МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Сумський державний університет
Факультет електроніки та інформаційних технологій
Кафедра комп'ютерних наук

«До захисту допущено»
В.о. завідувача кафедри
_____ Ігор ШЕЛЕХОВ
(підпис)
_____202_ р.

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня бакалавр

зі спеціальності 122 - Комп'ютерних наук,
освітньо-наукової програми «Інформатика»
на тему: «Інформаційна система обліку користувачів E-school»
здобувача групи ІН-06-2 – Бондаренка Олександра Федоровича

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

_____ Олександр БОНДАРЕКНО
(підпис)

Керівник, доцент,
кандидат фізико-математичних наук,
доцент Галина ОЛЕКСІЄНКО _____
(підпис)

Суми – 2024

Сумський державний університет
Факультет електроніки та інформаційних технологій
Кафедра комп'ютерних наук

«Затверджую»
В.о. завідувача кафедри
_____ Ігор ШЕЛЕХОВ
(підпис)

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня бакалавра

зі спеціальності 122 - Комп'ютерних наук, освітньо-професійної програми «Інформатика»
здобувача групи ІН-06-2 Бондаренка Олександра Федоровича

1. Тема роботи: «Інформаційна система обліку користувачів E-school»
затверджую наказом по СумДУ від 22 квітня 2024 року № 0414-VI

2. Термін здачі здобувачем кваліфікаційної роботи *до 29 травня 2024 року*

3. Вхідні дані до кваліфікаційної роботи

4. Зміст розрахунково-пояснювальної записки (перелік питань, що їх належить розробити)
1) Аналітичний огляд предметної області, аналіз аналогів та постановка задачі. 2) Огляд технологій для розробки інформаційної системи обліку користувачів. 3) Розробка Інформаційна система обліку користувачів E-school. 4) Аналіз результатів.

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

6. Консультанти до проекту (роботи), із значенням розділів проекту, що стосується їх

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання «____» _____202_ р.

Завдання прийняв до виконання _____ Керівник _____
(підпис) (підпис)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів кваліфікаційної роботи	Термін виконання	Примітка
1	*Аналітичний огляд предметної області, аналіз аналогів та постановка задачі*	06.05.24-10.05.24	
2	*Аналіз технологій для створення інформаційної системи обліку користувачів*	11.05.24-12.05.24	
3	*Розробка інформаційна система обліку користувачів E-school*	13.05.24-	

		24.05.24	
4	*Тестування інформаційна система обліку користувачів E-school.*	25.05.24- 27.05.24	
5	*Аналіз результатів, оформлення пояснювальної записки*	28.05.25- 31.05.24	

Здобувач вищої освіти _____ Керівник _____
(підпис) (підпис)

АНОТАЦІЯ

Записка: 97 стр., 24 рис., 3 додатки, 17 використаних джерел.

Обґрунтування актуальності теми роботи — тема кваліфікаційної роботи є важливою та актуальною, оскільки спрямована на вирішення практичної проблеми автоматизацій процесів управління та оплати дитячої школи E-School.

Об’єкт дослідження. інформаційні системи обліку користувачів.

Предмет дослідження. Розробка та впровадження інформаційної системи обліку користувачів для дитячої школи E-School, яка допоможе автоматизувати процеси роботи школи.

Мета роботи — розробка та запуск інформаційної системи обліку користувачів, яка дозволяє користувачам зручно та легко реєструватись на майстер класи, батькам оплачувати курси, вчителям керувати процесом відмічання учнів а адміністрації школи автоматизувати більшу кількість роботи школи.

Методи дослідження — аналіз існуючих інформаційних систем, моделювання системи, аналіз інструментів розробки, проектування бази даних та розробка програмної частини.

Результати — Було проведено аналіз інформаційної системи обліку користувачів, існуючого вебсайту та аналогічних рішень. На основі цього сформовано вимоги до проєкту. Визначено задачі. Обрано інструменти для розробки та проектування ІС, а також технології для клієнтської та серверної частин, СКБД і веб-інтерфейс. Розроблено структуру, базу даних, клієнтську та серверну частину ІС. Проведено тестування продукту.

ІНФОРМАЦІЙНА СИСТЕМА, ОБЛІК КОРИСТУВАЧІВ, HTML, CSS, JAVASCRIPT, MYSQL, PHP

ЗМІСТ

ВСТУП

Актуальність. Актуальність інформаційних систем (ІС) обліку користувачів у дитячих школах зростає у зв'язку з постійним розвитком цифрових технологій та необхідністю підвищення ефективності управління освітніми установами. Сучасні ІС дозволяють автоматизувати процеси реєстрації та обліку учнів, зберігання їх особистих даних, успішності, а також комунікацію з батьками. Це сприяє зменшенню паперової роботи, мінімізує ризик людських помилок і дозволяє адміністрації більше часу приділяти безпосередньо навчальному процесу та управлінській роботі.

Об’єкт дослідження. Інформаційні системи обліку користувачів.

Предмет дослідження. Розробка та впровадження інформаційної системи обліку користувачів для дитячої школи E-School, яка допоможе автоматизувати процеси роботи школи.

Гіпотеза. Розроблена інформаційна система обліку користувачів дитячої школи E-School значно покращить процес управління шкільною документацією, забезпечить високий рівень точності та швидкості обробки даних, спростить взаємодію між учнями та вчителями.

Новизна. Система сприяє підвищенню ефективності навчання, оптимізації та автоматизації роботи адміністрації, покращенню загальної якості обліку учнів.

Структура. Дана робота складається зі вступу, аналітичного огляду, постановки задачі, вибір методу розв’язання поставленої задачі, опису програмного забезпечення інформаційної системи, висновків, списку використаних джерел та додатків.

1. ІНФОРМАЦІЙНИЙ ОГЛЯД

1.1 Актуальність роботи

Розробка інформаційних систем (ІС) для обліку користувачів у школі програмування є дуже актуальною в сучасному світі, оскільки освіта інформативних наук стає все більш важливою для розвитку суспільства і економіки.

Причини актуальності проєкту:

1. Зростання попиту на програмістів, є явищем, яке відчувається у всіх куточках світу, і воно є результатом нашого все більш цифрового життя. Програмісти стають своєрідними архітекторами нашого майбутнього, будуючи програмне забезпечення, яке допомагає нам у всьому, від спілкування до управління складними системами.[1]

Зростання попиту можна побачити в усіх сферах життя. Від інтернет-крамниць, які розширюються на всіх континентах, до медичних систем, які автоматизують діагностику та лікування, програмісти грають ключову роль у цьому цифровому перетворенні. Компанії на всіх рівнях існують у світі програмного забезпечення, а багато з них мають постійне бажання збільшувати свої команди розробників.

Однією з ключових причин цього зростання є розширення цифрової технології у всі сфери життя. Від автоматизації виробництва до вдосконалення систем управління транспортом та логістики, програмування стає необхідним елементом. Більше того, інновації в сфері штучного інтелекту, інтернету речей та блокчейну породжують нові можливості, що потребують експертів з програмування.

Крім того, зростає попит на програмістів у зв'язку зі зростанням кількості технологічних стартапів і компаній, що спеціалізуються на інноваціях у сфері інформаційних технологій. Ці компанії відкривають нові ринки та створюють робочі місця для технічно обізнаних фахівців.

Таким чином, зростання попиту на програмістів не лише відображає розвиток сучасного суспільства, але й вказує на ключову роль, яку вони відіграють у створенні майбутнього, яке будується на технологіях.

2. **Ефективне управління навчальним процесом** є важливою складовою успіху будь-якої освітньої установи. Це вимагає ретельного планування, вивчення потреб учнів та вчителів, а також використання сучасних інструментів та технологій.[2]

На сучасному етапі освітнього процесу ефективне управління навчанням включає в себе ряд ключових аспектів. По-перше, це індивідуалізація навчання. Кожен учень має свої власні потреби та способи засвоєння матеріалу, тому важливо враховувати ці особливості при плануванні уроків та використанні різноманітних методів навчання.

Другим важливим аспектом є використання технологій у навчальному процесі. Сучасні інструменти та програмне забезпечення дозволяють вчителям ефективно ведення електронних журналів, створювати інтерактивні уроки та тести, а також відстежувати прогрес кожного учня в режимі реального часу.

Третій аспект - це забезпечення зручного доступу до навчальних матеріалів та ресурсів. Завдяки цифровій технології, учні можуть мати доступ до навчальних матеріалів з будь-якого місця та в будь-який час, що дозволяє їм навчатися більш ефективно та зручно.

Нарешті, ефективне управління навчальним процесом вимагає постійного аналізу та оновлення методів навчання. Світ швидко змінюється, і з ним має змінюватися і навчальний процес. Тому важливо постійно вивчати нові підходи та технології, які можуть підвищити ефективність навчання.

У підсумку, ефективне управління навчальним процесом є ключовим елементом успіху будь-якої освітньої установи. Це вимагає поєднання індивідуального підходу до кожного учня, використання сучасних технологій та постійного оновлення методів навчання.

Отже, розробка ІС для обліку користувачів у школі програмування є актуальною та важливою для ефективності навчального процесу та підготовки молодих людей до цифрового світу.

1.2 Аналіз принципів побудови ІС обліку користувачів

Побудова інформаційних систем(ІС) обліку користувачів включає в себе кілька ключових принципів, які забезпечують їхню ефективність, безпеку та зручність використання. [3]

Принципи побудови інформаційних систем обліку користувачів:

1. Автентифікація користувача: цей принцип передбачає перевірку ідентифікаційних даних користувача, таких як ім'я користувача та пароль. Це може бути реалізовано через різні методи, такі як введення пароля, використання біометричних даних (відбитків пальців, розпізнавання обличчя тощо) або використання токенів.

2. Авторизація доступу: цей принцип визначає, які ресурси або функції системи можуть бути доступні для конкретного користувача після успішної автентифікації. Він визначає права доступу користувачів на основі їхніх ролей або індивідуальних налаштувань.

3. Модульність і ієрархія ролей: система повинна бути модульною, щоб забезпечити простоту розширення та зміни прав доступу в майбутньому. Крім того, може бути ієрархія ролей, де деякі ролі можуть мати доступ до більшої кількості ресурсів або функцій, ніж інші.

4. Керування правами доступу: цей принцип визначає, яким чином адміністратори можуть керувати правами доступу користувачів до різних функцій або ресурсів системи. Включає створення, зміну та видалення облікових записів, а також налаштування рівнів доступу. [4]

5. Моніторинг і аудит: цей принцип передбачає ведення журналу подій, що дозволяє відстежувати дії користувачів в системі, а також спрощує виявлення та реагування на можливі порушення безпеки.

6. Захист даних: цей принцип включає заходи безпеки, такі як шифрування, захист від несанкціонованого доступу та резервне копіювання даних, що забезпечують конфіденційність, цілісність та доступність інформації.

7. Спрощена адміністрація: цей принцип спрямований на забезпечення зручності адміністрування системи обліку користувачів, включаючи інтерфейси для управління правами доступу, створення нових облікових записів та відновлення паролів.

8. Сумісність інтерфейсів: цей принцип передбачає створення інтерфейсів, які легко інтегруються з іншими системами обліку користувачів або додатками, що дозволяє ефективно обмінюватися даними та забезпечує сумісність з іншими інформаційними системами.

9. Підтримка відновлення паролів та безпека: система повинна мати механізми для відновлення забутого пароля та захисту від несанкціонованого доступу, наприклад, через обмеження кількості спроб введення пароля.

10. Зручний інтерфейс: система повинна мати зручний інтерфейс для кожної ролі, щоб забезпечити зручність використання та навігації.

1.3 Аналіз існуючого вебсайту школи E-School

E-School [5] – це школа електроніки, в місті Суми. Головна сторінка зображена на рисунку 1.1.

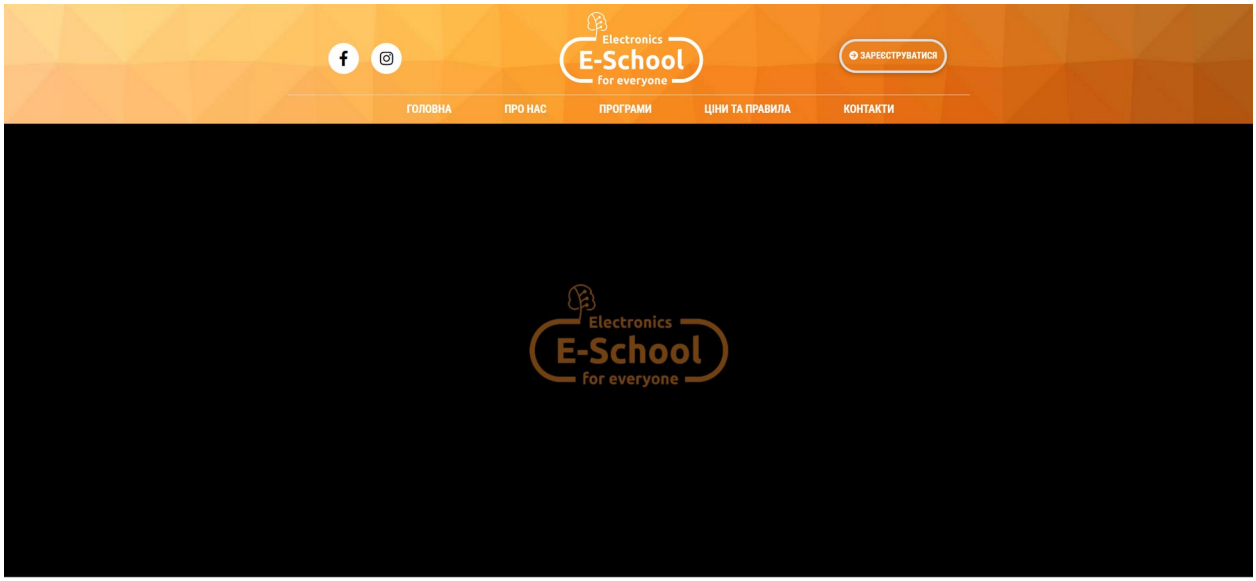


Рисунок 1.1 - Вебсайт дитячої школи «E-School»

Переваги:

– Великий вибір курсів навчання.

– Швидкість завантаження вебсайту.

– Проста та інтуїтивно зрозуміла навігація.

Недоліки:

– Не автоматизована форма реєстрації на майстер-клас(рис. 1.2). Форма реєстрації має скудну кількість полів вводу, тому не можливо вписати дані про батьків, дитину та головне – курс на який реєструється користувач. Також не зовсім зрозуміло чиї саме дані потрібно вводити батьків чи дитини.

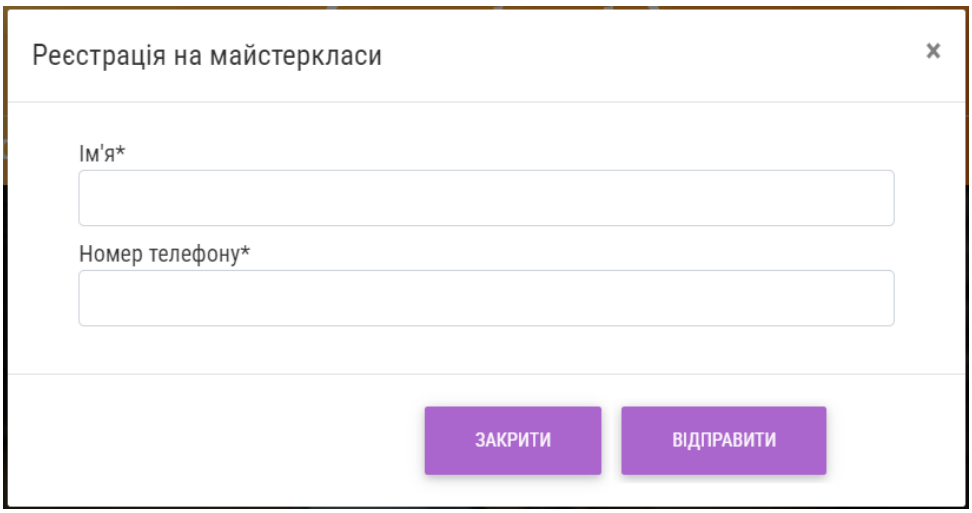


Рисунок 1.2 – Форма реєстрації на майстер-клас

– Переваги школи(рис. 1.3). При наведені мишки на одну з трьох переваг, вона змінює свої стилі. У користувачів може скластись уявлення, що це щось інтерактивне, тобто що на це можна натиснути та щось відбудеться але цього не відбувається, що заплутує користувача.

Рисунок 1.3 – Переваги школи

– Відгуки(рис. 1.4). Не зовсім зрозуміло як саме залишаються відгуки, адже на вебсайті немає особистого кабінету.

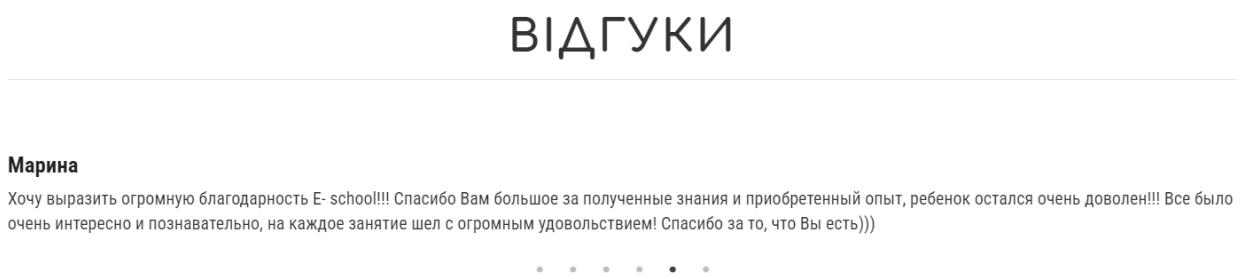


Рисунок 1.4 – Секція відгуки

– Колір кнопки «Реєстрація на майстер класи». Проблема в тому, що даний колір, зелений, не де більше не використовується на вебсайті. Через це складається відчуття, що дана кнопка взята взагалі не з цього вебсайту.

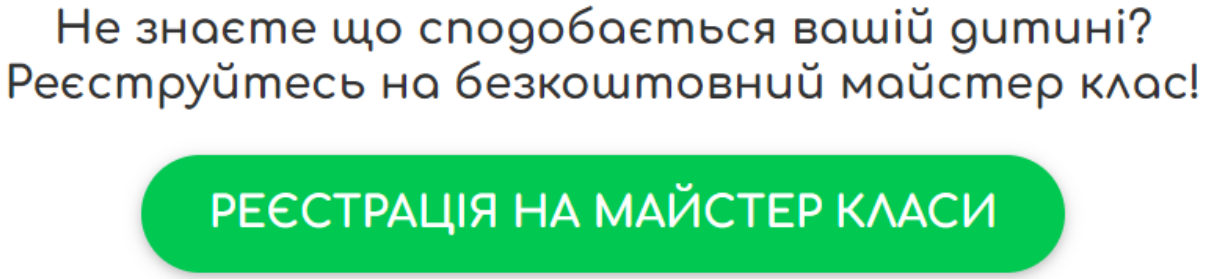


Рисунок 1.5 – Кнопка «Реєстрація на майстер класи»

– Посилання в блоці footer(Рис. 1.6). Як можна побачити з рисунку, посилання мають не коректні розміри, скоріше за все це пов’язано з відступами для тегів, які знаходяться в середині тегу гіперпосилання.

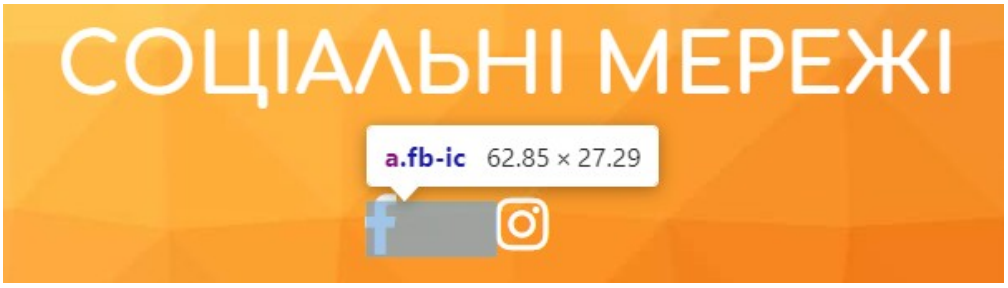


Рисунок 1.6 – Посилання в блоці footer

– Адаптивність. На сторінці «Ціни та правила» відсутній адаптив для таблиці(рисю 1.7). Також, назва даної сторінки не відповідає її наповненню – відсутні правила.

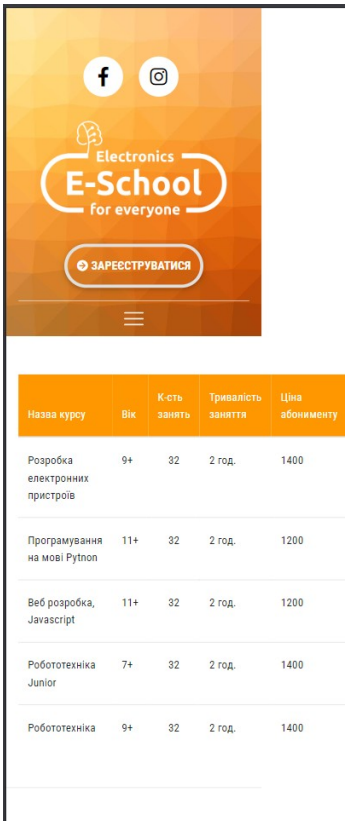


Рисунок 1.7 – Сторінка «Ціни та правила»

Окрім явних проблем з адаптивністю та недоліків у дизайні, було виявлено, що даний ресурс не має особистого кабінету також відсутня

автоматизація реєстрації на майстер клас, адже після заповнення форми реєстрації, дані з неї відправляються на електроніку адресу, після чого адміністрація школи зв’язується з батьками по телефону.

Для сучасного світу, відсутність автоматизації, це застарілий підхід, адже воно має декілька суттєвих недолік:

– Підвищення ризику людських помилок: Ручне введення та обробка даних збільшує ймовірність помилок, таких як неправильне введення інформації, дублювання записів або втрати важливих даних. Це може призвести до недостовірної інформації про учнів, що ускладнює прийняття обґрунтованих управлінських рішень.

– Низька ефективність і витрати часу: Ручна обробка даних потребує значних затрат часу та ресурсів. Вчителі та адміністративний персонал витрачають багато часу на ведення облікових записів, що зменшує їх здатність зосереджуватися на основних навчальних та виховних завданнях.

– Труднощі у відстеженні і доступі до інформації: У випадку відсутності автоматизованих систем, процес пошуку та аналізу інформації про учнів стає складним і тривалим. Це ускладнює моніторинг успішності, відвідуваності та інших важливих аспектів навчального процесу, що може негативно вплинути на якість освіти.

– Обмежені можливості для аналізу даних: Без автоматизованих систем обліку користувачів важко проводити комплексний аналіз даних, що обмежує можливості для покращення навчальних програм і прийняття стратегічних рішень на основі об'єктивних даних.

– Ненадійність і безпека даних: Ручне ведення обліку менш надійне у порівнянні з автоматизованими системами, які зазвичай мають вбудовані механізми захисту даних. Втрата або пошкодження паперових документів може мати серйозні наслідки, тоді як автоматизовані системи забезпечують резервне копіювання та захист даних від несанкціонованого доступу.

1.4 Аналіз аналогів

Аналіз аналогів у розробці вебсайтів допомагає зрозуміти сильні та слабкі сторони конкурентів. Вивчення аналогічних вебсайтів дозволяє визначити найкращі практики та тренди у галузі, що сприяє покращенню користувацького досвіду. Такий аналіз допомагає зменшити ризики та забезпечує більшу конкурентоспроможність вебсайту на ринку.

Для порівняння було обрано дві школи, сайти яких і будуть порівняні, обидві школи базуються у місті Суми.

Перший вебсайт, який буде розглянуто, це «Robot School» [6], - це дитяча школа робототехніки, яка заснована 26 вересня 2015 року.

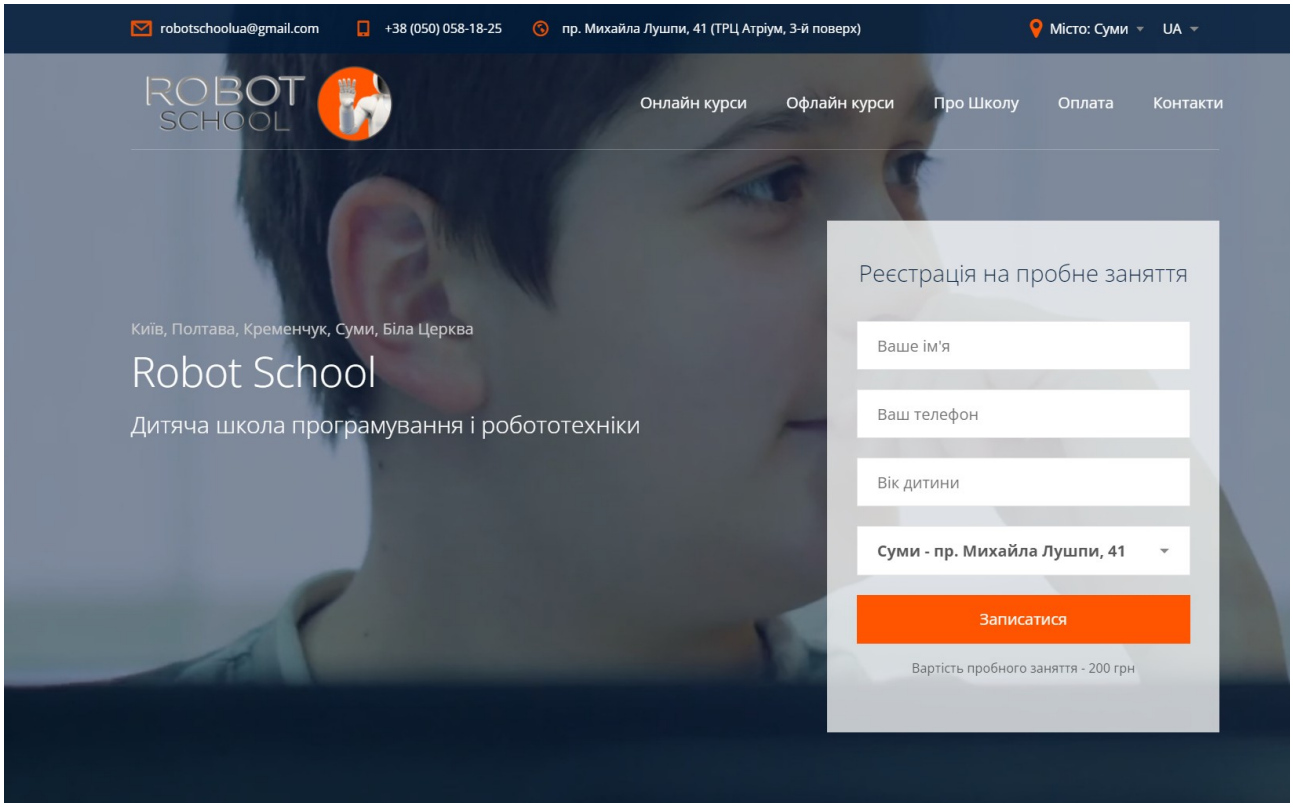


Рисунок 1.8 – Вебсайт дитячої школи «Robot School»

Переваги:

– Навчальні курси та тренування для дітей у вигляді відеоуроків та інтерактивних завдань.

– Простий та зрозумілий дизайн вебсайту,

– Можливість реєстрації для доступу до курсів та стеження за прогресом у навчанні.

– Проста реєстрація на майстер-клас.

– Можливість оплати.

Недоліки:

– Обмежений асортимент курсів порівняно з іншими освітніми платформами.

– Не зрозуміло як відбувається підтвердження оплати, адже вебсайт не пропонує особистого кабінету та рахунку.

Другий вебсайт школи – це «Sumy IT School» [7], головна сторінка зображена на рисунку 1.9.

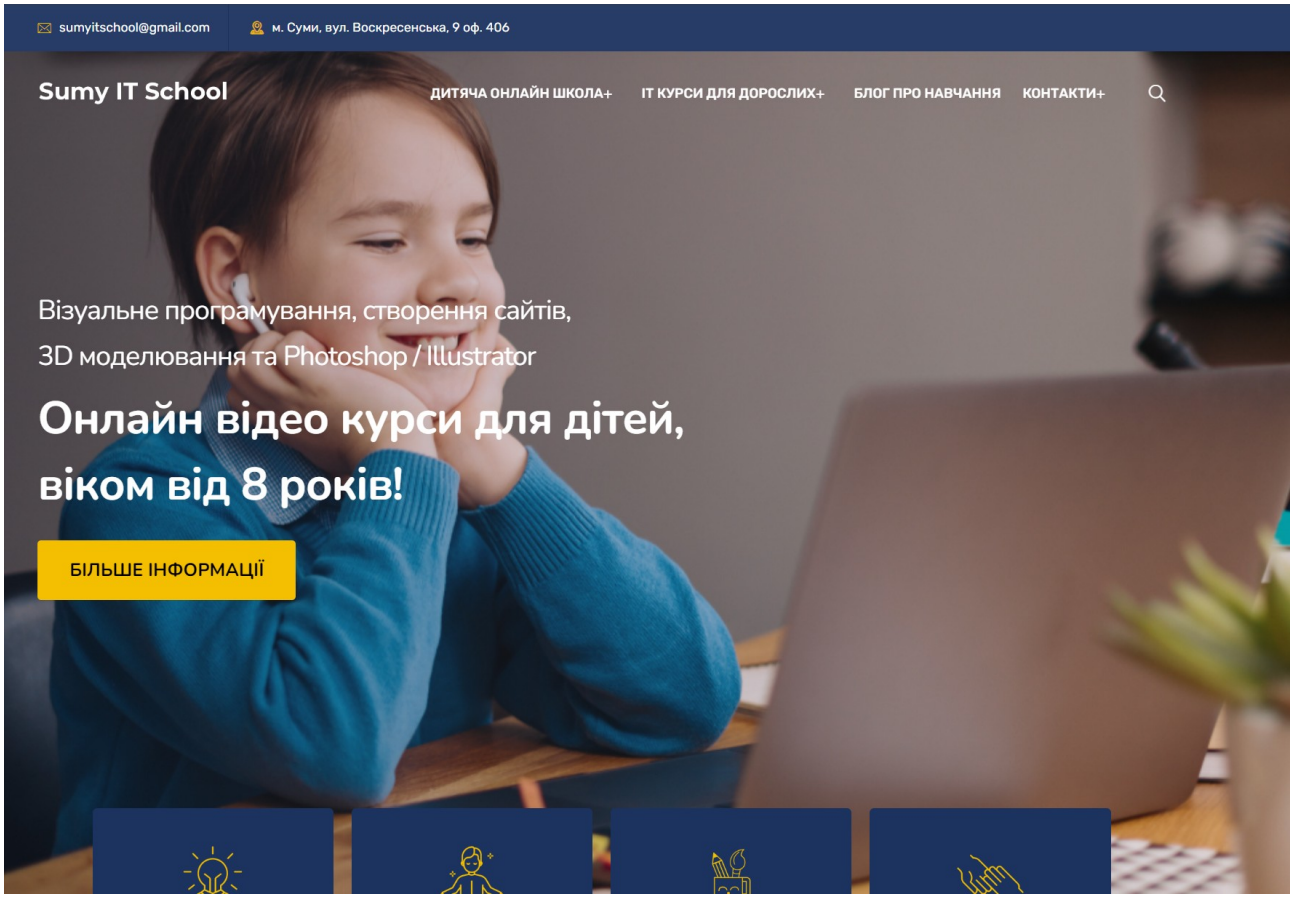


Рисунок 1.9 - Вебсайт дитячої школи «Sumy IT School»

Переваги:

– Великий вибір навчальних курсів.

– Курси для дітей та дорослих.

– Сучасний дизайн.

– Наявність публічної оферти.

Недоліки:

– Відсутній посилань на деяких кнопках.

– Відсутність сторінки курсу та детальної інформації про нього.

– Запис на курс відбувається через гугл форму.

– Відсутність форми зворотного зв’язку, якщо виникнуть питання.

До загальних недоліків можна віднести відсутність інформації про вчителів та керівний склад школи. Дана проблема актуальна для усіх прикладів, які були наведені в цьому підпункті.

Після проведення детального аналізу усіх вебсайтів, було отримано базове розуміння як має виглядати та який функціонал мати вебсайт ІТ-школи. Враховуючи особливість теми, того, що на вебсайті має бути присутні ролі та різні рівні користувачів були сформовані вимоги до майбутньої ІС.

1.5 Вимоги до ІС

Вимоги до проєкту визначаються на основі побажань замовника та аналізі аналогів проведеним у пункті 1.4.Вимоги не є остаточні в процесі розробки вимоги можуть змінюватись за попередньою домовленістю між виконавцем та замовником.

Поділимо вимоги до інформаційної системи управління користувачів E-School на функціональні та нефункціональні вимоги.

Не функціональні вимоги:

– Кольори: помаранчевий(його відтінки), чорний, білий

– Шрифти: Roboto, Roboto Condensed.

– Технології розробки: HTML5, CSS3\SCSS, JS, PHP.

– Завантаження на хостинг під доменним ім’ям (e-school.in.ua)

Функціональні вимоги:

– Наявність ролей (рівнів доступу): повинно бути присутні, як мінімум п’ять ролей, це адміністратор, модератор, вчитель, учень, батьки.

- Адміністратор – відповідає за наповнення ІС інформацією, може її додавати, редагувати та\або змінювати. Має доступ до адмін-панелі.
- Модератор – займається модерацією, а саме: відповіддю на запитання, призначення ролей вчителя, реєструє учнів у системі, тощо.
- Вчитель – відмічає присутніх в журналі, призначає відпрацювання.
- Батьки – перевіряють відвідуваність своїх дітей.

Роль адміністратора та модератора може бути об’єднана в одну.

Розділі та їх наповнення:

– Головна сторінка:

- Шапка сайту: логотип, навігаційне меню, особистий кабінет.
- E-School це: опис про школу.
- Відгуки.
- Перелік курсів: кнопка реєстрації на майстер-клас.
- Контакти: місцезнаходження, телефони, соц. мережі.
- Підвал вебсайту: копірайт.
- Викладачі: перелік викладачів, коротка інформація про них(що вони ведуть), їх зображення.
- Часті питання: питання та відповідь на них.
- Переваги школи.

– Про нас - сторінка з описом школи, секція з викладачами, відгуки та фотографії з навчання, особливості навчання.

– Програми - перелік усіх програм, які представлені в школі(Карточка товару: зображення, заголовок, підзаголовок, складність курсу, з якого віку, ціна, кнопка переглянути). Загальний опис програми, підхід до навчання, переваги курсу над курсами в інших школах.

– Ціни та правила – перелік курсів, їх ціни. Правили(Обов’язки та права) для учнів, батьків, вчителів.

– Контакти - контакти школи, мапа від зупинки до корпусу

Інший функціонал:

При натисканні на кнопку «Реєстрація на майстер клас» має з’являтись модальне вікно, на якому представлена форма реєстрації, яка складається з наступних полів вводу:

– Ім’я одного з батьків – обов’язково для заповнення.

– Телефон одного з батьків – обов’язково для заповнення.

– Ім’я дитини – обов’язково для заповнення.

– Вік дитини – обов’язково для заповнення.

Перелік курсів, які доступні, для введеного віку(можна обрати декілька)– обов’язково для заповнення

При відправці цієї форми, дані з неї заносяться в БД. Повідомлення про реєстрацію приходить на електронну адресу.

Обов’язкова валідація полів. Повідомлення про успішне відправлення.

1.6 Постановка задачі

В результаті роботи необхідно реалізувати наступні задачі:

– Обрати середовище розробки макетів.

– Вибір та обґрунтування середовища розробки для клієнтської та серверної частини.

– Обрати технології для клієнтської частини.

– Обрати технології для серверної частини.

– Обрати СКБД.

– Обрати веб-інтерфейс для керування БД.

– Розробити структуру вебдодатку.

– Розробити базу даних.

– Розробити програмний продукт.

– Провести тестування програмного продукту.

2. ВИБІР МЕТОДІВ РОВ'ЯЗАННЯ ПОСТАВЛЕНОЇ ЗАДАЧІ

2.1Інструменти розробки вебсайтів

2.1.1 Середовище розробки макетів

Для більшості вебсайтів, перед початком розробки, створюється його макет, те як він має виглядати в продакшені. Створюється даний макет на основі ТЗ\вимог до майбутнього проєкту. Для створення макетів існує велика кількість різних програм найвідоміша серед них це Adobe Photoshop але у нього є проблема він не призначений для створення макетів вебсайтів, з цього випливає проблема для розробника в тому, що немає вихідних стилів. Саме цю проблему вирішує програма під назвою Figma.

Figma - це онлайн інструмент для дизайну, який дозволяє створювати макети, прототипи та спільно працювати над ними в реальному часі. Переваги, через які було обрано дане середовище та переваги які роблять його найкращим середовищем для розробки макетів:

– Онлайн доступ: Ви можете працювати з Figma з будь-якого пристрою, що має доступ до Інтернету. Це робить спільну роботу з командою простою, навіть якщо ви всі знаходитесь у різних місцях.

– Спільна робота в реальному часі: Кілька користувачів можуть одночасно працювати над макетами, спостерігаючи за змінами в реальному часі. Це дозволяє команді швидко спілкуватися та вносити зміни без необхідності чекати на оновлення.

– Легка інтеграція: Figma легко інтегрується з іншими інструментами, такими як Slack, Jira, Zeplin тощо, що полегшує роботу з командою та управління проектами.

– Автоматичне збереження історії: Figma автоматично зберігає історію змін, що дозволяє повертатися до попередніх версій макетів у разі потреби.

– Багатофункціональність: Figma має велику кількість інструментів для дизайну, включаючи векторні форми, текстові блоки, макети та компоненти. Це дозволяє створювати деталізовані та професійні макети.

– Реалістичні прототипи: За допомогою Figma можна створювати прототипи, які відображають реальну взаємодію користувача з продуктом. Це допомагає уточнювати дизайн та отримувати зворотній зв'язок від команди та клієнтів.

2.1.2 Середовище розробки вебсайту

Для створення вебсайту можна використовувати багато середовищ розробки починаючі з стандартного, який є на кожному комп’ютері на ОС Windows – це блокнот. Якщо потрібне щось більш серйозніше, то можна обрати інтегроване середовище розробки(IDE) – такі як WebStorm, PhpStorm або Eclipse.

Якщо потрібне щось простіше ніж IDE але більш сучасніше ніж стандартний блокнот, в такому випадку слід звернути увагу на текстові редактори коду, наприклад на такий як VS Code.

Visual Studio Code (VS Code) - це безкоштовний текстовий редактор, який відкритий для розробки на різних платформах, розроблений компанією Microsoft. Ключові особливості та переваги VS Code:

– Крос-платформеність: VS Code підтримується на Windows, macOS та Linux, що дозволяє розробникам працювати у своєму улюбленому середовищі незалежно від операційної системи.

– Розширюваність: VS Code має велику кількість розширень, які дозволяють налаштовувати його під ваші потреби та зручності. Ви можете встановлювати розширення для різних мов програмування, фреймворків, інструментів відлагодження та багато іншого.

– Швидкодія: VS Code працює досить швидко, навіть на менш потужних комп'ютерах. Він має мінімальне споживання ресурсів, що робить його ідеальним вибором для розробки навіть на старіших пристроях.

– Вбудовані інструменти для розробників: VS Code має ряд вбудованих інструментів, таких як відлагоджувач, система керування версіями (Git), вбудовані термінал та інші, що полегшують роботу розробників та підвищують продуктивність.

– Спільність з іншими сервісами Microsoft: Оскільки VS Code розроблений компанією Microsoft, він має глибоку інтеграцію з іншими сервісами Microsoft, такими як Azure, Visual Studio Team Services (VSTS), Azure DevOps, що полегшує розробку та розгортання програмних продуктів.

У підсумку - Visual Studio Code є потужним та зручним інструментом для розробників будь-яких мов програмування та технологій. Його швидкодія, розширюваність та вбудовані інструменти роблять його популярним серед професіоналів та початківців у світі розробки програмного забезпечення.

2.1.3 Технології для розробки клієнтської частини

Клієнтська частина вебсайту - це та частина, яка відповідає за відображення та взаємодію з користувачем безпосередньо у браузері. Вона включає в себе HTML, CSS та JavaScript, що виконуються на боці клієнта (тобто у браузері користувача).

HyperText Markup Language

HTML - це стандартна мова розмітки, що використовується для створення веб-сторінок та їх структурированого відображення у веб-браузерах.

Характеристики HTML:

– Розмітка: HTML використовує теги для визначення різних елементів на веб-сторінці, таких як заголовки, параграфи, списки, таблиці, форми тощо. Кожен тег має свою властивість та призначення, що дозволяє створювати структуровану сторінку.

– Структура документа: HTML дозволяє визначити загальну структуру документа, включаючи заголовок, тіло та нижню частину. Заголовок зазвичай містить метадані, такі як назва сторінки та посилання на зовнішні

ресурси, тіло містить основний вміст сторінки, а нижня частина може містити додаткові відомості або скрипти.

– Взаємодія з іншими мовами: HTML легко поєднується з іншими мовами, такими як CSS для стилізації вигляду сторінки та JavaScript для додавання інтерактивності.

– Крос-платформенність: HTML підтримується усіма сучасними веб-браузерами та може відображатися на різних пристроях та платформах.

– Семантика: HTML дозволяє використовувати семантичні теги для визначення значення елементів на сторінці. Це допомагає пошуковим системам краще розуміти контент та поліпшує доступність сторінок для користувачів з інвалідністю.

HTML є основою веб-розробки та відіграє важливу роль у створенні структурованого та доступного веб-вмісту. Його простий синтаксис та потужність дозволяють розробникам створювати різноманітні та інтерактивні веб-сторінки для користувачів у всьому світі.

Cascading Style Sheets:

CSS - це мова стилів, яка використовується для оформлення та вигляду веб-сторінок, написаних мовою HTML. [9]

Характеристики CSS:

– Розділення відображення та структури: CSS дозволяє розділити зовнішній вигляд веб-сторінки від її структури, що робить код більш читабельним та підтримуваним.

– Селектори: CSS використовує селектори для визначення, які елементи HTML стилізувати. Селектори можуть бути елементами, класами, ідентифікаторами, псевдо-класами тощо.

– Властивості та значення: CSS визначає стилі елементів за допомогою властивостей та їх значень. Наприклад, властивість "color" визначає колір тексту, а "font-size" - розмір шрифту.

– Каскадність та спадкування: CSS використовує каскадність для визначення порядку застосування стилів до елементів, які мають кілька правил. Властивості також можуть успадковуватися від батьківських елементів, що дозволяє швидко та ефективно стилізувати багато елементів.

– Модульність: CSS дозволяє використовувати класи та ідентифікатори для стилізації конкретних елементів, що полегшує повторне використання та підтримку коду.

– Адаптивність та чуйний дизайн: CSS дозволяє створювати адаптивні та чуйні веб-сторінки, які коректно відображаються на різних пристроях та розмірах екранів.

CSS є важливою складовою веб-розробки, яка дозволяє розробникам створювати привабливі та користувацько орієнтовані веб-інтерфейси. Він дозволяє стилізувати різноманітні елементи та створювати консистентний та професійний вигляд для веб-сторінок.

JavaScript:

JavaScript - це високорівнева, об'єктно-орієнтована мова програмування, яка використовується для створення динамічних та інтерактивних веб-сайтів. [10]

Характеристики JavaScript:

– Клієнтська та серверна сторони: JavaScript може використовуватися як на клієнтській стороні (у веб-браузерах користувачів) так і на серверній стороні (за допомогою платформ, таких як Node.js). Це дозволяє розробникам створювати повноцінні веб-додатки та веб-сервіси.

– Динамічність: JavaScript дозволяє змінювати зміст, стилі та поведінку елементів на сторінці в реальному часі без перезавантаження сторінки. Це робить веб-сайти більш інтерактивними та захопливими для користувачів.

– Об'єктно-орієнтованість: JavaScript підтримує об'єктно-орієнтоване програмування, що дозволяє розробникам створювати об'єкти та класи для організації та структурування коду.

– Функціональність: JavaScript підтримує функціональне програмування, що дозволяє використовувати функції як першокласні об'єкти та працювати з ними як зі змінними.

– Широке застосування: JavaScript використовується для реалізації різних функцій на веб-сайтах, від валідації форм та анімації до обробки даних та взаємодії з сервером.

– Багатоплатформенність: JavaScript підтримується усіма сучасними веб-браузерами та може виконуватися на різних платформах, що дозволяє розробникам створювати веб-додатки, які працюють у будь-якому середовищі.

Розглянемо як працює JavaScript у браузері. Ці цікава тема, адже існує думка, що «JavaScript працює на JavaScript», адже браузери використовують JavaScript у своїй роботі.

– Веб-браузери та JavaScript двигуни: Коли відкривається веб-сторінка у веб-браузері, він інтерпретує (або виконує) JavaScript код, що міститься на сторінці. В кожному сучасному веб-браузері є вбудований JavaScript двигун, такий як V8 (у Google Chrome), SpiderMonkey (у Firefox), JavaScriptCore (у Safari) тощо. Ці двигуни відповідають за виконання JavaScript коду.[11]

– Обробка JavaScript коду: Коли веб-браузер зустрічає JavaScript код на сторінці, він виконує його по одному рядку або блоку за раз. Цей процес включає аналізування коду, виконання інструкцій та здійснення будь-яких необхідних дій, таких як зміна вмісту сторінки, обробка подій, робота з мережею тощо. [12]

– Взаємодія з DOM і браузерними API: JavaScript може взаємодіяти з HTML і CSS через DOM (Document Object Model), який представляє структуру документа як дерево об'єктів. Він також може використовувати різноманітні

браузерні API для доступу до функціональності браузера, такої як робота з вікнами, куки, локальне сховище, взаємодія з мережею та багато іншого.

JavaScript є однією з найпопулярніших мов програмування у світі, яка відіграє важливу роль у веб-розробці. Її потужність та гнучкість дозволяють створювати різноманітні та інноваційні веб-додатки для користувачів у всьому світі.

2.1.4 Технології для розробки серверної частини

Розробка серверної частини веб-додатка - це процес створення та налаштування серверного програмного забезпечення, яке відповідає за обробку запитів від клієнтів, виконання логіки бізнес-логіки та взаємодії з базою даних. Саме тут існує дуже великий вибір технологій, які можна обрати для проєкту. При виборі технологій або стеку важливо чітко розуміти необхідність у тих чи інших технологіях та не обирати стек тільки тому, що він популярний, вибір має базуватись на необхідностях а не на можливостях тих чи інших технологій\ стеків.

Мова програмування PHP:

PHP (Hypertext Preprocessor) - це скриптова мова програмування загального призначення, яка використовується для розробки веб-додатків та динамічних веб-сайтів. [13]

Характеристики PHP:

– Веб-орієнтованість: PHP спеціально створений для розробки веб-додатків. Він може вбудовуватися безпосередньо в HTML-код та генерувати веб-сторінки на льоту, що робить його дуже зручним для роботи з веб-сайтами та динамічним вмістом.

– Легкість вивчення та використання: PHP має простий та зрозумілий синтаксис, що робить його досить легким для вивчення та використання, навіть для початківців. Він має багатий набір вбудованих функцій та бібліотек, які полегшують розробку.

– Платформонезалежність: PHP може працювати на різних операційних системах, таких як Windows, Linux, macOS та інші, що робить його універсальним рішенням для розробки веб-додатків.

– Широкі можливості: PHP підтримує різні типи баз даних, включаючи MySQL, PostgreSQL, SQLite, та інші. Він також може взаємодіяти з різними сервісами та API через HTTP-запити та розбір JSON-даних.

– Розширюваність: PHP має велику кількість розширень та бібліотек, які дозволяють розширити його функціональність та використовувати готові рішення для різних задач.

– Активна спільнота розробників: PHP має велику та активну спільноту розробників, яка регулярно вносить внески до розвитку мови та підтримує її.

PHP є однією з найпопулярніших мов програмування для веб-розробки та використовується для створення різноманітних веб-додатків, від невеликих особистих сайтів до великих корпоративних веб-порталів. Його зручність, ефективність та широкі можливості роблять його популярним вибором серед розробників усього світу.

Apache HTTP Server:

Apache - це відкрите програмне забезпечення для створення та обслуговування веб-серверів. [14]

Характеристики сервера Apache:

– Система розповсюдження та відкритий код: Apache є безкоштовним програмним забезпеченням з відкритим вихідним кодом, що означає, що ви можете використовувати, змінювати та розповсюджувати його вільно.

– Платформо-незалежність: Apache підтримується на різних операційних системах, таких як Windows, Linux, macOS та інші, що робить його універсальним рішенням для різних середовищ.

– Масштабованість: Apache може працювати як з невеликими веб-сайтами, так і з великими веб-порталами та веб-додатками. Він підтримує

обробку тисяч одночасних підключень та може бути налаштований для оптимальної продуктивності в залежності від потреб користувача.

– Модульність: Apache має модульну архітектуру, що дозволяє розширювати його функціональність за допомогою різних модулів. Наприклад, ви можете використовувати модулі для обробки PHP, Perl, Python, SSL, компресії даних, аутентифікації тощо.

– Надійність: Apache є стабільним та надійним веб-сервером, який довгий час є одним з найпопулярніших рішень у світі. Він має широку спільноту користувачів та розробників, що активно підтримує та розвиває цей проект.

– Безпека: Apache має ряд вбудованих механізмів безпеки, таких як доступові контролі, шифрування з'єднань за допомогою SSL, захист від DDOS-атак та інші.

У підсумку, Apache HTTP Server є потужним та надійним рішенням для створення веб-серверів будь-якої складності. Він підтримується широкою спільнотою користувачів та має ряд вбудованих функцій для ефективного обслуговування веб-сайтів та веб-додатків.

Дана програма використовується для створення локального серверу та запуску вебсайту на локальній машині, що дозволяє працювати з БД.

2.1.5 Вибір СКБД

СКБД означає "Система керування базами даних". Це програмне забезпечення для створення, управління та оптимізації баз даних. СКБД дозволяє користувачам зберігати, організовувати і отримувати доступ до даних шляхом використання структурованих запитів. Вони є невід'ємною частиною багатьох сучасних програмних застосунків та систем, які потребують ефективного управління великими обсягами інформації. Найпопулярніші СКБД включають такі системи, як MySQL, PostgreSQL, Oracle Database та Microsoft SQL Server.

MySQL- це відкрите програмне забезпечення для управління базами даних, яке надає можливість зберігання, організацію та маніпулювання структурованою інформацією. [15]

Характеристики MySQL:

– Відкритий вихідний код та безкоштовність: MySQL є відкритою системою управління базами даних з ліцензією GPL (General Public License), що означає, що ви можете використовувати, змінювати та розповсюджувати її безкоштовно.

– Масштабованість: MySQL підтримує широкий діапазон потужностей, від невеликих веб-сайтів до великих корпоративних систем. Він може працювати на одному сервері або бути частиною розподіленої архітектури з кластерами серверів.

– Швидкодія: MySQL відомий своєю високою швидкодією та ефективністю обробки запитів до бази даних. Він має оптимізований движок для роботи з великими обсягами даних та великою кількістю одночасних підключень.

– Підтримка стандартів SQL: MySQL дотримується стандартів мови SQL (Structured Query Language), що робить його сумісним з іншими системами управління базами даних та інструментами.

– Розширюваність: MySQL має ряд розширень та доповнень, які дозволяють розширити його функціональність. Це можуть бути модулі для підтримки різних типів даних, виконання резервного копіювання, реплікації даних та інше.

– Активна спільнота користувачів та розробників: MySQL має велику та активну спільноту користувачів та розробників, яка надає підтримку, допомогу та розвиток системи. Це забезпечує стабільність та надійність продукту.

MySQL є однією з найпопулярніших систем управління базами даних у світі. Він широко використовується для веб-розробки, бізнес-додатків та інших

сфер застосування завдяки своїй швидкодії, надійності та відкритому вихідному коду.

2.1.6 Веб-інтерфейс для керування БД

PhpMyAdmin - це веб-інтерфейс для керування та адміністрування базами даних MySQL за допомогою веб-браузера. [16]

Характеристики phpMyAdmin:

– Веб-інтерфейс: phpMyAdmin забезпечує зручний веб-інтерфейс для взаємодії з базами даних MySQL без необхідності встановлення спеціального програмного забезпечення або використання командного рядка.

– Управління базами даних: phpMyAdmin дозволяє вам керувати структурою та вмістом баз даних, включаючи створення, редагування та видалення таблиць, виконання SQL-запитів, імпорт та експорт даних тощо.

– Зручність використання: phpMyAdmin має інтуїтивний і простий у використанні інтерфейс, що дозволяє навіть початківцям швидко освоїти його та виконувати різноманітні завдання з керування базами даних.

– Підтримка функцій MySQL: phpMyAdmin підтримує багато функцій та можливостей MySQL, включаючи роботу зі збереженими процедурами, тригерами, індексами, обмеженнями, операціями з кешуванням та багато іншого.

– Міжнародна підтримка: phpMyAdmin підтримується міжнародною спільнотою розробників та користувачів, що забезпечує регулярні оновлення, виправлення помилок та розвиток функціональності.

– Безпека: phpMyAdmin має ряд вбудованих механізмів безпеки, таких як автентифікація, авторизація, SSL-шифрування, обмеження доступу та інші, що робить використання цього інструменту безпечним.

Технології, обрані для реалізації проєкту, вибрані через їх переваги, надійність та простоту. Вони є базовими для веброзробки і забезпечують достатні можливості для виконання поставленої задачі.

3. ПРОГРАМНА РЕАЛІЗАЦІЯ

3.1 Структура вебдодатку

Під час проведення аналізу аналогів вебзастосунків дитячих шкіл програмування, стало зрозуміло уявлення майбутнього продукту, його структуру та дизайн. У кожного вебсайту існує своя структура, яка включає в себе ряд вебсторінок, перехід між якими відбувається за рахунок гіперпосилань. Якісно розроблена структура вебсайту надає користувачу можливість без технічних перешкод користуватися вебсайтом.

На рисунку 3.1 приведена структура майбутньої ІС школи програмування.

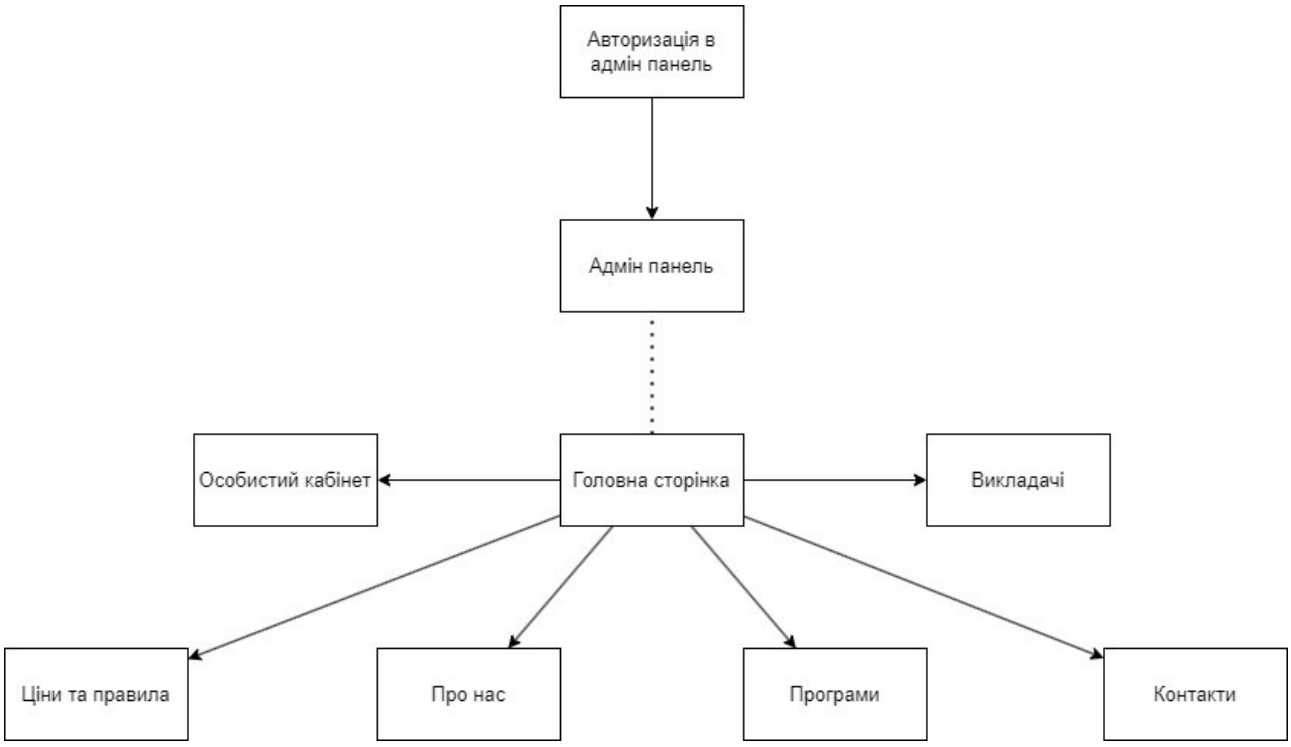


Рисунок 3.1 – Структура ІС дитячої школи програмування

З структури ІС (рис. 2.1), можна зробити наступний висновок, що вебзастосунок складається з семи вебсторінок які доступні користувачам, ще дві вебсторінки доступні адміністрації ІС. Структура вебсторінок складається з наступних блоків - це шапка та підвал сайту, однаковий для усіх вебсторінок окрім адміністративної панелі, а також блоку, який відповідає сутті

вебсторінки, наприклад на вебсторінці «Програми» - перелік усіх програм, які надає школа.

3.2 Розробка бази даних

БД дозволяє вебсайту динамічно генерувати сторінки на основі запитів користувачів, забезпечує зручний доступ до інформації та можливість взаємодії з відвідувачами, а також дозволяє адміністраторам вебсайту керувати та оновлювати його вміст.

Виходячи з вимог та структури вебсайту представленого вище, БД має описувати наступні процеси та сутності:

- Користувачі, їх ролі.
- Групи та курси.
- Курси, їх ціни, рекомендований вік.
- Можливість записатись на майстер-клас, для користувача.

Важливо зазначити, що у процесі розробки БД може змінюватись та доповнюватись, запропонований варіант не є остаточним.

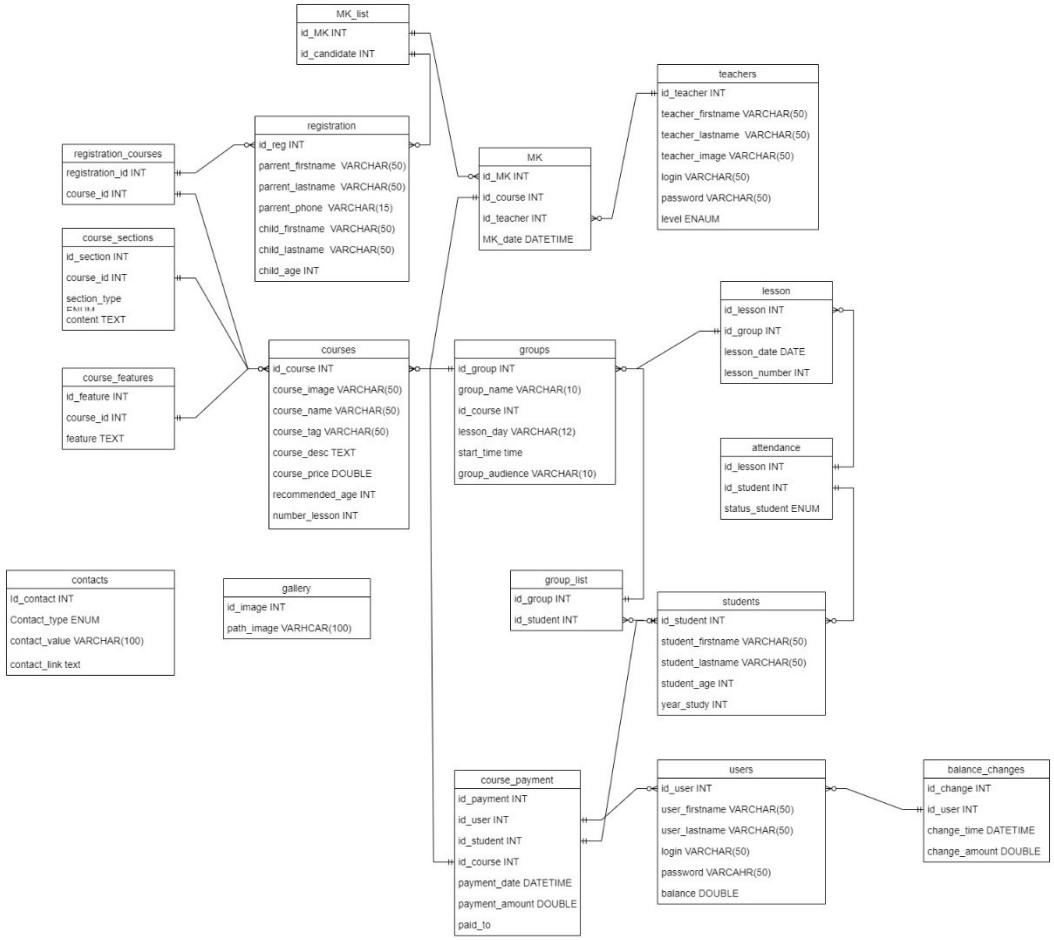


Рисунок 3.2 – Концептуальна модель БД [17]

Відношенню «contacts» відповідає повна ФЗ:id_contact: → contact_type, contact_value, contact_link

Відношенню «gallery» відповідає повна ФЗ: id_image: → path_image

Відношенню «group_list» відповідає повна ФЗ: id_group, id_student

Відношенню «students» відповідає повна ФЗ: id_student: → student_firstname, student_lastname, student_age, year_student

Відношенню «attendance» відповідає повна ФЗ: id_lesson, id_student, status_student

Відношенню «lesson» відповідає повна ФЗ: id_lesson: → id_group, lesson_date, lesson_number

Відношенню «groups» відповідає повна ФЗ: id_group: → group_name, id_coursem lesson_day, start_time, group_audience

Відношенню «courses» відповідає повна ФЗ: id_course: → course_imagem course_name, course_tag, course_desc, course_price, recommended_age.

Відношенню «course_features» відповідає повна ФЗ: id_feature: → course_id, feature

Відношенню «course_sections» відповідає повна ФЗ: id_sections: → course_id, section_type, content

Відношенню «registration_courses» відповідає повна ФЗ: registration_id: → course_id

Відношенню «registration» відповідає повна ФЗ: id_reg: → parrent_firstname, parrent_lastname, parent_phone, child_firstname, child_lastname, child_age

Відношенню «MK_list» відповідає повна ФЗ: id_MK, id_candidate

Відношенню «MK» відповідає повна ФЗ: id_MK: → id_course, id_teacher, MK_date

Відношенню «teachers» відповідає повна ФЗ: i_teacher: → teacher_firstname, teacher_lastname, teacher_image, login, password, level

Відношенню «lesson» відповідає повна ФЗ: id_lesson: → id_group, lesson_date, lesson_number

Проаналізувавши сутність, використовувані в моделі ІС, перейдемо до реалізації структури БД. Для цього представимо імена необхідних таблиць, атрибутів, типів, їх призначення та обмеження (табл. 3.1)

Таблиця 3.1 Структура БД

Таблиця	Поле	Зміст	Тип	Ключі	Обмеження
attendance	id_lesson	Номер лекції	INTEGER	FK	Не пустий
	id_student	Номер студента	INTEGER	FK	Не пустий
	status_student	Статус студента	ENUM		Не пустий
balance_changes	id_change	Номер зміни	INTEGER	PK	Не пустий
	id_user	Номер користувача	INTEGER	FK	Не пустий
	change_time	Час зміни	DATETIME		Не пустий
	change_amount	Сума на яку змінився баланс	DOUBLE		Не пустий
contacts	id_contact	Номер запису	INTEGER	PK	Не пустий
	contact_type	Тип контакту	ENUM		Не пустий
	contact_value	Значення контакту	VARCHAR(100)		
	contact_link	Посилання контакту	TEXT		

Таблиця 3.1 (Продовження) Структура БД

courses	id_course	Номер курсу	INTEGER	PK	Не пустий
	course_image	Зображення курсу	VARCHAR(50)		Не пустий
	course_name	Ім’я курсу	VARCHAR(50)		Не пустий
	course_tag	Короткий опис курсу	VARCHAR(50)		Не пустий
	course_desc	Опис курсу	TEXT		Не пустий
	course_price	Ціна курсу	DOUBLE		Не пустий
	recommended_age	Рекомендований вік	INT		
	number_lessons	Кількість лекцій у курсі	INT		
course_features	id_feature	Номер запису	INT	PK	Не пустий
	course_id	Номер курсу	INT	FK	Не пустий
	feature	Пункт списку	TEXT		Не пустий
course_payments	id_payment	Номер оплати	INT	PK	Не пустий
	id_user	Номер користувача	INT	FK	Не пустий
	id_student	Номер студента	INT	FK	Не пустий
	id_course	Номер курсу	INT	FK	Не пустий
	payment_date	Дата оплати	DATETIME		Не пустий

Таблиця 3.1 (Продовження) Структура БД

	payment_amount	Значення платежу	DOUBLE		Не пустий
	paid_to	Дата дії оплати	DATE		
course_sections	id_section	Номер секції	INT	PK	Не пустий
	course_id	Номер курсу	INT	FK	Не пустий
	section_type	Тип секції	ENUM		Не пустий
	content	Вміст секції	TEXT		Не пустий
gallery	id_image	Номер зображення	INT	PK	Не пустий
	path_image	Шлях до зображення	VARCHAR(50)		Не пустий
groups	id_group	Номер групи	INT	PK	Не пустий
	group_name	Назва групи	VARCHAR(10)		Не пустий
	id_course	Номер курсу	INT	FK	Не пустий
	lesson_day	День проведення уроку	VARCHAR(12)		Не пустий
	start_time	Час початку уроку	TIME		Не пустий
	group_audience	Кабінет групи	VARCHAR(10)		Не пустий
	id_teacher	Номер вчителя	INT	FK	Не пустий

Таблиця 3.1 (Продовження) Структура БД

group_list	id_student	Номер студента	INT	FK	Не пустий
	id_group	Номер групи	INT	FK	Не пустий
lessons	id_lesson	Номер уроку	INT	PK	Не пустий
	id_group	Номер групи	INT	FK	Не пустий
	id_substitute_teacher	Номер вчителя, який замінює	INT	FK	Не пустий
	lesson_date	Дата уроку	DATE		Не пустий
	lesson_number	Номер уроку	INT		Не пустий
master_class	id_master_class	Номер майстер класу	INT	PK	Не пустий
	id_course	Номер курсу	INT	FK	Не пустий
	id_teacher	Номер вчителя	INT	FK	Не пустий
	master_class_date	Дата майстер класу	DATETIME		Не пустий
master_class_ list	id_master_class	Номер майстер класу	INT	FK	Не пустий
	id_candidate	Номер кандидату	INT	FK	Не пустий
registrations	id_reg	Номер зареєстрованого користувача	INT	PK	Не пустий
	parrent_firstname	Ім’я одного батьків	VARCHAR(50)		Не пустий
	parrent_lastname	Прізвище одного батьків	VARCHAR(50)		Не пустий

Таблиця 3.1 (Продовження) Структура БД

	parrent_phone	Телефон одного батьків	VARCHAR(15)		Не пустий
	child_firstname	Ім’я дитини	VARCHAR(50)		Не пустий
	child_lastname	Прізвище дитини	VARCHAR(50)		Не пустий
	child_age	Вік дитини	INT		Не пустий
registration_courses	registration_id	Номер реєстрації	INT	FK	Не пустий
	course_id	Номер курсу	INT	FK	Не пустий
students	id_student	Номер учня	INT	PK	Не пустий
	student_firstname	Ім’я учня	VARCHAR(50)		Не пустий
	student_lastname	Прізвище учня	VARCHAR(50)		Не пустий
	student_age	Вік учня	INT		Не пустий
	year_study	Кількість років навчання	INT		Не пустий
	id_parrent	Номер батьків	INT	FK	Не пустий
teachers	id_teacher	Номер вчителя	INT	PK	Не пустий
	teacher_firstname	Ім’я вчителя	VARCHAR(50)		Не пустий
	teacher_lastname	Прізвище вчителя	VARCHAR(50)		Не пустий
	teacher_image	Зображення вчителя	VARCHAR(50)		Не пустий

Таблиця 3.1 (Продовження) Структура БД

	login	Логін вчителя	VARCHAR(50)		Не пустий
	password	Пароль вчителя	VARCHAR(50)		Не пустий
	level	Рівень вчителя	ENUM		Не пустий
users	id_user	Номер користувача	INT	PK	Не пустий
	user_firstname	Ім’я користувача	VARCHAR(50)		Не пустий
	user_lastname	Прізвище користувача	VARCHAR(50)		Не пустий
	user_patronymic	По батькові користувача	VARCHAR(50)		Не пустий
	login	Логін користувача	VARCHAR(50)		Не пустий
	password	Пароль користувача	VARCHAR(50)		Не пустий
	balance	Баланс користувача	DOUBLE		Не пустий

Для автоматизації процесу створення записів в таблиці «course_payments» та «balance_changes» були написані збережена процедура та тригер, запит представлений у додатку В.

3.3 Серверна частина

Основною функцією цієї частини інформаційної системи є робота з базою даних. Для цього були розроблені відповідні класи та їх методи. Приклад такого класу та його методу наведено нижче.

```
class Course {
    private $mysql;

    public function __construct(){
      $db = new DB();
      $this->mysql = $db->get_connection();
    }
    public function get_all_courses(){
      $result = $this->mysql->query("SELECT * FROM `courses`");

      if(!$result){
        return array("message" => "Помилка запиту або таблиця 'courses'
пуста!");
      }
      $courses = array();

      while($row = $result->fetch_assoc()){
        array_push($courses, $row);
      }

      return $courses;
    }
  }
```

Даний клас описує курси, які представлені для сайті школи. Метод «get_all_courses» повертає усю інформацію про курси, яка присутня в БД.

Виклик методів класу відбувається двома способами, перший це в місці де це потрібно. Наприклад, коли потрібно отримати перелік усіх курсів:

```
<?php
                $courses = $Course->get_number_courses(3);

                foreach($courses as $course){
```

```
            ?>
            <div class="curses__curs">
              <div class="cours__image">
                <a href="./pages/course.php?id_course=<?php echo
$course["id_course"]?>"
                  ><img src="./img/courses/<?php echo
$course["course_image"] ?>" alt=""
                /></a>
              </div>
              <div class="cours__title">
                <?php echo $course["course_name"] ?>
              </div>
              <div class="cours__text">
                <?php echo $course["course_desc"] ?>
              </div>
              <div class="cours__price">
                Вартість: <span><?php echo $course["course_price"] ?
></span> / місяць
              </div>
              <div class="cours__button">
                <a href="./pages/course.php?id_course=<?php echo
$course["id_course"]?>">Детальніше</a>
              </div>
            </div>
            <?php
              }
            ?>
```

В такому випадку виклик методу відбувається в тому місці, файлу, де будуть використовуватись дані. З іншого боку є другий варіант, це коли виклик методу класу відбувається напряму в файлі, який описує клас, наприклад:

```
 if($_SERVER['REQUEST_METHOD'] === 'POST'){
    $data = json_decode(file_get_contents('php://input'), true);

    $login = $data["login"];
    $password = $data["password"];

    $Teacher = new Teacher();
```

```
    $result = $Teacher->authorization($login, $password);

    if($result["status"]){
      $_SESSION["teacher_auth_id"] = $result["id_teacher"];
    }

    echo json_encode($result);
  }
```

В такому випадку дані, в даному прикладі – це логін та пароль, відправляють за допомогою JavaScript, а саме fetch-запиту:

```
fetch("../php/Teacher.php", {
        method: "POST",
        headers: {
          "Content-Type": "application/json",
        },
        body: JSON.stringify(data),
      })
        .then((data) => {
          if (data.status) {
            window.location.href = "../personal_office/teacher.php";
          }
        })
```

Цей підхід має декілька переваг, перша й основна це запит без перезавантаження сторінки. Другий – це так звана «декомпозиція», коли частини функціонального коду знаходяться окремо від коду який «створює» сторінку.

3.4 Клієнтська частина

ІС була розроблена у єдиному стилі, основними кольорами стали білий, темно сірий та відтінки помаранчевого.

Структура сторінок схожа, на всіх вебсторінок для користувачів представлені обов’язкові блоки – шапка(рис 3.3) та підвал вебсайту (рис. 3.4)

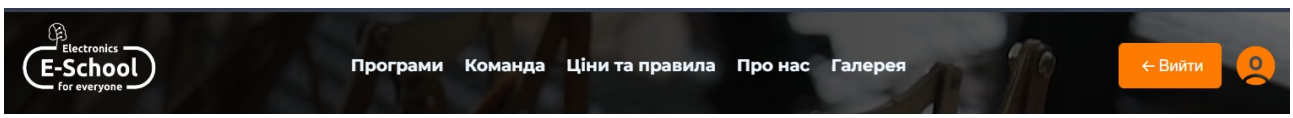


Рисунок 3.3 – Шапка вебсайту

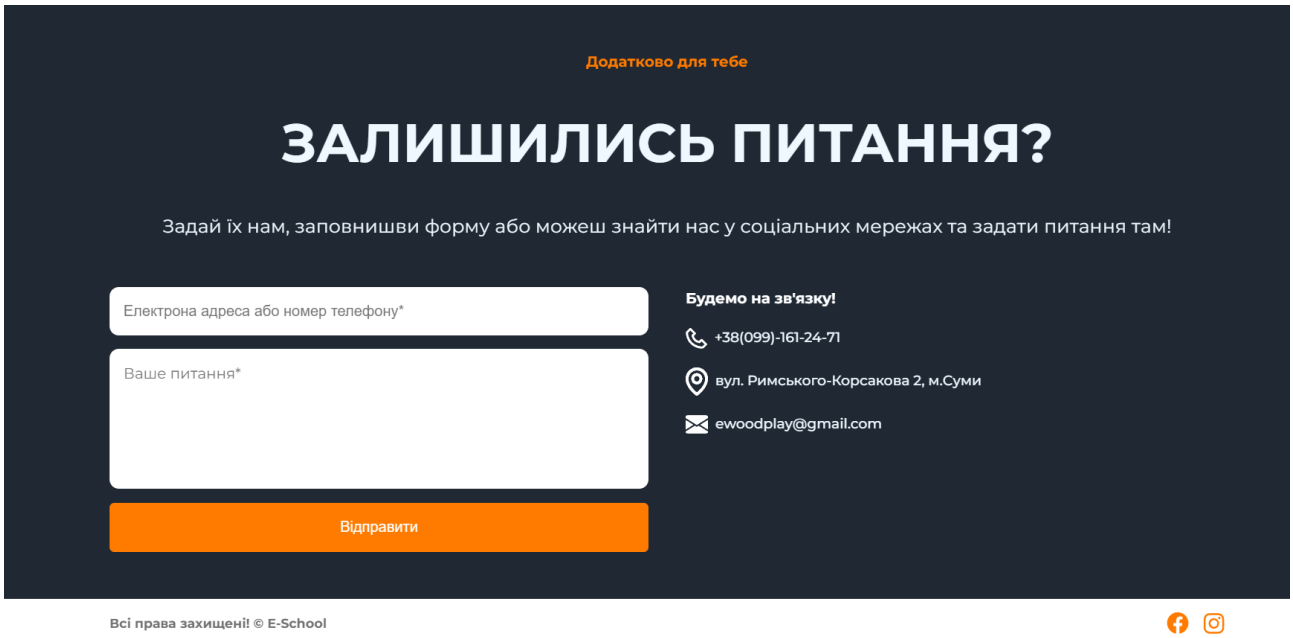


Рисунок 3.4 – Підвал вебсайту

Так як, вебсайт адаптивний під різні екрани, його зовнішній вигляд змінюється в залежності від розміру екрану. При цьому функціонал залишається незмінним.

На рисунку 3.5 зображений вигляд шапки сайту при самому мінімальному розширенні екрану у 320 пікселів.

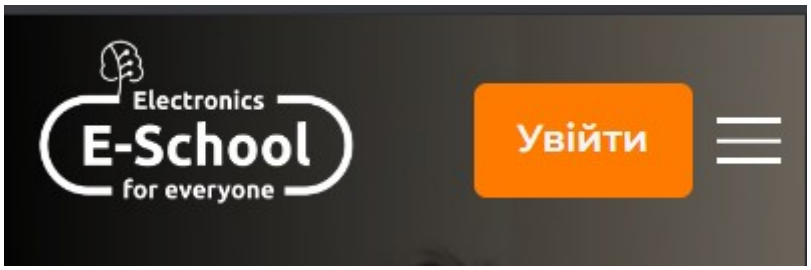


Рисунок 3.5 – Шапка вебсайту на мобільному пристрою

При такому розширені екрану навігаційне меню не вміщується в шапку вебсайту, тому було створено бургер меню, що дозоляє сховати меню. Якщо користувач натисне на бургер-меню, то отримає доступ до навігаційного меню. При цьому переміщуватись по основній сторінці користувач не зможе як і взаємодіяти з нею.

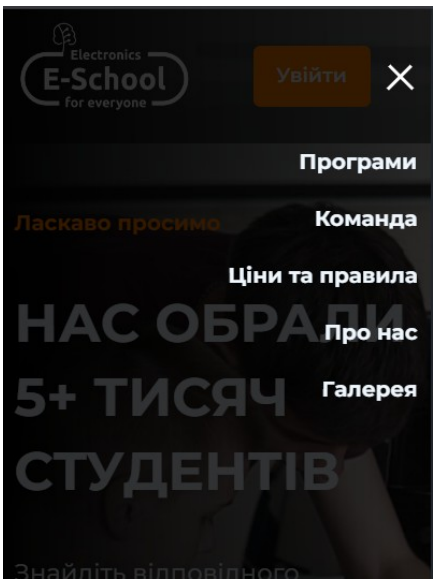


Рисунок 3.6 – Навігаційне меню на мобільному пристрою

Окрім двух вище описаних блоків на кожній сторінці для користувача присутній блок для швидкого переходу в самий вверх сторінки(рис. 3.7)

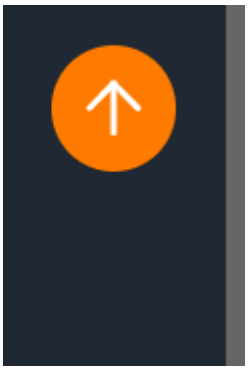


Рисунок 3.7 – Блок для швидкого переходу до гори сторінки

З’являється він, при переміщені екрану від гори вебсайту на 300 пікселів. При кліку переміщає екран користувача на початок сторінки.

```
const BUTTON_VISIBLE_PIXEL = 300;

  const buttonToTop = document.querySelector(".link__totop");

  addEventListener("scroll", () => {
    if (BUTTON_VISIBLE_PIXEL < window.scrollY) {
      buttonToTop.classList.add("_visible");
    } else {
      buttonToTop.classList.remove("_visible");
```

```
  }
});

buttonToTop.addEventListener("click", () => {
  const html = document.querySelector("html");
  html.scrollTo({ top: 0, behavior: "smooth" });
});
```

Для деяких кнопок, створена певна анімація «стрибка», коли кнопка переміщується в гору, після чого повертається в стартове положення.

На головній сторінці присутній блок з перевагами школи, де при наведенні мишки користувача на один з блоків, він розвертається, надаючи більш загальний опис. Його роботу графічно зображено на рисунку 3.8, з ліва блок до наведення, з права – після.

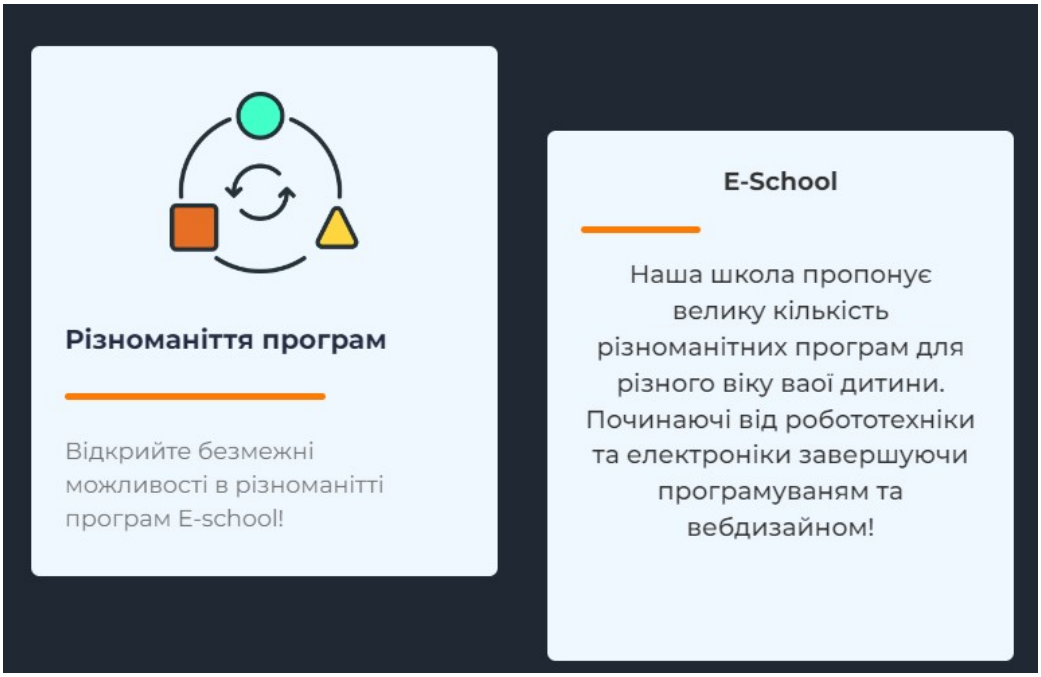
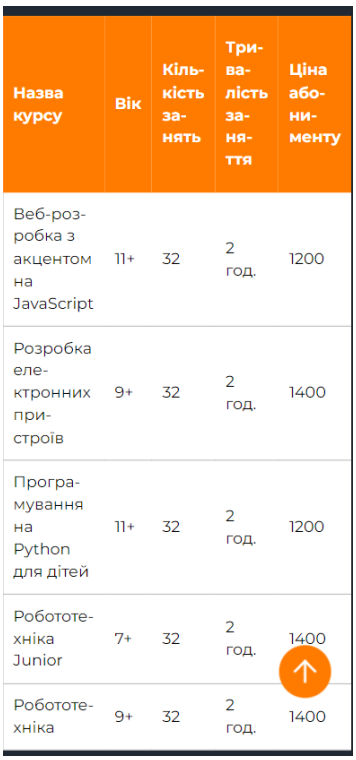


Рисунок 3.8 – Блок з перевагами школи

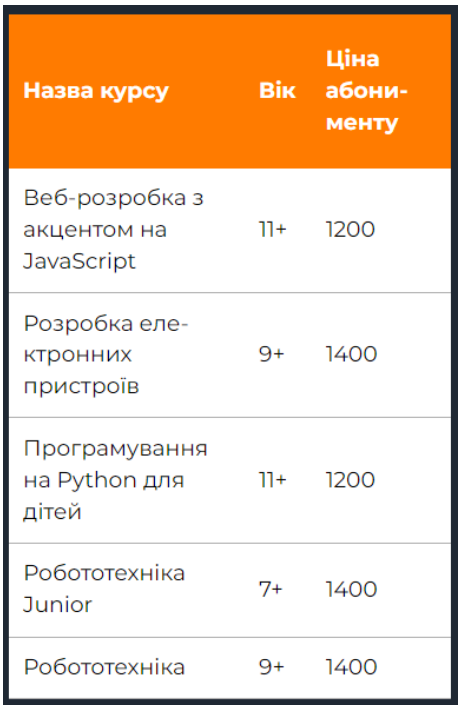
У процесі розробки сторінки «Ціни та правила» було виявлено, що на мобільних пристроях таблиця має низьку читабельність та поганий вигляд(рис 3.9).



Назва курсу	Вік	Кіль-кість за-нять	Три-ва-лість за-ня-ття	Ціна або-ни-менту
Веб-роз-робка з акцентом на JavaScript	11+	32	2 год.	1200
Розробка еле-ктронних при-строїв	9+	32	2 год.	1400
Програ-мування на Python для дітей	11+	32	2 год.	1200
Робототе-хніка Junior	7+	32	2 год.	1400
Робототе-хніка	9+	32	2 год.	1400

Рисунок 3.9 – Таблиця з цінами

Через це було прийнято рішення на мобільних пристроях показувати меншу кількість інформації(рис. 3.10)



Назва курсу	Вік	Ціна абони-менту
Веб-розробка з акцентом на JavaScript	11+	1200
Розробка еле-ктронних пристроїв	9+	1400
Програмування на Python для дітей	11+	1200
Робототехніка Junior	7+	1400
Робототехніка	9+	1400

Рисунок 3.10 – Таблиця з цінами

Це було реалізовано за допомогою скрипту:

```
 if (document.querySelector(".price__table")) {
  const elements = document.querySelectorAll(".element");

  window.addEventListener("resize", () => {
    if (window.innerWidth <= 510) {
      elements.forEach((e) => {
        e.classList.add("_hidden");
      });
    } else {
      elements.forEach((e) => {
        e.classList.remove("_hidden");
      });
    }
  });
}
```

3.5 Функціональна частина

На деяких вебсторінках присутній функціонал реєстрації на майстер клас[Додаток А]. Відбувається це через заповнення полів у формі(рис. 3.11), яка, в свою чергу, знаходиться на модальному вікні.

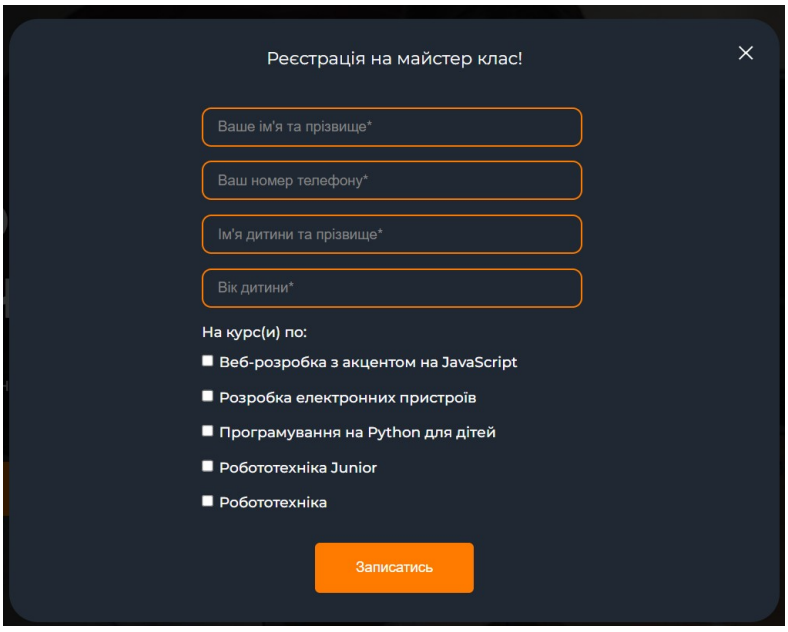


Рисунок 3.11 – Форма реєстрації на майстер-клас

Для даної форми присутня валідація полів вводу. Якщо поле не пройде валідацію, то користувач побачить відповідне повідомлення, рисунок 3.12

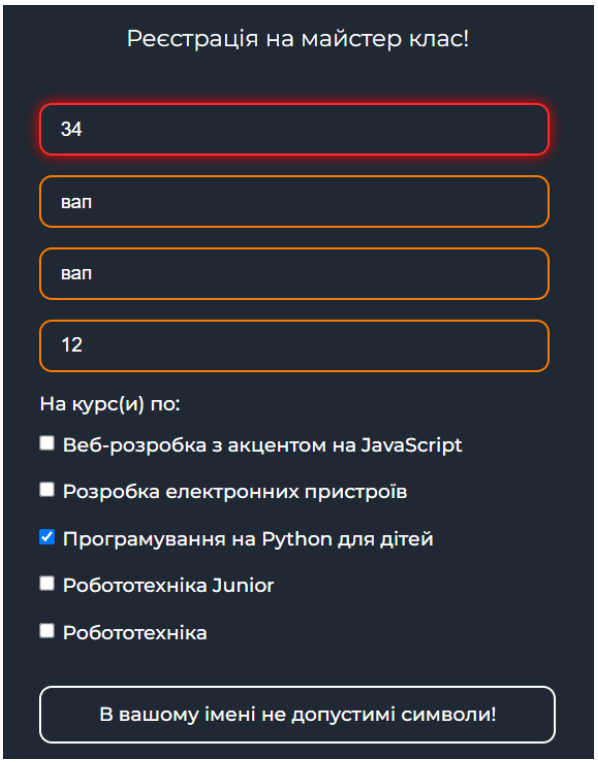


Рисунок 3.12 – Повідомлення про помилку

Якщо всі поля пройшли валідацію, дані з форми відправляються у метод класу «MasterClass»[Додаток Б], де розроблений функціонал, який записує дані у БД.

Для вчителів розроблений особистий кабінет(рис. 3.12), де у них є перелік усіх груп, які вони можуть відсортувати, та відмітити присутніх.

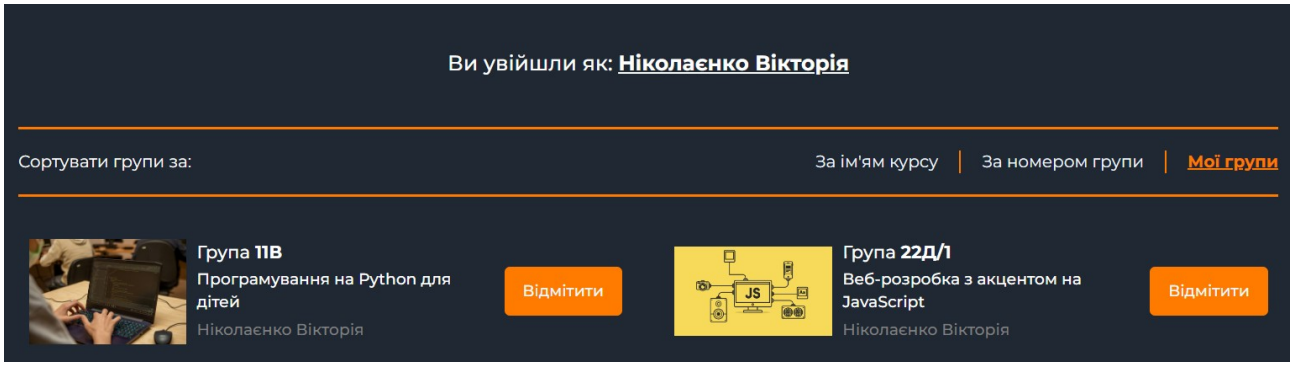


Рисунок 3.13 – Особистий кабінет вчителя

Для того, щоб потрапити на цю сторінку потрібно зареєструватись, через відповідну форму(рис. 3.13), яка з’являється при кліку на кнопку «Увійти», в шапці сайту.

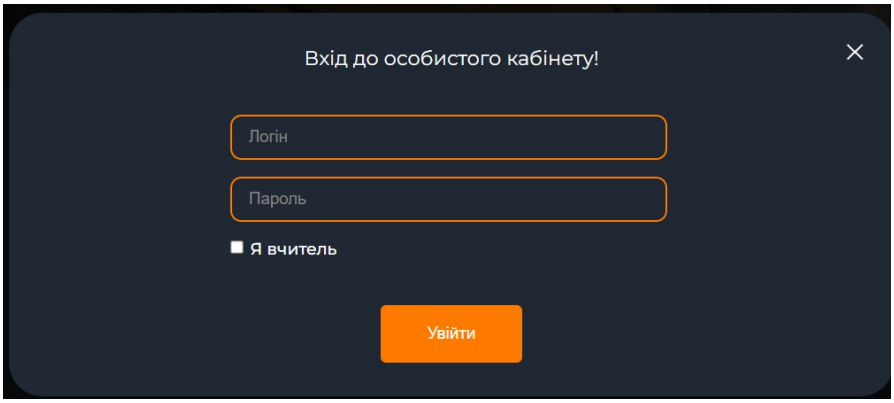


Рисунок 3.14 – Форма авторизації

При натисканні на кнопку «Відмітити» нас перенаправляє на журнал групи(рис. 3.15)

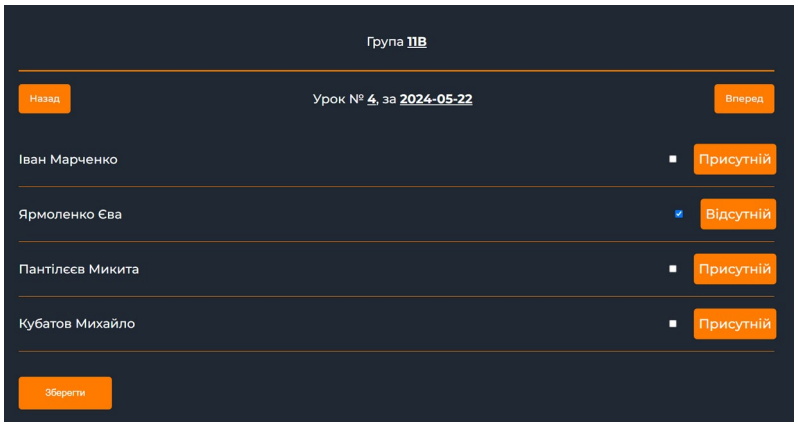


Рисунок 3.15 – Журнал групи

В цьому журналі є можливість:

– Відмітити присутність учня, при кліку на кнопку «Присутній».

– Змінити присутність, при кліку на кнопку «Відсутній»

– Зберегти дані в БД, при кліку на відповідну кнопку.

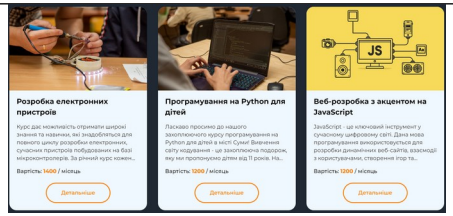
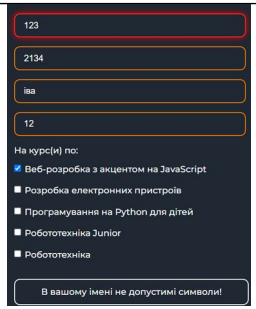
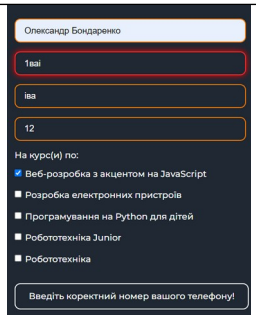
– Подивитись дані за попередній урок, кнопка «Назад»

– Подивитись дані за наступний урок, якщо такий був або буде сьогодні.

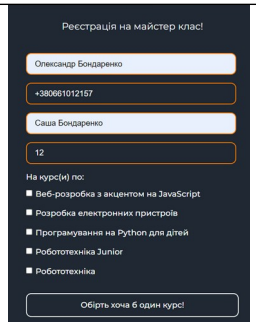
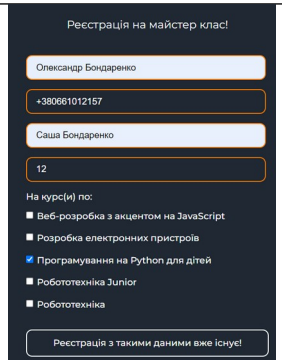
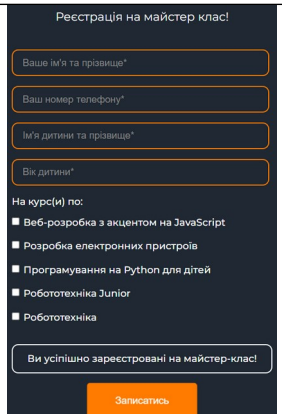
3.6 Тестування

Тестування проводилося методом чорного ящика. Результати тестування записані в таблицю 3.2.

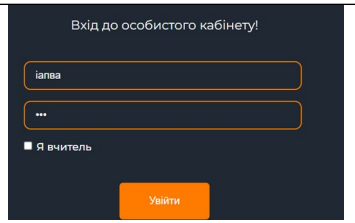
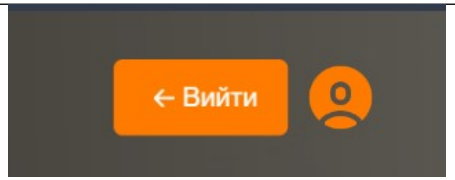
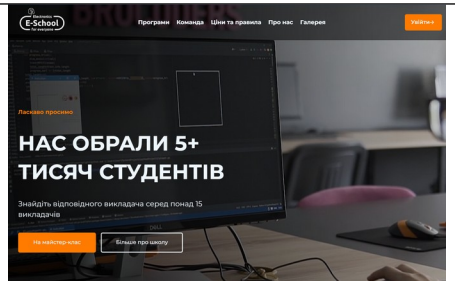
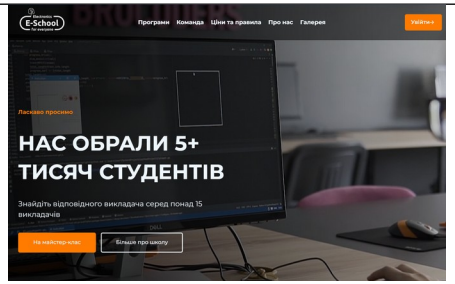
Таблиця 3.2 - Тестування вебзастосунку

№	Умова, що тестується	Очікуваний результат	Фактичний результат	1/0
1	Випадкове виведення курсів	Випадкове виведення курсів		1
2	Введення некоректних даних в поле «ім’я»	Повідомлення про помилку		1
3	Введення некоректних даних в поле «номер телефону»	Повідомлення про помилку		1

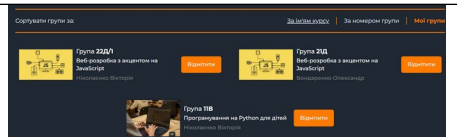
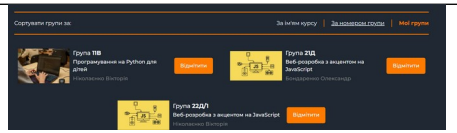
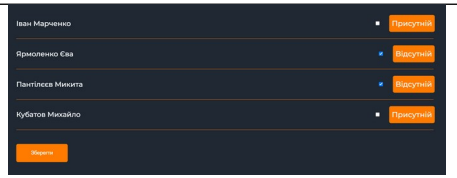
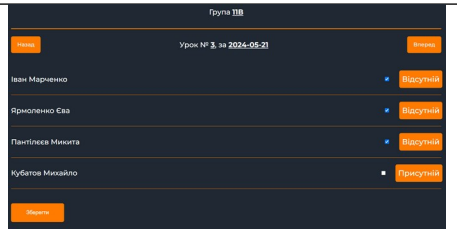
Таблиця 3.2(Продовження) - Тестування вебзастосунку

4	Спроба реєстрації без обраного курсу	Повідомлення про помилку		1
5	Спроба реєстрації з однаковими даними	Повідомлення про помилку		1
6	Введення коректних унікальних даних	Реєстрація на майстер клас		1

Таблиця 3.2(Продовження) - Тестування вебзастосунку

7	Спроба увійти під неправильними даними	Вхід не відбувається		1
8	Спроба входу з коректними даними	Спроба вдала		1
9	Спроба перейти в особистий кабінет не зареєструвавшись	Переадресація на головну сторінку		1
10	Спроба перейти на сторінку іншого вчителя	Переадресація на головну сторінку		1

Таблиця 3.2(Продовження) - Тестування вебзастосунку

11	Сортування груп за ім’ям курсу	Сортування груп		1
12	Сортування груп за номером групи	Сортування груп		1
13	Відображення груп певного викладача	Відображення груп певного викладача		1
14	Відмічання групи на сьогоднішньому уроці	Дані зберігаються		1
15	Змінна даних на попередньому уроці	Дані зберігаються		1

ВИСНОВКИ

Під час аналітичного огляду було виявлено, що системи обліку користувачів стали невід'ємною частиною сучасних вебсистем. Вони ефективно оптимізують процеси роботи школи, а саме оплату курсів, відмічання учнів, спрощують ведення бухгалтерії. У результаті аналізу принципу побудови ІС обліку користувачів були отримані аспекти надійної та зручної системи для обліку користувачів. На основі даних знань був проведений аналіз аналогів дітях шкіл програмування а також старого вебсайту E-School[1]. Під час аналізу були виявлені типові переваги та недоліки подібних вебсайтів. В результаті були сформовані вимоги до майбутньої ІС обліку користувачів. Основними цілями якої стали автоматизація та зручність у користуванні.

Під часу вибору інструментів для розробки були обрані: VS Code, Figma та phpMyAdmin. Як СКБД перевагу було віддано MySQL, на її основі було створено БД E-School. Для самого процесу проектування були обрані базові технології веброзробки: HTML, CSS, JavaScript, PHP.

На основі вимог було розроблена структура майбутнього проєкту, розроблено та спроектовано БД, яка відповідає усім вимогам та бізнес процесам проєкту. Після чого була створена верстка та розроблені PHP-класи для керування функціоналом ІС.

У результаті роботи, була отримана інформаційна система обліку користувачів E-School. Яка повністю відповідає сучасним тенденціям у вебдизайні, та функціоналом повністю відповідному вимогам, що робить даний проєкт добрим прикладом ІС з обліком користувачів. Окрім цього, даний проєкт було спроектовано таким чином, щоб залишилась можливість для подальших розробок та впроваджень.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Pidru4niki [Електронний ресурс] – Режим доступу до ресурсу: https://pidru4niki.com/11230410/buhgalterskiy_oblik_ta_audit/zagalnopriynyati_printsipi_sistemi_obliku
2. E-Schools [Електронний ресурс] – Режим доступу до ресурсу: https://e-schools.info/
3. Studlife [Електронний ресурс] – Режим доступу до ресурсу: https://studfile.net/preview/9895845/
4. Pdau [Електронний ресурс] – Режим доступу до ресурсу: https://www.pdau.edu.ua/sites/default/files/node/11342/sylabusisto2021rnnormatyvnyy.pdf
5. E-School [Електронний ресурс] – Режим доступу до ресурсу: https://e-school.in.ua/
6. Robot Scholl [Електронний ресурс] – Режим доступу до ресурсу: https://robotschool.com.ua/
7. Sumy IT School [Електронний ресурс] – Режим доступу до ресурсу: https://itschool.org.ua/
8. Developer mozilla JS - [Електронний ресурс] – Режим доступу до ресурсу: https://developer.mozilla.org/ru/docs/Learn/JavaScript
9. Developer mozilla CSS - [Електронний ресурс] – Режим доступу до ресурсу: https://developer.mozilla.org/ru/docs/Learn/CSS
10. Abelson H., Sussman G.J., Henz M., Wrigstad T., Sussman J. Structure and Interpretation of Computer Programs: JavaScript Edition. - The MIT Press, 2022. - 640 p.
11. Sethi R. Software Engineering: Basic Principles and Best Practices . - Cambridge: Cambridge University Press, 2023. - 361 p.
12. Montagne Euripides. Systems Software: Essential Concepts. - Cognella Academic Publishing, 2022. — 228 p.

13. PHP [Електронний ресурс] – Режим доступу до ресурсу: - https://www.php.net/

14. Tsui F., Karam O., Bernal B. Fundamentals Of Software Engineering. - 5th Edition. — Jones & Bartlett Publishers, 2022. — 450 p.

15. MySQL - [Електронний ресурс] – Режим доступу до ресурсу: https://www.mysql.com/

16. Adeolu O. Essential Guide to PHP for All Levels.- Young M. Computer Guru Institute, 2024. — 171.

17. Структура БД E-School [Електронний ресурс] – Режим доступу до ресурсу: https://drive.google.com/file/d/1OTO3JpdnAGG_Zj6ALoeBCudhnt7qm_sb/view?usp=sharing

ДОДАТОК А

Файл - groups.js

```
if (document.querySelector(".teacherpage__groups")) {
  const filters = document.querySelectorAll(".filter__item");
  const teacherId = getTeacherId();

  handleFilterClick(filters[2], 2, teacherId);

  filters.forEach((filter, index) => {
    filter.addEventListener("click", (e) => {
      e.preventDefault();
      handleFilterClick(filter, index, teacherId);
    });
  });
}
function getTeacherId() {
  const url = new URL(window.location.href);
  const params = new URLSearchParams(url.search);
  return params.get("teacher_id");
}
function removeActivity(filters) {
  filters.forEach((filter) => {
    filter.classList.remove("_activity");
  });
}
function handleFilterClick(filter, index, teacherId) {
  const filters = document.querySelectorAll(".filter__item");
  const groupBody = document.querySelector(".teacherpage__groups");
  groupBody.innerHTML = "";
  removeActivity(filters);
  filter.classList.add("_activity");

  let requestBody;

  switch (index) {
    case 0:
```

```
      requestBody = { action: "group_name" };
      break;
    case 1:
      requestBody = { action: "group_number" };
      break;
    case 2:
      requestBody = { action: "group_teacher", id_teacher: teacherId };
      break;
    default:
      return;
  }

  fetch("../php/Group.php", {
    method: "POST",
    headers: {
      "Content-Type": "application/json",
    },
    body: JSON.stringify(requestBody),
  })
    .then((response) => {
      if (!response.ok) {
        throw new Error("Network response was not ok");
      }
      return response.json();
    })
    .then((data) => {
      data.forEach((group) => {
        groupBody.innerHTML += `
          <div class="groups__group">
            <div class="group__body">
              <div class="group__image">
                <img src="../img/courses/${group.course_image}" alt="" />
              </div>
              <div class="group__info">
                <div class="group__title">Група <span>$
{group.group_name}</span></div>
                <div class="group__subtitle">${group.course_name}</div>
```

```
                <div class="group__teacher">${group.teacher_lastname} $
{group.teacher_firstname}</div>
              </div>
            </div>
            <a
              href="./marking.php?id_group=${group.id_group}"
              class="button group__button"
              >Відмітити</a>
          </div>
        `;
      });
    })
    .catch((error) => {
      console.error("Error:", error);
    });
}
```

Файл - map.js

```
var map = L.map("map").setView([50.8926693, 34.8400579], 16);

L.tileLayer("https://tile.openstreetmap.org/{z}/{x}/{y}.png", {
  attribution:
    '&copy; <a
href="https://www.openstreetmap.org/copyright">OpenStreetMap</a>
contributors',
}).addTo(map);

L.marker([50.8923352, 34.8419994])
  .addTo(map)
  .bindPopup("Вам сюди)")
  .openPopup();

L.marker([50.892993,
34.8374151]).addTo(map).bindPopup("Зупинка").openPopup();

L.Routing.control({
  waypoints: [
    L.latLng(50.8923352, 34.8419994),
    L.latLng(50.892993, 34.8374151),
```

```
  ],
  // draggableWaypoints: false,
  show: false,
}).addTo(map);
```

Файл - marking.js

```
if (document.querySelector(".main__marking")) {
  const spanDateInput = document.querySelector(".marking__date");
  let dateLesson = getCurrentDateTime();
  const form = document.querySelector(".marking__form");
  const formBody = document.querySelector(".form__marking-body");
  const groupId = getGroupId();
  const buttonLessonBack = document.querySelector(".button__back");
  const buttonLessonNext = document.querySelector(".button__next");
  const data = {};
  spanDateInput.textContent = dateLesson;

  const isNoteToday = form.getAttribute("is_note");

  requestToGroup(
    {
      action: "getStudent",
      status: isNoteToday !== "1" ? "preload" : "",
      id_group: groupId,
      date_lesson: dateLesson,
    },
    formBody
  );

  buttonLessonBack.addEventListener("click", (e) => {
    e.preventDefault();
    handleLessonNavigation("lessonBack", getCurrentDateTime());
  });
  buttonLessonNext.addEventListener("click", (e) => {
    e.preventDefault();

    handleLessonNavigation("lessonNext", getCurrentDateTime());
  });
```

```
  form.addEventListener("submit", (e) => {
    e.preventDefault();

    const lessonNumber = parseInt(
      document.querySelector(".marking__number").textContent
    );
    dateLesson = spanDateInput.textContent;
    attendance(formBody, lessonNumber, groupId, dateLesson, isNoteToday);
  });
  function handleLessonNavigation(action, currentDate) {
    if (form.getAttribute("id_lesson")) {
      const lessonNumberFunc =
        document.querySelector(".marking__number").textContent;
      requestToGroup(
        {
          action: action,
          id_group: groupId,
          lesson_number: lessonNumberFunc,
          current_date: currentDate,
        },
        formBody
      );
    }
  }
}
function refreshButtons() {
  if (document.querySelector(".present__button")) {
    const buttonsPresent = document.querySelectorAll(".present__button");
    const checkboxes = document.querySelectorAll(".marking__item input");

    buttonsPresent.forEach((button, index) => {
      if (checkboxes[index].checked) {
        checkboxes[index].checked
          ? (button.textContent = "Відсутній")
          : (button.textContent = "Присутній");
      }
    });
```

```
    buttonsPresent.forEach((button, index) => {
      button.addEventListener("click", () => {
        const checkbox = checkboxes[index];
        checkbox.checked = !checkbox.checked;

        checkbox.checked
          ? (button.textContent = "Відсутній")
          : (button.textContent = "Присутній");
      });
    });
  }
}
document.addEventListener("DOMContentLoaded", function () {
  flatpickr("#datepicker", {
    enableTime: true,
    dateFormat: "d-m-Y H:i",
  });
});
function getCurrentDateTime() {
  let currentDate = new Date();
  let year = currentDate.getFullYear();
  let month = String(currentDate.getMonth() + 1).padStart(2, "0");
  let day = String(currentDate.getDate()).padStart(2, "0");

  return `${year}-${month}-${day}`;
}
function getGroupId() {
  const url = new URL(window.location.href);
  const params = new URLSearchParams(url.search);
  return params.get("id_group");
}
function attendance(formBody, lessonNumber, groupId, dateLesson, status)
{
  const data = {
    action: status !== "1" ? "attendance" : "attendance_again",
    id_group: groupId,
    lesson_date: dateLesson,
    lesson_number: lessonNumber,
```

```
    attendance: getStudentStatus(),
  };
  console.log("attendance", data);
  requestToGroup(data, formBody);
}
function requestToGroup(data, form) {
  const lessonNumberInput = document.querySelector(".marking__number");

  fetch("../php/Group.php", {
    method: "POST",
    headers: {
      "Content-Type": "application/json",
    },
    body: JSON.stringify(data),
  })
    .then((response) => {
      if (!response.ok) {
        throw new Error("Network response was not ok");
      }
      return response.json();
    })
    .then((data) => {
      // console.log("data", data);

      const spanDateInput = document.querySelector(".marking__date");

      if (data.lesson_date) {
        spanDateInput.textContent = data.lesson_date;
        lessonNumberInput.textContent = data.lesson_number;
      } else if (data.last_lesson_number) {
        lessonNumberInput.textContent = data.last_lesson_number;
      } else if (
        spanDateInput.textContent !== getCurrentDateTime() &&
        data.lesson_date
      ) {
        lessonNumberInput.textContent = data.lesson_number + 1;
      } else {
        lessonNumberInput.textContent = data.lesson_number;
```

```
      }

      const students = data.students ? data.students : null;

      if (students) {
        form.innerHTML = "";
        students.forEach((student) => {
          form.innerHTML += `
          <div class="marking__item">
          <label for="marking_list">${student.student_firstname} ${
            student.student_lastname
          }</label>
          <input
            type="checkbox" name="marking_list" class="marking__input"
            ${student.status_student === "Присутній" ? "checked" : ""}
            id_student="${student.id_student}"
          />
          <div class="item__desc">
            <label class="button present__button"> Присутній</label>
          </div>
        </div>
          `;
        });
        refreshButtons();

        const formTag = document.querySelector(".marking__form");
        formTag.setAttribute("id_lesson", data.id_lesson);
      }
    })
    .catch((error) => {
      console.error("Error:", error);
    });
}
function getStudentStatus() {
  const inputs = document.querySelectorAll(".marking__input");
  const studentsAttendance = [];

  inputs.forEach((element) => {
```

```
    studentsAttendance.push({
      id_student: element.getAttribute("id_student"),
      status: element.checked ? "Присутній" : "Відсутній",
    });
  });
  return studentsAttendance;
}
```

Файл - popup.js

```
// МК

if (document.querySelector(".popup")) {
  const popup = document.querySelector(".popup");
  const openButtons = document.querySelectorAll(".master_class");
  const popupCloses = [
    document.querySelector(".popup__fild"),
    document.querySelector(".popup__close"),
  ];

  const buttonsPopup = [...openButtons, ...popupCloses];

  buttonsPopup.forEach((button) => {
    button.addEventListener("click", (e) => {
      e.preventDefault();
      errorClear();
      popupVisible(popup);
    });
  });
}
if (document.querySelectorAll(".popup__form input").length > 0) {
  const form = document.querySelector(".popup__form");

  form.addEventListener("submit", (e) => {
    e.preventDefault();

    const formError =
document.querySelector(".popup__form .form__error");
```

```
    const inputs = document.querySelectorAll(".popup__form input");
    const nameParrent =
document.querySelector("input[name='name_parrent']");
    const phoneParrent =
document.querySelector("input[name='phone_parrent']");
    const nameChild = document.querySelector("input[name='name_child']");
    const ageChild = document.querySelector("input[name='age_child']");
    let selectedCourses = Array.from(
      document.querySelectorAll('input[name="courses[]"]:checked')
    ).map((cb) => cb.value);

    inputs.forEach((e) => {
      e.classList.remove("_error");
    });
    formError.classList.remove("active");
    formError.textContent = "";

    if (!validationName(nameParrent.value)) {
      nameParrent.classList.add("_error");
      errorMessage(formError, "В вашому імені не допустимі символи!");
      return;
    }
    if (!validationPhone(phoneParrent.value)) {
      phoneParrent.classList.add("_error");
      errorMessage(formError, "Введіть коректний номер вашого
телефону!");
      return;
    }
    if (!validationName(nameChild.value)) {
      nameParrent.classList.add("_error");
      errorMessage(formError, "В імені дитини не допустимі символи!");
      return;
    }
    if (selectedCourses.length === 0) {
      errorMessage(formError, "Обірть хоча б один курс!");
      return;
    }
```

```
const data = {
  parrent_firstname: nameParrent.value.trim().split(" ")[0],
  parrent_lastname:
    nameParrent.value.trim().split(" ").length === 2
      ? nameParrent.value.trim().split(" ")[1]
      : "",
  parrent_phone: phoneParrent.value.trim(),
  child_firstname: nameChild.value.trim().split(" ")[0],
  child_lastname:
    nameChild.value.trim().split(" ").length === 2
      ? nameChild.value.trim().split(" ")[1]
      : "",
  child_age: ageChild.value,
  selected_courses: selectedCourses,
};

fetch("../php/MasterClass.php", {
  method: "POST",
  headers: {
    "Content-Type": "application/json",
  },
  body: JSON.stringify(data),
})
  .then((response) => {
    if (!response.ok) {
      throw new Error("Network response was not ok");
    }
    return response.json();
  })
  .then((data) => {
    errorMessage(formError, data.message);
    if (data.status) {
      formMKClear();
    }
  })
  .catch((error) => {
    console.error("Error:", error);
  });
```

```
  });
}

// Authorization

if (
  document.querySelector(".popup__auth") &&
  document.querySelector(".header__button")
) {
  const popup = document.querySelector(".popup__auth");
  const openButton = document.querySelector(".header__button");
  const popupCloses = [
    document.querySelector(".auth__fild"),
    document.querySelector(".auth__close"),
  ];

  const buttonsPopup = [openButton, ...popupCloses];

  buttonsPopup.forEach((button) => {
    button.addEventListener("click", (e) => {
      e.preventDefault();
      popupVisible(popup);
    });
  });
}
if (document.querySelector(".form__auth")) {
  const form = document.querySelector(".form__auth");
  const login = document.querySelector("input[name='login']");
  const password = document.querySelector("input[name='password']");

  login.value = "";
  password.value = "";

  form.addEventListener("submit", (event) => {
    event.preventDefault();

    const chcekbox = document.querySelector("#teacher");
```

```
    const data = {
      login: login.value,
      password: password.value,
    };

    if (chcekbox.checked) {
      fetch("../php/Teacher.php", {
        method: "POST",
        headers: {
          "Content-Type": "application/json",
        },
        body: JSON.stringify(data),
      })
        .then((response) => {
          if (!response.ok) {
            throw new Error("Network response was not ok");
          }
          return response.json();
        })
        .then((data) => {
          if (data.status) {
            window.location.href = "../personal_office/teacher.php";
          }
        })
        .catch((error) => {
          console.error("Error:", error);
        });
    }
  });
}

// Workingout

if (document.querySelector(".popup__workingout")) {
  const popup = document.querySelector(".popup__workingout");
  const openButtons = document.querySelectorAll(".button__item");
  const popupCloses = [
    document.querySelector(".workingout__fild"),
```

```
    document.querySelector(".workingout__close"),
  ];

  openButtons.forEach((button) => {
    button.addEventListener("click", () => {
      if (!button.classList.contains("_disable")) {
        popupVisible(popup);
      }
    });
  });

  popupCloses.forEach((e) => {
    e.addEventListener("click", () => {
      popupVisible(popup);
    });
  });
}

// Funcation

function popupVisible(popup) {
  document.body.classList.toggle("menu__active");
  popup.classList.toggle("_open");
}
function errorMessage(form, message) {
  form.classList.add("active");
  form.textContent = message;
}
function errorClear() {
  const formError = document.querySelector(".popup__form .form__error");

  formError.classList.remove("active");
  formError.textContent = "";
}
function validationPhone(phone) {
  return /^(\d{10}|\+\d{12})$/.test(phone);
}
function validationName(name) {
```

```
  return /^[a-яА-ЯiIїЇ\s]+$/.test(name);
}
function formMKClear() {
  const inputs = document.querySelectorAll(".form__input");
  const checkboxs = document.querySelectorAll('input[name="courses[]"]');

  inputs.forEach((input) => {
    input.value = "";
  });
  checkboxs.forEach((checkbox) => {
    checkbox.checked = false;
  });
}
```

Файл - script.js

```
if (document.querySelector(".button__menu")) {
  const buttonMenu = document.querySelector(".button__menu");

  buttonMenu.addEventListener("click", () => {
    openOrClseMenu(buttonMenu);
  });

  function openOrClseMenu(buttonMenu) {
    const menu = document.querySelector(".header__nav");
    menu.classList.toggle("active");
    buttonMenu.classList.toggle("button__menu-actie");
    document.body.classList.toggle("menu__active");
  }
}
if (document.querySelector(".link__totop")) {
  const BUTTON_VISIBLE_PIXEL = 300;

  const buttonToTop = document.querySelector(".link__totop");

  addEventListener("scroll", () => {
    if (BUTTON_VISIBLE_PIXEL < window.scrollY) {
      buttonToTop.classList.add("_visible");
    } else {
```

```
      buttonToTop.classList.remove("_visible");
    }
  });

  buttonToTop.addEventListener("click", () => {
    const html = document.querySelector("html");
    html.scrollTo({ top: 0, behavior: "smooth" });
  });
}
if (document.querySelector(".footer__form")) {
  const form = document.querySelector(".footer__form");
  const formErroe = document.querySelector(".form__error");

  form.addEventListener("submit", (e) => {
    e.preventDefault();

    formErroe.classList.remove("active");
    formErroe.textContent = "";

    const phoneOrEmail = form
      .querySelector("input[name='phone_or_email']")
      .value.trim();
    const question = form
      .querySelector("textarea[name='question']")
      .value.trim();
    const emailRegex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
    const phoneRegex = /^(\d{10}|\+\d{12})$/;

    let errors = [];

    if (!emailRegex.test(phoneOrEmail) && !phoneRegex.test(phoneOrEmail))
{
      errors.push("Введіть дійсну електронну адресу або номер
телефону.");
    }

    if (!question.length) {
      errors.push("Будь ласка, введіть своє запитання.");
```

```
    }

    if (errors.length > 0) {
      formErroe.classList.add("active");
      formErroe.textContent = errors.join("\n");
    }
  });
}
if (document.querySelector(".preloader")) {
  const preloader = document.querySelector(".preloader");

  window.addEventListener("load", () => {
    preloader.classList.add("hidden");
  });
}
if (document.querySelector(".parrent__childs")) {
  const blocks = document.querySelectorAll(".parrent__childs");

  for (let index = 0; index < blocks.length - 1; index++) {
    blocks[index].classList.add("_more");
  }
}
```

Файл - slider.js

```
const swiper = new Swiper(".swiper", {
  loop: true,

  pagination: {
    el: ".swiper-pagination",
  },
  autoplay: {
    delay: 5000,
  },
});
```

ДОДАТОК Б

Файл – Contact.php

```
<?php
  require_once "DB.php";

  class Contact {
    private $mysql;

    public function __construct(){
      $db = new DB();
      $this->mysql = $db->get_connection();
    }

    public function get_phone(){
      $result = $this->mysql->query("SELECT * FROM `contacts` WHERE
`contact_type` = 'phone'");
      return $result->fetch_assoc();
    }
    public function get_location(){
      $result = $this->mysql->query("SELECT * FROM `contacts` WHERE
`contact_type` = 'location'");
      return $result->fetch_assoc();
    }
    public function get_email(){
      $result = $this->mysql->query("SELECT * FROM `contacts` WHERE
`contact_type` = 'email'");
      return $result->fetch_assoc();
    }
    public function get_facebook(){
      $result = $this->mysql->query("SELECT * FROM `contacts` WHERE
`contact_type` = 'facebook'");
      return $result->fetch_assoc();
    }
    public function get_instagram(){
      $result = $this->mysql->query("SELECT * FROM `contacts` WHERE
`contact_type` = 'instagram'");
```

```
      return $result->fetch_assoc();
    }
    public function get_photo_team(){
      $result = $this->mysql->query("SELECT * FROM `contacts` WHERE
`contact_type` = 'image_team'");
      return $result->fetch_assoc();
    }
  }
```

Файл - Course.php

```
<?php
  require_once "DB.php";

  class Course {
    private $mysql;

    public function __construct(){
      $db = new DB();
      $this->mysql = $db->get_connection();
    }
    public function get_all_courses(){
      $result = $this->mysql->query("SELECT * FROM `courses`");

      if(!$result){
        return array("message" => "Помилка запиту або таблиця 'courses'
пуста!");
      }
      $courses = array();

      while($row = $result->fetch_assoc()){
        array_push($courses, $row);
      }

      return $courses;
    }
    public function get_number_courses($number){
      $result = $this->mysql->query("SELECT * FROM `courses` ORDER BY
RAND() LIMIT " . $number);
```

```
      if(!$result){
          return array("message" => "Помилка запиту або таблиця 'courses'
пуста!");
      }
      $courses = array();

      while($row = $result->fetch_assoc()){
          array_push($courses, $row);
      }

      return $courses;
    }
    public function get_course_for_id($id_course){
      $id_course = intval($id_course);// Типу захист від SQL-інєкції :)

      $result = $this->mysql->query("SELECT * FROM `courses` WHERE
`id_course` = '$id_course'");

      if(!$result){
        return array("message" => "Помилка запиту або таблиця 'courses'
пуста!");
      }
      if($result->num_rows == 0){
        return array("message" => "Курс з таким ID не знайдено!");
      }

      return $result->fetch_assoc();
    }
    public function get_course_sections_for_id($id_course){
      $id_course = intval($id_course);// Типу захист від SQL-інєкції :)

      $result = $this->mysql->query("SELECT * FROM `course_sections`
WHERE `course_id` = '$id_course'");

      if(!$result){
        return array("message" => "Помилка запиту або таблиця
'course_sections' пуста!");
      }
```

```
      if($result->num_rows == 0){
        return array("message" => "Курс з таким ID не знайдено!");
      }

      $sections = array();

      while($row = $result->fetch_assoc()){
        array_push($sections, $row);
      }

      return $sections;
    }
    public function get_course_features_for_id($id_course){
      $id_course = intval($id_course);// Типу захист від SQL-інєкції :)

      $result = $this->mysql->query("SELECT * FROM `course_features`
WHERE `course_id` = '$id_course'");

      if(!$result){
        return array("message" => "Помилка запиту або таблиця
'course_features' пуста!");
      }
      if($result->num_rows == 0){
        return array("message" => "Курс з таким ID не знайдено!");
      }

      $features = array();

      while($row = $result->fetch_assoc()){
        array_push($features, $row);
      }

      return $features;
    }
  }
```

Файл – DB.php

```
<?php
```

```
class DB{
  private $connection;

  public function __construct(){
    $this->connection = new mysqli("localhost", "root", "", "E-School");

    if (!$this->connection)
    die('Database connecting error!');

    $this->connection->query("SET NAMES 'utf8'");
  }

  public function get_connection(){
    return $this->connection;
  }
}
```

Файл – Group.php

```
<?php
  require_once "DB.php";

  class Group {
    private $mysql;

    public function __construct(){
      $db = new DB();
      $this->mysql = $db->get_connection();
    }
    public function get_all_groups_sort_number(){
      $result = $this->mysql->query("SELECT g.`id_group`, g.`group_name`,
c.`course_name`, c.`course_image`, t.`teacher_firstname`,
                                    t.`teacher_lastname` FROM `groups`
g
                                      JOIN `courses` c ON g.`id_course`
= c.`id_course`
                                      JOIN `teachers` t ON
t.`id_teacher` = g.`id_teacher`
                                    ORDER BY g.`group_name` ASC;
                                  ");
```

```
      if(!$result){
        return array("message" => "Таблиця 'groups' пуста!");
      }
      $groups = array();

      while($row = $result->fetch_assoc()){
        array_push($groups, $row);
      }

      return $groups;
    }
    public function get_all_groups_sort_name(){
      $result = $this->mysql->query("SELECT g.`id_group`, g.`group_name`,
c.`course_name`, c.`course_image`, t.`teacher_firstname`,
                                     t.`teacher_lastname` FROM `groups`
g
                                       JOIN `courses` c ON g.`id_course`
= c.`id_course`
                                       JOIN `teachers` t ON
t.`id_teacher` = g.`id_teacher`
                                     ORDER BY c.`course_name` ASC;
                                    ");

      if(!$result){
        return array("message" => "Таблиця 'groups' пуста!");
      }
      $groups = array();

      while($row = $result->fetch_assoc()){
        array_push($groups, $row);
      }

      return $groups;
    }
    public function get_group_for_teacher($id_teacher){
      $result = $this->mysql->query("SELECT g.`id_group`, g.`group_name`,
c.`course_name`, c.`course_image`, t.`teacher_firstname`,
```

```
                                        t.`teacher_lastname` FROM `groups`
g
                                          JOIN `courses` c ON g.`id_course`
= c.`id_course`
                                          JOIN `teachers` t ON
t.`id_teacher` = g.`id_teacher`
                                          WHERE t.`id_teacher` =
'$id_teacher'
                                    ");
       if(!$result){
         return array("message" => "Таблиця 'groups' пуста!");
       }
       $groups = array();

       while($row = $result->fetch_assoc()){
         array_push($groups, $row);
       }

       return $groups;
     }
     public function get_group_name($id_group){
       $stmt = $this->mysql->prepare("SELECT `group_name` FROM `groups`
WHERE `id_group` = ?");
       $stmt->bind_param('i', $id_group);
       $stmt->execute();
       $result = $stmt->get_result();
       $stmt->close();

       return $result->fetch_assoc()['group_name'];
     }
     public function get_students_of_group($id_group){
       $stmt = $this->mysql->prepare("SELECT s.`id_student`,
s.`student_firstname`, s.`student_lastname` FROM `groups` g
                                        JOIN `group_list` gl ON
g.`id_group` = gl.`id_group`
                                        JOIN `students` s ON s.`id_student`
= gl.`id_student`
                                    WHERE g.`id_group` = ?");
```

```
      $stmt->bind_param('i', $id_group);
      $stmt->execute();
      $result = $stmt->get_result();
      $stmt->close();

      $students = array();

      while($row = $result->fetch_assoc()){
        array_push($students, $row);
      }

      return $students;
    }
    public function get_students_by_group_lesson_date($id_group,
$id_lesson, $lesson_date){
      $stmt = $this->mysql->prepare("SELECT s.`id_student`,
s.`student_firstname`, s.`student_lastname`, a.`status_student`
                                    FROM `students` s
                                      JOIN `group_list` gl ON
s.`id_student` = gl.`id_student`
                                      JOIN `groups` g ON gl.`id_group`
= g.`id_group`
                                      LEFT JOIN `lessons` l ON
g.`id_group` = l.`id_group` AND l.`lesson_date` = ?
                                      LEFT JOIN `attendance` a ON
s.`id_student` = a.`id_student` AND l.`id_lesson` = a.`id_lesson`
                                    WHERE g.`id_group` = ? AND
l.`id_lesson` = ?;"
                                  );
      $stmt->bind_param('sii', $lesson_date, $id_group, $id_lesson);
      $stmt->execute();
      $result = $stmt->get_result();
      $stmt->close();

      $students = array();

      while($row = $result->fetch_assoc()){
        array_push($students, $row);
```

```
      }

      return $students;
    }
    public function get_last_number_lesson($id_group){
      $stmt = $this->mysql->prepare('SELECT `lesson_number` FROM
`lessons`
                                     WHERE `id_group` = ? ORDER BY
`lesson_number` DESC LIMIT 1;
                                    ');
      $stmt->bind_param('i', $id_group);
      $stmt->execute();
      $result = $stmt->get_result();

      $stmt->close();

      return $result->fetch_assoc()["lesson_number"] + 1;
    }
    public function create_lesson($date, $number, $id_group){
      $stmt = $this->mysql->prepare("INSERT INTO `lessons`
                                      (`id_group`, `lesson_date`,
`lesson_number`)
                                      VALUES (?, ?, ?)
                                    ");
      $stmt->bind_param("isi", $id_group, $date, $number);
      $stmt->execute();
      $id_lesson = $this->mysql->insert_id;

      $stmt->close();

      return $id_lesson;
    }
    public function attendance_create($id_lesson, $id_student,
$status_student){
      $stmt = $this->mysql->prepare("INSERT INTO `attendance`
(`id_lesson`, `id_student`, `status_student`) VALUES (?, ?, ?)");
      $stmt->bind_param("iis", $id_lesson, $id_student, $status_student);
      $stmt->execute();
```

```
      $stmt->close();
    }
    public function attendance_update($id_lesson, $id_student,
$status_student){
      $stmt = $this->mysql->prepare("UPDATE `attendance` SET
`status_student` = ? WHERE `id_lesson` = ? AND `id_student` = ?");
      $stmt->bind_param("sii", $status_student, $id_lesson, $id_student);
      $stmt->execute();
      $stmt->close();
    }
    public function check_last_lesson($id_group){
      $stmt = $this->mysql->prepare("SELECT `lesson_date`, `id_lesson`,
`lesson_number` FROM `lessons` WHERE `id_group` = ? ORDER BY
`lesson_date` DESC LIMIT 1");
      $stmt->bind_param('i', $id_group);
      $stmt->execute();
      $result = $stmt->get_result();

      $stmt->close();

      return $result->fetch_assoc();
    }
    public function get_lesson_number_by_date($id_group, $lesson_date){
      $stmt = $this->mysql->prepare("SELECT `id_lesson`, `lesson_number`
FROM `lessons` WHERE `id_group` = ? AND `lesson_date` = ?;");
      $stmt->bind_param('is', $id_group, $lesson_date);
      $stmt->execute();
      $result = $stmt->get_result();
      $stmt->close();

      return $result->fetch_assoc();
    }
    public function get_id_lesson_prev($id_group, $lessen_number){
      $stmt = $this->mysql->prepare("SELECT * FROM `lessons` WHERE
`id_group` = ? AND `lesson_number` < ? ORDER BY `lesson_number` DESC
LIMIT 1;");

      $stmt->bind_param('ii', $id_group, $lessen_number);
```

```
      $stmt->execute();
      $result = $stmt->get_result();
      $stmt->close();

      return $result->fetch_assoc();
    }
    public function get_id_lesson_next($id_group, $lessen_number){
      $stmt = $this->mysql->prepare("SELECT * FROM `lessons` WHERE
`id_group` = ? AND `lesson_number` > ? ORDER BY `lesson_number` ASC LIMIT
1;");

      $stmt->bind_param('ii', $id_group, $lessen_number);
      $stmt->execute();
      $result = $stmt->get_result();
      $stmt->close();

      return $result->fetch_assoc();
    }
    public function get_attendance_for_group_date($id_group,
$lesson_number){
      $stmt = $this->mysql->prepare("SELECT a.`status_student`,
l.`lesson_date`, l.`lesson_number`, s.`student_firstname`,
s.`student_lastname`
                                     FROM `attendance` a
                                     JOIN `students` s ON a.`id_student`
= s.`id_student`
                                     JOIN `lessons` l ON l.`id_lesson` =
a.`id_lesson`
                                     WHERE l.`id_group` = ? AND
l.`lesson_number` = ?;
                                   ");
      $stmt->bind_param("ii", $id_group, $lesson_number);
      $stmt->execute();
      $result = $stmt->get_result();
      $stmt->close();

      $students = [];
```

```
      while($row = $result->fetch_assoc()){
        array_push($students, $row);
      }

      return $students;
    }
  }

  if($_SERVER['REQUEST_METHOD'] === 'POST'){
    $data = json_decode(file_get_contents('php://input'), true);

    $Group = new Group();
    $result = null;

    switch ($data['action']) {
      case 'group_teacher':
        $result = $Group->get_group_for_teacher($data['id_teacher']);
        break;
      case 'group_number':
        $result = $Group->get_all_groups_sort_number();
        break;
      case 'group_name':
        $result = $Group->get_all_groups_sort_name();
        break;
      case 'getStudent':
        $id_group = $data['id_group'];
        $date_lesson = $data['date_lesson'];

        if($data["status"] === 'preload'){
          $students = $Group->get_students_of_group($id_group);
          $lesson_info = $Group->check_last_lesson($id_group);
          $id_lesson = $lesson_info['id_lesson'];

          $result = array('lesson_number' =>
$lesson_info["lesson_number"] + 1, "id_lesson" => $id_lesson,
'students'=> $students);
        }else{
```

```
          $lesson_info = $Group->get_lesson_number_by_date($id_group,
$date_lesson);
          $id_lesson = $lesson_info['id_lesson'];

          $students = $Group-
>get_students_by_group_lesson_date($id_group, $id_lesson, $date_lesson);

          $result = array('lesson_number' =>
$lesson_info["lesson_number"], "id_lesson" => $id_lesson, 'students'=>
$students);
        }
        break;
      case 'attendance':
        $id_group = $data["id_group"];
        $date_lesson = $data["lesson_date"];
        $attendance = $data["attendance"];

        $last_lesson_number = $Group->get_last_number_lesson($id_group);
        $id_lesson_created = $Group->create_lesson($date_lesson,
$last_lesson_number, $id_group);
        $lesson_info = $Group->get_lesson_number_by_date($id_group,
$date_lesson);

        foreach($attendance as $student_info){
          $Group->attendance_create($id_lesson_created,
$student_info["id_student"], $student_info["status"]);
        }
        $students = $Group->get_students_of_group($id_group);

        $result =  array('lesson_number' =>
$lesson_info["lesson_number"], 'id_lesson'=> $id_lesson_created,
"message" => "Учні відмічені!");
        break;
      case 'attendance_again':
        $id_group = $data["id_group"];
        $date_lesson = $data["lesson_date"];
        $attendance = $data["attendance"];
```

```
        $lesson_info = $Group->get_lesson_number_by_date($id_group,
$date_lesson);
        $id_lesson = $lesson_info['id_lesson'];

          foreach($attendance as $student_info){
            $Group->attendance_update($id_lesson,
$student_info["id_student"], $student_info["status"]);
          }
        $result = array('lesson_number' => $lesson_info["lesson_number"],
'id_lesson'=> $id_lesson, "message" => "Дані учнів оновлені!");
        break;
      case 'lessonBack':
        $id_group = $data["id_group"];
        $lesson_number = $data["lesson_number"];

        $other_lesson = $Group->get_id_lesson_prev($id_group,
$lesson_number);

        if(isset($other_lesson)){
          $id_lesson = $other_lesson['id_lesson'];
          $date_lesson = $other_lesson['lesson_date'];

          $students = $Group-
>get_students_by_group_lesson_date($id_group, $id_lesson, $date_lesson);

          $result = array('lesson_number' =>
$other_lesson["lesson_number"], 'id_lesson' => $id_lesson, 'lesson_date'
=> $date_lesson, 'students'=> $students);
        }else{
          $result = array("status"=> "Не знайдено лекції",
"last_lesson_number"=> $lesson_number);
        }
        break;
      case 'lessonNext':
        $id_group = $data["id_group"];
        $lesson_number = $data["lesson_number"];
        $current_date = $data['current_date'];
```

```
        $last_lesson = $Group->check_last_lesson($id_group);
        $next_lesson = $Group->get_id_lesson_next($id_group,
$lesson_number);

        if($last_lesson['lesson_date'] < $current_date && !
isset($next_lesson)){
          $date_lesson = $current_date;

          $lesson_info = $Group->check_last_lesson($id_group);
          $students = $Group->get_students_of_group($id_group);

          $result = array('lesson_number' =>
$lesson_info["lesson_number"] + 1, 'id_lesson' => null, 'lesson_date' =>
$date_lesson, 'students'=> $students);
        }else if(isset($next_lesson)){
          $id_lesson = $next_lesson['id_lesson'];
          $date_lesson = $next_lesson['lesson_date'];

          $students = $Group-
>get_students_by_group_lesson_date($id_group, $id_lesson, $date_lesson);

          $result = array('lesson_number' =>
$next_lesson["lesson_number"], 'id_lesson' => $id_lesson, 'lesson_date'
=> $date_lesson, 'students'=> $students);
        }else{
          $result = array("status"=> "Не знайдено лекції",
"last_lesson_number"=> $lesson_number);
        }
        break;
      default:
        $result = ['error' => 'Invalid action'];
        break;
    }
    echo json_encode($result);
  }
```

Файл - MasterClass.php

```
<?php
  require_once "DB.php";
```

```
  class MasterClass {
    private $mysql;

    public function __construct(){
      $db = new DB();
      $this->mysql = $db->get_connection();
    }

    public function registraion($parrent_firstname, $parrent_lastname,
$parrent_phone, $child_firstname, $child_lastname, $child_age,
$selected_courses){
      $stmt = $this->mysql->prepare("SELECT COUNT(*) as count FROM
`registrations` WHERE `parrent_firstname` = ? AND `parrent_lastname` = ?
AND `parrent_phone` = ? AND `child_firstname` = ? AND `child_lastname`
= ? AND `child_age` = ?");
      $stmt->bind_param("ssssi", $parrent_firstname, $parrent_lastname,
$parrent_phone, $child_firstname, $child_lastname, $child_age);
      $stmt->execute();
      $stmt->bind_result($count);
      $stmt->fetch();
      $stmt->close();

      if ($count > 0) {
          return array("message" => "Реєстрація з такими даними вже
існує!", "status" => false);
      }

      $stmt = $this->mysql->prepare("INSERT INTO `registrations`
(`parrent_firstname`, `parrent_lastname`, `parrent_phone`,
`child_firstname`, `child_lastname`, `child_age`) VALUES
(?, ?, ?, ?, ?, ?)");
      $stmt->bind_param("ssssi", $parrent_firstname, $parrent_lastname,
$parrent_phone, $child_firstname, $child_lastname, $child_age);
      $stmt->execute();
      $last_registration_id = $this->mysql->insert_id;
      $stmt->close();
```

```
      $stmt = $this->mysql->prepare("INSERT INTO `registration_courses`
(`registration_id`, `course_id`) VALUES (?, ?)");

      foreach ($selected_courses as $course_id) {
        $stmt->bind_param("ii", $last_registration_id, $course_id);
        $stmt->execute();
      }

      $stmt->close();
      $this->mysql->close();

      return array("message" => "Ви усіпішно зареєстровані на майстер-
клас!", "status" => true);
    }

}

if ($_SERVER['REQUEST_METHOD'] === 'POST') {
  $data = json_decode(file_get_contents('php://input'), true);

  $parrent_firstname = $data["parrent_firstname"];
  $parrent_lastname = isset($data["parrent_lastname"]) ?
$data["parrent_lastname"] : null;
  $parrent_phone = $data["parrent_phone"];
  $child_firstname = $data["child_firstname"];
  $child_lastname = isset($data["child_lastname"]) ?
$data["child_lastname"] : null;
  $child_age = intval($data["child_age"]);
  $selected_courses = $data["selected_courses"];

  $registration  = new MasterClass();
  $result = $registration->registraion($parrent_firstname,
$parrent_lastname, $parrent_phone, $child_firstname, $child_lastname,
$child_age, $selected_courses);

  echo json_encode($result);
}
```

Файл - Teacher.php

```
<?php
  require_once "DB.php";
  session_start();

  class Teacher {
    private $mysql;

    public function __construct(){
      $db = new DB();
      $this->mysql = $db->get_connection();
    }
    public function get_all_teacher(){
      $result = $this->mysql->query("SELECT `teacher_image`,
`teacher_firstname`, `teacher_lastname` FROM `teachers`");

      if(!$result){
        return array("message" => "Таблиця 'teachers' пуста!");
      }
      $teachrs = array();

      while($row = $result->fetch_assoc()){
        array_push($teachrs, $row);
      }

      return $teachrs;
    }
    public function authorization($login, $password){
      $stmt = $this->mysql->prepare("SELECT * FROM `teachers` WHERE login
= ?");
      $stmt->bind_param("s", $login);
      $stmt->execute();
      $result = $stmt->get_result();
      $teacher = $result->fetch_assoc();
      $stmt->close();

      if ($teacher && password_verify($password, $teacher['password'])) {
        return [
            'status' => true,
```

```
            'message' => 'Успішний вхід',
            'id_teacher' => $teacher["id_teacher"]
        ];
      } else {
          return [
              'status' => false,
              'message' => 'Невірний логін або пароль'
          ];
      }
    }
    public function get_teacher_for_id($id){
      $stmt = $this->mysql->prepare("SELECT * FROM `teachers` WHERE
`id_teacher` = ?");
      $stmt->bind_param("s", $id);
      $stmt->execute();
      $result = $stmt->get_result();
      $teacher = $result->fetch_assoc();
      $stmt->close();

      return $teacher;
    }
  }

  if($_SERVER['REQUEST_METHOD'] === 'POST'){
    $data = json_decode(file_get_contents('php://input'), true);

    $login = $data["login"];
    $password = $data["password"];

    $Teacher = new Teacher();
    $result = $Teacher->authorization($login, $password);

    if($result["status"]){
      $_SESSION["teacher_auth_id"] = $result["id_teacher"];
    }

    echo json_encode($result);
  }
```

ДОДАТОК В

Збережена процедура:

```
DELIMITER $$

CREATE PROCEDURE PayForCourse(
    IN p_id_user INT,
    IN p_id_student INT,
    IN p_id_course INT,
    IN p_payment_amount DOUBLE
)
BEGIN
    DECLARE v_balance DOUBLE;
    DECLARE v_paid_to DATE;

    -- Отримати поточний баланс користувача
    SELECT balance INTO v_balance
    FROM users
    WHERE id_user = p_id_user;

    -- Отримати дату "paid_to" для даного користувача та курсу
    SELECT MAX(paid_to) INTO v_paid_to
    FROM course_payments
    WHERE id_user = p_id_user AND id_course = p_id_course;

    -- Перевірити, чи достатньо коштів на балансі
    IF v_balance >= p_payment_amount THEN
        -- Оновити баланс користувача
        UPDATE users
        SET balance = balance - p_payment_amount
        WHERE id_user = p_id_user;

        -- Обчислити нову дату "paid_to"
        IF v_paid_to IS NULL THEN
            -- Якщо даних ще немає в стовпці "paid_to", встановити
поточну дату
            SET v_paid_to = CURDATE();
```

```
        ELSE
            -- Якщо дата вже є, додати один місяць
            SET v_paid_to = DATE_ADD(v_paid_to, INTERVAL 1 MONTH);
        END IF;

        -- Записати інформацію про платіж разом з новою датою "paid_to"
        INSERT INTO course_payments (id_user, id_student, id_course,
payment_amount, paid_to)
        VALUES (p_id_user, p_id_student, p_id_course, p_payment_amount,
v_paid_to);
    ELSE
        -- Якщо недостатньо коштів, видати повідомлення про помилку
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Insufficient
balance';
    END IF;
END$$

DELIMITER ;
```

Тригер:

```
CREATE TRIGGER after_balance_update
AFTER UPDATE ON users
FOR EACH ROW
BEGIN
    IF NEW.balance <> OLD.balance THEN
        INSERT INTO balance_changes (id_user, change_time, change_amount)
        VALUES (NEW.id_user, NOW(), NEW.balance - OLD.balance);
    END IF;
END;
```