Київський національний торговельно-економічний університет

Кафедра комп'ютерних наук та інформаційних систем

~~**ВИПУСКНИЙ КВАЛІФІКАЦІЙНИЙ ПРОЕКТ**~~

на тему:

«Розробка інтерактивного інтерфейсу для роботи з розкладом викладача»

Студента 4 курсу, 10 групи, спеціальності 122 «Комп'ютерні науки» Мухи Кирила Михайловича

_____

підпис студента

Науковий керівник кандидат економічних наук, доцент Шклярський Сергій Михайлович

_____

підпис керівника

Гарант освітньої програми кандидат технічних наук, доцент Демідов Павло Георгійович

_____

підпис керівника

Київ 2021

Київський національний торговельно-економічний університет

Факультет інформаційних технологій

Кафедра комп'ютерних наук та систем

Спеціальність 122 «Комп'ютерні науки»

Затверджую

Зав. кафедри _____Пурський О.І.

« » грудня 2020р.

Завдання

на випускний кваліфікаційний проект студенту

Мусі Кирилу Михайловичу

(прізвище, ім'я, по батькові)

1. Тема випускної кваліфікаційної роботи (проекту)

«Розробка інтерактивного інтерфейсу для роботи с розкладом викладача»

Затверджена наказом ректора від *«04» грудня 2020 р. № 4111*

2. Строк здачі студентом закінченої роботи *29 травня 2021 року*

3. Цільова установка та вихідні дані до роботи

Мета роботи: створення сайту для роботи з розкладом викладача з використанням сучасних методів веб-розробки

Об'єкт дослідження: процес розробки веб-сайту для роботи з розкладом викладача

Предмет дослідження: сучасні засоби створення веб-сайтів

4. Перелік графічного матеріалу_____

_____

5. Консультанти по роботі із зазначенням розділів, за якими здійснюється консультування:

Розділ	Консультант (прізвище, ініціали)	Підпис, дата	
		Завдання видав	Завдання прийняв
1	Шклярський С.М.	15.12.2020 р.	15.12.2020 р.
2	Шклярський С.М.	15.12.2020 р.	15.12.2020 р.
3	Шклярський С.М.	15.12.2020 р.	15.12.2020 р.

6. Зміст випускного кваліфікаційного проекту (перелік питань за кожним розділом)

ВСТУП

РОЗДІЛ 1. Аспекти моделювання електронного розкладу.

1.1 Аналіз поняття електронний розклад та його використання в навчальному процесі

1.2 Інформаційні системи та їх структура

РОЗДІЛ 2. Аналіз існуючих методів вирішення проблеми розробки та складових створення сучасних веб-сайтів

2.1 Теоретичні основи веб-програмування

2.2 Огляд засобів структурування даних для використання у системі електронного розкладу. MySQL та Postgresql

2.3 Особливості мови програмування Python та фреймворку Django. Її порівняння с PHP

2.4 Засоби та методи створення кліентської части веб додатку

РОЗДІЛ 3. Програмна реалізація web-додатку для роботи з розкладом викладача

3.1 Встановлення та налаштування WebPack'а та БД.

3.2 Стилізація веб-сайту з використанням Materialize.

3.3 Реалізація авторизації користувачів.

3.4 Розробка інтерактивного календаря.

ВИСНОВКИ

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

7. Календарний план виконання роботи

№ Пор.	Назва етапів випускної кваліфікаційної роботи	Строк виконання етапів роботи	
		За планом	фактично
1	2	3	4
1	*Вибір теми випускної кваліфікаційної роботи*	*01.10.2020*	*01.10.2020*
2	*Розробка та затвердження завдання на випускну кваліфікаційну роботу*	*15.12.2020*	*15.12.2020*
3	*Вступ*	*03.02.2021*	
4	*РОЗДІЛ 1. Теоретичні аспекти моделювання електронного розкладу.*	*28.02.2021*	
5	*РОЗДІЛ 2. Аналіз існуючих методів вирішення проблеми розробки та складових створення сучасних веб-сайтів*	*06.04.2021*	
6	*РОЗДІЛ 3. Програмна реалізація web-додатку для роботи з розкладом викладача*	*12.05.2021*	
7	*Висновки*	*15.05.2021*	

8	*Здача випускної кваліфікаційної роботи на кафедру науковому керівнику*	*20.05.2021*	
9	*Попередній захист випускної кваліфікаційної роботи*	*26.05.2021*	
11	*Виправлення зауважень, зовнішнє рецензування випускної кваліфікаційної роботи*	*27.05.2021*	
12	*Представлення готової зшитої випускної кваліфікаційної роботи на кафедру*	*29.05.2021*	
13	*Публічний захист випускної кваліфікаційної роботи*	*За розкладом роботи ЕК*	

8. Дата видачі завдання «15» грудня 2020 р.

9. Керівник випускної кваліфікаційної роботи (проекту)

Шклярський С.М.

(прізвище, ініціали, підпис)

10. Гарант освітньої програми Демідов П.Г.

(прізвище, ініціали, підпис)

11. Завдання прийняв до виконання студент-дипломник

Муха К.М.

(прізвище, ініціали, підпис)

12. Відгук керівника випускної кваліфікаційної роботи (проекту)

_____

_____

_____

Керівник випускної кваліфікаційної роботи (проекту)

_____ 30.05.2021 р.

(підпис, дата)

13. Висновок про випускну кваліфікаційну роботу (проект)

Випускна кваліфікаційна робота (проект) студента _____
(прізвище, ініціали)
може бути допущена до захисту в екзаменаційній комісії.

Гарант освітньої програми_____Демідов П.Г.____
(підпис, прізвище, ініціали)

Завідувач кафедри _____Пурський О.І.____
(підпис, прізвище, ініціали)
«_____»_____2021 р.

Анотація

У випускному кваліфікаційному проекті досліджено поняття «електронного розкладу викладача», що є метою даної роботи, його можливості та зумовлені ними переваги, зокрема, в умовах діджиталізації навчального процесу та повного його переносу в дистанційний режим; структура та види інформаційних систем. Розглянуто види структурування та представлення даних з практичним використанням висновків в контексті інформаційної системи. ~~В решті-решт~~ розроблено веб-додаток інформування викладача щодо розкладу на поточний чи наступний тиждень за рахунок окремого методу відструктурованих в інформаційній системі даних на сторінці веб-додатку.

Ключові слова: веб додаток, інтерфейс, інтерактивний розклад, структуризація даних, інформаційна система, дистанційний навчальний процес.

Annotation

The final qualification project explores the concept of "electronic teacher timetable", which is the purpose of this paper, its capabilities and resulting benefits, particularly in the context of the digitalisation of the learning process and its complete transfer to distance mode; the structure and types of information systems. Types of data structuring and presentation with practical use of the findings in the context of an information system were considered. Finally a web application has been developed to inform the teacher about the schedule for the current week or the next week by a separate method structured in the information system data on the web application page.

Keywords: web application, interface, interactive timetable, data structuring, information system, distance learning process.

ЗМІСТ

Вступ.....10

Розділ 1: Аспекти моделювання електронного розкладу.....12

1.1. Аналіз поняття електронний розклад та його використання в навчальному процесі.....12

1.2. Інформаційні системи та їх структура.....13

Розділ 2: ~~Аналіз існуючих методів вирішення проблеми розробки та складових створення сучасних веб-сайтів~~.....18

2.1. Теоретичні основи веб-програмування.....18

2.2. Огляд засобів структурування даних для використання у системі електронного розкладу. MySQL та Postgresql.....22

2.3. Особливості мови програмування Python та фреймворку Django. Її порівняння с PHP…….....28

2.4. Засоби та методи створення кліентської части веб додатку.....33

Розділ 3: Програмна реалізація web-додатку для роботи з розкладом викладача.....37

3.1. Встановлення та налаштування WebPack'а та БД.....37

3.2. Стилізація веб-сайту з використанням Materialize.....41

3.3. Реалізація авторизації користувачів.....43

3.4. Розробка інтерактивного

календаря.....46

Висновки.....52

Список використаних джерел.....54

ВСТУП

Сучасні технології проникли в усі сфери діяльності, і вища освіта не є винятком. Кожна сучасна освітня організація має свій сайт, електронний освітній портал, електронну бібліотечну систему та корпоративний портал. У зв'язку з функціонуванням єдиного інформаційного простору в університеті є доречним використання існуючої інформаційної системи для створення на її базі електронного розкладу.

Розклад занять – основний документ, що регламентує академічну роботу студентів і викладачів, передбачає проведення лекційних, лабораторних, семінарських занять відповідно до робочих навчальних планів, ухвалених вченою радою та затверджених ректором. Навчальна робота здійснюється за такими видами розкладів: начитки лекцій, семестрових занять студентів денної форми навчання, лабораторно-екзаменаційної сесій студентів заочної форми навчання, індивідуально-консультативної роботи, екзаменаційних сесій, державної атестації. Семестровий розклад занять та розклад лабораторно-екзаменаційних сесій складається в розрізі факультетів, форм навчання, курсів, академічних груп. У ньому зазначається прізвище викладача, час та місце проведення занять [1].

Електронний розклад – це електронна інтерпретація основного документу організації навчального процесу, що передбачає використання електронних інформаційних ресурсів, що розміщенні в мережі інтернет.

Створення електронного розкладу, розміщеного в мережі інтернет, дозволить кожному викладачу дізнатися які пари у нього будуть, час їх проведення і аудиторію. Додаток надасть користувачу інформацію про місце і час проведення заняття, викладача, що його проводить.

Мета і завдання дослідження. Метою випускного кваліфікаційного проекту є проектування і розробка веб-додатку електронного розкладу викладача. Для досягнення поставленної мети треба виконати такі завдання:

- Дослідити варіанти створення інформаційної бази;

- Дослідити можливі варіанти верстки веб-сторінки;
- Дослідити можливості бібліотек Django мови Python;
- Розробити клієнтську і серверну частини веб-додатку.

Об’єкт дослідження: електронний розклад та його складові.

Предмет дослідження: методи та інформаційні технології верстки у Web-розробці.

Методи дослідження: теоретичний аналіз науково-педагогічної літератури, порівняння, систематизація, класифікація дали змогу дослідити і узагальнити матеріали з питань організації веб-розробки та інформаційної бази.

Для практичного вирішення поставлених задач використовувалися такі методи:

- загальнонауковий аналітичний метод;
- методи програмного моделювання для побудови оптимальної архітектури додатку;
- методи побудови інформаційних структур та баз даних;
- метод порівняльного аналізу для порівняння різних варіантів верстання веб-сайтів;
- методи об’єктно-орієнтованого програмування для створення веб-додатку електронного розкладу викладача.

У відповідності до поставленної мети та конкретних завдань дослідження, визначено структуру роботи. Вона складається зі вступу, трьох розділів, висновків та списку використанної літератури.

Практичне значення. Матеріали дипломного проекту можуть бути використані для розгортання на базі існуючої інформаційної інфраструктури вищого навчального закладу сучасної системи інформування викладача про його розклад занять.

РОЗДІЛ 1.
АСПЕКТИ МОДЕЛЮВАННЯ ЕЛЕКТРОННОГО РОЗКЛАДУ.

1.1. Аналіз поняття «електронний розклад» та його використання в навчальному процесі

Електронний розклад занять – це документ, який регламентує хід навчального процесу, учасниками якого є здобувачі вищої освіти, деканати, кафедри та інші структурні підрозділи, тобто розробляється відповідно до навчального плану з урахуванням педагогічних та санітарно-гігієнічних вимог. Визначає навчальну діяльність професорського-викладацького складу, здобувачів вищої освіти та навчально-допоміжного персоналу.[2]

Розклад занять встановлює загальний режим навчання, початок і кінець кожного заняття та тривалість перерв між ними, передбачає теоретичну та практичну підготовку здобувачів вищої освіти в академічних групах на кожен робочий день тижня, забезпечує рівномірний розподіл їх навчального навантаження, сприяє збереженню працездатності упродовж робочого дня, тижня, семестру та навчального року в цілому і формуванню стійких знань, умінь та навичок.

Види розкладів занять:

- розклад навчальних занять денної форми навчання;
- розклад екзаменаційної сесії денної форми навчання;
- розклад навчально-екзаменаційних сесій заочної (дистанційної) форми навчання;
- графік захисту/складання випускних кваліфікаційних робіт (проектів), комплексних кваліфікаційних іспитів;
- графік ліквідації академічної заборгованості.[3]

Розклад містить наступні відомості: факультет, курс, група, тиждень (парний/непарний), назва дисципліни, форма проведення заняття (лекція, лабораторні, практичні заняття, консультації), прізвище та ініціали викладача, місце та час проведення заняття.

Сучасна система комунікації в університеті пройшла інноваційний шлях, утворюються університетські комп'ютерні системи, що дозволяють усі дані зберігати в електронному вигляді, а саме сервіси електронних розкладів.

Використання електронного розкладу також дозволяє будь-якому викладачу, дізнатися які заняття він має провести, в який час та в якій аудиторії, використовуючи будь-який пристрій з доступом до мережі Інтернет. Також є можливість миттєво відстежувати зміни в графіку навчального процесу. Розроблення та використання автоматизованих ресурсів може суттєво покращити якість освітніх послуг, завдяки підтриманню їх постійної актуальності для цільової аудиторії.

1.2. Інформаційні системи та їх структура

Велика кількість сучасної роботи пов'язана з роботою з великим обсягом даних. Дані є основним значенням або фактами і організовані в базі даних. Багато людей вважають дані синонімами інформації, однак інформація насправді складається з даних, які були організовані з метою відповісти на запитання та вирішення проблем.

Інформаційна система – організаційно впорядкована сукупність документів (масивів документів) та інформаційних технологій, в тому числі з використанням технічних засобів, що реалізують інформаційні процеси та призначені для зберігання, обробки, пошуку, розповсюдження, передачі та надання інформації. [4]

Мета інформаційної системи – перетворити необроблені дані в корисну інформацію, яка може бути використана для прийняття рішень.

Усі інформаційні системи виконують ряд функцій, які можна класифікувати наступним чином:

- введення інформації, отриманої з джерел інформації;
- опрацювання (перетворення) інформації;
- зберігання вхідної опрацьованної інформації;
- виведення інформації, призначенної для користувача;

- поширення інформаційною мережею.[5]

Інформаційна система складається з п’яти компонентів; апаратного забезпечення, програмного забезпечення, бази даних, мережі та персоналу. Ці п’ять компонентів об’єднуються для введення, обробки, виведення, зворотного зв’язку та контролю.

- Апаратне забезпечення складається з пристрою вводу/виводу, процесора, операційної системи, та мультимедійних пристроїв;
- Програмне забезпечення складається з різних програм та процедур;
- База даних складається з даних, організованих у потрібну структуру;
- Мережа складається з мережевих пристроїв та комунікаційних мереж;
- Персонал включає в себе операторів пристроїв, мережевих адміністраторів та системних спеціалістів.

Під час вхідної стадії вказівки даних надходять до систем, над якими на етапі процесу працюють програми та інші запити. Обробка інформації включає в себе введення даних, обробку даних, зберігання даних, виведення та контроль. На етапі вводу дані подаються у вигляді звітів.[6]

У більшості випадків для створення власної інформаційної системи неможливо обійтися без використання баз даних.

База даних – це сукупність взаємопов’язаних даних, організована відповідно до певних правил опису, зберігання та маніпулювання, подана у формі, придатній для автоматичного опрацювання й призначена задовольнити інформаційні потреби користувача.[7]

Система управління базами даних – це комп’ютерна програма чи комплекс програм, що забезпечує користувачам можливість створення, збереження, оновлення, пошуку інформації та контролю доступу в базах даних.[8]

СУБД має забезпечувати наступні функції:

- Паралельність: паралельний доступ (що означає «одночасно») до однієї бази даних декількома користувачами;
- Безпека: правила безпеки для визначення прав доступу користувачів;
- Резервне копіювання та відновлення: регулярно обробляє та відновляє резервні копії даних та відновлює дані, якщо виникає проблема;
- Цілісність: структура бази даних та правила покращують цілісність даних;
- Описи даних: словник даних забезпечує опис даних.[9]

Структуру інформаційної системи складає сукупність окремих її частин – підсистем. Підсистема – це частина системи, яка виділена за певною ознакою.

Існує два варіанти представлення структури інформаційних систем. Перший варіант представляє структуру будь-якої інформаційної системи як сукупність підсистем, що забезпечують інформаційне, технічне, математичне, програмне, організаційне і правове забезпечення.

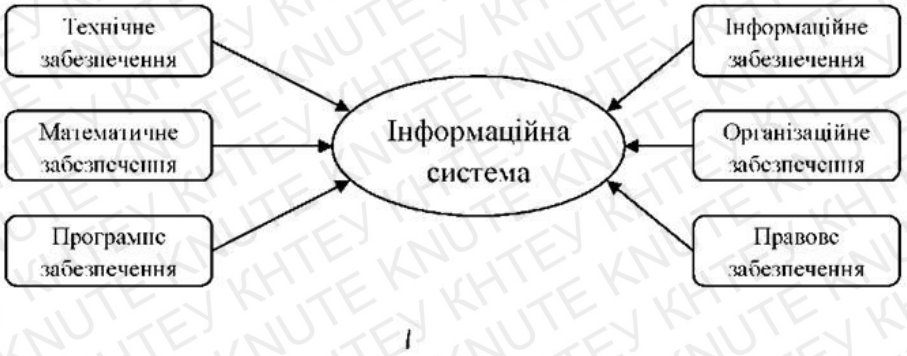


Рис. 1.1. Структура інформаційної системи

Інформаційне забезпечення – сукупність єдиної системи класифікації й кодування повідомлень, уніфікованих систем документації, схем інформаційних потоків, що циркулюють в організації, а також побудови баз даних.

Технічне забезпечення - комплекс технічних засобів, призначених для роботи інформаційної системи, а також відповідна документація на ці засоби й технологічні процеси.

Математичне й програмне забезпечення - сукупність математичних методів, моделей, алгоритмів і програм для реалізації цілей і завдань інформаційної системи, а також нормального функціонування комплексу технічних засобів.

Організаційне забезпечення - сукупність методів і засобів, що регламентують взаємодію працівників з технічними засобами й між собою в процесі розробки й експлуатації інформаційної системи.

Правове забезпечення - сукупність правових норм, що визначають створення, юридичний статус і функціонування інформаційних систем, що регламентують порядок одержання, перетворення й використання відомостей. [10]

Другий спосіб представляє структуру інформаційної системи як набір взаємопов'язаних між собою компонентів, що залежать один від одного.

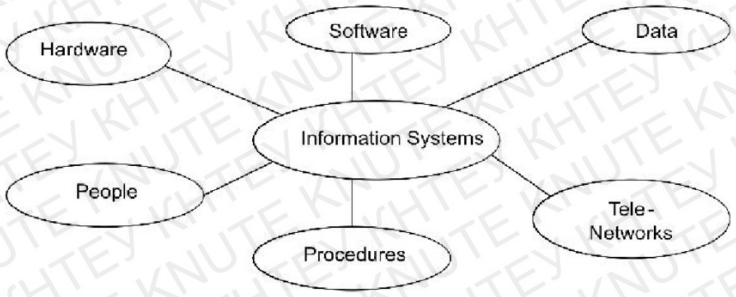


Рис. 1.2. Компоненти інформаційної системи

Компоненти інформаційної системи наступні:

- Апаратне забезпечення, що використовується для введення, виведення та обробки. Апаратне забезпечення включає пристрої вводу, виводу,

процесор та медіа-пристрої. Сюди також входять комп'ютерні периферійні пристрої.

- Програмне забезпечення включає операційні системи та програми, які містять набір інструкцій, що використовуються для обробки інформації.
- Дані – це неорганізовані факти і цифри, які згодом оброблюються для отримання інформації. Дані управляються за допомогою системи управління базами даних. Програмне забезпечення баз даних використовується для ефективного доступу до потрібних даних для управління базами знань.
- Ресурси мереж відносяться до телекомунікаційних мереж. Ці ресурси полегшують потік інформації в організації. Мережі складаються як з пристроїв для фізичних засобів, таких як мережеві карти, маршрутизатори, концентратори та кабелі, так і програмне забезпечення, таке як операційні системи, веб-сервери, сервери даних та сервери додатків. Телекомунікаційні мережі складаються з комп'ютерів, процесорів зв'язку та інших пристроїв, з'єднаних між собою засобами зв'язку та керованим програмним забезпеченням.
- Люди є кінцевим користувачем інформаційної системи, який використовує інформацію за власним призначенням. Люди також відповідають за розробку та експлуатацію інформаційних систем. Вони включають системних аналітиків, комп'ютерних операторів, програмістів та інших службовців ІС та методи управління. [11]
- Процедури: дозволи, що регулюють роботу інформаційної системи.

РОЗДІЛ 2.

~~АНАЛІЗ ІСНУЮЧИХ МЕТОДІВ ВИРІШЕННЯ ПРОБЛЕМИ РОЗРОБКИ ТА СКЛАДОВИХ СТВОРЕННЯ СУЧАСНИХ ВЕБ-САЙТІВ.~~

2.1. Теоретичні основи веб-програмування.

Веб-середовище – це величезний архів корисної та оригінальної інформації, інакше кажучи: бібліотека людських надбань. Ця інформація з кожним роком поповнюється новими знаннями.

Що ж собою являє веб-середовище? Це набір сторінок, наповнених інформацією. Для того, щоб користувачу отримати необхідну інформацію, потрібно набрати у браузері адрес сторінки, яка її цікавить. Існують випадки, коли потрібно знайти інформацію, а точний адрес необхідної сторінки не відомо. Тоді за допомогою звертаються до пошукових машин, створюють запит і у відповідь отримують набір web-сторінок, які можуть містити дані, що цікавлять людину. Web-сторінки також називають html-документами. Для того щоб створити html-документ, потрібно у текстовому редакторі за допомогою html-тегів розмітити документ. Для цього використовують HTML.

HTML - скорочення від "*HyperText Mark-up Language*" - перекладається як "Мова розмітка гіпертексту" (Гіпертекст - це текст, що не послідовно зв'язаний з іншими документами, тобто у вас є змога з першої сторінки документу перейти на останню). Іншими словами HTML - це мова розмітки, або ще один спосіб зберігання інформації. [12]

Існує консорціум Всесвітньої павутини (W3C – *World Wide Web Consortium*), створений у 1994 році Тімом Бернерс-Лі, який займається розробкою вільно розповсюджуваних технологій для Всесвітньої павутини.

Також W3C встановлює стандарти на web-технології. W3C вдосконалює і стандартизує HTML

Всесвітню мережу утворюють мільйони веб-серверів мережі Інтернет, розташованих по всьому світу. Веб-сервер є програмою, що запускається на підключеному до мережі комп’ютері і використовую протокол HTTP для передачі даних. У простому вигляді така програма отримує по мережі HTTP-запит на певний ресурс, знаходить відповідний файл на локальному жорсткому диску і відправляє його по мережі до того комп'ютера, який робив запит. Складніші веб-сервери здатні динамічно формувати ресурси у відповідь на HTTP-запит. Приклад такого веб-сервера можна назвати будь-яку пошукову систему, яка формує список веб-ресурсів на запит користувача.

Додатковими функціями багатьох веб-серверів є:

- ведення журналу серверу про звернення користувачів до ресурсів;
- автентифікація користувачів – процедура встановлення належності користувачеві інформації в системі пред'явленого ним ідентифікатора;
- підтримка сторінок, що динамічно генеруються;

HTTP - широко поширений протокол передачі даних, спочатку призначений для передачі гіпертекстових документів (тобто документів, які можуть містити посилання, що дозволяють організувати перехід до інших документів). Абревіатура HTTP розшифровується як *HyperText Transfer Protocol*, «протокол передачі гіпертексту». Відповідно до специфікації OSI, HTTP є протоколом прикладного (верхнього, 7-го) рівня. Актуальна на даний момент версія протоколу, HTTP 1.1, описана в специфікації RFC 2616.

Протокол HTTP припускає використання клієнт-серверної структури передачі даних. Клієнтську програму формує запит і відправляє його на сервер, після чого серверне програмне забезпечення обробляє цей запит,

формує відповідь і передає його назад клієнтові. Після цього клієнтську програму може продовжити відправляти інші запити, які будуть оброблені аналогічним чином.

Завдання, яке традиційно вирішується за допомогою протоколу HTTP - обмін даними між призначеним для користувача додатком, що здійснює доступ до веб-ресурсів (зазвичай це веб-браузер) і веб-сервером. На даний момент саме завдяки протоколу HTTP забезпечується робота Всесвітньої павутини.

Для того, щоб сформувати HTTP-запит, необхідно скласти стартову рядок, а також задати принаймні один заголовок - це заголовок Host, який є обов'язковим, і повинен бути присутнім в кожному запиті.

Стартовий (початковий) рядок запиту для HTTP 1.1 складається за такою схемою:

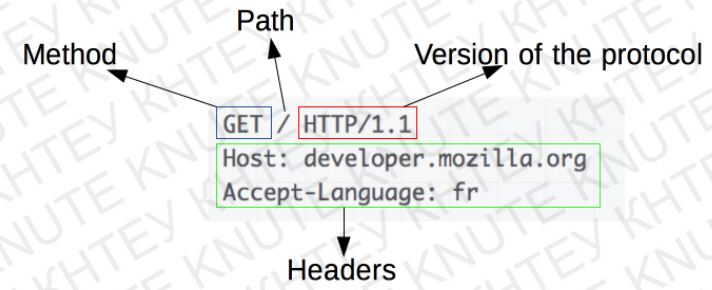


Рис. 2.1. Приклад HTTP запиту

Метод (в англомовній тематичної літератури використовується слово *method*, а також іноді слово *verb* - «дієслово») являє собою послідовність з будь-яких символів, крім керуючих і роздільників, і визначає операцію, яку потрібно здійснити з зазначеним ресурсом. Специфікація HTTP 1.1 не обмежує кількість різних методів, які можуть бути використані, проте в цілях відповідності загальним стандартам і збереження сумісності з максимально широким спектром програмного забезпечення як правило використовуються

лише деякі, найбільш стандартні методи, зміст яких однозначно розкритий в специфікації протоколу.

URI (*Uniform Resource Identifier*, уніфікований ідентифікатор ресурсу) - шлях до конкретного ресурсу (наприклад, документа), над яким необхідно здійснити операцію (наприклад, в разі використання методу GET мається на увазі отримання ресурсу). Деякі запити можуть не ставитися до будь-якого ресурсу, в цьому випадку замість URI в стартову рядок може бути додана зірочка (Астеріск, символ «*»). Наприклад, це може бути запит, який відноситься до самого веб-сервера, а не якого-небудь конкретного ресурсу.

Версія визначає, відповідно до якої версією стандарту HTTP складений запит (вказується як два числа, розділених крапкою).[13]

Сьогодні на зміну технологіям Web приходять технології Web2 та Web3. їх основою є соціальні мережі, спільна робота, спрямована на розробку інформаційних ресурсів. На основі цих нових технологій функціонують корпоративні блоги, енциклопедії Wiki тощо.[14]

Перевагою візуалізації на основі веб-технологій:

- незалежність від устаткування та операційної системи;
- проста системна інтеграція та обслуговування;
- дуже незначні витрати на навчання та розробку;
- придатність до конфігурування польових компонентів або візуалізації процесів, аж до централізованих або мобільних концепцій керування;
- відсутність витрат на ліцензії на використання;
- невисока вартість придбання та низьке енергоспоживання;
- надзвичайна гнучкість керування та моніторингу завдяки відсутності прив’язки до конкретного місця й часу.

На сьогодні web-сторінку можна повноцінно вважати представником компанії, організації в інтернеті. І вже сьогодні кожна web-сторінка, що створюється, повинна містити такий невід'ємний атрибут, як система адміністрування. Це в свою чергу робить сторінку не просто

рекламноінформаційним ресурсом, а й достатньо динамічним джерелом інформації. По своїй суті вона стає ще одним інструментом для відображення певної динамічної інформації про компанію. В такому випадку web-сторінку можна розглядати як набір певної бізнес-логіки, що забезпечує обробку і представлення певної інформації. Тобто web-сторінка стає достатньо вагомим інструментом комерції компанії. Тому перед створенням web-сторінки варто чітко визначитись щодо її мети, функціонального наповнення, інформаційного наповнення, зовнішнього вигляду, планом розкрутки та підтримки.

Отже, найвідомішою та найпопулярнішою службою Інтернету є Всесвітня павутина. Саме після її розповсюдження став можливий масовий доступ користувачів до Всесвітньої мережі. Служба Веб підтримується сукупністю серверів, які здатні обмінюватися даними за протоколом HTTP.

2.2. Огляд засобів структурування даних для використання у системі електронного розкладу. MySQL та PostgreSQL

Традиційно використовуються для WEB-розробок мови програмування (Perl, PHP, ASP та інші), що дозволяють реалізовувати практично будь-які завдання. Але обробляти з їх допомогою великі обсяги даних, що мають до того ж складну структуру, досить важко. Розробка подібних програм вимагає зростаючих затрат праці програмістів, у геометричній прогресії зростає обсяг програмного коду та кількість помилок, знижується надійність програмного забезпечення.

У такій ситуації на допомогу програмісту приходять бази даних.

База даних — це певний набір даних, які пов'язані між собою спільною ознакою або властивістю, та впорядковані, наприклад, за алфавітом.

Головною перевагою БД є швидкість внесення та використання потрібної інформації. Завдяки спеціальним алгоритмам, які використовуються для баз даних, можна легко знаходити необхідні дані всього за декілька секунд. Також в базі даних існує певний взаємозв'язок

інформації: зміна в одному рядку може спричинити зміни в інших рядках — це допомагає працювати з інформацією простіше і швидше.

Бази даних для сайтів дають змогу зберігати інформацію, що виглядає як зв'язані між собою таблиці. Саме в БД зберігаються вся необхідна та корисна інформація для функціонування сайту.

Щоб створити запит до бази даних часто використовують *Structured Query Language*. SQL дає змогу додавати, редагувати та видаляти інформацію, що міститься у таблицях. Під час програмування сайтів використовують різні системи управління БД.

До основних СУБД, відносять:

- об'єктно-реляційна система управління базами даних Oracle Database;
- вільна система управління базами даних PostgreSQL;
- система керування базами даних Microsoft SQL Сервер;
- вільна система управління базами даних MySQL;

Такі системи управління відрізняються централізованою обробкою запитів, забезпечують надійність, доступність та безпеку БД. [15]

У класичній теорії виділяють три типи, три структури баз даних: ієрархічну, мережеву та реляційну. В даний час домінуюче становище займають реляційні бази даних.

Реляційна модель бази даних являє собою централізоване сховище таблиць, що забезпечує безпечний одночасний доступ до інформації з боку багатьох користувачів. У рядках таблиць частина полів містить дані, що відносяться безпосередньо до запису, а частина - посилання на записи інших таблиць. Таким чином, зв'язки між записами є невід'ємною властивістю реляційної моделі. [16]

У реляційній моделі досягається інформаційна та структурна незалежність. Записи не пов'язані між собою настільки, щоб зміна одного з

них торкнулося інші, а зміна структури бази даних не обов'язково призводить до перекомпіляції працюючих з нею програм.

У реляційних СУБД застосовується мова SQL, що дозволяє формулювати довільні, нерегламентовані запити. Це мова четвертого покоління, тому будь-який користувач може швидко навчитися складати запити. До того ж, існує безліч програм, які дозволяють будувати логічні схеми запитів у графічному вигляді. Все це відбувається за рахунок посилення вимог до продуктивності комп'ютерів. На щастя, сучасні обчислювальні потужності більш ніж адекватні.

Реляційні бази даних страждають від відмінностей у реалізації мови SQL, хоча це і не проблема реляційної моделі. Кожна реляційна СУБД реалізує якусь підмножину стандарту SQL плюс набір унікальних команд, що ускладнює завдання програмістам, які намагаються перейти від однієї СУБД до іншої. Доводиться робити нелегкий вибір між максимальною переносимістю і максимальною продуктивністю.

MySQL виникла як спроба застосувати mSQL до власних розробок компанії: таблиць, для яких використовувалися ISAM - підпрограми низького рівня. В результаті був вироблений новий SQL-інтерфейс, але API-інтерфейс залишився у спадок від mSQL. Звідки походить назва «MySQL» - достеменно не відомо. Розробники дають два варіанти: або тому, що практично всі напрацювання компанії починалися з префікса My, або на честь дівчинки на ім'я My, дочки Майкла Монті Віденіуса, одного з розробників системи.[17]

Клієнтська програма MySQL являє собою утиліту командного рядка. Ця програма підключається до сервера по мережі. Команди, що виконуються сервером, зазвичай пов'язані з читанням і записом даних на жорсткому диску.

MySQL взаємодіє з базою даних на мові, що зветься SQL (*Structured Query Language* - мова структурованих запитів).

Програмне забезпечення MySQL - це ПЗ з відкритим кодом. ПЗ з відкритим кодом означає, що застосовувати і модифікувати його може будь-який бажаючий. Таке ПЗ можна отримувати за допомогою Internet і

використовувати безкоштовно. При цьому кожен користувач може вивчити вихідний код і змінити його відповідно до своїх потреб.

Технічні можливості СУБД MySQL ПО MySQL є системою клієнт-сервер, що містить багатопоточний SQLсервер, який забезпечує підтримку різних обчислювальних машин баз даних, а також кілька різних клієнтських програм і бібліотек, засоби адміністрування та широкий спектр програмних інтерфейсів (API).

Система безпеки заснована на привілеї та паролі з можливістю верифікації з віддаленого комп'ютера, за рахунок чого забезпечується гнучкість і безпека. Паролі при передачі по мережі під час з'єднання з сервером шифруються. Клієнти можуть з'єднуватися з MySQL, використовуючи сокета TCP / IP, сокета Unix або іменовані канали (named pipes, під NT).

Починаючи з MySQL версії 3.23, де використовується новий тип таблиць, максимальний розмір таблиці доведений до 8 мільйонів терабайт (263 Bytes). Однак слід зауважити, що операційні системи мають свої власні обмеження за розміром файлів. Нижче наведено кілька прикладів:

- 32-розрядна Linux-Intel - розмір таблиці 4 Гб;
- Solaris 2.7 Intel - 4 Гб;
- Solaris 2.7 UltraSPARC - 512 Гб;
- WindowsXP - 4 Гб

Як можна побачити, розмір таблиці в базі даних MySQL звичайно лімітується операційною системою. За замовчуванням MySQL-таблиці мають максимальний розмір близько 4 Гб. Для будь-якої таблиці можна перевірити / визначити її максимальний розмір за допомогою команд SHOW TABLE STATUS або myisamchk-dv table_name. Якщо велика таблиця призначена тільки для читання, можна скористатися myisampack, щоб об'єднати декілька таблиць в одну і стиснути її. Зазвичай myisampack стискує таблицю принаймі на 50%, тому в результаті можна отримати дуже великі таблиці. [18]

MySQL функціонує на більшості платформ:

- AIX;
- BSDi;
- FreeBSD;
- HP-UX;
- GNU/Linux;
- Mac OS X;
- NetBSD;
- OpenBSD;
- OS/2 Warp;
- SGIIRIX;
- Solaris;
- SunOS;
- SCO OpenServer;
- SCO UnixWare;
- Tru64;
- Windows 95;
- Windows 98;
- Windows NT;
- Windows 2000;
- Windows XP;
- Windows Server 2003;
- Windows Vista.

MySQL має API для мов C, C++, Ейфель, Java, Лисп, Perl, PHP, Python, Ruby, Smalltalk та Tcl, бібліотеки для мов платформи .NET, а також забезпечує підтримку для ODBC за допомогою засобів ODBC-драйвера MyODBC.

PostgreSQL починає свій «родовід» від некомерційної СУБД Postgres, розробленою, як і багато Open Source-проектів, в Каліфорнійському

університеті в Берклі. До розробки Postgres, що почалася в 1986-му році, мав безпосереднє відношення Майкл Стоунбрейкер, керівник більш раннього проекту Ingres, на той момент уже придбаного компанією Computer Associates. Сама назва «Postgres» розшифровується як «Post Ingres», відповідно, при створенні Postgres були застосовані багато ким вже раніше зроблені напрацювання.

PostgreSQL базується на мові SQL і підтримує багато з можливостей стандарту SQL: 2003 (ISO / IEC 9075). На даний момент (версія 8.3.5), в PostgreSQL є наступні обмеження, що приведені в табл 2.1.

Максимальний розмір бази даних	Не має обмежень
Максимальний розмір таблиці	32 ТБайт
Максимальний розмір запису	1,6 ТБайт
Максимальний розмір поля	1 ГБайт
Максимум записів у таблиці	Не має обмежень
Максимум полів у таблиці	250—1600, залежно від типу полів
Максимум індексів у таблиці	Не має обмежень

Таблиця 2.1. Обмеження PostgreSQL

Сильними сторонами PostgreSQL вважаються:

- підтримка БД практично необмеженого розміру;
- потужні і надійні механізми транзакцій і реплікацій;
- наслідування;
- легке масштабування.

Згідно з результатами автоматизованого дослідження різного ПЗ на предмет помилок, у вихідному коді PostgreSQL було знайдено 20 проблемних місць на 775 000 рядків вихідного коду (в середньому, одна помилка на 39

000 рядків коду). Для порівняння: MySQL - 97 проблем, одна помилка на 4 000 рядків коду; FreeBSD (повністю) - 306 проблем, одна помилка на 4 000 рядків коду; Linux (тільки ядро) - 950 проблем, одна помилка на 10 000 рядків коду.

На базі PostgreSQL компанією EnterpriseDB створені більш потужні варіанти цієї СУБД, що є платними для комерційного використання - Postgres Plus (складається цілком тільки з продуктів з відкритими вихідними кодами; плата потрібна тільки при необхідності придбання комерційної підтримки продукту) і Postgres Plus Advanced Server (розширення PostgreSQL спеціальними можливостями для забезпечення сумісності з Oracle Database). У комплекті поставки даних продуктів міститься великий набір ПЗ для розробників і DBA:

- Postgres Studio - більш потужний аналог pgAdmin;
- Postgres Plus Debugger - відладчик для коду на PL / pgSQL, інтегрований з попереднім пакетом;
- Migration Studio - інструмент для автоматичного перетворення баз даних з MySQL / Oracle в PostgreSQL.

PostgreSQL використовують:

- Google, зокрема Google Maps використовує розширення postGis з великою кількістю вбудованих функцій для роботи з картами;
- Cropio - система контролю вегетації і управління полями;
- Amazon;
- Yandex;
- eBay;
- GitHub;
- та багато інших.

2.3. Особливості мови програмування Python та фреймворку Django. ~~Її порівняння з~~ PHP

Мови веб-програмування - це мови, які в основному призначені для роботи з інтернет-технологіями, а окремі з них створювалися лише задля роботи з будь-яким ресурсом, і тривалий час до них приходила популярність і загальне визнання (наприклад, PHP).

Мови веб-програмування діляться на дві групи: клієнтські та серверні.

Python є широко використовуваним мовою програмування загального призначення, високого рівня. Його філософія дизайну підкреслює читаність коду, а його синтаксис дозволяє програмістам, висловити поняття в меншій кількості рядків коду, ніж було б можливо в таких мовах, як C++ або Java. ~~Говорячи простою людською мовою,~~ на Python можна написати практично що завгодно (веб, настільні додатки, ігри, скрипти по автоматизації, комплексні системи розрахунку, системи управління життєзабезпеченням і багато-багато іншого) без відчутних проблем. Більш того, поріг входження низький, а код багато в чому лаконічний і зрозумілий навіть того, хто ніколи на ньому не писав. За рахунок простоти коду, подальший супровід програм, написаних на Python, стає легше і приємніше в порівнянні з Java або C ++. А з точки зору бізнесу це тягне за собою скорочення витрат і збільшення продуктивності праці співробітників. [19]

Python підтримує кілька парадигм програмування, в тому числі об'єктно-орієнтованого, імперативний і функціональному програмуванні або процедурних стилів. Він має динамічну систему типів і автоматичне керування пам'яттю і має велику і всеосяжну стандартну бібліотеку. [20]

Безсумнівною перевагою є те, що інтерпретатор Python реалізований практично на всіх платформах і операційних системах. Першою такою мовою був C, проте його типи даних на різних машинах могли займати різну кількість пам'яті і це служило деякою перешкодою при написанні програм які повинні використовуватися на різних операційних системах. Python, в свою чергу, таким недоліком не володіє.

Наступна перевага - наявність великого числа підключаємих до програми модулів та бібліотек, що забезпечують різні додаткові можливості. Такі бібліотеки пишуться на C і на самому Python. [21]

Python використовується в багатьох сферах. У Web-розробці Python, мабуть, використовується найбільше. Веб-фреймворк Django продовжує набирати обертів, поповнюючи армію своїх фанатів.

Django (Джанго) - вільний фреймворк для веб-додатків на мові Python, що використовує шаблон проектування MVC (*Model View Controller*). Проект підтримується організацією Django Software Foundation.[22]

Сайт на Django будується з одного або декількох додатків, які рекомендується робити відчужуваними і підключаються. Це одне з істотних архітектурних відмінностей цього фреймворка від деяких інших (наприклад, Ruby on Rails). Один з основних принципів фреймворка - DRY (англ. Do not repeat yourself) Також, на відміну від інших фреймворків, обробники URL в Django конфігуруються за допомогою регулярних виразів, а не виводяться автоматично зі структури моделей контролерів. Для роботи з базою даних Django використовує власний ORM, в якому модель даних описується класами Python, і по ній генерується схема бази даних.[23]

Веб-фреймворк Django використовується в таких великих і відомих сайтах, як Instagram, Disqus, Mozilla, The Washington Times, Pinterest, YouTube, Google та ін.

Також Django використовується в якості веб-компонента в різних проектах, таких як Graphite[24] – система побудови графіків і спостереження, FreeNAS - вільна реалізація системи зберігання та обміну файлами та ін.

PHP (Personal Home Page) або Hypertext Preprocessor - це мова сценаріїв, що використовується для автоматизації різних завдань на веб-розробці на стороні сервера. Це мова програмування загального призначення, яка легко вбудовується в коди HTML. PHP дозволяє створювати динамічні веб-сторінки, веб-програми для електронної комерції та навіть додатки на базі

даних. Це мова сценаріїв з відкритим кодом, сумісна з MySQL, Oracle та іншими подібними службами баз даних.

Варіанти використання PHP:

- Веб-сайти зі спеціальним написанням кодів;
- Сайти з розгортанням немасштабного сервера на Linux, Apache та MySQL;
- Веб-сайти, які потребують широкої обробки зображень;
- Веб-сайти та веб-програми (сценарії на стороні сервера)
- Настільні (GUI) програми

Якщо порівнювати Python Django та PHP ми сможемо дійти до наступних плюсів використання тієї, чи іншої платформи (табл. 2.2.):

Python Django	PHP
Швидший розвиток	Попередньо написані сценарії - постачається із заздалегідь написаними, готовими до використання кодами, що економить час на розробку.
Масштабованість (надає багато можливостей для плавного масштабування і не відстаючи від зростаючих потреб)	Крос-платформа - Підтримується основними операційними системами, включаючи Linux, Solaris, UNIX, macOS та Windows.
Безпека (полегшує створення захищених веб-сайтів та додатків та захищає їх від поширених атак, таких як підробка міжсайтових запитів, введення SQL, розкрадання тощо)	Крива навчання (Легка для вивчення мова з послідовним і логічним синтаксисом, що поділяє подібність із C)
Гнучкість (підтримує швидкі зміни під час розробки завдяки чіткому	Підключення до бази даних (сумісний з такими популярними

програмуванню та безлічі бібліотек та пакетів)	службами баз даних, як MySQL або MariaDB, і забезпечує безперешкодний обмін даними, мінімізуючи час на створення веб-додатків)
Зручне для машинного навчання: (перевага віддається алгоритмам машинного навчання через його обчислювальні та статистичні можливості)	Швидкість - Підсилює залучення користувачів та рейтинг SEO, швидше завантажуючи веб-сторінки без перешкод.

Таблиця 2.2. Переваги використання Python Django та PHP.

Однак серед вищеописаних переваг не обійтися ~~«без ложки дьогтю в бочці меду».~~ Наприклад суттєвими недоліками Python Django є

- Один запит за раз: на відміну від інших популярних фреймворків, Django не може обробляти декілька запитів одночасно, що ускладнює розробникам роботу над базовим кодом.
- Не підходить для невеликих проектів: Django - це середовище, що вимагає великого кодування і займає багато пропускної здатності та часу обробки серверів. Тож якщо ви не плануєте розширювати проект у майбутньому, можливо, ви захочете довго подумати, перш ніж пройти через усі клопоти.
- Повільна еволюція: Джанго також вважається монолітним. Всі розроблені модулі повинні бути сумісними із зворотним зв'язком, тим самим сповільнюючи розвиток фреймворку.

Що ж до PHP, то його недоліками вважаються наступні пункти:

- Need for Interpreter - Вам знадобиться додаткова програма перекладача, яка дозволить інтерпретувати коди мовою, зручною для комп'ютера, що уповільнює швидкість веб-сайтів.

- Відсутність рівномірності - відсутність однорідності структурних схем структур може збільшити витрати на найм нових людських ресурсів.
- Проблеми зі швидкістю - Розробники потребують доповнень для кращої взаємодії з користувачем, що сповільнює роботу веб-сайту.[25]

2.4. Засоби та методи створення клієнтської частини веб-додатку

Клієнтська складова - це те, що бачить користувач на екрані браузера. Веб-проект будується з тексту, зображень, посилань, списків, форм введення даних, таблиць і тому подібне. По суті, веброзробка інтернет-сайту полягає в розміщенні всіх його текстових і графічних елементів в потрібному місці і надання їм необхідних форм/властивостей.

Існує три основних мови верстки:

- HTML-розмітка. Вона дозволяє створювати певні текстові / графічні об'єкти і розміщувати їх на екрані.
- CSS таблиці стилів. Якщо верстати тільки елементами розміченими за допомогою HTML, то багато хто з них будуть виглядати нудно, однаково, а код матиме захаращений вид. CSS надає можливість редагувати позиціонування і зовнішній вигляд веб-сторінки великою кількістю різних способів. Дизайн такого ресурсу стає красивішим і унікальним.
- JavaScript - а ось це вже мова веб-програмування.

Поряд з основними технологіями, є також і інші, що прискорюють процес розробки. Вони вже характерні для більш просунутих користувачів, активно застосовуються в топових веб-студіях таких як https://webkitchen.kiev.ua або при створенні сайтів професійними фрілансерами. В цілому, дозволяють працювати більш ефективно. це:

- SASS - використовується для поліпшення структури CSS-коду і збільшення зручності внесення правок в нього;

- jQuery - бібліотека мови JavaScript, яка допомагає в рази скоротити процес написання програмного коду, при цьому не на шкоду його призначенням;
- Emmet - спеціальний плагін під текстові редактори, за допомогою якого туди додаються готові конструкції для прискорення написання рутинного HTML / CSS коду

HTML (*HyperText Markup Language* – «мова гіпертекстової розмітки») – задає розташування елементів сторінок, а так же зв'язок сторінок між собою за допомогою посилань. Він дозволяє створювати блоки, таблиці, маркіровані та нумеровані списки, заголовки і параграфи. Але, вміючи розставляти об'єкти, HTML не вміє робити їх «красивими».

HTML використовує розмітку ("markup") для відображення тексту, зображень та іншої інформації в веб-браузері. HTML-розмітка включає в себе спеціальні "елементи", такі як <head>, <title>, <body>, <header>, <footer>, <article>, <section>, <p>, , < span>, <img>, <aside>, <audio>, <canvas>, <datalist »,« details »,« embed>, <nav>, <output>, <progress »,« video> і багато інших. [26]

Рис. 2.2. Структура HTML-розмітки

Для того щоб сторінка стала «красивою» використовується CSS. Cascading Style Sheets – це мова ієрархічних правил (таблиць стилів (en-US)), який використовується для представлення зовнішнього вигляду документа, написаного на HTML або XML (en-US) (включаючи різні мови XML, такі як SVG (en-US) і XHTML). CSS описує, яким чином елемент повинен відображатися на екрані, на папері, голосом або з використанням інших медіа

засобів. Інакше кажучи – це набір правил, які вказують параметри того чи іншого елемента такі як розмір, колір, позиція, тіні, фони і так далі.

Таким чином, зовнішній вигляд сторінки формується за рахунок взаємодії HTML і CSS. Взаємодія між HTML та CSS здійснюється за допомогою селекторів — спеціальних позначень, які вказують на конкретну вибірку HTML елементів до яких потрібно застосувати блок властивостей.

HTML в поєднанні з CSS має дивовижні можливості, але світ не стоїть на місці і розробники постійно створюють нові інструменти для підвищення своєї ефективності. Яскравими прикладами таких інструментів є CSS препроцесори та CSS бібліотеки або фреймворки.

CSS препроцесор (CSS preprocessor) - це програма, яка має свій власний синтаксис (syntax (en-US)), але може згенерувати з нього CSS код. Існує безліч препроцесорів. Більшість з них розширюють можливості чистого CSS, додаючи такі опції як: домішки, вкладені правила, селектори спадкування та ін. Ці особливості полегшують роботу з CSS: спрощують читання коду і його подальшу підтримку.[27] Найуживанішими CSS препроцесорами є: Less, Sass та Stylus. Відрізняються вони за синтаксисом і функціональними можливостями.

CSS фреймворк — набір готових рішень, створений для спрощення роботи розробника, пришвидшення розробки і зменшення кількості помилок.

Найбільш популярними є: Bootstrap, Foundation, Materialize CSS та інші.

Створений та розроблений Google, Material Design - це мова дизайну, що поєднує в собі класичні принципи успішного дизайну, а також інновації та технології. Використовує сітку Bootstrap.

Крім HTML та CSS в розробці клієнтської частини веб-сторінок активно використовується мова JavaScript.

JavaScript - це мова програмування, що дозволяє створити веб-історію в інтерактивному режимі, оскільки вона реагує на дії користувачів.

Послідовність інструкцій (що називається програмою, скриптом або

сценарієм) виконується інтерпретатором, вбудованим в звичайний Web - браузер. Іншими словами, код програми вбудовується в HTML - документ і виконується на боці клієнта. Для виконання програми не потрібно навідь перезавантажувати Web -сторінку, всі програми виконуються в відповідь на будь-яку подію. Наприклад, перед відправленням даних форми можна перевірити їх на допустимі значення і, якщо значення не відповідають очікуваним, заборонити відправлення даних.

В проекті буде частково використовуватись JavaScript бібліотека jQuery, яка є необхідною залежністю для інтерактивних компонентів Bootstrap. JQuery – це бібліотека, що фокусується на полегшенні взаємодії з JavaScript та HTML.

РОЗДІЛ 3.
ПРОГРАМНА РЕАЛІЗАЦІЯ ВЕБ-ДОДАТКУ ДЛЯ РОБОТИ З РОЗКЛАДОМ ВИКЛАДАЧА

3.1 Встановлення та налуштування Web-pack'а та БД

Налаштування БД.

Процес розробки почнемо зі встановлення графічного кліенту для управління БД PostgreSQL – pgAdmin 4.

Чому була обрана саме ця платформа? Вона була написана на Python та jQuery і підтримує всі функції PostgreSQL. Мы сможемо використовувати pgAdmin для будь-яких операцій, починаючи з запису базових SQL-запитів, завершуючи здійсненням моніторингу наших баз даних і налаштуванню продвинутих архітектур БД.

Встановлювати будемо з офіційного сайту [28].

Після встановлення пакету програм, відкриємо pgAdmin 4 (рис. 3.1.) та вводимо пароль суперкористувача «postgres», який був вказаний при встановленні PostgreSQL (рис. 3.2.).

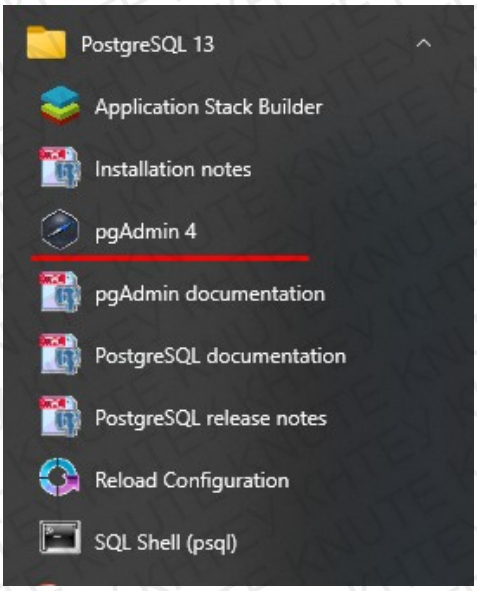


Рис. 3.1. Графічний редактор для роботи с БД pgAdmin 4

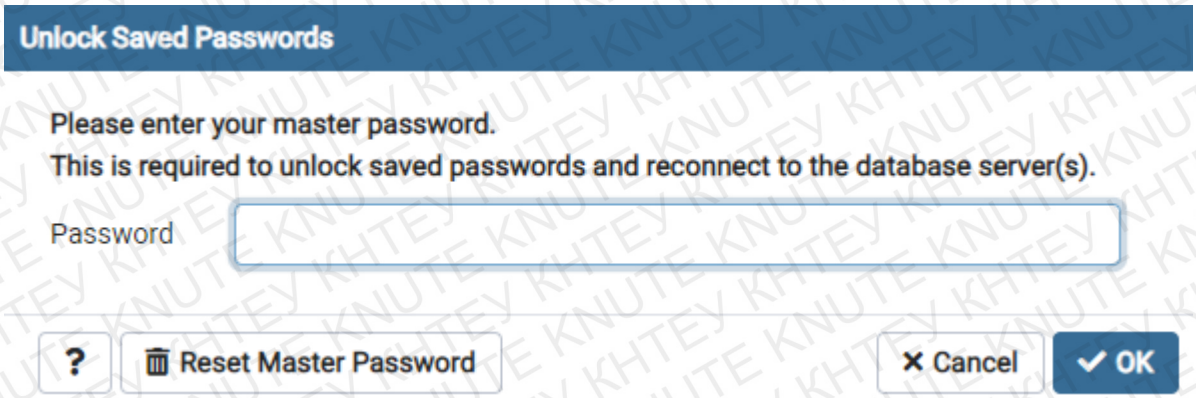


Рис. 3.2. Вікно для входу для входу на сервер

Після успішного логіну нам відкривається вміст серверу:

- в вузлі Databases ми можемо побачити всі наявні бази даних;
- вузол Login / Group Roles, який призначений для управління користувачами і їх ролями;
- Tablespaces дозволяє управляти місцем зберігання файлів баз даних.

Було вирішено зробити дві бази даних: 1 – для локального серверу, 2 – для роботи серверу на хостингу. Для цього натиснемо правою кнопкою миші на вузол Databases. І далі в контекстному меню виберемо Create-> Database. На виході отримаємо 2 сервера зі своїми базами даних (рис 3.3.):

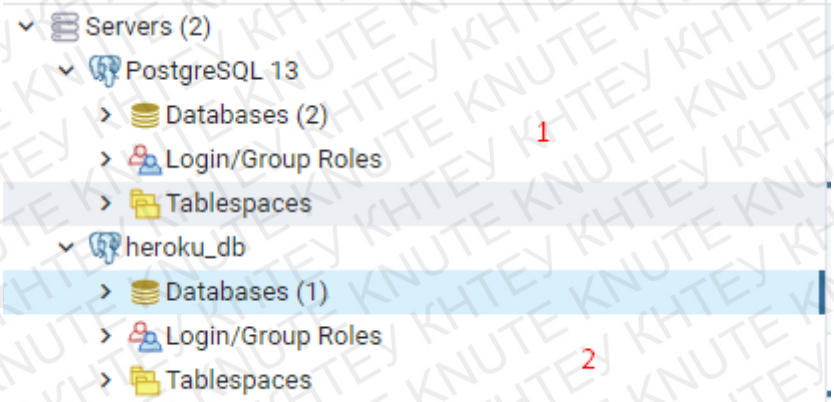


Рис. 3.3. Сервери для роботи додатку

Для більшої зручності створимо 3 таблиці (рис 3.4.):

- schedule_groups – відповідає за розподіл академічних груп у розкладі викладача (рис 3.5.);

- schedule_schedule – це основна таблиця, в якій відображено розклад, розподілений по назві предмету, тижням, парам та дням тижня (рис 3.6.);
- schedule_teacher – це таблиця з прізвищами викладачів (рис. 3.7.).

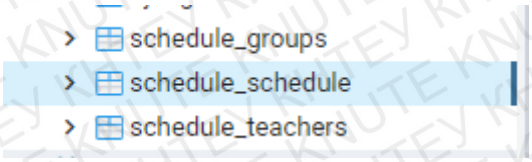


Рис. 3.4. Таблиці з даними, необхідними для створення додатку

id [PK] integer	group_name character varying (200)
1	IT 1-4
2	IT 1-5
3	IT 1-6
4	IT 1-11
5	IT 1-12
6	IT 2-8
7	IT 2-9
8	IT 2-10
9	IT 3-1
10	IT 3-8
11	IT 3-9
12	IT 3-10
13	IT 4-9
14	IT 4-10
15	IT 4-13

Рис. 3.5. Назви груп

Data Output | Explain | Messages | Notifications

id [PK] integer	subject character varying (200)	week integer	para integer	day integer	group_id integer	teacher_id integer
1	ІнформТехн у ПрофДіял Лб2	1	6	3	19	1
2	ІнформТехн у ПрофДіял Лб2	1	7	3	19	1
3	ІнформТехн у ПрофДіял Лб2	1	5	4	17	1
4	ІнформТехн у ПрофДіял Лб2	1	6	4	17	1
5	ІнформТехн у ПрофДіял Лб2	2	3	3	19	1
6	ІнформТехн у ПрофДіял Лб2	2	4	3	19	1
7	ІнформТехн у ПрофДіял Лб2	2	5	5	17	1
8	ІнформТехн у ПрофДіял Лб2	2	6	5	17	1
9	Штучний інтелект Лб1	1	1	1	14	2
10	Штучний інтелект Лб1	1	2	1	14	2
11	Штучний інтелект Лб2	1	3	1	14	2
12	Штучний інтелект Лб2	1	4	1	14	2
13	Штучний інтелект Лб1	1	5	1	13	2
14	Штучний інтелект Лб1	1	6	1	13	2
15	Штучний інтелект Л	1	1	2	13	2

Рис. 3.6. Розклад навчального процесу

Data Output | Explain | Messages | Notifications

	id [PK] integer	teacher_name character varying (200)	user_id integer
1	1	Дивак В.В.	3
2	2	Демідов П.Г.	4
3	3	Козлов В.В.	5
4	4	Краскевич В.Є.	6
5	5	Пурський О.І.	7
6	6	Самойленко Г.Т.	8
7	7	Селіванова А.В.	9
8	8	Тищенко І.А.	10
9	9	Філімонова Т.О.	11
10	10	Шклярський С.М.	12
11	11	Юрченко Ю.Ю.	13

Рис. 3.7. Список викладачів

Налаштування Web-pack'а

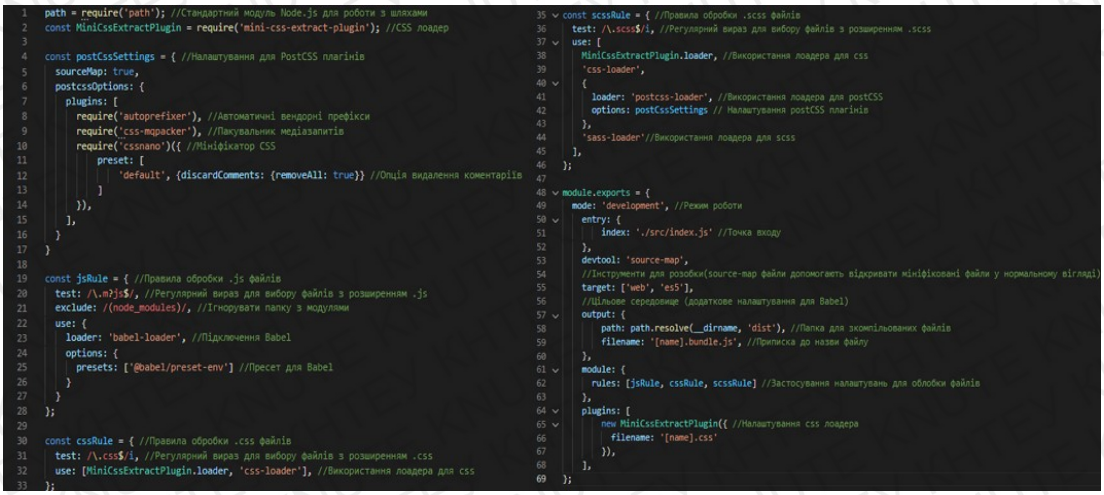
Функціональні можливості збірки напряму залежать від потреб проекту. Для того, щоб нам легше працювалося, налаштуємо скрипти для запуску webpack в файлі package.json (рис. 3.8.).

Новий файл матиме назву «webpack.config.js» та міститиме наступні функції (рис. 3.9.) :

- PostCssSettings – налаштування для PostScc плагінів;
- jsRule – правила обробки .js файлів;
- cssRule – правила обробки .css файлів.

```
{
  "name": "sc",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  Debug
  "scripts": {
    "build": "webpack",
    "dev": "webpack --watch"
  },
  "browserslist": [
    "> 1%",
    "last 3 version"
  ],
  "author": "",
  "license": "ISC",
  "dependencies": {
    "materialize-css": "^1.0.0-rc.2",
    "node-sass": "^5.0.0",
    "sass-loader": "^11.0.1"
  },
  "devDependencies": {
    "@babel/core": "^7.13.10",
    "@babel/preset-env": "^7.13.12",
    "autoprefixer": "^10.2.5",
    "babel-loader": "^8.2.2",
    "css-loader": "^5.2.0",
    "css-mqpacker": "^7.0.0",
    "cssnano": "^4.1.10",
    "exports-loader": "^2.0.0",
    "mini-css-extract-plugin": "^1.3.9",
    "postcss-loader": "^5.2.0",
    "regenerator-runtime": "^0.13.7",
    "webpack": "^5.28.0",
    "webpack-cli": "^4.5.0",
    "webpack-dev-server": "^3.11.2"
  }
}
```

Рис. 3.8. Вміст файлу package.json



```
path = require('path'); //Стандартний модуль Node.js для роботи з шляхами
const MiniCssExtractPlugin = require('mini-css-extract-plugin'); //CSS лоадер

const postCssSettings = { //Налаштування для PostCSS плагінів
  sourceMap: true,
  postcssOptions: {
    plugins: [
      require('autoprefixer'), //Автоматичні вендорні префікси
      require('css-mqpacker'), //Пакувальник медіазапитів
      require('cssnano')({ //Мініфікатор CSS
          preset: [
              'default', {discardComments: {removeAll: true}} //Опція видалення коментаріїв
          ]
      }),
    ],
  }
}

const jsRule = { //Правила обробки .js файлів
  test: /\.m?js$/, //Регулярний вираз для вибору файлів з розширенням .js
  exclude: /(node_modules)/, //Ігнорувати папку з модулями
  use: {
    loader: 'babel-loader', //Підключення Babel
    options: {
      presets: ['@babel/preset-env'] //Пресет для Babel
    }
  }
};

const cssRule = { //Правила обробки .css файлів
  test: /\.css$/i, //Регулярний вираз для вибору файлів з розширенням .css
  use: [MiniCssExtractPlugin.loader, 'css-loader'], //Використання лоадера для css
};

const scssRule = { //Правила обробки .scss файлів
  test: /\.scss$/i, //Регулярний вираз для вибору файлів з розширенням .scss
  use: [
    MiniCssExtractPlugin.loader, //Використання лоадера для css
    'css-loader',
    {
      loader: 'postcss-loader', //Використання лоадера для postCSS
      options: postCssSettings // Налаштування postCSS плагінів
    },
    'sass-loader'//Використання лоадера для scss
  ],
};

module.exports = {
  mode: 'development', //Режим роботи
  entry: {
      index: './src/index.js' //Точка входу
  },
  devtool: 'source-map',
  //Інструменти для розробки(source-map файли допомагають відкривати мініфіковані файли у нормальному вигляді)
  target: ['web', 'es5'],
  //Цільове середовище (додаткове налаштування для Babel)
  output: {
      path: path.resolve(__dirname, 'dist'), //Папка для зкомпільованих файлів
      filename: '[name].bundle.js', //Приписка до назви файлу
  },
  module: {
    rules: [jsRule, cssRule, scssRule] //Застосування налаштувань для облобки файлів
  },
  plugins: [
      new MiniCssExtractPlugin({ //Налаштування css лоадера
        filename: '[name].css'
      }),
  ],
};
```

Рис. 3.9. Основні налаштування webpack

Отже основні налаштування нашої бази даних та прописання скриптів завершено. Перейдемо до освновної частини нашої роботи.

3.2. Стилізація веб-сайту з використанням Materialize

З метою полегшення і прискорення процесу стилізації було прийнято рішення використати фреймворк Materialize, який являє собою набір готових

компонентів в стилі Material Design. Використання готових компонентів дає змогу сконцентруватись на процесі створення логіки додатку (рис. 3.10.).

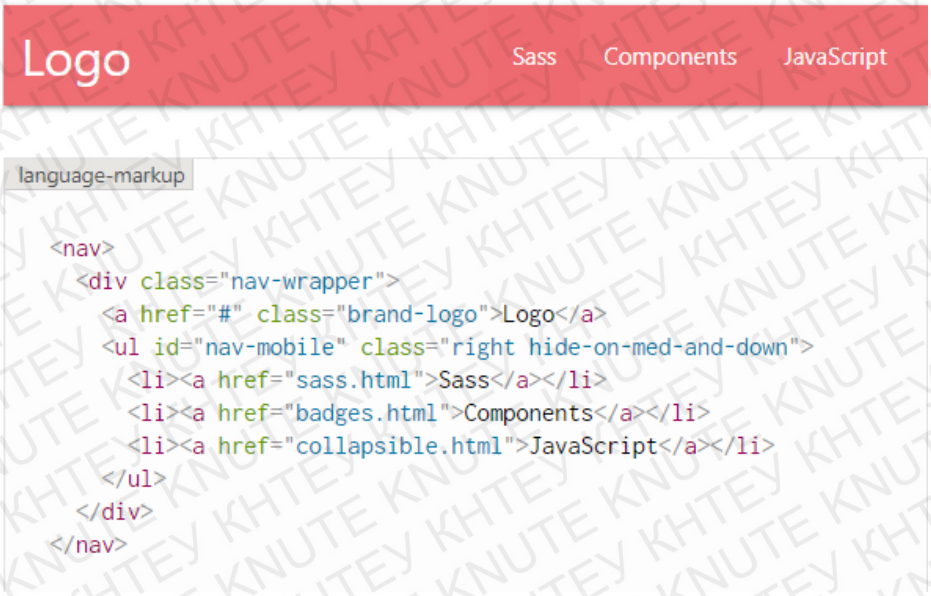


Рис. 3.10. Приклад готового компоненту з документації Materialize

Також для зміни зовнішнього виду та поводження деяких компонентів були використані власні стилі. Власне, CSS був написаний з використанням препроцесора SCSS та використані наступні його особливості: модулі та вкладені стилі (рис. 3.11.).

```
.mb-0 {margin-bottom: 0;}
.mb-1x {margin-bottom: 10px;}
.mb-2x {margin-bottom: 20px;}
.mb-3x {margin-bottom: 30px;}
.mb-4x {margin-bottom: 40px;}
.mb-5x {margin-bottom: 50px;}

.my-s0 {
    @media screen {
        margin-top: 0; margin-bottom: 0;
    }
}
.my-s1x {
    @media screen {
        margin-top: 10px; margin-bottom: 10px;
    }
}
```

Рис. 3.11. Класи для зовнішніх відступів

Результатом стилізації є наступний інтерфейс користувача (рис. 3.12.)

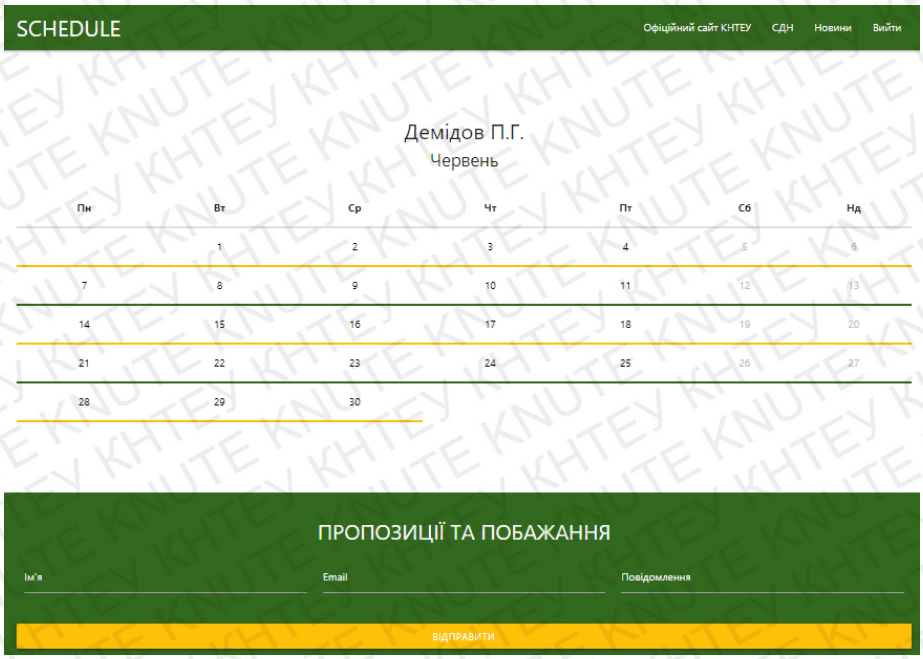


Рис. 3.12. Інтерфейс користувача

Для написання html-розмітки активно використовувався шаблонізатор Django – jinja2, який дозволяє створювати динамічну розмітку шляхом передавання в шаблони даних. Також дає можливість використовувати розгалудження та цикли всередині шаблону (рис. 3.13.).

```
{% for item in schedule %}
<tr>
    <td class="day-col-sm center-align">{{ item.para }}</td>
    <td class="center-align">{{ item.subject }}</td>
    <td class="day-col-sm center-align">{{ item.group }}</td>
</tr>
{% endfor %}
```

Рис. 3.13. Використання jinja2

3.3. Реалізація авторизації користувачів

При переході користувача на сторінку логіну потрібно відрендерити початкову сторінку.

Для цього в масиві urlpatterns потрібно внести новий елемент який є результатом виклику функції path з наступними параметрами (рис. 3.14):

- посилання за яким буде переходити користувач.
- відповідна функція з файлу views.py.

```python
from django.urls import path
from . import views

app_name = 'schedule'
urlpatterns = [
    path('days/', views.days_request, name='days_request'),  #  Відправляємо булевий список
    path('', views.render_calendar, name='calendar'),  #  Рендер календаря
    path('login/', views.login_form, name='login_form'),  #  Рендер логіна
    path('auth/', views.user_login, name='login'),  #  Аутентифікація користувача
    path('schedule/', views.schedule_list, name='schedule_list'),  #
    path('contact/', views.user_contact, name='contact_form'),  #  Відправка повідомлення
    path('logout/', views.logout_user, name='logout')  #  Вихід з аккаунта
]
```

Рис 3.14. Вміст папки urls.py

Опишемо функцію рендеру (рис. 3.15.):

```python
def login_form(request):
    """Функція відрисовування сторінки логіну"""
    if request.user.is_authenticated:  # Перевірка аутентифік
        return HttpResponseRedirect('/')  #  Якщо кристувач ау
    else:  #  Якщо користувач не аутентифікований, то
        return render(request, 'schedule/index.html')  #  Відр
```

Рис. 3.15. Функція відрисовування сторінки логіну

1. Якщо користувач вже аутентифікований – перенаправляємо його на сторінку з календарем.
2. Інакше відмальовуємо сторінку index.html.

Після відмалювання основної сторінки потрібно забезпечити функціонал форми авторизації. Робимо ми наступним чином: елементу path(’auth’) в якості другого параметру надаємо функцію аутентифікації з файлу views.py (рис. 3.16.).

```python
def user_login(request):
    """Функція для обробки запиту та аутентифікації користувача в аккаунт"""
    if request.method == 'POST':  #  Перевірка, чи метод запиту POST
        password = request.POST['password']  #  Отримуємо password із запиту
        username = request.POST['username']  #  Отримуємо username із запиту
        user = authenticate(username=username, password=password)  #  Аутентифікуємо користувача
        if user is not None and user.is_active:  #  Перевірка, чи аутентифікація пройшла успішно
            login(request, user)  #  Користувач заходить у свій профіль
            return HttpResponse('Successful')  #  У разі успіху відправляємо Successful
        else:  #
            raise ValidationError('Invalid value', code='invalid')  #  У разі невдачі піднімаємо помилку 500
```

Рис. 3.16. Функція для обробки запиту та аутентифікації користувача в аккаунт

Якщо метод запиту POST – ми отримуємо пароль та ім‘я користувача.

З використанням отриманим даних ми викликаємо функцію authenticate і передаємо наступні параметри. Якщо користувач є в БД і він активний - визиваємо функцію login.

Якщо користувача немає - визиваємо класс ValidationError та видаємо помилку на клієнт.

Тепер опишемо функціонал авторизації з клієнтської сторони додатку.

В головному файлі логіки необхідно назначити обробник події submit на елемент form (рис. 3.17.). При спрацьовуванні обробника події submit, викликається асинхронна функція. У випадку успіху користувач перенаправляється на сторінку з календарем, якщо дані користувача не вірні - відмальовується помилка.

```
if(renderer.isNodeExist('.login-form')) { //Якщо на сторінці є форма для авторизації
    const loginForm = document.querySelector('.login-form');

    loginForm.addEventListener('submit', async(event) => {//Обробник події відправки форми
        try {
            await requester.sendForm(event); //Відправляємо форму
        }catch(err) {
            renderer.error.showFormError(err); //Відмальвуємо помилку
        }
    });
}
```

Рис. 3.17. Логіка авторизації користувача на сайті

Детальніше опишемо функцію sentForm (рис. 3.18.):

- відміняємо перезавантаження сторінки;
- отримуємо об‘єкт з даними форми;
- відправляємо запит;
- отримуємо відповідь від сервера і робимо перевірку, якщо статус перевірки «Ок» (200) - перенаправляємо на календар.

```
const sendForm = async(event) => { //Функція відправки форми
    event.preventDefault();//Відміняємо перезавантаження сторінки

    const formData = new FormData(event.target); //Об'єкт з даними форми
    const csrfToken = document
        .querySelector("[name=csrfmiddlewaretoken]").value; //csrfToken для захисту від підробки запитів

    await fetch('/auth/', {// Відправляємо запит
        credentials: 'include',
        method: 'POST',
        mode: 'same-origin',
        headers: {
            'X-CSRFToken': csrfToken
        },
        body: formData
    }).then(res => {
        if(res.ok) {
            window.location.href = '/'; //Якщо все гаразд - перенапрвляємо користувача на сторінку з календарем
        }else {
            throw 'Невірний пароль або логін'; //Інакше генеруємо виключення
        }
    });
};
```

Рис. 3.18. Використання функції sendform для клієнтської сторони додатку

3.4. Розробка інтерактивного календаря

Після авторизації користувач потрапляє на сторінку календаря. Опишемо принцип її роботи. Для цього в масиві urlpatterns внесемо новий елемент який є результатом виклику функції path з наступними параметрами (рис. 3.19.):

- посилання за яким буде переходити користувач «_» (пусте посилання тому, що це головна сторінка);
- функція рендеру сторінки календаря.

```
def render_calendar(request):
    """Функція для відрисовування календаря"""
    teacher_name = Teachers.objects.filter(user=request.user.id).values_list('teacher_name').first()[0]
    current_month = gettext(NOW.strftime("%B"))  #  Отримуємо назву поточного місяця
    return render(request, 'schedule/calendar.html', {'name':teacher_name,
                                                      'month':current_month})  #  Відрисовуємо сторінку
```

Рис. 3.19. Функція для відсортування календаря

Розглянемо функцію рендеру докладніше.

Ми робимо запит до БД. За допомогою моделі Teachers знаходимо ім’я користувача по його ідентифікатору (рис. 3.20.).

```python
from django.db import models
from django.contrib.auth.models import User


class Teachers(models.Model):
    """Таблиця для даних про викладачів"""
    teacher_name = models.CharField(max_length=200)  #  Колонка з іменами викладачів
    user = models.ForeignKey(User, on_delete=models.CASCADE)  #  Колонка зі значенням id користувача


class Groups(models.Model):
    """Таблиця для даних про групи"""
    group_name = models.CharField(max_length=200)  #  Колонка з назвами груп


class Schedule(models.Model):
    """Таблиця для даних про розклад"""
    teacher = models.ForeignKey(Teachers, on_delete=models.CASCADE)  #  Колонка зі значенням id таблиці Teacher
    subject = models.CharField(max_length=200)  #  Колонка з назвами предмета
    week = models.IntegerField()  #  Колонка з номером тижня
    group = models.ForeignKey(Groups, on_delete=models.CASCADE)  #  Колонка зі значенням id таблиці Groups
    para = models.IntegerField()  #  Колонка з номером пари
    day = models.IntegerField()  #  Колонка з номером дня
```

Рис. 3.20. Моделі сайту

Далі знаходимо назву поточного місяця та рендеримо каркас календаря.

Потрібно відправити на клієнт список булевих значень, які позначають активні/неактивні (true/false) дні розкладу. Це завдання виконує функція days_request (рис. 3.21.). Опишемо її:

- отримуємо об‘єкт з даними викладача за його ідентифікатором;
- отримуємо парність першому тижню;
- отримуємо список із тижнів та днів для викладача;
- створюємо пустий список для булевих значень;
- виставляємо лічильник на перший день тижня (Понеділок);
- в тілі циклу заповнюємо список булевих значень;
- знаходимо номер першого дня місяця та кількість днів в цьому місяці;
- робимо зріз лише з тих днів, що належать лише до цього місяця;
- повертаємо клієнту відповідь у вигляді JSON.

```python
def days_request(request):
    """Функція для відправки даних про дні з парами викладача"""
    current_teacher = Teachers.objects.get(user=request.user.id)  #  Дістаємо дані
    current_week = int(request.GET.get('week'))  #  Отримуємо номер першого тижня м
    days = Schedule.objects.filter(teacher=current_teacher).values('week', 'day')

    boolean_days = []  #  Створюємо пустий список для булевих значень
    counter = 1  #  Виставляємо лічильник на перший день тиждня (Понеділок)
    while len(boolean_days) < 42:  #  Цикл по довжині 6-ти тижнів
        if counter == 8:   #  Коли лічильник 8 (Наступний день після Неділі)
            counter = 1  #  Ми переводимо його на 1 (Понеділок)
            if current_week == 1:  #  Якщо поточний тиждень перший
                current_week += 1  #  Змінюємо його на другий
            else:  #  Якщо поточний тиждень другий
                current_week -= 1  #  Змінюємо на перший
        else:  #  Якщо номер дня (лічильник) не дійшов до 8
            counter  #  Повертаємо номер дня (лічильник)

        check = {'week':current_week, 'day':counter}   #  Створюємо словник з дня т
        boolean_days.append(check in days)  #  Додаємо булеве значення в залежності
        counter += 1  #  Збільшуємо лічильник на 1

    month_range = calendar.monthrange(NOW.year, NOW.month)  #  Знаходимо номер перш
    boolean_days = boolean_days[month_range[0]:month_range[1]+month_range[0]]  #  P
    return JsonResponse({'boolean_days':boolean_days})  #  Віддаємо JSON
```

Рис. 3.21. Функція days_request

Переходимо до логіки клієнтської частини (рис. 3.22.):

- перевіряємо чи міститься на сторінці блок з календарем;
- отримуємо об‘єкт з даними про місяць;
- отримуємо порядок тижнів (парний або непарний);
- за допомогою функції getActiveDays отримуємо з серверу дані про активність днів і модифікуємо, з їх використанням, об‘єкт з даними про місяць.

```javascript
if(renderer.isNodeExist('.js-calendar')) { //Якщо на сторінці є календар
    const monthObj = dateFuncs.getMonth(), //Об'єкт з даними про місяць
        weekOrder =  monthObj.startsFromFirst ? 1 : 2; //Порядок тижнів

    const data = addActive( //Модифікуємо масив з даними про місяць, додавши в нього дані про активність днів
        monthObj,
        await requester.getActiveDays(weekOrder) //Отримуємо з сервера  дані про активність днів
    );

    renderer.drowCalendar('.js-calendar', data); //Відмальовуємо місяць на основі даних
}
```

Рис. 3.22. Логіка календаря в клієнтській частині

Детальніше розглянемо функцію getMonth (рис.3.22). Завдяки цій функції ми отримуємо масив, що містить дані про кожен окремий день. Поки що, кожен з цих внутрішніх масивів містить лише номер дня. Дні минулого місяця містять 0. Також, отриманий в результаті дії цієї функції, об’єкт міститиме інформацію про порядок тижнів.

При виконанні, функція getMonth задіє деякі утилітарні функції, а саме:

- nullify - додає в масив внутрішні масиви з 0;
- addDays - додає в масив внутрішні масиви з номерами днів;
- isFirstWeek - визначає чи починається місяць з 1 тижня, повертає будете значення.

Оскільки ми вже отримали необхідні дані, настав час реалізації функції рендеру календарю (рис. 3.23.).

1. Створюємо утилітарні функції checkWeek (бере два значення і вертає одне з низ в залежності від тижня), newWeek (закриває html елемент минулого тижня і відкриває новий, додає тиждень до коду місяця) , fillDay (заповнює і повертає html елемент дня).

2. Перебираємо масив днів і заповнюємо html код місяця.

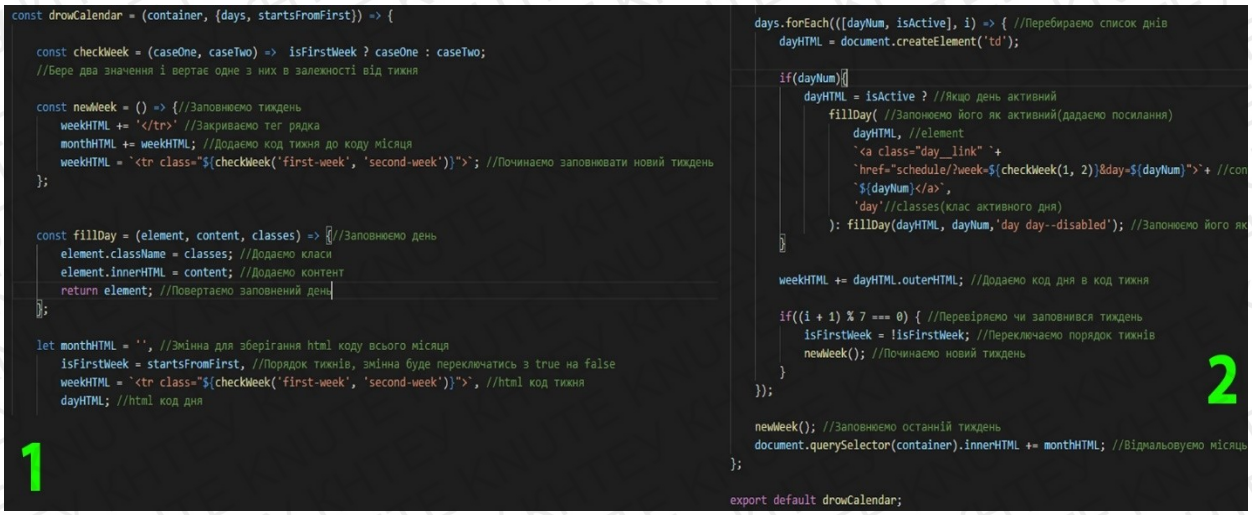


Рис. 3.23. Функція рендеру календаря

В результаті роботи функції рендерингу ми отримуємо календар на поточний місяць. Кожен активний день містить посилання, яке перенаправляє користувача на сторінку з розкладом на цей день. Для прикладу, нище зазначений розклад на червень для Демідова П. Г. (рис. 3.20.).

Рис. 3.20. Сторінка з розкладом викладача на місяць

Функціонування сторінки з розкладом на конкретний день забезпечується обробкою переходу користувача за посиланням «schedule/». При переході користувача за посиланням, запускається функція schedule_list. Опишемо принцип її роботи (рис. 3.21.):

- дізнаємося поточний тиждень із запиту;
- дізнаємося поточний день із запиту;
- отримуємо дані про викладача за його ідентифікатором;
- знаходимо номер дня тижня для поточного дня;
- отримуємо поточну дату;
- дістаємо дані, що відповідають розкладу викладача на поточний день, фільтруємо їх, сортуємо за номером пари;
- перебираємо дані, заміняємо ідентифікатор на назву;
- передаємо дані в html-шаблон.

```python
def schedule_list(request):
    """Відрисовуємо розклад для обраного дня"""
    current_week = int(request.GET.get('week'))  #  Дізнаємося поточний тиждень із запиту
    current_day = int(request.GET.get('day'))  #  Дізнаємося поточний день із запиту
    teacher_request = Teachers.objects.filter(user=request.user.id).values_list().first()

    day_of_the_week = datetime.datetime(NOW.year, NOW.month, current_day).weekday() + 1
    current_date = datetime.date(NOW.year, NOW.month, current_day)  #  Отримуємо поточну

    schedule = Schedule.objects.values('para', 'subject', 'group')  #  Дістаємо дані з па
    schedule = schedule.filter(week=current_week, teacher=teacher_request[0], day=day_of_
    schedule = schedule.order_by('para')  #  Сортуємо їх по номеру пари

    for e in schedule:  #  Цикл для заміни id групи на назву
        e['group'] = Groups.objects.filter(id=e['group']).values_list('group_name').first(

    schedule_dict = {'schedule':schedule,  #  Заповнюємо словник парами
                     'name':teacher_request[1],  #  Заповнюємо словник ім'м викладача
                     'day':current_date}  #  Заповнюємо словник датою
    return render(request, 'schedule/daySchedule.html', schedule_dict)  #  Відрисовуємо с
```

Рис. 3.21. Функція schedule_list

ВИСНОВКИ

У роботі було проведено аналіз сучасних інформаційних технологій щодо їх використання у сучасному освітньому процесі. Показано, що одним із найважливіших принципів функціонування високопродуктивних, гнучких та клієнтоорієнтованих сервісів можна забезпечити тільки за рахунок використання сучасних інформаційних технологій і глобальних телекомунікаційних мереж. Показано, що розвиток глобальної інформаційної інфраструктури і зростання можливостей IT-технологій створюють принципово нову ситуацію в освіті. Це дозволить скорочувати час реакції на нові запити та забезпечити постійну актуальність інформації.

Для вирішення поставленої в роботі мети було проаналізовано поняття «електронний розклад». Показано, що одним із передових напрямків підвищення рівня інформованості викладачів щодо їх розкладу є створення актуального web-ресурсу. Для цього було проведено аналіз технологій інформаційних систем, під час якого було з’ясовано структуру інформаційної системи, що буде використана для створення даного web-додатку.

В другому розділі були досліджені способи структурування даних та сучасні методи верстання сайтів. Аналіз методів структурування даних дав змогу порівняти методи організації даних та їх застосування у різних завданнях. Аналіз сучасних СУБД, дав змогу обрати оптимальну для використання базу даних, що не перезавантажуватиме сервер, та дасть змогу швидкого та якісного створення масивів інформації, яка міститься в БД.

Зважаючи на те, що сучасні інформаційні системи навчальних закладів засновані на мережевих технологіях, наслідком є скорочення часу реакції на нові запити і забезпечуються зв’язки, що постійно оновлюються та забезпечення постійної актуальності інформації.

Аналіз методів верстання сайтів дозволяє обрати найбільш зручний метод створення та відображення веб-сторінки на різних пристроях, їх пристосованість до реалізації певних завдань. Було порівняно між собою

декілька мов програмування, такі як: Python та PHP. Як висновок було обрано мову програмування Python, а саме фреймворк Django, за його модульність та гнучкість у використанні. Функціональний, інтуїтивно простий та що містить достатню кількість корисної інформації сайт допоможе у процесі підвищення інформованості викладача.

В третьому розділі здійснилась розробка інформаційної системи «електронний розклад викладача». На основі проведеного аналізу створено базу даних на основі PostgreSQL. Дана технологія гарантує стабільно роботу web-додатку, в парі с фреймворком Python Django, та зручність редагування файлів для їх подальшого використання.

Було створено структуру інформаційної системи, що відповідає поставленій задачі. Структуру та дизайн сайту створено за допомогою програми Visual Studio Code. В результаті цього створений інтуїтивно зрозумілий інтерфейс користувача навігації по сайту, що допомогає йому використовувати веб-ресурс в повному обсязі. Даний інтерфейс являє собою інтерактивний календар, який налаштовується під кожного викладача окремо, відповідно до парності тижня, місяця та ідентифікатора користувача.

За результатами розробки отримано сайт, який можна використовувати у навчальному процесі. Сайт включає такі можливості: перехід по веб-сторінкам, пов’язаним із навчальним закладом, аутентифікацію користувача відповідно до його логіну та паролю та зворотній зв’язок.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Організація освітнього процесу. [Електронний ресурс]. – Режим доступу: https://knute.edu.ua/blog/read/?pid=7330&uk.
2. Інструкція щодо розкладу занять. [Електронний ресурс]. Режим доступу: https://www.oa.edu.ua/publik_information/instruktsiya_shchodo_skladannya_rozkladu.pdf
3. Положення про розклад навчальних занять. [Електронний ресурс]. Режим доступу: https://www.kname.edu.ua/images/Files/Normativny_Dokumenty/pologennya_pro_rosklad_zanyat.pdf
4. Про державну статистику: Закон України від 17 вересня 1992 року № 2614-XII // Відомості Верховної Ради України. – 1992. - № 43. – Ст. 608.
5. Інформаційні системи та їх види. [Електронний ресурс]. Режим доступу: http://www.kievoit.ippo.kubg.edu.ua/kievoit/2013/95/95.html
6. Types of Information Systems - Components and Classification of Information Systems. [Електронний ресурс]. Режим доступу: https://www.managementstudyguide.com/types-of-information-systems
7. Українська бібліотечна енциклопедія. [Електронний ресурс]. Режим доступу: https://ube.nlu.org.ua/article/%D0%91%D0%B0%D0%B7%D0%B0%20%D0%B4%D0%B0%D0%BD%D0%B8%D1%85
8. Системи управління базами даних (СУБД), їх основні можливості та функції. [Електронний ресурс]. Режим доступу: https://vseosvita.ua/library/sistemi-upravlinna-bazami-danih-subd-ih-osnovni-mozlivosti-ta-funkcii-25461.html

9. What is a database management system? Purpose and function. [Електронний ресурс]. Режим доступу: https://study.com/academy/lesson/what-is-a-database-management-system-purpose-and-function-

10. Інформаційні технології та технічні засоби навчання - Буйницька О.П. [Електронний ресурс]. Режим доступу: https://westudents.com.ua/glavy/27490-15-klasifkatsya-nformatsynih-sistem.html

11. Components of information system. [Електронний ресурс]. Режим доступу: https://www.geeksforgeeks.org/components-of-information-system

12. Що таке HTML? [Електронний ресурс]. Режим доступу: https://css.in.ua/article/shcho-take-css_3

13. Простым языком об HTTP. [Електронний ресурс]. Режим доступу: https://habr.com/ru/post/215117/

14. Інформаційні системи і технології на підприємствах :підручник /В.Л. Плескач, Т.Г. Затонацька. —К. :Знання, 2011. —718с.

15. Що таке бази даних? [Електронний ресурс]. Режим доступу: http://apeps.kpi.ua/shco-take-basa-danykh

16. Офіційний сайт компанії UA5.ORG. [Електронний ресурс]. Режим доступу: https://www.ua5.org/database/189-reljacjjna-baza-danikh.html

17. Офіційний сайт компанії Oracle. [Електронний ресурс]. Режим доступу: https://wikis.oracle.com/pages/viewpage.action?pageId=27394602

18. Том Армстронг ActiveX – Создание Web-приложений, глава 6/ Том Армстронг : БХВ-Петербург, 1998.

19. Почему Python? [Електронний ресурс]. Режим доступу: https://khashtamov.com/ru/why-python/

20. Национальная библиотека им. Н. Э. Баумана. [Електронний ресурс]. Режим доступу: https://ru.bmstu.wiki/Python
21. Краткий обзор языка Python. [Електронний ресурс]. Режим доступу: http://www.helloworld.ru/texts/comp/lang/python/python2/index.htm
22. The official home of the Python Programming Language. [Електронний ресурс]. Режим доступу: https://www.python.org/
23. Где применяется Python? [Електронний ресурс]. Режим доступу: https://www.mvoronin.pro/en/blog/post-75
24. Working on Graphite-web. [Електронний ресурс]. Режим доступу: https://graphite.readthedocs.io/en/latest/development.html
25. Django vs. PHP: Who Will win the Battle of Backend Dominance? [Електронний ресурс]. Режим доступу: https://www.simform.com/django-vs-php/#section0
26. HTML Руководства для начинающих. [Електронний ресурс]. Режим доступу: https://developer.mozilla.org/ru/docs/Web/HTML
27. CSS препроцессор. [Електронний ресурс]. Режим доступу: https://developer.mozilla.org/ru/docs/Glossary/CSS_preprocessor
28. Офіційний сайт pgAdmin. [Електронний ресурс]. Режим доступу: https://www.pgadmin.org/

ЗАУВАЖЕННЯ

- додайте до кожного розділу короткий висновок
- винесіть скріншоти програмного коду у ДОДАТКИ,
- дайте посилання на працюючий веб-сайт, або на репозиторій GitHub