

Київський національний торговельно-економічний університет

Кафедра комп'ютерних наук та інформаційних систем

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

**«Розробка програмно - інструментальних засобів обліку
слухачів навчального центру»**

Студентки 4 курсу, 9 групи,

спеціальності

122 «Комп'ютерні науки»

Дьяченко Дар'я
Володимирівна

підпис студента

Науковий керівник

кандидат фізико-математичних наук

Філімонова Тетяна

Олегівна

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 інформаційних систем
Спеціальність 122 «Комп'ютерні науки»

Зав. кафедри _____

Затверджую
Пурський О.І.
«20» грудня 2020р.

Завдання на випускню кваліфікаційну роботу студентці

Дьяченко Дар'я Володимирівна

1. Тема випускної кваліфікаційної роботи (проекту)
«Розробка системи обліку слухачів навчального центру»
Затверджена наказом ректора від «15» грудня 2020 р. № 3780
2. Строк здачі студентом закінченої роботи 31 травня 2021 року
3. Цільова установка та вихідні дані до роботи
Мета роботи: обґрунтування та розробка системи обліку слухачів навчального центру.
Об'єкт дослідження: процес розробки програмно - інструментальних засобів обліку слухачів навчального центру.
Предмет дослідження: засоби створення автоматичної системи обліку успішності та відвідування слухачів навчального центру.
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	<u>Філімонова Т.О.</u>	22.12.2020 р.	22.12.2020 р.
2	<u>Філімонова Т.О.</u>	22.12.2020 р.	22.12.2020 р.
3	<u>Філімонова Т.О.</u>	22.12.2020 р.	22.12.2020 р.

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

ВСТУП

РОЗДІЛ 1. Аналітичне дослідження системи освіти в Україні.

1.1. Аналіз сучасного ведення обліку в навчальних закладах

1.2. Обґрунтування доцільності розробки

РОЗДІЛ 2. Розробка програмно - інструментального засобу обліку слухачів навчального центру

2.1. Загальна концепція

2.3. Проектування інтерфейсу та структури

2.4. Модель системи обліку слухачів

2.5. Вибір програмних засобів для роботи

РОЗДІЛ 3. Реалізація програмно - інструментального засобу обліку слухачів навчального центру

3.1. Підготовка до розробки програми

3.2. Створення майбутнього продукту

3.3. Тестування засобу обліку

ВИСНОВКИ

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

ДОДАТОК

7. Календарний план виконання роботи

№ Пор.	Назва етапів випускної кваліфікаційної роботи	Строк виконання етапів роботи	
		За планом	фактично

1	2	3	4
1	<i>Вибір теми випускної кваліфікаційної роботи</i>	05.10.2020	05.10.2020
2	<i>Розробка та затвердження завдання на випускну кваліфікаційну роботу</i>	18.12.2020	18.12.2020
3	<i>Вступ</i>	03.02.2021	03.02.2021
4	<i>РОЗДІЛ 1. Аналітичне дослідження системи освіти України.</i>	28.02.2021	28.02.2021
5	<i>РОЗДІЛ 2. Розробка програмно - інструментального засобу обліку слухачів навчального центру</i>	06.04.2021	06.04.2021
6	<i>РОЗДІЛ 3. Реалізація програмно - інструментального засобу обліку слухачів навчального центру</i>	12.05.2021	12.05.2021
7	<i>Висновки</i>	14.05.2021	14.05.2021
8	<i>Здача випускної кваліфікаційної роботи на кафедрі науковому керівнику</i>	20.05.2021	20.05.2021
9	<i>Попередній захист випускної кваліфікаційної роботи</i>	26.05.2021	26.05.2021
11	<i>Виправлення зауважень, зовнішнє рецензування випускної кваліфікаційної роботи</i>	27.05.2021	
12	<i>Представлення готової зшитої випускної кваліфікаційної роботи на кафедрі</i>	31.05.2021	
13	<i>Публічний захист випускної кваліфікаційної роботи</i>	За розкладом роботи ЕК	

8. Дата видачі завдання «22» грудня 2020 р.

9. Керівник випускної кваліфікаційної роботи (проекту)

Філімонова Т.О.

(прізвище, ініціали, підпис)

10. Гарант освітньої програми

Демідов П.Г.

(прізвище, ініціали, підпис)

11. Завдання прийняв до виконання студент-дипломник

Дьяченко Д.В.

(прізвище, ініціали, підпис)

12. Відгук керівника випускної кваліфікаційної роботи (проекту)

Керівник випускної кваліфікаційної роботи (проекту)

31.05.2020 р.

(підпис, дата)

13. Висновок про випускну кваліфікаційну роботу (проект)

Випускна кваліфікаційна робота (проект) студента Дьяченко Д.В.

(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 випускній кваліфікаційній роботі здійснено аналіз сучасного ведення обліку в навчальних закладах України. Поставлена задача по створенню програмно – інструментального засобу обліку слухачів навчального центру. Теоретично обґрунтовано актуальність та доцільність розробки. Обраний спосіб та формат виконання роботи і необхідні для цього програмні засоби. Створена система у вигляді журналу відвідування та успішності, котра знаходиться на базі основного сайту навчального закладу, також був розроблений зовнішній вигляд даної веб-сторінки.

Ключові слова: система, облік, код, веб-сторінка, CSS, Html, JavaScript, TypeScript.

Summary

The graduation qualification work the analysis of modern accounting in educational institutions of Ukraine is carried out. The task is to create a software and tool for accounting for students of the training center. The relevance and expediency of the development are theoretically substantiated. The chosen method and format of work and the necessary software. A system in the form of a log of attendance and performance, which is based on the main site of the educational institution, was also developed, and the appearance of this web page was developed.

Keywords: system, accounting, code, web page, CSS, Html, JavaScript, TypeScript.

ЗМІСТ

ВСТУП	8
РОЗДІЛ 1. АНАЛІТИЧНЕ ДОСЛІДЖЕННЯ СИСТЕМИ ОСВІТИ В УКРАЇНІ	9
1.1 Аналіз сучасного ведення обліку в навчальних закладах	11
1.2 Обґрунтування доцільності розробки	13
Висновки до розділу	14
РОЗДІЛ 2. РОЗРОБКА ПРОГРАМНО - ІНСТРУМЕНТАЛЬНОГО ЗАСОБУ ОБЛІКУ СЛУХАЧІВ НАВЧАЛЬНОГО ЦЕНТРУ	16
2.1 Загальна концепція	16
2.3 Проектування інтерфейсу та структури	17
2.4 Модель системи обліку слухачів	19
2.5 Вибір програмних засобів для роботи	20
Висновки до розділу	22
РОЗДІЛ 3. РЕАЛІЗАЦІЯ ПРОГРАМНО - ІНСТРУМЕНТАЛЬНОГО ЗАСОБУ ОБЛІКУ СЛУХАЧІВ НАВЧАЛЬНОГО ЦЕНТРУ	23
3.1 Підготовка до розробки програми	23
3.2 Створення майбутнього продукту	24
3.3 Тестування засобу обліку	42
Висновки до розділу	47
ВИСНОВКИ	48
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	49
ДОДАТОК	51

ВСТУП

У нашому світі найважливішу роль відіграє інформація. Сучасній людині просто необхідна віртуальна освітня система, що дає можливість відстежувати навчальний процес в режимі реального часу. Електронна система обліку у вигляді бази даних яка буде знаходитися в мережі інтернет та вирішить багато людських проблем в освітньому процесі. Ця система в відмінності від паперових журналів завжди буде під рукою, буде мати актуальні дані, не схильна до втручання сторонніх осіб, до механічного пошкодження, в ній завжди можна виправити допущену помилку, більше не доведеться чекати поки інший викладач заповнить журнал, не потрібно буде шукати його по кабінетам.

Ведення обліку відвідування та успішності не використовуючи паперові книжки – це майбутнє яке неминуче настане, але йому треба деякий час для того, щоб люди мали змогу звикнути.

Оскільки тема даної випускної кваліфікаційної роботи «Розробка програмно - інструментальних засобів обліку слухачів навчального центру». Перед нами поставлено завдання створити віртуальну систему обліку, яка буде мати вигляд електронного журналу.

Мета роботи: обґрунтування та розробка програмно - інструментального засобу обліку слухачів навчального центру.

Об'єкт дослідження: процес розробки програмно - інструментальних засобів обліку слухачів навчального центру.

Предмет дослідження: засоби створення автоматичної системи обліку успішності та відвідування слухачів навчального центру.

Методи дослідження: аналітичний метод, метод теорії, метод програмування.

Практичне значення: Отриманий кінцевий продукт можна буде використовувати як базу для створення обліку у навчальних закладах різного рівня освіти.

Інформаційна база: Електронні ресурси, інтернет.

РОЗДІЛ 1. АНАЛІТИЧНЕ ДОСЛІДЖЕННЯ СИСТЕМИ ОСВІТИ В УКРАЇНІ

Система освіти України складається із закладів освіти, наукових, науково-методичних і методичних установ, науково-виробничих підприємств, державних і місцевих органів управління освітою та самоврядування в галузі освіти. Структура освіти включає: дошкільну освіту, загальну середню освіту, позашкільну освіту, професійно-технічну освіту, вищу освіту, післядипломну освіту, аспірантуру, докторантуру, самоосвіту.

Дошкільна освіта і виховання здійснюються у сім'ї та дошкільних закладах освіти у взаємодії з сім'єю і мають на меті забезпечення фізичного, психічного здоров'я дітей, їх всебічного розвитку, вироблення умінь, навичок, необхідних для подальшого навчання. Дошкільними закладами освіти є: дитячі ясла, дитячі садки, дитячі ясла-садки, сімейні, прогулянкові, дошкільні заклади компенсуючого (для дітей, які потребують корекції фізичного і психічного розвитку) та комбінованого типів з короткотривалим, денним, цілодобовим перебуванням дітей, а також дитячі садки інтернатного типу, дитячі будинки та інші.

Загальна середня освіта забезпечує всебічний розвиток дитини як особистості, її нахилів, здібностей, талантів, трудову підготовку, професійне самовизначення, формування загальнолюдської моралі, засвоєння визначеного суспільними, національно-культурними потребами обсягу знань про природу, людину, суспільство і виробництво, екологічне виховання, фізичне вдосконалення.

Повна загальна освіта в Україні є обов'язковою і надається у різних типах закладів освіти. Основним з-поміж них є середня загальноосвітня школа трьох ступенів: I – початкова школа (1-4 класи), що забезпечує початкову загальну освіту, II – основна школа (5-9 класи), що забезпечує базову загальну середню освіту, III – старша школа (10-11 класи), що забезпечує повну загальну середню освіту.

Для розвитку здібностей, обдарувань і талантів дітей створюються профільні класи (з поглибленим вивченням окремих предметів або початковою допрофесійною підготовкою), спеціалізовані школи, гімназії, ліцеї, колегіуми, а також різні типи навчально-виховних комплексів, об'єднань.

Позашкільна освіта та виховання спрямовуються на розвиток здібностей, талантів у дітей, учнівської та студентської молоді, задоволення їх інтересів, духовних запитів і потреб у професійному самовизначенні.

До позашкільних закладів освіти належать: палаци, будинки, центри, станції дитячої, юнацької творчості, учнівські та студентські клуби, дитячо-юнацькі спортивні школи, школи мистецтв, студії, початкові спеціалізовані мистецькі навчальні заклади, бібліотеки, оздоровчі та інші заклади.

Професійно-технічна освіта зорієнтована на здобуття професії, перепідготовку, підвищення професійної кваліфікації. Відповідними закладами освіти є: професійно-технічні училища, професійно-художні училища, професійні училища соціальної реабілітації, училища-агрофірми, училища-заводи, вищі професійні училища, навчально-виробничі центри, центри підвищення і перепідготовки робітничих кадрів, навчально-курсіві комбінати, інші типи закладів, що надають робітничу професію.

Вища освіта забезпечує фундаментальну наукову, професійну та практичну підготовку, здобуття громадянами освітньо-кваліфікаційних рівнів відповідно до їх покликань, інтересів і здібностей, удосконалення наукової та професійної підготовки, перепідготовку та підвищення їх кваліфікації. Вищими закладами освіти є : технікуми, коледжі, інститути, консерваторії, академії, університети та інші.

Для вищих закладів освіти встановлено чотири рівні акредитації: I – технікум, училище, інші прирівняні до них вищі заклади освіти; II – коледж, інші прирівняні до нього вищі заклади освіти; III і IV рівні (залежно від наслідків акредитації) – інститут, консерваторія, академія, університет.

Післядипломна освіта (спеціалізація, стажування, клінічна ординатура,

підвищення кваліфікації та перепідготовка кадрів) сприяє одержанню нової кваліфікації, нової спеціальності та професії на основі раніше здобутої у закладі освіти і досвіду практичної роботи, поглибленню професійних знань, умінь за спеціальністю, професією. До закладів післядипломної освіти належать: академії, інститути, (центри) підвищення кваліфікації, перепідготовки, вдосконалення, навчально-курсів комбінати; підрозділи вищих закладів освіти (філіали, факультети, відділення та інші); професійно-технічні заклади освіти; відповідні підрозділи в організаціях та на підприємствах.

Для самоосвіти громадян державними органами, підприємствами, установами, організаціями, об'єднаннями громадян, громадянами створюються відкриті та народні університети, лекторії, бібліотеки, центри, клуби, навчальні програми тощо [1].

Пандемія коронавірусу негативно вплинула на освітній процес – школи та вчителі були змушені швидко переходити в онлайн-режим. Не всім це далось легко. Це, безперечно, негативно відіб'ється на навчальних результатах. Найбільше постраждають школярі з бідних родин (так, за опитуванням ОУЖД, 15% домогосподарств із дітьми не мають комп'ютера, ноутбука чи планшета) та учні сільських шкіл – адже в деяких селах немає ні інтернету, ні навіть мобільного зв'язку [2].

1.1 Аналіз сучасного ведення обліку в навчальних закладах

В наш час дуже велика швидкість темпу розвитку суспільства. Щоби відчувати себе комфортно, та бути у курсі всіх подій людина повинна поглинати і обробляти величезну кількість інформації. Тому інформаційні технології відіграють дуже важливу роль у нашому житті. Вони являють собою швидкий доступ до безмежної бази даних, да і в цілому дуже розширюють можливості сучасної людини.

У наш час в більшості навчальних закладів України використовують паперові журнали для обліку відвідування та успішності учнів. Так як в

більшості навчальних закладах, в зокрема школах, майбутніх фахівців у різних видах діяльності навчають люди близькі до пенсійного віку. Такі люди як правило погано, а інколи в загальні не мають змогу адаптуватись до швидкого темпу прогресу людства. Це є причиною того, що за часів розвинених технологій, розумних телефонів і комп'ютерів ми все ще ведемо облік через папір у вигляді журналів та щоденників.

Використання електронних щоденників і журналів з самого початку викликало розбіжності, особливо всередині школи. Так, стан комп'ютеризації окремих шкіл, особливо сільських, не дозволяє повноцінно використовувати можливості цього програмного ресурсу. Але давайте без урахування технічного боку розглянемо плюси і мінуси електронних журналів і щоденників [3].

Інколи причиною зупинки прогресу для закладів отримання різного рівня освіти є керівництво, пояснення можуть бути різними: недостатня кількість бюджетних коштів, не бажання вводити нові технології, так, як з приходом нового і незвіданого будуть обов'язкового з'являться проблеми які нікому не потрібні. Але прогрес все ж таки бере своє. Заклади різного рівня освіти вже почали використовувати ресурси, які знаходяться на базі всесвітньої павутини. Це означає, що в майбутньому ми не будемо пам'ятати що колись для обліку використовували папір.

По-перше, електронний щоденник вкрай зручний для багатьох батьків. Адже часто діти намагаються хитрувати і не дуже охоче діляться своїми успіхами в навчанні. Раніше таким батькам доводилося дзвонити класному керівнику, тепер вони можуть своєчасно відстежувати ситуацію з відмітками. Однак стежити за успішністю варто стежити всім, а не тільки батькам «двійшників», а з електронним щоденником це робити набагато зручніше [3].

По-друге електронний облік більш зручний ніж його паперовий аналог, зокрема для вчителів. Раніше викладачу було потрібно заповнювати журнал під час заняття по його предмету, інколи учні не встигали скласти навчальний матеріал вчасно, а журнал потрібно було вже нести на наступний урок, тому

багато вчителів мали свій власний зошит де записували оцінки і тільки потім мали змогу перенести їх до класного журналу, після уроків, чи під час наступного заняття по їх предмету. Це є причиною не рідких помилок у записах, а як результат сварка між учням та вчителями, яка інколи доводить до виклику батьків у школу.

Використання онлайн-щоденників учнями теж не менш зручне.

Перевага використання електронних журналів і щоденників полягає в тому, що вчитель сам записує домашнє завдання, а значить, навіть якщо учень пропустив заняття через хворобу, він зможе підготуватися до уроків [3].

Також зникне необхідність заповнювати щоденник кожної неділі, записуючи туди один и той же розклад уроків, котрий не міняється вздовж всього семестру. Вчителі не повинні будуть збирати щоденники кожен раз, коли вони хочуть поставити позначку, слухаючи виправдання про те, що хтось із учнів «щоденник забув удома», «втратив» і т. п. Також використовувати електронний формат значно зручніше для ведення звітності. У кілька натискань можна визначити «відмінників» і «двієчників», знайти пропуски, що впливають на річну оцінку. Такої можливості особливо раді класні керівники, які раніше повинні були записувати двадцять п'ять або навіть тридцять ПІБ до кожного предмету, кожен новий семестр [3].

Завдання завуча з навчально-виховної частини - стежити за навчальним процесом. Це означає контролювати успішність, відвідуваність, дисципліну. Раніше бідні завучі збирали паперові звіти від класних керівників. Хто скільки прогуляв, кому світить двійка в чверті. Бідні класні керівники щомісяця брали в руки паперовий журнал і відзначали відвідування кожного учня з усіх предметів. З електронним журналом досить натиснути дві кнопки - і потрібний звіт піде на друк. Можна дізнатися хоч середній бал класу з усіх предметів - було б бажання. .

1.2 Обґрунтування доцільності розробки

Паперовий журнал - річ. Він десь лежить, хтось його бере. Він не може

бути в двох місцях одночасно. Найзапекліші виступу - в квітні і травні. Буває, хтось із учнів вирішив обнулити оцінки, та й вчителі теж люди. Вони не застраховані від помилки «покласти туди, щоб не втратити». Іноді старий журнал випадково знаходять на початку наступного року і в зовсім несподіваних місцях. Після епопеї відновлення оцінок по вчительським записів і щоденникових оцінкам. Тепер цієї проблеми немає. З журналом одного класу можна працювати одночасно, вчителі більше не залежать один від одного. Після уроків і на перервах не треба йти до вчительської. Електронний журнал неможливо втратити або зіпсувати [4]. Електронний аналог дозволяє виставляти оцінки та відвідування як під час проходження заняття, так і після нього, вчителю не потрібно поспішати заповнювати його, так як він завжди під рукою у вигляді смартфона чи ноутбуку, це приводить до меншої кількості помилок які могли з'явитися під час поспіху.

Також його складніше пошкодити так, як він знаходиться на базі інтернет. Єдиний недолік – в навчальному закладі у вільному доступі повинен бути інтернет, але в наші часи практично во всіх закладах вже є Wi-Fi роутери, та мобільний інтернет практично у всіх власників смартфонів.

Такий вид обліку можна ввести в такі заклади як: школи, університеті, інститути, коледжі, навчальні центри та інші.

Електронні журнали у навчальних закладах дозволять зменшити потребу у паперовій продукції, що не може не радувати. Це не тільки заощадить бюджет навчального закладу, а і має позитивний вплив на екологічну сторону проблеми.

Висновки до розділу

Проаналізувавши сучасну систему освіти в Україні та методи ведення обліку ми можемо зробити наступні висновки:

1. На даний момент Україна має дуже розвинену систему освіти. Так, як діти можуть вчитися у класах з поглибленим вивченням конкретного предмету, а також ходити на відповідні курси. Існують багато позашкільних закладів котрі

розвивають творчі та соціальні здібності дітей. Також вища освіта доступна багатьом людям, що розширює можливості отримати професійні навички котрі будуть підтверджені державним документом, відповідно людина може відкрити для себе обрану нею професію.

2. Введення електронного обігу успішності студентів та школярів – це зручна і актуальна в наш час система, котра допоможе економити час для вчителів, та здобуття інформації що до оцінок для школярів та їх батьків. Зменшить використання паперу у навчальних закладах, та роботу з цим папером. Оскільки шкільний журнал буде знаходитися на базі віртуальної системи він буде доступний у будь який час при наявності інтернету. Єдиний недолік даної системи – це тяжка адаптація зі сторони вчителів та батьків.

РОЗДІЛ 2. РОЗРОБКА ПРОГРАМНО - ІНСТРУМЕНТАЛЬНОГО ЗАСОБУ ОБЛІКУ СЛУХАЧІВ НАВЧАЛЬНОГО ЦЕНТРУ

2.1 Загальна концепція

За останні кілька років інтернет перетворився на величезне культурне явище, яке не менш популярне і впливове, ніж кіно. Новому користувачеві, який потрапив в інтернет, доводиться знайомитися з масою нової інформації. Онлайн-терміни сиплються на нього, як з рогу достатку - впору розгубитися. Один з найбільш нових термінів - «електронний журнал обліку». Електронний журнал оцінок і електронний щоденник учня з правом доступу до нього є ефективним механізмом підвищення якості освіти. Загальноосвітні установи поінформовані про цю систему і розуміють, що підключення електронних щоденників стане невід'ємною частиною нашого життя.

Нещодавно Міністерство України запустило пілотний проект державних електронних щоденників і журналів для шкіл. З листопада вони будуть доступні всім, а дублювання в паперові журнали взагалі скасують [5].

Сучасною системою обліку для багатьох навчальних закладів є – журнал відвідувань. Оскільки тема даного диплому «Розробка програмно - інструментальних засобів обліку слухачів навчального центру», з цього можна зробити висновок, що журнал який буде використовуватись в навчальному центрі повинен мати функцію виставлення оцінок та відвідувань. Завдяки наданим даним можна уявити яким чином буде виглядати журнал. Для початку була виконана приблизна схема майбутнього проекту

використовуючи Microsoft Excel та Adobe Illustrator (рис.2.1).

Головна сторінка								
Розклад	1	Група №1						
Журнал	2	Прізвище	дата	дата	дата	дата	дата	Екзамен
Новини	3	ФІО	-	+	+	+	+	зданий
	4	ФІО	+	+	-	+	+	зданий
Повідомлення	5	ФІО	+	+	+	+	+	зданий
	6	ФІО	+	+	+	+	+	зданий
Налаштування	7	ФІО	+	+	+	+	+	зданий
	8	ФІО	+	+	+	+	+	зданий
	9	ФІО	-	+	+	+	+	зданий
	10	ФІО	+	+	+	+	+	зданий
	11	ФІО	+	+	+	+	+	зданий
	12	ФІО	+	+	+	+	+	зданий
	13	ФІО	+	+	+	+	+	зданий
	14	ФІО	-	+	+	+	-	не зданий
	15	ФІО	+	+	+	+	+	зданий

Журнал
Група №1
Група №2
Група №3
Група №4

Рис. 2.1. Схема майбутнього дизайну.

Був проведений аналіз в ході якого був зроблений висновок що дана робота буде виконуватись на базі Html, використовуючи CSS і TypeScript. Фінал майбутнього проекту повинен буде мати вигляд додаткових веб-сторінок, котрі будуть прив'язані до головного сайту навчального центру. Це означає що буде створений макет особистого кабінету для вчителя, в якому він буде мати змогу обирати необхідну йому групу та виставляти бали та відвідування.

2.3 Проектування інтерфейсу та структури

Далі використовуючи Adobe Illustrator був створений приблизний дизайн майбутнього проекту (рис.2.2.).

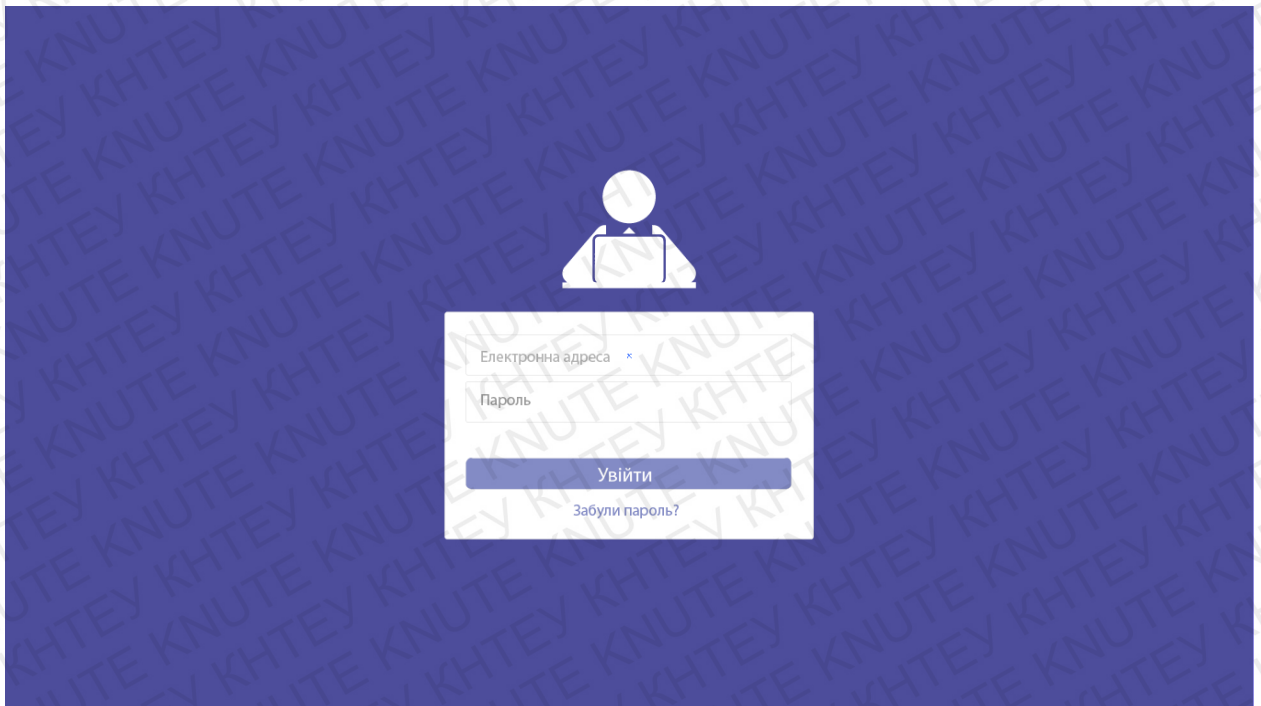


Рис. 2.2. Приблизний вигляд першої сторінки

На даному малюнку зображений приблизний майбутній вигляд веб-сторінки для входу у власний кабінет вчителя. Після введення вірної електронної адреси та паролю користувач буде потрапляти на сторінку особистого кабінету, вона буде мати вигляд як на рисунку 2.3.

Це основна сторінка, вверху буде розташовуватися логотип та назва підприємства, декілька кнопок, ПІБ вчителя, його пошта та фотографія.

В центрі сторінки буде відображатись саме журнал з ПІБ студентів та їх оцінками. Вчитель буде мати змогу обрати необхідну групу та вносити дані, котрі будуть зберігатись у системі.

ПРОГРАМАЙСТЕР

школа з підготовки IT-спеціалістів

Дяченко Дар'я

Володимирівна

kyiv@itprogramaist.com

Група №1

Група №2

Група №3

ГРУПА №2

#	ПІБ	05.09	12.09	19.09	26.09	05.10	12.10	Екзамен
1	Кравчук Вадим Анатолійович	○	✗	○	○	✗	○	Зданий
2	Кравчук Вадим Анатолійович	○	✗	○	○	✗	○	Не зданий
3	Кравчук Вадим Анатолійович	○	✗	○	○	✗	○	Зданий
4	Кравчук Вадим Анатолійович	○	✗	○	○	✗	○	Зданий
5	Кравчук Вадим Анатолійович	○	✗	○	○	✗	○	Зданий
6	Кравчук Вадим Анатолійович	○	✗	○	○	✗	○	Зданий
7	Кравчук Вадим Анатолійович	○	✗	○	○	✗	○	Зданий
8	Кравчук Вадим Анатолійович	○	✗	○	○	✗	○	Зданий
9	Кравчук Вадим Анатолійович	○	✗	○	○	✗	○	Зданий
10	Кравчук Вадим Анатолійович	○	✗	○	○	✗	○	Зданий

Рис. 2.3. Приблизний вигляд другої сторінки.

2.4 Модель системи обліку слухачів

Даний проект буде орієнтований на основний сайт навчального закладу. Тобто він буде мати вигляд нових функцій для основного сайту. Стежити за справністю сайту буде системний адміністратор закладу. Система буде працювати наступним чином. Вчитель під час занять використовуючи один з багатьох нині існуючих браузерів заходить на головний сайт закладу. Далі натискає кнопку «Увійти», тобто увійти у власний кабінет. Вводить електронну пошту і пароль який він заздалегідь зареєстрував у базі даних системи. Якщо дані які вводить вчитель є у базі – система надає йому доступ до власного кабінету. Якщо такі дані невірні або не існують сайт видає помилку та запросить користувача повторно ввести дані. У випадку якщо вчитель не пом’ятає вірних даних до входження у власний кабінет, він зв’язується з адміністрацією школи використовуючи інтернет або мобільний зв’язок.

Отримавши доступ вчитель потрапить на сторінку яка буде мати вигляд як на рисунку 2.3.

На даній сторінці викладач буде мати змогу виставляти бали зароблені студентами та проставляти відвідування.

Спочатку необхідно буде обрати з списку групу, потім обрати поле з прізвищем, ім'ям та по батькові студента. Після обрати сьогоднішню дату та виставити бали учню за роботу на парі, або залишити клітку пустою якщо за цю дату матеріал не був відпрацьований. У випадку коли студент не з'явився на занятті виставити у клітинці напроти його імені літеру «н» що буде означати «не був присутній». Після завершення заняття потрібно буде натиснути кнопку «Зберегти», для того, щоби зберегти дані не сервері системи. Вчитель буде мати змогу редагувати тільки клітинки, в яких можна виставити оцінки та присутність.

2.5 Вибір програмних засобів для роботи

Для реалізації проекту нам потрібно буде використовувати такі додатки:

- Google Chrome
- Html
- Css
- JavaScript(Type Script)
- Node.JS
- Angular
- VisualStudio

Далі розберемо кожний пункт зі списку наведеного вище.

Visual Studio - це собою інтегроване середовище розробки. Засіб для написання, складання, редагування і налагодження коду, а також для публікації створених у програмі додатків. Інтегроване середовище розробки (IDE) являє собою багатофункціональну програму, яку можна використовувати для різних аспектів розробки програмного забезпечення. Крім стандартного редактора і відладчика, які існують в більшості середовищ

IDE, Visual Studio включає в себе компілятори, засоби автозавершення коду, графічні конструктори і багато інших функцій для спрощення процесу розробки [6].

Google Chrome – це браузер через який буде створений вхід до робочого простору.

Node.js - Виконавча коду JavaScript поза браузера. Це платформа, яка дозволяє писати серверний код для динамічних веб-додатків і веб-сторінок, а також для програм командного рядка. За допомогою Node.js можна використовувати одну мову програмування для розробки веб-додатків замість застосування різних мов для роботи над фронтенда і бекенд.

Фронтенд (англ. Front-end) - клієнтська сторона призначеного для користувача інтерфейсу до програмно-апаратної частини сервісу.

Бекенд (англ. Back-end) - програмно-апаратна частина сервісу. Бекенд відповідає за здійснення функціонування внутрішньої частини веб-сайту [7].

Node.js - не окрема мова програмування, а платформа для використання JavaScript на стороні сервера. Якщо говорити про мову, то як для фронтенд, так і для бекенд використовується один і той же JavaScript. Різниця тільки в наборі API, які використовують фронтендери і бекендери. Браузерні JavaScript використовує Web API, які забезпечують доступ до DOM і призначеного для користувача інтерфейсу сторінок і веб-додатків. Серверний JavaScript використовує API, що забезпечують доступ до файлової системи додатків, http-запитів, потоків. Тобто Node.js - це технологія для використання JS на бекенд [8].

Angular JS (або клієнтський) фреймворк - це технологія, що забезпечує нас інструментами для створення веб-додатки, але також задає дизайн програми і організацію коду [9]. AngularJS - це фреймворк для роботи з JavaScript, підтримуваний Google. Він використовується для створення односторінкових додатків і дозволяє взаємодіяти з DOM (англ. Document Object Model - об'єктна модель документа).

Працює AngularJS за схемою MVC (англ. Model-View-Controller - модель-вид-контролер) - вона розділяє додаток на три окремих частини, які можна змінювати незалежно один від одного.

- Модель - надає інформацію і реагує на команди контролера.
- Вигляд - відповідає за відображення даних моделі і стежить за змінами.
- Контролер - реагує на дії користувача і оповіщає модель про те, що потрібно оновитися.

Така схема дозволяє легко працювати з фреймворком, швидко писати код і тестувати його [10].

Висновки до розділу

У даному розділі був проведений аналіз щодо технології виконання завдання, та вирішено які додатки будуть використовуватись, також був виконаний розпис функцій цих додатків. Була створена схема та дизайн майбутньої роботи, котрі допоможуть далі використовуючи код виконати візуальну частину проекту. Також розписано яким чином працює фреймворк Angular.

РОЗДІЛ 3. РЕАЛІЗАЦІЯ ПРОГРАМНО - ІНСТРУМЕНТАЛЬНОГО ЗАСОБУ ОБЛІКУ СЛУХАЧІВ НАВЧАЛЬНОГО ЦЕНТРУ

3.1 Підготовка до розробки програми

Для того щоби розпочати роботу треба мати встановлений Google Chrome та VisualStudio. Вони завантажуються на встановлюються в стандартному порядку. Далі потрібно завантажити Node.JS.

Далі установка Node.js на Windows Для початку необхідно встановити правильну консоль для Windows. Після цього можна переходити до установки Node.js. Для Windows є тільки один спосіб. Необхідно перейти на офіційний сайт <https://nodejs.org> і на головній сторінці завантажити останню стабільну версію. Після завантаження інсталятор слід запустити і встановити Node.js, як будь-яку іншу програму. Щоб переконатись що все встановилося. Для цього в cmdер перевіримо версію Node.js за допомогою команди `node -v` і `npm -v` [7]. Ця команда дозволить нам побачити версія яка встановлена на нашому персональному комп'ютері.

Для роботи з Angular необхідно встановити сервер Node.js і пакетний менеджер npm, якщо вони відсутні на робочій машині. Для установки можна використовувати програму установки node.js. Разом з сервером вона також встановить і npm. При цьому особливого якогось знання для роботи з NodeJS і npm не потрібно. Установка Angular CLI. Для компіляції додатка ми будемо використовувати інфраструктуру Angular CLI. Angular CLI спрощує створення додатка, його компіляцію. Angular CLI поширюється як пакет npm, тому для його використання його необхідно спочатку встановити [9]. Встановлюємо фреймворк відповідно до інструкції на офіційному сайті розробників.

І тепер, коли все готово, ми можемо запустити розробку проекту. Для цього в командному рядку (терміналі) перейдемо до папки проекту за допомогою команди `cd` і потім виконаємо команду `ng serve --open` (Рис 3.1.).

```
Windows PowerShell
Microsoft Windows [Version 10.0.19042.572]
(c) Корпорация Майкрософт (Microsoft Corporation), 2020. Все права защищены.

D:\Папкаспанками\Учеба\4 курс\ДИПЛОМ\diplom2\diplom2\diplom>ng serve --open
Your global Angular CLI version (11.2.6) is greater than your local version (11.2.5). The local Angular CLI version is used.

To disable this warning use "ng config -g cli.warnings.versionMismatch false".
✓ Browser application bundle generation complete.

Initial Chunk Files | Names | Size
vendor.js | vendor | 2.98 MB
polyfills.js | polyfills | 474.75 kB
styles.css, styles.js | styles | 346.91 kB
main.js | main | 52.54 kB
runtime.js | runtime | 6.15 kB
| Initial Total | 3.84 MB

Build at: 2021-04-19T11:16:01.536Z - Hash: e3aa69933fdc9da22760 - Time: 19562ms

** Angular Live Development Server is listening on localhost:4200, open your browser on http://localhost:4200/ **

✓ Compiled successfully.
```

Рис. 3.1. Консоль, яка показує що проект відкритий у браузері.

Консоль проінформує нас, про створені системою файли та їх розміри. Також вона надає нам адресу, яку використовує працюючий тестовий сервер, за замовченням вона буде - "http: // localhost: 4200 /".

3.2 Створення майбутнього продукту

Для початку необхідно створити сторінку через яку користувач буде мати змогу перейти до власного кабінету. Вона повинна працювати таким чином, щоби при внесенні невірних даних, вчитель не отримав доступ до даних. Html – це база будь-якої веб-сторінки, її наповнення у вигляді тексту та зображень.

Код буде виглядати наступним чином: (Див. Рис.3.2).

```
1 <div class="login">
2   <div class="logo-block">
3     
4   </div>
5   <div class="form-block">
6     <div class="inputs">
7       <input id="email" type="text" placeholder="Електронна адреса" [(ngModel)]="email">
8       <input id="password" type="password" placeholder="Пароль" [(ngModel)]="password">
9     </div>
10    <button id="login-btn" (click)="login()">Увійти</button>
11  </div>
12</div>
```

Рис.3.2. Html першої веб-сторінки.

Елемент `<div>` це блоковий елемент який призначений для виділення фрагмента документа з метою зміни виду вмісту.

Він буде впливати на контент тільки після того як буде стилізований за допомогою CSS. Тег `<div>` використовують для надання стилів та угруповання блокових елементів. Використовуючи атрибути `class` або `id` можна угруповувати контент та стилізувати його, також позначати розділ документа, написаний на різних мовах [11].

```

```

Дана команда потрібна для того, щоби додати зображення до веб-сторінки.

Елементи SVG можна змінювати за допомогою атрибутів, які задають докладні відомості про те, яким чином елемент повинен оброблятися.

Assets означає набори файлів, які використовуються на HTML-сторінці - це файли CSS, JavaScript, різні шрифти і зображення. Нерідко терміном assets називають тільки набори css- і js-файлів.

Атрибут `<id>` - це унікальне ім'я елемента, яке використовується для зміни його стилю і звернення до нього через скрипти [12].

В той час як `<class>` може використовуватись для декількох елементів, атрибут `<id>` надають тільки для одного.

Атрибут `<input>` призначений для створення різних кнопок, текстових полів, прапорців, перемикачів [13].

Placeholder виводить текст в текстовій скринці який зникає при отриманні фокусу. Тобто це текст який зникає коли користувач натискає на відповідне поле, та починає вводити дані [14].

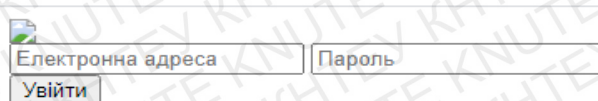
Type - Повідомляє браузеру, до якого типу належить елемент форми [15].

`[(ngModel)]` Ця директива вказує Angular, що необхідно двостороннє зв'язування даних. Що завжди застосовується в директивах `input`, `select`, `textarea` [16].

Button – це кнопка на яку користувач може натиснути.

Click – процес який відбувається після натискання кнопки користувачем.

Результатом всіх цих дій буде веб-сторінка з наступним виглядом(Рис 3.3).



The image shows a basic web form layout. It consists of two adjacent text input fields. The first field is labeled 'Електронна адреса' (Electronic address) and the second is labeled 'Пароль' (Password). Below the first field is a button labeled 'Увійти' (Log in). The entire form is enclosed in a thin rectangular border.

Рис.3.3. Вигляд першої веб-сторінки без CSS.

Тепер необхідно підключити CSS до проекту, щоб зовні вона виглядала більш знайомою для користувача.

```

login.component.scss  X
1  .host {
2      background: linear-gradient(90deg, #32327A 25%, #4D4D9B 50%, #32327A 75%);
3      display: flex;
4      justify-content: center;
5      height: 100%;
6      width: 100%;
7  }
8
9  .login {
10     height: 80%;
11     width: fit-content;
12     display: flex;
13     flex-direction: column;
14     justify-content: center;
15     font-family: Arial, sans-serif;
16 }
17
18 .login .logo-block {
19     height: 200px;
20     margin-bottom: 20px;
21     display: flex;
22     justify-content: center;
23 }
24
25 .login .logo-block img {
26     width: 80%;
27 }
28
29 .login .form-block {
30     width: 300px;
31     height: 200px;
32     padding: 30px;
33     background-color: white;
34     border-radius: 1%;
35     display: flex;
36     flex-direction: column;
37     justify-content: space-around;
38 }
39
40
41 .login .form-block .inputs {
42     width: 100%;
43     margin-bottom: 15px;
44 }
45
46 .login .form-block .inputs input {
47     width: 100%;
48     height: 40px;
49     margin-bottom: 5px;
50     border-radius: 5px;
51     padding: 5px;
52     border: 1px solid lightgray;
53     text-decoration: none;
54     outline: none;
55 }
56
57
58 .login .form-block .inputs input:focus {
59     border-color: #4D4D9B;
60     text-decoration: none;
61     outline: none;
62 }
63
64 .login .form-block .inputs input:focus::placeholder {
65     color: #4D4D9B;
66 }
67
68 .login .form-block button {
69     margin-bottom: 5px;
70     border-radius: 5px;
71     padding: 5px;
72     text-decoration: none;
73     outline: none;
74     border: 2px solid transparent;
75     background-color: #838BC5;
76     color: white;
77     font-size: 18px;
78 }
79
80 .login .form-block button:focus {
81     border: 2px solid #4D4D9B;
82 }
83
84 .login .form-block button:active {
85     background-color: #5966cc;
86 }
87
88 .login .form-block button:hover {
89     cursor: pointer;
90 }
91
92

```

Рис. 3.4. Код CSS першої веб-сторінки.

На зображенні під номером 4 ми маємо змогу побачити скріншот програми яка містить CSS код першої сторінки проекту. CSS відповідає за зовнішній вигляд веб-сторінки, надає кольори і стилі. Іншими словами Html – це зміст, а CSS – вигляд.

Далі розберемося що виконує кожна з наведених на рисунку функцій. Функція-псевдоклас: `host` вибираєносія `shadow DOM`-а, який містить CSS, який використовується всередині - але тільки якщо селектор, переданий як параметр функції, відповідає `shadow` хосту. Найбільш поширений спосіб використання його - поставити певний клас тільки на певні екземпляри призначених для користувача елементів, а потім передати відповідний класовий селектор як аргумент функції.

Тіньовий DOM («Shadow DOM») використовується для інкапсуляції. Background – фон даної сторінки. Далі був доданий градієнт використовуючи «`linear-gradient`», та додано такі значення як розмір та колір.

Наступним кроком була прописана команда «`display: flex`». Так у дісплея з'явиться властивість `flex`-контекст, яка в подальшому дозволить ним управляти набагато простіше, ніж з використанням стандартного CSS. Далі іде властивість `justify-content`, вона визначає, як браузер розподіляє простір між та навколо елементів контенту уздовж головної осі `flex` контейнера, або уздовж малої осі `grid` контейнера.

Оскільки за замовченням браузер розташовує контент з лівого боку, необхідно було додати функцію «`center`». Таким чином контент буде розташований по середині.

Далі йде функції `height` і `width`. CSS атрибут `height` і `width` дозволять позначати висоту та ширину елемента. За замовчуванням, властивість визначає висоту і ширину внутрішньої області. Після внесення даних всі створенні блоки будуть розташовані вздовж всього вікна браузера.

Наступним кроком працюємо з `<div class="login">`. Надаємо висоту 80% відсотків, та внутрішню мінімальну ширину. Дісплей також залишається гнучким.

Властивість `flex-direction` задає напрямок основних осей в контейнері і тим самим визначає положення флексів в контейнері. Використовуючи властивість `justify-content: center`, даємо браузеру зрозуміти що цей блок також розташовується по центру вікна.

Переходимо до класу `logo-block`. Ширина буде 200 пікселів, величина відступу від нижнього краю елемента - 20 пікселів. Залишаємо функцію гнучкості дисплею, для того щоб блок мав змогу налаштовуватись під розмір вікна браузера. `Justify-content` залишається по центру. Зображення логотипу робимо 80% від оригінального розміру.

Далі переходимо до блоку `form-block`. Ширина буде 300 пікселів, висота – 200 пікселів, та відступ – 30 пікселів. Робимо даний блок білого кольору та закруглюємо кутки на 1%. `Display: flex`; та `flex-direction: column`, залишаються таким ж самим як і у попередніх блоках. Також буде додана функція `space-around` до `justify-content`, щоб між елементами була відстань.

Для блоку `inputs` встановлюємо величину відступу від нижнього краю елемента – 15 пікселів, та ширину 100%.

Наступним чином працюємо з `.login .form-block .inputs input`. Ширина – 100%, висота - 40 пікселів. Відступ від нижнього краю робимо 5 пікселів, та закруглюємо кути нашого блоку на 5 пікселів, відступ з лівого боку також 5 пікселів. Змінюємо товщину рамки на 1 піксель, та додаємо колір під назвою `solid lightgray`.

`Text-decoration` створює оформлення тексту у вигляді його підкреслення, лінії над текстом та миготіння. Одночасно можна застосувати більш одного стилю, перераховуючи значення через пробіл. У нашому проєкті відсутня декорація тексту тому прописуємо – `none`.

`Outline` – це властивість, яка встановлює товщину, стиль і колір зовнішнього кордону на всіх чотирьох сторонах елемента. На відміну від лінії, що задається через `border`, властивість `outline` не впливає на положення блоку ширину блоку. Прописуємо `none`, тому що нам не потрібна дана функція.....

Переходимо до `.login .form-block .inputs input:focus`. CSS псевдоклас: `focus` застосовується, коли елемент отримує фокус. Зазвичай він активується під натисканням миші користувача або при виборі елемента за допомогою клавіші "tab" на клавіатурі. Робимо колір рамки `#4D4D9B`.

Далі робимо текст всередині поля форми, який вичерпується при отриманні фокуса кольору(placeholder) з наступним значенням - `#4D4D9B`.

Починаємо працювати з кнопками. Робимо нижній відступ на 5 пікселів, закруглення рамки на 5 пікселів, та відступи також на 5 пікселів. Додаємо колір для фону, та задаємо розмір шрифту. Додаємо функцію яка буде фарбувати колір фону в інший коли користувач натисне на кнопку: `.login .form-block button:active { background-color: #5966cc`.

Також для реалізму була додана функція яка перетворює курсор у руку з вказівним пальцем при наведенні на кнопку - `cursor: pointer`.

Завдяки роботі з CSS наша сторінка набула вигляд схожий до веб-сторінок які ми звикли бачити в інтернеті. Результат виконаних дій можна побачити на рис 3.5.



Рис. 3.5. Вигляд створеної першої сторінки проекту.

Переходимо до розпису TypeScript коду, та підключаємо Html, CSS і TS до Angular. Також зв'язуємо цей файл з можливістю обирати групи. (див рис.3.6)

```

1  import { Component, OnInit } from '@angular/core';
2  import { UserService } from 'src/app/services/user.service';
3
4  @Component({
5    selector: 'app-main',
6    templateUrl: './main.component.html',
7    styleUrls: ['./main.component.scss']
8  })
9  export class MainComponent implements OnInit {
10    selectedGroup: string;

```

Рис. 3.6. Код для підключення Angular.

Тепер створюємо конкретні групи, будемо вважати що всього існує три групи, та два викладача. Кожен з викладачів має доступ тільки до журналу своєї групи, також у них є одна спільна група до журналу якої доступ мають обидва вчителів. Тобто вчитель Дьяченко Дар'я Володимирівна має доступ до груп під номером 1 та 3, А вчитель Кравчук Антон Валерійович може відкривати список груп 2 та 3. З цього можна зробити висновок що обидва викладача мають спільний доступ до журналу групи №3. Код можна подивитися на рисунку 3.7.

```

accounts = [
  {
    name: "Дьяченко Дар'я Володимирівна",
    email: 'viciadach@gmail.com',
    password: '1234',
    groups: [
      'group1',
      'group3'
    ]
  },
  {
    name: "Кравчук Антон Валерійович",
    email: 'viciadach2@gmail.com',
    password: '4321',
    groups: [
      'group2',
      'group3'
    ]
  },
];

```

Рис. 3.7. Аккаунти вчителів з ПІБ, поштою та паролем.

Прописуємо в системі функцію яка дозволить користувачу вхід до власного кабінету тільки якщо він вкажить дані, які є у системі.(рис.3.8) Тобто вірне поєднання паролю та пошти. Після введення даних користувачем система почне звіряти ці дані з тими, що є у базі. Спочатку звірить імейл, якщо система знайшла таку скриньку, то вона почне далі звіряти пароль, і тільки якщо всі дані вірні, то система надає доступ.

```
39
40     constructor(private router: Router, private userService: UserService) { }
41
42     ngOnInit(): void {
43     }
44
45
46     login(): void {
47         const acc = this.accounts.find(acc => acc.email === this.email && acc.password === this.password);
48
49         if (acc) {
50             this.userService.user = acc;
51             this.router.navigate(['main'])
52         }
53     }
54 }
55
```

Рис. 3.8. Частина коду системи відповідності пошти та паролю.

На цьому закінчуємо роботу с TS на даний момент, та переходимо до створення другої веб-сторінки нашого проекту. Вона буде містити класний журнал у вигляді таблиці, ПІБ та пошту викладача, логотип і назву підприємства, та декілька кнопок. Починаємо з файлу HTML.

У перший блок додаємо зображення логотипу школи.(див рис. 3.9.)

```
main.component.html  main.component.ts  login.component.ts*  login.component.scss
1  <div class="main">
2    <header>
3      <div class="header-logo">
4        
5      </div>
6  </div>
```

Рис. 3.9. Код котрий додає зображення логотипу.

Далі додаємо зображення для кнопок меню у верхній частині екрану.(Рис.3.10)

```

7   <div class="header-buttons">
8     <div id="btn-home">
9       
10    </div>
11    <div id="btn-groups">
12      
13    </div>
14    <div id="btn-mails">
15      
16    </div>
17    <div id="btn-settings" (click)="clearStorage()">
18      
19    </div>
20  </div>
21

```

Рис.3.10. Код котрий додає зображення кнопок.

Наступним чином були додані пошта та ПІБ вчителя котрі у майбутньому будуть відображатись у верхньому правому кутку сайту. Також добавлена іконка користувача.(див рис.3.11)

```

<div class="header-info">
  <div class="text">
    <div id="name">{{userService.user?.name || ''}}</div>
    <div id="email">{{userService.user?.email || ''}}</div>
  </div>
  
</div>
</header>

```

Рис.3.11. Функція яка визиває відображення ПІБ та пошти на веб-сторінці.

Далі створюємо табличку для майбутнього виставлення оцінок.

Створюємо функцію яка дозволить обирати групу з існуючого списку, та вносити зміни до таблиці. Також додаємо кнопку «Зберегти», для того щоб після натиску на кнопку дані які ввів користувач зберігались у системі. Тег `<select>` дозволяє створити елемент інтерфейсу у вигляді списку, а також список з одним або множинним вибором, а директива `ngIf` дозволяє видалити або, навпаки, додати елемент за певної умови. Далі йде директива `ngFor` що дозволяє перебрати в шаблоні елементи масиву. Властивість `target` інтерфейсу Event є посиланням на об'єкт, який був ініціатором події. Ці дії можна побачити на рисунку 3.12.

```

<div class="content">
  <div class="workspace">
    <div class="table-head">
      <div class="btn-wrapper">
        <select name="groups" id="groups" *ngIf="groupNames" (change)="changeGroup($event.target.value)">
          <option>-</option>
          <option *ngFor="let group of userService.user?.groups" [value]="group">{{groups[group].name}}</option>
        </select>
      </div>
      <h2 id="group-name">{{groups[selectedGroup]?.name || ''}}</h2>
      <div class="btn-wrapper">
        <button class="save-btn" (click)="saveData()">Зберегти</button>
      </div>
    </div>
  </div>
</div>

```

Рис. 3.12. Таблиця для вибору групи

У результаті цих дій ми створили табличку яку користувач буде використовувати для вибору необхідної для нього групи.

Тепер потрібно створити таблицю яку викладач буде використовувати для запису оцінок та проставлення відвідуваності (рис.3.13).

```

<div class="table-wrapper">
  <table *ngIf="selectedGroup">
    <tr>
      <th>#</th>
      <th>ПІБ</th>
      <th *ngFor="let d of days">{{d}}</th>
      <th>Екзамен</th>
    </tr>
    <tr *ngFor="let student of groups[selectedGroup]?.data; let i = index">
      <td>{{i}}</td>
      <td>{{student.name}}</td>
      <td *ngFor="let d of days">
        <div class="td-input">
          <input type="text" [(ngModel)]="student.marks[d]">
        </div>
      </td>
      <td>
        <select name="exam" id="exam" [(ngModel)]="student.exam">
          <option [value]="false">Не зданий</option>
          <option [value]="true">Зданий</option>
        </select>
      </td>
    </tr>
  </table>
</div>
</div>
</div>

```

Рис. 3.13. Таблиця для заповнення відвідування і успішності.

Тут ми створили поле в якому записані ПІБ всіх студентів, поле для запису дати проведення заняття, та комірку для виставлення екзамену.

Екзамен може бути «Зданим», або «Не зданий».

Переходимо до другої сторінки, для якої потрібно прописати CSS, щоб вона отримала оформлений зовнішній вигляд.

Починаємо роботу з заднього фону веб-сторінки, робимо його на 100% розміру відносно сторінки браузеру, да додаємо колір під номером - #E1E1E2.(

Рис.3.14.)

```
1  
2 .main {  
3     height: 100%;  
4     background-color: #E1E1E2;  
5 }  
6
```

Рис.3.14. Робота з основним фоном.

Далі працюємо з верхнім полем, на якому містяться ПІБ викладача, та його фото, назва школи з логотипом, та декілька кнопок. Спочатку робимо цей об'єкт у ширину 100% і в висоту 10% від розміру екрану монітору, далі додаємо колір фону - #4D4D9B. Додаємо для дисплея функцію – flex. Прописуємо «justify-content: space-around» він рівномірно розподіляє всі елементи по ширині flex-блоку.(рис.3.15) Використовуємо властивість align-items яка вирівнює flex-елементи поточної flex-лінії таким же чином, як і justify-content, але в перпендикулярному напрямку.

```
9 .main header {  
10     height: 10%;  
11     width: 100%;  
12     background-color: #4D4D9B;  
13     display: flex;  
14     justify-content: space-around;  
15     align-items: center;  
16 }
```

Рис.3.15. Код зовнішнього вигляду верхнього поля.

Наступним кроком додаємо властивості у блок який містить у собі логотип (Рис.3.16.).

```
.main header .header-logo {  
    height: auto;  
    width: 200px;  
}
```

Рис.3.16. Властивості для блоку з логотипом

Переходимо до блоку у якому розташовані декілька верхніх кнопок.

Рівномірно розподіляємо всі елементи по ширині флекс-блоку, перший елемент спочатку, останній в кінці (рис.3.17).

```
.main header .header-buttons {  
  height: 100%;  
  display: flex;  
  justify-content: space-between;  
  align-items: center;  
}
```

Рис.3.17 Розподіл елементів для блоку з кнопками.

Далі працюємо з зображенням цих кнопок (рис.3.18).

```
.main header .header-buttons img {  
  margin: 0 10px;  
  width: 40px;  
  border: 2px solid white;  
  border-radius: 50%;  
  cursor: pointer;  
}
```

Рис.3.18. Робота з зображенням кнопок.

У результаті верхнє поле веб-сторінки буде виглядати як на зображенні рис 3.19.



Рис. 3.19. Вигляд верхньої панелі веб-сторінки.

Працюємо з правою частиною екрану де розташоване ПІБ викладача, його пошта, та зображення. Додаємо властивості для тексту у вигляді зміни шрифту на Arial, Helvetica, та sans-serif (рис.3.20).

```
.main header .header-info {  
  width: fit-content;  
  height: 100%;  
  display: flex;  
  justify-content: center;  
  align-items: center;  
  font-family: Arial, Helvetica, sans-serif;  
}
```

Рис.3.20. Робота з блоком де розташовано ПІБ викладача.

Переходимо до ім'я вчителя у верхньому правому кутку. Змінюємо колір шрифту з чорного на білий, та змінюємо розмір. Робимо відображення тексту з правої сторони(див рис.3.21).

```
.main header .header-info #name {
  width: 150px;
  text-align: right;
  font-size: 1.1rem;
  color: white;
}
```

Рис.3.21. Робота з ПІБ викладача.

Далі виконуємо зміни у блоці з назвою персональної пошти (рис.3.22). Тобто змінюємо розмір, колір, та даємо йому властивість відображатись з правої сторони, а не з лівої як це прописано по замовченню.

```
.main header .header-info #email {
  font-size: 0.8rem;
  text-align: right;
  color: #CDC8FF;
}
```

Рис.3.22. Блок персональної пошти.

Додаємо властивості для іконки де у майбутньому буде відображатись фотографія користувача.

```
.main header .header-info img {
  width: 50px;
  margin-left: 5px;
}
```

Працюємо з текстом яким надруковано ПІБ студентів (див рис 3.23).

Назначаємо йому розмір, та шрифт.

```
.main .content .workspace {
  width: 70%;
  height: 100%;
  font-family: Arial, Helvetica, sans-serif;
}
```

Рис.3.23. Робота с текстом яким надруковано ПІБ студентів.

Працюємо з блоком вибору групи зі списку так, як зображено на рисунку 3.24.

```

    .main .content .workspace .table-head {
      margin: 5px 0;
      display: flex;
      justify-content: space-between;
    }

    .main .content .workspace .table-wrapper {
      display: flex;
      justify-content: center;
    }

```

Рис.3.24. Робота з блоком вибору групи зі списку.

Далі переходимо до оформлення вигляду списку груп. Робимо колір верхньої строки відтінку - #838BC5, а колір фону звичайним білим. Редагуємо розташування, розмір тексту, відступи. Закруглюємо кути у полях які мають назву «Зданий» і «Не Зданий» (рис.3.25).

```

#groups {
  background-color: #838BC5;
  color: white;
  outline: none;
  border: none;
  text-decoration: none;
  margin-bottom: 5px;
  border-radius: 5px;
  font-size: 18px;
  padding: 7px;
}

```

Рис.3.25. Оформлення вигляду списку груп.

Додаємо значення коли користувач натискає на список груп, та робимо шрифт яким надрукований номер групи розміру 40 пікселів (рис 3.26).

```

#groups:active {
  border: none;
  text-decoration: none;
  outline: none;
}

#group-name {
  font-size: 40px;
}

```

Рис.3.26. Зміна зовнішнього вигляду списку груп при натисканні.

Робимо фон самої таблиці білим кольором, та призначаємо відступи. Далі надаємо верхньому рядку таблиці світло-синій відтінок. Додаємо значення для осередків таблиці.

Переходимо до роботи з контуром таблиці(див рис.3.27). Заповнюємо його чорним кольором. Працюємо з полями які називаються «Зданий» і «Не Зданий».

Колір вікна робимо світло-синього кольору, а текст у центрі блока змінюємо на білий. Міняємо відступи, розмір шрифту, та закруглюємо рамочку.

Редагуємо розмір шрифту та самої таблиці відносно вікна веб-сторінки.

```
table {
  border: 1px solid black;
  border-spacing: 0 0;

  select {
    background-color: #838BC5;
    color: white;
    outline: none;
    border: none;
    text-decoration: none;
    margin-bottom: 5px;
    border-radius: 5px;
    font-size: 18px;
    padding: 7px;
  }

  .td-input {
    display: flex;
    justify-content: center;
    width: fit-content;

    input {
      text-decoration: none;
      outline: none;
      border: none;
      text-align: center;
      font-size: 18px;
      width: 50px;
    }
  }
}
```

Рис.3.27. Робота з зовнішнім виглядом таблиці журналу.

Далі працюємо з кнопкою «Зберегти» (рис.3.28). Додаємо колір фону, ширину рамки, відступи, та колір тексту. Створюємо рамку яка буде змінювати колір при наведенні на неї курсору. Додаємо функцію для кнопки яка буде змінювати колір при натисканні на неї користувачем.

```
button.save-btn {  
  margin-bottom: 5px;  
  border-radius: 5px;  
  padding: 5px;  
  text-decoration: none;  
  outline: none;  
  border: 2px solid transparent;  
  background-color: #838BC5;  
  color: white;  
  font-size: 18px;  
}  
  
button.save-btn:focus {  
  border: 2px solid #4D4D9B;  
}  
  
button.save-btn:active {  
  background-color: #5966cc;  
}
```

Рис.3.28. Робота з кнопкою «Зберегти»

Працюємо з другою сторінкою TS.

Спочатку підключає наш файл до Angular. Потім працюємо з таблицею журналу, який буде відображатись на другій веб-сторінці. За замовченням створюємо однакові дати для всіх груп (Рис.3.29).

```
1 import { Component, OnInit } from '@angular/core';  
2 import { UserService } from 'src/app/services/user.service';  
3  
4 @Component({  
5   selector: 'app-main',  
6   templateUrl: './main.component.html',  
7   styleUrls: ['./main.component.scss']  
8 })  
9 export class MainComponent implements OnInit {  
10   selectedGroup: string;  
11  
12   days = ["05.09", "12.09", "19.09", "26.09", "05.10", "12.10"];  
13   groups: any;  
14 }
```

Рис.3.29. Код Angular та фіксовані дати для всіх груп.

Далі працюємо з конкретною групою. Додаємо у список ПІБ студентів.

У даному коді ми можемо побачити фіксовані дати в які будуть проводитися заняття, та місце у якому у майбутньому будуть записуватись бали які студент набрав за заняття. Також можемо побачити функцію exam, якщо стоїть false то студент не склав екзамен, якщо true – екзамен складений (див Рис.3.30).

```
reservedGroups = {  
  group1: {  
    name: 'Група №1',  
    data: [  
      {  
        name: "Рибка Максим Сергійович",  
        marks: {  
          "05.09": "",  
          "19.09": "",  
          "26.09": "",  
          "12.10": ""  
        },  
        exam: true  
      }  
    ]  
  },  
}
```

Рис.3.30. Код першого студента першої групи.

На рисунку 3.31 ми бачимо приклад коду для одного студента, в дані роботі ми маємо три групи по п'ять студентів, відповідно всього буде п'ятнадцять чоловік. Далі йде код котрий дозволить викладачеві обрати необхідну йому групу.

```

get groupNames(): string[] {
    return Object.keys(this.groups);
}

constructor(public userService: UserService) { }

ngOnInit(): void {
    this.loadData()
}

changeGroup(value) {
    this.selectedGroup = value;
}

saveData(): void {
    localStorage.setItem('groups', JSON.stringify(this.groups))
}

loadData(): void {
    const data = localStorage.getItem('groups');

    if (data) {
        this.groups = JSON.parse(data);
    } else {
        this.groups = this.reservedGroups;
    }
}

clearStorage(): void {
    localStorage.clear()
}
}

```

Рис.3.31. Код який дозволяє обирати групи

3.3 Тестування засобу обліку

На даному етапі ми завершуємо роботу над розробкою програми и переходимо до тестування. Уявимо що користувач вже перейшов з головної сторінки сайту до сторінки авторизації. Сторінка авторизації виглядає наступним чином (див рис.3.32).



Рис.3.32. Сторінка авторизації.

Тут він має змогу побачити логотип навчального закладу, назву. Далі йде таблиця з трьома блоками, з якими користувач може взаємодіяти. Уявимо що користувач, а якщо конкретніше – викладач вже зареєстрований у системі. Коли він натискає на необхідне йому поле для заповнення воно підсвічується синім кольором та дає йому змогу зрозуміти у яке поле він зараз буде вводити дані. Порівняння звичайного і активного поля можна побачити на рисунку 3.33

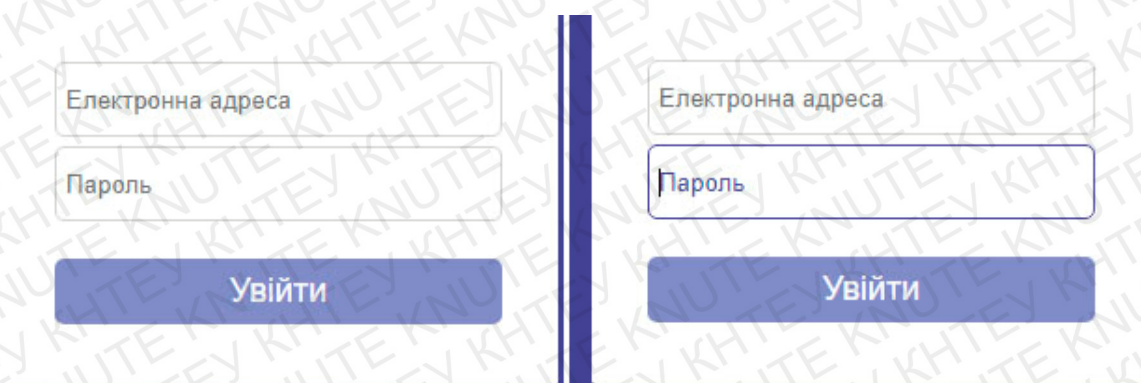


Рис.3.33. Зміна блоку введення паролю після натискання користувачем.

У перше поле вводу він прописує свою власну пошту, у наступне прописує пароль від акаунту, а далі натискає кнопку «Увійти» (див рис.3.34).

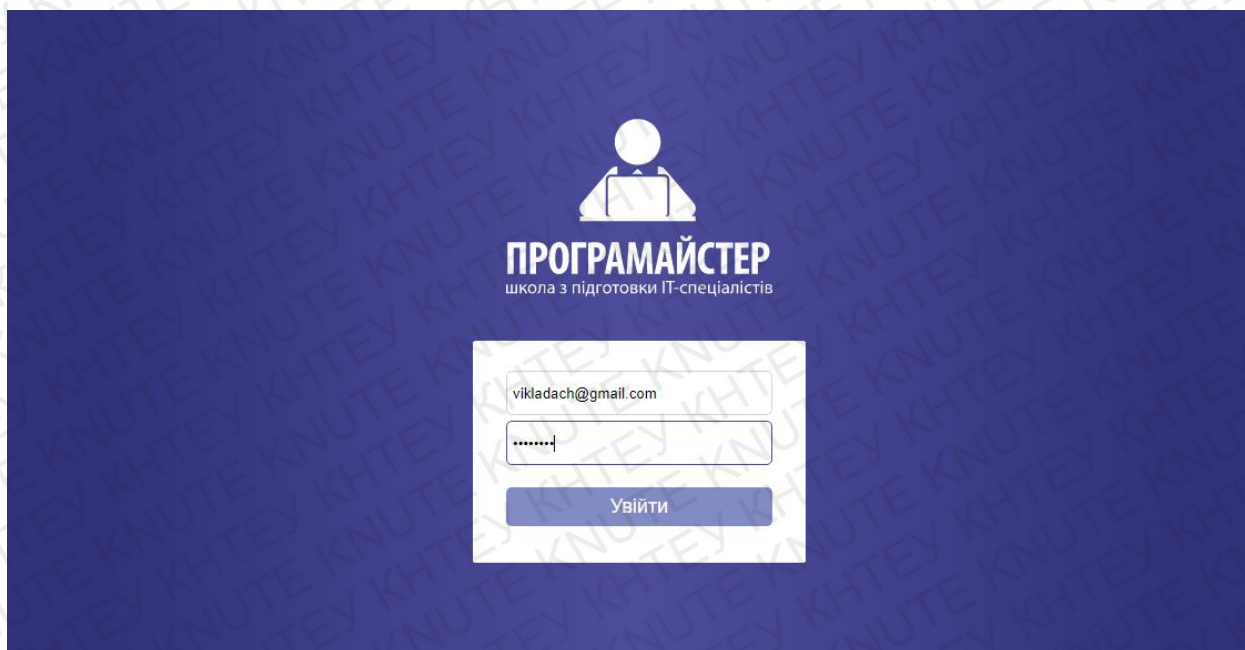


Рис.3.34. Вчитель ввів дані для авторизації.

Після того як кнопка змінила колір вчитель потрапляє на наступну сторінку(див рис.3.35).

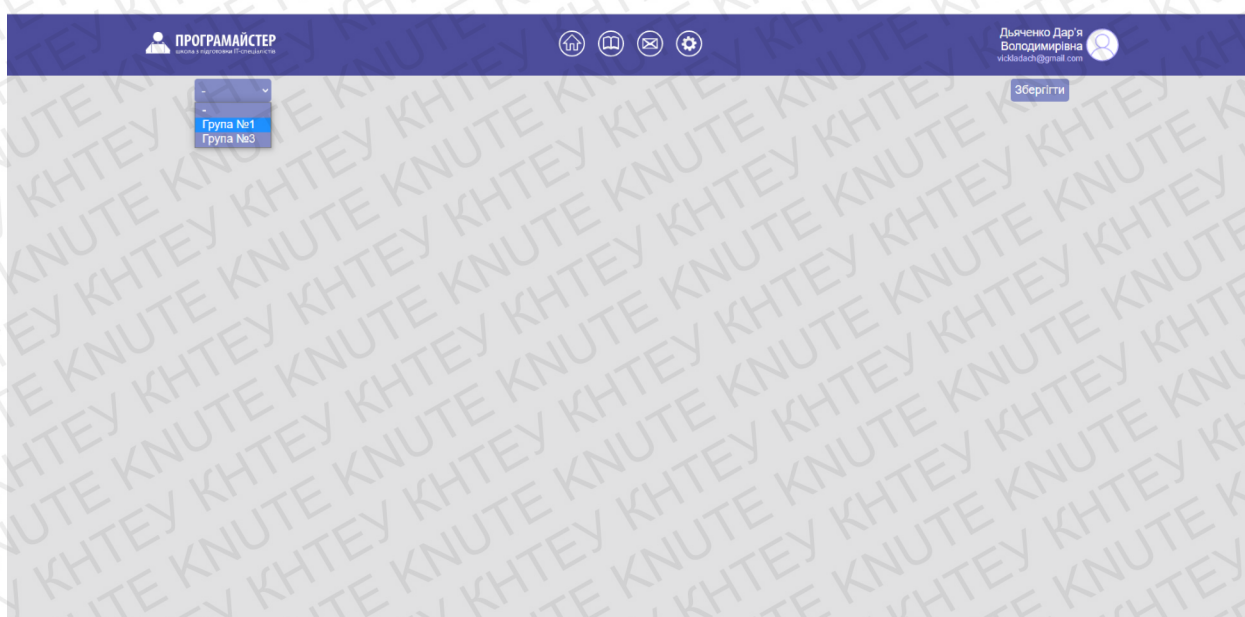


Рис.3.35. Головна сторінка журналу відвідуваності.

На дані сторінці ми бачимо ПІБ с поштою авторизованого користувача, чотири кнопки які розташовуються у центрі верхньої панелі, та з лівого боку бачимо назву та логотип підприємства. При заходженні у власний кабінет працює друга кнопка зі списку, яка має вигляд журналу. Але журнал за замовченням буде закритим, його можна легко відкрити використовуючи список під назвою навчального закладу. На зображенні 3. Ми можемо побачити що

користувач під ім'ям Дьяченко Дар'я Володимирівна має доступ до редагування даних тільки першої та третьої групи. Оскільки кожен з викладачів має свою групу та одну спільну, відповідно інший вчитель буде мати доступ до другої та третьої груп.

Припустимо що вчитель хоче відреагувати дані. Він відриває журнал першої групи (див рис.3.36). Зараз напроти ПІБ студентів пусті клітинки, але уявимо що пройшов навчальний семестр і бали вже виставлені(рис.3.37), з цих балів ми можемо зробити висновок що п'ятий студент погано відвідував заняття, йому не вистачило знань щоби здати екзамен, відповідно у стовпці «Екзамен» йому виставили «Не зданий». Також для того щоби внесені зміни у журналі збереглися вчителю потрібно натиснути кнопку «Зберегти».

На наступним зображеннях ми можемо побачити заповнений журнал другої та третьої груп (рис.3.38-3.39).

#	ПІБ	05.09	12.09	19.09	26.09	05.10	12.10	Екзамен
1	Крамаренко Михайло Сергійович							Зданий
2	Серода Олександр Володимирович							Зданий
3	Васильчук Анна Михайлівна							Зданий
4	Коваленко Олена Вікторівна							Зданий
5	Антоненко Денис Денисович							Не зданий

Рис.3.36. Журнал першої групи з пустими клітинками.

ПРОГРАМАЙСТЕР								Дяченко Дар'я Володимирівна vdiadachn@gmail.com	
Група №1		Група №1						Зберігти	
#	ПІБ	05.09	12.09	19.09	26.09	05.10	12.10	Екзамен	
1	Крамаренко Михайло Сергійович	15	15	15	15	15	15	Зданий	
2	Серета Олександр Володимирович	15	15	15	15	15	15	Зданий	
3	Васильчук Анна Михайлівна	15	10	15	15	15	15	Зданий	
4	Коваленко Олена Вікторівна	15	15	15	15	15	15	Зданий	
5	Антоненко Денис Денисович	15	н	н	15	н	н	Не зданий	

Рис.3.37. Заповнений першим вчителем журнал першої групи.

ПРОГРАМАЙСТЕР								Кравчук Антон Валерійович vdiadachn@gmail.com	
Група №2		Група №2						Зберігти	
#	ПІБ	05.09	12.09	19.09	26.09	05.10	12.10	Екзамен	
1	Мороз Марк Сергійович	15	15	15	15	15	15	Зданий	
2	Швець Дарина Олегівна	15	15	15	15	15	15	Зданий	
3	Карпенко Ірина Тимофіївна	15	10	15	15	15	15	Зданий	
4	Гончар Іван Миколайович	15	10	15	15	15	15	Зданий	
5	Вояк Володимир Валерійович	15	10	15	15	15	15	Зданий	

Рис.3.38. Заповнений другим вчителем журнал другої групи.

ПРОГРАМАЙСТЕР								Дяченко Дар'я Володимирівна vdiadach@gmail.com	
Група №3		Група №3						Зберігти	
#	ПІБ	05.09	12.09	19.09	26.09	05.10	12.10	Екзамен	
1	Панасюк Анастасія Віталівна	10	10	н	н	н	н	Не зданий	
2	Журба Микола Андрійович	15	10	15	15	15	15	Зданий	
3	Шевченко Діана Олександрівна	5	10	10	5	н	15	Не зданий	
4	Тарасенко Ольга Петрівна	15	10	н	15	15	15	Зданий	
5	Бондаренко Олексій Дмитрович	н	н	н	н	10	15	Не зданий	

Рис.3.39. Заповнений першим вчителем журнал третьої групи.

Висновки до розділу

Програмно - інструментальний засіб обліку слухачів навчального центру був успішно створений, та пройшов тестування за результатами якого помилок у створеному проекті не виявлено. Такий додаток можна використовувати не тільки для навчальних центрів, а також для шкіл, та закладів вищої освіти. Також дивлячись на журнал відвідуваності можна зробити висновок що у першій групі екзамен не склала одна людина, у другій екзамен отримали всі студенти, та в третій екзамен не змогли здати три студенти.

ВИСНОВКИ

У випускній кваліфікаційній роботі була поставлена задача розробити програмно - інструментальний засіб обліку слухачів навчального центру. По результатам проведеного аналізу були зроблені висновки відносно способу вирішення даної задачі. Було вирішено виконати систему обліку у вигляді додаткових веб-сторінок для офіційного сайту навчального закладу, тобто розширити можливості цього сайту. В результаті проведених досліджень були отримані такі висновки:

1. Розроблений засіб обліку слухачів навчального центру, котрий дозволить вести облік використовуючи онлайн-ресурси, та буде доступний у будь-який час за наявності інтернету. Оскільки дана розробка буде частиною офіційного сайту навчального центру, її можна в будь який час буде оновлювати та додавати нові функції, адміністрація закладу сама буде вирішувати що потрібно даній системі, а що можна прибрати. Тобто одним з плюсів буде повний контроль зі сторони навчального закладу по відношенню к системі.

2. У майбутньому систему можна розвинути та створити на базі сайту все необхідне для проведення онлайн-конференцій та занять. Тобто додати можливість заходження та перегляду оцінок та навчальних матеріалів для студентів. Таким чином весь навчальний процес буде розташований на одній платформі що буде зручно як для вчителів, так і для студентів.

3. Перехід на електронне ведення обліку буде не тільки зручно, а ще и економічно. Знизяться витрати на бумагу, також пошкодити дані котрі знаходяться на віртуальному носії набагато складніше ніж дані у паперовому вигляді. У кінці можна зробити висновок що дана розробка буде не тільки зручною, а також актуальною, так як, зараз більшість навчальних закладів проводить навчання у дистанційній формі, відповідно вони використовують онлайн-ресурси та платформи.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Тема 13. Система освіти в Україні [Електронний ресурс]. - Режим доступу: <http://studentam.net.ua/content/view/2313/85/>
2. Шкільна освіта у містах та селах: чи для всіх вона однаково доступна? [Електронний ресурс]. - Режим доступу: <https://studway.com.ua/shkilna-osvita-v-ukraini/>
3. Электронный дневник: хорошо или плохо? [Електронний ресурс]. - Режим доступу: <https://mega-talant.com/blog/elektronnyy-dnevnik-horosho-ili-ploho>
4. 3 причины, почему электронный журнал гораздо лучше бумажного [Електронний ресурс]. - Режим доступу: https://mel.fm/slovo_pedagoga/5487013-e_journal
5. Новини [Електронний ресурс]. – Режим доступу: <https://itc.ua/news/ministerstvo-obrazovaniya-ukrainy-zapustilo-pilotnyi-proekt-gosudarstvennyh-elektronnyh-dnevnikov-i-zhurnalov-dlya-shkol/>
6. Добро пожаловать в интегрированную среду разработки Visual Studio [Електронний ресурс]. - Режим доступу: <https://docs.microsoft.com/ru-ru/visualstudio/get-started/visual-studio-ide?view=vs-2019>
7. Как установить Node.js/ Попов С. [Електронний ресурс]. - Режим доступу: <https://htmlacademy.ru/blog/boost/tools/installing-nodejs>
8. Зачем изучать Node.js, или О перспективах бэкенда на JavaScript [Електронний ресурс]. - Режим доступу: <https://ru.hexlet.io/blog/posts/zachem-izuchat-node-js-ili-o-perspektivah-bekenda-na-javascript>
9. Введение в Angular [Електронний ресурс]. - Режим доступу: <https://metanit.com/web/angular2/1.1.php>
10. Как написать одностраничное приложение на Angular.JS [Електронний ресурс]. - Режим доступу: https://skillbox.ru/media/code/kak_napisat_odnostranichnoe_prilozhenie_na_angular_js/

11. Тег `<div>` [Электронный ресурс]. - Режим доступа:
<http://htmlbook.ru/html/div>
12. Атрибут `id` [Электронный ресурс]. - Режим доступа:
<http://htmlbook.ru/html/attr/id>
13. Тег `<input>` [Электронный ресурс]. - Режим доступа:
<http://htmlbook.ru/html/input>
14. Атрибут `placeholder` [Электронный ресурс]. - Режим доступа:
<http://htmlbook.ru/html/textarea/placeholder>
15. Атрибут `type` [Электронный ресурс]. - Режим доступа:
<http://htmlbook.ru/html/input/type>
16. Angular [Электронный ресурс]. - Режим доступа:
<https://angular.io/api/forms/NgModel>

ДОДАТОК

```
div class="login">
  <div class="logo-block">
    
  </div>
  <div class="form-block">
    <div class="inputs">
      <input id="email" type="text" placeholder="Електронна адреса"
[(ngModel)]="email">
      <input id="password" type="password" placeholder="Пароль"
[(ngModel)]="password">
    </div>
    <button id="login-btn" (click)="login()">Увійти</button>
  </div>
</div>

:host {
  background: linear-gradient(90deg, #32327A 25%, #4D4D9B 50%, #32327A
75%);
  display: flex;
  justify-content: center;
  height: 100%;
  width: 100%;
}

.login {
  height: 80%;
  width: fit-content;
  display: flex;
  flex-direction: column;
```

```
    justify-content: center;
    font-family: Arial, sans-serif;
}

.login .logo-block {
    height: 200px;
    margin-bottom: 20px;
    display: flex;
    justify-content: center;
}

.login .logo-block img {
    width: 80%;
}

.login .form-block {
    width: 300px;
    height: 200px;
    padding: 30px;
    background-color: white;
    border-radius: 1%;
    display: flex;
    flex-direction: column;
    justify-content: space-around;
}

.login .form-block .inputs {
    width: 100%;
    margin-bottom: 15px;
}

.login .form-block .inputs input {
    width: 100%;
    height: 40px;
    margin-bottom: 5px;
```

```
border-radius: 5px;
padding: 5px;
border: 1px solid lightgray;
text-decoration: none;
outline: none;
}

.login .form-block .inputs input:focus {
border-color: #4D4D9B;
text-decoration: none;
outline: none;
}

.login .form-block .inputs input:focus::placeholder {
color: #4D4D9B;
}

.login .form-block button {
margin-bottom: 5px;
border-radius: 5px;
padding: 5px;
text-decoration: none;
outline: none;
border: 2px solid transparent;
background-color: #838BC5;
color: white;
font-size: 18px;
}

.login .form-block button:focus {
border: 2px solid #4D4D9B;
}

.login .form-block button:active {
```

```

        background-color: #5966cc;
    }
    .login .form-block button:hover {
        cursor: pointer;
    }
    import { Component, OnInit } from '@angular/core';
    import { Router } from '@angular/router';
    import { UserService } from 'src/app/services/user.service';
    @Component({
        selector: 'app-login',
        templateUrl: './login.component.html',
        styleUrls: ['./login.component.scss']
    })
    export class LoginComponent implements OnInit {
        email: string;
        password: string; accounts = [
            {
                name: "Дьяченко Дар'я Володимирівна",
                email: 'vikladach@gmail.com',
                password: '1234',
                groups: [
                    'group1',
                    'group3'
                ]
            },
            {
                name: "Кравчук Антон Валерійович",
                email: 'vikladach2@gmail.com',
                password: '4321',
                groups: [

```

```

        'group2',
        'group3'
    ]
},
];

constructor(private router: Router, private userService: UserService) { }

ngOnInit(): void {
}

login(): void {
    const acc = this.accounts.find(acc => acc.email === this.email && acc.password
    === this.password);

    if (acc) {
        this.userService.user = acc;
        this.router.navigate(['main'])
    }
}
}

</div>

<div id="btn-mails">
    
</div>

<div id="btn-settings" (click)="clearStorage()">
    
</div>
</div>

<div class="header-info">
    <div class="text">
        <div id="name">{{userService.user?.name || ""}}</div>

```

```

    <div id="email">{{userService.user?.email || ""}}</div>
  </div>
  
</div>
</header>
<div class="content">
  <div class="workspace">
    <div class="table-head">
      <div class="btn-wrapper">
        <select      name="groups"      id="groups"      *ngIf="groupNames"
(change)="changeGroup($event.target.value)">
          <option>-</option>
          <option  *ngFor="let  group  of  userService.user?.groups"
[value]="group">{{ groups[group].name }}
          </option>
        </select>
      </div>
      <h2 id="group-name">{{ groups[selectedGroup]?.name || "" }}</h2>
      <div class="btn-wrapper">
        <button class="save-btn" (click)="saveData()">36еретт</button>
      </div>
    </div>
    <div class="table-wrapper">
      <table *ngIf="selectedGroup">
        <tr>
          <th>#</th>
          <th>ПІБ</th>
          <th *ngFor="let d of days">{{ d }}</th>
          <th>Екзамен</th>
        </tr>

```

```

<tr *ngFor="let student of groups[selectedGroup]?.data; let i = index">
  <td>{{i}}</td>
  <td>{{student.name}}</td>
  <td *ngFor="let d of days">
    <div class="td-input">
      <input type="text" [(ngModel)]="student.marks[d]">
    </div>
  </td>
  <td>
    <select name="exam" id="exam" [(ngModel)]="student.exam">
      <option [value]="false">Не зданий</option>
      <option [value]="true">Зданий</option>
    </select>
  </td>
</tr>
</table>
</div>
</div>
</div>

```

CSS друга сторінка

```

.main header .header-info #email {
  font-size: 0.8rem;
  text-align: right;
  color: #CDC8FF;
}
.main header .header-info img {
  width: 50px;
  margin-left: 5px;
}

```

```
.main .content {  
  height: 90%;  
  width: 100%;  
  display: flex;  
  justify-content: center;  
}  
  
.main .content .workspace {  
  width: 70%;  
  height: 100%;  
  font-family: Arial, Helvetica, sans-serif;  
}  
  
.main .content .workspace .table-head {  
  margin: 5px 0;  
  display: flex;  
  justify-content: space-between;  
}  
  
.main .content .workspace .table-wrapper {  
  display: flex;  
  justify-content: center;  
}  
  
#groups {  
  background-color: #838BC5;  
  color: white;  
  outline: none;  
  border: none;  
  text-decoration: none;  
  margin-bottom: 5px;  
  border-radius: 5px;  
  font-size: 18px;  
  padding: 7px;
```

```

}
#groups:active {
    border: none;
    text-decoration: none;
    outline: none;
}
#group-name {
    font-size: 40px;
}
.main .content .workspace .table-wrapper table {
    margin-top: 30px;
    background-color: white;
}
th {
    background-color: #838BC5;
}
th, td {
    border: 1px solid black;
    padding: 10px;
}
table {
    border: 1px solid black;
    border-spacing: 0 0;
    select {
        background-color: #838BC5;
        color: white;
        outline: none;
        border: none;
        text-decoration: none;
        margin-bottom: 5px;
    }
}

```

```

        border-radius: 5px;
        font-size: 18px;
        padding: 7px;
    }
    .td-input {
        display: flex;
        justify-content: center;
        width: fit-content;
        input {
            text-decoration: none;
            outline: none;
            border: none;
            text-align: center;
            font-size: 18px;
            width: 50px;
        }
    }
}
.btn-wrapper {
    display: flex;
    align-items: center;
}
button.save-btn {
    margin-bottom: 5px;
    border-radius: 5px;
    padding: 5px;
    text-decoration: none;
    outline: none;
    border: 2px solid transparent;
    background-color: #838BC5;

```

```

    color: white;
    font-size: 18px;
  }
  button.save-btn:focus {
    border: 2px solid #4D4D9B;
  }
  button.save-btn:active {
    background-color: #5966cc;
  }
  import { Component, OnInit } from '@angular/core';
  import { UserService } from 'src/app/services/user.service';

  @Component({
    selector: 'app-main',
    templateUrl: './main.component.html',
    styleUrls: ['./main.component.scss']
  })
  export class MainComponent implements OnInit {
    selectedGroup: string;

    days = ["05.09", "12.09", "19.09", "26.09", "05.10", "12.10"];
    groups: any;

    reservedGroups = {
      group1: {
        name: 'Група №1',
        data: [
          {
            name: "Крамаренко Михайло Сергійович",
            marks: {

```

```

    "05.09": "15",
    "12.09": "15",
    "19.09": "15",
    "26.09": "15",
    "05.10": "15",
    "12.10": "15"
  },
  exam: true
},
{
  name: "Серета Олександр Володимирович",
  marks: {
    "05.09": "15",
    "19.09": "15",
    "26.09": "15",
    "12.10": "15"
  },
  exam: true
},
{
  name: "Васильчук Анна Михайлівна",
  marks: {
    "05.09": "15",
    "12.09": "10",
    "19.09": "15",
    "26.09": "15",
    "05.10": "15",
    "12.10": "15"
  },
  exam: true
}

```

```

    },
    {
      name: "Коваленко Олена Вікторівна",
      marks: {
        "05.09": "15",
        "12.09": "15",
        "19.09": "15",
        "26.09": "15",
        "05.10": "15",
        "12.10": "15"
      },
      exam: true
    },
    {
      name: "Антоненко Денис Денисович",
      marks: {
        "05.09": "15",
        "12.09": "н",
        "19.09": "н",
        "26.09": "15",
        "05.10": "н",
        "12.10": "н"
      },
      exam: true
    },
  ],
},
group2: {
  name: 'Група №2',

```

```
data: [
  {
    name: "Мороз Марк Сергійович",
    marks: {
      "05.09": "15",
      "12.09": "15",
      "19.09": "15",
      "26.09": "15",
      "05.10": "15",
      "12.10": "15"
    },
    exam: false
  },
  {
    name: "Швець Дарина Олегівна",
    marks: {
      "05.09": "15",
      "12.09": "15",
      "19.09": "15",
      "26.09": "15",
      "05.10": "15",
      "12.10": "15"
    },
    exam: true
  },
  {
    name: "Карпенко Інна Тимофіївна",
    marks: {
      "05.09": "15",
      "12.09": "10",
```

```
"19.09": "15",  
"26.09": "15",  
"05.10": "15",  
"12.10": "15"  
},  
exam: false  
},  
{
```

```
name: "Гончар Іван Миколайович",  
marks: {  
"05.09": "15",  
"12.09": "10",  
"19.09": "15",  
"26.09": "15",  
"05.10": "15",  
"12.10": "15"  
},  
exam: false
```

```
},  
{  
name: "Вовк Володимир Валерійович",  
marks: {  
"05.09": "15",  
"12.09": "10",  
"19.09": "15",  
"26.09": "15",  
"05.10": "15",  
"12.10": "15"  
},  
exam: false
```

```

    },
  ],
},
group3: {
  name: 'Група №3',
  data: [{
    name: "Панасюк Анастасія Віталіївна ",
    marks: {
      "05.09": "10",
      "12.09": "10",
      "19.09": "Н",
      "26.09": "Н",
      "05.10": "Н",
      "12.10": "Н"
    },
    exam: false
  },
  {
    name: "Журба Микита Андрійович",
    marks: {
      "05.09": "15",
      "12.09": "10",
      "19.09": "15",
      "26.09": "15",
      "05.10": "15",
      "12.10": "15"
    },
    exam: false
  },
  {

```

name: "Шевченко Діана Олександрівна ",

marks: {

"05.09": "5",

"12.09": "10",

"19.09": "10",

"26.09": "5",

"05.10": "н",

"12.10": "15"

},

exam: false

},

{

name: "Тарасенко Ольга Петрівна",

marks: {

"05.09": "15",

"12.09": "10",

"19.09": "н",

"26.09": "15",

"05.10": "15",

"12.10": "15"

},

exam: true

},

{

name: "Бондаренко Олексій Дмитрович",

marks: {

"05.09": "н",

"12.09": "н",

"19.09": "н",

"26.09": "н",

```

        "05.10": "10",
        "12.10": "15"
    },
    exam: false
},
]
},
} get groupNames(): string[] {
    return Object.keys(this.groups);
}
constructor(public userService: UserService) { }
ngOnInit(): void {
    this.loadData()
}
changeGroup(value) {
    this.selectedGroup = value;
}
saveData(): void {
    localStorage.setItem('groups', JSON.stringify(this.groups))
} loadData(): void {
    const data = localStorage.getItem('groups');
    if (data) {
        this.groups = JSON.parse(data);
    } else {
        this.groups = this.reservedGroups;
    }
}
clearStorage(): void {
    localStorage.clear()
}

```