ДЕРЖАВНИЙ ТОРГОВЕЛЬНО-ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ

Кафедра комп'ютерних наук та інформаційних систем

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«Розробка системи обліку та аналізу навчального навантаження кафедри»

Студента 4 курсу, 10 групи,
спеціальності
122 «Комп'ютерні науки» _____ Донченко Тимофій Юрійович

підпис студента

Науковий керівник
кандидат технічних науки, доцент _____ Філімонова Тетяна Олегівна

підпис керівника

Гарант освітньої програми
кандидат технічних наук, доцент _____ Демідов Павло Георгійович

підпис керівника

Київ 2023

Державний торговельно-економічний університет

Факультет інформаційних технологій
Кафедра комп'ютерних наук та інформаційних систем
Спеціальність 122 «Комп'ютерні науки»

Затверджую
Зав. кафедри_____Пурський О.І.
«20» грудня 2022 р.

Завдання
на випускної кваліфікаційної роботи (проекту): студенту

Донченку Тимофію Юрійовичу
(прізвище, ім'я, по батькові)

1. Тема випускної кваліфікаційної роботи (проекту):
«Розробка системи обліку та аналізу навчального навантаження кафедри»
Затверджена наказом ректора від *«04» грудня 2022 р. № 4111*
2. Строк здачі студентом закінченої роботи *05 червня 2023 року*
3. Цільова установка та вихідні дані до роботи
Мета роботи: проектування і розробка веб-додатку електронного розкладу на кафедрі
Об'єкт дослідження: електронний розклад та його складові.
Предмет дослідження: процес розробки веб-додатку електронного розкладу на кафедрі
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	Філімонова Т.О.	15.12.2022	15.12.2022
2	Філімонова Т.О.	15.12.2022	15.12.2022
3	Філімонова Т.О.	15.12.2022	15.12.2022

6. Зміст випускної кваліфікаційної роботи (проекту): (перелік питань за кожним розділом)

ВСТУП

РОЗДІЛ 1. Теоретичні аспекти моделювання електронного розкладу навантаження кафедри

1.1.Теоретичний аналіз поняття електронний розклад

1.2. Інформаційні системи та їх структура

1.3. Використання електронного розкладу в навчальному процесі

РОЗДІЛ 2. Огляд методик структурування даних та розробки електронного розкладу навантаження кафедри

2.1. Огляд засобів структурування даних для використання у системі електронного розкладу навантаження кафедри

2.2. Методи табличного верстання інтернет-сайтів

2.3. Використання методу float для створення Web-додатку

2.4. Модульна системи верстки у веб-дизайні

2.5. Використання чатботів у месенджері Telegram для електронного розкладу

РОЗДІЛ 3. Створення електронного розкладу навантаження кафедри

3.1. Розробка алгоритму створення електронного розкладу навантаження кафедри

3.2. Моделювання процесу розробки електронного розкладу навантаження кафедри

3.3. Програмна реалізація інформаційної системи електронного розкладу

3.4. Технологія використання розробленого Web-додатку

ВИСНОВКИ

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

7. Календарний план виконання роботи

№ Пор.	Назва етапів випускної кваліфікаційної роботи (проекту):	Строк виконання етапів роботи	
		За планом	фактично
1	2	3	4
1	*Вибір теми випускної кваліфікаційної роботи (проекту)*	*04.10.2022*	*04.10.2022*
2	*Розробка та затвердження завдання на випускну кваліфікаційну роботу (проект)*	*15.12.2022*	*15.12.2022*

3	*Вступ*	*03.02.2023*	*03.02.2023*
4	*РОЗДІЛ 1. Теоретичні аспекти моделювання електронного розкладу навантаження кафедри*	*28.02.2023*	*28.02.2023*
5	*РОЗДІЛ 2. Огляд методик структурування даних та розробки електронного розкладу навантаження кафедри*	*06.04.2023*	*06.04.2023*
6	*РОЗДІЛ 3. Створення електронного розкладу навантаження кафедри*	*12.05.2023*	*12.05.2023*
7	*Висновки*	*15.05.2023*	*15.05.2023*
8	*Здача випускної кваліфікаційної роботи (проекту) на кафедру науковому керівнику*	*30.05.2023*	*30.05.2023*
9	*Попередній захист випускної кваліфікаційної роботи (проекту)*	*31.05.2023 -01.06.2023*	*31.05.2023 -01.06.2023*
11	*Виправлення зауважень, зовнішнє рецензування випускної кваліфікаційної роботи (проекту)*	*02.06.2023*	*02.06.2023*
12	*Представлення готового зшитого випускної кваліфікаційної роботи (проекту) на кафедру*	*05.06.2023*	*05.06.2023*
13	*Публічний захист випускної кваліфікаційної роботи (проекту)*	*За розкладом роботи ЕК*	*За розкладом роботи ЕК*

8. Дата видачі завдання «15» грудня 2022 р.

Керівник випускної кваліфікаційної роботи (проекту): Філімонова Т. О.
(прізвище, ініціали, підпис)

Гарант освітньої програми Демідов П.Г.
(прізвище, ініціали, підпис)

Завдання прийняв студент-дипломник Донченко Т. Ю.
(прізвище, ініціали, підпис)

9. Відгук керівника випускної кваліфікаційної роботи (проекту):

У випускній кваліфікаційній роботі розроблено систему обліку та аналізу навчального навантаження кафедри. Робота включає аналіз поточного стану, розробку системи та використання програмного інструментарію для обліку та аналізу навчального навантаження.

Випускна кваліфікаційна робота відповідає всім вимогам до випускних кваліфікаційних робіт. Всі поставлені завдання виконані. Випускна кваліфікаційна робота може бути допущена до захисту.

Керівник випускної кваліфікаційної роботи (проекту): Філімонова Т.О.
(підпис, дата)
30.05.2023

10. Висновок про випускну кваліфікаційну роботу (проект):

Випускна кваліфікаційна робота (проект) студента Донченка Т.Ю.
(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)

«_____»_____2023 р.

Анотація

У випускній кваліфікаційній роботі проаналізовано поняття «електронний розклад студента», його можливості в умовах інформатизації навчального процесу, структуру та види інформаційних систем. Розглянуто види структурування даних та види представлення даних. Виконане структурування даних для зручного їх представлення у інформаційній системі. Здійснена розробка діаграм візуалізації інформаційної системи. Розроблено метод відображення структурованих даних на сторінці веб-додатку. Створено веб-додаток інформування студента щодо розкладу занять на поточний чи наступний тиждень.

Ключові слова: веб-додаток, діаграма візуалізації, електронний розклад, інформаційна система.

Annotation

The graduation qualification work analyzes the concept of "electronic student schedule", its capabilities in terms of informatization of the educational process, the structure and types of information systems. Types of data structuring and types of data representation are considered. Structuring of data for their convenient representation in information system is executed. The development of information system visualization diagrams has been carried out. A method for displaying structured data on a web application page has been developed. A web application for informing students about the schedule of classes for the current or next week has been created.

Keywords: web application, visualization diagram, electronic schedule, information system.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, СКОРОЧЕНЬ І ТЕРМІНІВ..... 8
ВСТУП..... 9
РОЗДІЛ 1. 11
ТЕОРЕТИЧНІ АСПЕКТИ МОДЕЛЮВАННЯ ЕЛЕКТРОННОГО РОЗКЛАДУ НАВАНТАЖЕННЯ КАФЕДРИ 11
1.1. Теоретичний аналіз поняття «електронний розклад» 11
1.2. Інформаційні системи та їх структура 12
1.3 Використання електронного розкладу в навчальному процесі..... 16
РОЗДІЛ 2. 17
ОГЛЯД МЕТОДИК СТРУКТУРУВАННЯ ДАНИХ ТА РОЗРОБКИ ЕЛЕКТРОННОГО РОЗКЛАДУ НАВАНТАЖЕННЯ КАФЕДРИ..... 17
2.1. Огляд засобів структурування даних для використання у системі електронного розкладу 17
2.2. Методи табличного верстання інтернет-сайтів 21
2.3. Використання методу float для створення Web-додатку 22
2.4. Модульна система верстки у веб-дизайні 23
2.5. Використання чат-ботів у месенджері Telegram для електронного розкладу. 24
РОЗДІЛ 3. 26
СТВОРЕННЯ ЕЛЕКТРОННОГО РОЗКЛАДУ НАВАНТАЖЕННЯ КАФЕДРИ..... 26
3.1. Розробка алгоритму створення електронного розкладу студента навантаження кафедри..... 26
3.2. Моделювання процесу розробки електронного розкладу студента. 27
3.3. Програмна реалізація інформаційної системи електронного розкладу 30
3.4. Технологія використання розробленого Web-додатку 41
ВИСНОВКИ 43
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 45

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, СКОРОЧЕНЬ І ТЕРМІНІВ

СУБД – система управління базами даних.

NoSQL - нереляційна база даних, яка зберігає та отримує доступ до даних за допомогою ключ-значення.

MongoDB - документо-орієнтована система керування базами даних з відкритим вихідним кодом, яка не потребує опису схеми таблиць.

JSON – (JavaScript Object Notation), текстовий формат обміну даними, що дає змогу описувати об'єкти та інші структури даних.

BSON – (Binary JavaScript Object Notation), двійкове кодування документів, подібних JSON.

AJAX – (Asynchronous Javascript And Xml), технологія звернення до сервера без перезавантаження сторінки.

БД – база даних.

CSS – (Cascading Style Sheets), каскадні таблиці стилів.

HTML – (HyperText Markup Language), мова розмітки гіпертексту.

UML – (Unified Modeling Language), мова графічного опису для об'єктного моделювання в області розробки програмного забезпечення, для моделювання бізнес-процесів, системного проектування та відображення організаційних структур.

API – (Application Programming Interface), набір чітко визначених методів для взаємодії різних компонентів.

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-екзаменаційних 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 розміщені в мережі Інтернет.

Створення електронного розкладу розміщеного в мережі Інтернет дозволить кожному студенту 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 випускної кваліфікаційної роботи є проектування і розробка веб-додатку навчального навантаження на кафедрі. Для досягнення поставленої мети, треба виконати такі завдання:

- Дослідити варіанти створення інформаційної бази;
- Дослідити можливі варіанти верстки веб-сторінки;

- Дослідити можливості бібліотеки JQuery мови Javascript;
- 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.

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 веб-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 студента.

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

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

РОЗДІЛ 1.
ТЕОРЕТИЧНІ АСПЕКТИ МОДЕЛЮВАННЯ ЕЛЕКТРОННОГО РОЗКЛАДУ НАВАНТАЖЕННЯ КАФЕДРИ

1.1. Теоретичний аналіз поняття «електронний розклад»

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 екзаменаційних сесій 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.

1.2. Інформаційні системи та їх структура

Велика кількість сучасної роботи пов’язана з роботою з великим обсягом даних. Дані є основними значеннями або фактами і організовані в базі даних. Багато людей вважають дані синонімами інформації; однак інформація насправді складається з даних, які були організовані з метою відповіді 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 і опрацьованої інформації;
- виведення інформації, призначеної для користувача;
- поширення інформації 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 програмні програми та інші запити. Обробка інформації включає в себе введення даних, обробку даних, зберігання даних, виведення та контроль. На етапі виводу дані подаються у вигляді звітів. [6]

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

База даних - сукупність взаємопов'язаних даних, організована відповідно до певних правил опису, зберігання та маніпулювання, подана у формі, придатній для автоматичного опрацювання, й призначена задовольняти інформаційні потреби користувачів інформації.[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 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. Структурна схема інформаційної системи

Інформаційне забезпечення - сукупність єдиної системи класифікації й кодування повідомлень, уніфікованих систем документації, схем інформаційних потоків, що циркулюють в організації, а також методологія 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 - комплекс 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.

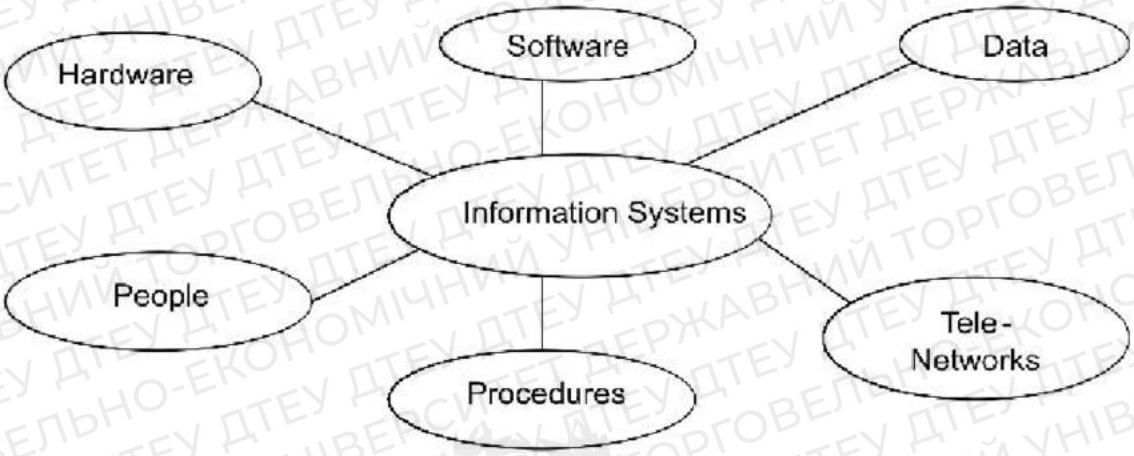


Рис.1.2. Компоненти інформаційної системи

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 обробляються для отримання інформації. Дані управляються за допомогою системи управління базами даних. Програмне забезпечення баз даних використовується для ефективного доступу до потрібних даних та для управління базами знань.
- Ресурси мереж відносяться до телекомунікаційних мереж, таких як інтранет, екстранет та Інтернет. 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 керованими 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 службових службовців ІС та методи управління.[11]

- Процедури: дозволи, що регулюють роботу інформаційної системи.

1.3 Використання електронного розкладу в навчальному процесі

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 студентові, дізнатися які заняття в нього будуть, в який час та в якій аудиторії, використовуючи будь-який пристрій із 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.

РОЗДІЛ 2.
ОГЛЯД МЕТОДИК СТРУКТУРУВАННЯ ДАНИХ ТА РОЗРОБКИ ЕЛЕКТРОННОГО РОЗКЛАДУ НАВАНТАЖЕННЯ КАФЕДРИ

2.1. Огляд засобів структурування даних для використання у системі електронного розкладу

Системи електронного розкладу широко використовуються в сучасних навчальних закладах для зручного та ефективного планування та організації навчального процесу. Одним з ключових аспектів розробки таких систем є структурування та обробка даних, що включають інформацію про розклад занять, аудиторії, викладачів та інші деталі, що необхідні для правильного функціонування системи.

Мета огляду:

Цей огляд має на меті дослідити та проаналізувати різні засоби структурування даних, які можуть бути використані в системах електронного розкладу. Будуть розглянуті різні аспекти, такі як формати даних, бази даних, мови програмування та інші засоби, що сприяють ефективному зберіганню, обробці та використанню розкладових даних.

Огляд засобів структурування даних:

1. Формати даних: Розглядаючи засоби структурування даних для систем електронного розкладу, важливо визначити оптимальний формат для збереження розкладової інформації. Розповсюджені формати, такі як JSON (JavaScript Object Notation) та XML (eXtensible Markup Language), можуть бути використані для зручного представлення даних у структурованому вигляді.

2. Бази даних: Використання баз даних є ключовим аспектом систем електронного розкладу. Реляційні бази даних, такі як MySQL, PostgreSQL або Oracle, можуть забезпечити надійне збереження та ефективне опрацювання розкладових даних. Додатково, нереляційні бази даних, наприклад MongoDB або Cassandra, можуть бути використані

для збереження невизначеної кількості атрибутів у розкладових записах.

3. Мови програмування: Вибір мови програмування для розробки системи електронного розкладу також має велике значення. Популярні мови, такі як Java, Python або PHP, надають потужні бібліотеки та фреймворки для роботи з базами даних та структурування даних у зручні структури. Додатково, мови програмування, які підтримують об'єктно-орієнтоване програмування, можуть полегшити моделювання розкладових об'єктів.

4. API та інтеграція: Забезпечення можливості взаємодії з іншими системами, такими як система управління студентськими записами або система відеоконференцій, є важливим аспектом систем електронного розкладу. Використання відкритих API або стандартів обміну даними, таких як REST або SOAP, може полегшити інтеграцію та обмін даними між різними системами.

Висновок:

Огляд засобів структурування даних для використання у системі електронного розкладу дозволяє визначити найбільш оптимальні технології та інструменти для розробки ефективної системи. Вибір форматів даних, баз даних, мов програмування та підходів до інтеграції впливає на продуктивність та функціональність системи. Огляд надає інформацію, яка допоможе розробникам та науковцям у виборі оптимальних засобів для структурування даних у системах електронного розкладу.[12]

Рис.2.1. Вигляд реляційної бази даних.

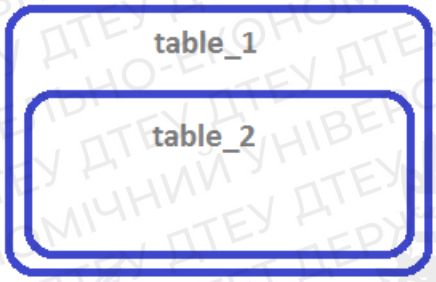


Рис.2.2. Вигляд нереляційної бази даних.

Детальніше розглянемо деякі аспекти засобів структурування даних для використання у системах електронного розкладу.

1. Формати даних:

- JSON (JavaScript Object Notation): JSON є легким та зрозумілим форматом обміну даними. Його проста структура на основі пар "ключ-значення" дозволяє зручно зберігати розкладові дані. JSON підтримує різноманітні типи даних, включаючи рядки, числа, булеві значення та масиви, що сприяє гнучкості при роботі з розкладом.

- XML (eXtensible Markup Language): XML є іншим популярним форматом для структурування даних. Він використовує теги для організації ієрархічної структури документа. XML дозволяє використовувати власні теги та атрибути, що робить його гнучким у структуруванні розкладових даних.

2. Бази даних:

- Реляційні бази даних: Реляційні бази даних (РБД) використовують таблиці зі зв'язками між ними для збереження даних. Вони надають структурований підхід до зберігання розкладових даних, де кожна таблиця може відповідати конкретному аспекту розкладу, наприклад, викладачам, групам або предметам. Реляційні бази даних також підтримують мову SQL (Structured Query Language), що дозволяє легко виконувати запити та звіти.

- Нереляційні бази даних: Нереляційні бази даних (NoSQL) набувають популярності в контексті систем електронного розкладу. Ці бази даних, такі як документні бази даних або ключ-значення, забезпечують гнучкішу модель зберігання даних, що відповідає вимогам динамічних розкладових записів. NoSQL бази даних дозволяють ефективно зберігати та опрацьовувати незначну кількість даних з розбиттям на кластери та горизонтальним масштабуванням.

3. Мови програмування:

- Java: Java є популярною мовою програмування у сфері веб-розробки. Вона пропонує розширені бібліотеки та фреймворки для роботи з базами даних, що полегшує структурування та доступ до розкладових даних.

- Python: Python є легкою вивченням та зрозумілою мовою програмування, що надає багато корисних бібліотек для роботи з даними. Python дозволяє швидко обробляти, аналізувати та структурувати розкладові дані.

- PHP: PHP є широко використовуваною мовою програмування для веб-розробки. Вона має багато фреймворків, що спрощують розробку систем електронного розкладу та роботу з базами даних.

4. API та інтеграція:

- REST (Representational State Transfer): RESTful API дозволяє створювати стандартний спосіб комунікації між системами. Використання REST дозволяє

зручно обмінюватися розкладовими даними між системою електронного розкладу та іншими додатками або сервісами.

- SOAP (Simple Object Access Protocol): SOAP є протоколом для обміну структурованими повідомленнями. Використання SOAP API дозволяє забезпечити надійну та безпечну інтеграцію систем електронного розкладу з іншими додатками або сервісами.

Загалом, вибір оптимальних засобів структурування даних для систем електронного розкладу залежить від потреб та вимог конкретного проекту. Ретельний аналіз та вивчення можливих варіантів дозволять розробникам та науковцям знайти найкращий підхід до структурування та оптимізації розкладових даних.

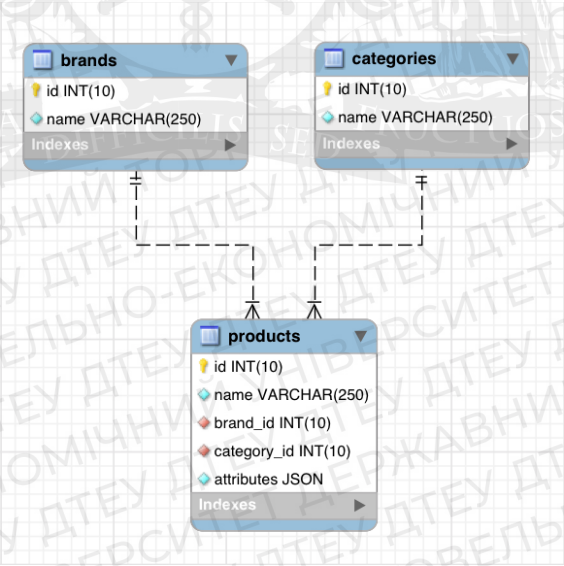


Рис.2.3. Приклад структури БД MySQL

Засоби структурування даних в системі електронного розкладу включають різноманітні технології та методи, що допомагають організувати, зберігати, обробляти та отримувати доступ до розкладової інформації. Розглянемо кілька

додаткових аспектів, які можуть бути важливими при виборі та використанні засобів структурування даних у системі електронного розкладу.

5. Формат даних:

- XML (eXtensible Markup Language): XML є розширенням текстового формату, яке дозволяє структурувати дані за допомогою тегів. XML забезпечує гнучкість та читабельність даних, що робить його популярним використанням для обміну розкладовою інформацією між системами.

- JSON (JavaScript Object Notation): JSON є легким та компактним форматом для обміну даними. Він зручний для структурування розкладової інформації, оскільки має просту синтаксичну структуру та підтримує різні типи даних.

6. Бази даних:

- Реляційні бази даних (наприклад, MySQL, PostgreSQL): Реляційні бази даних є одними з найпоширеніших та надійних засобів для зберігання структурованих даних. Вони використовують табличну модель для організації даних, що дозволяє забезпечити цілісність, надійність та швидкий доступ до розкладової інформації.

- NoSQL бази даних (наприклад, MongoDB, Cassandra): NoSQL бази даних створені для роботи з невстановленою або змінюваною структурою даних. Вони надають гнучкість та швидкий доступ до нереляційних даних, що може бути корисним для систем електронного розкладу зі змінною кількістю полів.

7. Інструменти для структурування даних:

- XSD (XML Schema Definition): XSD є мовою для опису структури XML документів. Використання XSD дозволяє визначити правила та обмеження щодо структури даних у розкладових файлах XML, що полегшує їх обробку та перевірку на коректність.

- ORM (Object-Relational Mapping): ORM є технологією, що дозволяє взаємодіяти з базою даних, використовуючи об'єктно-орієнтований підхід. ORM

дозволяє зручно створювати, зберігати та отримувати доступ до розкладової інформації, перетворюючи дані між об'єктами та таблицями бази даних.

8. Мови програмування:

- Python: Python є популярною мовою програмування для обробки даних та розробки систем електронного розкладу. Вона має велику кількість бібліотек та фреймворків для роботи з XML, JSON та базами даних.

- Java: Java також є потужною мовою програмування для розробки систем електронного розкладу. Вона має багато бібліотек для роботи з XML, JSON та базами даних, а також підтримує ORM-фреймворки, такі як Hibernate.

9. Алгоритми та структури даних:

- Графи: Графові структури даних можуть бути корисними для моделювання залежностей між різними елементами розкладу, такими як курси, викладачі та аудиторії. Алгоритми на графах дозволяють знаходити оптимальні розклади та розв'язувати різні оптимізаційні задачі.

- Дерева: Деревоподібні структури даних, такі як бінарні дерева пошуку, можуть бути використані для швидкого пошуку та організації розкладової інформації.

10. Аналіз даних:

- Методи аналізу даних, такі як статистичні методи, машинне навчання та штучні нейронні мережі, можуть бути використані для виявлення закономірностей у розкладових даних, прогнозування популярності курсів, оптимізації розкладів та багато іншого.

Ці аспекти структурування даних у системі електронного розкладу є лише деякими з можливих. Вибір конкретних засобів та методів залежить від конкретних вимог та обмежень системи, а також від потреб користувачів.

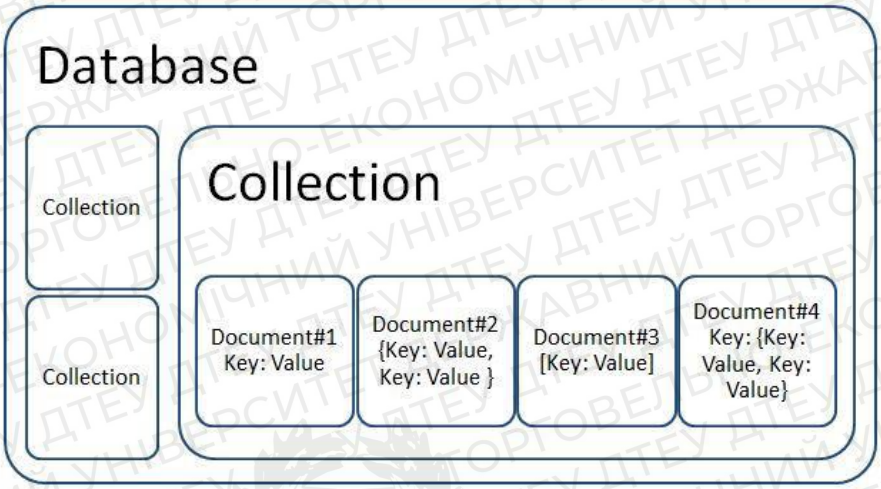


Рис.2.4. Вигляд структури БД MongoDB

11. Переваги структурування даних у системі електронного розкладу:

- Зручність та ефективність: Структурована розкладова інформація дозволяє зручно та ефективно організовувати та керувати навчальним процесом. Це сприяє швидкому доступу до потрібних даних, зменшенню помилок та полегшує внесення змін у розклад.

- Можливість автоматизації: Структуровані дані можуть бути легко оброблені та аналізовані за допомогою різноманітних алгоритмів та програмних засобів. Це відкриває можливості для автоматизації процесів, таких як генерація розкладу, розрахунок навантаження викладачів та оптимізація розкладу.

- Легкість обміну даними: Структуровані дані можуть бути легко обмінювані між різними системами та платформами. Використання стандартних форматів, таких як XML або JSON, дозволяє зручно передавати розкладову інформацію між різними програмами та системами.

- Масштабованість: Структуровані дані дають можливість ефективно масштабувати систему електронного розкладу. Засоби структурування даних, такі як бази даних та графові структури, дозволяють оптимізувати роботу з великими обсягами даних та забезпечити швидкий доступ до них.

12. Виклики та обмеження:

- Синхронізація даних: Одним з викликів є забезпечення синхронізації та консистентності даних між різними компонентами системи. Це особливо важливо у випадку, коли багато користувачів одночасно працюють з розкладовою інформацією та вносять зміни.

- Доступ до даних та конфіденційність: Розкладова інформація містить конфіденційні дані, такі як персональні дані студентів або викладачів. Тому необхідно забезпечувати відповідний рівень доступу до даних та захисту конфіденційності.

- Підтримка різноманітних потреб: Розкладова система повинна бути гнучкою та враховувати різноманітні потреби різних груп користувачів, таких як студенти, викладачі, адміністратори та батьки. Наприклад, система повинна підтримувати різні види розкладів, розкладові зміни та індивідуальні налаштування.

- Адаптація до змін: Розкладова система повинна бути готовою до змін, так як вимоги та потреби користувачів можуть змінюватися з часом. Структура даних повинна бути гнучкою та легко змінюватися без значних перешкод.

Враховуючи ці фактори, розробка та використання засобів структурування даних для системи електронного розкладу є важливим етапом у забезпеченні ефективності та якості навчального процесу. Належне планування, аналіз вимог користувачів та використання сучасних технологій можуть допомогти створити потужну та зручну систему, що задовольняє потреби всіх учасників навчального процесу.

Рис.2.5. Приклад структури файлу формату JSON

Зважаючи на продовження розкриття теми, надам додаткову інформацію про засоби структурування даних для системи електронного розкладу:

13. Технології та засоби структурування даних:

- Бази даних: Використання баз даних є поширеним підходом у системах електронного розкладу. База даних дозволяє зберігати та організовувати розкладові дані у структурованому форматі, а також забезпечувати швидкий доступ до них. Реляційні бази даних, такі як MySQL або PostgreSQL, можуть використовуватися для зберігання інформації про розклад, викладачів, групи, аудиторії та інше.

- XML та JSON: Формати XML (Extensible Markup Language) та JSON (JavaScript Object Notation) широко використовуються для структурування та обміну даними в системах електронного розкладу. Вони дозволяють представляти розкладову інформацію у вигляді структурованих документів, що полегшує її обробку та передачу між різними компонентами системи.

- Графові бази даних: Графові бази даних, наприклад, Neo4j, можуть бути використані для моделювання складних зв'язків між різними елементами розкладу, такими як взаємозв'язки між групами, предметами, викладачами та іншими

факторами. Графові бази даних забезпечують ефективне виконання складних запитів та аналітичних операцій.

- RESTful API: REST (Representational State Transfer) є архітектурним стилем для розробки веб-сервісів, що забезпечує структурування та обмін даними. За допомогою RESTful API можна створити інтерфейс, що дозволяє іншим програмам читати, записувати та оновлювати розкладову інформацію.

- Алгоритми та структури даних: Для оптимізації роботи системи електронного розкладу можуть бути використані різні алгоритми та структури даних. Наприклад, алгоритми для пошуку та сортування даних можуть використовуватися для швидкого доступу до розкладової інформації. Структури даних, такі як дерева або хеш-таблиці, можуть використовуватися для ефективного пошуку та організації даних.

14. Інтеграція з іншими системами: Система електронного розкладу може потребувати інтеграції з іншими системами, такими як система управління навчальними курсами (LMS), система електронного навчання (e-learning), система ведення обліку студентів та інші. Інтеграція забезпечує обмін даними та координацію між різними системами, що полегшує управління навчальним процесом.

15. Визначення структури даних: При розробці системи електронного розкладу важливо визначити оптимальну структуру даних, яка відповідає вимогам користувачів та дозволяє ефективно зберігати та обробляти розкладову інформацію. Планування структури даних включає визначення таблиць, полів, зв'язків між ними та правил валідації даних.

Враховуючи ці аспекти, розробка системи електронного розкладу потребує ретельного аналізу, проектування та використання засобів структурування даних.

Ефективне використання цих засобів дозволяє забезпечити гнучкість, швидкість та надійність системи, а також задовольнити потреби різних користувачів.

.[17]

2.2. Методи табличного верстання інтернет-сайтів

Методи табличного верстання інтернет-сайтів використовуються для створення розміщення та організації вмісту на веб-сторінках за допомогою HTML таблиць. Цей підхід широко застосовувався у минулому, але зараз вважається застарілим, оскільки були розроблені більш сучасні техніки верстання. Однак, детальніше розглянемо основні методи табличного верстання:

1. Використання HTML таблиць: Основна ідея полягає у використанні HTML тегів `<table>`, `<tr>` і `<td>` для створення таблиць. Тег `<table>` визначає саму таблицю, `<tr>` - рядок таблиці, а `<td>` - комірку в рядку. Ці теги дозволяють організувати вміст (текст, зображення, посилання тощо) у вигляді таблиці з різними рядками та комірками.

2. Розміщення елементів у комірках: У табличному верстанні елементи розміщуються у комірках таблиці за допомогою додаткових тегів або стилів CSS. Наприклад, можна використовувати теги `<div>` або `<span>` для обгортання контенту в комірці та застосовувати CSS стилі для вирівнювання, кольору, шрифту тощо.

3. Створення колонок і рядків: За допомогою таблиць можна легко створювати колонки та рядки. Для створення нового рядка використовується тег `<tr>`, а для створення нової комірки в рядку - тег `<td>`. Визначаючи кількість рядків і комірок, можна організувати таблицю з бажаною структурою.

4. Злиття комірок і об'єднання колонок: Таблиці також дозволяють злиття комірок та об'єднання колонок. Це корисна функція для створення більш складних макетів і структур. За допомогою атрибутів `rowspan` та `colspan` можна задавати кількість злитих комірок у вертикальному та горизонтальному напрямках відповідно.

5. Відформатування таблиць за допомогою CSS: Щоб надати таблицям більш привабливий вигляд, можна застосовувати стилі CSS. CSS дозволяє змінювати розміри комірок, кольори, шрифти, рамки, вирівнювання тексту тощо. Використання CSS дозволяє розділити стиль і вміст таблиці, що полегшує її редагування та оновлення.

Варто відзначити, що метод табличного верстання має деякі обмеження та недоліки. Зокрема, він може бути складним для створення адаптивного дизайну, оскільки таблиці мають фіксовані розміри. Крім того, використання таблиць для верстання макету може призводити до неправильної інтерпретації пошуковими системами та проблем з доступністю.

Сучасні підходи до верстання інтернет-сайтів, такі як використання CSS-сіток, флексбоксів та CSS-грідів, надають більш гнучкі і ефективні засоби для організації та розміщення вмісту. Вони дозволяють створювати адаптивні та мобільні дизайни, забезпечуючи кращу підтримку різних пристроїв та кращу оптимізацію для пошукових систем.

.[18]

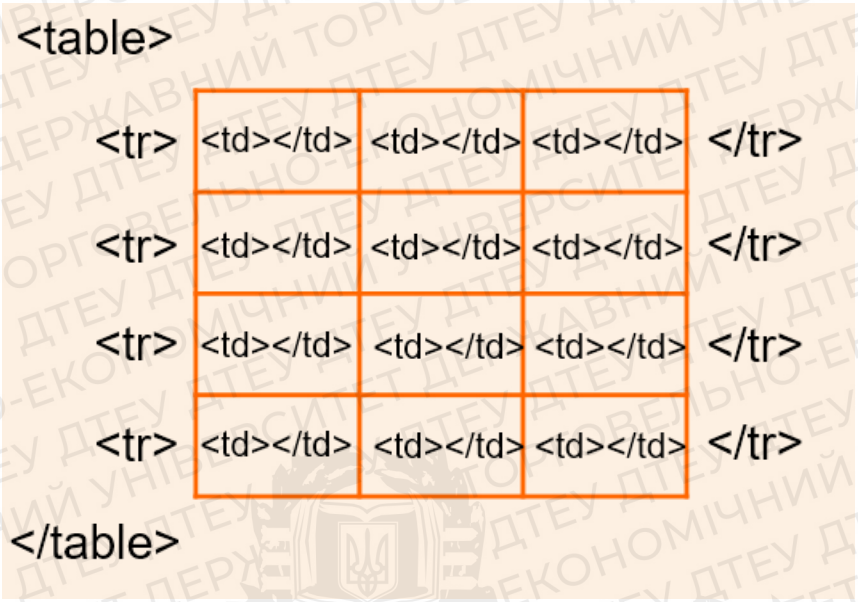


Рис.2.6. Приклад вигляду таблиці

2.3. Використання методу float для створення Web-додатку

Властивість float в CSS, дозволяє розробникові використовувати подібні таблиці стовпчики в розмітку HTML без використання таблиці.

Мета властивості float - притиснути елемент до лівого або правого краю батьківського елементу, а найближчий текст обходить елемент з інших сторін. Така поведінка тексту нагадує потік води, що оминає камінь, тому елементи з поплавком називають плаваючими або поплавками. Ця концепція аналогічна тій, яка представляється кожен день у друкованій літературі, де є фотографії та інші графічні елементи, вирівняні за будь-якою стороною, інформаційне наповнення (зазвичай текст), знаходиться навколо елементу, вирівняного навколо лівого або правого краю.[19]

Властивість Float може використовувати одне з 4 значень: вліво (left), вправо (right), без вирівнювання(none) та успадковане (inherit):

- Значення left у властивості float вирівнює елемент по лівому краю браузера, а всі інші елементи обтікають її по правій стороні.
- Значення right у властивості float вирівнює елемент по правому краю браузера, а всі інші елементи обтікають її по лівій стороні.

- Значення none у властивості float вказує на те, що обтікання елеенту не задається.
- Значення inherit у властивості float вказує на спадковість властивості від свого батьківського елемента.

2.4. Модульна система верстки у веб-дизайні

Модульна система верстання веб-сторінок – це система верстання, при якій основою композиції смуг і шпальт стає модульна сітка з певним кроком (модулем), однаковим або різним по горизонталі і вертикалі. Модульна система спрощує і прискорює художнє конструювання і створює сприятливі умови для автоматизації верстки при використанні комп'ютерних настільно-видавничих систем.[20]

Веб-дизайнери відносно недавно почали застосовувати модульні сітки в проектуванні інтерфейсів веб-сайтів. Цей метод значно спрощує як проектування інтерфейсу, так і подальшу верстку макета. На відміну від друкарні, модулі в веб-дизайні можуть мати непостійну ширину і розтягуватися в залежності від ширини вікна браузера (дозволу екрану монітора).

Начастіше як шаблон сітки використовується CSS-фреймворк Bootstrap[21]. В сітці цього фреймворку 12 колонок, при кастомізації цього фреймворку можно виставити будь-яке інше значення, але користувача влаштовує 12 колонок. Вся сітка розміщується в загальному контейнері, з класом container, що має фіксований розмір 1170px. Це контейнер завжди розтягується на 100% ширини вікна, тобто сайт буде гнучним.

Всередині контейнера є блок із класом row, тобто 1 рядок сітки. В самому рядку розміщаються колонки. Колонка має клас col-x-x, де перший x – визначення пристрою, а другий – кількість колонок от 1 до 12. Для найменших пристроїв, у яких ширина екрану менше 768 пікселів, префікс класу - col-xs- або якщо відкинути col, просто xs. Далі йде sm (small-devices, ширина від 768 до 991 пікселів), md (medium-devices, ширина від 992 до 1199 пікселів) і large-devices, з товщиною не менше 1200 пікселів.

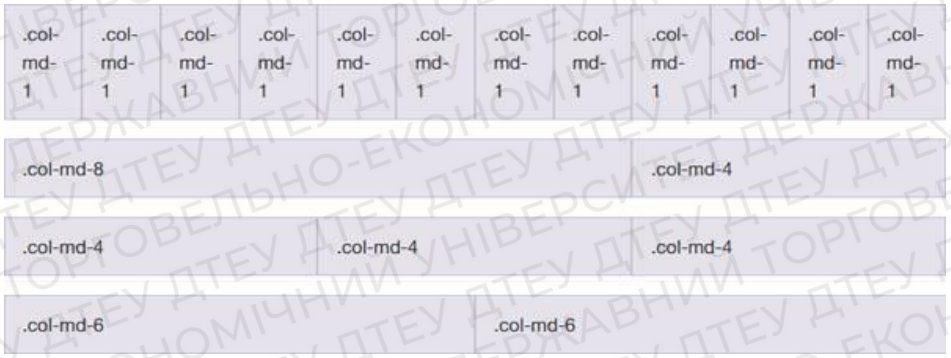


Рис.2.7. Приклад сітки фреймворку Bootstrap 4

2.5. Використання чат-ботів у месенджері Telegram для електронного розкладу.

Чат-бот - це інтерактивна програма, яка виконує прості та повторювані автоматизовані завдання через Інтернет.[22] Чат-боти можуть працювати на основі запрограмованих сценаріїв з множинним вибором, або можуть використовуватись, як цифрові помічники, що використовують машинне навчання для виявлення моделей спілкування. Завдяки постійній взаємодії з людьми вони вчаться наслідувати реальним розмов і реагують на усні або письмові запити, допомагаючи знайти відповіді. Оскільки чат-боти використовують штучний інтелект, то розуміють мову, а не просто команди. Таким чином, після кожного діалогу вони стають розумнішими.

Боти можуть виконувати практично будь-які завдання, які може робити кожен користувач акаунту Telegram з онлайн-сервісами. По суті, боти – це такий собі зручний для людини інтерфейс роботи з різноманітними веб-службами.

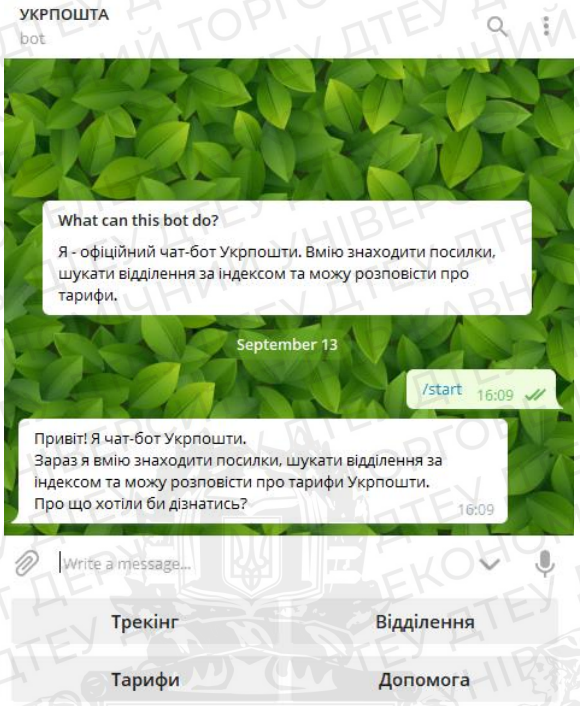


Рис.2.8. Приклад телеграм-боту

Переваги використання чат-ботів:

- ботам не потрібно відпочивати, вони працюють цілодобово і з машинної швидкістю, обробляючи тисячі запитів в хвилину;
- роботи не втомлюються і не зляться, вони не знають упереджень і поганих днів;
- з усіма користувачами боти спілкуються однаково запрограмуваним тоном;
- спрощена комунікація: користувачеві не варто відвідувати сайт;
- оперативність: користувач швидко отримує відповідь;
- Дешевизна в експлуатації.

Недоліками використання чат-ботів є:

- чат-бот відповідає із обмеженої бази даних, тобто користувач може не знайти правильної відповіді на те питання, що він шукає;
- у недосвідченого користувача є проблеми із правильним створенням чат-боту;
- На відміну від сайту, потребують постійного контролю, налаштування та оптимізації з метою розширення бази відповідей і знань.[23]

РОЗДІЛ 3.
СТВОРЕННЯ ЕЛЕКТРОННОГО РОЗКЛАДУ НАВАНТАЖЕННЯ КАФЕДРИ

3.1. Розробка алгоритму створення електронного розкладу студента навантаження кафедри

Розділимо весь процес розробки на основні етапи:

- Визначення задач веб-додатку;
- Створення бази розкладів;
- Верстка сторінки веб-додатку;
- Створення необхідного функціоналу;
- Аналіз веб-додатку.

На етапі «визначення задач веб-додатку» було визначено, що задачею веб-додатку є оперативне інформування студента про розклад та його можливі зміни впродовж тижня.

На етапі «створення бази розкладів» було сформовано необхідну інформаційну базу із наявних розкладів занять груп університету за допомогою Microsoft Excel[24] та Visual Studio Code[25].

Під час верстки сторінки веб-додатку були виконані наступні дії:

1. Створено файли index.html для написання структури сайту і styles.css для задання стилів;
2. Виконана розмітка сайту на зручні для користувача блоки;
3. Створені стилі для заданих блоків;

Після верстки сторінки були підключені скрипти обробки результатів форми та інтерпретації даних із бази розкладів в зручний для сприйняття вигляд.

Останнім етапом створення сайту є проведення його аналізу. Під час цього етапу проведено декілька видів тестування, що обов'язкові для кожного сайту:

1. Аналіз зручності користування сайтом. У ході такого тестування визначається якість виконання і зручність інтерфейсу сайту, а так само проводяться роботи по виявленню можливих помилок у структурі.

2. Аналіз на стійкість до великих навантажень. Цей аналіз імітує одночасне відвідування великої кількості користувачів для визначення працездатності ресурсу при великих навантаженнях. У ході тестування перевіряється не стільки сам ресурс, скільки комплексна робота апаратної частини сервера, веб - сервера, програмного ядра та інших компонентів сайту.
3. Аналіз HTML коду. Перевіряється весь сайт на наявність помилок у програмному коді та відповідність стандартам.

3.2. Моделювання процесу розробки електронного розкладу студента.

Для проведення моделювання було обрано дві UML діаграми: послідовності і прецедентів(Use Case).

Діаграма послідовностей – це графічне зображення послідовної взаємодії об’єктів системи в часі, тобто відображає часові особливості передачі і прийому повідомлень об’єктами. Завдяки використанню діаграми послідовності можливо з’ясувати склад компонентів взаємодії та описати потік повідомлень від одного компоненту до іншого.[26]

На діаграмі відображені наступні об’єкти: студент, сайт, форма, обробка даних, база даних, розклад і адміністратор. Об’єкти «студент» і «адміністратор» є також і акторами.

Актор «студент» заходить на сайт та заповнює форму. Сайт передає дані форми на сервер. Сервер опрацьовує дані та звертається до бази даних, знаходить в ній необхідні дані та відображає їх у об’єкті «Розклад».

Сайт – об’єкт, що призначений для надання інформації про розклад занять в університеті на поточний чи наступний тиждень. Тісно пов’язаний із об’єктами «форма» і «оброблювач даних».

Форма – об’єкт, що призначений для обміну даними між клієнтом та обробником даних. Після заповнення передає дані в спеціальний обробник.

Оброблювач даних витягує дані запиту з масиву байт, отриманого на етапі читання даних. Після інтерпретації запиту звертається до бази даних за необхідним розкладом та відображає його на сторінці сайту.

Об’єкт «база даних» містить в собі структуровані дані про розклад занять всіх груп в університеті.

Об’єкт «розклад» призначений для відображення даних про розклад групи після їх інтерпретації оброблювачем даних.

Актор «адміністратор» займається адмініструванням сайт, бази даних та контролюванням обробника даних.

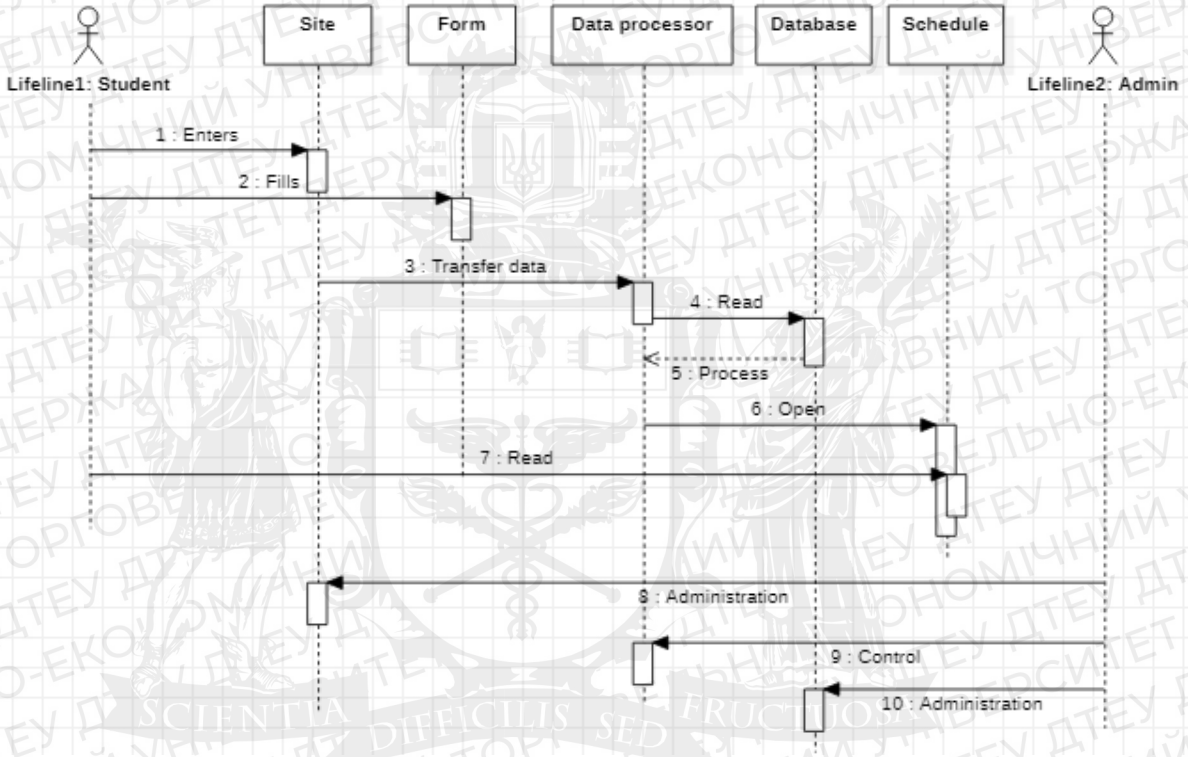


Рис.3.1. Діаграма послідовності виконання процедур веб-додатку

В UML діаграма прецедентів або Use Case діаграма або діаграма варіантів використання використовується для представлення динамічної поведінки системи. На діаграмі зображуються завдання, послуги та функції, необхідні системі/підсистемі програми. Крім того, зображуються функціональні можливості системи високого рівня, а також описується взаємодія користувача із системою.[27]

На діаграмі представлено два актори:

Основним актором на діаграмі є студент. Він виконує 4 основні дії:

1. Вхід на сайт;
2. Заповнення форми запиту;
3. Натиснення кнопки «Підтвердити»;
4. Перегляд інформації про розклад.

Вторинним актором є адміністратор. Він виконує дві дії:

1. Адміністрування бази даних;
2. Адміністрування сайту.

Генеральний сценарій успіху:

1. У студента з’являється необхідність дізнатися розклад занять на поточний день.
2. Студент заходить на сайт.
3. Заповнює форму для запиту на сайті.
4. Натискає кнопку “Підтвердити”.
5. На сайті відбувається обробка запиту із подальшим зверненням до бази даних із розкладами.
6. У таблиці на сторінці відображається розклад на поточний чи наступний тиждень.

Можливі розширення виконуваних дій:

- Корегування заповненої форми;
- Зміна заповнених даних у формі;
- Повторний запит;

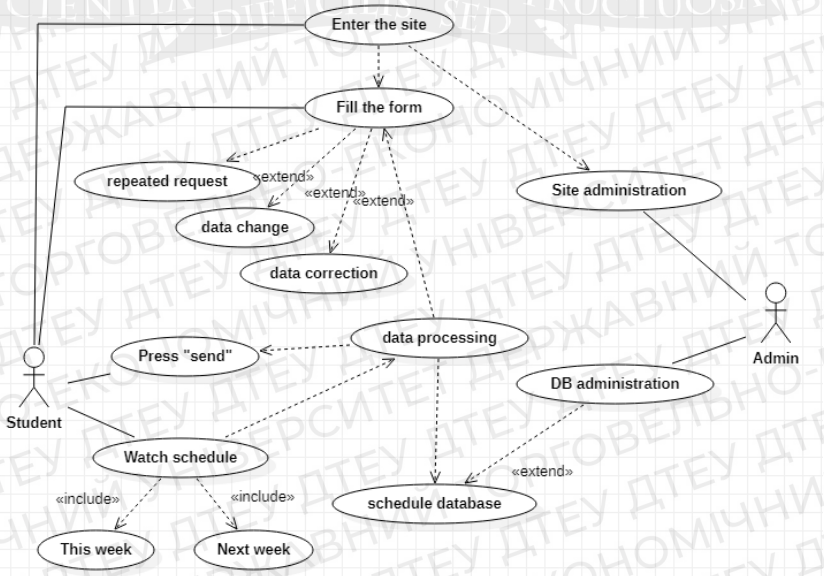


Рис.3.2. Діаграма використання веб-додатку

3.3. Програмна реалізація інформаційної системи електронного розкладу

Створення веб-додатку розподіляється на декілька етапів:

1. Наповнення JSON файлів необхідними даними;
2. Створення стилю та дизайну сайту із використанням зв'язки HTML і CSS;
3. Створення коду обробки інформації та запитів до JSON файлів із використанням бібліотеки JQuery мови JavaScript.

Для використання JSON-файлів для відображення розкладу занять, їх необхідно наповнити необхідними даними. В якості значень виступає набір впорядкованих пар ключ: значення, де ключем є рядок, а значенням будь-який тип даних. В нашому випадку ключами виступають рядки із наступними назвами:

- Ідентифікатор файлу("id");
- День тижня("day");
- Номер пари("pair");
- Початок пари("start");
- Кінець пари("end");
- Предмет("subject");
- Аудиторія("room");
- Група("group");
- Викладач("teacher").

Структура файлу виглядає наступним чином:

```
{
    "id":fak4_kurs4_9,
    "faculty":"ФІТ",
    "course":"4",
    "group": "9",
    "schedule": [
        {
            "day": "понеділок",
            "pair": "3",
            "start": "12:05",
            "end": "13:25",
            "subject": "Л Моделювання та аналіз ПрЗаб  В",
            "room": "Б-505",
            "group": "вся",
            "teacher": "доц. Рзаєва С.Л ."
        },
```

Рис.3.3. Шаблон структури файлу розкладу групи.

Для створення макету сайту використовують мову розмітки гіпертексту HTML і CSS(Каскадні таблиці стилів).

Мова гіпертекстової розмітки (HyperText Markup Language — HTML) використовується для створення і візуального відображення веб-сторінок. Основу кожної веб-сторінки в мережі Інтернет становить HTML. Під терміном «гіпертекст» мається на увазі текст, сформований за допомогою мови розмітки і який містить гіперпосилання, якими зв'язані веб-сторінки одна з одною, роблячи Всесвітню павутину тим, чим вона є сьогодні. HTML підтримує як візуальні зображення, так і інший медіаконтент. HTML є мовою, що описує структуру і семантику вмісту веб-документу. Контент веб-сторінки розмічений за допомогою тегів, що представляють HTML-елементи. Прикладами таких елементів є теги <head>, <title>, <body>, <article>, <section>, <p>, , , <img>, <picture>, і т.д. Ці елементи формують будівельні блоки для будь-якого веб-сайта.[28]

Cascading Stylesheets (CSS, каскадні таблиці стилів) — це код, що використовується для стилізації сайту. CSS не є справжньою мовою програмування. Це лише мова таблиць стилів, яка дозволяє задавати стилі обраним елементам у HTML документах.[29] Відокремлюючи стиль подання документів від вмісту документів, CSS спрощує створення веб-сторінок і обслуговування сайтів. CSS підтримує таблиці стилів для конкретних носіїв, тому автори можуть адаптувати подання своїх документів до візуальних браузерам, слуховим пристроїв, принтерів, брайльовським пристроїв, кишеньковим пристроям і т.д. Каскадні таблиці стилів описують правила форматування елементів за допомогою властивостей і допустимих значень цих властивостей. Для кожного елемента можна використовувати обмежений набір властивостей, інші властивості не будуть чинити на нього ніякого впливу.

jQuery – безкоштовний, швидкий, невеликий та багатофункціональний JavaScript фреймворк, ціль якого – полегшити веб-розробнику процес створення інтерактивних веб-сайтів та користувальницьких інтерфейсів завдяки мінімально можливій кількості рядків. Завдяки простому у користуванні API, який працює у

безлічі браузерів, такі речі, як обробка та маніпулювання документами HTML, обробка подій, анімація та Ajax, набагато простіші.[30] Оскільки бібліотека jQuery – це JavaScript, вона може бути вбудована у будь-який проект, в якому може використовуватися JavaScript.

Бібліотека jQuery містить функціональність, корисну для максимально широкого кола завдань. Тим не менш, розробниками бібліотеки не ставилося завдання суміщення в jQuery функцій, які підійшли б усюди, оскільки це призвело б до великого кода, більша частина якого не затребувана. Тому була реалізована архітектура компактного універсального ядра бібліотеки і плагінів. Це дозволяє зібрати ту JavaScript-функціональність, яка була б затребувана в проекті.

Створення дизайну сайту починається зі створення структури шаблону сайту.

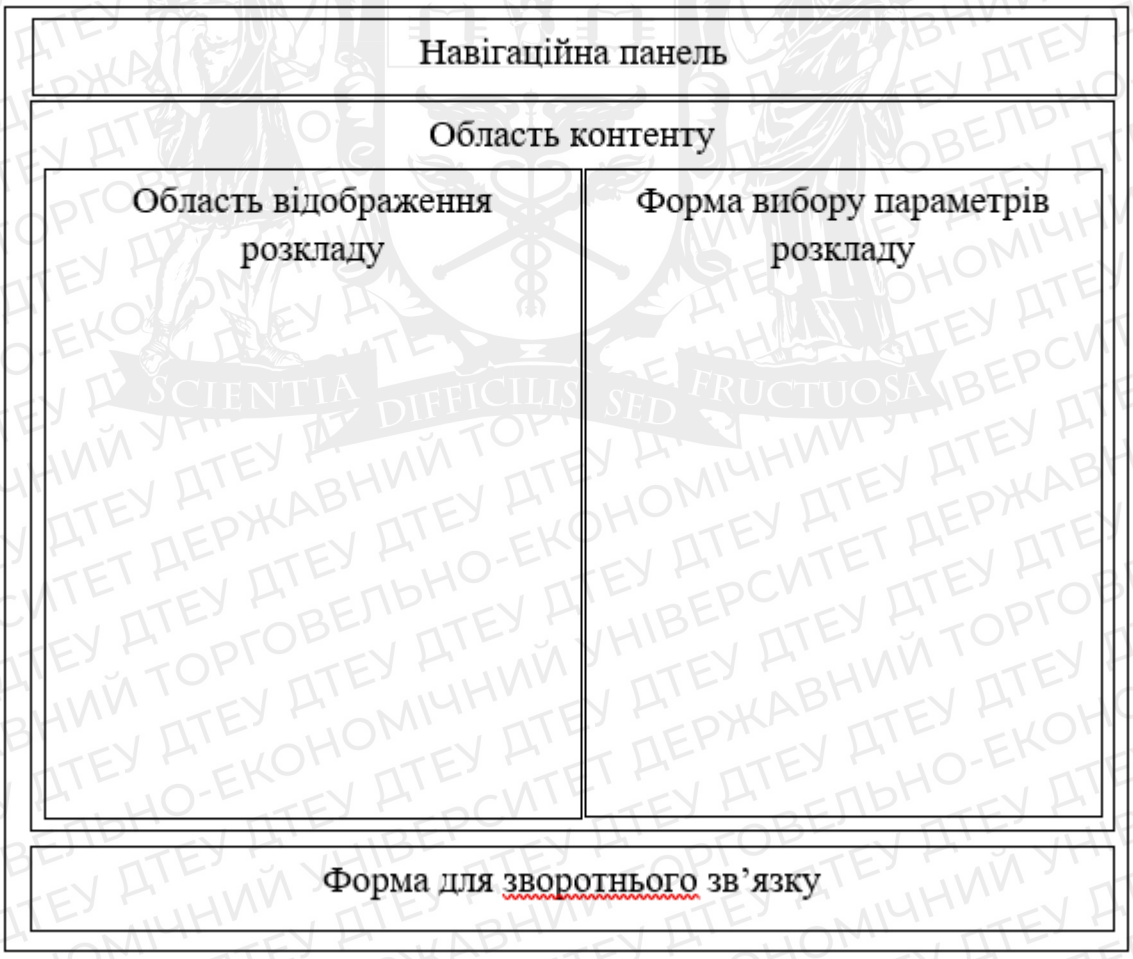


Рис.3.4. Шаблон сторінки веб-додатку

Після створення шаблону сайту можна переходити до створення його компонентів. Для макету був створений файл стилю style.css, який підключається

до головної сторінки index.html. Файл style.css містить глобальні стилі. Повний код файлу наведено у додатках.

Навігаційна панель призначена для можливості переходу студента по основним посиланням університету. На навігаційній панелі розміщені посилання на сайт університету, сайт дистанційного навчання та офіційні сторінки університету в соціальних мережах.

```
.contact{
    width: auto;
    height: 130px;
    display: grid;
    grid-template-columns: repeat(4, 1fr);
    grid-template-rows: repeat(2, 65px);
    align-items:center;
    justify-items:center;
    background: rgb(234, 240, 226);
    color: black;
}
.contact a{
    display: block;
    text-align: center;
    height: 100%;
    color: #666;
}
```

Рис.3.5. Код стилів навігаційної панелі.

Атрибут width встановлює висоту вмісту елементу. У даному випадку значення встановлене як auto, тобто ширина об’єкту встановлюється в залежності від типу та вмісту об’єкту. Атрибут height встановлює висоту вмісту елементу. За замовчуванням висота визначається автоматично, виходячи із вмісту елементу, але якщо задати фіксоване значення висоти, то вона буде встановлена, незважаючи на об’єм вмісту. Якщо вміст елементу більше вказаного значення висоти, то висота елементу залишиться незмінною, а вміст буде відображатись поверх цього елементу.

Властивість display визначає елемент як контейнер і встановлює новий контекст форматування сітки для його вмісту. Значення grid формує сітку як блок. Властивості grid-template-columns і grid-tempate-rows визначають колонки і рядки сітки за допомогою списку значень розділеного пробілами. Значення являють собою розмір треку, а прогалини між ними представляють лінії сітки.

Після застосування стилів, панель навігації виглядає наступним чином:

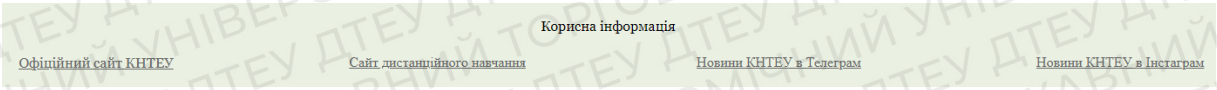


Рис.3.6. Меню корисних посилань

Область контенту буде містити в собі основне наповнення сайту. Для зручності використання вона розбивається на блоки: правий блок міститиме форму, що використовуватиметься для задання параметрів пошуку необхідного розкладу; лівий – область, в якій буде відображуватись результат пошуку необхідного розкладу.

```
.main{
    width: 100%;
    height:500px;
    display: grid;
    grid-template-columns: repeat(2, 1fr);
    grid-template-rows: repeat(1);
    background: rgb(234, 240, 226);
    color: black;
}
.main div:nth-child(1){
    grid-column:2;
    grid-row:1;
    font-family: Georgia, Times, serif;
    align-items:center;
    justify-items:center;
    justify-content: center;
}
.main div:nth-child(2){
    grid-column:1;
    grid-row:1;
    font-family: Georgia, Times, serif;
    align-items:center;
    justify-items:center;
    justify-content: center;
}
```

Рис.3.7. Код налаштування стилю поділу сторінки на дві частини

Псевдоклас :nth-child знаходить один чи більше елементів, базуючись на їх позиції серед групи сусідніх елементів. У даному випадку, псевдоклас шукає елементи типу div, що знаходяться всередині елементу із назвою main.

Властивість grid-row визначає з якого рядка в макеті сітки буде починатися елемент, скільки рядків буде займати елемент, або на якому рядку завершиться елемент в макеті сітки. Властивість grid-column вказує розмір і розташування елементу в рядку сітки, визначаючи місце для її розташування.

Властивість font-family визначає пріоритетний список з однієї або кількох назв сімейства шрифтів і / або загальну назву шрифту для обраного елемента.

Форма для вибору параметрів розкладу передбачає собою розміщення на ній трьох полів типу select, двох полів типу radio та кнопки підтвердження надсилання

форми. Поля типу select призначені для вибору необхідних параметрів пошуку розкладу студента, поля типу radio призначені для вибору відображення розкладу на поточний чи наступний тиждень. Поле типу submit призначене для підтвердження відправлення форми для отримання результатів пошуку необхідного розкладу. Створення форми представляє написання наступних стилів для випадаючих списків та кнопки «Підтвердити».

```
#myfakultet, #mykurs, #mygrupa {
    grid-column:2;
    display: block;
    font-size: 16px;
    font-family: sans-serif;
    font-weight: 700;
    color: #444;
    line-height: 1.3;
    padding: .6em 1.4em .5em .8em;
    width: 75%;
    height: 40px;
    max-width: 100%;
    box-sizing: border-box;
    margin: 0;
    border: 1px solid #aaa;
    box-shadow: 0 1px 0 1px rgba(0,0,0,.04);
    border-radius: .5em;
    -moz-appearance: none;
    -webkit-appearance: none;
    appearance: none;
    background-color: #fff;
    background-repeat: no-repeat, repeat;
    background-position: right .7em top 50%, 0 0;
    background-size: .65em auto, 100%;
}
```

Рис.3.8. Стилі полів типу select

Властивість font-size встановлює розмір шрифту елемента. В даному випадку розмір шрифта заданий як 14px. Властивість font-weight встановлює насиченість шрифту. Значення встановлюється від 100 до 900 з кроком 100. Значення 700 встановлює шрифт напівжирним.

Властивість -moz-appearance використовується в браузерах, що базуються на двигуні Firefox для відображення елемента, використовуючи базові стилі платформи на основі теми операційної системи. Властивість -webkit-appearance використовується в браузерах Safari, Chrome, Opera для того ж ефекту.

Універсальна властивість border дозволяє одночасно встановити товщину, стиль і колір кордону навколо елемента. В даному випадку товщина кордону дорівнює 1px, стиль границі – цільна, колір закодований шістнадцятковим значенням. Border-radius задає радіус заокруглення кутів рамки.

Властивість box-shadow додає тінь до елементу, значення пікселів задають значення зміщення тіні від елементу. Властивість text-shadow додає тінь до тексту,

а також встановлює її параметри: колір тіні, зміщення щодо напису і радіус розмиття.

Властивість border-box застосовується для зміни алгоритму розрахунку ширини і висоти елемента. Властивість box-sizing змінює алгоритм, щоб властивості width і height задавали розміри не контента, а розміри блоку.

```
#submit_form{
    height:60px;
    width: 150px;
    background: #B9DFFF;
    color: #fff;
    border: 1px solid #eee;
    border-radius: 20px;
    box-shadow: 5px 5px 5px #eee;
    text-shadow:none;
    outline:none;
}
#submit_form:hover {
    background: #016ABC;
    color: #fff;
    border: 1px solid #eee;
    border-radius: 20px;
    box-shadow: 5px 5px 5px #eee;
    text-shadow:none;
}
```

Рис.3.9. Стилі кнопки «Підтвердити»

Псевдоклас :hover визначає зміни із об’єктом, при наведенні курсора мишки на нього.

Властивість background встановлює колір фону для кнопки, значення кольору встановлене шістнадцяковим значенням. Властивість color встановлює колір тексту в об’єкті.

Властивість outline одночасно встановлює колір, стиль і товщину зовнішнього кордону на всіх чотирьох сторонах елементу. Значення none визначає, що жодних границь не буде створено.

Після застосування створених стилів форма вибору параметрів розкладу виглядатиме наступним чином:

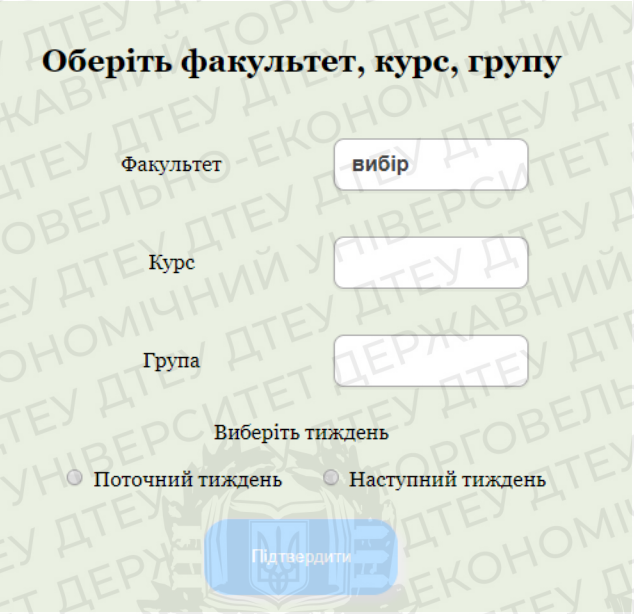


Рис.3.10. Форма вибору параметрів пошуку розкладу занять

Підвал сайту використовується для надання зворотнього зв'язку з метою пропозиції покращень користувачами адміністратору сайту. У підвалі сайту розміщені 2 поля типу text, поле типу textarea і поле типу submit. Поля типу text призначені для введення імені та електронної пошти користувача, у полі textarea користувач може ввести свою пропозицію покращення функціоналу сайту. Поле типу submit призначене для підтвердження відправлення форми на пошту адміністратора сайту.

```
.down
{
    height:150px;
    width:100%;
    background-color:rgb(63, 58, 58);
    display: grid;
    grid-template-columns: repeat(9, 1fr);
    grid-template-rows: repeat(2, 75px);
    justify-self:center;
    align-self:end;
    align-items:center;
    justify-items:center;
    justify-content: center;
}
.down div:nth-child(1)
{
    grid-row: 1;
    grid-column-start: 4;
    grid-column-end: 7;
}
.down div:nth-child(2)
{
    grid-row: 2;
    grid-column: 2/9;
}
```

Рис.3.11. Стилі побудови підвалу сайту

Властивість justify-self виконує вирівнювання елемента макету сітки всередині комірки по осі рядка grid-контейнера. Значення center вирівнює елемент по середині рядка в комірці.

Властивість justify-content визначає, як браузер розподіляє простір між та навколо елементів контенту уздовж головної осі їх контейнера. Значення center вирівнює елементи по центру блоку.

Властивість grid-column-start визначає стовпчик у сітці, де буде починатися розташування вибраного елементу і скільки треків він охоплює. Властивість grid-column-end визначає стовпчик у сітці, де буде закінчуватися розташування вибраного елементу і скільки треків він охоплює.

Після задання необхідних стилів підвал сайту виглядає наступним чином:

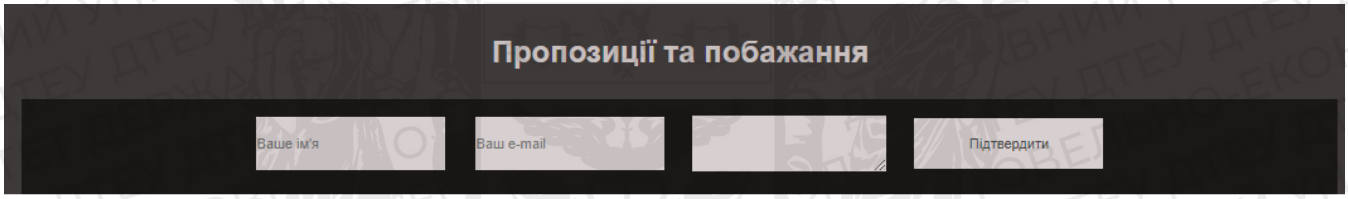


Рис.3.12. Форма зворотнього зв’язку

Для налаштування логіки роботи веб-додатку була використана бібліотека JQuery, а саме – її технологія AJAX, що дозволяє використати об'єкт XMLHttpRequest для спілкування з серверами. Він може надсилати та отримувати інформацію в різних форматах, включаючи JSON, XML, HTML та текстові файли. Найбільш привабливою характеристикою AJAX є її «асинхронність» характеру, а це означає, що він може спілкуватися з сервером, обмінюватися даними та оновлювати сторінку, не потребуючи оновлення сторінки.

Веб-додаток використовує бібліотеку jquery-3.4.1 та підключається на сторінку веб-додатку наступним кодом: <script src="https://code.jquery.com/jquery-3.4.1.min.js" integrity="sha256-CSXorXvZcTkaix6Yvo6HppcZGetbYMGWSFlBw8HfCJo=" crossorigin="anonymous"></script>.

Після підключення бібліотеки JQuery створюється код обробки інформації. Код обробки інформації наведено в додатках. Одним із головних завдань при

створенні веб-додатку було управління завантаженням вмісту файлу типу JSON на сторінку та його коректне відображення.

Щоб отримати JSON, було використане API XMLHttpRequest. Цей об'єкт JavaScript дозволяє виконувати мережеві запити для отримання ресурсів з сервера через JavaScript (наприклад, зображення, текст, JSON, навіть фрагменти HTML), що означає, що ми можемо оновлювати невеликі розділи контенту без необхідності перезавантаження всієї сторінки.

Для запису адреси файлу типу JSON, була створена функція readTextFile, що звертається до файлу за адресою. Після підключення файлу із даними про розклад, необхідно перетворити дані, що наявні в цьому файлі, в об’єкт для подальшого використання. Для цього використовується метод JSON.parse(). Метод бере JSON рядок та трансформує його в об’єкт, далі об’єкт записується в змінну. Після запису об’єкту в змінну, вона передається в наступну функцію getTableInfo. Код приведений нижче:

```
readTextFile(`rozklad/${grouppaid}.txt`, function(text)
{
    var data = JSON.parse(text);
    console.log(data);
    getTableInfo(data);
});
```

Рис.3.13. Код із перетворенням вмісту файлу

Для зручного сприйняття даних на сторінці, об’єкт необхідно записати у таблицю. Для запису даних був використаний метод document.getElementById. Цей метод повертає елемент, що має атрибут ID із необхідним значенням.

Наведений код звертається до елементу із ID schedule і створює в ньому рядок таблиці, що містить в комірках значення заголовків.

Метод appendChild() використовується для створення текстового вузла як останнього дочірнього вузла. Цей метод також використовується для переміщення елемента з одного елемента в інший. Він використовується для створення нового елемента з деяким текстом, потім спочатку створити текст як текстовий вузол, а потім додати його до елемента, а потім додати елемент до документа.

```
var scheduleTable = document.getElementById('schedule');
let headerTr = document.createElement('tr');
headerTr.innerHTML =
    `<th>День</th><th>Пара</th><th>Старт пари</th>
    <th>Кінець пара</th><th>Предмет</th><th>Кабінет</th>
    <th>Група</th><th>Викладач</th>`;
scheduleTable.appendChild(headerTr);
```

Рис.3.14. Код із генерацією побудови заголовку таблиці

Для заповнення таблиці був використаний метод forEach. Метод forEach() виконує надану функцію один раз для кожного елемента масиву. Дана функція створює елемент з тегом <tr>, що слугує контейнером для створення рядку таблиці. В цей рядок записуються елементи масиву як комірки рядка. Ця операція повторюється до тих пір, поки не закінчаться всі елементи масиву масивів.

```
data.firstweek.forEach(function(elem)
{
    let tr = document.createElement('tr');
    tr.innerHTML =
        `<td>${elem.day}</td><td>${elem.pair}</td>
        <td>${elem.start}</td><td>${elem.end}</td>
        <td>${elem.subject}</td><td>${elem.room}</td>
        <td>${elem.group}</td><td>${elem.teacher}</td>`;
    schedule.appendChild(tr);
});
```

Рис.3.15. Код генерації та заповнення таблиці

В наведеному коді створюється рядок для кожного елементу об’єкту із вкладеними масивами, і ця операція продовжується, поки не закінчаться масиви в об’єкті.

День	Пара	Старт пари	Кінець пара	Предмет	Кабінет	Група	Викладач
понеділок	1	8:20	9:40	Лб2 Штучний інтелект	Б-523	підгрупа	ст.в. Селіванова А.В .
понеділок	1	8:20	9:40	Лб1 Штучний інтелект	Б-515	підгрупа	доц. Демідов П.Г .
понеділок	2	10:05	11:25	Лб2 Штучний інтелект	Б-523	підгрупа	ст.в. Селіванова А.В .
понеділок	2	10:05	11:25	Лб1 Штучний інтелект	Б-515	підгрупа	доц. Демідов П.Г .
вівторок	3	12:05	13:25	Моделювання фінансово-господарської діяльності підприємства	корпус Н, ауд.Н-208	вся	undefined
вівторок	4	13:50	15:10	Моделювання фінансово-господарської діяльності підприємства	корпус Н, ауд.Н-208	вся	undefined

Рис.3.16. Таблиця виведеного розкладу

3.4. Технологія використання розробленого Web-додатку

При вході на сторінку веб-додатку, користувач побачить форму для вибору параметрів розкладу, поля із корисними посиланнями та форму зворотнього зв'язку.

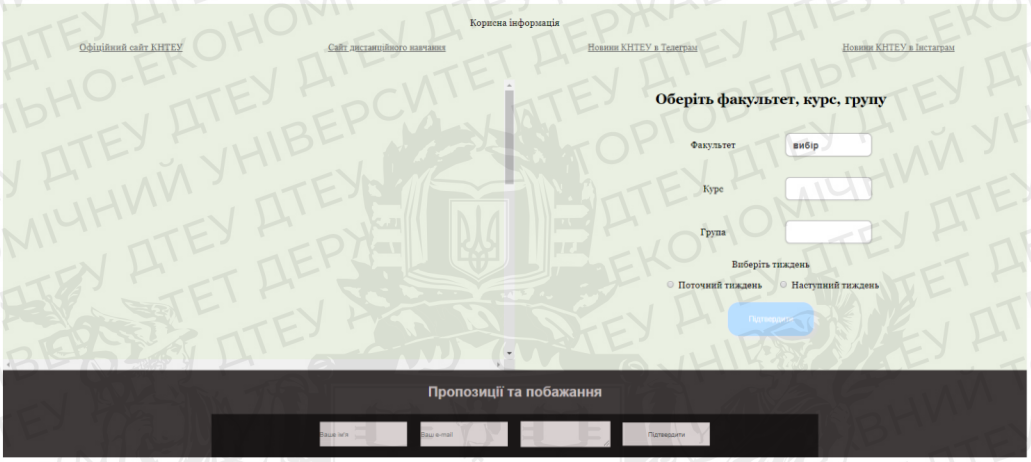


Рис.3.17. Готова сторінка веб-додатку

Користувач має можливість перейти на сторінки університету в соціальних мережах, зайти на сайт дистанційного навчання та відвідати офіційний сайт університету.

За необхідності подивитися розклад необхідно із випадаючих списків у формі обрати свій факультет, курс, групу, необхідний тиждень навчання та натиснути кнопку «Підтвердити».

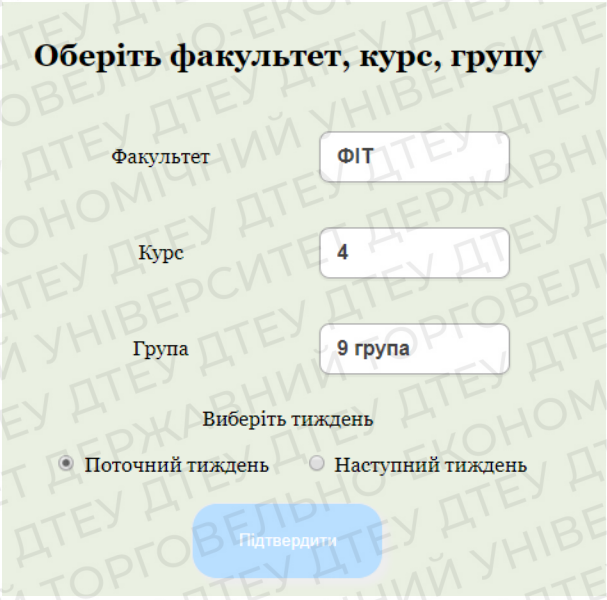
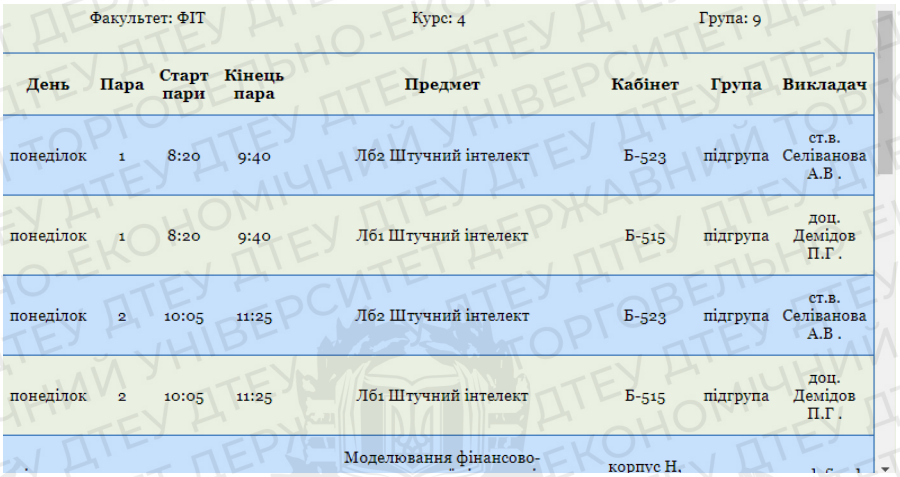


Рис.3.18. Приклад заповнення форми

Після цього користувачеві відкриється таблиця, в якій буде відображений розклад на обраний користувачем тиждень.



Факультет: ФІТ | Курс: 4 | Група: 9

День	Пара	Старт пари	Кінець пара	Предмет	Кабінет	Група	Викладач
понеділок	1	8:20	9:40	Лб2 Штучний інтелект	Б-523	підгрупа	ст.в. Селіванова А.В .
понеділок	1	8:20	9:40	Лб1 Штучний інтелект	Б-515	підгрупа	доц. Демідов П.Г .
понеділок	2	10:05	11:25	Лб2 Штучний інтелект	Б-523	підгрупа	ст.в. Селіванова А.В .
понеділок	2	10:05	11:25	Лб1 Штучний інтелект	Б-515	підгрупа	доц. Демідов П.Г .
				Моделювання фінансово-	корпус Н.		

Рис.3.19. Приклад виведеного розкладу

Щоб задати нові параметри розкладу, необхідно перезавантажити сторінку, аби очистилися попередні параметри запиту.

ВИСНОВКИ

У випускній кваліфікаційній 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 напрямів підвищення рівня інформованості студентів 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 перевантажуватиме 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 проаналізовано різні варіанти верстання веб-сторінок, такі як: таблиці, метод float, модульна система та використання чат-ботів. Як висновок, було обрано модульну систему верстання сайтів, що дозволяє створювати гнучкі веб-сторінки із мінімальним використанням ресурсів. Функціональний, інтуїтивно простий та що містить достатню кількість корисної інформації сайт допоможе у процесі підвищення інформованості студента.

В третьому розділі здійснювалась розробка інформаційної системи «електронний розклад студента». На основі проведеного аналізу використано організацію інформаційної бази на основі файлів типу JSON. Дана технологія гарантує стабільну роботу Web-додатку та зручність редагування файлів для їх

подальшого використання. Крім того, в подальшому файли даного типу можна інтегрувати в базу даних.

Було розроблено 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 допомагає користувачу використовувати веб-ресурс в повному обсязі. Даний інтерфейс являє собою панель навігації, що забезпечує звертання до сайтів, безпосередньо пов’язаних із навчальним процесом в університеті та структурована форма, що забезпечує зручну організацію пошуку необхідних даних про розклад студента.

За результатами розробки отримано сайт, який можна використовувати у навчальному процесі. Сайт включає такі можливості: перехід по веб-сторінкам, пов’язаним із навчальним закладом, зворотній зв’язок, форму вибору запиту для пошуку необхідного розкладу.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Організація освітнього процесу. [Електронний ресурс]. – Режим доступу: https://knute.edu.ua/blog/read/?pid=7330&uk
2. Інструкція щодо складання 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 затвердження Положення про електронні освітні ресурси. [Електронний ресурс]. – Режим доступу: https://zakon.rada.gov.ua/laws/show/z1695-12/ed20161007#n26)
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.htm
7. Бібліотечно-інформаційна діяльність. Терміни та визначення понять. [Електронний ресурс]. – Режим доступу: http://oipopp.ed-sp.net/public/repository/dstu_7448-2013_bibliotechno-informaciyna_diyalnist._terminy_i_vyznachennya.pdf
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.html
10. Структура інформаційної системи. [Електронний ресурс]. – Режим доступу: https://studfile.net/preview/5064248/page:9/
11. Components Of Information System. [Електронний ресурс]. – Режим доступу: https://www.geeksforgeeks.org/components-of-information-system/
12. Сучасні методи веб-програмування. [Електронний ресурс]. – Режим доступу: http://sites.znu.edu.ua/webprog/lect/1222.ukr.html
13. Oracle MySQL Database Service. [Електронний ресурс]. – Режим доступу: https://www.oracle.com/MySQL/
14. Семенюк Р. А. ПЕРЕВАГИ ТА НЕДОЛІКИ ВИКОРИСТАННЯ СУБД MONGODB. [Електронний ресурс]. – Режим доступу: http://eprints.zu.edu.ua/24821/1/Semenyuk_APSI2017.pdf
15. Introducing JSON. [Електронний ресурс]. – Режим доступу: https://www.json.org/json-en.html
16. What is JSON? Електронний ресурс]. – Режим доступу: https://www.hostinger.com/tutorials/what-is-json
17. JSON. [Електронний ресурс]. – Режим доступу: https://developer.mozilla.org/uk/docs/Glossary/JSON
18. Методи верстки сайтів. [Електронний ресурс]. – Режим доступу: http://programer.org.ua/metodi-verstki-sajtiv/
19. Розбираємо властивість float в CSS. [Електронний ресурс]. – Режим доступу:http://programer.org.ua/rozbirayemo-vlastivist-float-v-css/
20. Modular Grid: Complete Beginner's Guide. [Електронний ресурс]. – Режим доступу: https://icons8.com/articles/modular-grid-complete-beginners-guide/
21. Уроки з Bootstrap v3. [Електронний ресурс]. – Режим доступу: https://tokar.ua/read/6901
22. A guide to using Telegram bots [Електронний ресурс]. – Режим доступу: https://www.dignited.com/18444/guide-using-telegram-bots/

23. Переваги та недоліки використання чат-ботів для бізнесу. [Електронний ресурс]. – Режим доступу: https://internetdevels.ua/blog/pros-and-cons-of-using-chatbots-for-business
24. Microsoft Excel. [Електронний ресурс]. – Режим доступу:https://www.microsoft.com/uk-ua/microsoft-365/excel
25. Visual Studio Code. [Електронний ресурс]. – Режим доступу:https://code.visualstudio.com/
26. ЗАСТОСУВАННЯ UML (ЧАСТИНА 2). ДІАГРАМА ПОСЛІДОВНОСТІ - SEQUENCE DIAGRAM [Електронний ресурс]. – Режим доступу: http://www.dut.edu.ua/ua/news-1-626-7897-zastosuvannya-uml-chastina-2-diagrama-poslidovnosti---sequence-diagram
27. UML Use Case Diagram. [Електронний ресурс]. – Режим доступу: https://www.javatpoint.com/uml-use-case-diagram
28. HTML. [Електронний ресурс]. – Режим доступу: https://developer.mozilla.org/uk/docs/Web/HTML
29. Основи CSS. [Електронний ресурс]. – Режим доступу: https://developer.mozilla.org/uk/docs/Learn/Getting_started_with_the_web/CSS_basics
30. jQuery: write less, do more. [Електроний ресурс] – Режим доступу: https://jquery.com/