

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Інститут інформатики та радіoeлектроніки,
Факультет комп'ютерних наук і технологій
(повне найменування інституту, назва факультету)

Кафедра програмних засобів
(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

бакалавр

(ступінь вищої освіти)

на тему РОЗРОБКА ВЕБСАЙТУ ДЛЯ МОНІТОРИНГУ РОЗКЛАДУ
НАВЧАЛЬНОГО ПРОЦЕСУ. WEBSITE DEVELOPMENT FOR MONITORING
THE EDUCATIONAL TIMETABLE.

Виконав: студент(ка) 4 курсу, групи КНТ-117
Спеціальності 121 Інженерія програмного
забезпечення

(код і найменування спеціальності)

Освітня програма (спеціалізація) Інженерія
програмного забезпечення

Кузнєцов Т.В.

(прізвище та ініціали)

Керівник Каплієнко Т.І.

(прізвище та ініціали)

Рецензент Шитікова О.В.

(прізвище та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»
 (повне найменування закладу вищої освіти)

Інститут, факультет ІРЕ, ФКНТ

Кафедра програмних засобів

Ступінь вищої освіти бакалавр

Спеціальність 121 Інженерія програмного забезпечення
 (код і найменування)

Освітня програма (спеціалізація) Інженерія програмного забезпечення
 (назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.

С.О. Субботін

“ ” 20__ року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

Кузецова Тараса Валерійовича

(прізвище, ім'я, по батькові)

1. Тема проєкту (роботи) Розробка вебсайту для моніторингу розкладу навчального процесу. Website Development for Monitoring the Educational Timetable.

керівник проєкту (роботи) Каплієнко Тетяна Ігорівна,
 (прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої освіти від “30” березня 2021 року № 103

2. Строк подання студентом проєкту (роботи) 21 травня 2021 року

3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Вибір та обґрунтування структури системи, яка проєктується. 3. Основні рішення щодо реалізації компонентів системи 4. Керівництво програміста. 5. Керівництво оператора. 6. Тестування і результати. 7. Технічне завдання. 8. Текст програми.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-6 Основна частина	Каплієнко Т.І., доцент		
Нормоконтроль	Камінська Ж.К., асистент		

7. Дата видачі завдання 1 лютого 2021 р.**КАЛЕНДАРНИЙ ПЛАН**

/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи.	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області.	2–3 тижні	Розділ 1
3	Розробка архітектури програми.	4 тиждень	Розділ 2
4	Розробка програми.	5–6 тижні	Розділ 3
5	Тестування та експериментальне дослідження програми.	7 тиждень	Розділ 4
6	Оформлення пояснювальної записки та документів до неї. Нормоконтроль та рецензування	8 тиждень	Додатки
7	Захист роботи.	9 тиждень	

Студент

_____ Кузнєцов Т.В.
(підпис) (прізвище та ініціали)

Керівник проєкту (роботи)

_____ Каплієнко Т.І.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра:
118 с., 5 табл., 35 рис., 3 дод., 15 джерел.

МОНІТОРИНГ РОЗКЛАДУ, ВЕБЗАСТОСУНОК, JAVASCRIPT, TYPESCRIPT, РОЗКЛАД ЗАКЛАДІВ ВИЩОЇ ОСВІТИ, ОБРОБКА ДАНИХ

Об'єкт дослідження – програмні засоби моніторингу розкладу закладів вищої освіти.

Предмет дослідження – вебзастосунок для моніторингу розкладу закладів вищої освіти.

Мета роботи – створення вебзастосунку для обробки даних, необхідних для моніторингу розкладу закладів вищої освіти.

Матеріали, методи та технічні засоби: структурне та об'єктно-орієнтоване програмування, мова програмування JavaScript, мова програмування TypeScript; система для побудови клієнтських вебзастосунків, середовище розробки Visual Studio Code, персональний комп'ютер з процесором AMD Ryzen 5 під управлінням операційної системи Linux Ubuntu 20.04.

Результати. Створено програмний вебзастосунок, який виконує обробку даних, необхідних для моніторингу розкладу закладу вищої освіти.

Висновки. Розроблено вебзастосунок для моніторингу розкладу закладів вищої освіти, необхідний для зручного користування студентами та викладачами університету.

Галузь використання –заклади вищої освіти.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 118 pages, 5 tables, 35 figures, 3 appendixes, 15 sources.

SCHEDULE MONITORING, WEB APPLICATION,, JAVASCRIPT, TYPESCRIPT, SCHEDULE OF HIGHER EDUCATIONAL INSTITUTIONS, DATA PROCESSING..

Object of study is software tools for monitoring the schedule of higher education institutions.

Subject of study is web application for monitoring the schedule of higher education institutions.

The purpose of the work is to create web application of data processing required for monitoring the schedule of higher education institutions.

Materials, methods and technical means: structural and object-oriented programming, JavaScript programming language, TypeScript programming language; system for building client web applications, Visual Studio Code development environment, personal computer with AMD Ryzen 5 processor running Linux Ubuntu 20.04 operating system.

Results. A software web application has been created to process the data needed for monitoring the schedule of a higher education institution.

Conclusions.

A web application has been developed for monitoring the schedule of higher education institutions to be necessary for convenient use by students and teachers of the university.

The field of use is higher education institutions.

ЗМІСТ

	С.
Перелік скорочень та умовних познач	8
Вступ	9
1 Аналіз предметної області	10
1.1 Огляд існуючих аналогів та їх порівняльний аналіз	10
1.2 Огляд існуючих методів вирішення завдання	10
1.3 Огляд існуючих аналогів та їх порівняльний аналіз	11
1.4 Порівняльний аналіз аналогів	15
1.5 Постановка завдання до роботи	15
1.6 Підсумки аналізу існуючого стану проблеми	16
2 Вибір та обґрунтування структури системи, яка проєктується	18
2.1 Вибір інструментарію розробки програмного продукту	18
2.2 Вибір середовища проєктування клієнтської частини	23
2.3 Вибір середовища проєктування серверної частини	25
2.4 Вибір бази даних	28
2.5 Вибір середовища розробки	30
2.6 Архітектура веб застосунку	30
2.7 Вимоги до застосунку що розробляється	33
2.8 Висновки до розділу	34
3 Основні рішення щодо реалізації компонентів системи	35
3.1 Алгоритм функціонування програми	35
3.2 Основні рішення щодо реалізації компонентів системи	36
3.3 Підсумки опису програми	39
4 Керівництво програміста	40
4.1 Призначення та умови застосування програми	40
4.2 Характеристики програми	41
4.3 Звернення до програми	42
4.4 Вхідні та вихідні дані	43

4.5 Повідомлення програмісту	43
5 Керівництво оператора	45
5.1 Призначення та умови застосування програми	45
5.3 Виконання програми	46
5.4 Висновки по розділу	51
6 Тестування і результати	52
6.1 Опис методики проведення тестування	52
6.2 Результати тестування	52
Висновки	55
Перелік джерел посилань	57
Додаток А Технічне завдання	59
Додаток Б Текст програми	65
Додаток В Слайди презентації	110

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

БД	– база даних;
ЗВО	– заклад вищої освіти;
МР	– моніторингу розкладу;
НУ	– національний університет;
ПЗ	– програмне забезпечення;
ПП	– програмний продукт;
СУБД	– системи управління базами даних;
JS	– JavaScript;
SPA	– single page application;
TS	– TypeScript;

ВСТУП

Розклад навчальних занять є одним з основних документів, який регулює навчальну роботу закладу вищої освіти (ЗВО), зокрема проведення аудиторних занять і контрольних заходів.

У багатьох розвинених країнах моніторинг розкладу навчальних заходів вже давно відбувається онлайн, це значно підвищує швидкість та практичність пошуку навчальних пар як для студентів, так і для викладачів, паперові розклади вже давно забулися та перестали бути актуальними на даний момент.

Практичність моніторингу розкладу саме онлайн полягає в тому, що користувач може з легкістю знайти інформацію про свій навчальний процес, не виходячи з дому. Все що потрібно користувачу для пошуку інформації про свій розклад – це сучасний пристрій та підключення до інтернету.

На цей час найчастіше моніторинг розкладу закладів вищої освіти відбувається за допомогою вебзастосунків, а саме сайтів. Значна перевага використання саме сайту для моніторингу полягає в тому, що для самого перегляду розкладу потрібне лише підключення до інтернету, немає необхідності завантаження сторонніх програмних засобів, такого як менеджери. Можливість пошуку даних за критеріями, такими як аудиторія, група або викладач значно звужує кількість інформації для виведення на сайт, а тому дозволяє створити дуже зручний та не перенасичений користувацький інтерфейс.

Розроблене програмне забезпечення (ПЗ) має автоматизувати процес моніторингу розкладу Національного університету (НУ) «Запорізька політехніка», необхідне для зручного користування для студентів університету, а також для викладачів.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Огляд існуючих аналогів та їх порівняльний аналіз

Перед початком аналізу предметної області необхідно конкретизувати проблему, що розглядається. У нашому випадку такою проблемою слугує потреба у автоматизації процесу моніторингу розкладу закладу вищої освіти НУ «Запорізька політехніка», тобто потреба викладачів та студентів у даному засобі.

Переходячи до безпосереднього аналізу, необхідно детально розібрати процес відстеження точних даних розкладу за навчальний семестр, та зручний перегляд самого розкладу для усіх користувачів вебзастосунку. Ознайомившись з робочими практиками та методологіями роботи подібних сайтів, можна коротко сказати про наявну потребу у ПЗ, що вирішувало би наступний спектр проблем:

- зручний моніторинг розкладу для викладачів;
- зручний моніторинг розкладу для студентів;
- зручний дизайн сайту;
- адаптивний дизайн сайту для перегляду з усіх сучасних пристроїв;
- багатомовний інтерфейс вебзастосунку.

Вирішення наведених вище задач, тобто часткова автоматизація рутинних справ, дозволить значно пришвидшити пошук аудиторій та перегляд розкладу за спеціальними фільтрами.

1.2 Огляд існуючих методів вирішення завдання

Звичайним способом вирішення проблеми моніторингу розкладу у ЗВО, є написання розкладу на аркушах паперу та вивішування його на кафедрах, вже потім люди фотографують ці аркуші та викладають їх у різні групи у менеджерах. На жаль в НУ «Запорізька політехніка» досі використовують цей метод моніторингу розкладу ЗВО.

У більшості розвинених країн, в яких діджиталізація вже давно йде повним ходом вже давно існують спеціальні застосунки, вебзастосунки та боти для моніторингу розкладу у ЗВО.

Для вирішення проблеми оперативного доставлення інформації для студентів та викладачів, а також сортування цієї інформації виключно за визначними критеріями, було прийнято рішення розробки спеціального сайту для зручного моніторингу різної інформації про розклад.

1.3 Огляд існуючих аналогів та їх порівняльний аналіз

1.3.1 Система «Політех розклад»

Офіційний сайт програми «Політех розклад»:
<https://www.spbstu.ru/students/schedule/>

Даний програмний продукт (ПП), представлений на рис. 1.1, призначено для моніторингу розкладу у ЗВО [1].

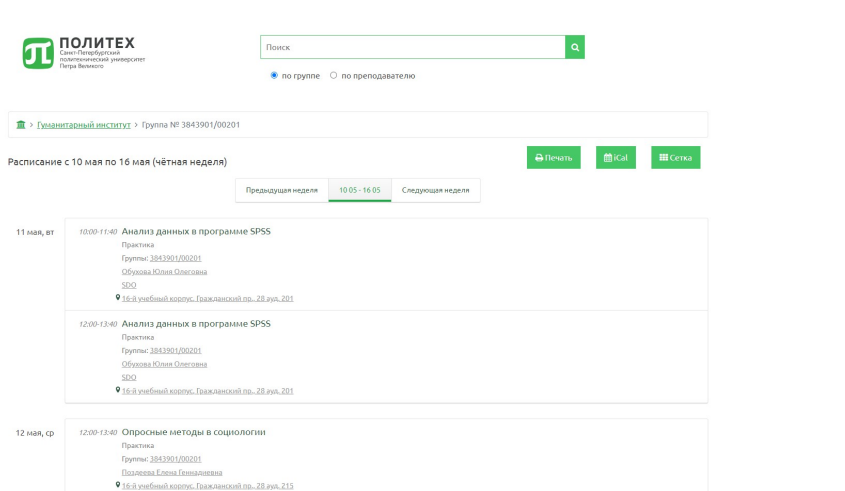


Рисунок 1.1 – Перегляд розкладу занять університету

Його базовий функціонал:

- перегляд розкладу за дану та наступну неділю;
- перегляд розкладу за групами;
- пошук розкладу за групами;

- перегляд даних за інститутами.

Переваги даного ПП:

- багатоплатформність;
- багатомовність;
- швидкий пошук за інститутами.

Його недоліки:

- незручний та перевантажений дизайн сайту;
- відсутність пошуку аудиторій за їх номером;
- відсутність перегляду розкладу наперед;
- відсутність нумерації пар.

1.3.2 Система «United States University Academic Calendar»

Офіційний сайт програми «United States University Academic Calendar»:

<https://www.usuniversity.edu/current-students/academic-calendar/>

Даний програмний продукт, представлений на рис. 1.2, призначено для моніторингу розкладу у ЗВО [2].

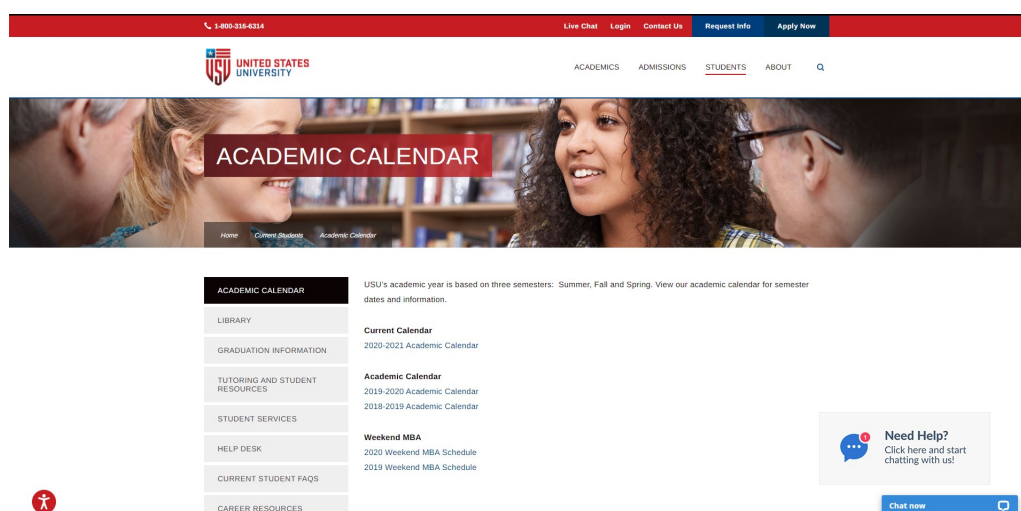


Рисунок 1.2 – Інтерфейс сайту моніторингу United States University Academic Calendar

Його базовий функціонал:

- перегляд розкладу ЗВО;
- можливість перегляду розкладу за минулі роки;

Переваги даного ПП:

- всеохопний функціонал;
- багатоплатформність;
- можливість перегляду розкладу за інститутами;
- онлайн підтримку для користувачів.

Його недоліки:

- відсутність пошуку за групою;
- відсутність пошуку за викладачем;
- можливість перегляду розкладу лише у PDF форматі;
- відсутність вибору різних мов;
- відсутність пошуку аудиторії.

1.3.3 Система «Cameron University Academic Calendar»

Офіційний сайт програми «Cameron University Academic Calendar»:

<https://www.cameron.edu/academics/calendar>

Даний ПП, представлений на рис. 1.3, призначено для моніторингу розкладу у ЗВО [3].

CAMERON UNIVERSITY		SEARCH		MENU
	(Juniors)	(Juniors)	(Juniors)	
	October 22 (Sophomores)	March 25 (Sophomores)	March 25 (Sophomores)	
	October 23 (Freshmen)	March 26 (Freshmen)	March 26 (Freshmen)	
	October 26 (Open to all students)	March 29 (Open to all students)	March 29 (Open to all students)	
Faculty Offices Open	January 7	March 25	August 9	

16 Week Fall or Spring/8 Week Summer Classes	Spring 2021	Summer 2021	Fall 2021
Classes Begin	January 11	May 27	August 16
Tuition/fees due	January 11	May 27	August 16
Martin Luther King Day Holiday	January 18	NA	NA
Memorial Day Holiday	NA	May 31	NA
Last date to enroll or add a class	January 19	June 3	August 23
Last date to drop without entry on transcript	January 25	June 3	August 27
Last date to drop with a refund	January 25	June 3	August 27
Penalty for late payment of tuition/fees begins	February 28	June 30	September 30
Independence Day Holiday	NA	July 5	NA

Рисунок 1.3 – Інтерфейс сайту моніторингу Cameron University Academic Calendar

Його базовий функціонал:

- перегляд розкладу ЗВО;
- можливість перегляду розкладу за минулі роки;
- можливість пошуку розкладу по даті;
- можливість перегляду різних університетських заходів.

Переваги даного ПП:

- всеохопний функціонал;
- багатоплатформність;
- можливість перегляду розкладу за інститутами;
- гарний та зручний інтерфейс;
- онлайн підтримку для користувачів.

Його недоліки:

- відсутність пошуку аудиторій за їх номером;
- відсутність нумерації пар;
- відсутність пошуку пар для викладача.

1.4 Порівняльний аналіз аналогів

Готові рішення, описані в попередньому розділі, було проаналізовано та порівняно за функціоналом перевагами та недоліками. Результати (табл. 1.1) засвідчують, що всі готові рішення мають як багато переваг, так і недоліків, що свідчить про важливість написання свого власного вебсайту.

Таблиця 1.1 – Результати порівняння

	Cameron Univercity Academic Calendar	United States University Academic Calendar	Политех розклад
Зручний інтерфейс	+	+	-
Можливість пошуку за викладачами	—	—	+
Можливість пошуку за групами	—	—	+
Можливість пошуку аудиторій	-	-	-
Багатомовність	-	-	+
Перегляд розкладу наперед	+	+	—

1.5 Постановка завдання до роботи

Після проведеного аналізу готових рішень проблеми автоматизації моніторингу розкладу ЗВО, було формалізовано завдання до роботи, що полягає у створенні клієнт-серверного застосунку (оскільки розроблюване програмне забезпечення має надати засоби отримання даних з централізованого сховища даних та надавати інтерфейс для роботи з нею), що відповідає наступним критеріям:

- надає можливість моніторингу розкладу за повним ім'ям викладача;
- надає можливість моніторингу розкладу за назвою групи;
- надає можливість моніторингу розкладу за назвою аудиторії;
- має зручний користувацький інтерфейс, що відповідає стандартам проектування UI/UX-інтерфейсів, а також Material Design;
- є кросплатформним: може бути використано на UNIX-подібних операційних системах, таких як Linux, MacOS, Android, iOS, а також на ОС Windows.

На вході програма повинна отримувати дані, за якими потрібно здійснити пошук, а також інформацію про викладачів, груп, аудиторій та саме їх розклад.

Оцінка успішності розробленого програмного забезпечення проводиться шляхом порівняння його на відповідність критеріям, зазначеним у пункті 1.3 пояснювальної записки.

1.6 Підсумки аналізу існуючого стану проблеми

По закінченню аналізу предметної області, а також огляду існуючих рішень проблеми можна зробити наступні висновки:

- більшість існуючого програмного забезпечення представляє собою комплексні рішення для автоматизації моніторингу розкладу ЗВО;
- існуюче програмне забезпечення не орієнтовано на перегляд розкладу ЗВО саме за певними критеріями;
- розглянуте програмне забезпечення (що є найбільш використовуваним і популярним) має застарілий дизайн і архітектуру побудови графічного інтерфейсу користувача (англ «Graphical User Interface», скорочено «GUI»).

Ознайомившись з висновками, можна підсумувати головне: створення сучасної спеціалізованої системи автоматизації моніторингу розкладу ЗВО з сучасним дизайном та набором певних параметрів для перегляду розкладу за

певними критеріями, такими як викладач, група чи аудиторія, є дуже актуальним на даний момент.

За результатами проведеного аналізу було складено технічне завдання, яке наведено в додатку А пояснювальної записки до дипломного проєкту.

2 ВИБІР ТА ОБҐРУНТУВАННЯ СТРУКТУРИ СИСТЕМИ, ЯКА ПРОЄКТУЄТЬСЯ

2.1 Вибір інструментарію розробки програмного продукту

Важливим етапом життєвого циклу моніторингової системи, і зокрема проєктування ПП, є вибір інструментарію та підключення необхідних технологій до проєкту.

Основна задача ПП – це програмна реалізація методів моніторингу розкладу ЗВО.

Розроблюваний ПП повинен обов'язково містити такі модулі:

- виведення та обробка даних;
- модуль зв'язку серверу з графічною частиною.

Для проєктування ПП були обрані такі засоби, як мова програмування, середовище розробки (Integrated Development Environment – IDE).

2.1.1 Вибір мови програмування

Для вирішення поставленої задачі, яка полягає у реалізації моніторингу розкладу ЗВО, необхідно зробити огляд сучасних мов програмування, які найкращим образом підходять для реалізації ПП. Нижче наведемо опис найбільш популярних мов програмування для розробки ПП, а також розберемо їх переваги та недоліки.

JavaScript (JS) — динамічна, об'єктно-орієнтована прототипна мова програмування. Реалізація стандарту ECMAScript. Найчастіше використовується для створення сценаріїв вебсторінок, що надає можливість на боці клієнта (пристрої кінцевого користувача) взаємодіяти з користувачем, керувати браузером, асинхронно обмінюватися даними з сервером, змінювати структуру та зовнішній вигляд веб сторінки [4].

TypeScript – мова програмування, представлена Microsoft в 2012 році і позиціонується як засіб розробки вебзастосунків, що розширює можливості JavaScript [5].

TypeScript виник через передбачувані недоліки в JavaScript в великомасштабних застосунках як в Microsoft, так і у інших користувачів JavaScript. Проблеми з розробкою складних програм на JavaScript привели до необхідності полегшення розробки компонентів мови.

Розробники TypeScript шукали рішення, яке не порушуватиме сумісність зі стандартом і його кроссплатформеною підтримкою. Знаючи, що тільки стандарт ECMAScript пропонує підтримку в майбутньому для програмування на базі класів (Class-based programming), TypeScript був заснований на цьому припущенні. Це призвело до створення компілятора JavaScript з набором синтаксичних мовних розширень, збільшеним на основі пропозиції, яке трансформує розширення в JavaScript. У цьому сенсі TypeScript є уявленням про те, що очікувати від ECMAScript. Унікальний аспект мови є в додаванні в TypeScript статичної типізації, що дозволяє статично аналізувати мову, полегшуючи оснащення і IDE підтримку.

JavaScript є об'єктно-орієнтованою мовою, але має відмінності в роботі з об'єктами в порівнянні з традиційними клас-орієнтованими мовами. Крім того, JavaScript має ряд властивостей, властивих функціональним мовам, такі як: функції як об'єкти першого класу, об'єкти як списки, каррінг, анонімні функції, замикання – що додає мові додаткову гнучкість.

Незважаючи на схожий з C синтаксис, JavaScript в порівнянні з мовою C має корінні відмінності:

- об'єкти з можливістю інтроспекції;
- функції як об'єкти першого класу;
- автоматичне приведення типів;
- автоматичне прибирання сміття;
- анонімні функції.

У мові відсутні такі корисні речі, як:

- стандартна бібліотека – зокрема, відсутній інтерфейс програмування застосунків по роботі з файловою системою, управління потоками введення-виведення, базових типів для бінарних даних;
- стандартні інтерфейси до вебсерверів і баз даних;
- система управління пакетами, яка б відстежувала залежності і автоматично встановлювала їх.

Python – високорівнева мова програмування загального призначення з динамічної строгою типізацією і автоматичним управлінням пам'яттю, орієнтований на підвищення продуктивності розробника, читання коду і його якості, а також на забезпечення переносимості написаних на ньому програм. Мова є повністю об'єктно-орієнтована – все є об'єктами. Незвичайною особливістю мови є виділення блоків коду пробільними відступами. Синтаксис ядра мови мінімалістичний, за рахунок чого на практиці рідко виникає необхідність звертатися до документації, сам же мова відома як інтерпретується і використовується в тому числі для написання скриптів. Недоліками мови є часто більш низька швидкість роботи і більш високе споживання пам'яті написаних на ньому програм в порівнянні з аналогічним кодом, написаним на компільованих мовах, таких як C або C ++.

Python є мультипарадигмальною мовою програмування, що підтримує імперативне, процедурне, структурний, об'єктно-орієнтоване програмування, метапрограмування і функціональне програмування. Завдання узагальненого програмування вирішуються за рахунок динамічної типізації. Аспектно-орієнтоване програмування частково підтримується через декоратори, більш повноцінна підтримка забезпечується додатковими фреймворками. Такі методики як контрактне і логічне програмування можна реалізувати за допомогою бібліотек або розширень. Основні архітектурні риси – динамічна типізація, автоматичне керування пам'яттю, повна інтроспекція, механізм обробки виключень, підтримка багатопоточних обчислень з глобальної блокуванням інтерпретатора (GIL), високорівневі структури даних. Підтримується розбиття програм на модулі, які, в свою чергу, можуть об'єднуватися в пакети [6].

PHP – скриптова мова загального призначення, інтенсивно застосовується для розробки вебзастосунків. В даний час підтримується переважною більшістю хостинг-провайдерів і є одним з лідерів серед мов, що застосовуються для створення динамічних вебсайтів.

В області вебпрограмування, зокрема серверної частини, PHP – один з популярних сценарних мов (разом з JSP, Perl і мовами, використовуваними в ASP.NET).

Популярність в області побудови вебсайтів визначається наявністю великого набору вбудованих засобів і додаткових модулів для розробки вебзастосунків. Основні з них:

- автоматичне вилучення POST- і GET-параметрів, а також змінних оточення вебсервера в зумовлені масиви;
- взаємодія з великою кількістю різних систем управління базами даних через додаткові модулі (MySQL, MySQLi, SQLite, PostgreSQL, Oracle Database (OCI8), Microsoft SQL Server, Sybase, ODBC, mSQL, IBM DB2, Cloudscape і Apache Derby, Informix, Ovrmos SQL, Lotus Notes, DB ++, DBM, dBase, DBX, FrontBase, FilePro, Ingres II, SESAM, Firebird і InterBase, Paradox File Access, MaxDB, інтерфейс PDO, Redis);
- автоматизована відправка HTTP-заголовків;
- робота з HTTP-авторизацією;
- робота з cookies і сесіями;
- робота з локальними і віддаленими файлами, сокетом;
- обробка файлів, що завантажуються на сервер;
- робота з Xforms.

Станом на 2010-і роки використовується сотнями тисяч розробників; згідно з рейтингом корпорації TIOBE, що базується на даних пошукових систем, в травні 2016 року PHP знаходився на 6 місці серед мов програмування [7].

Отже, кожна з вищесказаних мов підходить для реалізації даного проєкту, та володіють великою кількістю корисних можливостей, що можуть бути дуже корисними при реалізації моніторингу розкладу навчального закладу. Завдяки

вищесказаним мовам програмування здійснюється візуалізація, обробка та аналіз даних. Кожна з вищесказаних мов розвивається з кожним годом, а тому дають велику можливість швидкої реалізації того, для чого б раніше потрібно було витратити багато часу.

Під час аналізу різних мов програмування, було створено порівняльну характеристику вищесказаних мов програмування, яка наведено у таблиці 2.1.

Таблиця 2.1 – Порівняльна характеристика мов програмування

Критерії порівняння	JavaScript	TypeScript	PHP	Python
Мова створювалася спеціально для аналізу даних	+	—	—	+
Наявність динамічної типізація	+	++	+	+
Можливість статичної типізації	-	+	-	-
Наявність зрозумілої та повної документації	+	+	+	+
Можливість розробки серверної частини	+	+	+	+
Можливість розробки клієнтської частини	+	+	-	-
Продуктивність роботи	+	+	+	+ -
Можливість швидкого підключення сторонніх бібліотек	++	+ -	+	+
Наявність великої кількості наборів сторонніх пакетів	++	—	+	—
Назва пакетного менеджера	NPM	NPM	Composer	Pip
Взаємодіє з іншими мовами	+	+	+	+

Як видно з табл. 2.1 найбільші переваги мають дві мови програмування, такі як TypeScript (TS) та Javascript (JS). Мова JS має велику кількість написаних бібліотек, які можна встановити за допомогою пакетного менеджера NPM. Мова TS в свою чергу транслюється в мову програмування JS, та для зручного написання коду, за допомогою імітації статичної типізації. Мова JS підтримується

у всіх сучасних браузерах. Слід зазначити, що таку мову програмування як TS ми будемо використовувати лише на клієнтській частині, а JS на серверній частині.

Отже, підбиваючи вищесказане, для розробки ПП було обрані такі мови програмування як TS та JS.

2.2 Вибір середовища проєктування клієнтської частини

Для спрощення реалізації клієнтської частини сайту, потрібно використовувати фреймворк для швидкого та зручно створення користувацького інтерфейсу. Саме тому для вирішення задачі створення користувацького інтерфейсу для реалізації моніторингу розкладу було порівняно три популярних фреймворки, та вибрано один з них.

Vue – це прогресивний фреймворк для створення користувацьких інтерфейсів. На відміну від фреймворків-монолітів, Vue створений придатним для поступового впровадження. Його ядро в першу чергу вирішує завдання рівня уявлення (view), що спрощує інтеграцію з іншими бібліотеками та існуючими проєктами. З іншого боку, Vue повністю підходить і для створення складних односторінкових застосунків (SPA, Single-Page Applications) [8].

React – JavaScript-бібліотека з відкритим вихідним кодом для розробки призначених для користувача інтерфейсів.

React розробляється і підтримується Facebook, Instagram і співтовариством окремих розробників і корпорацій.

React може використовуватися для розробки односторінкових і мобільних застосунків. Його мета – надати високу швидкість, простоту і масштабованість. Як бібліотеки для розробки призначених для користувача інтерфейсів React часто використовується з іншими бібліотеками, такими як MobX, Redux і GraphQL [9].

Angular представляє фреймворк від компанії Google для створення клієнтських застосунків. Перш за все він націлений на розробку SPA-рішень (Single Page Application), тобто односторінкових застосунків. В цьому плані

Angular є спадкоємцем іншого фреймворка AngularJS. У той же час Angular це не нова версія AngularJS, а принципово новий фреймворк.

Angular надає таку функціональність, як двостороннє зв'язування, що дозволяє динамічно змінювати дані в одному місці інтерфейсу при зміні даних моделі в іншому, шаблони, маршрутизація і так далі [10].

Односторінковий застосунок (SPA) – це вебзастосунок або вебсайт, який взаємодіє з користувачем шляхом динамічного переписування поточної вебсторінки з новими даними з вебсервера, замість того, щоб веббраузер за замовчуванням завантажував цілі нові сторінки. Мета - швидші переходи, завдяки яким вебсайт відчуває себе більше як рідна програма.

Односторінкові програми дозволяють імітувати роботу десктоп застосунків. Архітектура влаштована таким чином, що при переході на нову сторінку, оновлюється тільки частина контенту. Таким чином, немає необхідності повторно завантажувати одні й ті ж елементи. Це дуже зручно для розробників і користувачів.

У SPA всі необхідні коди HTML, JavaScript та CSS отримуються браузером або з одним завантаженням сторінки, а відповідні ресурси динамічно завантажуються та додаються на сторінку за необхідності, як правило, у відповідь на дії користувача. Сторінка не перезавантажується в будь-який момент процесу, а також не передає управління на іншу сторінку, хоча хеш розташування або API історії HTML5 можуть бути використані для забезпечення сприйняття та навігації окремих логічних сторінок у програмі [11].

Під час аналізу різних графічних фреймворків було створено порівняльну характеристику вищесказаних фреймворків, яка наведено у таблиці 2.2.

Таблиця 2.2 – Порівняльна характеристика графічних фреймворків

Критерії порівняння	React	Vue.js	Angular
Низький поріг входження	-	+	—
Підтримка TypeScript	+	++	+
Підтримка SPA	+	+	+
Наявність зрозумілої та повної документації	+	+	+
Легке підключення графічних фреймворків	-	+	-
Легке підключення сторонніх бібліотек	+	+	-
Продуктивність роботи	+	++	+
Управління станами	+	+	+

Як видно з таблиці 2.2 найбільші переваги має клієнтський фреймворк Vue.js, бо він надає багато корисних функцій з коробки. Фреймворк Vue.js надає ряд можливостей для збільшення продуктивності і легкості використання.

Отже, підбиваючи вищесказане для розробки ПП було обрано фреймворк Vue.js.

2.3 Вибір середовища проєктування серверної частини

Для того щоб використовувати вибрану мову програмування, потрібно використовувати відповідний сервер.

Для мови JS існує розроблений для неї сервер NodeJs. Для даного серверу існує багато серверних фреймворків, таких як Express.JS, Meteor.JS, Koa.JS а також Loopback.JS.

Кожен з цих фреймворків по-своєму унікальний, відмінно підходить для вирішення певного кола завдань, використовуючи для цього мову JS. Але найбільшого поширення отримав саме Express.JS.

Express – це популярний, швидкий і гнучкий фреймворк Node.js для контролера представлення моделі, який пропонує потужний набір функцій для веброзробки і розробки мобільних застосунків. Він має приголомшливий API, який дозволяє користувачам налаштовувати різні маршрути для відправки.

Основні переваги Express.JS:

- швидка розробка застосунків;
- пряма координація зовнішнього адміністрування;
- простота вивчення.

Основні недоліки Express.JS:

- організація коду;
- логіка модулів;
- зворотні виклики, пов'язані з подіями.

Meteor – це досить простий фреймворк node.js з повним стеком для створення мобільних, а також вебзастосунків. Він сумісний з вебверсіями iOS, Android або настільними комп'ютерами.

Більш того, ця структура node.js значно спрощує розробку вебзастосунків, пропонуючи платформу для того, щоб весь рівень застосунку був на одній мові, javascript.

Meteor.js може обслуговувати великі проєкти, такі як response commerce, який є одним з найбільших і найпопулярніших проєктів електронної комерції з відкритим вихідним кодом.

Основні переваги Meteor:

- набір пакетів і бібліотек;
- постійний інструмент тестування та налагодження;
- узгоджена зв'язок клієнт-сервер.

Основні недоліки Meteor:

- відсутність рендеринга на стороні сервера;
- відсутність вбудованої підтримки прогресивних вебзастосунків;
- відсутність підтримки MongoDB;
- відсутність власної бібліотеки віджетів.

Koa.js - це новий вебфреймворк, створений розробниками Express. Він використовує асинхронні функції ES2017. Він вважається меншою, більш виразною і надійною основою для створення вебзастосунків і API.

Більш того, Коа не пов'язує якесь проміжне програмне забезпечення в ядрі, що означає, що проміжне програмне забезпечення більш оптимізовано, а кожен рядок коду досить елегантна і деталізована. Крім того, це дозволяє структурувати компоненти як заманеться.

Крім того, Коа використовує асинхронні функції для спрощення обробки помилок.

Loopback.js – це популярний фреймворк node.js, який має простий у використанні інтерфейс командного рядка і динамічний провідник API. Більш того, він дозволяє створювати моделі на основі вашої схеми.

Loopback сумісний з великою кількістю служб REST і широким спектром баз даних, включаючи MySQL, Oracle, MongoDB, Postgres тощо.

Loopback дозволяє користувачеві створювати серверний API, який зіставляється з іншим сервером. Також він підтримує власні SDK для мобільних пристроїв і браузерів для декількох клієнтів, таких як Android / Java, iOS і JavaScript для браузера (Angular) [12].

Основні переваги Loopback:

- розробка повного стека;
- дуже швидка розробка;
- вбудована функція користувачів і ролей доступу;
- код модульний і структурований.

Основні недоліки Loopback:

- важко використовувати у великих і довгострокових проєктах;
- монолітна архітектура.

Під час аналізу різних серверних фреймворків, було створено порівняльну характеристику вищесказаних фреймворків, яка наведено у таблиці 2.3.

Таблиця 2.3 – Порівняльна характеристика серверних фреймворків

Критерії порівняння	Koa	Meteor	Express	Loopback
Низький поріг входження	-	—	+	—
Підтримка TypeScript	+	-	++	-
Можливість статичної типізації	+	-	-	-
Наявність зрозумілої та повної документації	-	+	+	+
Легке підключення до бази даних	-	-	+	-
Можливість розробки клієнтської частини	+	-	+	+
Продуктивність роботи	+	+	++	+

Як видно з таблиці 2.3 найбільші переваги має серверний фреймворк Express, бо він надає багато корисних функцій з коробки. Фреймворк Express надає ряд можливостей для збільшення продуктивності і легкості використання.

Отже, підбиваючи вищесказане для розробки ПП було обрано фреймворк Express.

2.4 Вибір бази даних

Розроблюваний ПП повинен обов’язково містити такі модулі:

- зберігання даних;
- розрахунковий;
- виводу даних;
- модуль зв’язку всіх елементів ПП.

Розроблюваний ПП використовує дані розкладу викладачів та груп ЗВО, тобто комбінації уроків їх груп та викладачів. Ці дані можуть бути представлені у таких сховищах:

- текстові файли (CSV, txt, xml,...).
- бінарні файли даних.
- не реляційні бази даних (NoSQL сховища, MongoDB).
- реляційні бази даних (MySQL, MS SQL Server, Firebird, тощо);

Обраний тип зберігання повинен забезпечувати достатню гнучкість форматів, високу швидкість отримання даних, здатність зберігати великі об'єми даних, здатність до розширення функціоналу. Всім цим вимогам відповідають реляційні системи управління базами даних (СУБД). Переваги використання реляційної СУБД:

- простота і доступність для розуміння користувачем. Єдиною використовуваною інформаційною конструкцією є «таблиця» ;
- суворі правила проєктування, які базуються на математичному апараті;
- повна незалежність даних. Зміни в прикладній програмі при зміні реляційної бази даних (БД) мінімальні;
- для організації запитів і написання прикладного ПЗ немає необхідності знати конкретну організацію БД у зовнішній пам'яті.

SQLite – це найбільш використовуваний механізм баз даних у світі. SQLite вбудований у всі мобільні телефони та більшість комп'ютерів і входить до складу безлічі інших програм, якими люди користуються щодня. Формат файлу SQLite стабільний, крос-платформний і зворотно сумісний, і розробники зобов'язуються зберігати його таким чином протягом 2050 року. Файли бази даних SQLite зазвичай використовуються як контейнери для передачі багатого вмісту між системами та як довгостроковий архівний формат даних. Активно використовується понад 1 трлн баз даних SQLite.

Knex.js – це конструктор запитів SQL для Postgres, MSSQL, MySQL, MariaDB, SQLite3, Oracle та Amazon Redshift, розроблений для гнучкості, портативності та розваги у використанні. Він має як традиційні зворотні виклики у стилі вузла, так і обіцяний інтерфейс для більш чистого управління асинхронним потоком, інтерфейс потоку, повнофункціональні конструктори запитів та схем, підтримку транзакцій (з точками збереження), об'єднання з'єднань та стандартизовані відповіді між різними клієнтами запитів та діалектами [13].

Отже, підсумовуючи вищесказане у якості системи зберігання даних було обрано SQLite.

2.5 Вибір середовища розробки

Для того, щоб програмування на вибраній мові програмування було швидке та зручне було обране спеціальне середовище розробки, таке як Visual Studio Code.

IDE Visual Studio Code – це легкий, але потужний редактор вихідного коду, який працює на робочому столі та доступний для Windows, macOS та Linux. Він постачається з вбудованою підтримкою JavaScript, TypeScript та Node.js і має багату екосистему розширень для інших мов (таких як C ++, C #, Java, Python, PHP, Go) та середовищ виконання (таких як .NET та Unity) [14]. VScode – це редактор вихідного коду. Він підтримує ряд мов програмування, виділення синтаксису, IntelliSense, рефакторинг, налагодження, навігацію кодом, підтримку Git та інші функції. Багато функцій Visual Studio Code недоступні через графічний інтерфейс і часто використовуються за допомогою палітри команд або файлів JSON (наприклад, спеціальних налаштувань). Палітра команд схожа на командний рядок, який викликається комбінацією клавіш.

2.6 Архітектура веб застосунку

В якості архітектури веб застосунку була обрана SPA архітектура.

SPA – це збірна назва набору технологій, що дозволяють реалізувати WEB-застосунок, який виконувався WEB-браузером як одна WEB-сторінка, як, наприклад, реалізований сервіс Gmail від Google. З точки зору користувача, дана технологія залучає в першу чергу швидкістю відгуку на дії в інтерфейсі, так як не потрібно повної або навіть часткової перезавантаження WEB-сторінки з сервера, а всі візуальні елементи конструюються прямо в браузері за допомогою JavaScript шляхом маніпуляцій з DOM – структурою документа.

Таким чином, WEB-застосунки стають дуже схожі на звичайні програми для робочих станцій, що завантажують інформацію з мережі Інтернет, тільки середовищем виконання для них є не операційна система, а браузер, який в

результаті змушений нести на собі всю навантаження, пов'язану з виконанням стороннього коду, а саме управління пам'яттю, забезпечення безпечного оточення, надання функціоналу для роботи з системними функціями і апаратним оточенням.

Центральне місце SPA-архітектури займає уявлення (View) – то, що бачить і з чим взаємодіє користувач. Результатом роботи уявлення є самий звичайний HTML, що відображається браузером. На відміну від «перехідних» «WEB 2.0»-застосунків, що активно працюють з DOM-структурою документа, наприклад, за допомогою jQuery або underscore, SPA-застосунок використовує DOM тільки для запису змін, але не для читання, тобто не для зберігання даних. Для зберігання даних тепер використовується ще один компонент SPA-архітектури – модель (Model).

Модель являє собою сукупність даних, функцій для маніпуляції з даними і подіями. Всі дані моделі повністю зберігаються в пам'яті. Для того, щоб дані, що знаходяться в моделі, і дані, які відображаються поданням, зберігали цілісність, уявлення підписується на події моделі, відстежуючи таким чином зміни даних в моделі. У свою чергу, модель також реагує на повідомлення уявлення і забезпечує нерозривний зв'язок WEB-застосунки з сервером, виконуючи запити для отримання або відправки даних (зокрема із застосуванням методології REST).

Принцип роботи SPA фреймворку VueJs показано на рисунку 2.1.

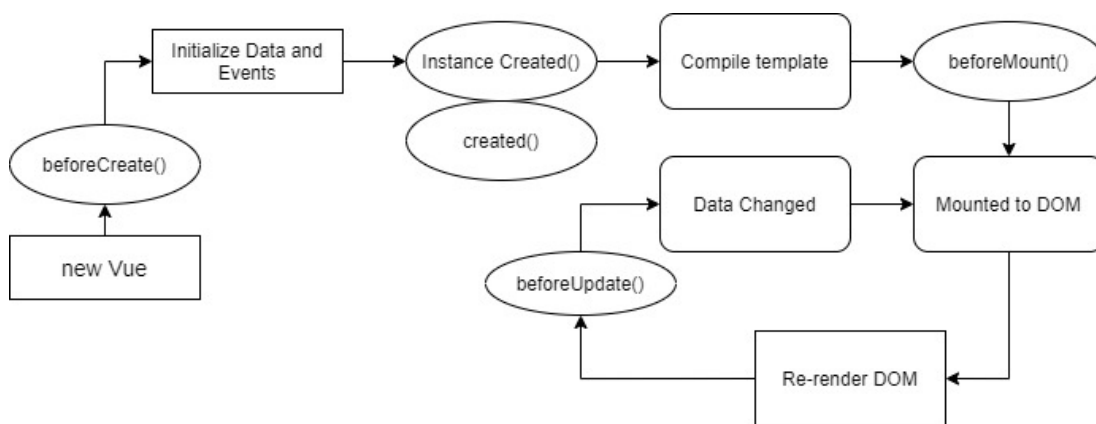


Рисунок 2.1 – Принцип роботи SPA фреймворку Vue.js

Нижче наведені хуки життєвого циклу фреймворку Vue, та їх опис:

- спочатку відбувається ініціалізація самого фреймворку за допомогою `new Vue()`;
- `BeforeCreate` – викликається синхронно відразу після ініціалізації екземпляра, перед спостереженням даних та подією, далі ініціалізуються всі події та дані;
- `Created` – викликається синхронно після створення екземпляра. На цьому етапі екземпляр закінчив обробку параметрів, а це означає, що було налаштовано наступне: спостереження даних, обчислювані властивості, методи, зворотні виклики спостереження / події. Однак фаза монтажу ще не розпочата, і властивість `$el` ще не буде доступною;
- `BeforeMount` – Визивається безпосередньо перед початком рендерінгу: функція рендерінгу буде викликана в перший раз;
- `Mounted` – Викликається після монтування екземпляра, де елемент, переданий `app.mount`, замінюється на щойно створений `vm`. `$El`. Якщо кореневий екземпляр змонтований до елемента в документі, `vm`. `$El` також буде в документі при виклику монтованого.

Після цього при змінні даних відбувається пререндер зміненого компонента.

У якості архітектури застосунку було обрано REST API.

REST (від англ. Representational State Transfer – «передача стану представлення») – архітектурний стиль взаємодії компонентів розподіленого застосунку в мережах. REST представляє собою узгоджений набір обмежених, здійснених при проєктуванні розподіленої гіпермедіа-системи. У визначених випадках (інтернет-магазини, пошукові системи, інші системи, ґрунтовані на даних) це призводить до підвищення продуктивності та узагальнення архітектури. У широкому змісті компоненти в REST взаємодіють, використовуючи взаємодії клієнтів та серверів у всесвітній павутині [15].

У мережах Інтернет виклик видаленій процедурі може бути представлений звичайний HTTP-запит (зазвичай GET або POST; такий запит називається «REST-запит»), необхідні дані передаються у якості параметрів запиту.

Для вебслужби, побудованої на REST застосовується термін «RESTful».

Принцип роботи REST показано на рисунку 2.2.

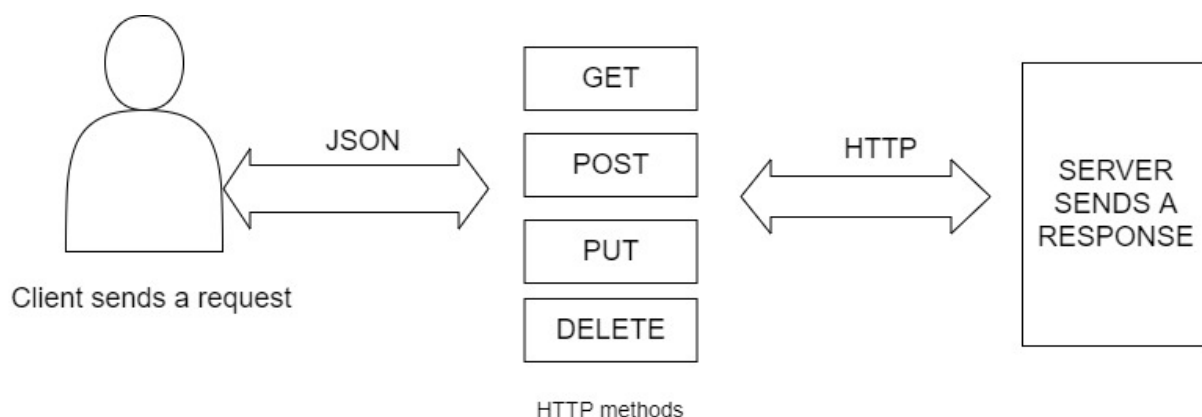


Рисунок 2.2 – Принцип роботи REST API застосунку

2.7 Вимоги до застосунку що розробляється

Даний програмний продукт повинен мати такий функціонал, як:

- пошук аудиторії за її номером;
- пошук розкладу викладача за його повним ім'ям;
- сортування викладачів за факультетом та кафедрою;
- пошук розкладу групи за її номером;
- сортування груп за курсом;
- перегляд розкладу дзвоників;
- перегляд розкладу викладачів за знаменником;
- перегляд розкладу викладачів за чисельником;
- перегляд розкладу груп за знаменником;
- перегляд розкладу груп за чисельником;
- інтерфейс застосунку повинен бути розроблений англійською та українською мовами;

- перегляд розкладу викладачів за знаменником та чисельником водночас;
- перегляд розкладу груп за знаменником та чисельником водночас.

2.8 Висновки до розділу

На етапі вибору та обґрунтування системи що проєктуються, було зроблено такі висновки:

- мовами програмування веб застосунку була обрано TypeScript та JavaScript;
- клієнтська частина проєкту буде розроблена за допомогою фреймворку Vue.js використовуючи мову програмування TypeScript;
- серверна частина проєкту буде розроблена за допомогою фреймворку Express.js використовуючи мову програмування JavaScript;
- у якості архітектури застосунку було обрано REST API;
- у якості системи зберігання даних було обрано SQLite, для швидкої та зручної роботи з базою даних була обрана бібліотека Knex;
- у якості середовища розробки, буде використовуватись Visual studio code.

3 ОСНОВНІ РІШЕННЯ ЩОДО РЕАЛІЗАЦІЇ КОМПОНЕНТІВ СИСТЕМИ

В третьому розділі наводиться опис алгоритму функціонування програми, а також обґрунтування прийнятих в ході розробки рішень щодо реалізації компонентів системи, визначених на етапі проєктування.

3.1 Алгоритм функціонування програми

Порядок прийому та обробки запитів до серверу регламентується стандартом WSGI (з англ. «Web Server Gateway Interface»), розробленого з метою опису взаємодії вебзастосунків, написаних на мові програмування Javascript. Схема роботи WSGI-застосунків зображена на рисунку 3.2.

Як було визначено ще на етапі проєктування, структурно програма складається з двох окремих модулів: клієнтської і серверної частини. Обмін даними і взаємодія цих компонентів реалізується за рахунок використання протоколу передачі даних HTTP, призначений для надсилання запитів клієнтом і відповідей на ці запити сервером.

Структура взаємодії між компонентами системи зображена на рисунку 3.1.

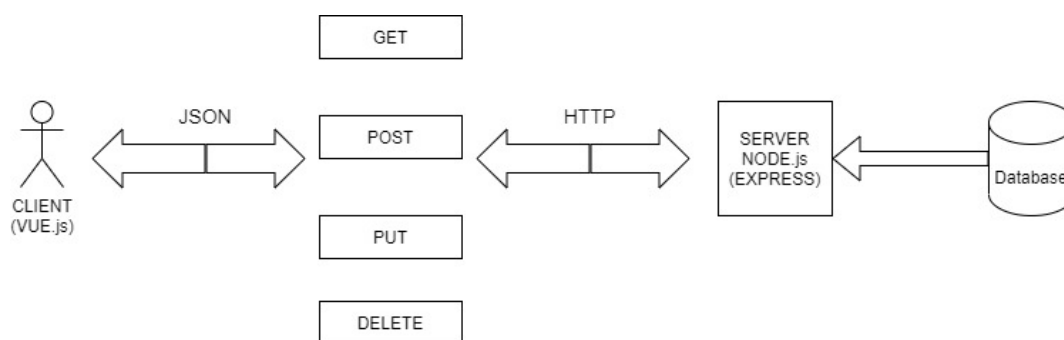


Рисунок 3.1 – Структура взаємодії компонентів системи

Відправляючи запит, клієнт початково буде отримувати згенеровані клієнтським застосунком HTML-сторінки (а також файли стилів .css та файли коду .js, що відносяться до цієї сторінки). Потім, з використанням Fetch API (браузерного програмного інтерфейсу для відправки HTTP-запитів) мови

програмування JavaScript, клієнт буде надсилати запити до серверу на отримання чи обробку необхідних даних (інформація, що зберігається у базі даних і необхідна для заповнення поточної сторінки, або дані, відправлені на сервер для обробки, подальшого зберігання тощо).

Така практика є широко розповсюдженою при побудові клієнт-серверних застосунків.

Потрапивши до Javascript-застосунку запит спочатку обробляється проміжними шаром (англ. «Middleware»), що видозмінює запити перед їх потраплянням до відповідного контролеру і відповіді, після його роботи. Необхідний контролер обирається в залежності від того, до якої адреси був направлений запит. Контролер, використовуючи інтерфейс моделі для роботи з даними і реалізації бізнес-логіки, тобто правил обробки інформації, відправляє відповідь клієнту, в нашому випадку, в текстовому форматі JSON з вказанням коду статусу HTTP (наприклад, 200 у випадку вдалого запиту, 400 – у випадку некоректно наданих даних, 403 через заборону звернення і так далі). Алгоритм взаємодії окремих модулів серверної частини (представлення, контролерів, моделі) зображено на рисунку 2.2.

Для приведення отриманих даних до Javascript-об'єктів, а також структурування відповідей до формату JSON у NodeJs використовуються серіалізатори (англ. «serializers»). Вони також відповідають за початкову валідацію даних, їх відповідність висунутим вимогам.

3.2 Основні рішення щодо реалізації компонентів системи

3.2.1 Основні рішення щодо реалізації серверної частини

NodeJs надає можливість розбиття застосунків на окремі структурні частини, та реалізують у собі архітектуру SPA. Нижче наведено перелік виділених частин:

- application.js – файл з набором інструкцій для серверу NodeJs;

- `auditoriesController` – файл з набором маршрутизації для усіх маршрутів, зв'язаних з роботою з аудиторіями;
- `studentController.js` – файл з набором маршрутизації для усіх маршрутів, зв'язаних з роботою з групами;
- `teacherController.js` – файл з набором маршрутизації для усіх маршрутів, зв'язаних з роботою з викладачами;
- `knexfile.js` – файл з набором інструкцій для підключення до бази даних;
- «`package.json`» – перелік встановлених залежностей, а також режимів виконання застосунку;
- «`node_modules/`» – директорія для зберігання файлів залежностей (сторонніх бібліотек, фреймворків тощо);

Для обробки запитів на виконання CRUD операцій до серверу, реалізована модель заявок за відносним посиланням `/controllers/`, наступна:

- при відправлені GET запиту за посиланням `/api/auditories/getAuditories` будуть отримана усі аудиторії з бази даних, та відправлення відповіді у JSON форматі до серверу;
- при відправлені GET запиту за посиланням `/api/auditories/getAuditoryInfo` буде отримана уся інформація про конкретну аудиторію за її полем ID у базі даних, та відправлення відповіді у JSON форматі до серверу;
- при відправлені GET запиту за посиланням `/api/students/getStudyGroups` будуть отримані усі навчальні групи з бази даних, відправлення відповіді у JSON форматі до серверу;
- при відправлені GET запиту за посиланням `/api/auditories/getStudyLessonInfo` буде отримана уся інформація про конкретну навчальну групу за її полем ID у базі даних, та відправлена у JSON форматі до серверу;
- при відправлені GET запиту за посиланням `/api/teachers/getTeachers` будуть отримані усі викладачі з бази даних, відправлення відповіді у JSON форматі до серверу;

– при відправленні GET запиту за посиланням `/api/teachers/getTeacherLessonInfo` буде отримана уся інформація про конкретного викладача за його полем ID у базі даних відправлення відповіді у JSON форматі до серверу.

3.2.2 Основні рішення щодо реалізації клієнтської частини

Звичайний Vue-застосунок має наступну структуру файлів і директорій:

- «`package.json`» – перелік встановлених залежностей, а також режимів виконання застосунку
- «`node_modules/`» – директорія для зберігання файлів залежностей (сторонніх бібліотек, фреймворків тощо);
- «`dist/`» – зібраний (скомпільований) проєкт у готовому вигляді, файли, що змінюються під час зборки знаходяться у директорії `dist/`;
- «`src/`» – директорія з кодом програми, саме тут знаходяться файли `main.js` та `App`, що є точкою входу у програму;
- «`components/`» – загальні компоненти системи;
- «`views/`» – директорій з усіма сторінками проєкту;
- «`router/`» – сторінки сайту та компоненти, що відносяться до нього;
- «`enums/`» – директорія з усіма константами у проєкті;
- «`locales/`» - директорія з усіма локалізаціями проєкту;
- «`plugins/`» - директорія з усіма плагінами системи;
- «`models/`» - директорія з усіма моделями системи, призначенні для роботи з даними після прийняття їх з БД.

Для верстки (створення дизайну) сайту було використано графічний-фреймворк `Vuetify js`, що включає в себе багато готових компонентів (таких, як спливаючі повідомлення, діалогові вікна, готові компоненти форм тощо), надає можливості 12-колонної розмітки сайту та створення адаптивного інтерфейсу, як того вимагає завдання до роботи.

Для роботи з певними запитами з клієнтської частини на серверну, було використано `axios`. Цей модуль дозволяє віддавати з клієнтської частини запити (такі як `POST` чи `GET`), а також приймати чи віддавати данні.

3.3 Підсумки опису програми

В ході виконання етапу реалізації програми було реалізовано визначені раніше компоненти програми. Для спрощення процесу розробки, були використані наявні готові рішення у вигляді модулів для роботи з `NodeJs` та `Vue`, що описані у пункті 3.2.

По завершенню етапу реалізації отримано програмне забезпечення, що відповідає всім функціональним і не функціональним вимогам, що висуваються до нього згідно зі складеним раніше технічним завданням. Розроблений програмний продукт відповідає наявним сьогодні тенденціям розробки вебзастосунків і може бути легко модифікований в майбутньому.

4 КЕРІВНИЦТВО ПРОГРАМІСТА

4.1 Призначення та умови застосування програми

4.1.1 Призначення та функції

Дане ПЗ призначене для автоматизації та прискорення моніторингу розкладу ЗВО.

Функції програмного забезпечення, які розроблені у даному ПЗ:

- пошук аудиторії за її номером;
- пошук розкладу викладача за його повним ім'ям;
- сортування викладачів за факультетом та кафедрою;
- пошук розкладу групи за її номером;
- сортування груп за курсом;
- перегляд розкладу дзвоників;
- перегляд розкладу викладачів за знаменником;
- перегляд розкладу викладачів за чисельником;
- перегляд розкладу груп за знаменником;
- перегляд розкладу груп за чисельником;
- інтерфейс застосунку розроблений англійською та українською мовами;
- перегляд розкладу викладачів за знаменником та чисельником водночас;
- перегляд розкладу груп за знаменником та чисельником водночас.

4.1.2 Умови виконання програми

Для того, щоб виконання програмного засобу було коректне, необхідні такі параметри системи а також такі периферійні пристрої як:

- однопоточний процесор з тактовою частотою 2.4 ГГц. Та більше;
- обсяг оперативної пам'яті не менше ніж 512Мб;
- жорсткий диск не менш ніж з 50 Гб. пам'яті;

- блок живлення не менш ніж 240 ватт;
- Windows 7 та вище.

4.2 Характеристики програми

Система розрахована на цілодобовий режим роботи протягом 365 днів на рік. Засобами контролю моніторингу є автоматизовані вбудовані в платформу програмні компоненти, які самостійно виконують такі дії в разі помилки:

- якщо відсутнє підключення до бази даних, то система видає помилку адміністратору у консолі серверу, про те що підключення відсутнє;
- якщо виникла якась системна помилка, то система видає помилку адміністратору у консолі серверу, про те яка сама помилка виникла.

Для відновлення програми потрібно перезапустити сервер та запустити програму через консоль серверу за допомогою команди «npm run start».

Обсяг пам'яті, якій займає програма – 270 Мб.

Програма написана на мовах програмування JavaScript та TypeScript, за допомогою серверу NodeJs.

Перелік файлів з яких складається програма:

- application.js – файл з набором інструкцій для серверу NodeJs;
- auditoriesController – файл з набором маршрутизації для усіх маршрутів, зв'язаних з роботою з аудиторіями;
- studentController.js – файл з набором маршрутизації для усіх маршрутів, зв'язаних з роботою з групами;
- teacherController.js – файл з набором маршрутизації для усіх маршрутів, зв'язаних з роботою з викладачами;
- knexfile.js – файл з набором інструкцій для підключення до бази даних;
- «package.json» – перелік встановлених залежностей, а також режимів виконання за стосунку;

- «node_modules/» – директорія для зберігання файлів залежностей (сторонніх бібліотек, фреймворків тощо);
- «public/» – зібраний (скомпільований) проєкт у готовому вигляді, файли, що змінюються під час зборки знаходяться у директорії public/build/;
- «src/» – директорія з кодом програми, саме тут знаходяться файли main.js та App, що є точкою входу у програму;
- «components/» – загальні компоненти системи;
- «views/» – директорій з усіма сторінками проєкту;
- «router/» – сторінки сайту та компоненти, що відносяться до нього;
- «enums/» – директорія з усіма константами у проєкті;
- «locales/» - директорія з усіма локалізаціями проєкту;
- «plugins/» - директорія з усіма плагінами системи;
- «models/» - директорія з усіма моделями системи, призначенні для роботи з даними після прийняття їх з бази даних.

4.3 Звернення до програми

Управління вмістом системи відбувається лише через завантажену програму. Для того, щоб передати повне управління над програмою потрібно закатити папку з програмою, та сервер NodeJs, а з ним буде встановлений npm, після цього треба встановити деякі модулі, такі як:

- express – модель для роботи серверної частини проєкту [8];
- knex – модуль для простої роботи з базою даних;
- mysql2 – модуль для підключення до бази даних;
- axios – модуль для роботи з HTTP запитами;
- vue – модуль для роботи з клієнтським фреймворком;
- vue-i18n – модуль для роботи з багатомовністю;
- vue-meta – модуль для роботи з заголовками вебсторінки;
- vue-router – модуль для роботи з клієнтською маршрутизацією;

- vuetify – графічний фреймворк вебзастосунку.

Після цього для запуску програми треба вписати у консоль «npm run start».

4.4 Вхідні та вихідні дані

Вхідними даними програми є вже готовий файл з базою даних, а також розклад дзвоників у json форматі.

Вихідними даними програми є результат згенерованого розкладу для викладачів за їх повним ім'ям, та студентів за їх групою, а також результат пошуку аудиторії за її номером.

4.5 Повідомлення програмісту

Усі повідомлення, що може отримати програміст представлений нижче:

- відсутність доступу до бази даних – означає, що програма не може підключитися до бази даних серверу з даної IP адреси (рис. 4.1);
- відсутність підключення до бази даних – означає, що програма не може підключитися до вказаної бази даних (рис. 4.2).
- сервер запущено на порті – означає, що програма запущена успішно (рис. 4.3).

Приклад системних помилок показано на рисунку 4.1 – 4.2.

```
Error: Access denied for user 'zp'@'localhost' (using password: YES)
    at Packet.asError (/home/taras/javascrrip/rozklad-diplom/node_modules/mysql2/lib/packets/packet.js:712:17)
    at ClientHandshake.execute (/home/taras/javascrrip/rozklad-diplom/node_modules/mysql2/lib/commands/command.js:28:26)
    at Connection.handlePacket (/home/taras/javascrrip/rozklad-diplom/node_modules/mysql2/lib/connection.js:425:32)
    at PacketParser.onPacket (/home/taras/javascrrip/rozklad-diplom/node_modules/mysql2/lib/connection.js:75:12)
    at PacketParser.executeStart (/home/taras/javascrrip/rozklad-diplom/node_modules/mysql2/lib/packet_parser.js:75:16)
    at Socket.<anonymous> (/home/taras/javascrrip/rozklad-diplom/node_modules/mysql2/lib/connection.js:82:25)
    at Socket.emit (node:events:378:20)
    at addChunk (node:internal/streams/readable:313:12)
    at readableAddChunk (node:internal/streams/readable:288:9)
    at Socket.Readable.push (node:internal/streams/readable:227:10) {
  code: 'ER_ACCESS_DENIED_ERROR',
  errno: 1045,
  sqlState: '28000',
  sqlMessage: "Access denied for user 'zp'@'localhost' (using password: YES)"
}
Відсутність доступу до бази даних!
```

Рисунок 4.1 – Приклад системних помилок під час підключення до бази даних

```
Error: connect ECONNREFUSED 127.0.0.1:3306
    at TCPConnectWrap.afterConnect [as oncomplete] (node:net:1137:16) {
  errno: -111,
  code: 'ECONNREFUSED',
  syscall: 'connect',
  address: '127.0.0.1',
  port: 3306,
  fatal: true
}
Відсутність підключення до бази даних!
```

Рисунок 4.2 – Приклад системних помилок під час підключення до бази даних

```
[node@centos7 ~]$ node server.js
сервер запущено на порті: 3000
```

Рисунок 4.3 – Приклад системних повідомлень під час успішного запуску програми

5 КЕРІВНИЦТВО ОПЕРАТОРА

5.1 Призначення та умови застосування програми

5.1.1 Призначення та функції

Дане ПЗ призначене для автоматизації та прискорення моніторингу розкладу ЗВО.

Функції програмного забезпечення, які розроблені у даному ПЗ:

- пошук аудиторії за її номером;
- пошук розкладу викладача за його повним ім'ям;
- сортування викладачів за факультетом та кафедрою;
- пошук розкладу групи за її номером;
- сортування груп за курсом;
- перегляд розкладу дзвоників;
- перегляд розкладу викладачів за знаменником;
- перегляд розкладу викладачів за чисельником;
- перегляд розкладу груп за знаменником;
- перегляд розкладу груп за чисельником;
- інтерфейс застосунку розроблений англійською та українською мовами;
- перегляд розкладу викладачів за знаменником та чисельником водночас;
- перегляд розкладу груп за знаменником та чисельником водночас.

5.1.2 Умови виконання програми

Для того, щоб виконання програмного засобу було коректне, необхідні такі параметри системи а також такі периферійні пристрої як:

- одноядерний процесор з тактовою частотою 2.4 ГГц. Та більше;
- обсяг оперативної пам'яті не менше ніж 512Мб;

- жорсткий диск не менш ніж з 50 Гб. пам'яті;
- блок живлення не менш ніж 240 ватт;
- Windows 7 та вище.

5.3 Виконання програми

Для запуску програмного забезпечення необхідно відкрити головний файл програми «diplom» та вписати в консоль «npm run start && cd/rozkład npm run serve».

Для нормальної роботи програми у оператора повинно бути встановлений браузер з найновішою версією Javascript.

Звертання до програми відбувається у той момент, коли користувач перейшов на любую сторінку сайту.

Для використання функцій програми оператор повинен перейти на головну сторінку сайту «<http://node.zntureiting.pp.ua/>», у нього з'являється сторінка пошуку розкладу для викладачів (приклад наведено на рис. 5.1).

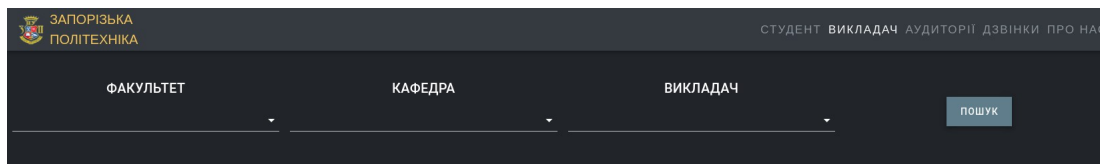


Рисунок 5.1 – Сторінка пошуку розкладу для викладачів

Для моніторингу розкладу для викладача оператор повинен вибрати повне ім'я викладача, та після цього натиснути кнопку пошук. Після цього оператора перекине на сторінку розкладу викладача (приклад наведено на рис. 5.2).



ЗАПОРІЗЬКА

ПОЛІТЕХНІКА

СТУДЕНТ

ВИКЛАДАЧ

АУДИТОРІЇ

ДЗВІНКИ

ПРО НАС

ЗАГАЛЬНЕ

ЧИСЕЛЬНИК

ЗНАМЕННИК

Афендікова Наталія Олександрівна

Понеділок

Пара	Дисципліна	Аудиторія	Група	Тип заняття
1(3)	Економічна теорія	533 539a	M-317 M-318сн M-328сн M-338сн	практ.роб.
4	Економічна теорія	533 539a	M-317 M-318сн M-328сн M-338сн	практ.роб.

Вівторок

Пара	Дисципліна	Аудиторія	Група	Тип заняття
3	Економічна теорія	533 539a	M-317 M-318сн M-328сн M-338сн	практ.роб.

Середа

Пара	Дисципліна	Аудиторія	Група	Тип заняття
2(3)	Економічна теорія	533 539a	M-317 M-318сн M-328сн M-338сн	практ.роб.
4(4)	Економічна теорія	533 539a	M-317 M-318сн M-328сн M-338сн	практ.роб.



2021

©Усі права захищено

Рисунок 5.2 – Сторінка розкладу для викладачів

Для моніторингу розкладу за групою оператор повинен перейти на сторінку “студент” (приклад наведено на рис. 5.3) та вибрати назву групи, після цього натиснути кнопку пошук. Після цього оператора перекине на сторінку розкладу групи (приклад наведено на рис. 5.4).

ЗАПОРІЗЬКА ПОЛІТЕХНІКА

СТУДЕНТ ВИКЛАДАЧ АУДИТОРІЇ ДЗВІНКИ ПРО НАС

КУРС

ГРУПА

ПОШУК

2021 © Усі права захищено

Рисунок 5.3 – Сторінка пошуку розкладу за групою

ЗАПОРІЗЬКА ПОЛІТЕХНІКА

СТУДЕНТ ВИКЛАДАЧ АУДИТОРІЇ ДЗВІНКИ ПРО НАС

ЗАГАЛЬНЕ ЧИСЕЛЬНИК ЗНАМЕННИК

М-338сп

Понеділок

Пара	Дисципліна	Аудиторія	Викладач	Тип заняття
1(3)	Економічна теорія	533 539a	Афендікова Наталія Олександрівна	практ.роб.
2	Вантажопідіймна, транспортуюча та транспортна техніка	11 11a	Лятуринський Василь Олександрович	лаб.роб.
3	Експлуатація та обслуговування машин	543 11	Глушко Василь Іванович	лаб.роб.
4	Економічна теорія	533 539a	Афендікова Наталія Олександрівна	практ.роб.

Вівторок

Пара	Дисципліна	Аудиторія	Викладач	Тип заняття
3	Економічна теорія	533 539a	Афендікова Наталія Олександрівна	практ.роб.
5	Вантажопідіймна, транспортуюча та транспортна техніка	11 11a	Лятуринський Василь Олександрович	лаб.роб.

Середа

Пара	Дисципліна	Аудиторія	Викладач	Тип заняття
2	Експлуатація та обслуговування машин	543 11	Глушко Василь Іванович	лаб.роб.
2(3)	Економічна теорія	533 539a	Афендікова Наталія Олександрівна	практ.роб.
4(4)	Економічна теорія	533 539a	Афендікова Наталія Олександрівна	практ.роб.

Четвер

2021 © Усі права захищено

Рисунок 5.4 – Сторінка розкладу для групи

Для моніторингу даних за аудиторією оператор повинен перейти на сторінку “аудиторії” (приклад наведено на рис. 5.5) та вибрати назву аудиторії, після цього натиснути кнопку пошук. Після цього оператора перекине на сторінку з описом аудиторії (приклад наведено на рис. 5.6).

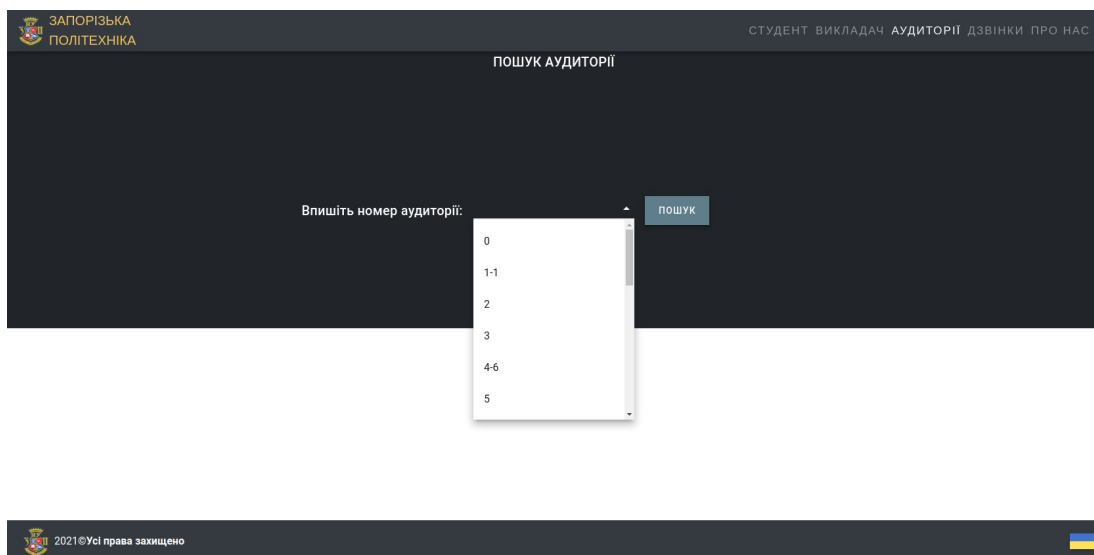


Рисунок 5.5 – Сторінка пошуку даних за номером аудиторії

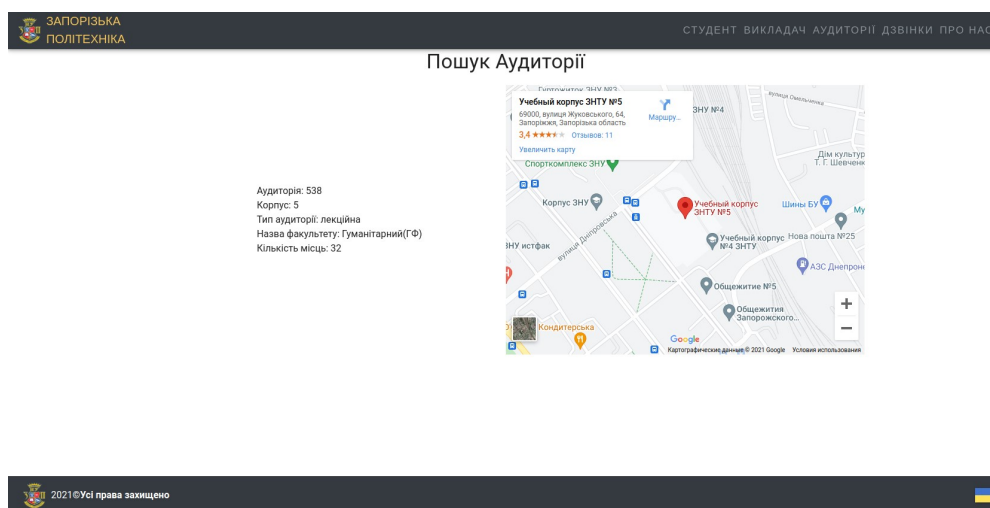


Рисунок 5.6 – Сторінка інформації про аудиторію

Для моніторингу даним про розклад дзвоників оператор повинен перейти на сторінку “дзвінки” (приклад наведено на рис. 5.7).



ЗАПОРІЗЬКА

ПОЛІТЕХНІКА

СТУДЕНТ

ВИКЛАДАЧ

АУДИТОРІЇ

ДЗВІНКИ

ПРО НАС

Розклад дзвінків

Пара	Початок	Кінець	Перерва
I	8:30	9:50	15
II	10:05	11:25	10
III	11:55	13:15	10
IV	13:25	14:45	10
I	14:55	16:15	30
VI	16:45	18:05	10
VII	18:15	19:35	10
VIII	19:45	21:05	-



Рисунок 5.7 – Сторінка інформації про розклад дзвінків

Для перегляду інформації про сайт оператор повинен перейти на сторінку “про нас” (приклад наведено на рис. 5.8).

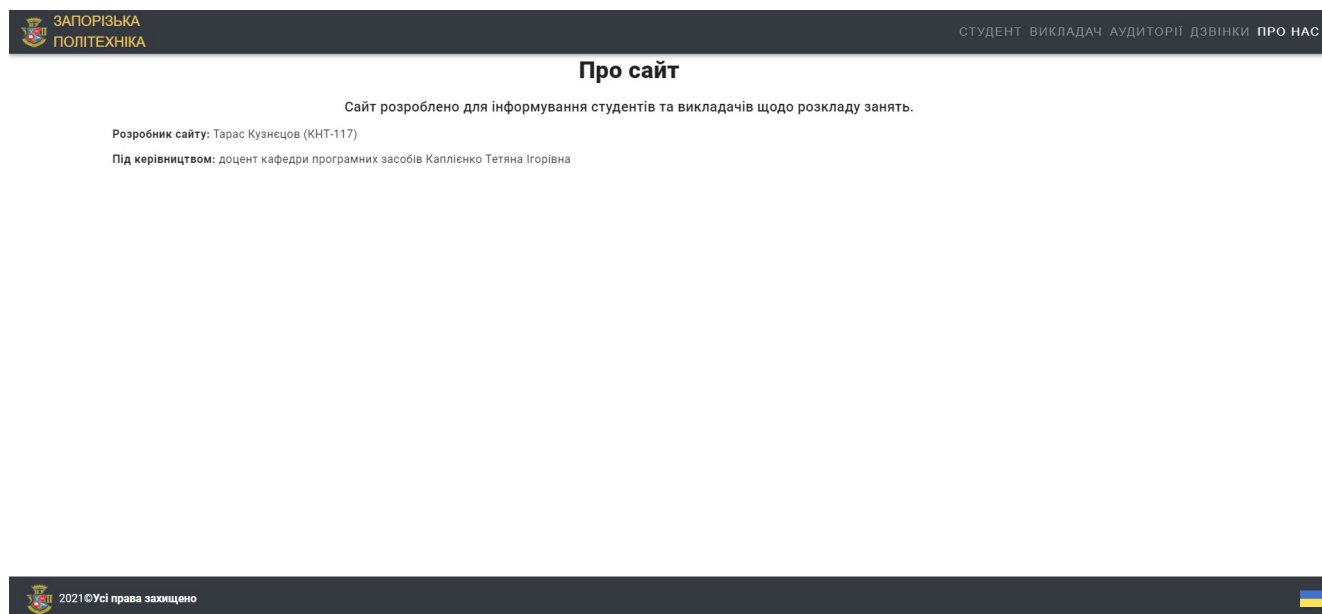


Рисунок 5.8 – Сторінка інформації про сайт

5.4 Повідомлення оператору

В ході роботи оператор може отримати такі повідомлення, описи яких міститься в таблиці 5.1.

Таблиця 5.1 — Повідомлення та їхні описи

Повідомлення українською	Повідомлення англійською	Опис
Будь ласка виберіть аудиторію	Please select the audience	Оператор не вибрав аудиторію на сторінці пошуку аудиторій
Група студента повинна бути вибрана!	Student's group must be selected!"	Оператор не вибрав групу на сторінці пошуку груп
Викладач повинен бути вибраний!	Teacher must be selected	Оператор не вибрав викладача на сторінці пошуку викладача

5.4 Висновки по розділу

На даному етапі роботи було складено «Керівництво оператора», що містить призначення і умови виконання програми, описано процес виконання програми, а також приведено перелік повідомлень та їхні описи.

6 ТЕСТУВАННЯ І РЕЗУЛЬТАТИ

В цьому розділі проводиться опис обраних методологій тестування програмного забезпечення, а також результати проведеного тестування, перевірки програмного забезпечення на відповідність поставленим задачам у технічному завданні до проєкту.

6.1 Опис методики проведення тестування

Тестування клієнтської та серверної частини було проведено мануально, тобто власноруч. Таким чином вдалося уникнути написання складної логіки написання автоматичних тестів для графічного інтерфейсу користувача, а також було проведено оцінку якості верстки вебсайту.

6.2 Результати тестування

Для клієнтської частини тестування було проведено мануально. За допомогою інструментів розробника, вбудованих у браузер Google Chrome було перевірено адаптивність верстки вебзастосунку, те, як графічний інтерфейс змінюється при зміні ширини дисплею. Перевірено зовнішній вигляд з мобільних пристроїв і планшетів, а також персональних комп'ютерів.

Було перевірено реакцію системи на введення як коректних, так і некоректних даних, її поведінку під час помилок валідації. Процес тестування наведено нижче:

- якщо під час введення даних у поля пошуку викладача заповнити всі поля окрім повного ім'я викладача, то користувач отримає повідомлення «Викладач повинен бути вибраний!» (див. рис 6.1);
- якщо під час введення даних у поля пошуку групи заповнити всі поля окрім повного назви групи, то користувач отримає повідомлення «Група студента повинна бути вибрана!» (див. рис 6.2);

– якщо під час введення даних у поля пошуку аудиторії не поле назви аудиторії, то користувач отримає повідомлення «Будь ласка виберіть аудиторію» (див.рис 6.3).

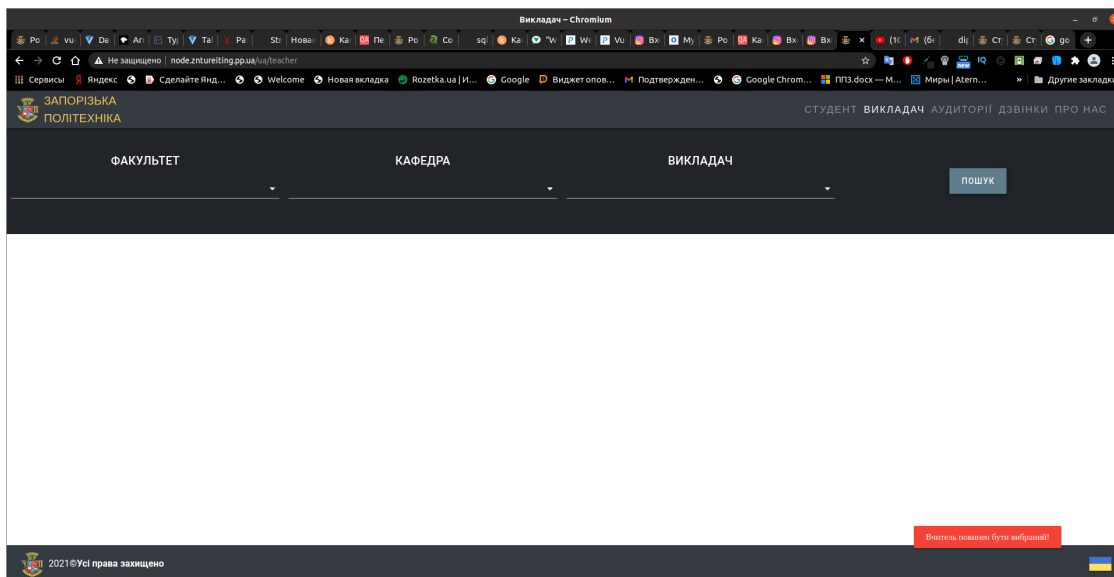


Рисунок 6.1 – Реакція на помилку під час заповнення форми пошуку викладача

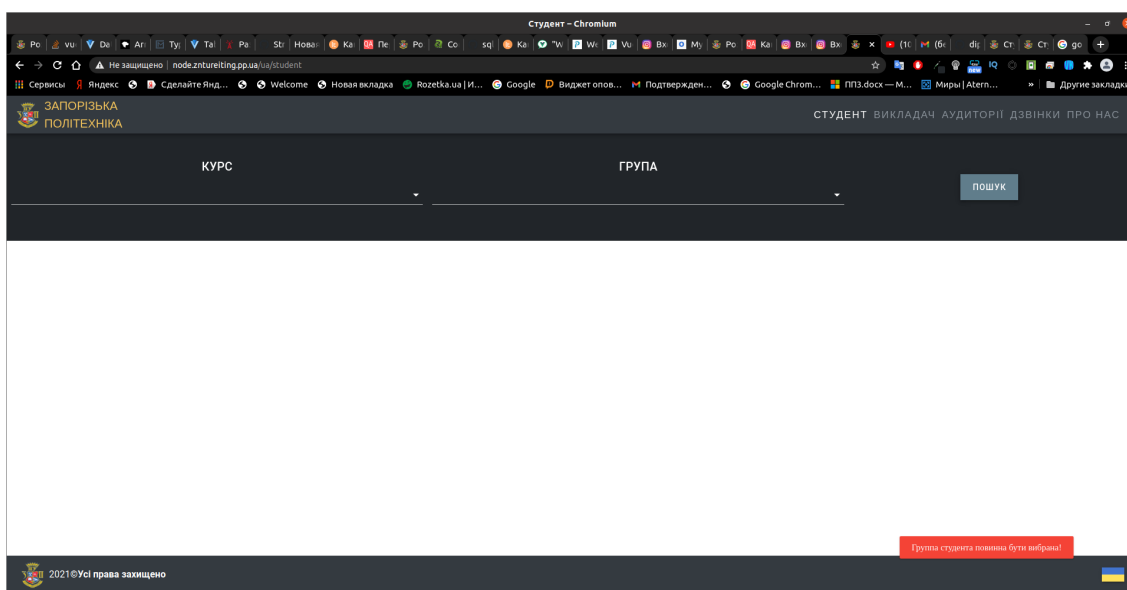


Рисунок 6.2 – Реакція на помилку під час заповнення форми пошуку групи

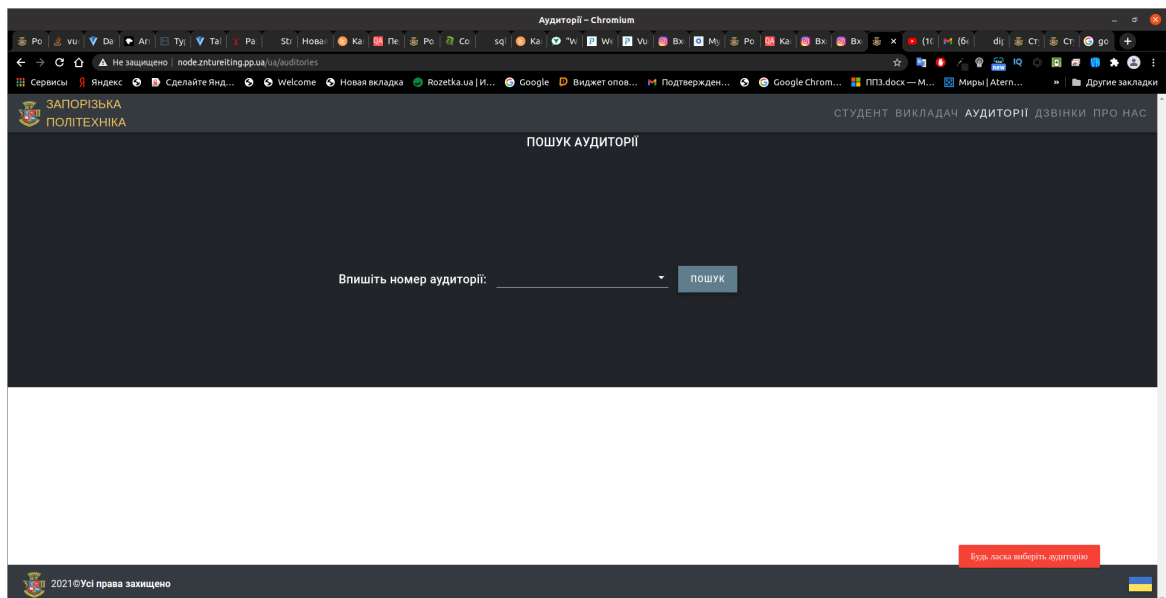


Рисунок 6.3 – Реакція на помилку під час заповнення форми пошуку аудиторії

ВИСНОВКИ

Під час виконання роботи було досліджено предметну область, розглянуто проблематику поставленого завдання, наведено його актуальність, доцільність. В даній роботі вивчалася проблема моніторингу розкладу закладів вищої освіти.

На першому етапі роботи було ретельно проаналізовано предметну область: сформовано проблему та актуальність роботи, розглянуто існуючі методи вирішення розв'язуваної проблеми та готові рішення, виявлено їх переваги та недоліки. Наступним кроком стала розробка архітектури системи, в ході якої було описано структуру розроблюваної програми, обрано інструменти для розробки ПЗ, а також спроектовано майбутню БД. На наступному етапі було реалізовано компоненти програми на основі «Технічного завдання» та надано їхнє обґрунтування.

Мета роботи – створення вебзастосунку обробки даних, необхідних для моніторингу розкладу закладів вищої освіти.

Було проведено порівняльний аналіз основних методів для вирішення цього завдання та встановлено, що існуючі засоби не відповідають повному функціоналу, якій потрібен для вирішення завдання дипломної роботи. Тому виникає необхідність в реалізації нового вебзастосунку з розширеним функціоналом.

Відповідно до мети були поставлені і вирішені наступні завдання:

- реалізований графічний інтерфейс користувача;
- реалізована можливість вибору студента, викладача або аудиторії з БД;
- реалізована можливість пошуку розкладу студентів НУ «Запорізька політехніка»;
- реалізована можливість пошуку місцезнаходження аудиторії за її номером;
- реалізована можливість переведення інтерфейсу сайту на англійську мову;

- реалізована можливість перегляду розкладу за чисельником та знаменником;
- реалізована сторінка для перегляду розкладу дзвінків у ЗВО.

Розроблено програмний продукт для моніторингу розкладу закладів вищої освіти. Серверна частина створеного ПП було написаний на мові програмування JS. Клієнтська частина створеного ПП була написана на мові програмування TS. У якості середовища розробки було використано Visual studio code. Для роботи з базою даних була використана бібліотека knex.js.

В результаті роботи було отримано програмний продукт, що дозволяє виконати поставлене завдання, а саме – моніторингу розкладу для ЗВО. Роботу програми було протестоване на тестових даних, отриманих вже з готової бази даних ЗВО НУ «Запорізька політехніка». За необхідності програма може бути модифікована та використана під час розробки більш складної, комплексної системи, що вирішуватиме завдання моніторингу розкладу ЗВО «Запорізька політехніка».

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Расписание политехнического института [Электрон. ресурс]. – Режим доступа: <https://www.spbstu.ru/students/schedule/>
2. United States University Academic Calendar [Electronic resource]. – Access mode: <https://www.usuniversity.edu/current-students/academic-calendar/>
3. Cameron Univercity Academic Calendar [Electronic resource]. – Режим доступа: <https://www.cameron.edu/academics/calendar>
4. JavaScript [Электрон. ресурс]. – Режим доступа: <https://uk.wikipedia.org/wiki/JavaScript>
5. TypeScript [Электрон. ресурс]. – Режим доступа: <https://ru.wikipedia.org/wiki/TypeScript>
6. Python [Электрон. ресурс]. – Режим доступа: <https://ru.wikipedia.org/wiki/Python>
7. PHP [Электрон. ресурс]. – Режим доступа: <https://ru.wikipedia.org/wiki/PHP>
8. Vue.js [Электрон. ресурс]. – Режим доступа: <https://ru.vuejs.org/v2/guide/index.html>
9. React [Электрон. ресурс]. – Режим доступа: <https://ru.wikipedia.org/wiki/React>
10. Angular [Электрон. ресурс]. – Режим доступа: <https://metanit.com/web/angular2/1.1.php>
11. SPA [Electronic resource]. – Access mode: https://en.wikipedia.org/wiki/Single-page_application.com
12. Node.js frameworks [Электрон. ресурс]. – Режим доступа: <https://habr.com/ru/post/486886/>.
13. Knex.js [Electronic resource]. – Access mode: <http://knexjs.org/>.
14. VS code [Электрон. ресурс]. – Режим доступа: <https://code.visualstudio.com/docs>.

15. Rest API [Электрон. ресурс]. – Режим доступа:
<https://habr.com/ru/post/483202/>.

ДОДАТОК А
Технічне завдання

ВСТУП

Розроблений програмний продукт використовується для вирішення проблеми моніторингу розкладу закладів вищої освіти, яка пов'язана з розвитком діджиталізації у всьому світі. Одним з основних завдань першого етапу при створенні раціональної системи моніторингу розкладу є побудування логічного та зрозумілого дизайну.

ПП «NUZP digital schedule» забезпечує моніторинг розкладу з різних цифрових девайсів, таких як: комп'ютер, планшет та телефон. ПП дозволяє моніторити розклад дзвінків, розклад навчання різних навчальних груп, розклад для кожного окремого викладача, а також пошук аудиторій за їх номером.

A.1 Мета розробки і її призначення

Об'єкт дослідження – програмні засоби моніторингу розкладу закладів вищої освіти.

Предмет дослідження – вебзастосунок для моніторингу розкладу закладів вищої освіти.

Мета роботи – створення вебзастосунку обробки даних, необхідних для моніторингу розкладу закладів вищої освіти.

Актуальність та проблематика роботи, полягає в тому, що останнім часом усі країни світу роблять великі кроки для діджиталізації різних систем, а тому паперові розклади занять відійшли далеко у минуле. При проєктуванні систем моніторингу використовується дуже великий обсяг даних, дуже складно вибрати саме потрібну інформацію, без зайвих параметрів, а також без ускладненого графічного інтерфейсу системи. Особлива потреба в моніторингу розкладу є саме доступ до розкладу з усіх місць, де є інтернет. Використання паперового розкладу для моніторингу розкладу призвели саме до розробки нових методів моніторингу розкладу ЗВО.

А.2 Стислий зміст основних теоретичних посилок до розробки на основі проведених пошуків

При проєктуванні моніторингу розкладу ЗВО виникає завдання вибору раціональних даних для виводу їх на сайт. Під вибором раціональних даних розуміють правильне конструювання запитів до БД та витягнення потрібних даних для певного викладача, студента або аудиторії.

А.3 Підстави для розробки

Підставою для виконання розробки є завдання на дипломний проєкт на тему “Розробка вебсайту для моніторингу розкладу навчального процесу”, затверджене наказом по НУ «Запорізька політехніка» № 103 від 30 березня 2021 року.

А.4 Основні вимоги до програми, що розробляється, початковим даним і результату

А.4.1 Вимоги до функціональних характеристик

Вимоги до функціональних характеристик:

- повинен бути реалізований графічний інтерфейс користувача;
- повинна бути реалізована можливість вибору студента, викладача або аудиторії з БД;
- повинна бути реалізована можливість пошуку розкладу студентів НУ «Запорізька політехніка»;
- повинна бути реалізована можливість пошуку місцезнаходження аудиторії за її номером;
- повинна бути реалізована можливість переведення інтерфейсу сайту на англійську мову;

- повинна бути реалізована можливість перегляду розкладу за чисельником та знаменником;

- повинна бути реалізована сторінка для перегляду розкладу дзвінків у ЗВО.

A.4.2 Вимоги до надійності

ПП повинна виконувати наступні вимоги до надійності:

- захищати ПП від спроби завантаження непотрібних даних до БД;
- захищати ПП від помилкових дій користувача (наприклад користувач ввів невалідні дані);

- обробляти виняткові ситуації (наприклад, відсутність підключення до бази даних);

ПП повинен нормально функціонувати при безперервній роботі комп'ютера. У разі виникнення збоїв в роботі серверу, відновлення нормальної роботи ПП повинно проводитися після перезавантаження системи.

A.4.3 Умови експлуатації

Умови експлуатації:

- програмне забезпечення можна з легкістю запуснути на різних серверах, за допомогою клонування з репозиторію.

- для користування програмним забезпеченням не потрібна додаткова кваліфікація та додаткові навички.

- програмне забезпечення повинне мати адаптивну верстку для пристроїв з яких можна запуснути сучасний браузер.

A.4.4 Вимоги до складу та параметрів технічних засобів

Вимоги до складу та параметрів технічних засобів:

- платформа x64-based;
- операційна система Microsoft Windows 10/8/7 , Linux Ubuntu 16.04 та інші аналоги сучасних UNIX-оїдних систем;
- процесор з тактовою частотою не нижче 1.4 ГГц;
- 1 ГБ оперативної пам'яті, рекомендується 8 ГБ оперативної пам'яті для швидкого реагування інтерфейсу на різні запити;
- вільне місце на диску 500 МБ мінімум.

A.4.5 Вимоги до маркування та упакування

Програмне забезпечення не потребує додаткових дій при маркуванні та упакуванні.

A.4.6 Вимоги до транспортування та збереження

Вимоги до транспортування й зберігання аналогічні вимогам, які пропонуються до носія, на якому записана програма.

A.5 Вимоги до програмної документації

Склад програмної документації повинен включати в себе:

- технічне завдання;
- керівництво програміста;
- керівництво оператора;
- результати тестування.

А.6 Основні обмеження на установку і використання програмного продукту

Розроблюваний програмний продукт потребує запуск на сервері та внесення певних змін у конфігурацію проєкта. Програмний продукт повинен використовуватись на комп'ютерах, які задовільняють вимогам та параметрам технічних засобів. Програмний продукт може використовуватись користувачем у будь-який момент часу, за наявності підключення до інтернету.

ДОДАТОК Б
Текст програми

Б.1 Файл application.js

```

const express = require('express')
const path = require('path')
const cors = require('cors')
const port = process.env.PORT || 8000

const app = express()
app.use(cors())

const production = false

app.use(express.json({extended: true}))

app.use('/api/teacher', require('./controllers/teacherController'));
app.use('/api/student', require('./controllers/studentController'));
app.use('/api/auditories', require('./controllers/auditoriesControllet'));

async function start() {
  try {
    if(production) {

      app.use(express.static(__dirname + '/rozklad/dist'))

      app.get(/.*/, (req,res) => {
        res.sendFile(path.join(__dirname + '/rozklad/dist/index.html'))
      })
    }

    // connection.connect((err)=> {
    //   if(err) throw err;
    //   console.log("Connected to database!")
    // })

    app.listen(port, () => {
      console.log(`Server has been started on port: ${port}`)
    })

  } catch(e) {
    {
      console.log(e)
    }
  }
}

start();

```

Б.2 Файл knexfile.js

// Update with your config settings.

```

module.exports = {

  development: {
    client: 'mysql',
    connection: {
      host: '127.0.0.1',
      database: 'app_data_3',
      user: 'root',
      password: '240500'
    },
    migrations: {

```

```

    tableName: 'knex_migrations'
  }
},
production: {
  client: 'mysql',
  connection: {
    host: '127.0.0.1',
    database: 'rozklad',
    user: 'root',
    password: '240500'
  },
  migrations: {
    tableName: 'knex_migrations'
  }
},
};

```

Б.3 Файл `auditoriesController.js`

```

const {Router} = require('express')
const router = Router()

const knexConfig = require("../knexfile")
const knex = require('knex')
const db = knex(knexConfig.development);

router.get('/getAuditories', async(req,res) => {
  try {
    let auditoires = await db
      .select('auditory_id', 'auditory_name', 'not_used')
      .from('auditory')

    res.send(auidtoires);
  }
  catch(e) {
    res.status(400).send("Something went wrong!")
  }
})

router.get('/getAuditoryInfo', async(req, res) => {
  try {
    let auditoryResult = await db
      .join('auditory_type', 'auditory.type_auditory', '=', 'auditory_type.auditory_type_id')
      .join('department', 'auditory.department_id', '=', 'department.department_id')
      .join('faculty', 'department.faculty_id', '=', 'faculty.faculty_id')
      .select()
      .where({'auditory_id': req.query.auditory})
      .from('auditory').first()
    return res.send(auditoryResult)
  }
  catch(e) {
    return res.status(400).send("this auditory not found");
  }
})

module.exports = router

```

Б.4 Файл studentController.js

```

const knexConfig = require("../knexfile")
const knex = require('knex')
const db = knex(knexConfig.development);

const {Router} = require('express')
const router = Router()

router.get('/getStudyGroups', async(req,res) =>
{
  try
  {
    let groups = await db('study_group')
    res.send(groups);
  }
  catch(e)
  {
    res.status(400).send("Something wrong");
  }
})

router.get('/getStudentLessonsInfo', async(req, res) => {
  try {
    const lessonsInfo = await db
      .join('study_group', 'lesson_group.group_id', '=', 'study_group.study_group_id')
      .join('lesson', 'lesson_group.lesson_id', '=', 'lesson.lesson_id')
      .join('discipline', 'lesson.discipline_id', '=', 'discipline.discipline_id')
      .join('department', 'lesson.department_id', '=', 'department.department_id')
      .join('faculty', 'department.faculty_id', '=', 'faculty.faculty_id')
      // .join('lesson_auditory', 'lesson_auditory.lesson_id', '=', 'lesson.lesson_id')
      // .join('auditory', 'auditory.auditory_id', '=', 'lesson_auditory.auditory_id')
      // .join('auditory_type', 'auditory.type_auditory', '=', 'auditory_type.auditory_type_id')
      .join('lesson_teacher', 'lesson_teacher.lesson_id', '=', 'lesson.lesson_id')
      .join('teacher', 'teacher.teacher_id', '=', 'lesson_teacher.teacher_id')
      .join('department as teacher_department', 'teacher.department_id', '=', 'teacher_department.department_id')
      .join('department as group_department', 'study_group.department_id', '=', 'group_department.department_id')
      .join('lesson_time', 'lesson_time.lesson_id', '=', 'lesson.lesson_id')
      .select(
        { lesson_type: 'lesson.type' },
        { lesson_count_of_hours: 'lesson.countOfHours' },
        { lesson_control : 'lesson.control' },
        { study_group_code: 'study_group.study_group_code' },
        { study_full_name: 'study_group.full_name' },
        { study_course_number: 'study_group.course_number' },
        { study_count_of_students: 'study_group.count_of_students' },
        { discipline_full_name: 'discipline.full_name' },
        { discipline_short_name: 'discipline.short_name' },
        { department_full_name: 'department.full_name' },
        { department_short_name: 'department.short_name' },
        { faculty_faculty_code: 'faculty.faculty_code' },
        { faculty_full_name: 'faculty.full_name' },
        // { auditory_auditory_name: 'auditory.auditory_name' },
        // { auditory_id: 'auditory.auditory_id' },
        // { auditory_not_used: 'auditory.not_used' },
        // { auditory_count_of_places: 'auditory.count_of_places' },
        // { auditory_corps_number: 'auditory.corps_number' },
        // { auditory_type_name: 'auditory_type.auditory_type_name' },
        { teacher_full_name: 'teacher.full_name' },
        { teacher_sex: 'teacher.sex' },
        { teacher_post: 'teacher.post' },
      )
  }
})

```

```

    { teacher_status: 'teacher.status' },
    { teacher_is_not_working: 'teacher.is_not_working' },
    { teacher_favourite_days: 'teacher.favourite_days' },
    { teacher_favourite_time: 'teacher.favourite_time' },
    { teacher_priority: 'teacher.priority' },
    { teacher_id: 'teacher.teacher_id' },
    { teacher_department_full_name: 'teacher_department.full_name' },
    { teacher_department_short_name: 'teacher_department.short_name' },
    { study_group_department_full_name: 'group_department.full_name' },
    { study_group_department_short_name: 'group_department.short_name' },
    { pare_number: 'lesson_time.number' },
    { type_of_week: 'lesson_time.type_of_week' },
    { day_of_week: 'lesson_time.day_of_week' },
  )
  .where({ 'group_id': req.query.groupId })
  .from('lesson_group')

let app = await db
  .join('study_group', 'lesson_group.group_id', '=', 'study_group.study_group_id')
  .join('lesson_auditory', 'lesson_auditory.lesson_id', '=', 'lesson_group.lesson_id')
  .join('auditory', 'auditory.auditory_id', '=', 'lesson_auditory.auditory_id')
  .join('auditory_type', 'auditory.type_auditory', '=', 'auditory_type.auditory_type_id')
  .join('lesson_time', 'lesson_time.lesson_id', '=', 'lesson_group.lesson_id')
  .select(
    { study_group_id: 'study_group.study_group_id' },
    { study_group_code: 'study_group.study_group_code' },
    { study_full_name: 'study_group.full_name' },
    { study_course_number: 'study_group.course_number' },
    { auditory_auditory_name: 'auditory.auditory_name' },
    { auditory_id: 'auditory.auditory_id' },
    { type_of_week: 'lesson_time.type_of_week' },
    { day_of_week: 'lesson_time.day_of_week' },
    { pare_number: 'lesson_time.number' },
  )
  .where({ 'group_id': req.query.groupId })
  .from('lesson_group')

const GroupInfo = await db
  .join('department', 'study_group.department_id', '=', 'department.department_id')
  .join('faculty', 'department.faculty_id', 'faculty.faculty_id')
  .select(
    { study_full_name: 'study_group.full_name' },
    { study_course_number: 'study_group.course_number' },
    { study_count_of_students: 'study_group.count_of_students' },
    { department_full_name: 'department.full_name' },
    { department_short_name: 'department.short_name' },
    { faculty_faculty_code: 'faculty.faculty_code' },
    { faculty_full_name: 'faculty.full_name' },
  )
  .where({ 'study_group_id': req.query.groupId })
  .from('study_group').first()

lessonsInfo.map(e => {
  newData = [];
  Auditories = [];
  return app.map(l => {
    if(e.type_of_week === l.type_of_week && e.day_of_week === l.day_of_week && e.pare_number ===
l.pare_number)
    {
      if(!newData.find(x => x.study_full_name === l.study_full_name))
      {

```

```

        newData.push(l);
    }

    if(!Auditories.find(x => x.auditory_auditory_name === l.auditory_auditory_name))
    {
        Auditories.push({
            auditory_auditory_name: l.auditory_auditory_name,
            auditory_id: l.auditory_id
        });
    }
    e.newData = newData;
    e.Auditories = Auditories;
}
}))
})

const Monday = [];
const Tuesday = [];
const Wednesday = [];
const Thursday = [];
const Friday = [];
const Saturday = [];

lessonsInfo.map(e => {
    if(e.day_of_week === 1)
    {
        return Monday.push(e)
    }
    if(e.day_of_week === 2)
    {
        return Tuesday.push(e)
    }
    if(e.day_of_week === 3)
    {
        return Wednesday.push(e)
    }
    if(e.day_of_week === 4)
    {
        return Thursday.push(e)
    }
    if(e.day_of_week === 5)
    {
        return Friday.push(e)
    }
    if(e.day_of_week === 6)
    {
        return Saturday.push(e)
    }
})

return res.send({
    Monday: Monday,
    Tuesday: Tuesday,
    Wednesday: Wednesday,
    Thursday: Thursday,
    Friday: Friday,
    Saturday: Saturday,
    GroupInfo: GroupInfo
});
}
catch(e) {

```

```

    return res.status(400).send(e);
  }
})

```

```
module.exports = router
```

Б.5 Файл teacherController.js

```

const knexConfig = require("../knexfile")
const knex = require('knex')
const db = knex(knexConfig.development);

const {Router} = require('express')
const router = Router()

router.get('/getTeacher', async(req, res) => {
  try
  {
    const teacherData = await db
      .join('department', 'teacher.department_id', '=', 'department.department_id')
      .join('faculty', 'department.faculty_id', '=', 'faculty.faculty_id')
      .select({ teache_name: 'teacher.full_name' },
        { teacher_id: 'teacher.teacher_id' },
        { faculty_id: 'faculty.faculty_id' },
        { faculty_name: 'faculty.full_name' },
        { department_id: 'department.department_id' },
        { department_name: 'department.full_name' } )
      .from('teacher');

    const departmentData = await db('department')

    const facultyData = await db('faculty')

    const data = {
      teacherData: teacherData,
      departmentData: departmentData,
      facultyData: facultyData
    }

    res.status(200).send(data)
  }
  catch(e)
  {
    res.status(400).send("Something wrong!")
  }
})

router.get('/getTeacherLessonsInfo', async(req, res) => {
  try {
    const lessonsInfo = await db
      .join('lesson_teacher', 'lesson_teacher.teacher_id', '=', 'teacher.teacher_id')
      .join('lesson', 'lesson.lesson_id', '=', 'lesson_teacher.lesson_id')
      .join('lesson_time', 'lesson_time.lesson_id', '=', 'lesson_teacher.lesson_id')
      .join('discipline', 'lesson.discipline_id', '=', 'discipline.discipline_id')
      .join('department', 'lesson.department_id', '=', 'department.department_id')
      .join('faculty', 'department.faculty_id', 'faculty.faculty_id')
      // .join('lesson_group', 'lesson_group.lesson_id', '=', 'lesson.lesson_id')
      // .join('study_group', 'lesson_group.group_id', '=', 'study_group.study_group_id')
      // .join('lesson_auditory', 'lesson_auditory.lesson_id', '=', 'lesson.lesson_id')

```

```

    // .join('auditory', 'auditory.auditory_id', '=', 'lesson_auditory.auditory_id')
    // .join('auditory_type', 'auditory.type_auditory', '=', 'auditory_type.auditory_type_id')
    // .join('department as group_department', 'study_group.department_id', '=', 'group_department.department_id')
    .select(
      { lesson_type: 'lesson.type' },
      { lesson_count_of_hours: 'lesson.countOfHours'},
      { lesson_control : 'lesson.control'},
      // { study_group_code: 'study_group.study_group_code'},
      // { study_full_name: 'study_group.full_name' },
      // { study_course_number: 'study_group.course_number'},
      // { study_count_of_students: 'study_group.count_of_students'},
      { discipline_full_name: 'discipline.full_name' },
      { discipline_short_name: 'discipline.short_name'},
      { department_full_name: 'department.full_name'},
      { department_short_name: 'department.short_name'},
      { faculty_faculty_code: 'faculty.faculty_code'},
      { faculty_full_name: 'faculty.full_name'},
      // { auditory_auditory_name: 'auditory.auditory_name'},
      // { auditory_id: 'auditory.auditory_id'},
      // { auditory_not_used: 'auditory.not_used'},
      // { auditory_count_of_places: 'auditory.count_of_places'},
      // { auditory_corps_number: 'auditory.corps_number'},
      // { auditory_type_name: 'auditory_type.auditory_type_name'},
      { teacher_full_name: 'teacher.full_name'},
      { teacher_sex: 'teacher.sex'},
      { teacher_post: 'teacher.post'},
      { teacher_status: 'teacher.status' },
      { teacher_is_not_working: 'teacher.is_not_working' },
      { teacher_favourlesson_teacherity: 'teacher.priority'},
      { pare_number: 'lesson_time.number'},
      { type_of_week: 'lesson_time.type_of_week'},
      { day_of_week: 'lesson_time.day_of_week'},
    )
  )
  .where({'teacher.teacher_id': req.query.groupId})
  .from('teacher')

let newData = [];
let Auditories = [];

const app = await db
.join('lesson_teacher', 'lesson_teacher.teacher_id', '=', 'teacher.teacher_id')
.join('lesson_time', 'lesson_time.lesson_id', '=', 'lesson_teacher.lesson_id')
.join('lesson', 'lesson.lesson_id', '=', 'lesson_teacher.lesson_id')
.join('lesson_group', 'lesson_group.lesson_id', '=', 'lesson.lesson_id')
.join('study_group', 'lesson_group.group_id', '=', 'study_group.study_group_id')
.join('lesson_auditory', 'lesson_auditory.lesson_id', '=', 'lesson.lesson_id')
.join('auditory', 'auditory.auditory_id', '=', 'lesson_auditory.auditory_id')
.join('auditory_type', 'auditory.type_auditory', '=', 'auditory_type.auditory_type_id')
.join('department as group_department', 'study_group.department_id', '=', 'group_department.department_id')
.select(
  { study_group_id: 'study_group.study_group_id'},
  { study_group_code: 'study_group.study_group_code'},
  { study_full_name: 'study_group.full_name' },
  { study_course_number: 'study_group.course_number'},
  { auditory_auditory_name: 'auditory.auditory_name'},
  { auditory_id: 'auditory.auditory_id'},
  { type_of_week: 'lesson_time.type_of_week'},
  { day_of_week: 'lesson_time.day_of_week'},
  { pare_number: 'lesson_time.number'},
)
)
.where({'teacher.teacher_id': req.query.groupId})
.from('teacher')

```

```

let teacher = await db
.select()
.where({'teacher.teacher_id': req.query.groupId})
.from('teacher')
.first()

lessonsInfo.map(e => {
  newData = [];
  Auditories = [];
  return app.map(l => {
    if(e.type_of_week == l.type_of_week && e.day_of_week == l.day_of_week && e.pare_number ==
l.pare_number)
    {
      if(!newData.find(x => x.study_full_name == l.study_full_name))
      {
        newData.push(l);
      }

      if(!Auditories.find(x => x.auditory_auditory_name == l.auditory_auditory_name))
      {
        Auditories.push({
          auditory_auditory_name: l.auditory_auditory_name,
          auditory_id: l.auditory_id
        });
      }
      e.newData = newData;
      e.Auditories = Auditories;
    }
  })
})

const Monday = [];
const Tuesday = [];
const Wednesday = [];
const Thursday = [];
const Friday = [];
const Saturday = [];

lessonsInfo.map(e => {
  if(e.day_of_week == 1)
  {
    return Monday.push(e)
  }
  if(e.day_of_week == 2)
  {
    return Tuesday.push(e)
  }
  if(e.day_of_week == 3)
  {
    return Wednesday.push(e)
  }
  if(e.day_of_week == 4)
  {
    return Thursday.push(e)
  }
  if(e.day_of_week == 5)
  {
    return Friday.push(e)
  }
  if(e.day_of_week == 6)
  {
    return Saturday.push(e)
  }
})

```

```

    }

  })

  return res.send({
    Monday: Monday,
    Tuesday: Tuesday,
    Wednesday: Wednesday,
    Thursday: Thursday,
    Friday: Friday,
    Saturday: Saturday,
    Teacher: teacher
  });
}
catch(e) {
  return res.status(400).send(e);
}
})

module.exports = router

```

Б.6 Файл app.vue

```

<template>
  <v-app>
    <AppBar/>
    <v-main>
      <v-container fluid>
        <router-view></router-view>
      </v-container>
    </v-main>
    <Footer/>
  </v-app>
</template>

<script lang="ts">
import { Vue } from 'vue-property-decorator';
import AppBar from './components/appBar/AppBar.vue';
import Footer from './components/footer/Footer.vue';
export default Vue.extend({
  name: 'App',

  components: {
    AppBar,
    Footer
  },
  data: () => ({
    //
  }),
});
</script>

<style scoped>
.v-sheet
{
  background: rgb(52,58,64) !important;
}
</style>

```

```

<style>
.col
{
  padding: 0;
  margin: 0;
}
.row
{
  padding: 0;
  margin: 0;
}
.container
{
  padding: 0;
  margin: 0;
}
</style>

<style>
html{
  background-color: rgb(227,227,227);
}
</style>

```

Б.7 Файл i18n.ts

```

import Vue from 'vue'
import VueI18n, { LocaleMessages } from 'vue-i18n'

Vue.use(VueI18n)

function loadLocaleMessages (): LocaleMessages {
  const locales = require.context('./locales', true, /[A-Za-z0-9-_\s]+\.(json)$/i)
  const messages: LocaleMessages = {}
  locales.keys().forEach(key => {
    console.log(locales(key))
    const matched = key.match(/([A-Za-z0-9-_\s]+\.)\./i)
    if (matched && matched.length > 1) {
      const locale = matched[1]
      messages[locale] = locales(key)
    }
  })
  return messages
}

export default new VueI18n({
  locale: 'ua',
  fallbackLocale: 'ua',
  messages: loadLocaleMessages()
})

```

Б.8 Файл main.ts

```

import Vue from 'vue'
import App from './App.vue'
import router from './router'
import store from './store'
import vuetify from './plugins/vuetify';

```

```

import i18n from './i18n'
import VueMeta from 'vue-meta'
import axios from 'axios';

import Toasted from 'vue-toasted';

// you can also pass options, check options reference below
Vue.use(Toasted, {
  position: 'bottom-right',
  duration: '3000',
})

Vue.config.productionTip = false

axios.defaults.baseURL = "http://localhost:8000";
axios.defaults.headers.post['Content-Type'] = 'application/x-www-form-urlencoded';

router.beforeEach((to, from, next) =>
{
  let language = to.params.lang;
  if(!language) {
    language = 'ua'
  }
  i18n.locale = language;
  next();
})

Vue.use(VueMeta)

new Vue({
  router,
  store,
  vuetify,
  i18n,
  render: h => h(App)
}).$mount('#app')
```

Б.9 Файл Teacher.vue

```

<template>
<v-container fluid>
  <teacher-found/>
</v-container>
</template>

<script lang="ts">
import { MetaInfo } from 'vue-meta';
import { Component, Vue } from 'vue-property-decorator';
import TeacherFound from '@components/Teacher/TeacherFound.vue';

@Component({
  components: {
    TeacherFound
  },
  metaInfo(): MetaInfo {
    return {
      title: `${this.$t('nav.teacher')}`
    }
  }
})
```

```
export default class Teacher extends Vue {
}
</script>
```

Б.10 Файл Student.vue

```
<template>
<v-container fluid>
  <student-find/>
</v-container>
</template>

<script lang="ts">
import { MetaInfo } from 'vue-meta'
import { Component, Vue } from 'vue-property-decorator'
import StudentFind from '@components/Student/StudentFound.vue';

@Component({
  components: {
    StudentFind
  },
  metaInfo(): MetaInfo {
    return {
      title: `${this.$t('nav.student')}`
    }
  }
})
export default class Student extends Vue {

}
</script>
```

Б.11 Файл Bells.vue

```
<template>
<v-row>
  <v-col cols="12">
    <v-row justify="center">
      <h1>{{ $t('components.bellsSchedule') }}</h1>
    </v-row>
  </v-col>
  <v-col cols="12" class="text-center">
    <v-data-table
      :headers="[
        {
          text: this.$t('components.pair'),
          align: 'center',
          sortable: false,
          value: 'pair',
        },
        { text: this.$t('components.start'), value: 'start', align:'center' },
        { text: this.$t('components.end'), value: 'end', align:'center' },
        { text: this.$t('components.break'), value: 'break', align:'center' } ]"
      :items="data"
      hide-default-footer
      disable-sort
      class="elevation-1 dataTableClass">
    </v-data-table>
```

```

    </v-col>
  </v-row>
</template>

<script lang="ts">
import { MetaInfo } from 'vue-meta'
import { Component, Vue, Watch } from 'vue-property-decorator'

@Component({
  components: {
  },

  metaInfo(): MetaInfo {
    return {
      title: `${this.$t('nav.bells')}`
    }
  }
})

export default class Bells extends Vue{
  private data = require('../data/bells.json');
}
</script>

<style scoped>
.dataTableClass >>> td
{
  font-size: 18px !important;
}
.dataTableClass >>> th
{
  font-size: 20px !important;
}
.v-data-table > .v-data-table__wrapper .v-data-table__mobile-row
{
  display: table-caption !important;
}
</style>

```

Б.11 Файл AuditoryResult.vue

```

<template>
  <v-layout>
    <v-container v-if="AudienceSearch" fluid>
      <v-row no-gutters justify="center" class="display-1 pb-5">
        <span>{{ $t('components.auditoriesSearch') }}</span>
      </v-row>
      <v-row no-gutters>
        <v-col offset-md="3" offset-sm="1" offset-lg="3" xs="12" sm="12" lg="3" md="3" align-self="center">
          <v-row no-gutters justify="start">{{ $t('components.auditoryName') }}:
            ${auditoryResult.auditory_name} </v-row>
          <v-row no-gutters justify="start">{{ $t('components.auditoryCorpus') }}:
            ${auditoryResult.corps_number} </v-row>
          <v-row no-gutters justify="start">{{ $t('components.auditoryType') }}:
            ${auditoryResult.auditory_type_name} </v-row>
          <v-row no-gutters justify="start">{{ $t('components.auditoryFacultName') }}:
            ${auditoryResult.full_name}(${auditoryResult.faculty_code}) </v-row>
          <v-row no-gutters justify="start">{{ $t('components.auditoryPlaces') }}:
            ${auditoryResult.count_of_places} </v-row>

```

```

</v-col>
<v-col xs="8" sm="8" md="5" lg="5" >
  <v-row no-gutters v-if="auditoryResult.corps_number == corpuse.FirstCorpus ||
!auditoryResult.corps_number"><iframe
src="https://www.google.com/maps/embed?pb=!1m18!1m12!1m3!1d563.2135855614889!2d35.1808679640167!3d47.818
0778164619!2m3!1f0!2f0!3f0!3m2!1i1024!2i768!4f13.1!3m3!1m2!1s0x0%3A0x2dde5310c609e987!2z0JfQsNC_0L7Rg
NGW0LfRjNC60LjQuSDQvdCw0YbRltC-
0L3QsNC70YzQvdC40Lkg0YPQvdGW0LLQtdGA0YHQuNGC0LXRgg!5e0!3m2!1sru!2sua!4v1619980164090!5m2!1sr
u!2sua" width="600" height="450" style="border:0;" allowfullscreen="" loading="lazy"></iframe></v-row>
  <v-row no-gutters v-if="auditoryResult.corps_number == corpuse.SecondCorpus"><iframe
src="https://www.google.com/maps/embed?pb=!1m18!1m12!1m3!1d1126.4108873588439!2d35.17944143142274!3d47.8
1882836944344!2m3!1f0!2f0!3f0!3m2!1i1024!2i768!4f13.1!3m3!1m2!1s0x40dc5e0a2516f1c1%3A0xf524dbd0fc0bdb6e!
2z0KPRh9C10LHQvdGL0Lkg0LrQvtGA0L_Rg9GBINCX0J3QotCjIOKEljI!5e0!3m2!1sru!2sua!4v1619981993607!5m2!
1sru!2sua" width="600" height="450" style="border:0;" allowfullscreen="" loading="lazy"></iframe></v-row>
  <v-row no-gutters v-if="auditoryResult.corps_number == corpuse.ThirdCorpus"><iframe
src="https://www.google.com/maps/embed?pb=!1m18!1m12!1m3!1d2679.114357102718!2d35.18360591580861!3d47.81
800137919928!2m3!1f0!2f0!3f0!3m2!1i1024!2i768!4f13.1!3m3!1m2!1s0x40dc5e0abc8f7ad7%3A0xff4424b2b527d5ab!2
z0KPRh9C10LHQvdGL0Lkg0LrQvtGA0L_Rg9GBINCX0J3QotCjIOKEljM!5e0!3m2!1sru!2sua!4v1619982027268!5m2
!1sru!2sua" width="600" height="450" style="border:0;" allowfullscreen="" loading="lazy"></iframe></v-row>
  <v-row no-gutters v-if="auditoryResult.corps_number == corpuse.FourthCorpus"><iframe
src="https://www.google.com/maps/embed?pb=!1m18!1m12!1m3!1d2679.1524229140305!2d35.18354951580851!3d47.8
1726367919918!2m3!1f0!2f0!3f0!3m2!1i1024!2i768!4f13.1!3m3!1m2!1s0x40dc5e0ac8fc078d%3A0x2335820b3494e9de
!2z0KPRh9C10LHQvdGL0Lkg0LrQvtGA0L_Rg9GBIOKEljQg0JfQndCi0KM!5e0!3m2!1sru!2sua!4v1619982094737!5
m2!1sru!2sua" width="600" height="450" style="border:0;" allowfullscreen="" loading="lazy"></iframe></v-row>
  <v-row no-gutters v-if="auditoryResult.corps_number == corpuse.FifthCorpus"><iframe
src="https://www.google.com/maps/embed?pb=!1m18!1m12!1m3!1d2679.1131031989858!2d35.182579615808564!3d47.
81802567919914!2m3!1f0!2f0!3f0!3m2!1i1024!2i768!4f13.1!3m3!1m2!1s0x40dc5e0a946f52e1%3A0xef3eb11e99a8985e
!2z0KPRh9C10LHQvdGL0Lkg0LrQvtGA0L_Rg9GBINCX0J3QotCjIOKEljU!5e0!3m2!1sru!2sua!4v1619982111186!5m2
!1sru!2sua" width="600" height="450" style="border:0;" allowfullscreen="" loading="lazy"></iframe></v-row>
  </v-col>
</v-row>
</v-container>
<v-container v-else fluid>
  <v-row no-gutters justify="center" class="display-1">{{$(components.AudienceDoesntExist)}}</v-row>
</v-container>
</v-layout>
</template>

<script lang="ts">
import { MetaInfo } from 'vue-meta'
import { Component, Vue, Prop } from 'vue-property-decorator'
import AuditoryResultModel from '@models/auditoryResultModel';
import ECorpuse from '@enums/corpusesEnum';
import axios from 'axios'

@Component({
  components: {
  },
  metaInfo(): MetaInfo {
    return {
      title: `${this.$t('nav.auditories')}`
    }
  }
})
export default class Auditories extends Vue {
  private auditoryResult: AuditoryResultModel = {} as AuditoryResultModel;
  private corpuse = ECorpuse;
  private AudienceSearch = true;

  @Prop({default: null})
  auditory: string

  private async mounted() {

```

```

    await axios.get('api/auditories/getAuditoryInfo',
    { params:
    { auditory: this.auidtory}
    }).then(res => this.auditoryResult = res.data)
    .catch(err => Vue.toasted.error(err.message))
    if(!this.auditoryResult.auditory_name) {
      this.AudienceSearch = false;
    }
  }
}
</script>

```

Б.12 Файл Auditories.vue

```

<template>
  <v-layout>
    <AuditoriesComponent/>
  </v-layout>
</template>

<script lang="ts">
import { MetaInfo } from 'vue-meta'
import { Component, Vue } from 'vue-property-decorator'
import AuditoriesComponent from '../components/Auditories/Auditories.vue';

@Component({
  components: {
    AuditoriesComponent
  },
  props: ['auditory'],
  metaInfo(): MetaInfo {
    return {
      title: `${this.$t('nav.auditories')}`
    }
  }
})
export default class Auditories extends Vue {

}
</script>

```

Б.13 Файл About.vue

```

<template>
<v-container class="pt-1">
  <v-row no-gutters>
    <v-col cols="12" class="display-1 pa-0 ma-0 font-weight-black">
      <v-row justify="center">{{ $t('components.aboutSite') }}</v-row>
    </v-col>
  </v-row>
  <v-row no-gutters class="pt-4" justify="center">
    <v-col cols="10" class="title text-center">
      {{ $t('components.thisSiteIsDesignedTo') }}
    </v-col>
  </v-row>

```

```

<v-row no-gutters>
  <v-col cols="1"/>
  <v-col cols="11" class="ma-0 pt-2 body-1">
    <v-row>
      {{ `${$t('components.siteDeveloper')}: ${$t('components.TarasKuznietsow')}` }}
    </v-row>
  </v-col>
</v-row>
</v-container>
</template>

<script lang="ts">
import { Component, Vue } from 'vue-property-decorator'
import Loader from '@components/loader/loader.vue';
import { MetaInfo } from 'vue-meta';
import axios from 'axios';

@Component({
  components: {
    Loader,
  },
  metaInfo(): MetaInfo {
    return {
      title: `${this.$t('nav.about')}`
    }
  }
})
export default class About extends Vue {

}
</script>

```

Б.14 Файл ResultTeacher.vue

```

<template>
  <v-container fluid>
    <v-tabs
      v-model="tab"
      centered
      dark>
      <v-tab>{{ `${$t('components.numerator')}` }}</v-tab>
      <v-tab>{{ `${$t('components.denominator')}` }}</v-tab>
    </v-tabs>
    <v-row no-gutters justify="center" class="display-1 pb-4">
      <span>{{ Lessons.Teacher.full_name }}</span>
    </v-row>
    <v-row no-gutters justify="center" class="display-1">
      <span>{{ `${$t('components.Monday')}` }}</span>
    </v-row>
    <v-row no-gutters>
      <v-col cols="12">
        <v-simple-table dense>
          <template v-slot:default>
            <thead>
              <tr>
                <th class="text-left">
                  {{ `${$t('components.pair')}` }}
                </th>

```

```

<th class="text-left">
  {{ $t('components.AcademicSubject') }}
</th>
<th class="text-left">
  {{ $t('components.Auditory') }}
</th>
<th class="text-left">
  {{ $t('components.Group') }}
</th>
<th class="text-left">
  {{ $t('components.LessonType') }}
</th>
</tr>
</thead>
<tbody>
<tr>
  v-for="item in Lessons.Monday"
  :key="item.pare_number"
  >
    <td class="text-left" v-if="item.type_of_week == tab || item.type_of_week ==
2"><span>{{ item.pare_number }}</span></td>
    <td class="text-left" v-if="item.type_of_week == tab || item.type_of_week ==
2"><span>{{ item.discipline_full_name }}</span></td>
    <td class="text-left" v-if="item.type_of_week == tab || item.type_of_week == 2">
      <v-col cols="1">
        <router-link style="text-decoration: none;" class="pb-2" :to="{name: 'auditoriesResult', params:
{auditory: auditData.auditory_id}}" v-for=" auditData in item.Auditories" :key="auditData.auditory_id">
          <v-row no-gutters>
            {{ auditData.auditory_auditory_name }}
          </v-row>
        </router-link>
      </v-col>
    </td>
    <td class="text-left" v-if="item.type_of_week == tab || item.type_of_week == 2">
      <v-col cols="1">
        <router-link style="text-decoration: none;" class="pb-2" :to="{name: 'group-result', params: {groupId:
auditData.study_group_id}}" v-for=" auditData in item.newData" :key="auditData.study_full_name">
          <v-row no-gutters>{{ auditData.study_full_name }}</v-row>
        </router-link>
      </v-col>
    </td>
    <td class="text-left" v-if="item.type_of_week == tab || item.type_of_week ==
2"><span>{{ item.lesson_type }}</span></td>
  </tr>
</tbody>
</template>
</v-simple-table>
</v-col>
</v-row>
<v-row no-gutters justify="center" class="display-1">
  <span>{{ $t('components.Tuesday') }}</span>
</v-row>
<v-row no-gutters>
  <v-col cols="12">
    <v-simple-table dense>
      <template v-slot:default>
        <thead>
          <tr>
            <th class="text-left">
              {{ $t('components.pair') }}
            </th>
            <th class="text-left">
              {{ $t('components.AcademicSubject') }}

```

```

</th>
<th class="text-left">
  {{ $t('components.Auditory') }}
</th>
<th class="text-left">
  {{ $t('components.Group') }}
</th>
<th class="text-left">
  {{ $t('components.LessonType') }}
</th>
</tr>
</thead>
<tbody>
<tr>
  v-for="item in Lessons.Thursday"
  :key="item.pare_number"
  >
    <td class="text-left" v-if="item.type_of_week == tab || item.type_of_week ==
2"><span>{{ item.pare_number }}</span></td>
    <td class="text-left" v-if="item.type_of_week == tab || item.type_of_week ==
2"><span>{{ item.discipline_full_name }}</span></td>
    <td class="text-left" v-if="item.type_of_week == tab || item.type_of_week == 2">
      <v-col cols="1">
        <router-link style="text-decoration: none;" class="pb-2" :to="{name: 'auditoriesResult', params:
{auditory: auditData.auditory_id}}" v-for=" auditData in item.Auditories" :key="auditData.auditory_id">
          <v-row no-gutters>
            {{ auditData.auditory_auditory_name }}
          </v-row>
        </router-link>
      </v-col>
    </td>
    <td class="text-left" v-if="item.type_of_week == tab || item.type_of_week == 2">
      <v-col cols="1">
        <router-link style="text-decoration: none;" class="pb-2" :to="{name: 'group-result', params: {groupId:
auditData.study_group_id}}" v-for=" auditData in item.newData" :key="auditData.study_full_name">
          <v-row no-gutters>{{ auditData.study_full_name }}</v-row>
        </router-link>
      </v-col>
    </td>
    <td class="text-left" v-if="item.type_of_week == tab || item.type_of_week ==
2"><span>{{ item.lesson_type }}</span></td>
  </tr>
</tbody>
</template>
</v-simple-table>
</v-col>
</v-row>
<v-row no-gutters justify="center" class="display-1">
  <span>{{ $t('components.Wednesday') }}</span>
</v-row>
<v-row no-gutters>
  <v-col cols="12">
    <v-simple-table dense>
      <template v-slot:default>
        <thead>
          <tr>
            <th class="text-left">
              {{ $t('components.pair') }}
            </th>
            <th class="text-left">
              {{ $t('components.AcademicSubject') }}
            </th>
            <th class="text-left">

```

```

        {{ $t('components.Auditory') }}
      </th>
      <th class="text-left">
        {{ $t('components.Group') }}
      </th>
      <th class="text-left">
        {{ $t('components.LessonType') }}
      </th>
    </tr>
  </thead>
  <tbody>
    <tr
      v-for="item in Lessons.Wednesday"
      :key="item.pare_number"
    >
      <td class="text-left" v-if="item.type_of_week == tab || item.type_of_week ==
2"><span>{{ item.pare_number }}</span></td>
      <td class="text-left" v-if="item.type_of_week == tab || item.type_of_week ==
2"><span>{{ item.discipline_full_name }}</span></td>
      <td class="text-left" v-if="item.type_of_week == tab || item.type_of_week == 2">
        <v-col cols="1">
          <router-link style="text-decoration: none;" class="pb-2" :to="{name: 'auditoriesResult', params:
{auditory: auditData.auditory_id}}" v-for=" auditData in item.Auditories" :key="auditData.auditory_id">
            <v-row no-gutters>
              {{ auditData.auditory_name }}
            </v-row>
          </router-link>
        </v-col>
      </td>
      <td class="text-left" v-if="item.type_of_week == tab || item.type_of_week == 2">
        <v-col cols="1">
          <router-link style="text-decoration: none;" class="pb-2" :to="{name: 'group-result', params: {groupId:
auditData.study_group_id}}" v-for=" auditData in item.newData" :key="auditData.study_full_name">
            <v-row no-gutters>{{ auditData.study_full_name }}</v-row>
          </router-link>
        </v-col>
      </td>
      <td class="text-left" v-if="item.type_of_week == tab || item.type_of_week ==
2"><span>{{ item.lesson_type }}</span></td>
    </tr>
  </tbody>
</template>
</v-simple-table>
</v-col>
</v-row>
<v-row no-gutters justify="center" class="display-1">
  <span>{{ $t('components.Thursday') }}</span>
</v-row>
  <v-row no-gutters>
    <v-col cols="12">
      <v-simple-table dense>
        <template v-slot:default>
          <thead>
            <tr>
              <th class="text-left">
                {{ $t('components.pair') }}
              </th>
              <th class="text-left">
                {{ $t('components.AcademicSubject') }}
              </th>
              <th class="text-left">
                {{ $t('components.Auditory') }}
              </th>

```

```

        <th class="text-left">
            {{ $t('components.Group') }}
        </th>
        <th class="text-left">
            {{ $t('components.LessonType') }}
        </th>
    </tr>
</thead>
<tbody>
    <tr
        sort-by="item.pare_number"
        v-for="item in Lessons.Thursday"
        :key="item.pare_number"
    >
        <td class="text-left" v-if="item.type_of_week == tab || item.type_of_week ==
2"><span>{{ item.pare_number }}</span></td>
        <td class="text-left" v-if="item.type_of_week == tab || item.type_of_week ==
2"><span>{{ item.discipline_full_name }}</span></td>
        <td class="text-left" v-if="item.type_of_week == tab || item.type_of_week == 2">
            <v-col cols="1">
                <router-link style="text-decoration: none;" class="pb-2" :to="{name: 'auditoriesResult', params:
{auditory: auditData.auditory_id}}" v-for=" auditData in item.Auditories" :key="auditData.auditory_id">
                    <v-row no-gutters>
                        {{ auditData.auditory_auditory_name }}
                    </v-row>
                </router-link>
            </v-col>
        </td>
        <td class="text-left" v-if="item.type_of_week == tab || item.type_of_week == 2">
            <v-col cols="1">
                <router-link style="text-decoration: none;" class="pb-2" :to="{name: 'group-result', params: {groupId:
auditData.study_group_id}}" v-for=" auditData in item.newData" :key="auditData.study_full_name">
                    <v-row no-gutters>{{ auditData.study_full_name }}</v-row>
                </router-link>
            </v-col>
        </td>
        <td class="text-left" v-if="item.type_of_week == tab || item.type_of_week ==
2"><span>{{ item.lesson_type }}</span></td>
    </tr>
</tbody>
</template>
</v-simple-table>
</v-col>
</v-row>

<v-row no-gutters justify="center" class="display-1">
    <span>{{ $t('components.Friday') }}</span>
</v-row>

    <v-row no-gutters>
        <v-col cols="12">
            <v-simple-table dense>
                <template v-slot:default>
                    <thead>
                        <tr>
                            <th class="text-left">
                                {{ $t('components.pair') }}
                            </th>
                            <th class="text-left">
                                {{ $t('components.AcademicSubject') }}
                            </th>
                            <th class="text-left">
                                {{ $t('components.Auditory') }}
                            </th>

```

```

<th class="text-left">
  {{${('components.Group')}}}
</th>
<th class="text-left">
  {{${('components.LessonType')}}}
</th>
</tr>
</thead>
<tbody>
  <tr
    v-for="item in Lessons.Friday"
    :key="item.pare_number"
  >
    <td class="text-left" v-if="item.type_of_week == tab || item.type_of_week ==
2"><span>{{item.pare_number}}</span></td>
    <td class="text-left" v-if="item.type_of_week == tab || item.type_of_week ==
2"><span>{{item.discipline_full_name}}</span></td>
    <td class="text-left" v-if="item.type_of_week == tab || item.type_of_week == 2">
      <v-col cols="1">
        <router-link style="text-decoration: none;" class="pb-2" :to="{name: 'auditoriesResult', params:
{auditory: auditData.auditory_id}}" v-for=" auditData in item.Auditories" :key="auditData.auditory_id">
          <v-row no-gutters>
            {{auditData.auditory_auditory_name}}
          </v-row>
        </router-link>
      </v-col>
    </td>
    <td class="text-left" v-if="item.type_of_week == tab || item.type_of_week == 2">
      <v-col cols="1">
        <router-link style="text-decoration: none;" class="pb-2" :to="{name: 'group-result', params: {groupId:
auditData.study_group_id}}" v-for=" auditData in item.newData" :key="auditData.study_full_name">
          <v-row no-gutters>{{auditData.study_full_name}}</v-row>
        </router-link>
      </v-col>
    </td>
    <td class="text-left" v-if="item.type_of_week == tab || item.type_of_week ==
2"><span>{{item.lesson_type}}</span></td>
  </tr>
</tbody>
</template>
</v-simple-table>
</v-col>
</v-row>
<v-row no-gutters justify="center" class="display-1">
  <span>{{${('components.Saturday')}}}</span>
</v-row>
  <v-row no-gutters>
    <v-col cols="12">
      <v-simple-table dense>
        <template v-slot:default>
          <thead>
            <tr>
              <th class="text-left">
                {{${('components.pair')}}}
              </th>
              <th class="text-left">
                {{${('components.AcademicSubject')}}}
              </th>
              <th class="text-left">
                {{${('components.Auditory')}}}
              </th>
              <th class="text-left">
                {{${('components.Group')}}}

```

```

        </th>
        <th class="text-left">
            {{ $t('components.LessonType') }}
        </th>
    </tr>
</thead>
<tbody>
    <tr
        v-for="item in Lessons.Saturday"
        :key="item.pare_number"
    >
        <td class="text-left" v-if="item.type_of_week == tab || item.type_of_week ==
2"><span>{{ item.pare_number }}</span></td>
        <td class="text-left" v-if="item.type_of_week == tab || item.type_of_week ==
2"><span>{{ item.discipline_full_name }}</span></td>
        <td class="text-left" v-if="item.type_of_week == tab || item.type_of_week == 2">
            <v-col cols="1">
                <router-link style="text-decoration: none;" class="pb-2" :to="{name: 'auditoriesResult', params:
{auditory: auditData.auditory_id}}" v-for=" auditData in item.Auditories" :key="auditData.auditory_id">
                    <v-row no-gutters>
                        {{ auditData.auditory_auditory_name }}
                    </v-row>
                </router-link>
            </v-col>
        </td>
        <td class="text-left" v-if="item.type_of_week == tab || item.type_of_week == 2">
            <v-col cols="1">
                <router-link style="text-decoration: none;" class="pb-2" :to="{name: 'group-result', params: {groupId:
auditData.study_group_id}}" v-for=" auditData in item.newData" :key="auditData.study_full_name">
                    <v-row no-gutters>{{ auditData.study_full_name }}</v-row>
                </router-link>
            </v-col>
        </td>
        <td class="text-left" v-if="item.type_of_week == tab || item.type_of_week ==
2"><span>{{ item.lesson_type }}</span></td>
    </tr>
</tbody>
</template>
</v-simple-table>
</v-col>
</v-row>
</v-container>
</template>

```

```

<script lang="ts">
import { Component, Prop } from 'vue-property-decorator';
import Vue from 'vue';
import axios from 'axios';
import EWeek from '@enums/dateEnums';
import EPare from '@enums/pareEnum';

```

```

@Component({})
export default class GroupResult extends Vue {
    @Prop({default: null})
    teacherId: number;

```

```

    private Week = EWeek;
    private Pare = EPare;
    private Lessons = null;
    private pair = [1,2,3,4,5,6,7,8];
    private MondayAudience = [];
    private tab: number = null;

```

```

private back() {
  this.$router.push({name: 'student-page'})
  this.Week.Sunday
}

private async mounted() {
  await axios.get('api/teacher/getTeacherLessonsInfo', {params: {groupId: this.teacherId}})
  .then(res => this.Lessons = res.data)
  this.Lessons.Monday.sort((a, b) => a.pare_number > b.pare_number ? 1 : -1);
  this.Lessons.Tuesday.sort((a, b) => a.pare_number > b.pare_number ? 1 : -1);
  this.Lessons.Wednesday.sort((a, b) => a.pare_number > b.pare_number ? 1 : -1);
  this.Lessons.Thursday.sort((a, b) => a.pare_number > b.pare_number ? 1 : -1);
  this.Lessons.Friday.sort((a, b) => a.pare_number > b.pare_number ? 1 : -1);
  this.Lessons.Saturday.sort((a, b) => a.pare_number > b.pare_number ? 1 : -1);
}
}
</script>

```

Б.15 Файл GroupResult.vue

```

<template>
  <v-container fluid>
    <v-tabs
      v-model="tab"
      centered
      dark>
      <v-tab>{{ $t('components.numerator') }}</v-tab>
      <v-tab>{{ $t('components.denominator') }}</v-tab>
    </v-tabs>
    <v-row no-gutters justify="center" class="display-1 pb-4">
      <span>{{ Lessons.GroupInfo.study_full_name }}</span>
    </v-row>
    <v-row no-gutters justify="center" class="display-1">
      <span>{{ $t('components.Monday') }}</span>
    </v-row>
    <v-row no-gutters>
      <v-col cols="12">
        <v-simple-table dense>
          <template v-slot:default>
            <thead>
              <tr>
                <th class="text-left">
                  {{ $t('components.pair') }}
                </th>
                <th class="text-left">
                  {{ $t('components.AcademicSubject') }}
                </th>
                <th class="text-left">
                  {{ $t('components.Auditory') }}
                </th>
                <th class="text-left">
                  {{ $t('nav.teacher') }}
                </th>
                <th class="text-left">
                  {{ $t('components.LessonType') }}
                </th>
              </tr>
            </thead>
            <tbody>
              <tr>

```

```

v-for="item in Lessons.Monday"
:key="item.pare_number"
>
<td class="text-left" style="width:20px" v-if="item.type_of_week == tab || item.type_of_week ==
2"><span>{{item.pare_number}}</span></td>
<td class="text-left" style="width:200px" v-if="item.type_of_week == tab || item.type_of_week ==
2"><span>{{item.discipline_full_name}}</span></td>
<td class="text-left" style="width:40px" v-if="item.type_of_week == tab || item.type_of_week == 2">
<v-col cols="1">
<router-link style="text-decoration: none;" class="pb-2" :to="{name: 'auditoriesResult', params:
{auditory: auditData.auditory_id}}" v-for=" auditData in item.Auditories" :key="auditData.auditory_id">
<v-row no-gutters>
{{auditData.auditory_auditory_name}}
</v-row>
</router-link>
</v-col>
</td>
<td class="text-left" style="width:200px" v-if="item.type_of_week == tab || item.type_of_week == 2">
<router-link style="text-decoration: none;" :to="{name: 'teacherResult', params: {teacherId:
item.teacher_id} }">
{{item.teacher_full_name}}
</router-link>
</td>
<td class="text-left" style="width:70px" v-if="item.type_of_week == tab || item.type_of_week ==
2"><span>{{item.lesson_type}}</span></td>
</tr>
</tbody>
</template>
</v-simple-table>
</v-col>
</v-row>
<v-row no-gutters justify="center" class="display-1">
<span>{{ $t('components.Tuesday')}}</span>
</v-row>
<v-row no-gutters>
<v-col cols="12">
<v-simple-table dense>
<template v-slot:default>
<thead>
<tr>
<th class="text-left">
{{ $t('components.pair')}}
</th>
<th class="text-left">
{{ $t('components.AcademicSubject')}}
</th>
<th class="text-left">
{{ $t('components.Auditory')}}
</th>
<th class="text-left">
{{ $t('nav.teacher')}}
</th>
<th class="text-left">
{{ $t('components.LessonType')}}
</th>
</tr>
</thead>
<tbody>
<tr>
v-for="item in Lessons.Thursday"
:key="item.pare_number"
>

```

```

        <td class="text-left" style="width:20px" v-if="item.type_of_week == tab || item.type_of_week ==
2"><span>{{item.pare_number}}</span></td>
        <td class="text-left" style="width:200px" v-if="item.type_of_week == tab || item.type_of_week ==
2"><span>{{item.discipline_full_name}}</span></td>
        <td class="text-left" style="width:40px" v-if="item.type_of_week == tab || item.type_of_week == 2">
            <v-col cols="1">
                <router-link style="text-decoration: none;" class="pb-2" :to="{name: 'auditoriesResult', params:
{auditory: auditData.auditory_id}}" v-for=" auditData in item.Auditories" :key="auditData.auditory_id">
                    <v-row no-gutters>
                        {{auditData.auditory_auditory_name}}
                    </v-row>
                </router-link>
            </v-col>
        </td>
        <td class="text-left" style="width:200px" v-if="item.type_of_week == tab || item.type_of_week == 2">
            <router-link style="text-decoration: none;" :to="{name: 'teacherResult', params: {teacherId:
item.teacher_id}}">
                {{item.teacher_full_name}}
            </router-link>
        </td>
        <td class="text-left" style="width:70px" v-if="item.type_of_week == tab || item.type_of_week ==
2"><span>{{item.lesson_type}}</span></td>
    </tr>
</tbody>
</template>
</v-simple-table>
</v-col>
</v-row>
<v-row no-gutters justify="center" class="display-1">
    <span>{{t('components.Wednesday')}}</span>
</v-row>
<v-row no-gutters>
    <v-col cols="12">
        <v-simple-table dense>
            <template v-slot:default>
                <thead>
                    <tr>
                        <th class="text-left">
                            {{t('components.pair')}}
                        </th>
                        <th class="text-left">
                            {{t('components.AcademicSubject')}}
                        </th>
                        <th class="text-left">
                            {{t('components.Auditory')}}
                        </th>
                        <th class="text-left">
                            {{t('nav.teacher')}}
                        </th>
                        <th class="text-left">
                            {{t('components.LessonType')}}
                        </th>
                    </tr>
                </thead>
                <tbody>
                    <tr>
                        v-for="item in Lessons.Wednesday"
                        :key="item.pare_number"
                    >
                        <td class="text-left" style="width:20px" v-if="item.type_of_week == tab || item.type_of_week ==
2"><span>{{item.pare_number}}</span></td>
                        <td class="text-left" style="width:200px" v-if="item.type_of_week == tab || item.type_of_week ==
2"><span>{{item.discipline_full_name}}</span></td>

```

```

        <td class="text-left" style="width:40px" v-if="item.type_of_week == tab || item.type_of_week == 2">
            <v-col cols="1">
                <router-link style="text-decoration: none;" class="pb-2" :to="{name: 'auditoriesResult', params:
{auditory: auditData.auditory_id}}" v-for=" auditData in item.Auditories" :key="auditData.auditory_id">
                    <v-row no-gutters>
                        {{auditData.auditory_auditory_name}}
                    </v-row>
                </router-link>
            </v-col>
        </td>
        <td class="text-left" style="width:200px" v-if="item.type_of_week == tab || item.type_of_week == 2">
            <router-link style="text-decoration: none;" :to="{name: 'teacherResult', params: {teacherId:
item.teacher_id} }">
                {{item.teacher_full_name}}
            </router-link>
        </td>
        <td class="text-left" style="width:70px" v-if="item.type_of_week == tab || item.type_of_week ==
2"><span>{{item.lesson_type}}</span></td>
    </tr>
</tbody>
</template>
</v-simple-table>
</v-col>
</v-row>
<v-row no-gutters justify="center" class="display-1">
    <span>{{ $t('components.Thursday')}}</span>
</v-row>
<v-row no-gutters>
    <v-col cols="12">
        <v-simple-table dense>
            <template v-slot:default>
                <thead>
                    <tr>
                        <th class="text-left">
                            {{ $t('components.pair')}}
                        </th>
                        <th class="text-left">
                            {{ $t('components.AcademicSubject')}}
                        </th>
                        <th class="text-left">
                            {{ $t('components.Auditory')}}
                        </th>
                        <th class="text-left">
                            {{ $t('nav.teacher')}}
                        </th>
                        <th class="text-left">
                            {{ $t('components.LessonType')}}
                        </th>
                    </tr>
                </thead>
                <tbody>
                    <tr
                        sort-by="item.pare_number"
                        v-for="item in Lessons.Thursday"
                        :key="item.pare_number"
                    >
                        <td class="text-left" style="width:20px" v-if="item.type_of_week == tab || item.type_of_week ==
2"><span>{{item.pare_number}}</span></td>
                        <td class="text-left" style="width:200px" v-if="item.type_of_week == tab || item.type_of_week ==
2"><span>{{item.discipline_full_name}}</span></td>
                        <td class="text-left" style="width:40px" v-if="item.type_of_week == tab || item.type_of_week == 2">
                            <v-col cols="1">

```



```

        {{auditData.auditory_auditory_name}}
      </v-row>
    </router-link>
  </v-col>
</td>
<td class="text-left" style="width:200px" v-if="item.type_of_week == tab || item.type_of_week == 2">
  <router-link style="text-decoration: none;" :to="{name: 'teacherResult', params: {teacherId:
item.teacher_id} }">
    {{item.teacher_full_name}}
  </router-link>
</td>
<td class="text-left" style="width:70px" v-if="item.type_of_week == tab || item.type_of_week ==
2"><span>{{item.lesson_type}}</span></td>
</tr>
</tbody>
</template>
</v-simple-table>
</v-col>
</v-row>
<v-row no-gutters justify="center" class="display-1">
  <span>{{ $t('components.Saturday')}}</span>
</v-row>
  <v-row no-gutters>
    <v-col cols="12">
      <v-simple-table dense>
        <template v-slot:default>
          <thead>
            <tr>
              <th class="text-left">
                {{ $t('components.pair')}}
              </th>
              <th class="text-left">
                {{ $t('components.AcademicSubject')}}
              </th>
              <th class="text-left">
                {{ $t('components.Auditory')}}
              </th>
              <th class="text-left">
                {{ $t('nav.teacher')}}
              </th>
              <th class="text-left">
                {{ $t('components.LessonType')}}
              </th>
            </tr>
          </thead>
          <tbody>
            <tr>
              v-for="item in Lessons.Saturday"
              :key="item.pare_number"
            >
              <td class="text-left" style="width:20px" v-if="item.type_of_week == tab || item.type_of_week ==
2"><span>{{ item.pare_number}}</span></td>
              <td class="text-left" style="width:200px" v-if="item.type_of_week == tab || item.type_of_week ==
2"><span>{{ item.discipline_full_name}}</span></td>
              <td class="text-left" style="width:40px" v-if="item.type_of_week == tab || item.type_of_week == 2">
                <v-col cols="1">
                  <router-link style="text-decoration: none;" class="pb-2" :to="{name: 'auditoriesResult', params:
{auditory: auditData.auditory_id} }" v-for=" auditData in item.Auditories" :key="auditData.auditory_id">
                    <v-row no-gutters>
                      {{auditData.auditory_auditory_name}}
                    </v-row>
                  </router-link>
                </v-col>

```

```

        </td>
        <td class="text-left" style="width:200px" v-if="item.type_of_week == tab || item.type_of_week == 2">
            <router-link style="text-decoration: none;" :to="{name: 'teacherResult', params: {teacherId:
item.teacher_id} }">
                {{item.teacher_full_name}}
            </router-link>
        </td>
        <td class="text-left" style="width:70px" v-if="item.type_of_week == tab || item.type_of_week ==
2"><span>{{item.lesson_type}}</span></td>
    </tr>
</tbody>
</template>
</v-simple-table>
</v-col>
</v-row>
</v-container>
</template>

```

```

<script lang="ts">
import { Component, Prop } from 'vue-property-decorator';
import Vue from 'vue';
import axios from 'axios';
import EWeek from '@enums/dateEnums';
import EPare from '@enums/pareEnum';

@Component({})
export default class GroupResult extends Vue {
    @Prop({default: null})
    groupId: number;

    private Week = EWeek;
    private Pare = EPare;
    private Lessons = null;
    private pair = [1,2,3,4,5,6,7,8];
    private tab: number = null;

    private back() {
        this.$router.push({name: 'student-page'})
        this.Week.Sunday
    }

    private async mounted() {
        await axios.get('api/student/getStudentLessonsInfo', {params: {groupId: this.groupId}})
        .then(res => this.Lessons = res.data)
        this.Lessons.Monday.sort((a, b) => a.pare_number > b.pare_number ? 1 : -1);
        this.Lessons.Tuesday.sort((a, b) => a.pare_number > b.pare_number ? 1 : -1);
        this.Lessons.Wednesday.sort((a, b) => a.pare_number > b.pare_number ? 1 : -1);
        this.Lessons.Thursday.sort((a, b) => a.pare_number > b.pare_number ? 1 : -1);
        this.Lessons.Friday.sort((a, b) => a.pare_number > b.pare_number ? 1 : -1);
        this.Lessons.Saturday.sort((a, b) => a.pare_number > b.pare_number ? 1 : -1);
    }
}
</script>

```

Б.16 Файл index.vue

```

import i18n from '@i18n'
import Vue from 'vue'
import VueRouter, { RouteConfig } from 'vue-router'

```

Vue.use(VueRouter)

```
const routes: Array<RouteConfig> = [
  {
    path: '/',
    redirect: `${i18n.locale}/teacher`
  },
  {
    path: '/:lang',
    component: {
      render(c) {return c('router-view')}
    },
    children: [
      {
        path: 'about',
        name: 'About',
        // route level code-splitting
        // this generates a separate chunk (about.[hash].js) for this route
        // which is lazy-loaded when the route is visited.
        component: () => import(/* webpackChunkName: "about" */ './views/About.vue')
      },
      {
        path: 'teacher',
        name: 'teacher',
        component: () => import(/* webpackChunkName: "teacher" */ './views/Teacher.vue')
      },
      {
        path: 'teacher/:teacherId',
        name: 'teacherResult',
        props: true,
        component: () => import(/* webpackChunkName: "teacher" */ './views/TeacherResult/ResultTeacher.vue')
      },
      {
        path: 'bells',
        name: 'bells',
        component: () => import(/* webpackChunkName: "bells" */ '@views/Bells.vue')
      },
      {
        path: '404',
        name: "notFound",
        component: () => import(/* webpackChunkName: "notFound" */ '@components/NotFound/NotFound.vue')
      },
      {
        path: 'auditories',
        props: true,
        name: 'auditories',
        component: () => import(/* webpackChunkName: "auditories" */ '@views/Auditories.vue')
      },
      {
        path: 'auditories/:auditory',
        props: true,
        name: 'auditoriesResult',
        component: () => import(/* webpackChunkName: "auditories" */ '@views/AuditoryResult.vue')
      },
      {
        path: 'student',
        name: "student",
        props: true,
        component: {
          render(c) {return c('router-view')}
        },
        children: [
          {
```

```

      name: 'group-result',
      path: 'group/:groupId?',
      props: true,
      component: () => import(/* webpackChunkName: "student" */'@/views/ResultGroup/GroupResult.vue')
    },
    {
      name: 'student-page',
      path: '/',
      component: () => import(/* webpackChunkName: "student" */'@/views/Student.vue'),
    }
  ]
},
{
  path: '*',
  redirect: '404'
},
],
},
]

const router = new VueRouter({
  mode: 'history',
  base: process.env.BASE_URL,
  routes
})

export default router

```

Б.17 Файл `AuditoryModel.ts`

```

export default class Auditory {
  public auditory_id: number;
  public auditory_name: string;
  public not_used: boolean;
}

```

Б.18 Файл `AuditoryResultModel.ts`

```

export default class AuditoryResult {
  public auditory_name: string;
  public not_used: number;
  public auditory_type_name: string;
  public count_of_places: number;
  public corps_number: number;
  public full_name: string;
  public short_name: string;
  public faculty_code: string;
}

```

Б.19 Файл `GroupModel.ts`

```

export default class Group {
  public count_of_students: number;
  public course_number: string;
  public department_id: number;
  public full_name: string;
  public study_group_code: string;
}

```

```

    public study_group_id: number;
}

```

Б.20 Файл en.json

```

{
  "nav": {
    "scheidung": "Scheidung",
    "teacher": "Teacher",
    "student": "Student",
    "auditories": "Audience",
    "bells": "Bells",
    "AllRightsReserved": "All rights reserved",
    "about": "About",
    "ZaporizhzhiaPolytechnic": "Zaporizhzhia Polytechnic"
  },
  "components": {
    "audienceSearch": "audience search",
    "enterAudienceNumber": "Enter audience number:",
    "find": "Find",
    "pair": "Pair",
    "start": "Start",
    "end": "End",
    "break": "Break",
    "bellsSchedule": "Bells schedule",
    "aboutSite": "About site",
    "siteDeveloper": "Site developer",
    "TarasKuznietsov": "Taras Kuznietsov (KNT-117)",
    "thisSiteIsDesignedTo": "The site is designed to inform students and teachers about the lessons schedule.",
    "group": "group",
    "course": "course",
    "faculty": "faculty",
    "department": "department",
    "teacher": "teacher",
    "studentsGroupMustBeSelected": "Student's group must be selected!",
    "auditoriesSearch": "Search audiences",
    "auditoryName": "Audience",
    "auditoryCorpus": "Education building",
    "auditoryType": "Audience type",
    "auditoryFacultyName": "Faculty name",
    "auditoryPlaces": "Quantity of places",
    "PleaseSelectedAudience": "Please select the audience",
    "AudienceDoesntExist": "Audience doesn't found!",
    "Sunday": "Sunday",
    "Monday": "Monday",
    "Tuesday": "Tuesday",
    "Wednesday": "Wednesday",
    "Thursday": "Thursday",
    "Friday": "Friday",
    "Saturday": "Saturday",
    "AcademicSubject": "Academic subject",
    "Auditory": "Audience",
    "LessonType": "Lesson type",
    "Group": "Group",
    "numerator": "numerator",
    "denominator": "denominator"
  }
}

```

Б.21 Файл ua.json

```
{
  "nav": {
    "schedule": "Розклад",
    "teacher": "Викладач",
    "bells": "Дзвінки",
    "AllRightsReserved": "Усі права захищено",
    "about": "Про нас",
    "ZaporizhzhiaPolytechnic": "Запорізька Політехніка",
    "auditories": "Аудиторії",
    "student": "Студент"
  },
  "components" : {
    "audienceSearch" : "Пошук Аудиторії",
    "enterAudienceNumber": "Впишіть номер аудиторії:",
    "find": "Пошук",
    "pair": "Пара",
    "start": "Початок",
    "end": "Кінець",
    "break": "Перерва",
    "bellsSchedule": "Розклад дзвінків",
    "aboutSite": "Про сайт",
    "siteDeveloper": "Розробник сайту",
    "TarasKuznietsow": "Тарас Кузнецов (КНТ-117)",
    "thisSiteIsDesignedTo": "Сайт розроблено для інформування студентів та викладачів щодо розкладу занять.",
    "group": "група",
    "course": "курс",
    "faculty": "факультет",
    "department": "кафедра",
    "teacher": "викладач",
    "studentsGroupMustBeSelected": "Група студента повинна бути вибрана!",
    "auditoriesSearch": "Пошук Аудиторії",
    "auditoryName": "Аудиторія",
    "auditoryCorpus": "Корпус",
    "auditoryType": "Тип аудиторії",
    "auditoryFacultyName": "Назва факультету",
    "auditoryPlaces": "Кількість місць",
    "PleaseSelectedAudience": "Будь ласка виберіть аудиторію",
    "AudienceDoesntExist": "Аудиторію не знайдено!",
    "Sunday": "Неділя",
    "Monday": "Понеділок",
    "Tuesday": "Вівторок",
    "Wednesday": "Середа",
    "Thursday": "Четвер",
    "Friday": "П'ятниця",
    "Saturday": "Суббота",
    "AcademicSubject": "Предмет",
    "Auditory": "Аудиторія",
    "LessonType": "Тип уроку",
    "Group": "Група",
    "numerator": "чисельник",
    "denominator": "знаменник"
  }
}
```

Б.22 Файл courpusesEnums.ts

```
enum ECourpuse {
  FirstCorpuse = 1,
  SecondCorpus = 2,
  ThirdCorpus = 3,
  FourthCorpus = 4,
  FifthCorpus = 5,
  SportCorpus = 6
}

export default ECourpuse;
```

Б.23 Файл dateEnums.ts

```
enum EWeek {
  Sunday = 1,
  Monday = 2,
  Tuesday = 3,
  Wednesday = 4,
  Thursday = 5,
  Friday = 6,
  Saturday = 7
}

export default EWeek;
```

Б.24 Файл pateEnums.ts

```
enum EPare {
  FirsPare = 1,
  SecondPare = 2,
  ThirdPare = 3,
  FourthPare = 4,
  FifthPare = 5,
  SixthPare = 6,
  SeventhPare = 7,
  EighthPare = 8,
  NinethPare = 9
}

export default EPare;
```

Б.25 Файл data.json

```
[
  {
    "pair" : "I",
    "start" : "8:30",
    "end" : "9:50",
    "break": "15"
```

```

    },
    {
      "pair" : "II",
      "start" : "10:05",
      "end" : "11:25",
      "break": "10"
    },
    {
      "pair" : "III",
      "start" : "11:55",
      "end" : "13:15",
      "break": "10"
    },
    {
      "pair" : "IV",
      "start" : "13:25",
      "end" : "14:45",
      "break": "10"
    },
    {
      "pair" : "I",
      "start" : "14:55",
      "end" : "16:15",
      "break": "30"
    },
    {
      "pair" : "VI",
      "start" : "16:45",
      "end" : "18:05",
      "break": "10"
    },
    {
      "pair" : "VII",
      "start" : "18:15",
      "end" : "19:35",
      "break": "10"
    },
    {
      "pair" : "VIII",
      "start" : "19:45",
      "end" : "21:05",
      "break": "-"
    }
  ]

```

Б.26 Файл components/Auditories.vue

```

<template>
  <v-container fluid wrap dark style="height:50vh; background: #212529">
    <v-row no-gutters justify="center" class="title" style="background: #212529">
      <span style="color: white">{{ $t("components.audienceSearch").toUpperCase() }} </span>
    </v-row>
    <v-row no-gutters align="center" class="fill-height" justify="center">
      <v-col cols="12">
        <v-row no-gutters justify="center" align="center">
          <v-col md="12" lg="5" xl="5" xs="12" sm="12" cols="12">
            <v-row no-gutters class="title" justify="center" justify-lg="end" justify-xl="end">
              <span style="color:white;">{{ $t('components.enterAudienceNumber')}} </span>
            </v-row>
          </v-col>
        </v-row>
      </v-col>
    </v-row>
  </v-container>

```

```

<v-col md="5" lg="2" xl="2" xs="5" sm="5" cols="5" class="pa-4">
  <v-row no-gutters>
    <v-autocomplete
      return-object
      v-model="seletedAuditorie"
      :items="auditories"
      item-text="auditory_name"
      item-value="auditory_id"
      dark
      class="align-center justify-center">
    </v-autocomplete>
  </v-row>
</v-col>
<v-col md="12" lg="5" xl="5" xs="12" sm="12" cols="12">
  <v-row no-gutters justify="center" justify-xl="start" justify-lg="start">
    <v-btn large tile color="blue-grey" class="white--text" @click="GoToAuditorie"
elevation=4>{{ $t('components.find')}} </v-btn>
  </v-row>
</v-col>
</v-row>

</v-col>
</v-row>
</v-container>
</template>

<script lang="ts">
import Vue from 'vue'
import Component from 'vue-class-component';
import axios from 'axios';
import AuditoryModel from '@models/auditoryModel';

@Component({
})
export default class AuditoriesComponent extends Vue {
  private auditories: Array<AuditoryModel> = [];
  private seletedAuditorie: AuditoryModel = {} as AuditoryModel;
  private loader = false;

  private created() {
    this.loader = true;
  }

  private async mounted() {
    axios.get('api/auditories/getAuditories')
      .then(res => this.auditories = res.data);
    this.loader = false;
  }

  private GoToAuditorie() {
    try {
      if(this.seletedAuditorie.auditory_id) {
        this.$router.push({name: 'auditoriesResult', params: {auditory: this.seletedAuditorie.auditory_id}})
      }
      else {
        throw new Error(this.$t('components.PleaseSelectedAudience').toString());
      }
    }
    catch(e) {
      Vue.toasted.error(e.message);
    }
  }
}

```

```

    }
  }
</script>

```

Б.26 Файл NotFound.vue

```

<template>
  <center>
    <h1>Not Found!</h1>
    <p>
      <a href="/">Go home?</a>
    </p>
  </center>
</template>

```

```

<script>

export default {
  name: 'NotFound'
}

```

```

</script>

```

```

<style scoped>

```

```

  center {
    margin: 15vw;
  }

```

```

  h1 {
    color: var(--border);
    font-size: 2em;
  }

```

```

</style>

```

Б.27 Файл StudentFound.vue

```

<template>
  <v-container fluid class="pt-10 pb-10" style="background:#212529">
    <v-row no-gutters>
      <v-col sm="12" xs="12" md="9" cols="12">
        <v-row no-gutters>
          <v-col class="px-2 title" sm="12" xs="12" md="6" lg="6" xl="6" cols="12" style="color:white; text-
transform: uppercase;">
            <v-row justify="center">
              {{ $t('components.course') }}
            </v-row>
            <v-autocomplete
              v-model="courseSelected"
              :items = "courses"
              dark></v-autocomplete>
          </v-col>
          <v-col class="px-2 title" sm="12" xs="12" md="6" lg="6" xl="6" cols="12" style="color:white; text-
transform: uppercase;">
            <v-row justify="center" >
              {{ $t('components.group') }}
            </v-row>

```

```

      <v-autocomplete
      :items="groups.filter(el => courseSelected ? el.course_number == courseSelected: el)"
      item-text="full_name"
      v-model="selectedGroup"
      return-object
      dark></v-autocomplete>
    </v-col>
  </v-row>
</v-col>
<v-col sm="12" xs="12" md="3" lg="3" xl="3" cols="12" class="fill-height" align-self="center">
  <v-row justify="center" class="fill-height">
    <v-btn large tile color="blue-grey" class="white--text" @click="teacherFind"
elevation=4>{{ $t('components.find')}}</v-btn>
  </v-row>
</v-col>
</v-row>
</v-container>
</template>

```

```

<script lang="ts">
import Vue from 'vue';
import Component from 'vue-class-component';
import axios from 'axios';
import GroupModel from '@models/groupModel';

@Component({
})

export default class StudentFound extends Vue{

  private groups: Array<GroupModel> = [];
  private courseSelected = "";
  private courses = [];
  private selectedGroup: GroupModel = {} as GroupModel;
  private loader = false;

  private async created() {
    this.loader = true;
  }

  private async mounted()
  {
    this.courses = ['1','2','3','4','5'];
    await axios.get('api/student/getStudyGroups')
    .then((res) => {this.groups = res.data;})
    .catch((err) => Vue.toasted.error(err.response.data))
    this.loader = false;
  }

  private teacherFind() {
    try {
      if(this.selectedGroup.study_group_id)
      {
        this.$router.push({name: 'group-result', params: {groupId: this.selectedGroup.study_group_id}})
      }
      else
      {
        throw new Error(this.$t('components.studentsGroupMustBeSelected').toString());
      }
    }
    catch(ex) {

```

```

    return Vue.toasted.error(ex.message);
  }
}
</script>

```

Б.28 Файл TeacherFound.vue

```

<template>
  <v-container fluid class="pt-10 pb-10" style="background:#212529">
    <v-row no-gutters>
      <v-col sm="12" xs="12" md="9" cols="12">
        <v-row no-gutters>
          <v-col class="px-2 title" sm="12" xs="12" md="4" lg="4" xl="4" cols="12" style="color:white; text-
transform: uppercase;">
            <v-row justify="center">
              {{ $t('components.faculty')}}
            </v-row>
            <v-autocomplete
              v-model="selectedFacults"
              :items = "facults"
              item-text = "full_name"
              item-value = "id"
              return-object
              dark></v-autocomplete>
          </v-col>
          <v-col class="px-2 title" sm="12" xs="12" md="4" lg="4" xl="4" cols="12" style="color:white; text-
transform: uppercase;">
            <v-row justify="center" >
              {{ $t('components.department')}}
            </v-row>
            <v-autocomplete
              v-model="selectedDepartment"
              :items="department.filter(el => this.selectedFacults ? el.faculty_id == this.selectedFacults.faculty_id : el)"
              item-text="full_name"
              item-value="id"
              return-object
              dark></v-autocomplete>
          </v-col>
          <v-col class="px-2 title" sm="12" xs="12" md="4" lg="4" xl="4" cols="12" style="color:white; text-
transform: uppercase;">
            <v-row justify="center" >
              {{ $t('components.teacher')}}
            </v-row>
            <v-autocomplete
              v-model="selectedTeacher"
              :items="teachers.filter(el => (this.selectedFacults ? el.faculty_id == this.selectedFacults.faculty_id : el) ||
(this.selectedDepartment ? el.department_id == this.selectedDepartment.department_id : el) )"
              item-value="id"
              item-text="teache_name"
              return-object
              dark></v-autocomplete>
          </v-col>
        </v-row>
      </v-col>
      <v-col sm="12" xs="12" md="3" lg="3" xl="3" cols="12" class="fill-height" align-self="center">
        <v-row justify="center" class="fill-height">
          <v-btn @click="teacherFind" large tile color="blue-grey" class="white--text"
elevation=4>{{ $t('components.find')}}</v-btn>
        </v-row>
      </v-col>
    </v-row>
  </v-container>

```

```

    </v-col>
  </v-row>
</v-container>
</template>

<script lang="ts">
import Vue from 'vue';
import Component from 'vue-class-component';
import axios from 'axios';
@Component({
})

export default class TeacherFound extends Vue {
  private teachers: Array<object> = [];
  private facults: Array<object> = [];
  private department: Array<object> = [];
  private loader = false;

  private selectedTeacher: object = {};
  private selectedFacults: object = null;
  private selectedDepartment: object = null;

  private async created()
  {
    this.loader = true;
  }
  private async mounted()
  {
    await axios.get('api/teacher/getTeacher')
    .then(res => {
      this.teachers = res.data.teacherData;
      this.facults = res.data.facultyData;
      this.department = res.data.departmentData;
      this.loader = false;
    })
  }

  private teacherFind() {
    console.log()
  }
}
</script>

```

Б.29 Файл AppBar.vue

```

<template>
<div>
  <v-app-bar app>
    <v-navigation-drawer
      v-model="drawer"
      absolute
      temporary
      app
    >
    <v-list v-if="window.width <= 800">
      <v-list-item-group
        v-model="group"
      >
        <router-link v-for="items in rotePaths" :key="items" :to="'/${$118n.locale}/${items}'" tag="div">
          <v-list-item>
            <v-list-item-title>{{ $(`nav.${items}`) }}</v-list-item-title>
          </v-list-item>
        </router-link>
      </v-list-item-group>
    </v-list>
  </div>

```

```

    </v-list-item-group>
  </v-list>
</v-navigation-drawer>
<div class="d-flex align-center">
  <v-img
    alt="Vuetify Logo"
    class="shrink mr-2"
    contain
    :src="require('@assets/nuzp.png')"
    transition="scale-transition"
    width="40"
  />
  <span class="NUZP-title">{{ $t('nav.ZaporizhzhiaPolytechnic')}} </span>
</div>
<v-spacer></v-spacer>
<v-app-bar-nav-icon color="white" @click="drawer = true" v-if="window.width <= 800"></v-app-bar-nav-icon>

  <div v-if="window.width > 800">
    <router-link v-for="items in rotePaths" :key="items" :name="items" :to="'/' + $18n.locale + '/' + items" >{{ $t('nav.' + items)}} </router-link>
  </div>

</v-app-bar>
</div>
</template>

<script>
export default {
  data() {
    return { window: {
      width: 0,
      height: 0
    },
    drawer: false,
    group: null,
    rotePaths: ["student", "teacher", "auditories", "bells", "about"]
  },
  created() {
    window.addEventListener('resize', this.handleResize);
    this.handleResize();
  },
  destroyed() {
    window.removeEventListener('resize', this.handleResize);
  },
  methods: {
    handleResize() {
      this.window.width = window.innerWidth;
      this.window.height = window.innerHeight;
    }
  }
}
</script>

<style scoped>
div.router-link-exact-active
{
  background:rgba(255, 255, 255, 0.5);;
}
a.router-link-exact-active

```

```

{
  color: white !important;
}
a
{
  text-decoration: none;
  font-size: 17px;
  font-family: 'Poppins', sans-serif;
  margin-left: 10px;
  color: rgb(139,147,155) !important;
  letter-spacing: 2px;
  text-transform: uppercase;
}
.v-sheet.v-app-bar.v-toolbar:not(.v-sheet--outlined)
{
  background: rgb(52,58,64);
}
.NUZP-title
{
  width: 25px;
  text-transform: uppercase;
  font-family: 'Poppins', sans-serif;
  font-size: 20px;
  color: rgb(243,196,92);
}
.v-navigation-drawer
{
  background: rgb(52,58,64) !important;
  color: azure;
}
.v-list-item__title{
  color: white;
  text-transform: uppercase;
  padding: 10px;
}
</style>

```

Б.30 Файл Footer.vue

```

<template>
  <v-footer
    dark
    padless
    app
    fixed
    flat
  >
    <v-card
      class="flex"
      flat
      tile
    >
      <v-card-actions>
        <v-list-item class="grow">
          <v-row no-gutters justify="start" align="center">
            <v-img
              alt="Vuetify Logo"
              class="shrink mr-2"
              contain
              :src="require('@/assets/nuzp.png')"
              transition="scale-transition"
            >

```

```

        width="40"
      />
      {{ new Date().getFullYear() }} <strong>©{{ $t('nav.AllRightsReserved') }}</strong>
    </v-row>
    <v-menu>
      <template v-slot:activator="{ attrs, on }">
        <v-btn
          v-bind="attrs"
          v-on="on"
        >
          <v-img :src="require(`@/assets/svg/${$i18n.locale}.svg`)" width="38px" height="24px"/>
        </v-btn>
      </template>
      <v-list>
        <v-list-item-group v-model="model">
          <v-list-item
            v-for="item in items"
            :key="item.langCode"
            link
            :value="item.langCode"
          >
            <v-img :src="require(`@/assets/svg/${item.langCode}.svg`)" width="38px" height="24px"/>
            <v-list-item-title v-text="item.langName"></v-list-item-title>
          </v-list-item>
        </v-list-item-group>
      </v-list>
    </v-menu>
  </v-list-item>
</v-card-actions>
</v-card>
</v-footer>
</template>

<script lang="ts">
import Vue from 'vue'
import Component from 'vue-class-component';
import { Watch } from 'vue-property-decorator';
@Component({

})
export default class Footer extends Vue {
  private items=[
    {'langName': 'English', 'langCode': 'en'},
    {'langName': 'Ukraine', 'langCode': 'ua'},
  ];
  private model = this.$i18n.locale;

  @Watch('model')
  private onChangeLanguage(value: string)
  {
    if(value !== this.$i18n.locale && value)
    {
      this.$router.push({
        params: {lang: value}
      });
    }
  }
}
}
</script>

```

```

<style scoped>
.v-sheet.v-footer:not(.v-sheet--outlined)
{
  height: 64px;
  background: rgb(52,58,64);
  width: 100vw;
}
.v-sheet.v-card:not(.v-sheet--outlined)
{
  background: rgb(52,58,64);
}
.v-btn:not(.v-btn--round).v-size--default
{
  min-width: 0;
  max-width: 38px;
  height: 24px;
}
</style>

```

Б.31 Файл loader.vue

```

<template>
<div class="text-center">
  <v-overlay>
    <v-progress-circular
      indeterminate
      size="64"
    ></v-progress-circular>
  </v-overlay>
</div>
</template>

```

```

<script>
export default {
}
</script>

```

ДОДАТОК В
Слайди презентації

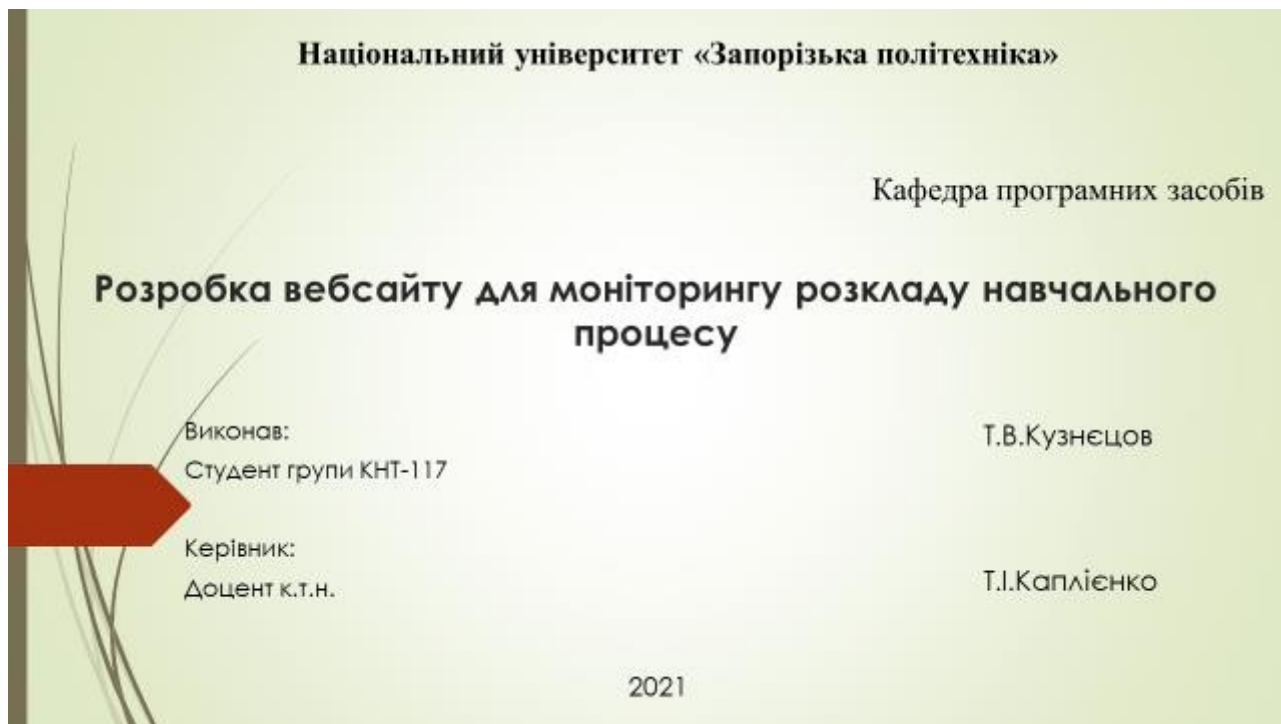


Рисунок В.1 – Слайд № 1



Рисунок В.2 – Слайд № 2

Вибір мови програмування 3

Критерії порівняння	JavaScript	TypeScript	PHP	Python
Мова створювалася спеціально для аналізу даних	+	-	-	-
Наявність динамічної типізація	+	++	+	+
Можливість статичної типізації	+	-	-	-
Наявність зрозумілої та повної документації	+	+	+	+
Можливість розробки серверної частини	+	+	+	+
Можливість розробки клієнтської частини	+	+	-	-
Продуктивність роботи	+	+	+	+
Можливість швидкого підключення сторонніх бібліотек	++	+	+	+
Наявність великої кількості наборів сторонніх пакетів	++	-	+	-
Назва пакетного менеджера	NPM	NPM	Composer	Pip
Взаємодія з іншими мовами	+	+	+	+

Рисунок В.3 – Слайд № 3

Вибір середовища проєктування серверної частини 4

Критерії порівняння	Koa	Meteor	Express	Loopback
Низький поріг входження	-	-	+	-
Підтримка TypeScript	+	-	++	-
Можливість статичної типізації	+	-	-	-
Наявність зрозумілої та повної документації	-	+	+	+
Легке підключення до бази даних	-	-	+	-
Можливість розробки клієнтської частини	+	-	+	+
Продуктивність роботи	+	+	++	+

Рисунок В.4 – Слайд № 4

Вибір середовища проектування клієнтської частини⁵

Критерії порівняння	React	Vue.js	Angular
Низький поріг входження	-	+	-
Підтримка TypeScript	+	++	+
Підтримка SPA	+	+	+
Наявність зрозумілої та повної документації	+	+	+
Легке підключення графічних фреймворків	-	+	-
Легке підключення сторонніх бібліотек	+	+	-
Продуктивність роботи	+	++	+
Управління станами	+	+	+

Рисунок В.5 – Слайд № 5

Архітектура клієнтської частини вебзастосунку⁶

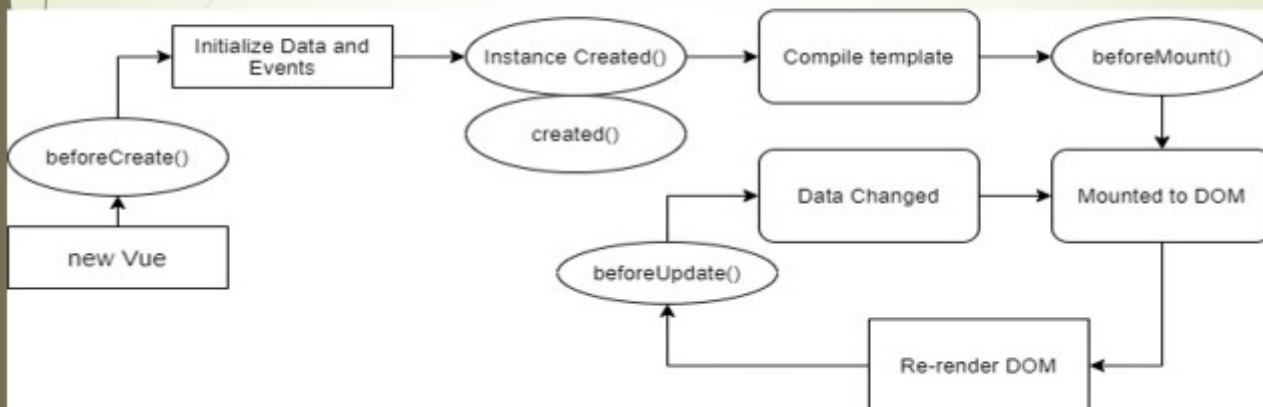


Рисунок В.6 – Слайд № 6



Рисунок В.7 – Слайд № 7



Рисунок В.8 – Слайд № 8

Приклад системних повідомлень програмісту

```

Error: Access denied for user 'zp@localhost' (using password: YES)
    at Packet.asError (/home/taras/javascript/rozklad-diplom/node_modules/mysql2/lib/packets/packet.js:712:17)
    at ClientHandshake.execute (/home/taras/javascript/rozklad-diplom/node_modules/mysql2/lib/commands/command.js:28:26)
    at Connection.handlePacket (/home/taras/javascript/rozklad-diplom/node_modules/mysql2/lib/connection.js:425:32)
    at PacketParser.onPacket (/home/taras/javascript/rozklad-diplom/node_modules/mysql2/lib/connection.js:175:12)
    at PacketParser.executeStart (/home/taras/javascript/rozklad-diplom/node_modules/mysql2/lib/packet_parser.js:75:16)
    at Socket.<anonymous> (/home/taras/javascript/rozklad-diplom/node_modules/mysql2/lib/connection.js:82:25)
    at Socket.emit (node:events:370:28)
    at addChunk (node:internal/streams/readable:313:12)
    at readableAddChunk (node:internal/streams/readable:288:9)
    at Socket.Readable.push (node:internal/streams/readable:227:10) {
  code: 'ER_ACCESS_DENIED_ERROR',
  errno: 1045,
  sqlState: '28000',
  sqlMessage: 'Access denied for user 'zp@localhost' (using password: YES)'
}
відсутність доступу до бази даних!

```

```

Error: connect ECONNREFUSED 127.0.0.1:3306
    at TCPConnectWrap.afterConnect [as oncomplete] (node:net:1137:16) {
  errno: -111,
  code: 'ECONNREFUSED',
  syscall: 'connect',
  address: '127.0.0.1',
  port: 3306,
  fatal: true
}
відсутність підключення до бази даних!

```

```

[connection] starting node address: 127.0.0.1
сервер запущено на порті: 3000

```

Рисунок В.9 – Слайд № 9

Приклад системних повідомлень оператору

Будь ласка виберіть викладача!

Please select the teacher

Будь ласка виберіть аудиторію

Please select the audience

Група студента повинна бути вибрана!

Student's group must be selected!

Рисунок В.10 – Слайд № 10

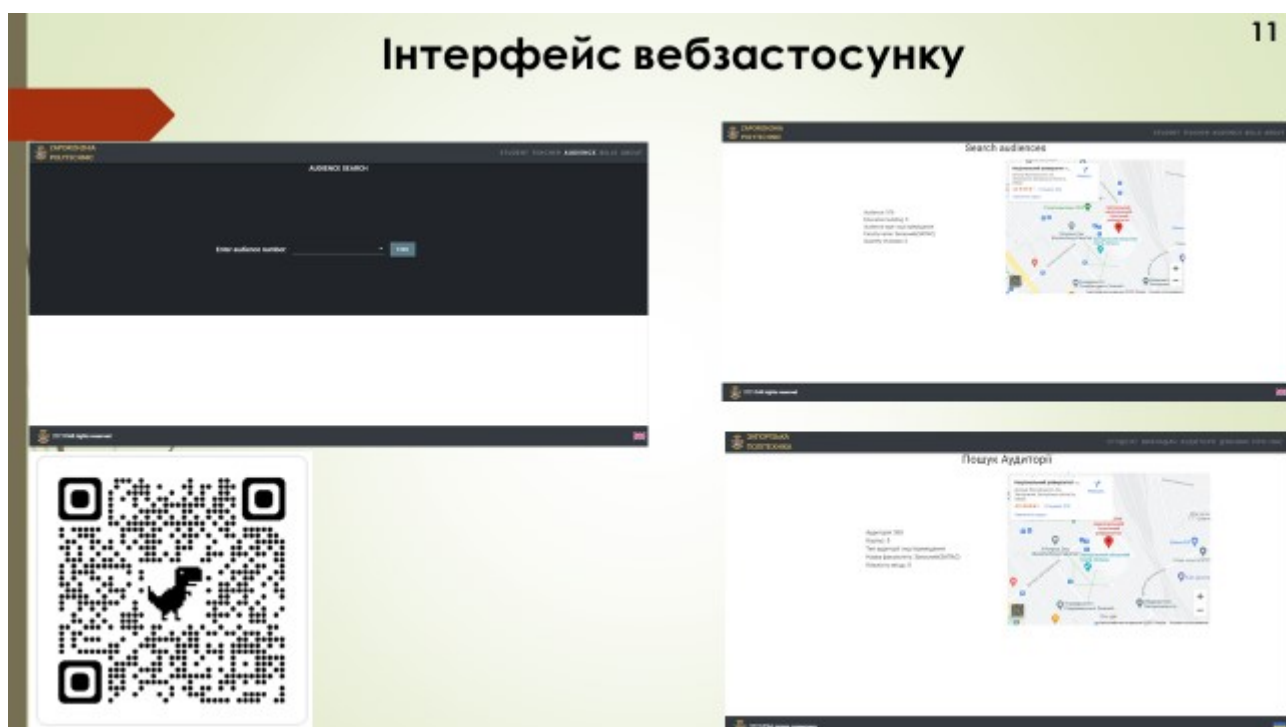


Рисунок В.11 – Слайд № 11

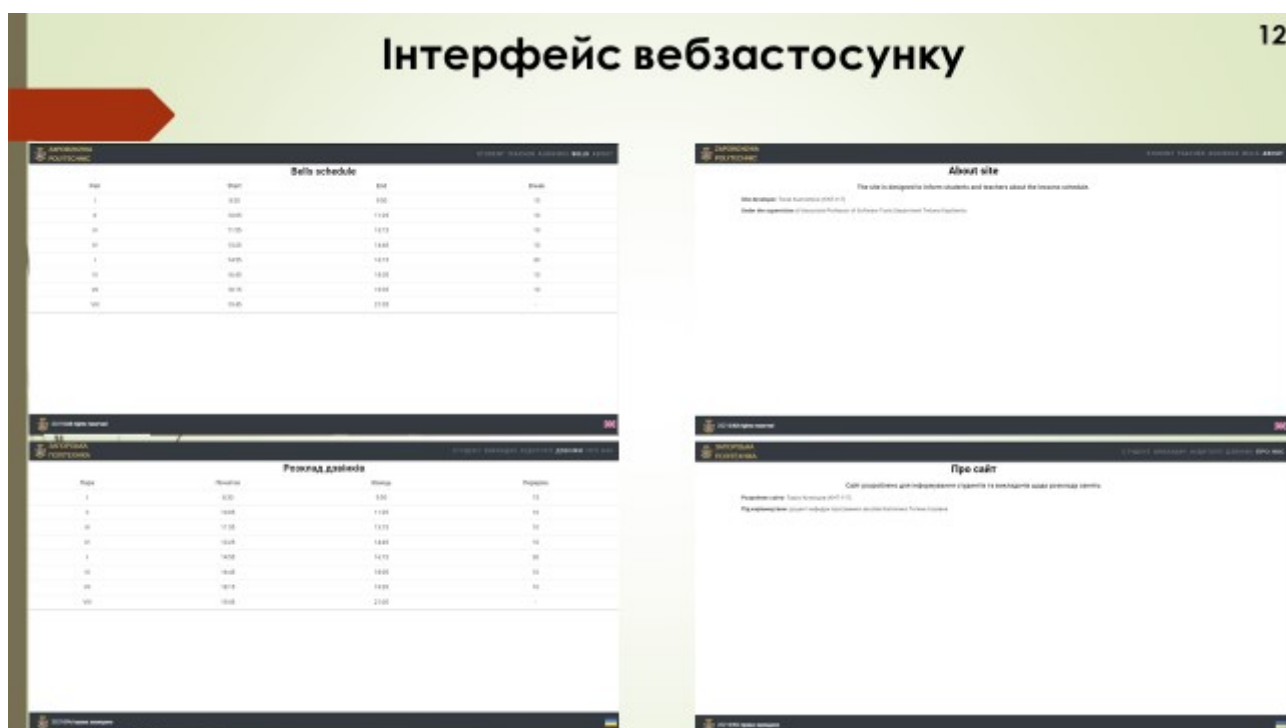


Рисунок В.12 – Слайд № 12

Інтерфейс вебзастосунку

Рисунок В.13 – Слайд № 13

Інтерфейс вебзастосунку

Рисунок В.14 – Слайд № 14

Висновок

15



- Під час виконання роботи було досліджено предметну область, розглянуто проблематику поставленого завдання, наведено його актуальність, доцільність. В даній роботі вивчалася проблема моніторингу розкладу закладів вищої освіти.
- На першому етапі роботи було ретельно проаналізовано предметну область: сформуовано проблему та актуальність роботи, розглянуто існуючі методи вирішення розв'язуваної проблеми та готові рішення, виявлено їх переваги та недоліки. Наступним кроком стала розробка архітектури системи, в ході якої було описано структуру розроблюваної програми, обрано інструменти для розробки ПЗ, а також спроектовано майбутню БД. На наступному етапі було реалізовано компоненти програми на основі «Технічного завдання» та надано їхнє обґрунтування.
- Було проведено порівняльний аналіз основних методів для вирішення цього завдання та встановлено, що існуючі засоби не відповідають повному функціоналу, який потрібен для вирішення завдання дипломної роботи. Тому виникає необхідність в реалізації нового вебзастосунку з розширеним функціоналом.
- Розроблено програмний продукт для моніторингу розкладу закладів вищої освіти. Серверна частина створеного ПП було написаний на мові програмування JS. Клієнтська частина створеного ПП була написана на мові програмування TS. У якості середовища розробки було використано Visual studio code. Для роботи з базою даних була використана бібліотека knex.js.
- В результаті роботи було отримано програмний продукт, що дозволяє виконати поставлене завдання, а саме – моніторингу розкладу для ЗВО. Роботу програми було протестоване на тестових даних, отриманих вже з готової бази даних ЗВО НУ «Запорізька політехніка». За необхідності програма може бути модифікована та використана під час розробки більш складної, комплексної системи, що вирішуватиме завдання моніторингу розкладу ЗВО «Запорізька політехніка».

Рисунок В.15 – Слайд № 15