Міністерство освіти і науки України
Національний технічний університет
«Дніпровська політехніка»

Інститут електроенергетики
(інститут)

Факультет інформаційних технологій
(факультет)

Кафедра Програмного забезпечення комп'ютерних систем
(повна назва)

ПОЯСНЮВАЛЬНА ЗАПИСКА
кваліфікаційної роботи ступеня

бакалавра
(назва освітньо-кваліфікаційного рівня)

студента *Кондрашова Юрія Володимировича*
(ПІБ)

академічної групи *122-19ск-2*
(шифр)

спеціальності *122 Комп'ютерні науки*
(код і назва спеціальності)

освітньої програми *Комп'ютерні науки*
(назва освітньої програми)

на тему: *Розробка Web сервісу електронний щоденник учня для закладів середньої освіти з використанням фреймворку JavaScript Atom*

Керівники	**Прізвище, ініціали**	**Оцінка за шкалою**		**Підпис**
		рейтинговою	**інституційною**	
кваліфікаційної роботи	*доц. Кабак Л.В.*			
розділів:				
спеціальний	*доц. Кабак Л.В.*			
економічний	*доц. Касьяненко Л.В.*			
Рецензент				
Нормоконтролер	*доц. Гуліна І.Г.*			

Дніпро
2022

Міністерство освіти і науки України
НТУ «Дніпровська політехніка»

ЗАТВЕРДЖЕНО:
завідувач кафедри
програмного забезпечення комп'ютерних систем
(повна назва)

_____ І.М. Удовик
(підпис) (прізвище, ініціали)

« » _____ 2022 року

ЗАВДАННЯ
на кваліфікаційну роботу

бакалавра
(назва освітньо-кваліфікаційного рівня)

студента *122-19ск-2* *Кондрашова Юрія Володимировича*
(група) (прізвище та ініціали)

тема кваліфікаційної роботи *Розробка Web сервісу електронний щоденник учня для закладів середньої освіти з використанням фреймворку JavaScript Atom*

затверджена наказом ректора НТУ «ДП» від *18.05.2022 р.* № *268-с*

Розділ	Зміст виконання	Термін виконання
Спеціальний	***На основі матеріалів виробничої практики та інших науково-технічних джерел провести аналіз стану рішення проблеми та постановку задачі.*** ***Обґрунтувати вибір та здійснити реалізацію методів вирішення проблеми***	***13.05.2022 р.***
Економічний	***Провести розрахунок трудомісткості розробки програмного забезпечення, витрат на створення ПЗ й тривалості його розробки***	***27.05.2022 р.***

Завдання видав _____ *доц. Кабак Л.В.*
(підпис) (посада, прізвище, ініціали)

Завдання прийняв до виконання _____ *Кондрашов Ю.В*
(підпис) (прізвище, ініціали)

Дата видачі завдання: *14.01.2022 р.*

Термін подання кваліфікаційної роботи до ЕК: *13.06.2022 р.*

РЕФЕРАТ

Пояснювальна записка: 87 с., 18 рис., 5 дод., 42 джерел.

Об'єкт розробки: Розробка Web сервісу електронний щоденник учня для закладів середньої освіти з використанням фреймворку JavaScript Atom.

Мета дипломного проекту: забезпечити зручний Web сервіс для вчителів, учнів та їх батьків.

У вступі розглядається аналіз та сучасний стан проблеми, конкретизується мета кваліфікаційної роботи та галузь її застосування, наведено обґрунтування актуальності теми та уточнюється постановка завдання.

У першому розділі проведено аналіз предметної області, визначено актуальність завдання та призначення розробки, розроблена постановка завдання, задані вимоги до програмної реалізації, технологій та програмних засобів.

У другому розділі виконано аналіз існуючих рішень, обрано вибір платформи для розробки, виконано проектування і розробка програми, наведено опис алгоритму і структури функціонування системи, визначені вхідні і вихідні дані, наведені характеристики складу параметрів технічних засобів, описаний виклик та завантаження застосунку, описана робота програми.

В економічному розділі визначено трудомісткість розробленої інформаційної підсистеми, проведений підрахунок вартості роботи по створенню застосунку та розраховано час на його створення.

Актуальність даного програмного забезпечення визначається загальними світовими проблемами під час дистанційного навчання.

Список ключових слів: ВЕБ-ДОДАТОК, JAVASCRIPT, CSS, HTML, SQL.

ABSTRACT

Explanatory note: 87 pp., 18 figs., 5 appendices, 42 sources.

Object of development: Development of Web service electronic student diary for secondary schools using Atom JavaScript framework.

The purpose of the diploma project: to provide a convenient Web service for teachers, students and their parents.

The introduction considers the analysis and current state of the problem, specifies the purpose of the qualification work and the field of its application, provides a justification for the relevance of the topic and clarifies the problem.

In the first section the analysis of the subject area is carried out, the urgency of the task and purpose of development are defined, the statement of the task is developed, requirements to software realization, technologies and software are set.

The second section analyzes the existing solutions, selects the platform for development, performs design and development of the program, describes the algorithm and structure of the system, determines the input and output data, provides characteristics of the parameters of hardware, describes the call and download application, describes the program.

The economic section determines the complexity of the developed information subsystem, calculates the cost of work to create an application and calculates the time for its creation.

The relevance of this software is determined by the general world problems during distance learning.

List of keywords: WEB APP, JAVASCRIPT, CSS, HTML, SQL.

ЗМІСТ

СПИСОК УМОВНИХ ПОЗНАЧЕНЬ

ЕОМ – електронно-обчислювальна машина.

CRUD – create, read, update, delete.

API – інтерфейс програмування додатків.

HTML – HyperText Markup Language.

CSS – Cascading Styles Sheets.

W3C – World Wide Web Consortium.

DOM – об’єктна модель документа.

SQL – Structured Query Language.

СУБД – Система Управління Базами Данних.

Вступ

Електронні освітні ресурси – засоби навчання на цифрових носіях будь-якого типу або розміщені в інформаційно-телекомунікаційних системах, які відтворюються за допомогою електронних технічних засобів і застосовуються в освітньому процесі.

Вимушене дистанційне навчання через поширення пандемії COVID-19 стало викликом для всіх учасників освітнього процесу: вчителів, учнів та батьків. Карантин поставив нові виклики перед закладами освіти, а також органами управління освітою на місцевому і національному рівнях. Організація ефективного дистанційного або змішаного навчання, адаптація освітньої політики до вимог сьогодення, посилення фінансового забезпечення закладів освіти, модернізація їх матеріально-технічної бази — це перелік тих труднощів, які мають здолати управлінці та освітяни зараз.

Світовою тенденцією останнього десятиріччя є розвиток інформаційного суспільства. Урядом України також поставлено завдання поглиблення інформатизації національної економіки, широкого запровадження інформаційно-комунікаційних технологій у різні сфери суспільного життя. Інформатизація освітньої галузі в Україні має певні здобутки. Цифрові технології все активніше використовуються для вдосконалення організації освітнього процесу та покращення управління в школах, коледжах, університетах. Для управління на центральному рівні створено ряд освітніх реєстрів, функціонують освітні інформаційні системи, за допомогою яких збирається, обробляється, зберігається різноманітна статистична та адміністративна інформація. На сьогодні уявити сучасну освіту без інформаційних технологій неможливо. Утім в освітній галузі залишаються та з’являються нові проблеми, які необхідно розв’язувати. Зокрема, завершення 2019/2020 н. р. відбувалось у дистанційній формі, вимушене використання якої зумовлено пандемією COVID-19. Провадження ефективного дистанційного навчання потребує вирішення багатьох технічних завдань із залученням сучасних комп’ютерних технологій.

Серед головних викликів, які постали у сфері інформатизації освіти Необхідність створення єдиної інформаційної платформи для відповідних сегментів системи освіти, передусім повної загальної середньої освіти, яка є обов'язковою для всіх громадян України відповідно до ст. 53 Конституції України. Створення на національному рівні інформаційних баз з індивідуальними деперсоналізованими даними про здобувачів освіти та педагогічних працівників а також відповідних державних освітніх реєстрів. У розвинених країнах світу вже більше десятиріччя такий інструментарій використовується як основа для ідентифікації освітньої траєкторії здобувача освіти. За рівнем успішності його реалізації в дорослому житті оцінюється якість освітніх послуг та ефективність функціонування закладів освіти. Індивідуальні дані педагогічних працівників забезпечать облік реальної чисельності педагогічного персоналу закладів освіти.[1-5]

Метою дипломного проекту є створення веб-додатку, який надасть можливість батькам, учням та вчителям навчальних закладів взаємодіяти між собою, та спростить моменти контролю отримання та виконання домашніх завдань.

Беручи це все до уваги було сформовано тему дипломної роботи: «Розробка Web сервісу електронний щоденник учня для закладів середньої освіти з використанням фреймворку JavaScript Atom»

РОЗДІЛ 1
АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Загальні відомості с предметної галузі

Цифрові технології стають дедалі помітнішою особливістю сучасного надання освіти та практики в усьому світі та займають центральне місце в популярній уяві про майбутнє освіти. Освітнє значення цифрових технологій посилилося через широке використання ресурсів цифрової освіти під час пандемії. Таким чином, знову активно пропагується надія на те, що цифрові технології можуть трансформувати освіту в широких та розширюючих напрямках, і, незважаючи на їх нерівномірну історію на сьогоднішній день, активно просувається.

Цифрові технології явно зросли і стали значною присутністю в освіті з 1980-х років, коли комп'ютери та елементарне освітнє програмне забезпечення було вперше введено в класи в невеликій кількості країн. У країнах з високим рівнем доходу сьогодні важко уявити навчання без ноутбуків, Google і текстової обробки. Такі ресурси, як YouTube і Вікіпедія, є першими місцями, куди звертаються багато учнів, бажаючи знайти інформацію та дізнатися нове, і в переважній більшості шкіл у цих країнах зараз є сотні (якщо не тисячі) цифрових пристроїв і екранів. Все, що можна оцифрувати, зберігається в Інтернеті. Уроки транслюються в прямому ефірі, ресурси доступні для завантаження, а спілкування зазвичай відбувається через програми та електронну пошту. У той же час в освіті в країнах з низьким і середнім рівнем доходів ширше використовується «m-learning» (мобільне навчання), електронні читалки та інші цифрові ресурси, які можуть підтримати розширені можливості для участі в освіті в громадах, яким інакше не вистачає надійних традиційне освітнє забезпечення Ці зрушення відображають постійний розвиток потужніших і дешевших цифрових технологій, намагання розширити доступ до мобільного Інтернету для сільських громад і громад з низьким рівнем доходів у поєднанні з

постійними зусиллями політиків у всьому світі щодо розробки національних стратегій «цифрової освіти» протягом останніх 40 років.

Найбільш чіткі зміни в освіті, з якими пов'язувалися цифрові технології – це ті, які рідко визнаються в основних дискусіях про освітні технології. По-перше, і найголовніше, цифрові технології були складовою частиною переходу до все більш стандартизованих форм вимірювання освіти, метризації та міжнародного порівняння практики вчителів – стандартизації, яка була важливою для створення сприятливих відносин у освіті між студентами та вчителями. По-друге, «edtech» швидко виріс у багатомільярдну індустрію, яка підтримується широким фінансуванням та залучає до участі широкий спектр компаній та комерційних інтересів із-за меж традиційного сектору освіти.

Непослідовні та нерівномірні результати технологічної освіти були різко зосереджені на постійній відповіді на пандемію COVID-19, оскільки країни намагаються запровадити форми «дистанційного навчання», де діти і молоді люди можуть підтримувати певну безперервність освітньої діяльності далеко від своїх звичайних очних класів. Цей різкий розрив у забезпеченні освіти підкреслив як можливості, так і обмеження цифрової освіти. Наприклад, у країнах від Європи до Індії були введені надзвичайні форми «тимчасової дистанційної освіти» з використанням власних цифрових пристроїв сімей, а також шкільних навчальних платформ та різних інших освітніх програм і програмного забезпечення. Ці офіційні положення були підкріплені інноваційним неформальним використанням цифрових технологій, оскільки окремі вчителі та сім'ї імпровізували за допомогою популярних платформ соціальних мереж, щоб забезпечити продовження шкільного навчання протягом наступних періодів карантину. Багатьом країнам з високим рівнем доходу доводиться покладатися на додаткові нецифрові підходи для підтримки повторюваних періодів блокування шкіл – наприклад, звертаючись до телевізійних трансляцій навчальних матеріалів, щоб забезпечити доступ всього населення до дистанційного навчання. Це свідчить про те, що навіть у найбільш забезпечених країнах зберігається «цифровий розрив», через що багато сімей з

низьким рівнем доходів не мають базового доступу до пристроїв та підключення до Інтернету, необхідних для доступу до цифрових навчальних матеріалів вдома, а там, де такий доступ є, не мають особистого доступу. Пандемія COVID-19 також підкреслила нерівність «другого порядку» у характері та формі освіти, яка може здійснюватися переважно через цифрові канали. Виникали занепокоєння щодо якості викладання та навчання, які стали можливими завдяки дистанційній онлайн-освіті, поряд із вираженою «втратою навчання» серед малозабезпечених та інших груп із неблагополучним становищем, у поєднанні зі значними психосоціальними потребами тривалого онлайн-навчання. Коротше кажучи, хоча цифрові технології відіграли ключову роль у підтримці базових форм освіти в багатьох країнах з високим рівнем доходу під час пандемії COVID-19, це виявилося явно обмеженою формою навчання.

2020-ті поки що характеризуються постійними розмовами про цифрові технології як про «готові рішення зверху вниз». Наприклад, лунають помітні заклики розглядати дистанційне навчання COVID як «переломний момент», після якого продовження після пандемічного розширення онлайн-навчання надасть «безпрецедентну можливість трансформувати освіту в цілих системах». Так само зростає ентузіазм щодо застосування технології штучного інтелекту в освіті. Як припускає Ден Аюб з Microsoft: штучний інтелект має великий вплив на стан освіти сьогодні, і наслідки є величезними. Штучний інтелект має потенціал трансформувати роботу нашої системи освіти, підвищити конкурентоспроможність закладів і надати вчителям і учням усі здібності. Цей ентузіазм зумовлений трьома різними формами нових освітніх технологій.

По-перше, ми бачимо появу низки постCOVID-форм технологічних конфігурацій очного навчання. Поряд із постійним впровадженням «систем управління навчанням» для полегшення спільного використання ресурсів та групового спілкування, зростає інституційний ентузіазм щодо «змішаних», «гібридних» і «гіфлексних» підходів, які передбачають розміщення навчання, принаймні частково, в Інтернеті. Також слід звернути увагу на платформи соціального навчання в Інтернеті, де молоді люди можуть навчатися разом поза

школою і оцінювати один одного. Перехід на дистанційне навчання часто пов’язаний з повним переміщенням освіти надання комерційних платформ – як показано на платформах онлайн-репетиторства, які забезпечують додаткове навчання після школи. Тим не менш, оскільки студенти в усьому світі поступово повертаються до очного навчання, зростає інтерес до потенційних можливостей переходу на комбіноване навчання в класі та онлайн.

По-друге, це постійне зростання індивідуалізованого навчання. Системи, розроблені для того, щоб спрямувати індивідуальну взаємодію студентів з онлайновими навчальними ресурсами за допомогою використання складної аналітики, керованої даними, для керівництва учнями при прийнятті рішень. Тут стверджується, що кожен учень отримує вигоду від величезної кількості аналізованих даних, які, як вважають деякі постачальники, надають цим системам можливість знати більше про навчання будь-якої людини, ніж на це міг сподіватися вчитель.

По-третє, це низка інших форм технологій, керованих штучним інтелектом, переважно розроблених для підтримки автоматизованого прийняття рішень установами, викладачами та студентами. Сюди входить загальносистемне «автоматизоване управління освітою», засноване на моделюванні на основі штучного інтелекту, а також специфічне використання інституційного підбору персоналу, закупівель і прогнозної «бізнес-аналітики». Поряд із цими інституційними формами ШІ, тепер доступні й інші технології, керовані ШІ, для виконання завдань, які раніше виконували вчителі. Сюди входять «лінгастилометричні показники» на основі штучного інтелекту та автоматизоване оцінювання есе.

Прихильники цих технологій обіцяють:

- Ефективність освіти. Це технології, які обіцяють призвести до економічної ефективності, заощадження часу та пришвидшення освітніх процесів та загального уникнення інституційної інерції. Автоматизовані технології обіцяють досягти більшої ефективності за рахунок зменшення (або видалення) кількості «людей у циклі»

- «Точне» навчання: це технології, які обіцяють адаптувати освітні заходи відповідно до особистих потреб і характеристик людини, які часто беруться з особистих дані. Ця точність також є основою для передбачення – передбачати ймовірний розвиток і прогрес і відповідно змінювати дії.

- Диференціація навчання: це технології, які обіцяють підтримувати різноманітні форми навчання, які найкраще відповідають потребам людини. Це один із ключових аспектів технологій «персоналізованого навчання», тобто. уявлення про те, що люди найкраще можуть змінювати та «саморегулювати» своє власне навчання у світлі зворотного зв'язку з машинами.

- Покращений «інсайт» і «знання»: це технології, які обіцяють запропонувати уявлення про невидимі та непізнані в іншому випадку аспекти освіти. «Завжди ввімкнений» моніторинг і комплексний збір даних відкривають перспективу бути в змозі знати все – те, що описують як «безрамкову» логіку.

Як і попередні «хвилі» інновацій edtech, ці харизматичні заклики супроводжуються передбачуваним зростанням комерційної активності та вимогами трансформаційного впливу. Негайно пандемічний поворот до онлайн-викладання мобілізував і розширив комерційний ринок для надання цифрової освіти. [5-19]

Одним серед популярних інструментів, який забезпечує базову цифровізацію системи шкільної освіти, є система електронних журналів та електронних щоденників. Такі системи спрямовані на забезпечення безперервної та ефективної взаємодії між чотирма групами учасників процесу – адміністрація школи, вчителі, учні та батьки.

У багатьох країнах електронними журналами та щоденниками користуються вже давно. Електронні модулі мають свої особливості в залежності від освітньої системи певної країни, але функціонують за одними й тими ж принципами.

У **Фінляндії** діє електронна система Wilma. Батьки, використовуючи свій код доступу, можуть ознайомитися не лише з оцінками дитини, а й з

інформацією, яку залишає особисто для них шкільний психолог, лікар чи соціальний працівник.

В **Естонії** з 2008 року діє система eKool, що забезпечує взаємодію всіх учасників навчального процесу. Учні можуть ознайомитися з розкладом, завданнями та оцінками. Батьки також мають доступ до цієї інформації та спілкуються в системі з учителями.

У **Литві** ще з 2008 популярна електронна система «Моя школа» від Nevda, там її використовують майже у 80% шкіл. Це електронний журнал, щоденник та платформа, що забезпечують комунікацію всіх учасників освітнього процесу. Наприклад, учні можуть слідкувати за розкладом та своїм рейтингом успішності зі смартфона, вчителі мають можливість робити автоматизовані звіти, а батьки обов‘язково отримують повідомлення, якщо дитина не з‘явилась на заняттях.[24-28]

Впровадження системи електронних журналів та щоденників на державному рівні в Україні реалізується у рамках затвердженої Урядом Концепції розвитку електронного урядування. За офіційними даними, станом на 1 лютого 2021 року електронними журналами та щоденниками користуються 134 школи по всій країні. Найбільше таких навчальних закладів у Київській, Дніпропетровській, Івано-Франківській та Сумській областях. Навчальних закладів, які ще не здійснили перехід, але вже зареєструвалися в системі, значно більше.

Перехід на електронні версії щоденників нині необов‘язковий: навчальний заклад сам вирішує, впроваджувати інновацію чи ні. Очікується, що у 2022 році школи активно підтримають цю ініціативу. Паралельно вдосконалюється функціонал самих електронних журналів та щоденників.

E-Journal почав роботу в тестовому режимі в жовтні 2020 року. Наприкінці грудня МОН доручив розпоряднику системи АІКОМ — Інституту освітньої аналітики — ввести модуль «у промислову експлуатацію». Відтоді ним почали користуватися понад 100 шкіл. Заступник міністра освіти і науки України з питань цифровізації Артур Селецький зазначив, що розроблений функціонал

модуля E-Journal «є об'єктивно компромісним з урахуванням необхідності оперативного впровадження сучасних цифрових технологій в освітній процес у всіх школах країни».

Функціонал E-Journal досить великий та має такі функції:

Електронний журнал (рис 1.1)

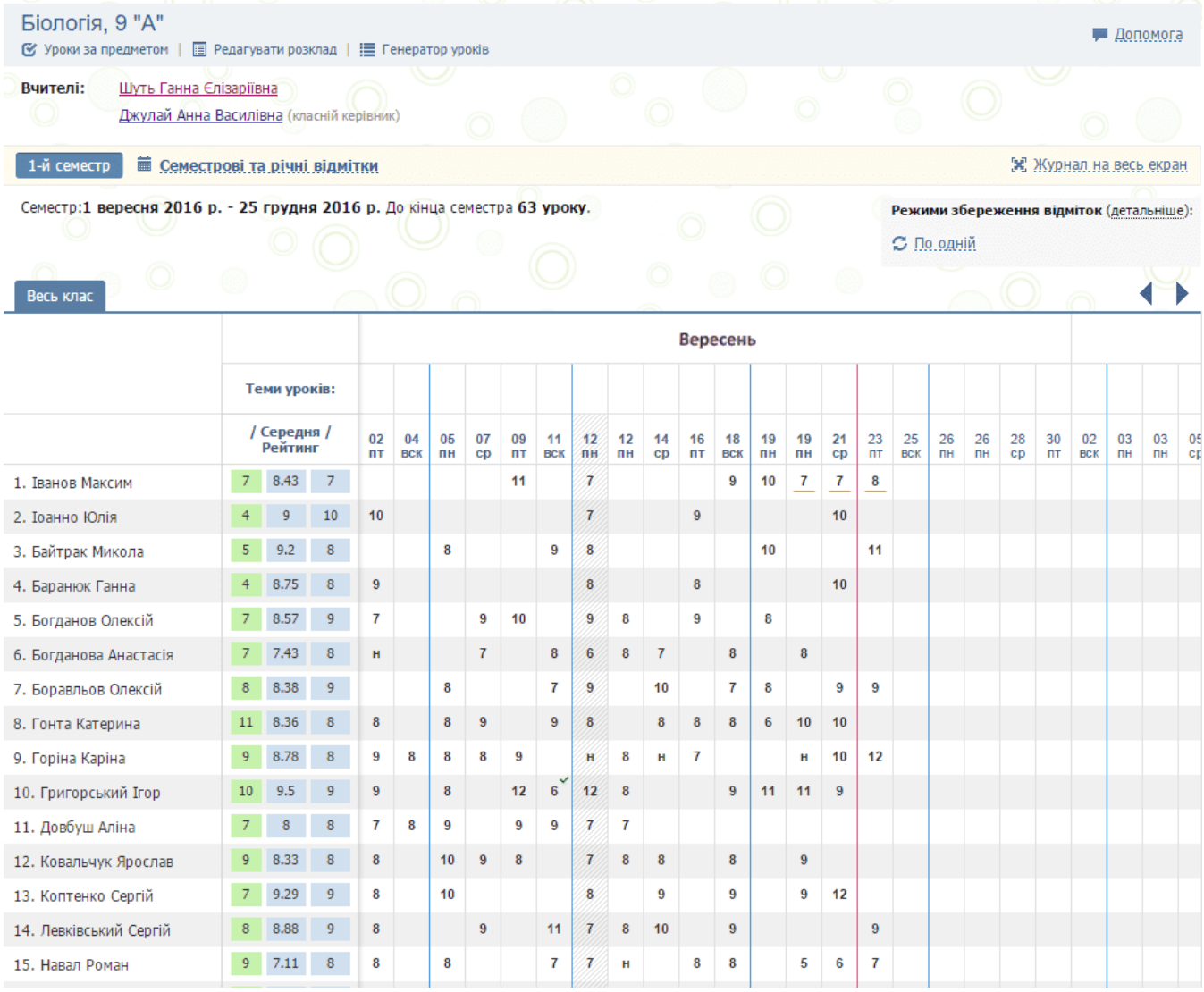


Рис. 1.1 – інтерфейс розділу «Електронний журнал»

Для кожного класу по кожному предмету створюється журнал. Доступ до нього мають вчителі-предметники, класний керівник, директор.

Батькам і учням журнал недоступний.

За допомогою електронних журналів вчителі за два кліка можуть:

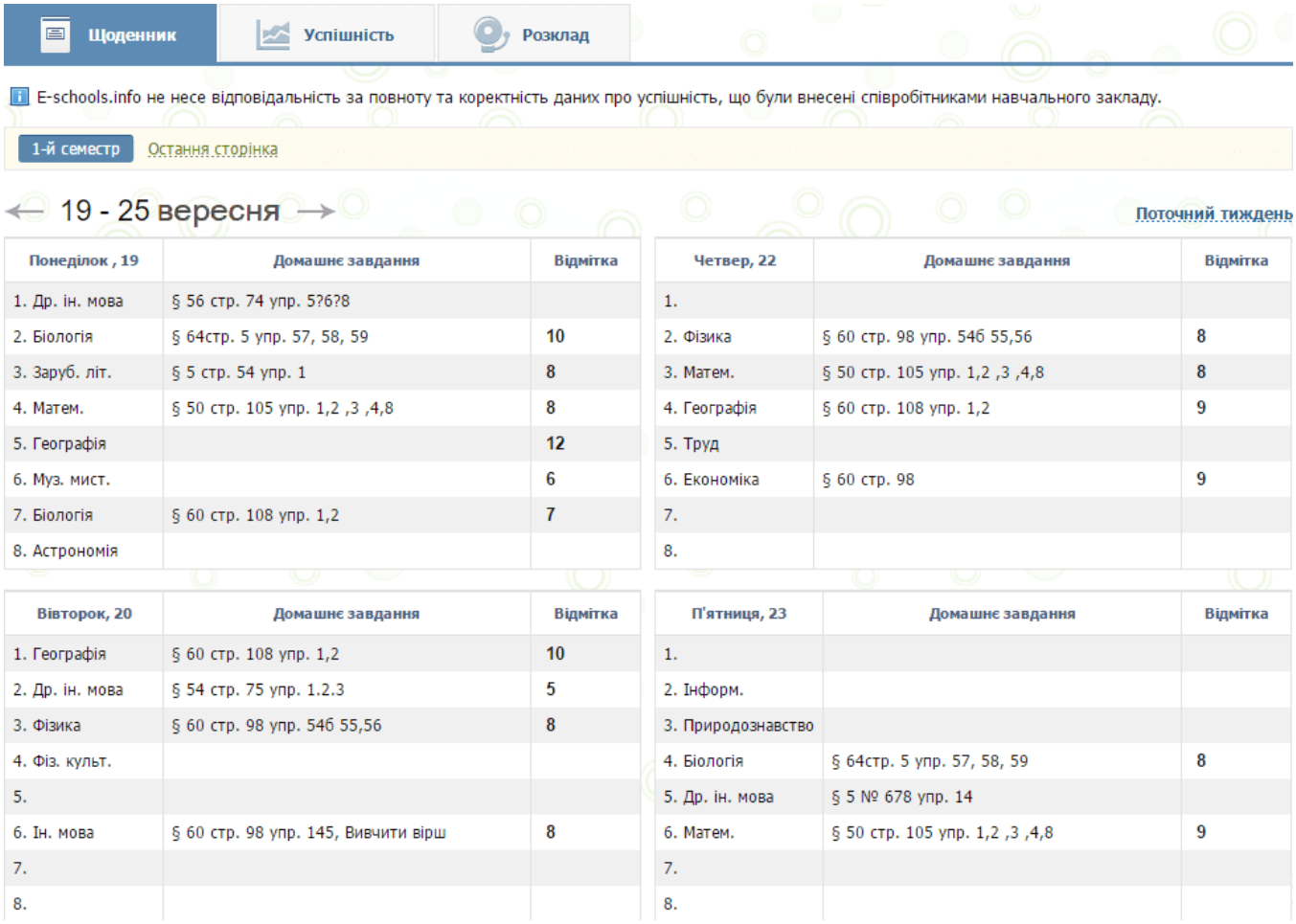
- виставити оцінки,
- відзначити відсутніх,

• написати зауваження до відміток і пропусків, залишити коментарі до уроків (наприклад, «контрольна робота»),

• внести домашні завдання.

По кожному предмету вираховується кількість пропусків і середня відмітка за чверть.

Електронний щоденник(Рис. 1.2)



Понеділок , 19	Домашнє завдання	Відмітка
1. Др. ін. мова	§ 56 стр. 74 упр. 5?6?8	
2. Біологія	§ 64стр. 5 упр. 57, 58, 59	10
3. Заруб. літ.	§ 5 стр. 54 упр. 1	8
4. Матем.	§ 50 стр. 105 упр. 1,2 ,3 ,4,8	8
5. Географія		12
6. Муз. мист.		6
7. Біологія	§ 60 стр. 108 упр. 1,2	7
8. Астрономія		

Четвер, 22	Домашнє завдання	Відмітка
1.		
2. Фізика	§ 60 стр. 98 упр. 54б 55,56	8
3. Матем.	§ 50 стр. 105 упр. 1,2 ,3 ,4,8	8
4. Географія	§ 60 стр. 108 упр. 1,2	9
5. Труд		
6. Економіка	§ 60 стр. 98	9
7.		
8.		

Вівторок, 20	Домашнє завдання	Відмітка
1. Географія	§ 60 стр. 108 упр. 1,2	10
2. Др. ін. мова	§ 54 стр. 75 упр. 1.2.3	5
3. Фізика	§ 60 стр. 98 упр. 54б 55,56	8
4. Фіз. культ.		
5.		
6. Ін. мова	§ 60 стр. 98 упр. 145, Вивчити вірш	8
7.		
8.		

П'ятниця, 23	Домашнє завдання	Відмітка
1.		
2. Інформ.		
3. Природознавство		
4. Біологія	§ 64стр. 5 упр. 57, 58, 59	8
5. Др. ін. мова	§ 5 № 678 упр. 14	
6. Матем.	§ 50 стр. 105 упр. 1,2 ,3 ,4,8	9
7.		
8.		

Рис. 1.2 – інтерфейс розділу «Щоденник»

На підставі даних, внесених вчителями до журналів, для кожного учня формується його електронний щоденник.

У щоденнику відображено все, що вчителі внесли до журналу (відмітки, пропуски, коментарі тощо), а також поведінка і зауваження за кожен тиждень.

Для батьків є можливість «підписувати» щоденник своєї дитини.Батьки учня мають доступ до всіх даних тільки своєї дитини.

Таблиця успішності(Рис. 1.3)

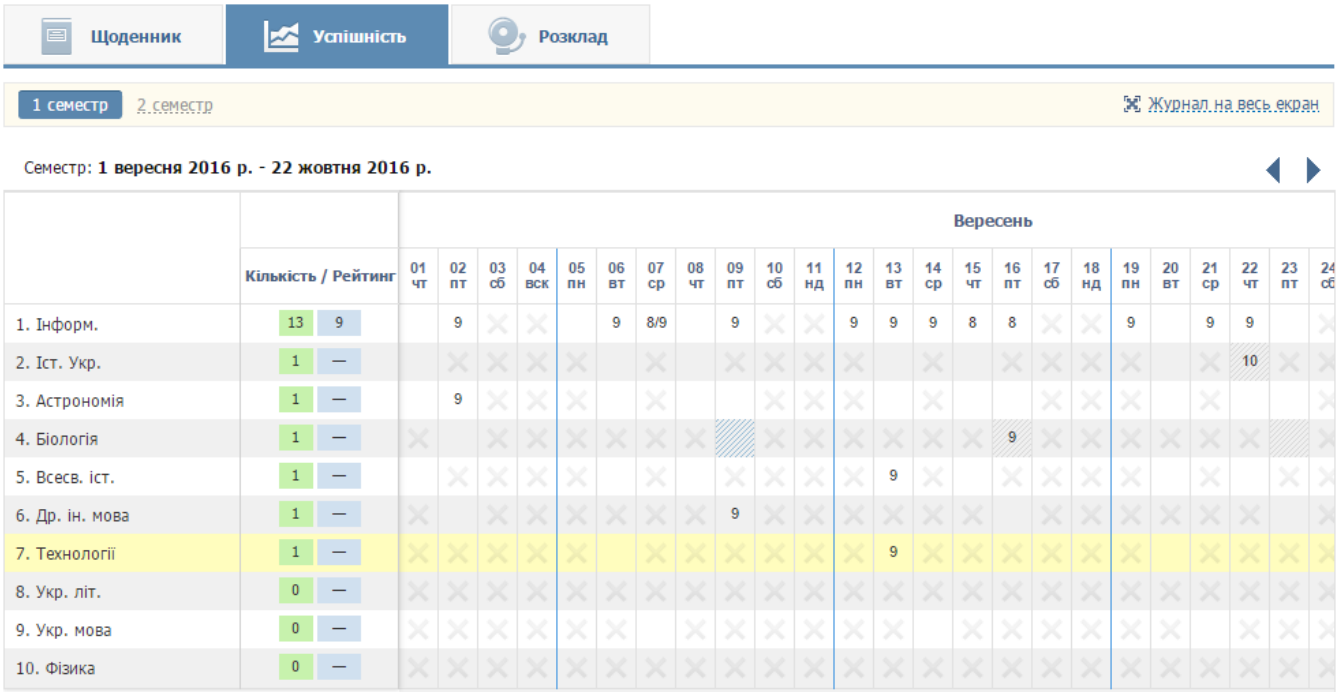


Рис. 1.3 – інтерфейс розділу «Успішність»

Для кожного учня є зведена таблиця всіх відміток, отриманих за чверть з усіх предметів. У таблиці містяться також усі пропуски, середній бал і четвертна відмітка по кожному предмету.

При наведенні миші на клітину з відміткою з'являється додаткова інформація про неї: ким і коли була виставлена відмітка, а також коментар вчителя.

Таблиця успішності доступна як вчителям, так і батькам, і учням. Таким чином, можна легко і зручно простежити успішність кожного учня всередині чверті.

Графіки успішності (1.4)

Для кожного учня і кожного класу доступна статистика успішності в графічному вигляді.

Є можливість переглянути графіки успішності всередині обраної чверті, а також порівняти успішність в усіх чвертях навчального року.

Також можна переглянути статистику з обраних предметів і/або для певних учнів.

Доступний графічний рейтинг успішності учнів класів.

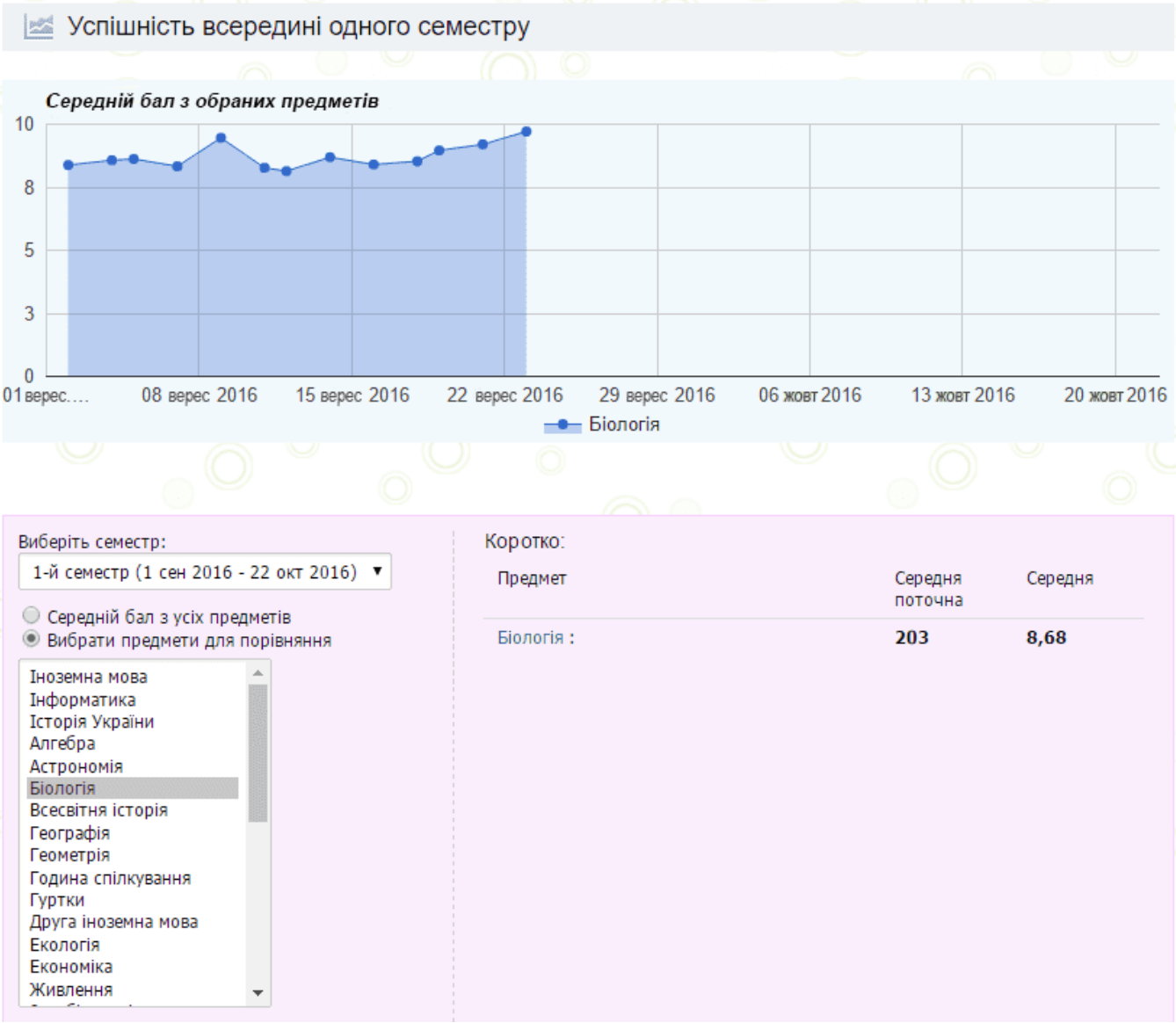


Рис. 1.4 – інтерфейс розділу «Графік успішності»

Також існує декілька інших можливостей, в тому числі:

- розклад уроків учня і вчителя;
- розклад шкільних дзвінків, чвертей і канікул;
- персональні сторінки кожного користувача;
- спілкування всередині школи.[41]

1.2 Призначення розробки та область застосування.

Темою бакалаврської дипломної роботи виступає: «Розробка Web сервісу електронний щоденник учня для закладів середньої освіти з використанням

фреймворку JavaScript Atom». Тобто головною метою роботи є створення веб-сайту, який представляє онлайн щоденник в якому вчителя зможуть виставляти домашнє завдання для учнів.

Головними критеріями розроблювального веб-додатку є:

- Зручність в використанні.
- Легкість в освоєнні.
- Максимальна доступність для користувача

Система призначена для:

- Створення записів.
- Взаємодії із компонентами середовища та самим середовищем.

Система позиціонується як веб-додаток, який дає можливість використовувати щоденник для самоорганізації, простеження нових завдань, та їх виконання.

1.3 Підстава для розробки

Відповідно до ОКХ та ОПП, згідно навчального плану та графіків навчального процесу, в кінці навчання студент виконує кваліфікаційну роботу (проект). Тема роботи узгоджується з керівником проекту, випускаючою кафедрою, та затверджується наказом ректора.

Отже, підставами для розробки (виконання кваліфікаційної роботи) є:

- ОКХ та ОПП за напрямом підготовки 6.050101 «Комп'ютерні науки»;
- Графік навчального процесу та навчальний план;
- наказ ректора Національного технічного університету «Дніпровська політехніка» № 286-с від 18.05.2022 р;
- завдання на дипломний проект на тему «Розробка Web сервісу електронний щоденник учня для закладів середньої освіти з використанням фреймворку JavaScript Atom».

1.4 Постановка завдання

Метою дипломного проекта є розробка веб-додатку для шкільного щоденника. Система призначена для швидкого та зручного простеження завдань та контролю виконання.

Веб-додаток повинен реалізувати такі дії як:

- Збереження даних в базі даних.
- Надавати можливість додавання та перегляду записів.
- Давати змогу інтерактивно взаємодіяти між елементами програмного інтерфейсу.

Для виконання проекту необхідно:

- Проаналізувати існуючі рішення.
- Спроектувати архітектуру системи.
- Розробити дизайн та зручний інтерфейс додатку.
- Реалізувати front-end та back-end спроектованої системи.

1.5. Вимоги до програми або програмного виробу.

1.5.1. Вимоги до функціональних характеристик.

Розроблене програмне забезпечення, для того, щоб досягнути поставлених цілей, повинно підтримувати виконання таких дій:

- Реагування на дії користувача в онлайн режимі.
- Збереження будь яких змін користувачем.
- Надання максимально швидкої навігації по сайту.
- Коректна візуалізація елементів веб-додатку.

Для підтримки вище перераховані функцій у додатку має бути реалізовано:

- Підтримка веб-браузера та доступ до програми через нього.
- Програмна та апаратна сумісності.

• Стандартна конфігурація яка дає змогу ввести застосунок в експлуатацію.

1.5.2 Вимоги до інформаційної безпеки.

Для коректної роботи програми потрібно реалізувати:

• Можливість редагування даних.

• Можливість тривалої роботи.

• Контроль та обробка вхідних даних.

• Збереження цілісності даних у випадку збою системи.

• Локальне збереження даних для більшої захищеності.

Також система візуального планування задач повинна мати наступні характеристики:

• Захист від несанкціонованого доступу.

• Захист даних користувача при повторному підключені.

1.5.3 Вимоги до складу та параметрів технічних засобів.

Для забезпечення надійного функціонування програмного забезпечення необхідно, щоб обчислювальна машина, на якій буде експлуатуватися веб-додаток, мала такі характеристики:

• Маніпулятор “миша” та клавіатура.

• Доступ до онлайн мережі.

• 1 Гб вільного місця на жорсткому диску.

• Процесор Intel Pentium.

• Не менше 2 Гб оперативної пам’яті.

• Монітор.

Вище наведені характеристики являють собою рекомендовані. Це означає, що при наявності характеристик не нижче зазначених, розроблений додаток буде

функціонувати відповідно до вимог щодо надійності, безпеки та швидкості обробки даних.

1.5.4 Вимоги інформаційної та програмної сумісності.

Для коректного функціонування програми необхідно, щоб програмне забезпечення обчислювальної машини, на якій буде експлуатуватися веб-додаток, відповідало наступним вимогам:

- Веб браузер Google Chrome
- Операційна система Windows 7/10.

Веб-орієнтована підсистема має бути реалізована на мові програмування JavaScript з веб-компонентів та зберігання даних. Для візуалізації було використано CSS який описує зовнішній вигляд сторінки.

РОЗДІЛ 2
ПРОЕКТУВАННЯ ТА РОЗРОБКА ІНФОРМАЦІЙНОЇ СИСТЕМИ

2.1. Функціональне призначення системи

Під час виконання дипломної роботи було розроблено веб-додаток у вигляді шкільного щоденника для поліпшення процесу дистанційного навчання, його головною метою є забезпечити зручний та якісний інструментарій для самоорганізації учнів і батьківського контролю з виконання завдань.

Призначення розробленого додатку:

- Надання можливості створення записів.
- Надання можливості маніпулювання створеними записами в онлайн середовищі.
- Надання можливості навігації між доступними для перегляду елементами.
- Збереження всіх дій та даних в сховищі.
- Надання візуального простеження за створеними записами.

Для досягнення вище перерахованих функціональних можливостей розроблена програма повинна:

- Мати сумісність з стандартною апаратною конфігурацією обчислювальної машини.
- Мати підтримку програмного забезпечення, а саме операційних систем Windows 10/7.
- Мати підтримку сучасних браузерів Firefox та Google Chrome.

2.2. Опис застосованих математичних методів

Під час розробки проекту для реалізації відображення елементу в таблиці «середня оцінка» використовувався метод обчислення середнього арифметичного числа.

Середнє арифметичне число називають «середнім» або «середнім арифметичним». Воно обчислюється шляхом додавання всіх чисел у даному наборі даних, а потім ділення їх на загальну кількість елементів у цьому наборі. Середнє арифметичне для рівномірно розподілених чисел дорівнює самому середньому числу, крім того розраховується за допомогою численних методів, які ґрунтуються на кількості даних та розподілі даних.

Загальна формула для визначення середнього арифметичного даних:

$$X_{\text{сер}} = \frac{\text{сума всіх спостережень}}{\text{кількість спостережень}} \qquad (2.1)$$

Дані можуть бути представлені в різних формах. Наприклад, коли ми маємо вихідні дані, як-от оцінки учня з п’яти предметів, ми додаємо оцінки, отримані з п’яти предметів, і ділимо суму на 5, оскільки всього 5 предметів.

2.3. Опис використаних технологій та мов програмування

Згідно з темою веб-додаток має бути реалізований на мові програмування JavaScript з використанням редактору Atom. Для візуалізації було використано CSS який описує зовнішній вигляд сторінки.

Опис мови програмування JavaScript

Вже майже десять років про JavaScript говорять як про одну з найбільш затребуваних мов програмування на ринку. Здебільшого це пов’язано з його широким і зростаючим діапазоном зручності та можливостей для розробників, які сприймають його динамічну природу та вивчають його.

За даними GitHub, JavaScript стабільно оцінюється як мова програмування, яка найчастіше використовується веб-розробниками. Станом на цей рік W3Tech Surveys зазначає , що JavaScript використовується для програмування на стороні клієнта 97,1% усіх веб-сайтів, включаючи деякі з найпопулярніших веб-сайтів у світі, таких як Google, YouTube, Facebook та Amazon.

JavaScript – це динамічна мова програмування, яка використовується для веб-розробки, у веб-додатках, для розробки ігор та багато іншого. Дозволяє

реалізувати динамічні функції на веб-сторінках, які неможливо зробити лише за допомогою HTML і CSS.

Багато браузерів використовують JavaScript як мову сценаріїв для виконання динамічних речей в Інтернеті. Кожного разу, коли ви бачите меню «Показати при натисканні», додатковий вміст доданий на сторінку і динамічно змінювані кольори елементів на сторінці, зокрема деякі функції, ви бачите наслідки JavaScript.

Однією з основних переваг JavaScript є те, що він не вимагає дорогих інструментів розробки. Ви можете почати з простого текстового редактора, такого як Блокнот. Оскільки це інтерпретована мова в контексті веб-браузера, вам навіть не потрібно купувати компілятор.

Перевагами JavaScript є:

- Менше взаємодії з сервером, ви можете перевірити введення користувача перед відправкою сторінки на сервер. Це економить трафік сервера, що означає менше навантаження на ваш сервер.
- Негайний зворотній зв’язок із відвідувачами, їм не потрібно чекати перезавантаження сторінки, щоб побачити, чи не забули вони щось ввести.
- Підвищена інтерактивність, ви можете створювати інтерфейси, які реагують, коли користувач наводить курсор на них за допомогою миші або активує їх за допомогою клавіатури.
- Розширені інтерфейси, ви можете використовувати JavaScript, щоб включити такі елементи, як компоненти перетягування й повзунки, щоб надати відвідувачам вашого сайту багатий інтерфейс.

Оскільки JavaScript був створений з основною метою — увімкнути динамічний вміст на статичних веб-сторінках, мало хто очікував, що JavaScript також буде використовуватися у бекенд-програмуванні. З появою NodeJS у 2009 році JavaScript почав все частіше використовуватися при розробці серверних служб. NodeJS – це технологія, створена на основі Chrome V8 JS. Принцип, на якому він працює, дозволяє виконувати велику кількість паралельних запитів, що знайшло застосування в розробці багатьох типів проектів, таких як бекенд API,

додатки реального часу та відео комунікації, програми обробки звуку та відео, додатки IoT, мікро сервіси та без серверні архітектури тощо. Крім великих додатків, NodeJS підходить не для всіх типів проектів. Як і у випадку майже з усіма технологіями.[26,27]

Як було сказано раніше, JavaScript це динамічна мова програмування яка використовується для опису клієнтського скрипту і серверної мови - це відноситься до можливості поновлення відображення веб-сторінки або додатків, щоб показувати різні речі в різних обставинах, генеруючи новий контент в міру необхідності. Серверний код динамічно генерує новий контент на сервері, наприклад дістає дані з бази даних, тоді як клієнтський JavaScript динамічно генерує новий зміст всередині браузера на клієнті, наприклад створює нову HTML таблицю, вставляючи в неї дані отримані з сервера, а потім відображає таблицю на веб-сторінці , яку бачить користувач.

Багато баз даних NoSQL зберігають дані у форматі JSON, що означає нотацію об’єктів JavaScript , тому, якщо ви знайомі з JavaScript, у вас не виникне проблем із цими базами даних. Крім того, багато з них мають вбудовану підтримку технологій JavaScript, таких як NodeJS та його фреймворків і бібліотек.

Також дуже корисною є функціональність, створена поверх основної мови Javascript. Так звані інтерфейси прикладного програмування (API), які надають додаткові надможливості для використання в JavaScript.

Інтерфейси прикладного програмування (API) — це конструкції, доступні мовами програмування, щоб дозволити розробникам легше створювати складні функції . Вони абстрагують більш складний код від вас, забезпечуючи деякий легший синтаксис для використання замість нього. [18]

Опис мови CSS

CSS переводиться як каскадні таблиці стилів. Коротше кажучи, CSS — це мова дизайну, яка робить веб-сайт привабливішим, ніж просто звичайний або ненадихаючий фрагмент тексту. Тоді як HTML значною мірою визначає

текстовий вміст, CSS визначає візуальну структуру, макет та естетику. HTML — це мова розмітки, а CSS — мова таблиць стилів. CSS - це основна технологія всесвітньої павутини, поряд із HTML та JavaScript.

Існує ряд переваг CSS, зокрема:

1) Вища швидкість сторінки - Більше коду означає повільну швидкість сторінки. А CSS дозволяє використовувати менше коду. CSS дозволяє використовувати одне правило CSS і застосовувати його до всіх входжень певного тегу в HTML-документ.

2) Кращий користувальницький досвід - CSS не тільки робить веб-сторінки зручними для ока, але також дозволяє зручне форматування. Коли кнопки та текст розташовані в логічних місцях і добре організовані, покращується взаємодія з користувачем.

3) Швидший час розробки - За допомогою CSS ви можете застосувати певні правила форматування та стилі до кількох сторінок за допомогою одного рядка коду. Одну каскадну таблицю стилів можна відтворити на кількох сторінках веб-сайту. Якщо, наприклад, у вас є сторінки продуктів, які мають однакове форматування, вигляд і відчуття, написання правил CSS для однієї сторінки буде достатньо для всіх сторінок того самого типу.

4) Прості зміни форматування - Якщо вам потрібно змінити формат певного набору сторінок, це легко зробити за допомогою CSS. Немає необхідності виправляти кожну окрему сторінку. Просто відредагуйте відповідну таблицю стилів CSS, і ви побачите зміни, застосовані до всіх сторінок, які використовують цю таблицю стилів.

5) Сумісність між пристроями - Адаптивний веб-дизайн має значення. У наш час веб-сторінки мають бути повністю видимими та легко навігаційними на всіх пристроях. Будь то мобільний пристрій чи планшет, настільний комп’ютер або навіть смарт-телевізор, CSS поєднується з HTML, щоб зробити адаптивний дизайн можливим.

Щоб зрозуміти основи роботи CSS, ви повинні спочатку трохи зрозуміти сучасний HTML. Веб-розробники розкладають сторінки відповідно до

«коробкової моделі». Веб-сторінка — це серія блоків, кожна з яких містить дискретний елемент. Ці коробки вкладені один в інший.

Наприклад, заголовок сторінки — це поле, і воно містить кілька менших блоків, які містять усі елементи, які утворюють заголовок: логотип, навігацію, кнопки соціальних мереж тощо. Наприклад за допомогою CSS розробник призначає стилі до "заголовок", припустимо, що розробник робить текст всередині заголовка фіолетовим, шрифт Arial і п’ятнадцять пунктів. Ось де в гру вступає «каскадна» частина каскадних таблиць стилів. Стилі шрифтів, застосовані до заголовка, каскадно опускаються до всіх елементів, що містяться всередині заголовка. Усі елементи, що містять текст, як-от навігація, посилання або заклики до дії, мають фіолетовий колір, Arial і п’ятнадцять балів.

Специфікації CSS підтримує Всесвітній консорціум із веб-сторінок (W3C). Текст CSS типу Інтернет-медіа (тип MIME) зареєстрований для використання в CSS RFC 2318 (березень 1998 р.).[18-19]

Опис Atom

Atom - це кросплатформний текстовий редактор, розроблений для того, щоб бути достатньо універсальним, щоб його можна було використовувати для будь-чого, від простого тексту до цілих проектів розробки програмного забезпечення. Щорічні опитування Stack Overflow у 2016 році повідомляли , що Atom використовували 12,5% розробників програмного забезпечення на основі відповідей понад 46 000 людей. Наступного року Atom використовували 20% веб-розробників, 20,7% системних адміністраторів і 15,9% спеціалістів з обробки даних. Приблизно в той час, коли Visual Studio Code почав зростати в популярності, який є більш-менш офіційним наступником. Atom - був розроблений Microsoft, власником GitHub, і має багато інтегрованих функцій GitHub.

Atom також завоював місце в історії обчислень як перша велика програма, яка використовує фреймворк Electron , який зараз є одним із найпопулярніших способів розробки кросплатформних настільних додатків. Slack, Discord, Skype,

Facebook Messenger та незліченна кількість інших програм для настільних комп’ютерів створені за допомогою Electron.

Також Atom можна модифікувати завантажуючи так звані «пакети» які можуть розширювати функціонал самої програми та покращувати роботу. Це означає, що пакети можуть бути неймовірно потужними і можуть змінити все, від самого зовнішнього вигляду всього інтерфейсу до базової роботи і навіть основної функціональності.[21-22]

Опис MS SQL Server

MS SQL Server — це система керування реляційними базами даних (RDBMS), розроблена Microsoft. Цей продукт створений для базової функції зберігання даних, які вимагають інші програми. Як і в інших популярних СУБД для запитів використовують мову структурованих запитів (SQL) - це мова бази даних, розроблена для керування даними, що зберігаються в системі керування реляційною базою даних. [42]

2.4. Опис структури системи та алгоритмів її функціонування

З часом Інтернет змінився на активну взаємодію з користувачами, а також на розширену функціональність за допомогою візуально приємних та потужних веб-додатків. Сучасні сайти інтерактивні і динамічні, тобто вони реагують на дії користувача, обробляють його запити і видають результат. Для створення інтерактивних і динамічних сайтів зазвичай використовується архітектурний шаблон MVC.

Шаблон MVC був винайдений Трігве Реєнскаугом, коли він був запрошеним науковцем у групі Smalltalk у відомому дослідницькому центрі Xerox Palo Alto. Свою першу статтю про MVC він написав у 1978 році. Спочатку він назвав її шаблоном Thing Model View Editor, але швидко змінив назву шаблону на шаблон Model View Controller.

У сфері веб-розробки, Model-View-Controller є одним з найбільш обговорюваних шаблонів дизайну у світі веб-програмування сьогодні.

Наприкінці 2000-х і на початку 2010-х років перше покоління фреймворків інтерфейсу користувача, таких як AngularJS, Backbone, Dojo Toolkit, Ember, JavaScriptMVC і Knockout, представило моделі Model-View-Controller і Model-View-View Model (MVVM) для розробка веб-сайтів на стороні клієнта. Це дозволило нам відокремити логіку презентації від стану програми, зберігаючи при цьому горизонтальні рівні програмного забезпечення синхронізованими.

Поточне покоління фреймворків інтерфейсу користувача, таких як Angular, Aurelia, Dojo, Inferno, Preact, React, Svelte і Vue, базуються на компонентах. Вони зосереджені на віджетах інтерфейсу користувача, але деталі відокремлення презентаційних шарів від решти програми залишають за нами.

Шаблон MVC був оголошений багатьма розробниками як корисний шаблон для повторного використання об’єктного коду і шаблон, який дозволяє їм значно скоротити час, необхідний для розробки додатків з користувацькими інтерфейсами.

Шаблонь Model-View-Controller пропонує три основні компоненти або об'єкти, які будуть використовуватися при розробці програмного забезпечення:

- Model представляє основну логічну структуру даних у програмному додатку та пов’язаний з ним клас високого рівня. Ця об’єктна модель не містить жодної інформації про інтерфейс користувача.
- View являє собою набір класів, що представляють елементи в інтерфейсі користувача (все те, що користувач може бачити на екрані та на що реагувати, наприклад кнопки, вікна відображення тощо)
- Controller який представляє класи , що з'єднують модель і подання, і використовується для зв'язку між класами в моделі і представленні.

Архітектура MVC (рис. 2.3.)

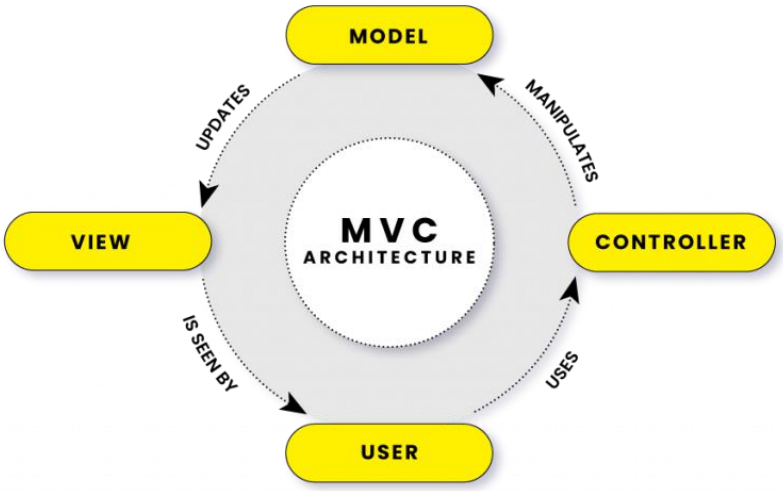


Рис. 2.1. Архітектура MVC додатку.

Model - керують даними для програми. Вони не стосуються ні інтерфейсу користувача, ні рівня презентації, а натомість представляють унікальні форми даних, які можуть знадобитися додатку. Коли модель змінюється (наприклад, коли вона оновлюється), вона зазвичай сповіщає своїх спостерігачів (наприклад, погляди, концепцію, яку ми розглянемо найближчим часом), що відбулася зміна, щоб вони могли відреагувати відповідним чином.

Вбудовані можливості моделей різняться в різних фреймворках, однак вони досить часто підтримують перевірку атрибутів, де атрибути представляють властивості моделі, наприклад, ідентифікатор моделі. Використовуючи моделі в реальних програмах, ми, як правило, також бажаємо зберігати модель. Постійність дозволяє нам редагувати та оновлювати моделі, знаючи, що останній стан буде збережено в пам’яті, у localStorage сховищі даних користувача або синхронізовано з базою даних.

Нерідко сучасні фреймворки MVC/MV* забезпечують засоби для групування моделей (наприклад, у Backbone ці групи називаються «колекціями»). Керування моделями в групах дозволяє нам писати логіку програми на основі сповіщень від групи, якщо будь-яка модель, яку вона містить, буде змінена. Це дозволяє уникнути необхідності вручну спостерігати за окремими екземплярами моделі.

View - це візуальне представлення моделей, які представляють відфільтроване уявлення про їх поточний стан. У той час як уявлення Smalltalk стосуються малювання та підтримки растрового зображення, представлення JavaScript стосуються створення та підтримки елемента DOM..

Подання зазвичай спостерігає за моделлю і отримує сповіщення, коли модель змінюється, що дозволяє подання відповідно оновлюватися. У літературі про шаблони проектування погляди зазвичай називають «тупими», враховуючи, що їхні знання про моделі та контролери в додатку обмежені.

Користувачі можуть взаємодіяти з представленнями, і це включає в себе можливість читати та редагувати (тобто отримувати або встановлювати значення атрибутів у) моделі. Оскільки представлення є шаром презентації, ми зазвичай представляємо можливість редагування та оновлення зручним для користувача способом.

Controller є посередниками між моделями та представленнями, які, як правило, відповідають за оновлення моделі, коли користувач маніпулює поданням.

Controller полегшує нам керування сценаріями. Це тому, що Controller містить функції, які ми можемо програмувати, як завгодно. Наприклад, після отримання даних у нашому **UserController** у методі **Store** ми просто створимо обліковий запис користувача. Після створення облікового запису користувача ми перевіримо, чи є якісь завдання, які користувач надіслав разом із іменем та електронною поштою.[30-35]

У MVC є також свої недоліки, а саме:

• Необхідність використання більшої кількості ресурсів. Складність в тому що всі основні блоки є абсолютно незалежними та взаємодіють між собою одночасно. Контролер повинен завжди завантажувати та при необхідності створювати всі можливі комбінації змінних і передавати їх в Model. Тоді коли Model, в свою чергу, повинен завантажувати всі дані для візуалізації та передавати їх во View.

• Ускладнений механізм розділення програм на модулі. В концепції MVC наявність блоків прописано жорстко, кожен функціональний модуль повинен складатись з трьох блоків що ускладнює можливість використовувати архітектуру модулів програми.

• Використання розширювальних функціональних процесів. Недостатньо просто написти функціональний модуль та підключити його в одному місці програми. Кожний функціональний модуль повинен складатися з трьох частин, а кожна з цих частин повинна бути підключена до відповідного блоку.

Проблема в тому, що вся система з часом стає крихкою та нестабільною коли кількість models і views зростає (рис. 2.4.).

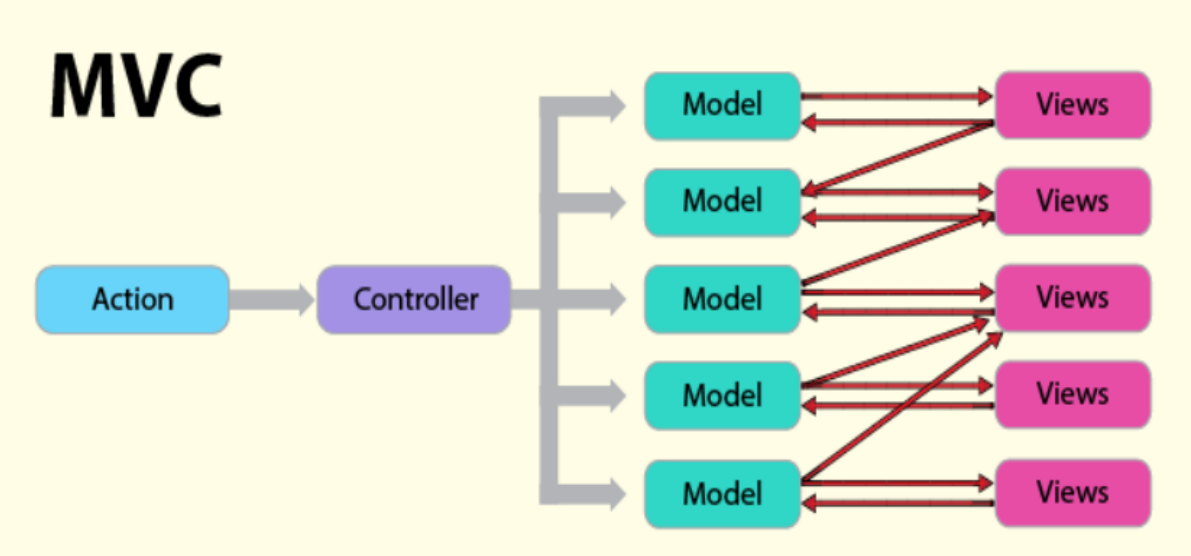


Рис. 2.2. Приклад того як зростає кількість models та views з додаванням нового функціоналу.

Тому було розроблено, як альтернативу, або заміну, архітектуру яка має назву Flux.

Flux - це архітектура JavaScript або шаблон для інтерфейсу користувача, який працює в односпрямованому потоці даних і має централізований диспетчер. Це архітектура програми, розроблена для створення веб-програм на стороні клієнта. Це також одна з популярних архітектур, які Facebook використовує для створення клієнтських веб-додатків.

Також можна сказати, що Flux більше ніж шаблон, більше ніж фреймворк і має 4 головних компонента.

Архітектура Flux базується на наступних компонентах:

- Диспетчер (Dispatcher).
- Сховище (Stores).
- Вид (Views)(React компонент).
- Дія (Action).

Архітектуру Flux можна представити в такому вигляді (рис. 2.5.):

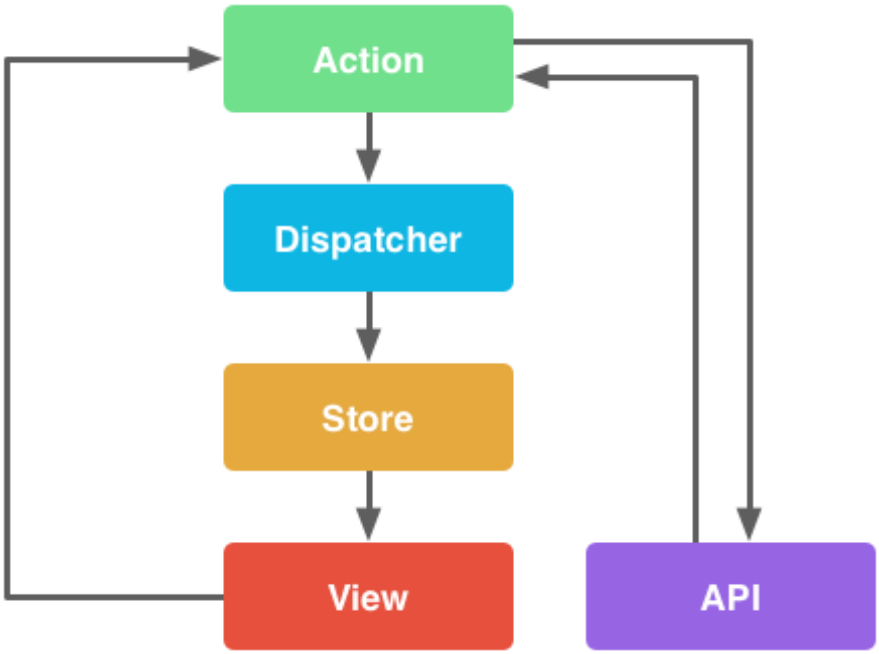


Рис. 2.3. архітектура Flux.

Flux має відкритий вихідний код, і це скоріше шаблон проектування, а не фреймворк, тому його можна використовувати відразу. Розробники можуть створювати додатки, не турбуючись про складні взаємодії між джерелами даних.

Архітектура Flux корисна для дій які включають ефект покращення чіткості коду, оновлення інших уявлень та налагодження новими розробниками. Він також включає в себе одиночний диспетчер, і всі дії проходять через диспетчера. Диспетчер є центром, який керує всім потоком даних, це реєстр зворотних викликів до сховищ, які не мають власного інтелекту:

- Це епіцентр потоку даних у будь-якій програмі Flux

- Він контролює те, що надходить у Сховище програми Flux.
- Він служить місцем розміщення для зворотних викликів, які створюються магазинами і пов’язані з диспетчером.
- Кожене Сховище у програмі створює зворотний виклик, який реєструє його у диспетчера.
- Коли користувач надсилає нову дію диспетчеру, диспетчер гарантує, що завдяки зворотному виклику всі зареєстровані сховища отримають цю дію.

Сховище (Store) - є центром стану та логіки програми, роль схожа на модель у традиційному MVC. Він реєструється в диспетчері і надає йому зворотний виклик, який отримує дію як параметр.

Архітектура Flux використовує Stores для кешування будь-яких програм, пов’язаних з даними або станом. Він також має можливості логічної обробки та гнучкість, щоб вирішувати, які частини ваших даних показувати публічно.

Дія (Action) - набір методів, які викликаються всередині, або всередині представлень для надсилання дій диспетчеру. Це одне з фактичних корисних вантажів, які доставляються через диспетчера. Набори прикладів типу дії використовуються для визначення дії, яка має відбутися, і надсилаються разом із даними дії.

Коли диспетчер відкриває метод, який дозволяє нам ініціювати відправку до сховищ і включати корисне навантаження даних, що ми називаємо дією. Дії також можуть надходити з інших місць, таких як сервери, наприклад, типи даних під час ініціалізації або коли сервер повертає код помилки, або коли сервери мають оновлення для надання програми.

Вид (Views) - Це не технічна частина Flux, але в той же час перегляди, очевидно, є важливою частиною нашої програми. Здебільшого це розуміється як частина нашої архітектури, яка відповідає за відображення даних користувачеві. Це остання зупинка нашої архітектури потоку даних. Погляди існують за межами flux, але обмежені односпрямованим характером Flux.

Розглянемо стандартну архітектуру Flux більш детально на (рис. 2.6.)

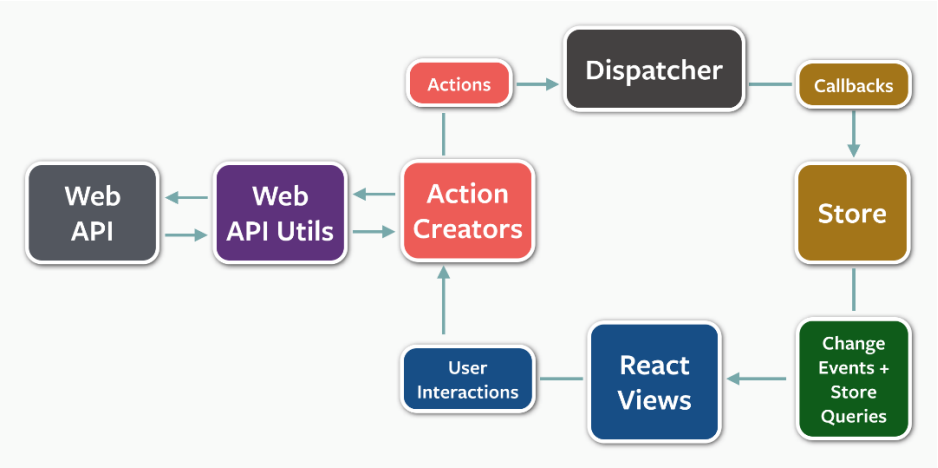


Рис. 2.4. стандартна архітектура Flux.

Ні один великий проект зараз не обходиться без використання великих файлових систем які відповідають за збереження даних, починаючи с простої реєстрації, закінчуючи збереженням різних даних та файлів. Бази даних і технології баз даних відіграють важливу роль у більшості сфер використання комп'ютерів, включаючи бізнес, освіту та медицину.[25-30]

Найкращий спосіб зберігати дані – це використовувати програмне забезпечення для керування базами даних. Це потужний програмний інструмент, який дозволяє зберігати, маніпулювати та отримувати дані різними способами.

Більшість компаній відстежують інформацію про клієнтів, зберігаючи її в базі даних. Ці дані можуть включати клієнтів, співробітників, продукти, замовлення або будь-що інше.

Базу даних можна розглядати як свого роду електронну картотеку що містить дані, і зберігається у певному постійному сховищі. Користувачі можуть вставляти нову інформацію в базу даних, видаляти, змінювати або отримувати наявну інформацію в базі даних, надаючи *запити* програмному забезпеченню, яке керує базою даних, тобто системі керування базою даних, тобто СУБД.

Серед властивостей бази даних є:

- Відображення сукупності елементів даних, що представляють інформацію реального світу.
- База даних завжди логічна, узгоджена і внутрішньо несуперечлива.

• База даних проектується, створюється і наповнюється даними для певної мети.

• Кожен елемент даних зберігається в полі.

• Комбінація полів складає таблицю. Наприклад, кожне поле в таблиці співробітників містить дані про окремого працівника.

Система управління базами даних (СУБД) - це набір програм, які дозволяють користувачам створювати і підтримувати бази даних і контролювати весь доступ до них. Основною метою СУБД є забезпечення середовища, яке є зручним і ефективним для користувачів для отримання та зберігання інформації.

Використовуючи СУБД, інформація, яку ми збираємо та додаємо до її бази даних, більше не підлягає випадковій дезорганізації. Вона стає доступнішою та інтегрованою з рештою нашої роботи. Управління інформацією за допомогою бази даних дозволяє нам стати стратегічними користувачами даних, які ми маємо.

Через універсальність баз даних вони можуть забезпечити потреби усіх видів проектів. База даних може бути пов’язана з:

• Веб-сайтом, який збирає зареєстрованих користувачів.

• Додатком для відстеження клієнтів, або для організацій соціальних служб.

• Систему медичної документації закладам охорони здоров’я.

• Особиста адресна книга у вашому поштовому клієнті.

• Великий збірник текстових документів.

• Система бронювання авіакомпаній та інші.

Найпопулярнішою моделлю даних, яка використовується сьогодні, є реляційна модель даних. Цю модель підтримують відомі СУБД, такі як Oracle, MS SQL Server, DB2 і MySQL. Інші традиційні моделі, такі як ієрархічні моделі даних і моделі мережевих даних, все ще використовуються в промисловості в основному на платформах мейнфрейма. Однак вони не часто використовуються через їх складність.

Реляційна модель даних була введена у 1970 році Е. Ф. Коддом. Зараз вона є найбільш широко використовуваною моделлю даних. Реляційна модель даних описує світ як «набір взаємопов'язаних таблиць».

Таблиця – це підмножина декартового добутку списку доменів, що характеризуються іменем. В таблиці кожен рядок представляє групу пов'язаних значень даних. Стовпці в таблиці є полем і також називаються атрибутом.

Зазвичай база даних складається з кількох таблиць, і кожна таблиця містить дані. На Рис. 2.5 представлено приклад зв'язку таблиць реляційної моделі.

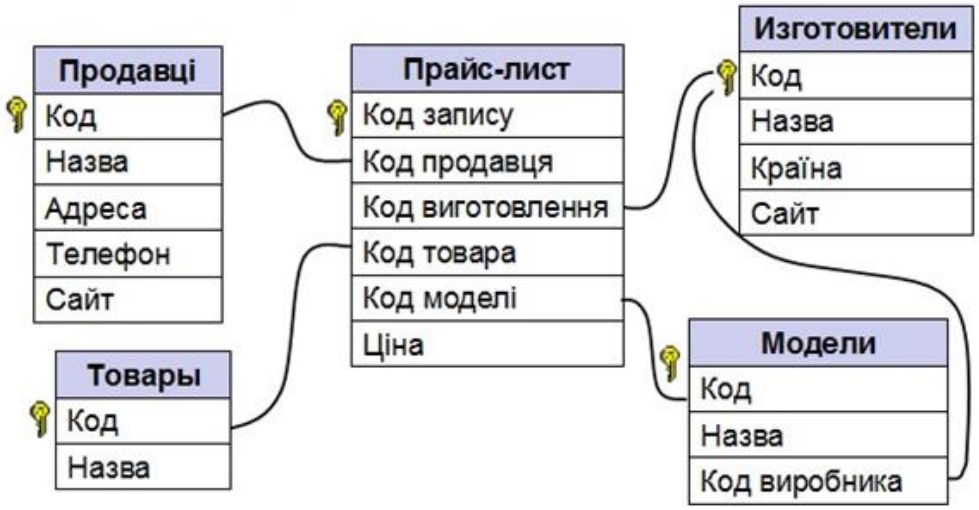


Рис. 2.5 Зв'язок таблиць реляційної моделі.

Основні одиниці зберігання називаються стовпцями полями або атрибутами . Вони містять в собі основні компоненти даних, на які можна розбити ваш вміст. Вирішуючи над створенням полів ви повинні думати про свою інформацію узагальнено, взяти загальні компоненти інформації, яку ви будете зберігати в базі даних, і уникати особливостей, які відрізняють один елемент від іншого.

Подібно до того, як вміст будь-якого окремого документа чи елемента потрібно розбити на складові частини даних для зберігання в полях, зв'язок між ними також має бути доступним, щоб їх можна було відновити у своїй цілісній формі. Записи дозволяють нам це зробити.

Записи містять пов’язані поля, наприклад, клієнт або співробітник, записи та поля складають основу всіх баз даних, наприклад проста таблиця дає нам картину того, як записи та поля працюють разом в базі даних.

Основним аспектом розробки програмного забезпечення є поділ процесу розробки на серію фаз або кроків, кожен з яких зосереджений на одному аспекті розробки. Набір цих кроків іноді називають життєвим циклом розробки програмного забезпечення (SDLC) . Програмний продукт проходить через цей життєвий цикл і іноді неодноразово, коли його вдосконалюють або переробляють доки його остаточно не припиняють використовувати.

Ми можемо використовувати цикл водоспаду як основу для моделі розробки бази даних, яка включає три припущення:

1. Можемо відокремити розробку бази даних, тобто специфікацію та створення схеми для визначення даних у базі даних, від процесів користувача, які використовують базу даних.

2. Також можемо використовувати архітектуру трьох схем як основу для розрізнення діяльності, пов’язаної зі схемою.

3. Та можемо представити обмеження для забезпечення семантики даних один раз у базі даних, а не в кожному процесі користувача, який використовує дані.

Модель даних відношення сутності (ER) існує понад 35 років, вона добре підходить для моделювання даних та для використання з базами даних, оскільки є досить абстрактною і його легко обговорювати та пояснювати. Моделі ER легко перекладаються на відносини. Моделі ER, також названі схемою ER, представлені діаграмами ER.

Моделювання ER базується на двох концепціях:

- Сутності , визначені як таблиці, які містять конкретні дані.
- Відносини, що визначаються як асоціації або взаємодії між сутностями.

Важливим обмеженням для сутності є ключ. Ключ — це атрибут або група атрибутів, значення яких можна використовувати для однозначної ідентифікації окремої сутності в наборі сутностей.

Відносини - це клей, який скріплює таблиці. Вони використовуються для з’єднання пов’язаної інформації між таблицями.

Міцність зв’язку залежить від того, як визначається первинний ключ пов’язаної сутності. Слабкий або неідентифікуючий зв’язок існує, якщо первинний ключ пов’язаної сутності не містить компонент первинного ключа батьківського об’єкта.

Одна важлива теорія, розроблена для реляційної моделі сутності (ER), включає поняття функціональної залежності (FD). Метою вивчення цього є покращити ваше розуміння взаємозв’язків між даними та отримати достатній формалізм, щоб допомогти у практичному проектуванні бази даних. У розробці бази даних надмірність, як правило, небажана, оскільки це спричиняє проблеми зі збереженням узгодженості після оновлення. Однак надмірність іноді може призвести до підвищення продуктивності; наприклад, коли замість об'єднання можна використовувати резервування для з'єднання даних. Об’єднання використовується, коли потрібно отримати інформацію на основі двох пов’язаних таблиць .

Ці всі таблиці, ключі та зв’язки прописуються в СУБД за допомогою запитів на мові структурованих запитів (SQL) - це мова бази даних, розроблена для керування даними, що зберігаються в системі керування реляційною базою даних. SQL спочатку був розроблений IBM на початку 1970-х років. Початкова версія була названа SEQUEL (Structured English Query Language) була розроблена для маніпулювання та отримання даних, що зберігаються в системі керування квазіреляційною базою даних IBM System R. Потім наприкінці 1970х Relational Software Inc., яка зараз є Oracle Corporation, представила першу комерційно доступну реалізацію SQL, Oracle V2 для комп’ютерів VAX.

Багато з доступних на даний момент реляційних СУБД, таких як Oracle Database, Microsoft SQL Server, MySQL, IBM DB2, IBM Informix і Microsoft Access, використовують SQL.

У СУБД мова бази даних SQL використовується для:

- Створіть бази даних і структури таблиць

• Виконуйте основні завдання керування даними (додавання, видалення та змінення)

• Виконуйте складні запити, щоб перетворити вихідні дані в корисну інформацію

Розробка додатків для баз даних — це процес отримання реальних вимог, аналізу вимог, проектування даних і функцій системи, а потім виконання операцій у системі. Після створення бази даних і додатку до неї, існує два способи заповнення таблиць – або з наявних даних, або за допомогою додатку створеного для бази даних і покращення взаємодії з користувачем.[42]

Тож для реалізації дипломного проекту було розроблено та створено базу даних, яка являє собою основу усієї системи, і в яку збираються і виводяться дані за допомогою Веб додатку.

Для кращої демонстрації БД була створена ER діаграма що показує вміст бази с таблицями та їх зв’язками (Рис. 2.6)

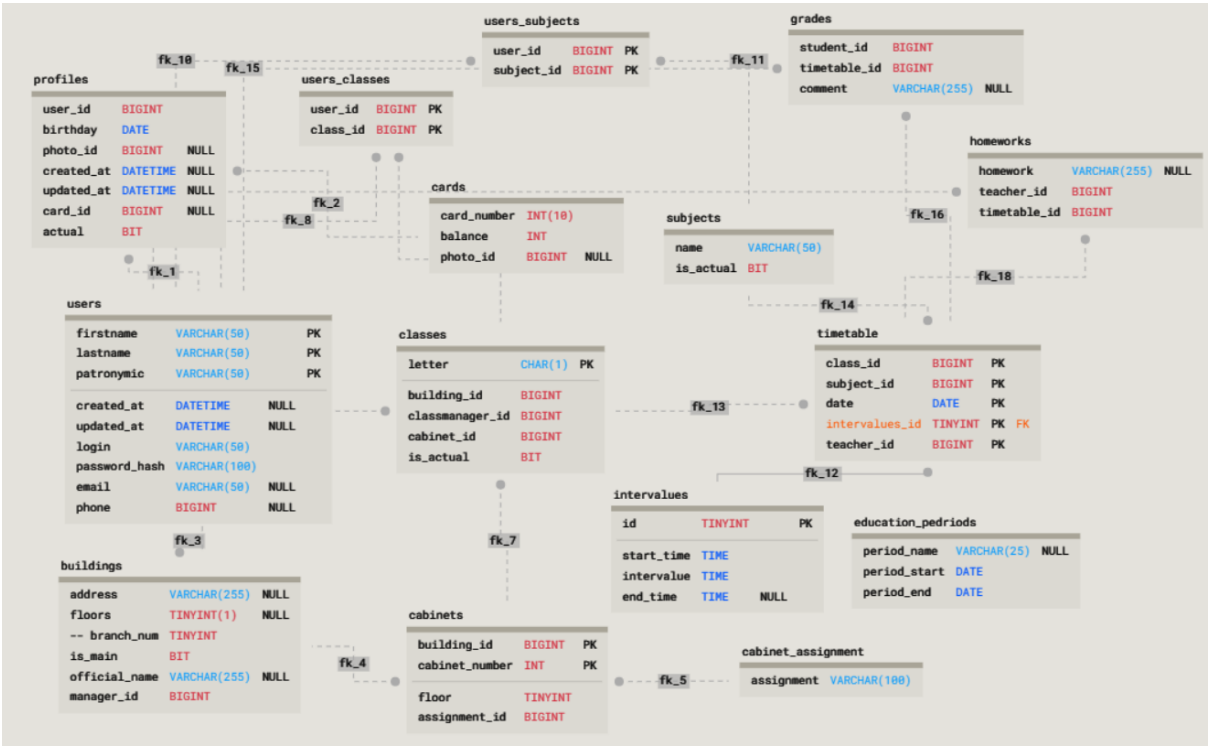


Рис. 2.6 ER діаграма створеної БД для реалізації інформаційної системи

Данна база даних та її діаграма була розроблена та створена за допомогою MS SQL Server.

Далі за допомогою Atom було створено Веб-додаток для взаємодії з БД (Рис. 2.7).

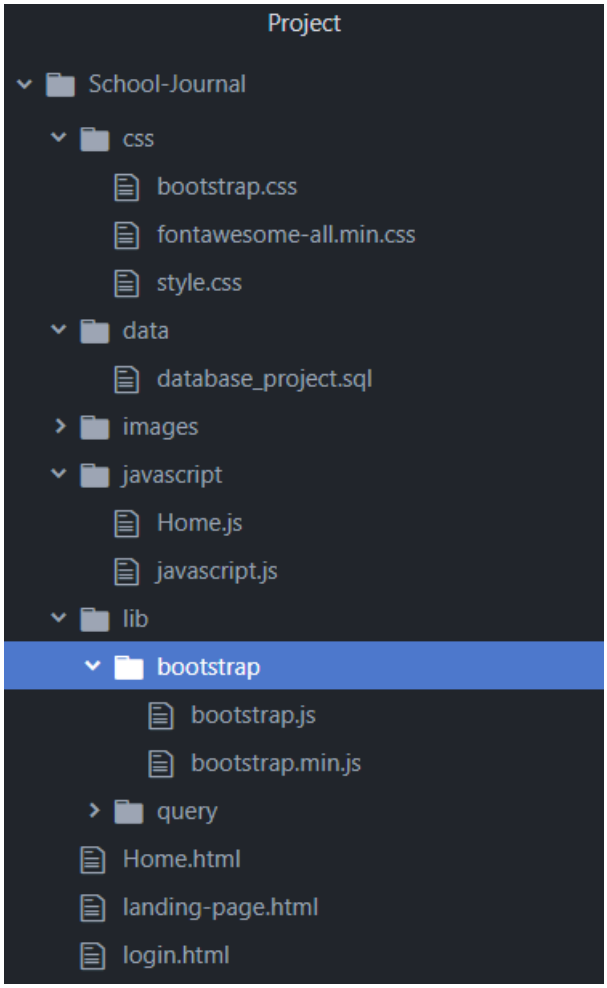


Рис 2.7 Структура створеного Веб додатку

Файли додатка було розділено та структуровано для покращення роботи з ними. Кожен файл відповідає за свою, окрему, функціональність і тому вносити будь-які зміни в проект не складає труднощів.

Також було використано CSS файл fontawesome який надає набір інструментів для використання шрифтів та піктограм на основі CSS.

2.5. Обгрунтування та організація вхідних та вихідних даних програми

Вхідні данні отримуються через заповнювані користувачем компонентів веб-форм а також з місця розташування даних.

Вихідні данні представлені в виді:

- Веб-сторінок додатку.
- Інтерактивних компонентів системи.

2.6. Опис роботи розробленої системи

2.6.1. Використані технічні засоби

Так як вся інформація зберігається на стороні клієнта локально, то наступні рекомендовані характеристики обчислювальної машини наведені виключно для клієнтського обладнання:

- Процесор Intel(R) Core(TM) i3-2348M з тактовою частотою 2.3GHz.
- Оперативна пам’ять в розмірі 4ГБ пам’яті.
- Вільного місця на диску в розмірі 1ГБ пам’яті.
- Рідкокристалічний монітор з діагоналлю не менше 17 ".
- Маніпулятор «миша».
- Клавіатура.
- Доступ до глобальної мережі.

Вище наведені характеристики являють собою рекомендовані. Це означає, що при наявності характеристик не нижче зазначених, розроблений додаток буде функціонувати відповідно до вимог щодо надійності, безпеки та швидкості обробки даних.

2.6.2. Використані програмні засоби

Веб-застосунок був створений за допомогою мов програмування JavaScript та CSS в редакторі Atom. Також База даних була реалізована за допомогою MySQL.

На стороні клієнта потребується лише мінімум програмних засобів, а саме:

- Один з браузерів Firefox або Google Chrome.
- Операційна система Windows 7/10.

2.6.3. Виклик та завантаження програми

Створений веб-додаток використовується онлайн, для його експлуатації необхідний веб-браузер з підтримкою JavaScript. Зараз додаток являє собою функціонуючий макет для особливого користування і його неможливо знайти в інтернеті.

2.6.4. Опис інтерфейсу користувача

Для того щоб розпочати роботу з веб-додатком потрібно запустити сам проект за допомогою локального сервера, або використовуючи хостинг. Після цього користувача буде направлено на сторінку з привітанням(рис. 2.8.). Яка представляє собою сторінку на якій вам пропонують увійти у систему. Реєстрування відбувається адміністратором школи, та на початку навчання учням видаються данні для входу в систему.

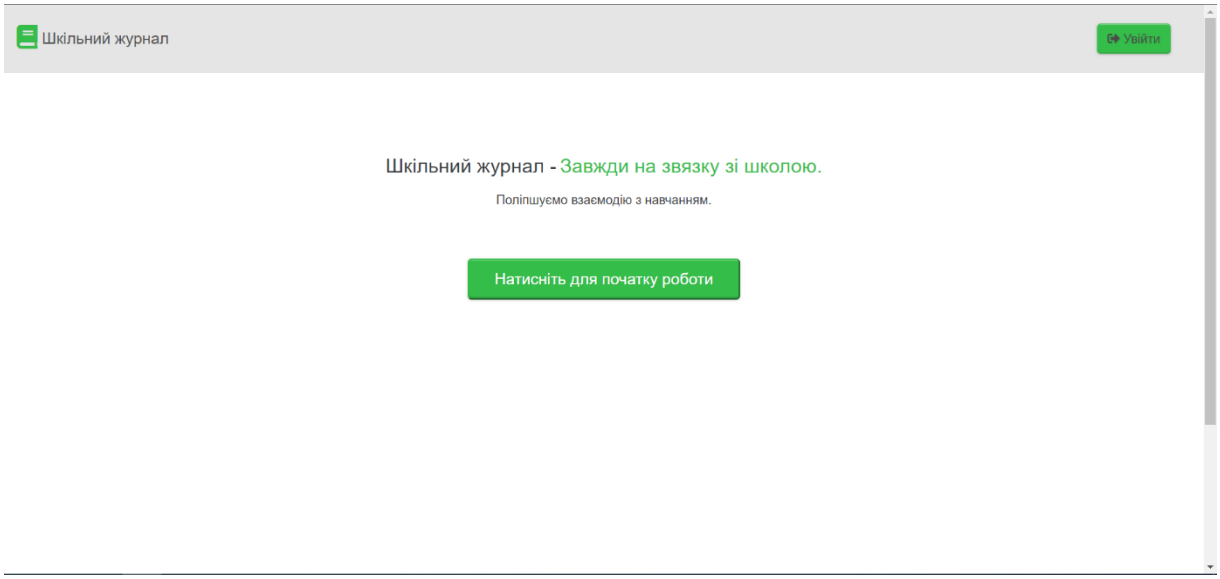


Рис. 2.8. Головна сторінка веб-додатку

Далі після того як користувач тисне на кнопку «Увійти» сайт нас направляє до сторінки з формами для заповнювання логіну та паролю користувача, який він отримав від вчителя. (рис. 2.9.)

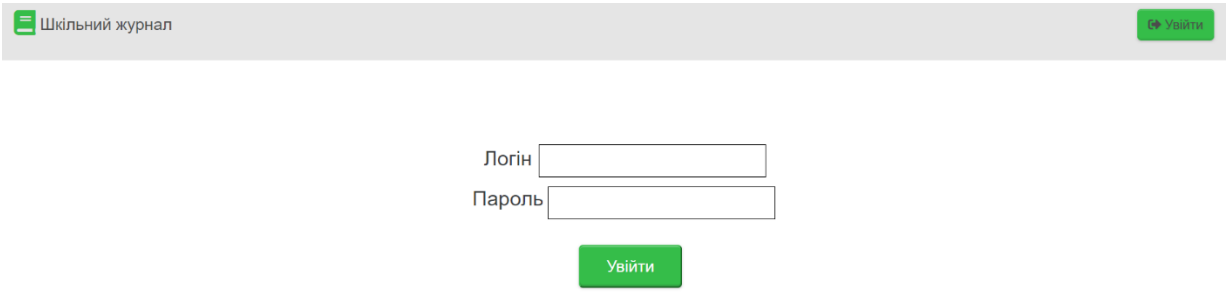


Рис. 2.9. Вигляд сторінки для входу в додаток

Розглянемо послідовно які функції має веб-застосунок, для учнів та вчителів функції різні, наприклад як тільки учень заходить на сайт він одразу бачить розділ «особиста сторінка» куди по запиту з БД виводиться його домашнє завдання на декілька днів вперед. (Рис 2.10.)

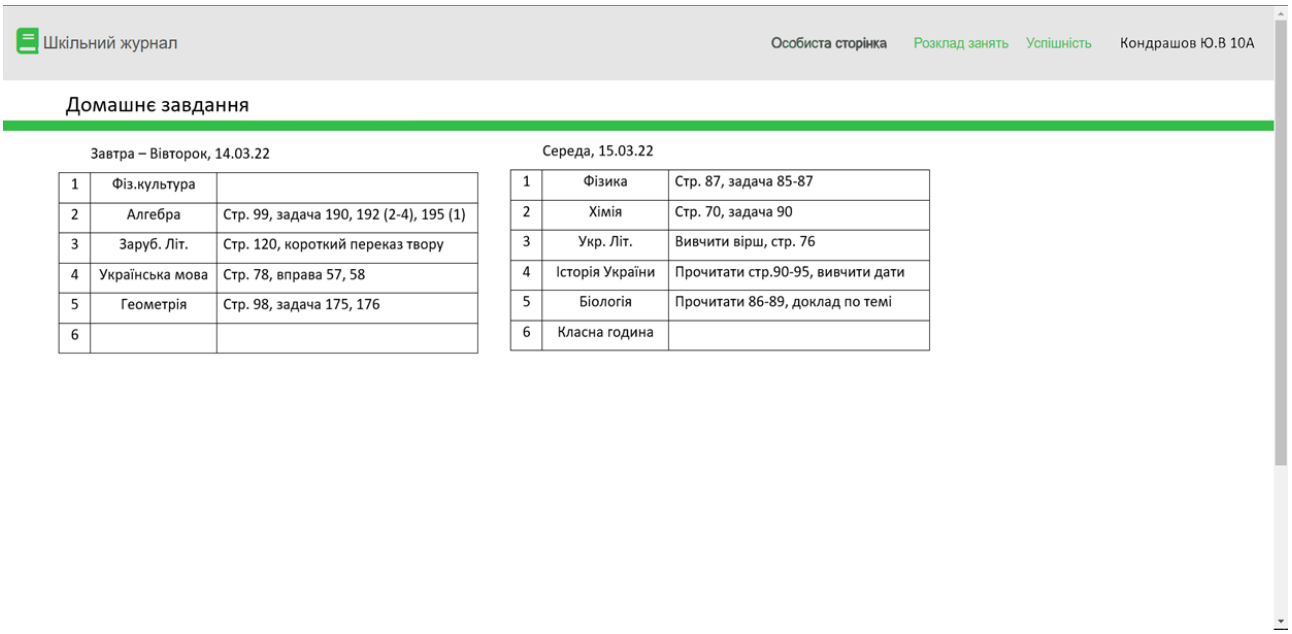


Рис. 2.10. Розділ «особиста сторінка» в якому можна переглянути своє домашнє завдання.

Далі можна перейти до розділу «Розклад занять» (рис. 2.11.) Де користувач може переглянути весь свій розклад без домашнього завдання, а також поверх та номер кабінету в якому проводять заняття.

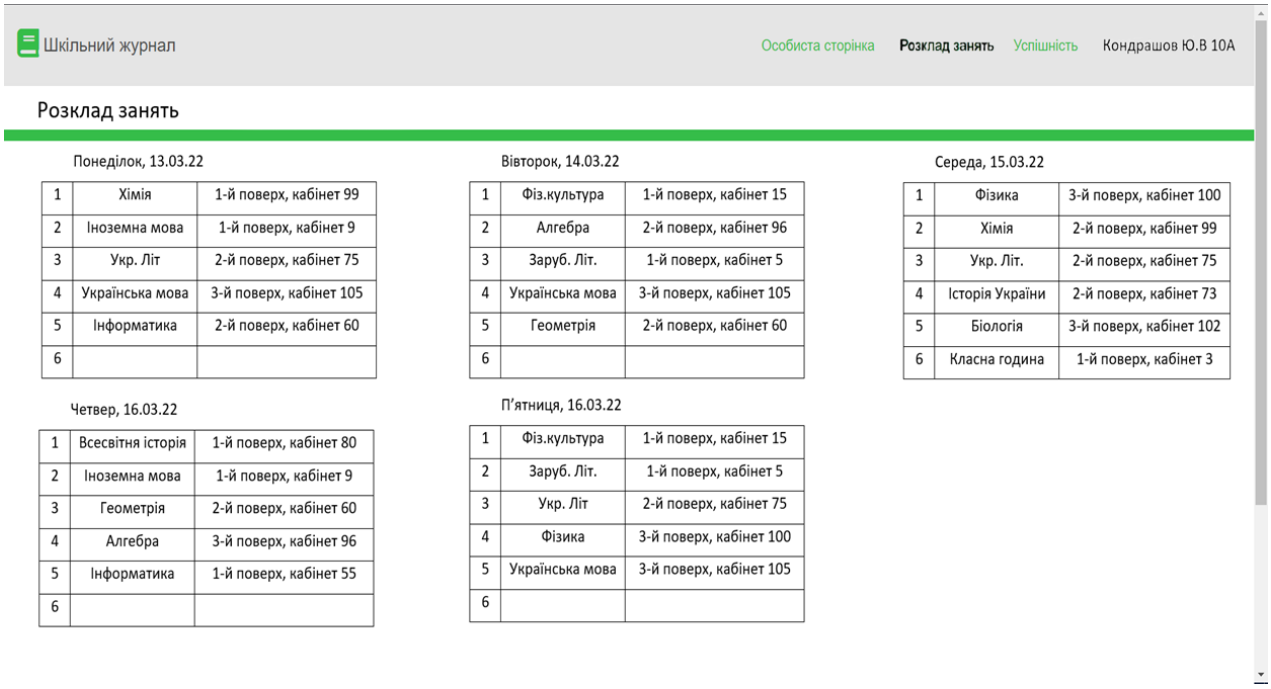


Рис. 2.11. Розклад занять

Натиснувши на «Успішність» користувач може побачити свої успіхи у навчанні, оцінки а також середню оцінку (рис. 2.12.).

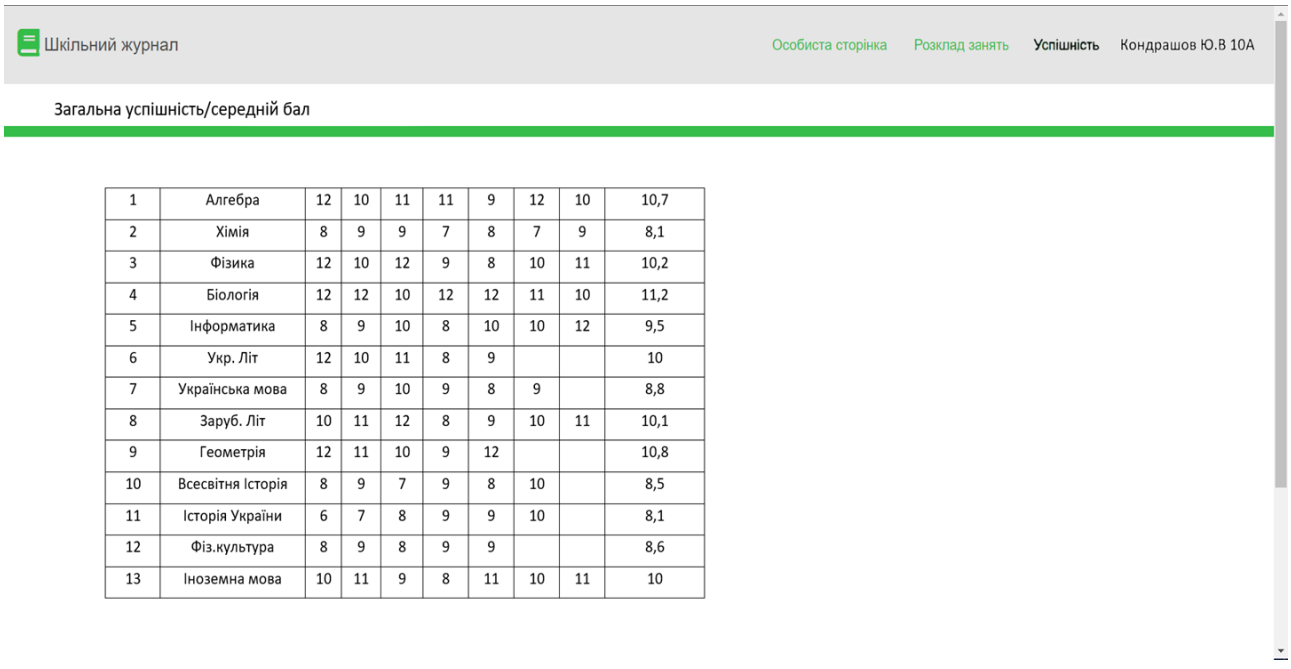
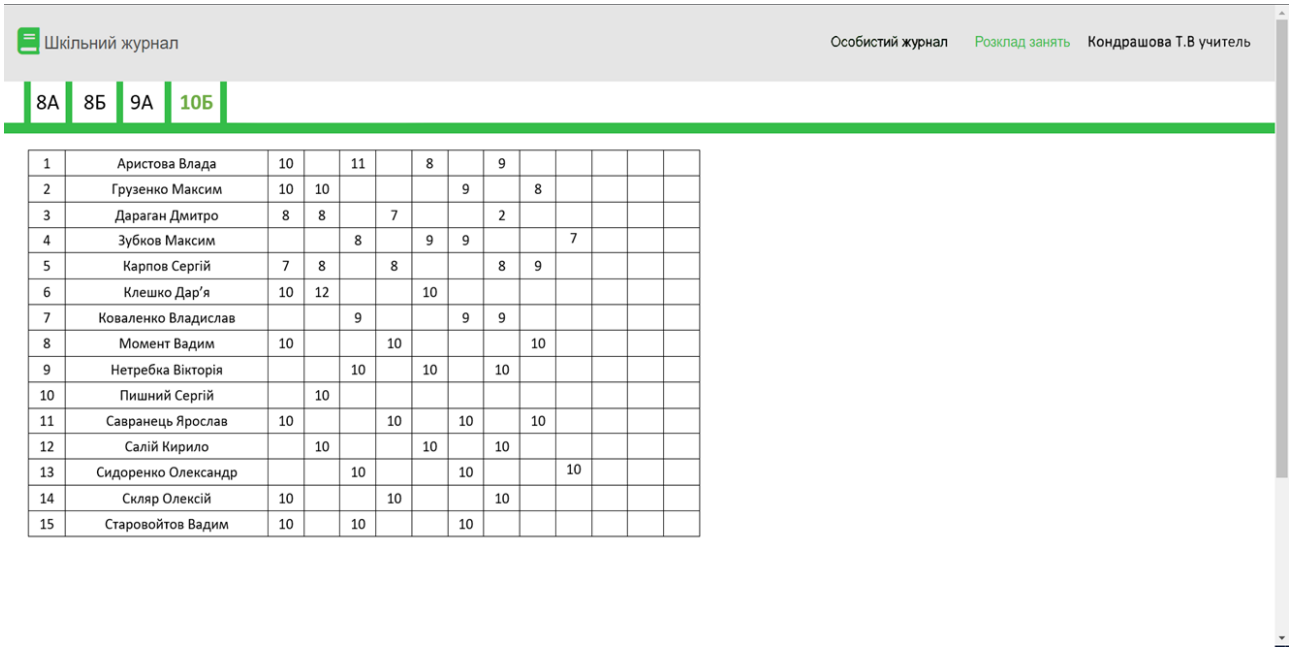


Рис.2.12. Розділ успішність

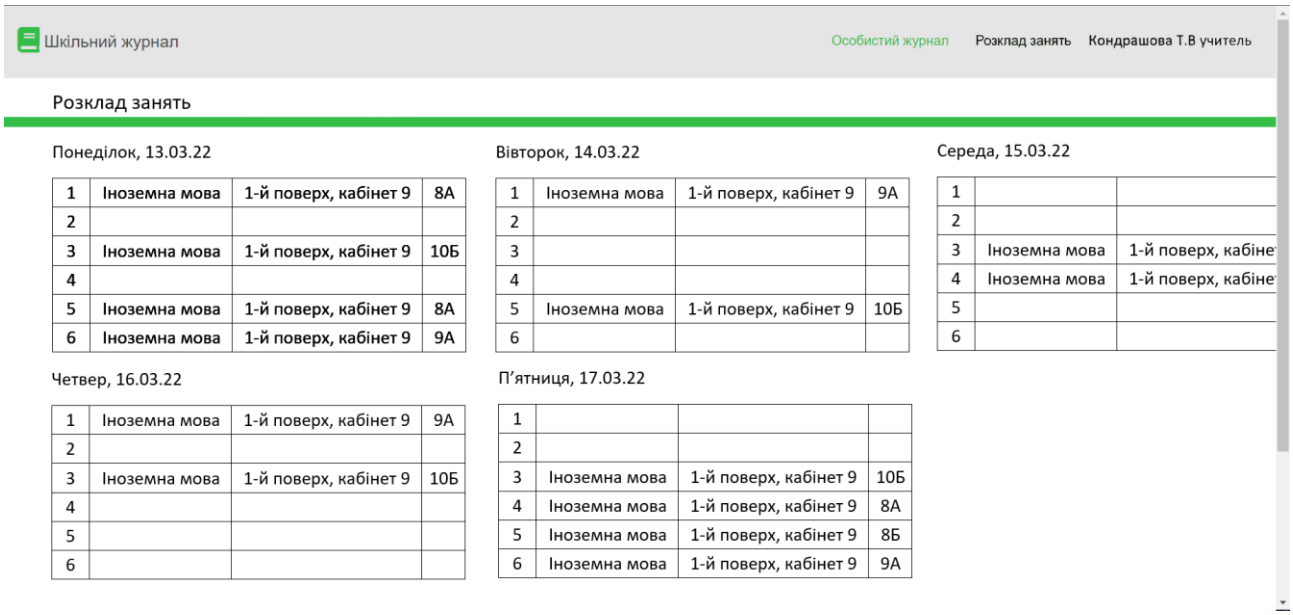
Якщо розглядати можливості вчителів то після входу в систему, перша сторінка що нас зустрічає «особистий журнал», де можна обрати потрібний журнал класу, переглянути список учнів та оцінки. (Рис. 2.13.)



1	Аристова Влада	10		11		8		9				
2	Грузенко Максим	10	10				9		8			
3	Дараган Дмитро	8	8		7			2				
4	Зубков Максим			8		9	9			7		
5	Карпов Сергій	7	8		8			8	9			
6	Клешко Дар'я	10	12			10						
7	Коваленко Владислав			9			9	9				
8	Момент Вадим	10			10				10			
9	Нетребка Вікторія			10		10		10				
10	Пишний Сергій		10									
11	Савранець Ярослав	10			10		10		10			
12	Салій Кирило		10			10		10				
13	Сидоренко Олександр			10			10			10		
14	Скляр Олексій	10			10			10				
15	Старовойтов Вадим	10		10			10					

Рис. 2.13. Зовнішній вигляд «особистого журналу» вчителя

Далі переключившись на «розклад занять» ми бачмо розклад занять учителя, де вказаний номер уроку, назва уроку, кабінет в якому його проводитимуть та номер класу в якого буде заняття. (Рис. 2.14.)



Понеділок, 13.03.22

1	Іноземна мова	1-й поверх, кабінет 9	8А
2			
3	Іноземна мова	1-й поверх, кабінет 9	10Б
4			
5	Іноземна мова	1-й поверх, кабінет 9	8А
6	Іноземна мова	1-й поверх, кабінет 9	9А

Вівторок, 14.03.22

1	Іноземна мова	1-й поверх, кабінет 9	9А
2			
3			
4			
5	Іноземна мова	1-й поверх, кабінет 9	10Б
6			

Середа, 15.03.22

1		
2		
3	Іноземна мова	1-й поверх, кабіне
4	Іноземна мова	1-й поверх, кабіне
5		
6		

Четвер, 16.03.22

1	Іноземна мова	1-й поверх, кабінет 9	9А
2			
3	Іноземна мова	1-й поверх, кабінет 9	10Б
4			
5			
6			

П'ятниця, 17.03.22

1			
2			
3	Іноземна мова	1-й поверх, кабінет 9	10Б
4	Іноземна мова	1-й поверх, кабінет 9	8А
5	Іноземна мова	1-й поверх, кабінет 9	8Б
6	Іноземна мова	1-й поверх, кабінет 9	9А

Рис. 2.14. Розклад занять учителя

РОЗДІЛ 3
ЕКОНОМІЧНА ЧАСТИНА

3.1. Визначення трудомісткості розробки програмного забезпечення

Задані дані:

1. передбачуване число операторів – 500;

2. коефіцієнт складності програми – 3;

3. коефіцієнт корекції програми в ході її розробки – 0,2;

4. годинна заробітна плата програміста, грн/год – 75;

5. коефіцієнт збільшення витрат праці внаслідок недостатнього опису задачі – 1,2;

6. коефіцієнт кваліфікації програміста, обумовлений від стажу роботи з даної спеціальності – 1,0;

7. вартість машино-години ЕОМ, грн/год – 25.

Нормування праці в процесі створення ПЗ істотно ускладнено в силу творчого характеру праці програміста. Тому трудомісткість розробки може бути розрахована на основі системи моделей з різною точністю оцінки.

Трудомісткість розробки ПЗ можна розрахувати за формулою:

$$t = t_o + t_u + t_a + t_n + t_{отл} + t_д, \text{ людино-годин,} \qquad (3.1)$$

де t_o - витрати праці на підготовку й опис поставленої задачі (приймається 52);

t_u - витрати праці на дослідження алгоритму рішення задачі;

t_a - витрати праці на розробку блок-схеми алгоритму;

t_n - витрати праці на програмування по готовій блок-схемі;

$t_{отл}$ - витрати праці на налагодження програми на ЕОМ;

$t_д$ - витрати праці на підготовку документації.

Складові витрати праці визначаються через умовне число операторів у ПЗ, яке розробляється.

Умовне число операторів (підпрограм):

$$Q = q * C * (1 + p), \tag{3.2}$$

де q - передбачуване число операторів;

C - коефіцієнт складності програми;

p - коефіцієнт корекції програми в ході її розробки.

$$Q = 500 * 3 * (1 + 0,2) = 1800.$$

Витрати праці на вивчення опису задачі t_u визначається з урахуванням уточнення опису і кваліфікації програміста:

$$t_u = \frac{Q * B}{(75..85) * k}, \text{ людино-годин.} \tag{3.3}$$

де B - коефіцієнт збільшення витрат праці внаслідок недостатнього опису задачі;

k - коефіцієнт кваліфікації програміста, обумовлений від стажу роботи з даної спеціальності.

$$t_u = \frac{1800 * 1,2}{80 * 1} = \frac{2160}{80} = 27 \text{ людино-годин.}$$

Витрати праці на розробку алгоритму рішення задачі:

$$t_a = \frac{Q}{(20...25) * k}, \text{ людино-годин,} \tag{3.4}$$

$$t_a = \frac{1800}{20 * 1} = 90, \text{ людино-годин.}$$

Витрати на складання програми по готовій блок-схемі:

$$t_n = \frac{Q}{(20...25) * k}, \text{ людино-годин,} \quad (3.5)$$

$$t_n = \frac{1800}{25 * 1} = 72, \text{ людино-годин,}$$

Витрати праці на налагодження програми на ЕОМ:

- за умови автономного налагодження одного завдання:

$$t_{отл} = \frac{Q}{(4..5) * k}, \text{ людино-годин,} \quad (3.6)$$

$$t_{отл} = \frac{1800}{4 * 1} = 450, \text{ людино-годин;}$$

- за умови комплексного налагодження завдання:

$$t^k_{отл} = 1,5 * t_{отл}, \text{ людино-годин,} \quad (3.7)$$

$$t^k_{отл} = 1,5 * 450 = 675, \text{ людино-годин.}$$

Витрати праці на підготовку документації:

$$t_д = t_{др} + t_{до}, \text{ людино-годин,} \quad (3.8)$$

де $t_{др}$ - трудомісткість підготовки матеріалів і рукопису.

$$t_{др} = \frac{Q}{15..20 * k}, \text{ людино-годин,} \quad (3.9)$$

$$t_{др} = \frac{1800}{20 * 1} = 90, \text{ людино-годин.}$$

$t_{до}$ - трудомісткість редагування, печатки й оформлення документації:

$$t_{до} = 0,75 * t_{др}, \text{ людино-годин,} \quad (3.10)$$

$$t_{до} = 0,75 * 90 = 67,5, \text{ людино-годин,}$$

$$t_{д} = 90 + 67,5 = 157,5, \text{ людино-годин.}$$

Тепер розрахуємо трудомісткість ПЗ:

$$t = 52 + 27 + 90 + 72 + 450 + 157,5 = 848,5, \text{ людино-годин.}$$

3.2. Витрати на створення програмного забезпечення

Витрати на створення ПЗ *Кпо* включають витрати на заробітну плату виконавця програми *Зз/п* і витрат машинного часу, необхідного на налагодження програми на ЕОМ:

$$К_{по} = З_{зп} + З_{мв}, \text{ грн.} \quad (3.11)$$

Заробітна плата виконавців визначається за формулою:

$$З_{зп} = t * С_{пр}, \text{ грн,} \quad (3.12)$$

де t - загальна трудомісткість, людино-годин;

$C_{пр}$ - середня годинна заробітна плата програміста, грн/година.

$$З_{зп} = 848,5 * 75 = 63637,5, \text{ грн.}$$

Вартість машинного часу, необхідного для налагодження програми:

$$З_{мв} = t_{отл} * C_{мч}, \text{ грн,} \quad (3.13)$$

де $t_{отл}$ - трудомісткість налагодження програми на ЕОМ, год,

$C_{мч}$ - вартість машино-години ЕОМ, грн/год,

$C_{мч}$ = 25, грн/год.

$$З_{мв} = 450 * 25 = 11250, \text{ грн.}$$

Визначені в такий спосіб витрати на створення програмного забезпечення є частиною одноразових капітальних витрат на створення ПЗ:

$$К_{по} = 63637,5 + 11250 = 74887,5, \text{ грн.}$$

Очікуваний період створення ПЗ:

$$T = \frac{t}{B_k * F_p}, \text{ міс.,} \quad (3.14)$$

де B_k - число виконавців (приймається 1),

F_p - місячний фонд робочого часу (40 годин на тиждень F_p = 176 годин).

$$Т = \frac{848,5}{1 * 176} = 4,82, \text{ міс.}$$

Тож за результатами розрахунків, на проект потрібно приблизно 5 місяців та 74887,5 гривень.

ВИСНОВКИ

В рамках дипломного проекту був розроблений Web сервіс який являє собою електронний щоденник учня для закладів середньої освіти.

Він призначений для того, щоб покращити взяток між учнями та вчителями під час дистанційного навчання та частково виключити потребність у фізичних щоденниках загалом. Використання допоможе користувачам відстежувати свої успіхи у навчанні, полегшити комунікацію між учителем на батьками учня, а також спрощує отримання домашнього завдання.

Під час виконання даного дипломного проекту були виконані наступні задачі:

- Проаналізовано предметну область поставленої задачі.
- Розроблено базу данних для застосунку.
- Спроектована внутрішня архітектура системи.
- Створено клієнтську частину системи.
- Визначено трудомісткість розробленої системи.
- Підрахована вартість додатку та загальний час його розробки.

Розроблений веб-застосунок дозволяє:

- Переглядати записи бази данних за допомогою веб-сайту.
- Можливість зручної навігації між середовищами веб-додатку.
- Зберігати внесені дані.

Програмний продукт реалізований за допомогою мови програмування JavaScript, що був використаний для реалізації клієнтської частини додатку також для візуальної составної веб-сайту був використаний CSS. База данних була спроєктована та створена за допомогою MS SQL Server на мові SQL.

За підрахунками, загальний час на створення проєкту приблизно 5 місяців, а сама праця обійдется у майже 75 тис. гривень.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1) Ames, M. (2018). Hackers, computers, and cooperation: A critical history of Logo and constructionist learning.Proceedings of the ACM on Human-Computer Interaction,2(CSCW), 1-19.
2) Ames, M. (2019) The charisma machine: the life, death, and legacy of one laptop per child. MIT Press
3) Apple, M. and Jungck, S. (1992). You don't have to be a teacher to teach this unit. In Hargreaves, A. and Fullan, M.(eds)Understanding teacher development, Teachers College Press (pp.20-42).
4) Broussard, M. (2019) Artificial unintelligence: how computers misunderstand the world. MIT Press
5) Bubb, S., & Jones, M. (2020). Learning from the COVID-19 home-schooling experience: Listening to pupils, parents/carers and teachers.Improving Schools,23(3), 209-222.
6) Edwards, B. & Cheok, A. (2018). Why not robot teachers: artificial intelligence for addressing teacher shortage.Applied Artificial Intelligence,32(4), 345-360.
7) Emihovich, C., & Miller, G. E. (1988). Effects of Logo and CAI on black first graders' achievement, reflectivity, and self-esteem.The Elementary School Journal,88(5), 473-487.
8) Engzell, P., Frey, A. & Verhagen, M. (2020). Learning inequality during the COVID- pandemic / https://osf.io/download/5f995b4687b7df03233b06fe
9) Eynon, R., & Malmberg, L. (2021). Lifelong learning and the Internet: Who benefits most from learning online?British Journal of Educational Technology,52(2), 569-583.
10) Facer, K. (2012) Personal, relational and beautiful: education, technologies and John MacMurray's philosophy, Oxford Review of Education, 38:6, 709-725
11) Facer, K. (2018) Governing education through anticipation... or, how to avoid being a useful idiot when talking about educational futures. in I. Grosvenor and L.Rasmussen (eds) Governing Education through Design. Routledge

12) Facer, K (2021) Futures in education: towards an ethical practice, Background Paper for the Unesco Futures of Education Commission / https://unesdoc.unesco.org/ark:/48223/pf0000375792

13) Fitzgerald, S., McGrath-Champ, S., Stacey, M., Wilson, R. & Gavin, M. (2019). Intensification of teachers' work under devolution.Journal of Industrial Relations,61(5), 613-636.

14) Fullan, M., Quinn, J., Drummy, M. & Gardner, M. (2020) Education reimagined: thefuture oflearning / https://edudownloads.azureedge.net/msdownloads/Microsoft-EducationReimagined-Paper.pdf?utm_medium=social&utm_source=twitter

15) Gallagher, M. (2019). Moving beyond microwork: Rebundling digital education and reterritorialising digital labour. InEducation and Technological Unemployment(pp.279-296). Springer.

16) Glass, E. (2018). Engaging the knowledge commons: setting up virtual participatory spaces for academic collaboration and community. inDigital humanities, libraries, and partnerships(pp.75-90). Chandos Publishing.

17) Grant, L. (2017)'Don't Use Professional Judgement, Use the Actual Number': The production and performance of educational data practice in an English secondary school (Doctoral thesis). University of Bristol.

18) Grant, L. (forthcoming) 'Reconfiguring education through data: how data practices reconfigure teacher professionalism and curriculum' in, Hepp, A., Jarke, J. and Kramp, L. (eds).The ambivalences of Data Power: New perspectives in critical data studies. Palgrave (Springer).

19) Greenhow, C., Lewin, C., & Staudt Willet, K. (2021). The educational response to Covid-19 across two countries: a critical examination of initial digital pedagogy adoption.Technology, Pedagogy and Education, 1-19.

20) Hao, K. (2019). China has started a grand experiment in AI education. It could reshape how the world learns. MIT Technology Review2ndAugust / https://fully-human.org/wp-content/uploads/2019/09/China-AI-experiment-education.pdf

21) Indian Express (2021). Online classes to continue even as schools reopen. Indian Express / https://indianexpress.com/article/education/online-classes-to-continue-even-as-schools-reopen-education-minister-7151205/

22) International Commission on the Futures of Education (2020)Education in a post-COVID world: Nine ideas for publicaction.

23) Krutka, D., Smits, R. & Willhelm, T. (2021) Don't Be Evil: Should We Use Google in Schools? TechTrends(in press)

24) Kuhfeld, M., Soland, J., Tarasawa, B., Johnson, A., Ruzek, E. & Liu, J. (2020). Projecting the potential impact of COVID-19 school closures on academic achievement.Educational Researcher,49(8), 549-565.

25) Pea, R. (1987). The aims of software criticism.Educational Researcher16(5):4-8.

26) Pollock, K., & Hauseman, D. C. (2018). The use of e-mail and principals' work. Leadership and Policy in Schools, 18(3), 382-393.

27) Robinson, L., Schulz, J., Khilnani, A., Ono, H., Cotten, S., Mcclain, N. & Tolentino, N. (2020). Digital inequalities in time of pandemic: COVID-19 exposure risk profiles and new forms of vulnerability.First Monday25(7) / https://firstmonday.org/ojs/index.php/fm/article/view/10845

28) Rohs, M. & Ganz, M. (2015). MOOCs and the claim of education for all: a disillusion by empirical data.International Review of Research In Open and Distributed Learning,16(6), 1-19.

29) Scholz, T. (2019). A portfolio of platform cooperativism, in progress.Ökologisches Wirtschaften-Fachzeitschrift,33(4), 16-19.

30) Selwyn, N. (2019). Should robots replace teachers?Polity

31) Selwyn, N. (2020) How creative use of technology may have helped save schooling during thepandemic. The Conversation (October 26th)

32) Selwyn, N., Nemorin, S., & Johnson, N. (2017). High-tech, hard work: An investigation of teachers' work in the digital age. Learning, Media and Technology, 42(4), 390-405.

33) Sims, C. (2017).Disruptive fixation: school reform and the pitfalls of techno-idealism. Princeton University Press.

34) Tawfik, A., Reeves, T. and Stich, A. (2016). Intended and unintended consequences of educational technology on social inequality.TechTrends,60(6): 598-605.

35) Tewathia, N., Kamath, A., & Ilavarasan, P. (2020). Social inequalities, fundamental inequities, and recurring of the digital divide:Insights from India.Technology in Society,61, 101251.

36) UNESCO (2020) Global education monitoring report summary, 2020: Inclusion and education: all means all. UNESCO -Global Education Monitoring

37) UNICEF Innocenti (2020) Promisingpractices for equitable remote learning: Emerging lessons from COVID-19 education responses in 127 countries. Innocenti Research Brief 2020-10 (May)

38) Wagner, D. (2018). Technology for education in low-income countries: Supporting the UN sustainable development goals.ICT-Supported Innovations in Small Countries and Developing Regions, 51-74.

39) Welner, K. (2020) NEPC Review:Public-private virtual-school partnerships and federal flexibility for schools during COVID-19. Colorado, National Education PolicyCenter / https://nepc.colorado.edu/thinktank/coronavirus

40) Williamson, B. & Hogan, A. (2020) Commercialisation and privatisation in/of education in the context of COVID-19. Education International

41) Інформаційна дошка проекту E-schools / https://e-schools.info/e-service

42) Проектування бази даних – 2-е видання / https://opentextbc.ca/dbdesign01

ДОДАТОК А

КОД ПРОГРАМИ

Створення бази данних

```
DROP DATABASE IF EXISTS school _db;
CREATE DATABASE school _db;
USE school _db;

DROP TABLE IF EXISTS users;
CREATE TABLE users (
        id SERIAL,
        created_at DATETIME DEFAULT NOW(),
  updated_at DATETIME DEFAULT NOW() ON UPDATE NOW(),
        firstname VARCHAR(50) NOT NULL ,
        lastname VARCHAR(50) NOT NULL,
        patronymic VARCHAR(50) DEFAULT,
        login VARCHAR(50) NOT NULL,
        password_hash VARCHAR(100) NOT NULL COMMENT,
        email VARCHAR(50) UNIQUE COMMENT,
        phone BIGINT UNSIGNED UNIQUE COMMENT,

        PRIMARY KEY (id, firstname, lastname, patronymic),
        CHECK(YEAR(created_at) >= YEAR(updated_at))

);

DROP TABLE IF EXISTS cards;
CREATE TABLE cards(
        id SERIAL,
        card_number INT(10) ZEROFILL UNSIGNED NOT NULL  COMMENT 'Номер, напечатанный на карте',
        balance INT UNSIGNED NOT NULL DEFAULT 0  COMMENT 'Баланс денег на карте',
        photo_id BIGINT UNSIGNED COMMENT 'Идентификатор файла из хранилища фотографий'

);

DROP TABLE IF EXISTS subjects;
CREATE TABLE subjects (
        id SERIAL,
        name VARCHAR(50) NOT NULL COMMENT,
        is_actual BIT DEFAULT 1 NOT NULL COMMENT,
        PRIMARY KEY (id)

);


DROP TABLE IF EXISTS profiles;
CREATE TABLE profiles (
        user_id BIGINT UNSIGNED NOT NULL UNIQUE COMMENT 'ID из таблицы users',
        gender ENUM('f', 'm') NOT NULL COMMENT 'Пол пользователя', -- пол
        birthday DATE NOT NULL COMMENT 'Дата рождения пользователя',
        photo_id BIGINT UNSIGNED COMMENT 'ID файла из хранилища фотографий',
  created_at DATETIME DEFAULT NOW() COMMENT 'Когда создана запись',
  updated_at DATETIME DEFAULT NOW() ON UPDATE NOW() COMMENT 'Когда запись в последний раз обновлена',
  -- id электронной карты:
  card_id BIGINT UNSIGNED UNIQUE COMMENT 'ID карты', -- нет параметра NOT NULL так как при поступлении могут не сразу выдать
карту
  `role` ENUM ('Ученъ', 'Учитель', 'Завуч', 'Директор') NOT NULL DEFAULT 'Ученъ',
  actual BIT NOT NULL DEFAULT 1,

  FOREIGN KEY (user_id) REFERENCES users(id),
  FOREIGN KEY (card_id) REFERENCES cards(id)
);

DROP TABLE IF EXISTS buildings;
CREATE TABLE buildings(
        id SERIAL,
        address VARCHAR(255),
        floors TINYINT(1),
        -- branch_num TINYINT UNSIGNED NOT NULL UNIQUE,
        is_main BIT DEFAULT 0 NOT NULL COMMENT,
        official_name VARCHAR(255) DEFAULT СЗШ №999'
        manager_id BIGINT UNSIGNED NOT NULL UNIQUE,
```

```
        FOREIGN KEY (manager_id) REFERENCES users(id)
);

DROP TABLE IF EXISTS cabinet_assignment;
CREATE TABLE cabinet_assignment(
        id SERIAL,
        `assignment` VARCHAR(100) NOT NULL,
        PRIMARY KEY (id)
);

DROP TABLE IF EXISTS cabinets;
CREATE TABLE cabinets(
        id SERIAL,
        building_id BIGINT UNSIGNED NOT NULL ,
        cabinet_number INT UNSIGNED NOT NULL,
        `floor` TINYINT UNSIGNED NOT NULL
        assignment_id BIGINT UNSIGNED NOT NULL
        PRIMARY KEY (id, building_id, cabinet_number),
        FOREIGN KEY (building_id) REFERENCES buildings(id),
        FOREIGN KEY (assignment_id) REFERENCES cabinet_assignment(id)

);

DROP TABLE IF EXISTS classes;
CREATE TABLE classes(
        id SERIAL,
        num ENUM ('1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11') NOT NULL DEFAULT '1'
        letter CHAR(1) NOT NULL
        building_id BIGINT UNSIGNED NOT NULL
        classmanager_id BIGINT UNSIGNED NOT NULL
        cabinet_id BIGINT UNSIGNED NOT NULL
        is_actual BIT DEFAULT 1 NOT NULL

        PRIMARY KEY (id, num, letter),
        FOREIGN KEY (building_id) REFERENCES buildings(id),
        FOREIGN KEY (cabinet_id) REFERENCES cabinets(id)
);

INSERT classes
        (num, letter, building_id, classmanager_id, cabinet_id)
DROP TABLE IF EXISTS users_classes;
CREATE TABLE users_classes (
        user_id BIGINT UNSIGNED NOT NULL,
        class_id BIGINT UNSIGNED NOT NULL,

        PRIMARY KEY (user_id, class_id),
        FOREIGN KEY (user_id) REFERENCES users(id),
        FOREIGN KEY (class_id) REFERENCES classes(id)
);

DROP TABLE IF EXISTS users_subjects;
CREATE TABLE users_subjects(
        user_id BIGINT UNSIGNED NOT NULL,
        subject_id BIGINT UNSIGNED NOT NULL,

        PRIMARY KEY (user_id, subject_id),
        FOREIGN KEY (user_id) REFERENCES users(id),
        FOREIGN KEY (subject_id) REFERENCES subjects(id)
);

DROP TABLE IF EXISTS education_pedriods;
CREATE TABLE education_pedriods (
        id SERIAL,
        period_name VARCHAR(25),
        period_start DATE NOT NULL,
        period_end DATE NOT NULL,

        PRIMARY KEY (id),
        CHECK (period_start <> period_end)
);

DROP TABLE IF EXISTS intervalues;
CREATE TABLE intervalues (
        id TINYINT UNSIGNED NOT NULL UNIQUE,
        start_time TIME NOT NULL
        intervalue TIME NOT NULL DEFAULT '00:45:00'
        end_time TIME COMMENT

        PRIMARY KEY (id)
```

```
);

DROP TABLE IF EXISTS timetable;
CREATE TABLE timetable(
	id SERIAL,
	class_id BIGINT UNSIGNED NOT NULL
	subject_id BIGINT UNSIGNED NOT NULL
	`date` DATE NOT NULL COMMENT
	intervalues_id TINYINT UNSIGNED NOT NULL
	teacher_id BIGINT UNSIGNED NOT NULL

	PRIMARY KEY (class_id, subject_id, `date`, teacher_id, intervalues_id),
	FOREIGN KEY (intervalues_id) REFERENCES intervalues(id),
	FOREIGN KEY (class_id) REFERENCES classes(id),
	FOREIGN KEY (subject_id) REFERENCES subjects(id)


CREATE INDEX data_when ON timetable (`date`);


DROP TABLE IF EXISTS grades;
CREATE TABLE grades (
	id SERIAL,
	student_id BIGINT UNSIGNED NOT NULL
	timetable_id BIGINT UNSIGNED NOT NULL
	`comment` VARCHAR(255) DEFAULT NULL COMMENT 'Комментарий',
	grade ENUM('1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12') NOT NULL,

	FOREIGN KEY (student_id) REFERENCES users(id),
	FOREIGN KEY (timetable_id) REFERENCES timetable(id)
);


DROP TABLE IF EXISTS homeworks;
CREATE TABLE homeworks (
	id SERIAL,
	homework VARCHAR(255) DEFAULT '()
	teacher_id BIGINT UNSIGNED NOT NULL
	timetable_id BIGINT UNSIGNED NOT NULL

	PRIMARY KEY (id),
	FOREIGN KEY (teacher_id) REFERENCES users(id),
	FOREIGN KEY (timetable_id) REFERENCES timetable(id)

)
```

Запити

Оцінки

```
SELECT
	CONCAT(u.firstname, ' ',u.lastname)
	t.`date`,
	s.name,
	g.grade
	g.grade_type
	g.comment
FROM grades g
JOIN users u
	ON g.student_id = u.id
JOIN timetable t
	ON t.id = g.timetable_id
JOIN subjects s
	ON s.id = t.subject_id
WHERE
	s.id = (SELECT id FROM subjects WHERE name = 'Алгебра')
	AND u.id = 17;
```

Усі оцінки учня

```
SELECT
	CONCAT(u.firstname, ' ',u.lastname)
	t.`date`
	s.name,
	g.grade
	g.grade_type
	g.comment
FROM grades g
JOIN users u
	ON g.student_id = u.id
```

```
JOIN timetable t
	ON t.id = g.timetable_id
JOIN subjects s
	ON s.id = t.subject_id
JOIN profiles p
	ON p.user_id = 17 -- ID ученика
WHERE u.id = 17 AND -- ID ученика
	p.actual = 1 AND s.is_actual = 1;
```

Представления

```
-- представление всех учеников школы по классам:
CREATE OR REPLACE VIEW view_all_students AS
SELECT
	CONCAT(u.firstname, ' ', u.lastname) AS student,
	CONCAT(c.num, ' ', c.letter) AS class
FROM users u
JOIN classes c
JOIN users_classes uc
	ON uc.user_id = u.id AND c.id = uc.class_id;

-- SELECT * FROM view_all_students;


-- представление для просмотра какой учитель что ведёт:
CREATE OR REPLACE VIEW teachers_subjects
SELECT
	CONCAT(u.firstname, ' ', u.lastname)
	s.name
FROM users u
JOIN users_subjects us
	ON us.user_id = u.id
JOIN subjects s
	ON s.id = us.subject_id;

-- SELECT * FROM teachers_subjects;


-- представление для просмотра классных руководителей:
CREATE OR REPLACE VIEW classmanagers
SELECT
	CONCAT(c.num, ' ',UPPER(c.letter))
	CONCAT(u.firstname, ' ', u.lastname)

FROM classes c
JOIN users u
	ON c.classmanager_id = u.id;

-- SELECT * FROM classmanagers;


-- представление всех предметов в школе:
CREATE OR REPLACE VIEW subjects_view AS
SELECT
	id,
	name AS 'Предмет',
	-- is_actual AS 'Актуальность',
	CASE
		WHEN is_actual = 1

		WHEN is_actual = 0

	END
	category
FROM subjects;



CREATE OR REPLACE VIEW first_bulding_cabinets AS
SELECT
	cabinet_number
	`assignment`
	`floor`
FROM
	cabinets c
JOIN
	cabinet_assignment ca
ON
```

```
	c.assignment_id = ca.id
WHERE
	building_id = 1;


-- представление кабинетов для второго здания:
CREATE OR REPLACE VIEW second_bulding_cabinets AS
SELECT
	cabinet_number,
	`assignment`
	`floor`
FROM
	cabinets c
JOIN
	cabinet_assignment ca
ON
	c.assignment_id = ca.id
WHERE
	building_id = 2;


-- представление расписания звонков:
CREATE OR REPLACE VIEW rings_time AS
SELECT
	i.id,
	i.start_time,
	(ADDTIME(i.start_time, i.intervalue)) AS end_time
FROM
	intervalues i;
```

Розклад занять

```
CREATE OR REPLACE VIEW student_timetable AS
SELECT
	t.id, t.class_id,
	s.name,
	t.`date`,
	DAYNAME(t.`date`),
	t.intervalues_id
FROM subjects s
JOIN timetable t
	ON t.subject_id = s.id
ORDER BY DAYNAME(t.`date`);

CREATE OR REPLACE VIEW teacher_timetable AS
SELECT
	t.`date`,
	DAYNAME(t.`date`) AS day_name,
	-- t.class_id,
	CONCAT(c.num,c.letter) AS class,
	t.intervalues_id,
	t.teacher_id
FROM timetable t
JOIN classes c
	ON c.id = t.class_id
ORDER BY t.`date`, t.intervalues_id;

-- представление на все оценки по всем предметам конкретного ученика:
CREATE OR REPLACE VIEW view_one_stud_grades AS
SELECT
	CONCAT(u.firstname, ' ',u.lastname)
	t.`date`
s.name,
	g.grade
g.grade_type,
	g.comment
FROM grades g
JOIN users u
	ON g.student_id = u.id
JOIN timetable t
	ON t.id = g.timetable_id
JOIN subjects s
	ON s.id = t.subject_id
JOIN profiles p
	ON p.user_id = 17 -- ID ученика
WHERE u.id = 17 AND -- ID ученика
```

```
	p.actual = 1 AND s.is_actual = 1;

CREATE OR REPLACE VIEW average_grade AS
SELECT
	SUM(g.grade)/COUNT(g.grade)
FROM grades g
JOIN users u
	ON g.student_id = u.id
JOIN timetable t
	ON t.id = g.timetable_id
JOIN subjects s
	ON s.id = t.subject_id
JOIN profiles p
WHERE s.id = 7
	AND u.id = 17
	AND p.actual = 1 AND s.is_actual = 1;

CREATE OR REPLACE VIEW view_class_by_subject AS
SELECT
	CONCAT(u.firstname, ' ', u.lastname) AS student,
	g.grade,
	t.`date`,
	s.name
FROM users_classes uc
JOIN users u
	ON uc.user_id = u.id
JOIN timetable t
	ON t.class_id = uc.class_id
JOIN grades g
	ON g.student_id = u.id AND g.timetable_id = t.id
JOIN subjects s
	ON s.id = t.subject_id
WHERE t.class_id = 49 AND t.subject_id = 7; -- ID класса и предмета
```

Тригери

```
DROP TRIGGER IF EXISTS tg_check_actual_student_class//
CREATE TRIGGER `tg_check_actual_student_class` BEFORE INSERT ON `users_classes` FOR EACH ROW

	IF
		NEW.user_id IN (SELECT user_id FROM profiles WHERE actual = 0)
	THEN
		SIGNAL SQLSTATE '45059'

	ELSE
		IF
			NEW.class_id IN (SELECT id FROM classes WHERE is_actual = 0)
		THEN
			SIGNAL SQLSTATE '45060'
								END IF;
	END IF;
//

DROP TRIGGER IF EXISTS tg_check_actual_teacher_subj//
CREATE TRIGGER `tg_check_actual_teacher_subj` BEFORE INSERT ON `users_subjects` FOR EACH ROW

	IF
		NEW.user_id IN (SELECT user_id FROM profiles WHERE actual = 0)
	THEN
		SIGNAL SQLSTATE '45059'
				ELSE
		IF
			NEW.subject_id IN (SELECT id FROM subjects WHERE is_actual = 0)
		THEN
			SIGNAL SQLSTATE '45061'
								END IF;
	END IF;
//


-- триггер на проверку, чтобы классным руководителем мог быть только учитель, завучи или директор:
-- на вставку данных:
DROP TRIGGER IF EXISTS school_global_db.tg_classmngr_role_ins//
CREATE TRIGGER `tg_classmngr_role_ins` BEFORE INSERT ON `classes`FOR EACH ROW
	IF
		(SELECT p.`role` FROM profiles p WHERE p.user_id = NEW.classmanager_id) NOT IN ('Учитель', 'Завуч', 'Директор')
	THEN
		SIGNAL SQLSTATE '45047'
				SET MESSAGE_TEXT = 'Только директор или учитель может быть классным руководителем!';
```

```
	ELSE
		IF
			NEW.classmanager_id IN (SELECT user_id FROM profiles WHERE actual = 0)
		THEN
			SIGNAL SQLSTATE '45059'
								END IF;
	END IF;
//

-- на обновление данных:
DROP TRIGGER IF EXISTS school_global_db.tg_classmngr_role_upd//
CREATE TRIGGER `tg_classmngr_role_upd` BEFORE UPDATE ON `classes`FOR EACH ROW
	IF
		(SELECT p.`role` FROM profiles p WHERE p.user_id = NEW.classmanager_id) NOT IN ('Учитель', 'Завуч', 'Директор')
	THEN
		SIGNAL SQLSTATE '45047'
					ELSE
		IF
			NEW.classmanager_id IN (SELECT user_id FROM profiles WHERE actual = 0)
		THEN
			SIGNAL SQLSTATE '45059'
								END IF;
	END IF;
//


DROP TRIGGER IF EXISTS tg_check_student_for_grade//
CREATE TRIGGER tg_check_student_for_grade
BEFORE INSERT
ON grades FOR EACH ROW

	IF
		grades.student_id NOT IN (SELECT u.id FROM users u WHERE `role` = 'Ученъ')
	THEN
		SIGNAL SQLSTATE '45050'
					ELSE
		IF
			grades.student_id IN (SELECT p.user_id FROM profiles p WHERE p.actual = 0)
		THEN
			SIGNAL SQLSTATE '45049'
								END IF;
	END IF;
//


DROP TRIGGER IF EXISTS tg_one_main_buildng//
CREATE TRIGGER tg_one_main_buildng
BEFORE INSERT ON buildings FOR EACH ROW

BEGIN

	IF
		NEW.is_main = 1 AND
		(SELECT b.is_main FROM buildings b WHERE b.is_main = 1) = 1

	THEN
		SIGNAL SQLSTATE '45048'
				END IF;

END//



DROP TRIGGER IF EXISTS tg_check_floor_ins//
DELIMITER //
CREATE TRIGGER tg_check_floor_ins
BEFORE INSERT
ON cabinets FOR EACH ROW
	IF
		NEW.`floor` > (SELECT b.floors FROM buildings b WHERE NEW.building_id = b.id)
	THEN
		SIGNAL SQLSTATE '45058'
		END IF;
//

DROP TRIGGER IF EXISTS tg_check_floor_upd//
CREATE TRIGGER tg_check_floor_upd
BEFORE UPDATE
ON cabinets FOR EACH ROW
```

```
	IF
		NEW.`floor` > (SELECT b.floors FROM buildings b WHERE NEW.building_id = b.id)
	THEN
		SIGNAL SQLSTATE '45058'
	END IF;
//


DROP TRIGGER IF EXISTS tg_student_not_teacher//
CREATE TRIGGER tg_student_not_teacher
BEFORE INSERT ON users_subjects FOR EACH ROW

BEGIN
	-- если выбран id ученика:
	IF
		NEW.user_id IN (SELECT user_id FROM profiles WHERE `role` = 'Учащийся')
	THEN
		SIGNAL SQLSTATE '45045'
			SET MESSAGE_TEXT = 'Невозможно назначить ученику вести предмет!';
	END IF;

END//


-- триггер на проверку добавления ученика (а не учителя) в класс
DROP TRIGGER IF EXISTS tg_teacher_not_student//
CREATE TRIGGER tg_teacher_not_student
BEFORE INSERT ON users_classes FOR EACH ROW
	-- если выбран id не ученика:
	IF
		NEW.user_id NOT IN (SELECT user_id FROM profiles WHERE `role` = 'Учень')
	THEN
		SIGNAL SQLSTATE '45045'
					END IF;
//

CREATE TRIGGER `tg_check_actual_student_for_grades_ins` BEFORE INSERT ON `grades` FOR EACH ROW
	IF
		NEW.student_id IN (SELECT user_id FROM profiles WHERE actual = 0)
	THEN
		SIGNAL SQLSTATE '45059'
	END IF;
//

DROP TRIGGER IF EXISTS tg_check_actual_student_for_grades_upd//
CREATE TRIGGER `tg_check_actual_student_for_grades_upd` BEFORE UPDATE ON `grades` FOR EACH ROW
	IF
		NEW.student_id IN (SELECT user_id FROM profiles WHERE actual = 0)
	THEN
		SIGNAL SQLSTATE '45059'
	END IF;
//
DELIMITER ;
```

Процедури

```
DROP PROCEDURE IF EXISTS `sp_teacher_timetable`;
DELIMITER //

CREATE PROCEDURE `sp_teacher_timetable`(
	var_teacher_id BIGINT, var_mode CHAR(1))
BEGIN

	DECLARE var_date_start DATE;
	DECLARE var_date_end DATE;

	IF
		var_mode NOT IN ('c', 'a', '1', '2', '3', '4')
	THEN
		SET var_mode = 'c';
	END IF;

	IF
		var_mode = 'a' -- all,
	THEN
		IF
			MONTH(NOW()) < 9
		THEN
```

```
                        SET var_date_start = CONCAT((YEAR(NOW()) - 1),'-09-01');
                ELSE
                        SET var_date_start = CONCAT(YEAR(NOW()),'-09-01');
                END IF;

        SET var_date_end = ADDDATE(var_date_start, INTERVAL 10 MONTH);
                        ELSE
                IF
                        -- если var_mode = 'c'
                        var_mode = 'c'
                THEN
                        IF

                                (SELECT (NOW() BETWEEN ep.period_start AND ep.period_end) FROM education_pedriods ep) = 1
                        THEN

                                SET var_date_start = (SELECT ep.period_start FROM education_pedriods ep WHERE (NOW()
BETWEEN ep.period_start AND ep.period_end) = 1);
                                SET var_date_end = (SELECT ep.period_end FROM education_pedriods ep WHERE (NOW()
BETWEEN ep.period_start AND ep.period_end) = 1);
                        END IF;
                END IF;
        END IF;

        IF
                        (SELECT p.`role` FROM profiles p WHERE p.user_id = var_teacher_id) IN ('Учитель', 'Директор', 'Завуч')
        THEN

                SELECT
                        tt.`date`
                        tt.day_name,
                        tt.class
                        tt.intervalues_id
                FROM teacher_timetable tt
                WHERE
                        tt.teacher_id = var_teacher_id AND
                        tt.`date` >= var_date_start AND
                        tt.`date` BETWEEN var_date_start AND var_date_end
                ORDER BY tt.`date`;


                SELECT
                        t.`date`
                        DAYNAME(t.`date`)
                        CONCAT(c.num,UPPER(c.letter))
                        t.intervalues_id
                FROM timetable t
                JOIN classes c
                        ON c.id = t.class_id
                WHERE
                        t.teacher_id = var_teacher_id AND
                        t.`date` >= var_date_start AND
                        t.`date` BETWEEN var_date_start AND var_date_end
                ORDER BY t.`date`;
                */
        END IF;

END//


:
DROP PROCEDURE IF EXISTS sp_show_classtimetable//
CREATE PROCEDURE sp_show_classtimetable(
        var_class_id BIGINT UNSIGNED)
BEGIN
        SELECT
                st.`date`, DAYNAME(st.`date`), st.name, st.intervalues_id
        FROM student_timetable st
        WHERE st.class_id = var_class_id
        ORDER BY st.`date`, st.intervalues_id;
END//
/*

        CALL sp_show_classtimetable(49);
*/

DROP PROCEDURE IF EXISTS school_global_db.sp_average_grade//
CREATE PROCEDURE school_global_db.sp_average_grade(
        var_user_id BIGINT, var_subj_id BIGINT)
```

```
BEGIN

	IF
		(SELECT p.`role` FROM profiles p WHERE p.user_id = var_user_id) IN ('Учитель', 'Директор', 'Завуч')
	THEN
		SIGNAL SQLSTATE '45050'

	ELSE
		IF
			var_subj_id IN (SELECT s.id FROM subjects s WHERE s.is_actual = 0)
		THEN
			SIGNAL SQLSTATE '45051'
					ELSE
			IF
				var_user_id IN (SELECT p.user_id FROM profiles p WHERE p.actual = 0)
			THEN
				SIGNAL SQLSTATE '45052'
						ELSE
				IF
					var_subj_id IN (SELECT s.id FROM subjects s)
				THEN
					(SELECT
					CONCAT(u.firstname, ' ',u.lastname)
					t.`date` ,
					s.name,
					g.grade, g.grade_type, g.comment
					FROM grades g
					JOIN users u
						ON g.student_id = u.id
					JOIN timetable t
						ON t.id = g.timetable_id
					JOIN subjects s
						ON s.id = t.subject_id
					WHERE s.id = var_subj_id
						AND u.id = var_user_id)
					UNION
					(SELECT
						'Средняя оценка:',
						DATE(NOW()),
						s2.name,
						ag.`SUM(g.grade)/COUNT(g.grade)`,
						'балла(ов)',
						NULL
					FROM average_grade ag
					JOIN subjects s2
						ON s2.id = var_subj_id);
				ELSE
					SIGNAL SQLSTATE '45053'

				END IF;
			END IF;
		END IF;
	END IF;
END//
/*

	CALL sp_average_grade(17,7);
*/


DROP PROCEDURE IF EXISTS school_global_db.sp_student_diary//
CREATE PROCEDURE school_global_db.sp_student_diary(
	var_user_id BIGINT, var_edperiod_id TINYINT)
BEGIN

	DECLARE
		var_class_id BIGINT
	DEFAULT
		(SELECT class_id
		FROM users_classes
		WHERE user_id = var_user_id);

	DECLARE
		start_d DATE
	DEFAULT
		(SELECT ep.period_start
		FROM education_pedriods AS ep
		WHERE ep.id = var_edperiod_id);
```

```
	DECLARE
		end_d DATE
	DEFAULT
		(SELECT ep.period_end
		FROM education_pedriods AS ep
		WHERE ep.id = var_edperiod_id);

	IF

		var_user_id IN (SELECT p.user_id FROM profiles p WHERE p.`role` = 'Учень')
	THEN
		IF

			var_user_id IN (SELECT p.user_id FROM profiles p WHERE p.actual = 1)
		THEN
			-- выборка:
			(SELECT
				st.`date`, DAYNAME(st.`date`) AS `dayname`, st.name AS subject,

				CASE
					WHEN
						(h.timetable_id = st.id) = 1
					THEN
						h.homework
				END AS hw,


				g.grade AS gradetype_lessonwork,
				g2.grade AS gradetype_homework,
				g3.grade AS gradetype_other

			FROM student_timetable st

			LEFT JOIN homeworks h
			ON
				h.timetable_id = st.id AND
				st.class_id = var_class_id AND
				h.timetable_id IN (
					SELECT st2.id
					FROM student_timetable st2
					WHERE st2.class_id = var_class_id)

			LEFT JOIN grades g
			ON
				g.timetable_id = st.id AND
				g.student_id = var_user_id AND
				g.grade_type = 'Работа на уроке'

			LEFT JOIN grades g2
			ON
				g2.timetable_id = st.id AND
				g2.student_id = var_user_id AND
				g2.grade_type = 'Домашнее задание'

			LEFT JOIN grades g3
			ON
				g3.timetable_id = st.id AND
				g3.student_id = var_user_id AND
				g3.grade_type = 'Иное'

			WHERE
				st.class_id = var_class_id AND
				st.`date` BETWEEN start_d AND end_d
			GROUP BY st.id, h.timetable_id
			ORDER BY st.`date`, st.intervalues_id);

		ELSE
			SELECT 'Выбран неактуальный пользователь!';
		END IF;
	ELSE
		SELECT 'Необходимо ввести id учника!';
	END IF;

END//
/*
	CALL sp_student_diary(17, 1);
*/
```

```
DROP PROCEDURE IF EXISTS school_global_db.sp_class_grades_by_subj//
CREATE PROCEDURE school_global_db.sp_class_grades_by_subj(
	var_class_id BIGINT, var_subj_id BIGINT, var_edperiod_id BIGINT)
BEGIN


	DECLARE
		start_d DATE
	DEFAULT
		(SELECT ep.period_start
		FROM education_pedriods AS ep
		WHERE ep.id = var_edperiod_id);

	DECLARE
		end_d DATE
	DEFAULT
		(SELECT ep.period_end
		FROM education_pedriods AS ep
		WHERE ep.id = var_edperiod_id);

	-- тело выборки:
	SELECT
		CONCAT(c.num, UPPER(c.letter)) AS class,
		CONCAT(u.firstname, ' ', u.lastname) AS student
		g.grade AS lessonwork
g2.grade AS homework,
		g3.grade AS other,
		t.`date` AS lesson_date
		t.intervalues_id AS lesson_num,
		s.name AS subject
	FROM users_classes uc
	JOIN users u

		ON uc.user_id = u.id
	JOIN timetable t
					ON t.class_id = uc.class_id
	JOIN subjects s
					ON s.id = t.subject_id
	JOIN student_timetable st
		ON st.id = t.id
	JOIN classes c
		ON c.id = uc.class_id
	WHERE
					t.class_id = var_class_id AND
		t.subject_id = var_subj_id AND
		t.`date` BETWEEN start_d AND end_d
	GROUP BY
		student, t.id, t.`date`, subject
	ORDER BY
		t.`date`, t.intervalues_id, student;

END//

/*
CALL sp_class_grades_by_subj(49, 1, 1);
*/



DROP PROCEDURE IF EXISTS sp_homework_ins//
CREATE PROCEDURE sp_homework_ins(
	var_teacher_id BIGINT, var_tt_id BIGINT, var_homehork VARCHAR(255),
	OUT tr_result VARCHAR(200))  -- переменная на вывод результата транзакции
BEGIN

	DECLARE stop_tr BIT DEFAULT 0; -- обозначение факта ошибки транзакции
	DECLARE err_code VARCHAR(100); -- код ошибки
	DECLARE err_string VARCHAR(150); -- текст ошибки

		DECLARE CONTINUE HANDLER FOR SQLEXCEPTION
	BEGIN
	  SET stop_tr = 1;
		GET stacked DIAGNOSTICS CONDITION 1
	    err_code = RETURNED_SQLSTATE, err_string = MESSAGE_TEXT;
		SET tr_result := CONCAT('Error occured. Code: ', err_code, '. Text: ', err_string);
  END;

	START TRANSACTION;
```

```
        INSERT homeworks
                (homework, teacher_id, timetable_id)
        VALUES
                (var_homework, var_teacher_id, var_tt_id);

        -- результат транзакции:
  IF
                stop_tr = 1
        THEN
                ROLLBACK;
        ELSE
                COMMIT;
                SET tr_result = 'Transaction successfully completed.';
        END IF;
END//
```

Код сайту

HTML код файлу «login»

```
<!Doctype Html>
<Html Lang="En">
<Head>
<Meta Charset="Utf-8">
<Meta Name="Viewport" Content="Width=Device-Width, Initial-Scale=1, Shrink-To-Fit=No">
<Title>Шкільний Журнал</Title>
<Link Href="Https://Use.Fontawesome.Com/Releases/V5.0.6/Css/All.Css" Rel="Stylesheet">
<Link Rel="Stylesheet" Href="Css/Bootstrap4-Charming.Min.Css">
<Style>
Body {
Margin: 0;
Padding: 0;
Display: Flex;
Flex-Direction: Column;
Justify-Content: Space-Between;
Min-Height: 100vh;
}
.Navbar-Text {
Margin-Right: 10px;
Margin-Left: 20px;
}
Img {
Width: 48px;
Filter: Contrast(200%);
Filter: Brightness(1.35);
}
.Green {
Color: #35bd49;
}
</Style>
</Head>
<Body>
<!--  Navigation -->
<!-- Image And Text -->
<Nav Class="Navbar Navbar-Light Bg-Light">
<A Class="Navbar-Brand" Href="#">
<I Class="Fas Fa-Book Fa-Lg" Style="Font-Size:1.5em; Color:#35bd49"></I>
Шкільний Журнал
</A>
<Div Class="Navbar Navbar-Right">
<A Class="Nav-Item Nav-Link Active" Href="#">Особистий Журнал<Span Class="Sr-Only">(Current)</Span></A>
<A Class="Nav-Item Nav-Link" Href="#">Розклад Занять</A>
<Div Class="Navbar-Text">
<A Href="#!" Class="Ht-Tm-Element Btn">
<A Href="#!" Class="Ht-Tm-Element Btn Btn-Success">
```

```
<Span Class="Fas Fa-Sign-Out-Alt"></Span> Увійти</A>
</Div>
</Div> <!--End Of The Right Part Of Navbar -->
</Nav>    <!-- End Of Navigation -->
<!-- Announcement-->
<Div Class="Container-Fluid" Style="Margin-Top:100px; Text-Align: Center;">
<Div Class="Row">
<Div Class="Col">
<P>
<Span Style = "Font-Size: 25px">Логін -</Span>
<Span Class="Green" Style="Font-Size: 25px"></Span>
</Ul>
</P>
<P>
<Span Style = "Font-Size: 25px">Пароль -</Span>
</P>
</Div>
</Div>
</Div>
<!-- Button For Start-->
<Div Class="Container" Style="Margin-Top: 50px; Margin-Bottom: 50px; Text-Align: Center; Border-Radius: 10px; Box-Shadow: 5px Grayscale">
<Div Class="Row">
<Div Class="Col">
<Button Type="Button" Class="Btn-Success Btn-Lg Animated " Role="Button" Aria-Pressed="True Visibility: Visible; Animation-Delay: 0.4s; Animation-
Name: Fadeindown;">Увійти</Button>
</Div>
</Div>
</Div>
<!-- Services -->
<Div Class="Container-Fluid" Style="Min-Height:540px; Background-Color:#373a3c; Color:#F2f2f2">
<!-- Titles -->
<Div Class="Row" Style="Padding-Top: 50px;">
<Div Class="Col-12" Style="Text-Align: Center">
<Span Class="Green" Style="Text-Transform: Uppercase;">Services</Span>
<H2>
Our Best Features</H2>
</Div>
</Div>
<Div Class="Row">
<Div Class="Col-12" Style="Text-Align: Center">
<Img Src="Images/Substract.Png" Alt="Line">
</Div>
</Div>
<!-- Icons -->
<Div Class="Container" Style="Text-Align: Center" Id="Icons">
<Div Class="Row">
<!-- Backpack Icon -->
<Div Class="Col-Xl-4 Col-Sm-12">
</Div>
</Div>
</Div>
</Div>
</Body>
```

Html Код Файлу «Home»

```
<!Doctype Html>
<Html>
<Head>
<Title>Online Journal</Title>
```

```
<Meta Charset="Utf-8">
<Meta Name="Viewport" Content="Width=Device-Width, Initial-Scale=1, Shrink-To-Fit=No">
<!-- Links-->
<!-- Google Fonts-->
<Link Href="Https://Fonts.Googleapis.Com/Css?Family=Montserrat|Roboto" Rel="Stylesheet">
<!-- Bootstrap-4-->
<Link Rel="Stylesheet" Href="Https://Bootstrap/4.0.0-Alpha.2/Css/Bootstrap.Min.Css" Integrity="Sha384-
Y3tfxazxuh4hwsyylfb+J125mxis6mr5fohampbg064zb+Afewh94ndvacbm8qnd" Crossorigin="Anonymous">
<!-- Chaming-Themes-->
<Link Rel="Stylesheet" Href="Css/Bootstrap4-Charming.Min.Css">
<!-- Font Awesome-->
<Link Rel="Stylesheet" Href="Font-Awesome/4.7.0/Css/Font-Awesome.Min.Css"
<!-- Mail Styles-->
<Link Rel="Stylesheet" Href="Css/Style.Css">
</Head>
<Body>
<Div Class="Navbar Navbar-Expand-Lg Navbar-Fixed-Top Sticky">
<Div Class="Container">
<Div Class="Navbar-Header">
<Button Class="Navbar-Toggler" Type="Button" Data-Toggle="Collapse" Data-Target="#Navbartoggleexternalcontent" Aria-Controls="Navbartoggleexternalcontent"
Aria-Expanded="False" Aria-Label="Toggle Navigation">
<Span Class="Navbar-Toggler-Icon"></Span>
</Button>
<Nav Class="Navbar Navbar-Collapse">
<Span Class="Navbar-Brand Mb-0 H1">
<I Class="Fas Fa-Book Fa-Lg" Style="Font-Size:1.5em; Color:#35bd49"></I>
School Journal</Span>
</Nav>
</Div>
<Div Class="Navbar-Collapse Collapse" Id="Navbar-Menu">
<Ul Class="Nav Justify-Content-Center">
<Li Class="Nav-Item">
<A Class=" Nav-Item Nav-Link Active" Href="#">Home</A>
</Li>
<Li Class="Nav-Item">
<A Class="Nav-Link" Href="#">Features</A>
</Li>
<Li Class="Nav-Item Dropdown">
<A Class="Nav-Link Dropdown-Toggle" Href="#" Id="Navbardropdownmenulink" Data-Toggle="Dropdown" Aria-Haspopup="True" Aria-Expanded="False">
Plans
</A>
<Div Class="Dropdown-Menu" Aria-Labelledby="Navbardropdownmenulink">
<A Class="Dropdown-Item" Href="#">Plan For Today</A>
<A Class="Dropdown-Item" Href="#">Plan For Week</A>
<A Class="Dropdown-Item" Href="#">Plan For Term</A>
</Div>
</Li>
<!-- Emails-->
<Li Class="Nav-Item">
<A Class="Nav-Link" Href="#">Email</A>
</Li>
<!-- About-->
<Li Class="Nav-Item">
<A Class="Nav-Link" Href="#">About</A>
</Li>
</Ul>
<Ul Class="Nav Justify-Content-End">
<Li Class="Nav-Item">
<A Class="Nav-Link" Href="#">Login</A>
</Li>
```

```
<Li Class="Nav-Item">
<A Href="#!" Class="Ht-Tm-Element Btn Btn-Success Btn-Bordered Btn-Inverse">
<Span Class="Fas Fa-Sign-Out-Alt"></Span>Signup</A>
</Li>
</Ul>
<!-- End Of The Part Menu-->
</Div>
<!--End Menu -->
</Div>
</Div>
<Div Class="Container-Fluid" Id ="Announcement" Style="Margin-Top:100px; Text-Align: Center">
<Div Class="Row">
<Div Class="Col">
<P>
<Span Style = "Font-Size: 25px">School Journal -</Span>
<Span Style="Color:#00b33c; Font-Size: 25px">Be Always In Touch With Your School!</Span>
</Ul>
</P>
<P>
Our Journal Makes Communication Between School And Parents Easy As A Breeze. Be On Top Of Things!
</P>
</Div>
</Div>
</Div>
<Div Class="Container Btn_Start">
<Div Class="Row">
<Div Class="Col">
<Button Type="Button" Class="Btn-Success Btn-Lg Animated " Role="Button" Aria-Pressed="True Visibility: Visible; Animation-Delay: 0.4s; Animation-Name:
Fadeindown;">Get Started</Button>
</Div>
</Div>
</Div>
Const Pressspinner = Document.Queryselector('#Spinner');
//Console.Log(Pressspinner);
Let Openhideswitcher = Function(Event){
Let Prsspinner = Event.Target.Parentnode.Parentelement.Previouselementsibling;
//Console.Log(Prsspinner);
If (Prsspinner.Style.Left == "0px") {
Prsspinner.Style.Left = "-189px";
} Else {
Prsspinner.Style.Left = "0px";
}
}
Pressspinner.Addeventlistener('Click', Openhideswitcher);
</Script>
<Script Defer Src="Https://Use.Fontawesome.Com/Releases/V5.0.6/Js/All.Js"></Script>
<Script Src="Https://Maxcdn.Bootstrapcdn.Com/Bootstrap/4.0.0-Alpha.6/Js/Bootstrap.Min.Js" Integrity="Sha384" Crossorigin="Anonymous"></Script>
</Body>
```

Javascrypt Документ

```
Const Btnmenu = Document.Queryselector('#Dropdownmenu2');
Const Menulinks = Document.Queryselector('.Dropdown-Menu');
Const Btnsubmitregistration = Document.Queryselector('.Btn-Primary.Login');
Const Btndropdownmenu = Document.Queryselector('.Dropdown-Menu');
Const Partnavbar = Document.Queryselector('#Navbar');
Const Btnfirstregistration = Document.Queryselector('#Btnregistration');
Const Journalnavbar = Document.Queryselector('.Journal_Navbar');
Const Firstregistrationform = Document.Queryselector('#First-Registration');
Const Activitytable = Document.Queryselector('.Activity');
```

```
Const Scheduleday = Document.Queryselector('.Schedule');
Const Gradesactivity = Document.Queryselector('#Grades');
Const  Firstregistrationsendform = Document.Queryselector('#Registration');
Console.Log(Firstregistrationsendform);


Window.Onload = Function(Elements){
                Btndropdownmenu.Style.Display = 'None';
                Partnavbar.Style.Display = 'None';
}
const Thumbnails = styled.div`
  flex: 1;
  height: 60%;
  margin: auto;
  display: flex;
  flex-wrap: row;
  justify-content: center;
`;

const HomeContainer = styled`
  display: flex;
  flex-direction: column;
  align-items: center;
  box-sizing: border-box;
`;

const CreateTitle = styled.h3`
  font-size: 48px;
  color: white;
  font-weight: bold;
  font-family: Arial`;

const CreateInput = styled.input`
  width: 400px;
  height: 80px;
  font-size: 28px;
  padding: 20px;
  box-sizing: border-box;
  border-radius: 8px;
  border: none;
  outline-color: blue;
  box-shadow: 0 2px 4px grey;
  align-self: center;
`;

const DeleteButton = styled(Icon)`
  cursor: pointer;
  transition: opacity 0.3s ease-in-out;
  opacity: 0.4;
  &:hover {
    opacity: 0.8;
  }
`;

Let Openhidedropdownmenu = Function(Event, Elem){
                Console.Log(Menulinks);
                If (Menulinks.Style.Display == 'Block'){
                                Menulinks.Style.Display = 'None'
                } Else {Menulinks.Style.Display = 'Block'


Let Showhidenavbar = Function(Event, Elem = Btndropdownmenu){
                                Event.Preventdefault()
                                If(!Event) Event=Window.Event;
                                Let Registrationform = Event.Target;
                                Let Hideelementregistration = Registrationform.Parentnode.Parentnode.Parentnode.Parentnode;
                                Hideelementregistration.Style.Display = 'None';
                                Let Showelementnavbar =
Registrationform.Parentnode.Parentnode.Parentnode.Parentnode.Nextsibling.Nextsibling.Nextsibling.Nextsibling.Nextsibling.Nextsibling.Nextsibling.Nextsibling;
                                Console.Log(Showelementnavbar);
                                Showelementnavbar.Style.Display = 'Block';
                };
```

```
Let Showhideregistrationform = Function(Event, Elem){
                Console.Log('Hura!');
                Journalnavbar.Style.Display = 'None';
                Firstregistrationform.Style.Display = 'Block';
}

Let Sendform = Function(Event, Elem){
                Console.Log('Hura!');
                Firstregistrationform.Style.Display = 'None';
                Journalnavbar.Style.Display = 'Block';
                Let Addelementcontainer = Document.Createelement('Div');
                Addelementcontainer.Classlist.Add("Redbox");
                Addelementcontainer.Innerhtml = "Please, Check Your Email For Confirm";
                Journalnavbar.Appendchild(Addelementcontainer);

}
Btnfirstregistration.Addeventlistener("Click", Showhideregistrationform, True );
Btnsubmitregistration.Addeventlistener('Click', Showhidenavbar);
Btnmenu.Addeventlistener('Click', Openhidedropdownmenu);
Firstregistrationform.Addeventlistener('Click', Sendform);
```

Css Код Документу Style

```
* {
      -Moz-Box-Sizing: Border-Box;
 }
 Body {
      Background-Color:#F2f2f2;
      Opacity: 0.9;
      Background-Size: 100% 100%;
      /*Filter: Grayscale(0.5);*/
      Font-Family: 'Montserrat', Sans-Serif;

      Color: #4c5667;
      Background: #Fff;
      Font-Size: 15px;
      Line-Height: 22px;
      Overflow-X: Hidden;
}

H1 {
      Text-Align: Center;
   Padding-Top: 150px;
}

H2 {
      Text-Align: Center
}
A {
      Color: #35bd49;
}

A:Hover {
      Background-Color:#D9d9d9;
   Color: #00b33c;
   Border-Radius: 5px;
   Box-Shadow: 1px White;
   Font-Size: 1.1em;
   Filter: Contrast(1.5);
}
```

```
Ul>Li {
   Display: Block;
}

Li {
      Display: List-Item;
}

Li:Hover {
   Box-Shadow: 1px White;
   Font-Size: 1.1em;
   Filter: Contrast(1.5);
   Color: Black;
}
Img {
   Width: 48px;
   Filter: Contrast(200%);
   Filter: Brightness(1.35);
 }
Ul.Pattern {
   List-Style: None;
   Margin: 0;
   Overflow: Hidden;
   Padding: 0;
   Border-Radius: 0;
}

Ul.Pattern Li {
   Float: Left;
   Margin: 2px;
}


.Dropdown-Menu{
      Display:None;
}

.Title {
      Display: Flex;
      Flex-Direction: Row;
      Justify-Content: Space-Around;
      Margin: 20px 0;
      -Moz-Justify-Content: Space-Around;
}

.Table-Name {
      Font-Size: 32px;
}

.Grade {
      Display: Inline-Block;
      Width: 20px;
      Height: 35px;
}

.Activity {
      Background-Repeat: No-Repeat;
      Background-Position: Bottom;
}
```

```
/*Img {
        Margin-Top: 100px;
}*/
#First-Registration {
        Display: None;
}
#Navbar {
        Display: None;
}
.Schedule {
        Display: None;
}
#Grades {
        Display: None;
}
.Redbox {
        Color: Red;
        Width: 620px;
        Position: Relative;
        Margin: 10%;
        Height: 100px;
        Border-Color: #008a77;
    Border-Style: Solid;
    Padding: 5px;
}

/* Navbar */
.Navbar-Brand {
        Color: #F2f2f2;
    Font-Size: 20px;
    Text-Transform: Uppercase;;
 }
.Navbar-Light {
        Font-Family: Sans-Serif;
    Font-Size: 20px;
    Text-Transform: Uppercase;
 }

.Navbar-Text {
        Margin-Right: 10px;
    Margin-Left: 20px;
}

.Title {
        Padding:1.5em 0;
   }
.Link_L_Arrow::Before {
        Content:"< ";
}

.Link_R_Arrow::After {
        Content:" >";
}

.Navbar-Expand-Lg {
        -Moz-Box-Shadow: 0 4px 4px Rgba(0,0,0,.1);
        Height: 90px;
        Background-Color: #262626;
```

```
}
.Justify-Content-End {
	Margin-Left: 200px;
}


/* Button "Getstarted"*/
.Btn_Start {
	Margin-Top: 70px;
	Margin-Bottom: 70px;
	Text-Align: Center;
	Border-Radius: 20px;
	Box-Shadow: 10px Grayscale;
}

/* Button Color */
.Navbar-Toggler {
	Background-Color: #35bd49;
}
/*  Icons */
#Icons {
   Padding-Top: 30px;
   Vertical-Align: Middle;
}
/* Switcher */
/*#Style-Switcher .Bottom {
   Background: #Fff;
   Color: #252525;
   Padding: 0;
}
#Style-Switcher {
	Left: -189px;
	Position: Fixed;
   Top: 17%;
   Width: 189px;
   Z-Index: 9999;
	Padding: 10px 5px;
}

#Style-Switcher A {
	Background: #Fff;
   Box-Shadow: 2px 2px 0 0 Rgba(0, 0, 0, .08);
}
#Style-Switcher Div H3 {
   Color: #1d1d1d;
   Font-Size: 16px;
   Margin: 8px 3px 12px;
}
*/
/* Footer */
.Fixed-Bottom {
	Height:540px;
	Background-Color:#595959;
	Color:#F2f2f2;
}
.Best_Features {
	Text-Transform: Uppercase;
	Color:#00b33c;
}
(Function (Global, Factory) {
```

```
      Typeof Exports === 'Object' && Typeof Module !== 'Undefined' ? Factory(Exports, Require('Jquery')) :
      Typeof Define === 'Function' && Define.Amd ? Define(['Exports', 'Jquery'], Factory) :
      (Factory((Global.Bootstrap = {}),Global.Jquery));
}(This, (Function (Exports,$) { 'Use Strict';

$ = $ && $.Hasownproperty('Default') ? $['Default'] : $;

Function _Defineproperties(Target, Props) {
  For (Var I = 0; I < Props.Length; I++) {
    Var Descriptor = Props[I];
    Descriptor.Enumerable = Descriptor.Enumerable || False;
    Descriptor.Configurable = True;
    If ("Value" In Descriptor) Descriptor.Writable = True;
    Object.Defineproperty(Target, Descriptor.Key, Descriptor);
  }
}

Function _Createclass(Constructor, Protoprops, Staticprops) {
  If (Protoprops) _Defineproperties(Constructor.Prototype, Protoprops);
  If (Staticprops) _Defineproperties(Constructor, Staticprops);
  Return Constructor;
}

Function _Extends() {
  _Extends = Object.Assign || Function (Target) {
    For (Var I = 1; I < Arguments.Length; I++) {
      Var Source = Arguments[I];

      For (Var Key In Source) {
        If (Object.Prototype.Hasownproperty.Call(Source, Key)) {
          Target[Key] = Source[Key];
        }
      }
    }

    Return Target;
  };

  Return _Extends.Apply(This, Arguments);
}

Function _Inheritsloose(Subclass, Superclass) {
  Subclass.Prototype = Object.Create(Superclass);
  Subclass.Prototype.Constructor = Subclass;
  Subclass.__Proto__ = Superclass;  var Util = {
    TRANSITION_END: 'bsTransitionEnd',
    getUID: function getUID(prefix) {
      do {
            prefix += ~~(Math.random() * MAX_UID); // "~~" acts like a faster Math.floor() here
      } while (document.getElementById(prefix));

      return prefix;
    },
    getSelectorFromElement: function getSelectorFromElement(element) {
      var selector = element.getAttribute('data-target');

      if (!selector || selector === '#') {
        selector = element.getAttribute('href') || '';
      } // If it's an ID
```

```
    if (selector.charAt(0) === '#') {
      selector = escapeId(selector);
    }

    try {
      var $selector = $$$1(document).find(selector);
      return $selector.length > 0 ? selector : null;
    } catch (err) {
      return null;
    }
  },
  reflow: function reflow(element) {
    return element.offsetHeight;
  },
  triggerTransitionEnd: function triggerTransitionEnd(element) {
    $$$1(element).trigger(transition.end);
  },
  supportsTransitionEnd: function supportsTransitionEnd() {
    return Boolean(transition);
  },
  isElement: function isElement(obj) {
    return (obj[0] || obj).nodeType;
  },
  typeCheckConfig: function typeCheckConfig(componentName, config, configTypes) {
    for (var property in configTypes) {
      if (Object.prototype.hasOwnProperty.call(configTypes, property)) {
        var expectedTypes = configTypes[property];
        var value = config[property];
        var valueType = value && Util.isElement(value) ? 'element' : toType(value);

        if (!new RegExp(expectedTypes).test(valueType)) {
          throw new Error(componentName.toUpperCase() + ": " + ("Option \"" + property + "\" provided type \"" + valueType + "\" ") + ("but expected type \"" +
expectedTypes + "\"."));
        }
      }
    }
  };
  setTransitionEndSupport();
  return Util;
}($);
}
Var Name = 'Alert';
  Var Version = '4.0.0';
  Var Data_Key = 'Bs.Alert';
  Var Event_Key = "." + Data_Key;
  Var Data_Api_Key = '.Data-Api';
  Var Jquery_No_Conflict = $$$1.Fn[Name];
  Var Transition_Duration = 150;
  Var Selector = {
    Dismiss: '[Data-Dismiss="Alert"]'
  };
  Var Event = {
    Close: "Close" + Event_Key,
    Closed: "Closed" + Event_Key,
    Click_Data_Api: "Click" + Event_Key + Data_Api_Key
  };
  Var Classname = {
    Alert: 'Alert',
```

```
    Fade: 'Fade',
    Show: 'Show'
  };
  Var Defaulttype = {
    Offset: 'Number',
    Method: 'String',
    Target: '(String|Element)'
  };
  Var Event = {
    Activate: "Activate" + Event_Key,
    Scroll: "Scroll" + Event_Key,
    Load_Data_Api: "Load" + Event_Key + Data_Api_Key
  };
  Var Classname = {
    Dropdown_Item: 'Dropdown-Item',
    Dropdown_Menu: 'Dropdown-Menu',
    Active: 'Active'
  };
nav ul a,
nav .brand-logo {
  color: #444;
}

p {
  line-height: 2rem;
}

.button-collapse {
  color: #26a69a;
}

.parallax-container {
  min-height: 380px;
  line-height: 0;
  height: auto;
  color: rgba(255,255,255,.9);
}
  .parallax-container .section {
    width: 100%;
  }

@media only screen and (max-width : 992px) {
  .parallax-container .section {
    position: absolute;
    top: 40%;
  }
  #index-banner .section {
    top: 10%;
  }
}

@media only screen and (max-width : 600px) {
  #index-banner .section {
    top: 0;
  }
}

.icon-block {
  padding: 0 15px;
}
```

```
.icon-block .material-icons {
  font-size: inherit;
}

footer.page-footer {
  margin: 0;
}
```

ДОДАТОК Б

ВІДЗИВ

на дипломний проект бакалавра
на тему:
«Розробка Web сервісу електронний щоденник учня для закладів середньої освіти з використанням фреймворку JavaScript Atom.»
студента групи 122-19ск-2 Кондрашова Юрія Володимировича

ДОДАТОК В

РЕЦЕНЗІЯ

на дипломний проект бакалавра
на тему:
«Розробка Web сервісу електронний щоденник учня для закладів середньої освіти з використанням фреймворку JavaScript Atom.»
студента групи 122-19ск-2 Кондрашова Юрія Володимировича

ДОДАТОК Г

Відзив керівника економічного розділу

ДОДАТОК Д

ПЕРЕЛІК ФАЙЛІВ НА ОПТИЧНОМУ НОСІЇ

Ім’я файла	**Опис**
Пояснювальні документи	
Диплом.doc	Пояснювальна записка до дипломного проекту. Документ Word.
Диплом.pdf	Пояснювальна записка до дипломного проекту в форматі PDF
Програма	
Idea_board.zip	Архів. Містить коди програми
Презентація	
Презентація.ppt	Презентація дипломного проекту