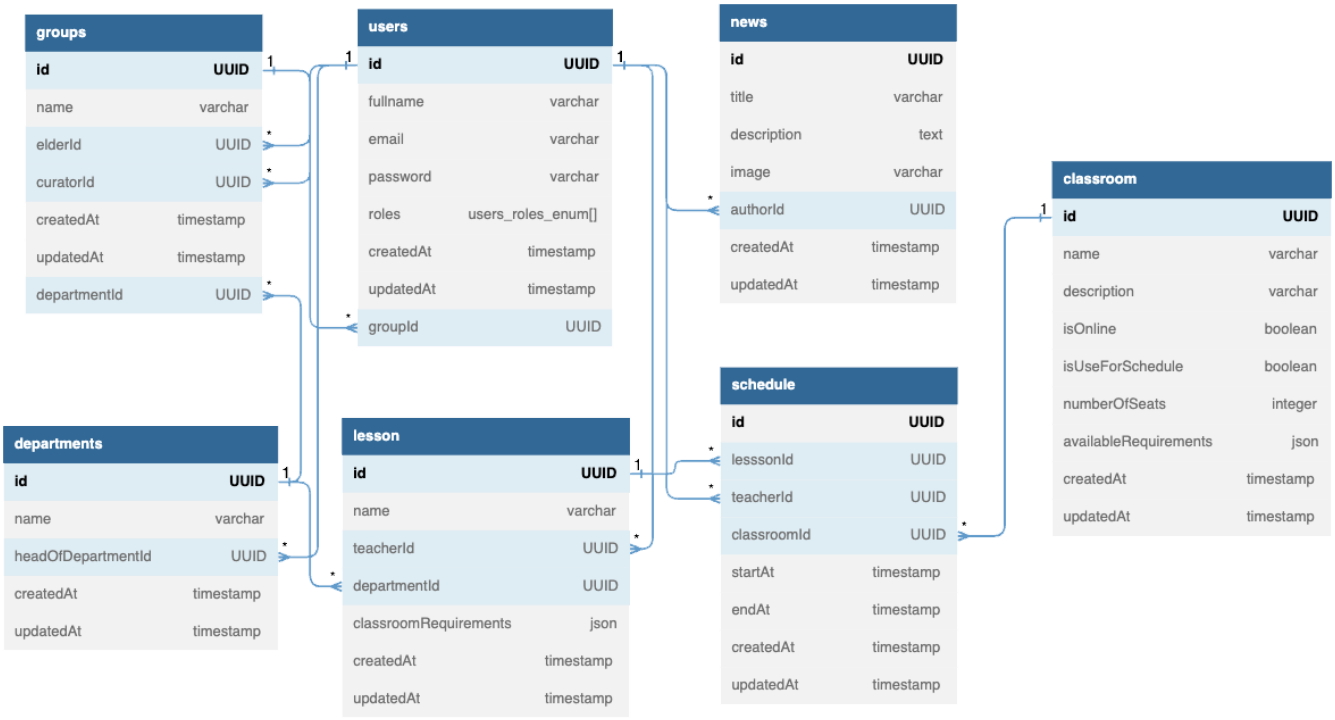

Рисунок 3.4 – Схема бази даних UKD-NEXT

Таблиця "users":

1. id: унікальний ідентифікатор у форматі UUID, не може бути порожнім, значення за замовчуванням генерується за допомогою функції uuid_generate_v4().

2. fullname: повне ім'я користувача, не може бути порожнім, значення за замовчуванням - порожній рядок.

3. email: електронна пошта користувача, не може бути порожнім.

4. password: пароль користувача, не може бути порожнім.

5. roles: ролі користувача у форматі перерахування users_roles_enum[], не може бути порожнім, значення за замовчуванням - ['STUDENT'].

6. createdAt: дата та час створення запису, не може бути порожнім, значення за замовчуванням - поточний час.

7. updatedAt: дата та час останнього оновлення запису, не може бути порожнім, значення за замовчуванням - поточний час.

8. groupId: ідентифікатор групи, до якої належить користувач (посилання на таблицю "groups").

Таблиця "departments":

1. id: унікальний ідентифікатор в форматі UUID, не може бути порожнім, значення за замовчуванням генерується за допомогою функції uuid_generate_v4().

2. name: назва відділу, не може бути порожньою.

3. headOfDepartmentId: ідентифікатор керівника відділу (посилання на таблицю "users").

4. createdAt: дата та час створення запису, не може бути порожнім, значення за замовчуванням - поточний час.

5. updatedAt: дата та час останнього оновлення запису, не може бути порожнім, значення за замовчуванням - поточний час.

Таблиця "groups":

1. id: унікальний ідентифікатор в форматі UUID, не може бути порожнім, значення за замовчуванням генерується за допомогою функції uuid_generate_v4().

2. name: назва групи, не може бути порожньою.

3. elderId: ідентифікатор старости групи (посилання на таблицю "users").

4. curatorId: ідентифікатор куратора групи (посилання на таблицю "users").

5. createdAt: дата та час створення запису, не може бути порожнім, значення за замовчуванням - поточний час.

6. updatedAt: дата та час останнього оновлення запису, не може бути порожнім, значення за замовчуванням - поточний час.

7. departmentId: ідентифікатор відділу, до якого належить група (посилання на таблицю "departments").

Таблиця "lesson":

1. id: унікальний ідентифікатор в форматі UUID, не може бути порожнім, значення за замовчуванням генерується за допомогою функції uuid_generate_v4().

2. name: назва уроку, не може бути порожньою.

3. teacherId: ідентифікатор викладача (посилання на таблицю "users").

4. departmentId: ідентифікатор відділу, до якого належить урок (посилання на таблицю "departments").

5. classroomRequirements: вимоги до класу у форматі JSON, не може бути порожнім, значення за замовчуванням - порожній об'єкт JSON.

6. createdAt: дата та час створення запису, не може бути порожнім, значення за замовчуванням - поточний час.

7. updatedAt: дата та час останнього оновлення запису, не може бути порожнім, значення за замовчуванням - поточний час.

Таблиця "schedule":

1. id: унікальний ідентифікатор в форматі UUID, не може бути порожнім, значення за замовчуванням генерується за допомогою функції uuid_generate_v4().

2. lesssonId: ідентифікатор уроку (посилання на таблицю "lesson").

3. teacherId: ідентифікатор викладача (посилання на таблицю "users").

4. classroomId: ідентифікатор класу (посилання на таблицю "classroom").

5. startAt: дата та час початку заняття, не може бути порожнім.

6. endAt: дата та час закінчення заняття, не може бути порожнім.

7. createdAt: дата та час створення запису, не може бути порожнім, значення за замовчуванням - поточний час.

8. updatedAt: дата та час останнього оновлення запису, не може бути порожнім, значення за замовчуванням - поточний час.

Таблиця "classroom":

1. id: унікальний ідентифікатор в форматі UUID, не може бути порожнім, значення за замовчуванням генерується за допомогою функції uuid_generate_v4().

2. name: назва класу, не може бути порожнім.

3. description: опис класу, не може бути порожнім, значення за замовчуванням - порожній рядок.

4. isOnline: прапорець, що вказує, чи є клас онлайн (true - онлайн, false - офлайн).

5. createdAt: дата та час створення запису, не може бути порожнім, значення за замовчуванням - поточний час.

6. updatedAt: дата та час останнього оновлення запису, не може бути порожнім, значення за замовчуванням - поточний час.

Таблиця "news":

1. id: унікальний ідентифікатор в форматі UUID, не може бути порожнім, значення за замовчуванням генерується за допомогою функції uuid_generate_v4().

2. title: заголовок новини, не може бути порожнім.

3. description: опис новини, не може бути порожнім, значення за замовчуванням - порожній рядок.

4. image: посилання на зображення новини, не може бути порожнім, значення за замовчуванням - порожній рядок.

5. authorId: ідентифікатор автора новини (посилання на таблицю "users").

6. createdAt: дата та час створення запису, не може бути порожнім, значення за замовчуванням - поточний час.

7. updatedAt: дата та час останнього оновлення запису, не може бути порожнім, значення за замовчуванням - поточний час.

3.2 Реалізація розділу розкладу

Розділ "Реалізація розділу розкладу" може включати функціонал для отримання розкладу занять на сьогоднішній день, на неділю та на весь семестр. Для цього можна розширити клас SchedulesService з коду, який ви надали. Нижче наведений приклад розширеного коду з доданими методами для отримання розкладу на потрібні періоди:

```
import { Injectable } from '@nestjs/common';
import { InjectRepository } from '@nestjs/typeorm';
import { Repository } from 'typeorm';
import { ScheduleEntity } from './schedule.entity';

@Injectable()
export class SchedulesService {
  constructor(
    @InjectRepository(ScheduleEntity)
    private readonly scheduleRepository: Repository<ScheduleEntity>,
  ) {}

  create(createScheduleDto: CreateScheduleDto) {
    return this.scheduleRepository.save(createScheduleDto);
  }

  findAll() {
    return this.scheduleRepository.find();
  }

  findOne(id: string) {
    return this.scheduleRepository.findOne({ id });
```

```
}

update(id: string, updateScheduleDto: UpdateScheduleDto) {
  return this.scheduleRepository.update(id, updateScheduleDto);
}

remove(id: string) {
  return this.scheduleRepository.delete(id);
}

findTodaySchedule() {
  const today = new Date();
  today.setHours(0, 0, 0, 0);
  const tomorrow = new Date();
  tomorrow.setDate(today.getDate() + 1);
  tomorrow.setHours(0, 0, 0, 0);
  return this.scheduleRepository.find({
    where: {
      startAt: MoreThanOrEqual(today),
      endAt: LessThan(tomorrow),
    },
  });
}

findSundaySchedule() {
  const today = new Date();
  today.setHours(0, 0, 0, 0);
  const sunday = new Date();
  sunday.setDate(today.getDate() + (7 - today.getDay()));
  sunday.setHours(0, 0, 0, 0);
  const nextSunday = new Date(sunday.getTime() + 7 * 24 * 60 * 60 * 1000);
  return this.scheduleRepository.find({
    where: {
      startAt: MoreThanOrEqual(sunday),
      endAt: LessThan(nextSunday),
    },
```

```
    });
  }

  findSemesterSchedule() {
    const semesterStartDate = new Date('2023-09-01'); // Замініть на фактичну дату початку
семестру
    const semesterEndDate = new Date('2023-12-31'); // Замініть на фактичну дату закінчення
семестру
    return this.scheduleRepository.find({
      where: {
        startAt: MoreThanOrEqual(semesterStartDate),
        endAt: LessThanOrEqual(semesterEndDate),
      },
    });
  }
}
```

В розширеному коді додані методи **findTodaySchedule()**, **findSundaySchedule()** і f**indSemesterSchedule()**, які дозволяють отримати розклад занять на сьогоднішній день, на неділю та на весь семестр відповідно. Будь ласка, зверніть увагу, що дати початку та закінчення семестру потрібно відповідно налаштувати в методі **findSemesterSchedule()** на основі фактичних дат вашого семестру.

Зверніть увагу, що в коді використовуються імпорти, такі як **CreateScheduleDto**, **UpdateScheduleDto**, **MoreThanOrEqual** та **LessThan**, які можуть залежати від вашої конкретної реалізації та використаного фреймворку. Впевніться, що ви додали відповідні імпорти перед використанням цього коду.

3.3 Реалізація розділу новин

Розділ "Реалізація розділу новин" може включати функціонал для створення, отримання, оновлення та видалення новин. Кожна новина має заголовок, опис, зображення та автора. Крім того, повинні бути доступні повні режими отримання новин: всі новини, важливі новини та беззвучні новини. Нижче наведений приклад коду з доданими класами та методами для реалізації цих функцій:

```
import { Injectable } from '@nestjs/common';
import { InjectRepository } from '@nestjs/typeorm';
import { Repository } from 'typeorm';
import { NewsEntity } from './news.entity';

@Injectable()
export class NewsService {
 constructor(
  @InjectRepository(NewsEntity)
  private readonly newsRepository: Repository<NewsEntity>,
 ) {}

 create(createNewsDto: CreateNewsDto) {
  return this.newsRepository.save(createNewsDto);
 }

 findAll() {
  return this.newsRepository.find({ relations: ['author'], order: { createdAt: 'DESC' } });
 }

 findOne(id: string) {
  return this.newsRepository.findOne({ relations: ['author'], where: { id } });
 }

 update(id: string, updateNewsDto: UpdateNewsDto) {
  return this.newsRepository.update(id, updateNewsDto);
 }

 remove(id: string) {
```

```
  return this.newsRepository.delete(id);
 }

 findImportantNews() {
  return this.newsRepository.find({ where: { isImportant: true }, relations: ['author'], order: { createdAt:
'DESC' } });
 }

 findSilentNews() {
  return this.newsRepository.find({ where: { isSilent: true }, relations: ['author'], order: { createdAt:
'DESC' } });
 }
}
```

В розширеному коді додані методи **findImportantNews()** і **findSilentNews()**, які дозволяють отримувати важливі та беззвучні новини відповідно. Код також містить методи **findAll(), findOne(), create(), update()** і **remove()**, які дозволяють виконувати звичайні операції над новинами.

Крім того, в коді також є класи **CreateNewsDto, UpdateNewsDto** і **NewsController**, які допомагають при роботі з даними новин через HTTP запити.

Зверніть увагу, що в коді використовуються імпорти, такі як **fakeRandomUuid**, **UserEntity, ApiProperty, Column, Entity, JoinColumn, OneToOne, PrimaryGeneratedColumn, CreateDateColumn, UpdateDateColumn** та інші, які можуть залежати від вашої конкретної реалізації та використаного фреймворку. Впевніться, що ви додали відповідні імпорти перед використанням цього коду.

Будь ласка, також переконайтеся, що у вас встановлені всі необхідні залежності та налаштування для коректної роботи з базою даних та HTTP запитами у вашому середовищі виконання.

3.4 Реалізація розділу Профіль

Розділ "Профіль" на основі наведеного тексту реалізований за допомогою React і містить інформацію про користувача, включаючи його фотографію, повне ім'я та групу або кафедру. Також у розділі відображаються кнопки для доступу до різних функцій.

Нижче наведений код розділу "Профіль":

```
import React from 'react';
import './ProfileScreen.scss';
import ProfileButton from '../../components/ProfileButton/ProfileButton';
export default function ProfileScreen() {
 return (
   <div className="profilescreen-container">
     <div className="profile-menu">
       <div className="profile-avatar">
         <img
src="https://cdn.discordapp.com/attachments/673322183761854484/1095018191530512424/image.p
ng" alt="" />
       </div>
       <p className="profile-fullname">Уенсдей Адамс</p>
       <p className="profile-group">ІПЗ-666</p>
       <p className="profile-email">wednesday.addams@ukd.edu.ua</p>

       <ProfileButton name="Конфіденційність" />
       <ProfileButton name="Налаштування профілю" />
       <ProfileButton name="Повідомлення" />
       <ProfileButton name="Налаштування" />
       <ProfileButton name="Про програму" />
     </div>
   </div>
 );
}
```

У даному коді створюється компонент ProfileScreen, який відображає контейнер для розділу "Профіль". Усередині контейнера знаходиться блок profile-menu, який містить наступні елементи:

1. Блок profile-avatar: відображає фотографію користувача.
2. Елемент p з класом profile-fullname: відображає повне ім'я користувача (Уенсдей Адамс).
3. Елемент p з класом profile-group: відображає групу користувача (ІПЗ-666).
4. Елемент p з класом profile-email: відображає електронну пошту користувача (wednesday.addams@ukd.edu.ua).
5. Кнопки ProfileButton для доступу до різних функцій: "Конфіденційність", "Налаштування профілю", "Повідомлення", "Налаштування", "Про програму".

Цей код можна використовувати як початковий шаблон для реалізації розділу "Профіль" у вашому проєкту. Залежно від потреб вашого проєкту, ви можете додати додаткові елементи або змінити відображення інформації про користувача.

Висновок до розділу 3

У розділі 3 була представлена програмна система UKD NEXT як ключовий компонент інформаційного сервісу університету, яка надає студентам широкий спектр зручних та корисних функціональних можливостей. Ця інноваційна система розроблена з метою повної автоматизації процесу складання розкладу занять з урахуванням різних факторів, що впливають на оптимальне планування навчальних занять.

UKD NEXT використовує розумні алгоритми, які враховують доступні аудиторії та їх специфічні особливості, індивідуальні потреби груп студентів, часовий графік та важливу інформацію про кількість предметів, які необхідно

викладати для конкретних груп на тиждень. Це допомагає забезпечити оптимальне розподілення навчальних занять, уникнути перекриття та конфліктів у розкладі.

Крім того, система UKD NEXT має інтуїтивно зрозумілий та легкий у використанні інтерфейс, що дозволяє студентам швидко та зручно отримувати доступ до своїх розкладів занять. Вона також надає студентам можливість відстежувати свій прогрес, отримувати індивідуальні поради та рекомендації щодо навчання.

UKD NEXT є незамінним інструментом для університетського співтовариства, який допомагає оптимізувати процес планування навчальних занять та забезпечує ефективну організацію навчального процесу для всіх студентів. Вона сприяє максимальному рівню задоволення студентів щодо їх навчального досвіду та покращує загальну продуктивність навчального процесу.

UKD NEXT має декілька особливостей, які роблять його корисним інструментом для студентів університету, зокрема гнучкість та оптимізацію розкладу, особистий профіль учня, доступ до новин та оголошень, інформацію про різноманітні заходи, а також відображення рейтингу студентів на основі їхніх академічних досягнень.

Загалом, UKD NEXT є потужним інструментом, який спрощує процес навчання та допомагає студентам бути організованими, інформованими та активними учасниками університетського життя. Розділ 2 детально розглянув архітектуру системи, реалізацію розділу розкладу, розділу новин та розділу профілю, надаючи важливу інформацію про функціональні можливості та особливості системи UKD NEXT.

ВИСНОВКИ

Здобувши досвід у розробці мобільних додатків, вебсайтів та бекенд серверів, можна зробити кілька висновків.

Розробка мобільних додатків та вебсайтів з додатковими функціями, такими як розклад занять, новини та інші особливості, дозволяє забезпечити зручну взаємодію з користувачами. Це підвищує рівень зручності та доступності отримання необхідної інформації.

Використання технологій, таких як React.js, Nest.js, Node.js і PostgresSQL, дозволяє розробникам ефективно створювати складні системи. React.js дозволяє розробляти інтерактивні та швидкі інтерфейси, а Nest.js і Node.js забезпечують швидку та масштабовану роботу з серверною частиною. Використання PostgresSQL дозволяє забезпечити надійне зберігання та керування даними.

Здобутий досвід у розробці складних систем свідчить про високий рівень професійної компетентності. Розробка мобільного додатку, вебсайту та бекенд сервера вимагає вміння працювати з різними технологіями та забезпечувати їх.

Розробка таких систем дозволяє відповідати на потреби користувачів, що сприяє покращенню їх задоволення та ефективності використання. Введення новин та додаткових особливостей робить систему цікавішою та корисною для користувачів. Такий підхід до розробки систем забезпечує гнучкість та адаптивність у відповіді на вимоги користувачів. Завдяки постійному вдосконаленню та розширенню функціональності, користувачі можуть насолоджуватись більш широким спектром можливостей, що пропонує система.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. React.js: це бібліотека JavaScript для побудови інтерфейсу користувача. Веб-сайт, URL: https://uk.legacy.reactjs.org/docs/getting-started.html (дата звернення: 03.06.2023).

2. PWA (Progressive Web Application): це технологія, що дозволяє створювати вебдодатки з можливостями, подібними до нативних мобільних додатків. Веб-сайт, URL: https://developer.mozilla.org/en-US/docs/Web/Progressive_web_apps (дата звернення: 03.06.2023).

3. React-router-dom: це бібліотека для маршрутизації в React додатку. Веб-сайт, URL: https://reactrouter.com/en/main/start/tutorial (дата звернення: 03.06.2023).

4. SASS: це препроцесор CSS, який надає покращені можливості для створення стилів. Веб-сайт, URL: https://sass-lang.com/documentation/ (дата звернення: 03.06.2023).

5. TypeScript: це мова програмування, що розширює JavaScript додатковими функціями типізації. Веб-сайт, URL: https://www.typescriptlang.org/docs/handbook/typescript-in-5-minutes.html (дата звернення: 03.06.2023).

6. Ant Design: це набір готових компонентів для побудови інтерфейсу користувача в React додатках. Веб-сайт, URL: https://ant.design/components/overview/ (дата звернення: 03.06.2023).

7. JavaScript: це мова програмування, яка використовується для реалізації клієнтської логіки. Веб-сайт, URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript (дата звернення: 03.06.2023).

9. Vite: це швидкий і мінімалістичний інструмент для розробки вебдодатків. Веб-сайт, URL: https://vitejs.dev/guide/ (дата звернення: 03.06.2023).

10. NX: це інструмент для розробки монорепозиторіїв з багатьма проєктами. Веб-сайт, URL: https://nx.dev/getting-started/intro (дата звернення: 03.06.2023).

12. Node.js: це середовище виконання JavaScript, засноване на двигуні V8. Веб-сайт, URL: https://nodejs.org/docs/latest-v18.x/api/ (дата звернення: 03.06.2023).

13. Nest.js: це фреймворк для побудови масштабованих серверних додатків на Node.js. Він надає структуру та конвенції для розробки, що допомагають створювати ефективні та організовані бекенд-додатки. Веб-сайт, URL: https://docs.nestjs.com/ (дата звернення: 03.06.2023).

14. PostgreSQL: це реляційна база даних, яка забезпечує надійне зберігання та керування даними. Веб-сайт, URL: https://www.postgresql.org/docs/15/index.html (дата звернення: 03.06.2023).

15. TypeOrm: це ORM (Object-Relational Mapping), яке дозволяє зв'язувати базу даних з об'єктами. Веб-сайт, URL: https://typeorm.io/ (дата звернення: 03.06.2023).

16. Lodash: це бібліотека з корисними функціями для маніпулювання даними. Вона надає різноманітні утиліти, які допомагають спростити та прискорити операції з масивами, об'єктами, рядками та іншими типами даних. Веб-сайт, URL: https://lodash.com/docs/4.17.15 (дата звернення: 03.06.2023).

17. pg: це модуль для зв'язку з PostgreSQL базою даних з використанням Node.js. Він надає простий та зручний спосіб взаємодії з базою даних з використанням JavaScript. Веб-сайт, URL: https://node-postgres.com/ (дата звернення: 03.06.2023).

18. WebSocket: це протокол для двостороннього зв'язку між клієнтом та сервером через веб-сокети. Він дозволяє реалізувати взаємодію в режимі реального часу та передавати потокові дані між клієнтом та сервером. Веб-сайт, URL: https://developer.mozilla.org/en-US/docs/Web/API/WebSockets_API (дата звернення: 03.06.2023).

19. Ant Design Pro: це набір готових компонентів для побудови інтерфейсу користувача в React додатках для адміністраторів. Він надає різноманітні готові компоненти та шаблони для створення ефективних та зручних панелей управління. Веб-сайт, URL: https://pro.ant.design/docs/getting-started/ (дата звернення: 03.06.2023).

20. Class-Validator: Технологія Class Validator є бібліотекою валідації для TypeScript та JavaScript, яка надає зручний спосіб перевірки та підтвердження правильності даних. Я використовував Class Validator для проведення валідації вхідних даних у моєму проєкті. Веб-сайт, URL: https://www.npmjs.com/package/class-validator#class-validator (дата звернення: 03.06.2023).