МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ ЕЛЕКТРОНІКИ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК
СЕКЦІЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ПРОЕКТУВАННЯ

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

на тему: «Web-додаток підтримки роботи спортивної секції «Goalkeeper society»

за спеціальністю 122 «Комп'ютерні науки»,
освітньо-професійна програма «Інформаційні технології проектування»

Виконавець роботи: студент групи ІТ-61-8 Синицин Єгор Андрійович
Кваліфікаційна робота бакалавра
захищена на засіданні ЕК
з оцінкою _____«_____»
2020 р.

Науковий керівник _____ к.т.н.,доц., Алексенко О.В.
(підпис) (науковий ступінь, вчене звання, прізвище та ініціали)

Голова комісії _____ Шифрін Д.М.
(підпис) (науковий ступінь, вчене звання, прізвище та ініціали)

Засвідчую, що у цій дипломній роботі немає
запозичень з праць інших авторів
без відповідних посилань.

Студент _____
(підпис)

Суми-2020

Сумський державний університет
Факультет електроніки та інформаційних технологій
Кафедра комп'ютерних наук
Секція інформаційних технологій проектування
Спеціальність 122 «Комп'ютерні науки»
Освітньо-професійна програма «Інформаційні технології проектування»

ЗАТВЕРДЖУЮ
Зав. секцією ІТП

_____В. В. Шендрик

«__» _____ 2020 р.

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ БАКАЛАВРА СТУДЕНТУ

Синицин Єгор Андрійович

1 Тема роботи *Web-додаток підтримки роботи спортивної секції «Goalkeeper society*
керівник роботи__ *Алексенко Ольга Василівна., к.т.н., доцент*_____,
затверджені наказом по університету від « 14 » травня 2020 р. № 0576-III

2 Строк подання студентом роботи «1» червня 2020 р.
3 Вхідні дані до роботи: дані про користувачів, інформація про секцію
4 Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1) Аналіз предметної області
2) Моделювання та проектування
3) Практична реалізація

5 Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень): презентація із 23 слайдів: актуальність, постанова задачі, аналіз аналогів, порівняння сайтів-аналогів, моделювання процесу роботи секції, діаграма варіантів використання, засоби реалізації, база даних, архітектура додатку, демонстрація роботи сайту, висновки.

6. Консультанти розділів роботи:

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7.Дата видачі завдання_____01.10.2019_____

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Оформлення планування робіт	До 08.03.2020	
2	Оформлення технічного завдання	До 03.04.2020	
3	Проведення аналізу предметної області	До 08.04.2020	
4	Проведення структурно-функціонального моделювання процесів	До 05.05.2020	
5	Розробка проекту	До 10 05.2020	
6	Публікація в мережі інтернет	До 19 05.2020	
7	Здача пояснювальної записки та файлів розробленого проекту	До 01.06.2020	

Студент _____ Синицин Є.А.

(підпис)

Керівник роботи _____ к.т.н., доц. Алексенко О.В.

(підпис)

РЕФЕРАТ

Тема кваліфікаційної роботи бакалавра «Web-додаток підтримки роботи спортивної секції «Goalkeeper society». Пояснювальна записка складається зі вступу, 3 розділів, висновків, списку використаних джерел із 20 найменувань, додатків. Загальний обсяг роботи – 91 сторінка, у тому числі 61 сторінка основного тексту, 2 сторінки списку використаних джерел, 27 сторінок додатків.

Метою даного проекту є розробка веб-додатку "Goalkeeper society" для підтримки роботи спортивної секції по футболу.

В роботі проведено аналіз аналогів, проектування і розробку додатку. Результатом проведеної роботи є веб-додаток підтримки роботи спортивної секції «Goalkeeper society». Практичне значення роботи полягає у застосуванні сучасних технологій, які допоможуть в залученні молоді займатись спортом.

Ключові слова: PHP, LARAVEL, JAVASCRIPT, ФУТБОЛ, ПЛАГІНИ, ПРОТОТИП, СКРИПТИ, САЙТ

ЗМІСТ

ВСТУП 6
1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ 7
1.1 Огляд останніх досліджень і публікацій 7
1.2 Аналіз програмних продуктів – аналогів 10
1.3 Вибір засобів реалізації 13
1.4 Постановка задачі 25
2 МОДЕЛЮВАННЯ ТА ПРОЕКТУВАННЯ 28
2.1 Моделювання процесу роботи спортивної секції 28
2.2 Проектування інформаційної системи 36
2.3 Проектування моделі бази даних 38
3 ПРОГРАМНА РЕАЛІЗАЦІЯ 42
3.1 Архітектура веб додатку 42
3.2 Програмна реалізація 42
3.3 Використання програмного додатку 48
ВИСНОВКИ 61
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 62
ДОДАТОК А. ТЕХНІЧНЕ ЗАВДАННЯ 64
ДОДАТОК Б. ПЛАНУВАННЯ РОБІТ 68
ДОДАТОК В. ПРОГРАМНА РЕАЛІЗАЦІЯ 84

ВСТУП

Інформаційні технології забезпечують усе більшу кількість аспектів життя людини – від споживання до персональних уподобань, у т.ч. щодо видів спортивної активності. В умовах 21 століття, коли більшість людей веде сидячий і малорухливий спосіб життя, необхідно виділяти час для заняття спортом, щоб елементарно підтримувати своє тіло в тонусі. Це стосується також молоді яка останнім часом все менше веде здоровий і активний спосіб життя, має місце необхідність популяризації різних видів спорту.

Для підтримки фізичного здоров'я як дорослих так і дітей були створені тренажерні зали, спортивні гуртки і секції які спеціалізуються різними видами спорту в тому числі гри в футбол. Проте сьогодні для зацікавлення нових відвідувачів для даних закладів виникає необхідність в розробці власного сайту. При цьому веб-сайт повинен бути максимально цікавим з точки зору молоді, а саме залучати своєю оригінальністю. На сайті користувачі зможуть ознайомитися з різними напрямками в цій сфері, скачати інформаційні та навчальні матеріали, написані досвідченими спортсменами. Також можна ознайомитися з останніми новинами і подіями з життя відомих спортсменів.

Метою даного проекту є розробка веб-додатку "Goalkeeper society" для підтримки роботи спортивної секції по футболу.

Для реалізації поставленої мети потрібно вирішити такі задачі:

– Вивчити процеси організації проведення занять в гри в футбол.

– Провести аналіз сайтів-аналогів для розроблення технічного завдання.

– Обрати та налаштувати інструменти реалізації.

– Визначити структуру даних та спроектувати веб-сайт.

– Реалізувати додаток та розробити інструкції користувача.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Огляд останніх досліджень і публікацій

В сфері освіти дітей, підлітків та молоді з питань організації здорового способу життя потрібен механізм, який забезпечує здоровий спосіб життя як важливого чинника оптимізації життєдіяльності людини. Засоби фізичної культури і спорту, що є однією з ефективних форм реабілітації функціонального стану людини, здатні істотно підвищити загальний функціональний і духовний статус в екстремальних умовах організму. Реалізація даного проекту викликана необхідністю кардинальних перетворень, спрямованих на оздоровлення дітей, вирішення стратегічних завдань. Реалізація веб сайту воротарської школи дозволить значно підвищити масовість футболу. Також футбол вкрай важливий з точки зору популяризації здорового способу життя.

Сучасному школяреві, який проводить більшу частину дня безпосередньо на навчальних заняттях і за підготовкою домашніх завдань, доводиться витримувати великі розумові навантаження при гострому дефіциті рухової активності. В результаті гострого дефіциту рухової активності порушуються захисні функції організму школярів, зростає число випадків з негативним впливом на здоров’я. Природно, в умовах підвищеного навчального навантаження і дефіциту рухової активності учнів, особливої актуальності набуває необхідність ефективної організації фізкультурно-оздоровчої роботи, особливо в позаурочний час. До числа найбільш ефективних засобів такої роботи, безумовно, відноситься футбол – улюблена гра дітей і підлітків. Гра, що відрізняється простотою, доступністю і високою емоційністю.

Для популяризації і підтримки роботи футбольної секції сьогодні досить актуальною є необхідність розробки власного сайту. Адже більшість інформації люди дізнаються саме з мережі інтернет.

Сайт дозволяє структурувати інформацію, представлену у всесвітній павутині завдяки використанню сторінок. Наприклад, сайт IBM містить тисячі сторінок. Перша сторінка сайту (головна сторінка) завдяки використанню меню забезпечує навігацію по усім іншим сторінкам.

Сторінка веб-сайту – це набір текстових файлів з тегами HTML. Коли відвідувач завантажує ці файли на свій комп'ютер, вони обробляються браузером і виводяться на інструмент відображення користувача. HTML дозволяє форматувати текст, виділяти функціональні елементи, створювати гіпертекстові посилання та вставляти на сторінку зображення, записи та інші мультимедійні елементи. Можливо змінити відображення сторінки, додавши стиль CSS, який дозволяє групувати всі елементи форматування (розмір та колір великих літер, розмір та тип вставлених блоків тощо) або сценарій на мові JavaScript, за допомогою якого є можливість переглядати сторінки з подіями.

Сайти можуть містити підрозділи, орієнтовані цілком на ту чи іншу аудиторію. Кожен сайт створюється з якоюсь певною метою. Перш за все, вони потрібні для передачі певної інформації користувачу мережі [1].

Відповідно до інформаційного змісту веб-сайту, сайти можна розділити на сайти-візитки, що містять загальну інформацію власників веб-сайтів чи організацій; представницькі сайти, які іноді також називаються сайтами візитних карток із розширеними функціями, такими як: послуги, портфоліо, зворотній зв'язок, форма зворотного зв'язку, детальний опис; сайт компанії, який містить повну інформацію про компанію власника, послуги, продукцію, події в житті компанії. Він відрізняється від сайтів візитних карток та представницьких сайтів цілісністю наданої інформації, і зазвичай містить різні функціональні інструменти для обробки вмісту (пошук та фільтри, календарі подій, фотогалереї, блоги компаній, форуми). Однак найпопулярнішим типом веб-сайту є інтернет-блог. Блог – це інтернет-журнал, основним наповненням якого є записи, що постійно публікуються [2]. Записи включають текст, фотографії, графіку або мультимедіа. Записи в

блозі зазвичай короткі та розташовані у зворотному хронологічному порядку. У більшості випадків читачі публічних блогів вступають в полеміку з авторами блогу через публікації коментарів чи особисті блоги. Види блогів наведені в таблиці 1.1.

Таблиця 1.1 – Типи веб-блогів

Назва	Опис
Подорожі блоги	Блоги для подорожей мають величезний потенціал для залучення багато трафіку та заробітку. Люди люблять подорожувати, і вони завжди шукатимуть нових можливостей, щоб поїхати кудись, де раніше не були, і вивчити нові культури.
Блоги про здоров'я	Ця ніша також має великі партнерські можливості, оскільки можна писати про продукти харчування, добавки, продукти для фізичних вправ та багато іншого.
Блоги електронного навчання	Електронне навчання зростає блискавично, оскільки інтернет зробив навчання та знання доступними для всіх. Завжди знайдуться люди, які прагнуть отримати нові знання та навички в інтернеті, не виходячи з дому. Можна давати уроки гітари, уроки кодування, уроки щодо використання будь-якого програмного забезпечення та багато іншого.
Технологічні та ігрові блоги	Можна писати огляди додатків та відеоігор, огляди технічних гаджетів, поради щодо ремонту мобільних пристроїв та ПК, навчальні відео-ігри чи будь-яку іншу тему.
Блоги для батьків	Є кілька тем, що стосуються догляду за дітьми та надання допомоги новим мамам вибирати найкращу їжу, одяг, безпечні іграшки та розважальні заходи.

Продовження таблиці 1.1

Назва	Опис
Блоги спільноти	Блоги спільноти створені для того, щоб об’єднати однодумців. Ці блоги зазвичай висвітлюють теми, що цікавлять однодумців, і містять розділ коментарів та сильну підтримку соціальних медіа для сприяння спілкуванню.
Живі блоги	Писати про світові напрямки у своєму блозі – це одне, але показувати їх у реальному часі своїй аудиторії – це інше. Блоги веб-камер, що живуть, включають також ті, де творці вмісту не пишуть вміст, а вербально передають його аудиторії. У поєднанні з чатом у прямому ефірі цей вид ведення блогів є дуже привабливим, а також може бути прибутковим.
Суміші всіх типів	І нарешті, у є блоги, які виступають як суміші всіх типів, про які було сказано раніше. Ці блоги можуть висвітлювати різні теми, націлювати на кілька демографічних даних та містити всі типи мультимедійних форматів.

Отже опираючись на усі тенденції 2020 року та власні інтереси було обрано тему – блог на футбольну тематику. Звичайно, є багато успішних блогів, які можна взяти за модель [3].

1.2 Аналіз програмних продуктів – аналогів

Перед початком розробки поставленої задачі потрібно провести аналіз існуючих аналогів. Аналіз веб-сайтів конкурентів, аналіз цільової аудиторії майбутнього сайту, виявлення «сильних» і «слабких» сторін проекту.

Для досягнення мети було проведено дослідження таких веб-сайтів:

– goalkeeper.at.ua.

– dush.varashosvita.rv.ua.

Сайт goalkeeper.at.ua призначений для надання інформації про школу воротарської майстерності, головна сторінка якого зображена на рисунку 1.1.

Рисунок 1.1 – Головна сторінка сайту goalkeeper.at.ua

До переваг даного сайту можна віднести актуальну інформацію, наявність форуму, розкладу занять і можливість залишати відгуки. Також на даному сайті постійно публікуються новини спорту і школи.

Проте представлений сайт має досить велику кулькість недолівків. До них можна віднести застарілий дизай, що є критичним для користувача. Відсутність зручної панелі навігації.

Схожу тематику і призначення також має сайт dush.varashosvita.rv.ua, представлена на рисунку 1.2.

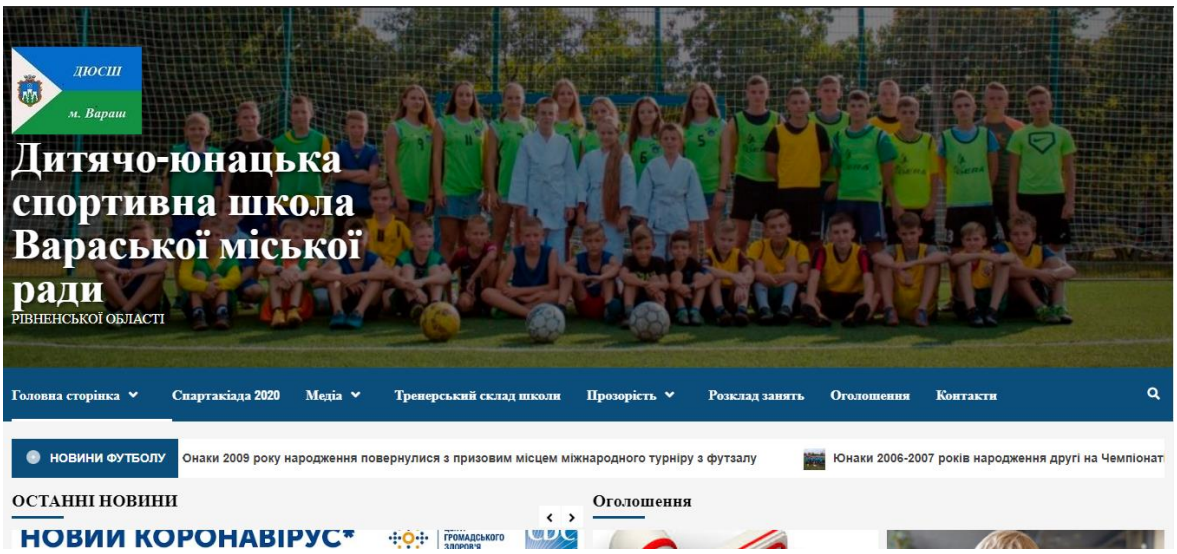

Рисунок 1.2 – Головна сторінка сайту dush.varashosvita.rv.ua

Перевагою даного сайте є сучасний дизайн, зручну панель завігації, досит велика кількість інфрмації і контенту, що більше зацікавлює користувача і користувач довше залишається на сайті.

Проте даний сайт також має недоліки. Можна визначити основні недоліка, а саме перенавантажений інтерфейс який ускладнює користувачу знаходження необхідної інформації. Досить повільна робота сайту, що змушує користувача чекати пока сайт завантажеться.

Проаналізувавши представлені сайти можна виокремити декілька недоліків, а саме:

– Відсутність зручного інтерфейсу.
– Застарілий дизайн.
– Повільна робота сайтів.

Розглянемо детальніше інформацію про сайти та проведемо загальний аналіз (табл.1.2).

Таблиця 1.2 – Аналіз розглянутих сайтів

Критерії	Сайт goalkeeper.at.ua	Сайт dush.varashosvita.rv.ua
Відсутність непотрібної реклами	+	+
Неактуальна інформація	-	-
Можливість зворотного зв'язку	+	-
Зручність використання	-	-
Повільна робота сайту	+	+
Застарілий дизайн	+	-

Отже, в результаті проведеного аналізу сайтів зі схожою тематикою і призначенням було вирішено розробити сайт який не матиме недоліків виявлених на сайтах представлених в таблиці 1.2.

Дана розробка матиме всі переваги даних сайтів, а також матиме унікальний дизайн та зручний інтерфейс.

1.3 Вибір засобів реалізації

Розробка веб-сайтів – це процес створення веб-сайтів та додатків для інтернету або для приватної мережі, відомий як інтернет. Розробка веб-сайтів не стосується дизайну веб-сайту; швидше, це все про кодування та програмування, що забезпечує функціональність веб-сайту [4].

Від найпростіших, статичних веб-сторінок до платформ і додатків соціальних медіа, від веб-сайтів електронної комерції до систем управління контентом (CMS); всі інструменти, якими ми користуємось через інтернет щодня, були розроблені веб-розробниками.

Веб-розробка може бути розбита на три шари (табл. 1.3): кодування на стороні клієнта (фронтенд), кодування на стороні сервера (бекенд) та технології баз даних.

Таблиця 1.3 – Частини веб розробки

№	Назва	Опис	Приклад технологій розробки
1	Клієнтська сторона	Сценарії на стороні клієнта або розробка інтерфейсу стосується всього, що переживає кінцевий користувач безпосередньо. Код клієнта виконується у веб-браузері і безпосередньо стосується того, що бачать люди, відвідуючи веб-сайт. Такі речі, як компонування, шрифти, кольори, меню та контактні форми, керуються фронтендом.	HTML, CSS, Vue.js, Vuetify, JavaScript, React.js, Bootstrap.

Продовження таблиці 1.3

№	Назва	Опис	Приклад технологій розробки
2	На стороні сервера	Сценарії на стороні сервера або розробка бекенду – це все, що відбувається за кадром. Бекенд – це по суті частина веб-сайту, яку користувач насправді не бачить. Він несе відповідальність за зберігання та впорядкування даних та забезпечує те, що все на стороні клієнта працює безперебійно.	PHP, Laravel, Node.js, Python, Express.js, Docker.
3	Технологія баз даних	Веб-сайти також покладаються на технологію баз даних. База даних містить усі файли та вміст, необхідний веб-сайту для функціонування, зберігаючи його таким чином, щоб полегшити його роботу. База даних працює на сервері, і більшість веб-сайтів зазвичай використовують певну форму управління реляційною базою даних	MySQL, PostgresSQL, OracleSQL, MongoDB

1.3.1 Розробка інтерфейсу

Завдання розробника frontend – кодувати інтерфейс веб-сайту чи програми; тобто частина веб-сайту, яку бачить і взаємодіє користувач. Вони беруть дані і перетворюють їх у те, що легко зрозуміле, візуально приємне та повністю функціональне для повсякденного користувача. Вони працюють над розробками, наданими веб-дизайнером, і реалізують їх за допомогою HTML, JavaScript та CSS [5].

Мови розмітки використовуються для визначення форматування текстового файлу. Іншими словами, мова розмітки повідомляє програмне забезпечення, яке відображає текст, як слід форматувати текст. Мови розмітки є повністю розбірливими для читання – вони містять стандартні слова, але теги розмітки приховані від користувача в кінцевому варіанті роботи.

Дві найпопулярніші мови розмітки – HTML та XML. HTML розшифровується як мова розмітки HyperText та використовується для створення веб-сайтів. Додані до звичайного текстового документа, всі теги HTML описують, як цей документ повинен відображатися веб-браузером.

Невідмінна частина будь-якої веб сторінки – каскадні стилі розмітки сторінки або CSS. Стилі – це в основному сукупність стилістичних правил. Мови аркушів стилів використовуються, доволі буквально, для оформлення документів, написаних мовами розмітки.

CSS розшифровується як каскадні таблиці стилів з акцентом на «стиль». Хоча HTML використовується для структури веб-документа, визначає такі речі, як заголовки та абзаци, і дозволяє вставляти зображення, відео та інші медіа, CSS визначає стиль створеного документа.

1.3.2 Розробка серверної частини

Код, який створюють розробники бекенду, гарантує, що все, що будує розробник frontend, є повністю функціональним. Головне завдання

розробника серверної частини – переконатися, що сервер, програма та база даних взаємодіють між собою. Для вирішення такої складної задачі розробники використовують серверні мови, такі як PHP, Ruby, Python та Java для створення програми.

Веб-технології стрімко розвиваються, і щоб встигати за швидкістю розвитку нових технологій, у веб-програмістів часто бракує часу для реалізації повсякденних завдань з нуля, таких як: автентифікація, API, тестування, дебаг, кешування і т.д. Саме для спрощення розробки сайтів і швидкого вирішення цих завдань були придумані веб-фреймворки [6].

Фреймворк – це каркас, призначений для створення динамічних веб-сайтів, веб-додатків, служб або ресурсів. Він спрощує розробку і позбавляє від необхідності писати звичайний код. Фреймворк спрощує доступ до бази даних, розробку інтерфейсу та зменшує дублювання коду [7].

Розглянемо один з популярних фреймворків Yii 2. Yii – це універсальний фреймворк, і він може бути задіяний у всіх типах веб-додатків.

Завдяки своїй складовій структурі та чудовій підтримці кешу, фреймворки особливо підходять для розробки великих проектів, таких як портали, форуми, CMS, магазини або програми RESTful. Yii – це командний проект. Він підтримується і розвивається сильною командою та великою спільнотою розробників. Автор цього фреймворку стежать за тенденціями розвитку веб-сайтів та інших проектів.

Найбільш підходящі можливості і кращі практики регулярно впроваджуються в фреймворк в вигляді простих і елегантних інтерфейсів:

– Як і багато інших PHP фреймворків, для організації коду Yii використовує архітектурний патерн MVC.

– Yii дотримується філософії простого і елегантного коду не намагаючись ускладнювати дизайн тільки заради проходження будь-якими шаблонами проектування.

– Yii є full-stack фреймворком і включає в себе перевірені і добре зарекомендовані в себе можливості: ActiveRecord для реляційних і NoSQL баз даних, підтримку REST API, багаторівневе кешування та інше.

– Yii відмінно масштабований. Можна налаштувати або замінити практично будь-яку частину основного коду. Використовуючи архітектуру розширень, легко ділитися кодом або використовувати код спільноти.

– Одна з головних цілей Yii – продуктивність [8].

Розглянемо також не менш популярний фреймворк Laravel 7. Laravel – це фреймворк для веб-додатків з виразним і елегантним синтаксисом. Він дозволить спростити вирішення основних наболілих завдань, таких як автентифікація, маршрутизація, сесії і кешування. Laravel – це спроба об'єднати все найкраще, що є в інших PHP фреймворк, а також Ruby on Rails, ASP.NET MVC і Sinatra [9].

Laravel – доступний, але потужний. Має безліч відмінних інструментів для великих, надійних додатків:

– Як і багато інших PHP фреймворків, для організації коду Laravel використовує архітектурний патерн MVC.

– Laravel дотримується філософії виразного, красивого синтаксису. Він призначений для людей, які цінують елегантність, простоту і читаність.

– Функціонал Laravel є простим для розуміння і використання.

– Більшість функцій Laravel чудово працюють, не вимагаючи додаткових налаштувань. Вони спираються на загальноприйняті стандарти написання коду, роблячи його інтуїтивно зрозумілим.

– Документація Laravel закінчена і постійно оновлюється.

– Чудова IoC (Інверсія управління).

– Зручна система міграцій.

– Інтегрована система модульного тестування.

Обидва фреймворка для організації коду використовують архітектурний патерн MVC [10]. При детальному розгляді ці фреймворки складаються з трьох основних компонентів:

– Model – це сам додаток який нічого не знає про HTTP запитах.

– View – це HTTP запит / відповідь і уявлення даних, які вимагає клієнт від сервера.

– Controller – це кілька і більше класів, чиє завдання – абстрагування моделі від HTTP.

За архітектурою дані фрейморкі ідентичні, так як реалізують один і той же архітектурний патерн. Він дозволяє швидко реалізувати проект, а також легко супроводжувати і масштабувати його в подальшому.

Розширення важлива частина фреймворку, тому розглянемо їх докладніше. Розширення – це можливість підключати додаткові модулі або бібліотеки для фрейморка, які дозволять розширити його можливості. Yii і Laravel підтримують таку можливість. На даний момент, розширень набагато більше, звичайно ж, у Yii, так як фреймворк живе набагато довше, ніж Laravel, однак другий, в свою чергу, розвивається дуже великими темпами і включає в себе багато всього для роботи відразу «з коробки».

Розширення є абсолютно для будь-якого завдання, будь то панель адміністратора або ж платіжна система.

Чимало важливим фактором при виборі фреймворка є його документація. Через малу кількість років розробки або особистих переконань авторів документація Laravel дуже мізерна і неінформативна. Тому для навчання доведеться дивитися вихідний код, і як наслідок, не завжди буде зрозуміло, що робить та чи інша функція. В Yii ж є велика і корисна документація, де описана кожна функція із зазначенням її призначення і приклад використання. Так що, безсумнівно, це величезний плюс фреймворка Yii.

Для належної роботи сайту важлива підтримка REST API. REST – це архітектурний стиль взаємодії між компонентами розподілених додатків у мережі. REST – це набір послідовних обмежень, які слід враховувати при проектуванні розподіленої системи гіпермедіа. У деяких випадках (інтернет-магазини, пошукові системи, інші системи на основі даних) це може

підвищити продуктивність та спростити архітектуру. Загалом, компоненти REST взаємодіють із клієнтами та серверами у всесвітній мережі.

В Інтернеті віддалені процедурні виклики можуть бути звичайними HTTP-запитами (зазвичай "GET" або "POST"; це називається "REST-запит"), а необхідні дані передаються як параметри запиту.

Для веб-служб, які відповідають REST тобто, не порушуючи жодних обмежень, що застосовуються до нього, використовується термін "RESTful". На відміну від веб-служб на основі SOAP, веб-API RESTful не має "офіційного" стандарту. Справа в тому, що REST – це архітектурний стиль, а SOAP – протокол. Хоча сам REST не є стандартом, більшість реалізацій RESTful використовують стандарти HTTP, URL, JSON та XML [11].

В кінцевому рахунку, в даний час всі сучасні PHP фреймворки зобов'язані підтримувати REST архітектуру. Нижче наведено список можливостей фреймворків для роботи з REST API.

Yii 2:

– Підтримка JSON, JSONP і XML.
– Роутинг відповідно до REST-запитами.
– Підтримка HATEAOS.
– Кешування запитів.
– Обмеження швидкості.

Laravel:

– Налаштування роутінга REST-запитів.
– Підтримка JSON, JSONP.

На основі проведеного аналізу був зроблений вибір на сторону, що активно розвивається Laravel, його структура дозволяє легко масштабувати веб-додаток. Yii є більш стабільним, але менш масштабованим.

1.3.3 Розробка бази даних

Для розробки бази даних використовують такі інструменти, як MySQL, Oracle чи SQL Server, щоб знаходити, зберегти або відредагувати дані та повернути їх користувачеві за допомогою коду візуальної частини.

Вище перелічені мови використовуються не лише для створення веб-сайтів, програмного забезпечення та додатків; вони також використовуються для створення та управління базами даних.

Бази даних не розроблені для розуміння тих же мов, на яких запрограмовані програми, тому важливо мати мову, яку вони розуміють – як SQL, стандартну мову для доступу та маніпулювання реляційними базами даних. SQL означає структурована мова запитів.

Вона має власну розмітку і в основному дозволяє програмістам працювати з даними, що зберігаються в системі баз даних [12]. Розглянемо детальніше різни типи, переваги та недоліки баз даних (табл.1.4).

Таблиця 1.4 – Бази даних

№	Назва	Переваги	Недоліки
1	Oracle	Має останні інновації та функції, що надходять від їх продукції, оскільки Oracle прагне встановити планку для інших інструментів управління базами даних.	Вартість Oracle може бути непосильною, особливо для менших організацій.

Продовження таблиці 1.4

№	Назва	Переваги	Недоліки
1	Oracle	Інструменти управління базами даних Oracle також надзвичайно надійні, і ви можете знайти той, який може робити практично все, про що ви можете подумати [13].	Після встановлення система може вимагати значних ресурсів, тому для впровадження Oracle може знадобитися оновлення обладнання.
2	MySQL	Безкоштовний.	Ви можете витратити багато часу і сил, щоб змусити MySQL робити те, що інші системи роблять автоматично.
		Пропонує безліч функціональних можливостей.	Немає вбудованої підтримки для XML або OLAP.
		Існують різноманітні інтерфейси користувачів.	Підтримка доступна для безкоштовної версії, але за неї потрібно заплатити.
		Це може бути зроблено для роботи з іншими базами даних, включаючи DB2 та Oracle.	

Продовження таблиці 1.4

№	Назва	Переваги	Недоліки
3	Microsoft SQL Server	Це дуже швидко і стабільно.	Ціни на підприємства можуть перевищувати багато організацій.
		Пропонує можливість регулювання та відстеження рівнів продуктивності, що може зменшити використання ресурсів.	Навіть при налаштуванні продуктивності Microsoft SQL Server може обробляти ресурси.
		Є можливість отримати доступ до візуалізації на мобільних пристроях.	Багато людей мають проблеми з використанням інтеграційних служб SQL Server для імпорту файлів [14].
		Дуже добре працює з іншими продуктами Microsoft.	
4	PostgreSQL	Ця система управління базами даних є масштабованою і може обробляти терабайти даних.	Не чітка документація.
		Підтримує JSON.	Конфігурація може бути заплутаною.
		Існують різноманітні заздалегідь визначені функції.	Швидкість може постраждати під час великих масових операцій чи запитів.
		Доступна низка інтерфейсів.	

Продовження таблиці 1.4

№	Назва	Переваги	Недоліки
5	MongoDB	Швидка і проста у використанні.	Установки за замовчуванням не захищені
		Підтримує JSON та інші документи NoSQL.	Налаштування може бути тривалим процесом.
		Дані будь-якої структури можна зберігати та отримувати доступ до них швидко та легко.	Інструменти для перекладу SQL на запити MongoDB доступні, але вони додають додатковий крок до використання системи.
		SQL не використовується як мова запитів.	
6	MariaDB	Шифрування доступне на мережевому, серверному та додатковому рівнях.	Як і у багатьох інших безкоштовних двигунів бази даних, вам доведеться платити за підтримку.
		Розширювана архітектура та плагіни дозволяють налаштувати інструмент відповідно до ваших потреб.	Двигун все ще досить новий, тому немає гарантій, що наступні оновлення та версії будуть найближчими.
		Система швидка і стабільна.	

В ході виконання даного проекту, вибір було зроблено на користь MySQL. По-перше, дана СУБД є безкоштовною. По-друге, супутній продукт

MySQL Workbench дозволяє просто взаємодіяти зі встановленою БД, надаючи можливість через зручний інтерфейс користувача виконувати будь-які маніпуляції з даними [15].

Та нарешті, ця СУБД є однією з найбільш розповсюджених, займаючи друге місце за популярністю у світі, поступаючись лише корпоративній СУБД від Oracle. А це означає, що на випадок проблем чи збоїв в роботі цієї системи, в мережі можна знайти величезних кількість матеріалу для їх вирішення.

Для розробки клієнтської частини сайт було обрано мови програмування HTML, CSS і JavaScript так як вони є най популярнішими мовами для веб розробки і досить легкі у вивченні.

Розробка серверної частини буде відбуватись за допомогою мови програмування PHP з використанням фреймфорку Laravel. Ця мова активно використовується багатьма сайтами і досить легка у вивченні. Також найбільш популярні системи контролю контентом використовують саме цю мову програмування, що надає можливість розробляти додатковий функціонал.

1.4 Постановка задачі

Виходячи із визначеної вище актуальності проблеми забезпечення секції «Goalkeeper society», було визначено, що потрібно створити сайт для того, щоб кожен бажаючий міг отримати інформаційну підтримку щодо розвитку навичок в грі футбол, зокрема в якості голкіпера. Аудиторією сайту будуть як починаючі, так просунуті голкіпери.

Метою роботи є розробка веб-додатку для підтримки роботи спортивної секції «Goalkeeper society» у формі сайту.

Для реалізації поставленої мети потрібно вирішити такі задачі:

– Вивчити процеси організації проведенні занять в гри в футбол та провести аналіз сайтів-аналогів для розроблення технічного завдання.

– Обрати та налаштувати інструменти реалізації.

– Визначити структуру даних та спроектувати веб-сайт.

– Реалізувати додатки та розробити інструкції користувача.

Перелік вимог до сайту:

1. Сайт повинен складатись з двох частин, а саме користувацької та адміністративної.

2. Користувацька частина призначена для не зареєстрованих користувачів які маю можливість користуватись сайтом не виконуючи авторизацію.

Користувацька частина повинна складатися із наступних сторінок:

– Головна.

– Розклад занять.

– Новини футболу.

– Новини школи.

– Фотогалерея.

– Відео-галерея.

– Контакти.

– Заповнити заявку на тренування.

На головній сторінці сайту повинні бути такі елементи:

– Навігаційне меню.

– Останні публікації сайту.

– Слайдер з фото проведення занять.

Доступ до адміністративної частини матимуть лише адміністратори сайту. Для них повинен бути розроблений функціонал який надає можливість додавати публікації, змінювати інформацію, переглядати список заявок на тренування.

Адміністративна частина повинна складатися із наступних сторінок:

– Головна.

– Загальна інформація.

– Галерея.

– Новини.
– Заявки користувачів.
– Графік занять.

Детальна інформація щодо розробки веб-додатку знаходиться наведена у розробленому в ході аналізу предметної області технічному завданні (Додаток А).

2 МОДЕЛЮВАННЯ ТА ПРОЕКТУВАННЯ

2.1 Моделювання процесу роботи спортивної секції

2.1.1 Моделювання процесу роботи спортивної секції у нотації IDEF

Для розробки веб-додатку підтримки роботи секції спочатку потрібно вивчити процеси її роботи, в яких буде використаний додаток.

Рисунок 2.1 – Контекстна діаграма процесу прийому заявок

Діаграми першого рівня деталізують функції процесу нульового рівня. Так, функціональний блок 0 розкладається на сукупність взаємопов'язаних підфункцій, а саме створення заявки, перевірка заявки, назначення часу, заняття.

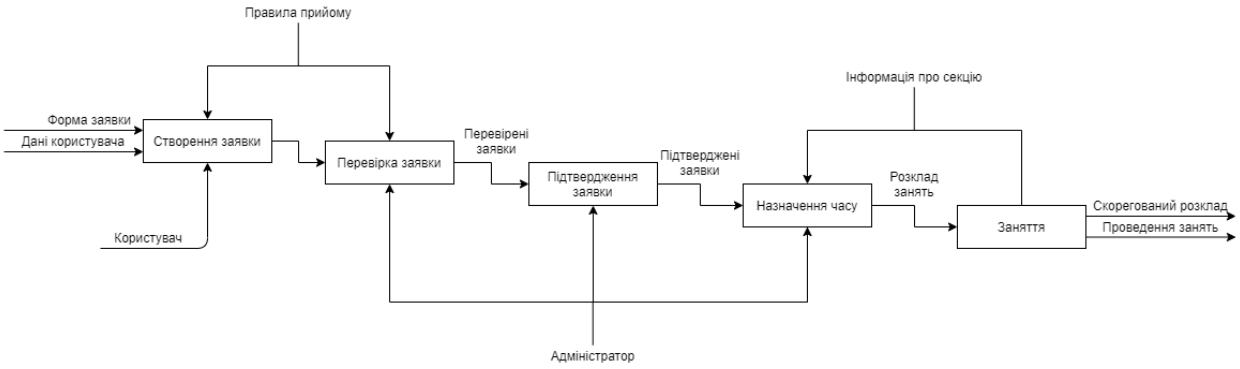

Рисунок 2.2 – Діаграма декомпозиції процесу прийому заявок

2.1.2 Модель варіантів використання веб-додатку

За результатами вивчення роботи секції було виділено акторів майбутнього веб-додатку (табл.2.1) та сформовано опис варіантів його використання (табл.2.2).

Таблиця 2.1 – Опис акторів

Назва	Опис
	Актори-користувачі
Користувач	Користувач додатка, що використовує функції «гість».
Адміністратор	Користувач додатка, що використовує функції «адміністратора».
	Актори-зовнішні системи
База даних	База даних, що зберігає інформацію.

Таблиця 2.2 – Опис варіантів використання

Назва	Опис
Вхід в систему	Функція входу в систему.
Залишити коментар	Фукція, щоб надати можливість користувачу залишити коментарій.

Продовження таблиці 2.2

Назва	Опис
Перегляд інформації	Кожен користувач має можливість переглянути інформацію на сайті.
Пошук інформації	Можливість виконувати пошук інформації на сайті.
Відправка заявок	Функція для відправки заявки на тренування користувачем.
Редагувати дані	Адміністратор може редагувати дані на сайті.
Додавати публікації	Адміністратор може додати нові публікації.
Перегляд заявок	Адміністратор може переглядати заявки.

Рисунок 2.3 – Діаграма варіантів використання

2.1.3 Модель аналізу розроблюваного ПЗ

Модель аналізу містить деталізований опис сценаріїв виділених у попередній моделі варіантів використання.

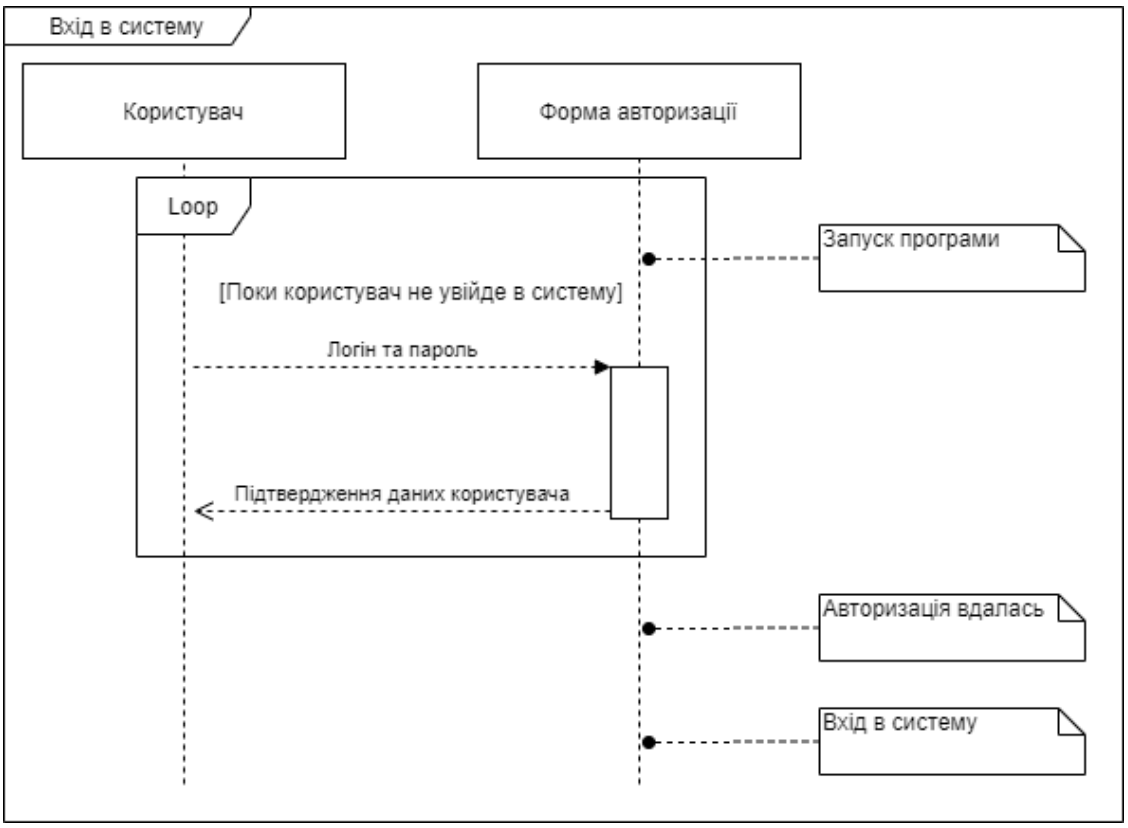

Рисунок 2.4 – Діаграма послідовності входу в систему

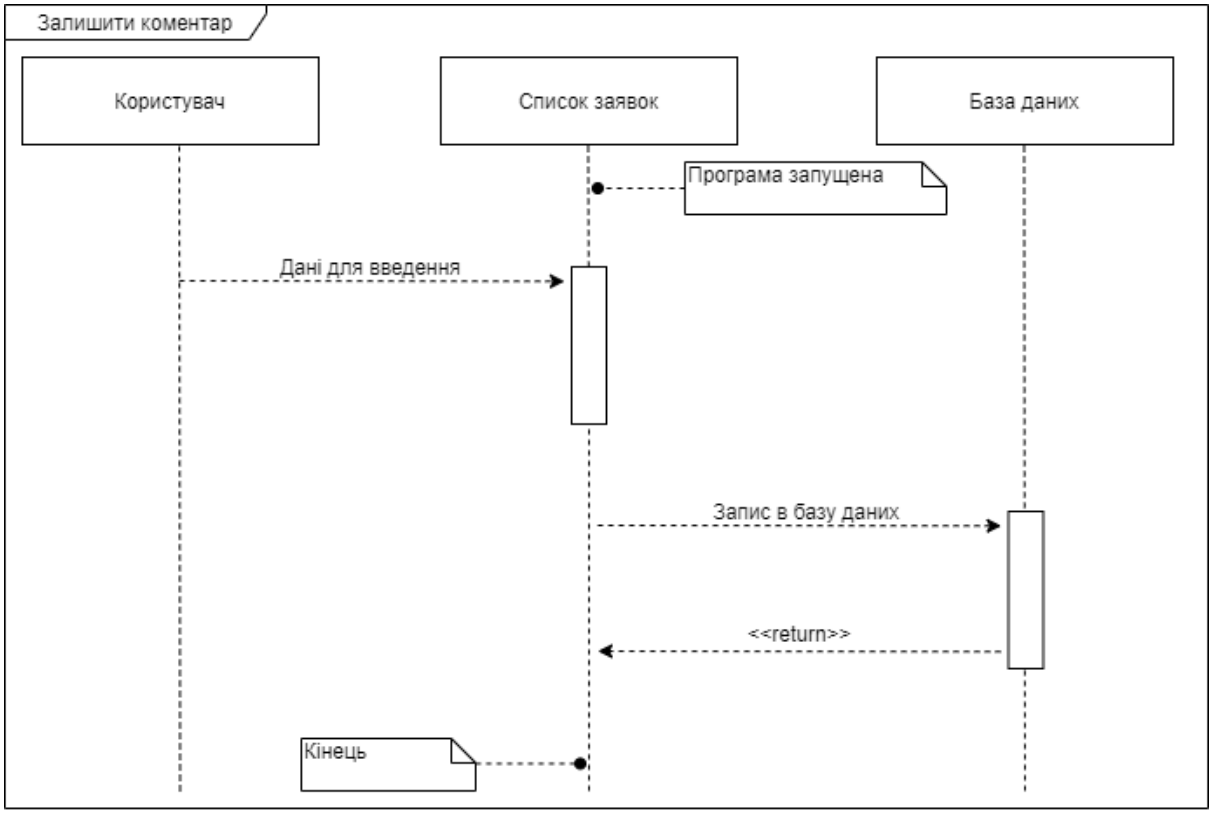

Рисунок 2.5 – Діаграма послідовності коментування

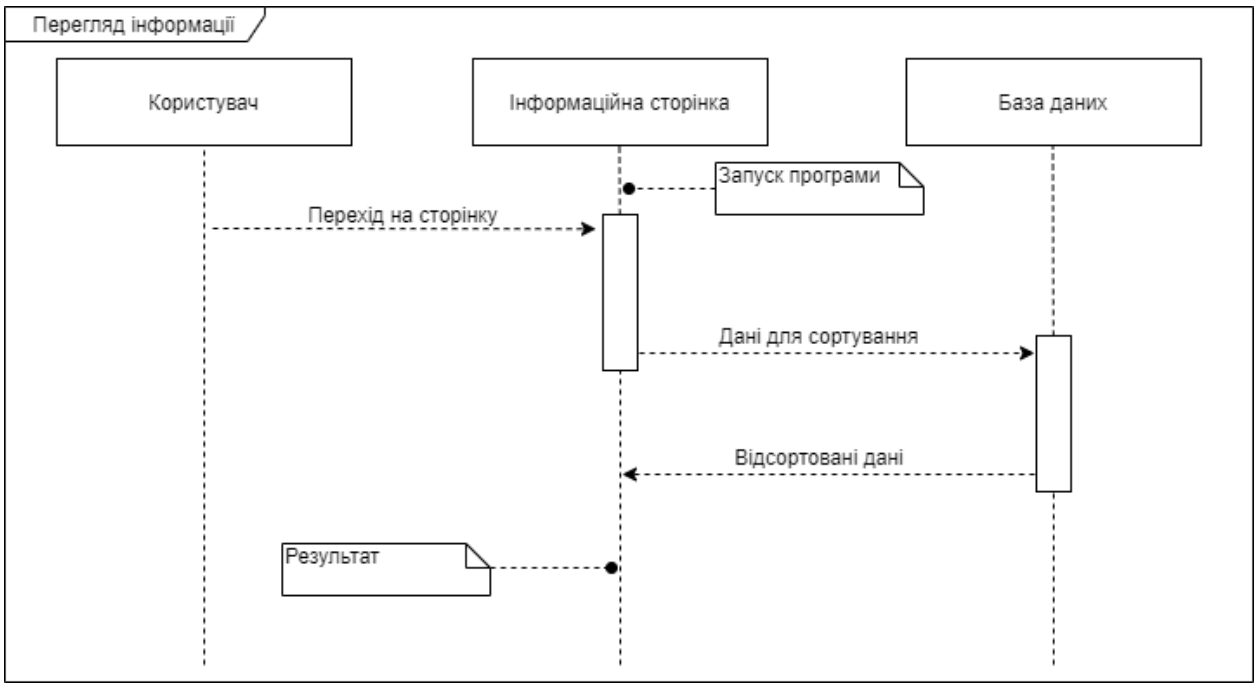

Рисунок 2.6 – Діаграма послідовності перегляду інформації

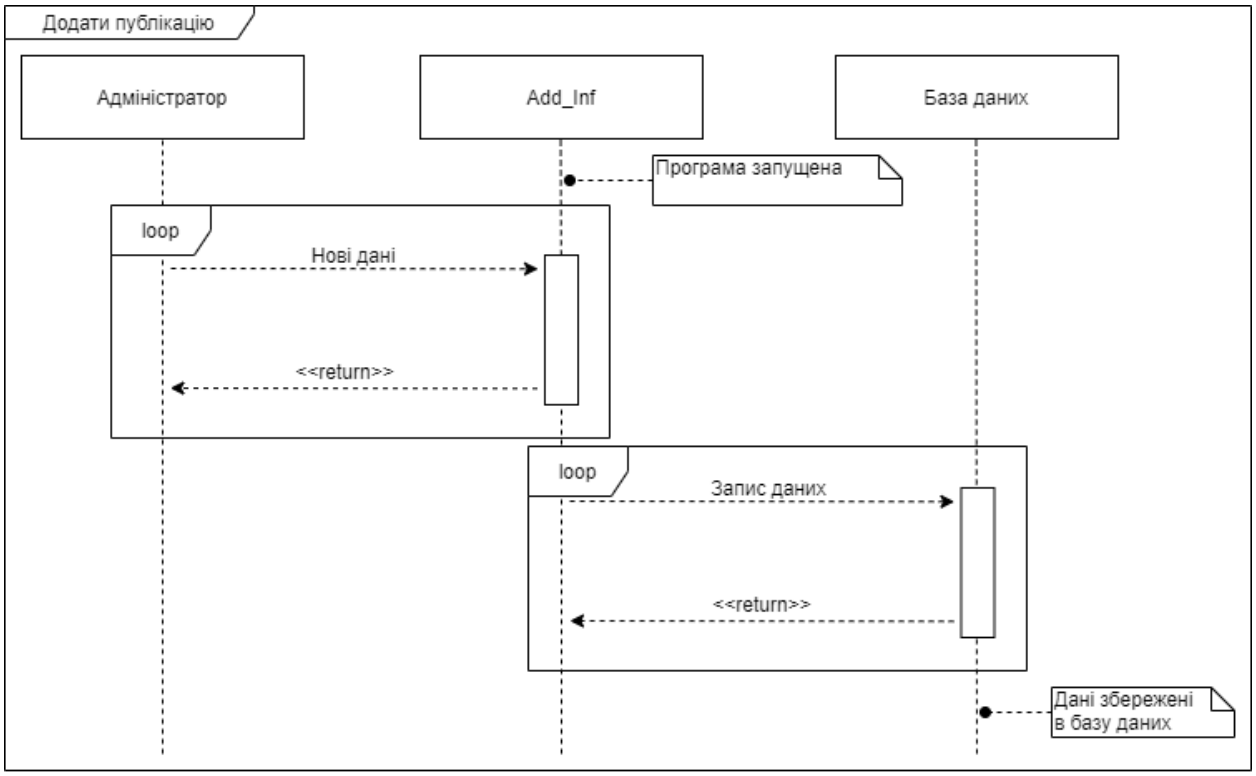

Рисунок 2.7 – Діаграма послідовності додавання інформації

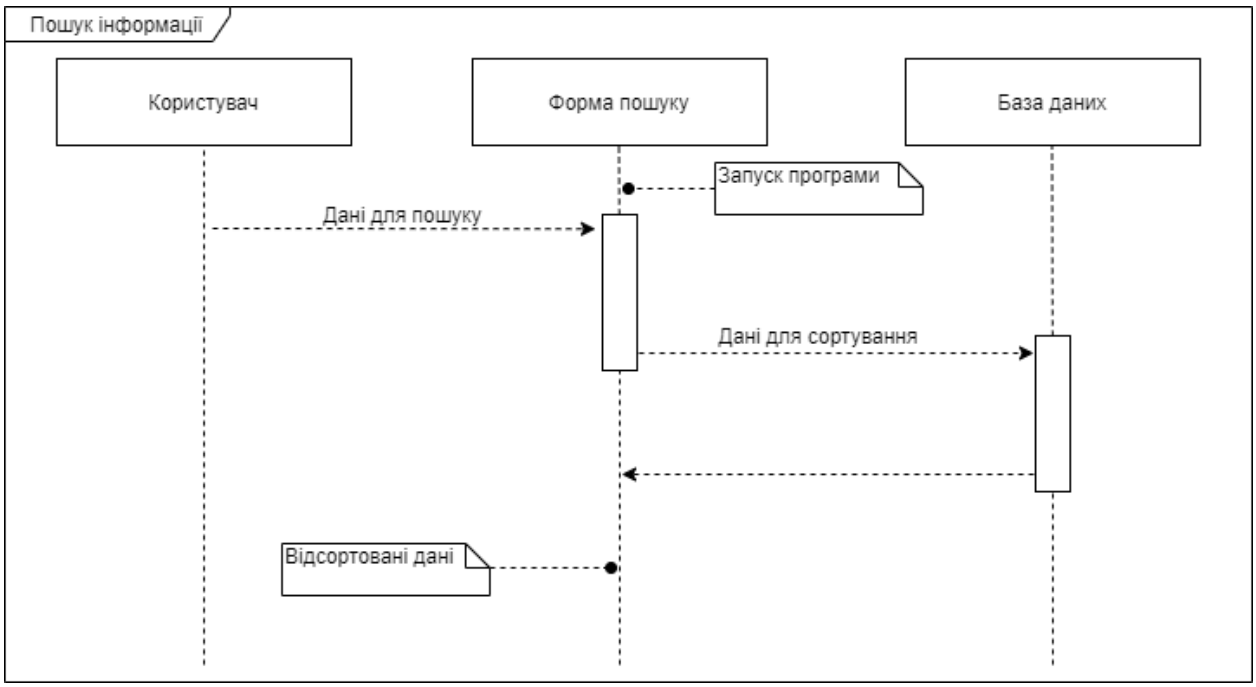

Рисунок 2.8 – Діаграма послідовності пошуку інформації

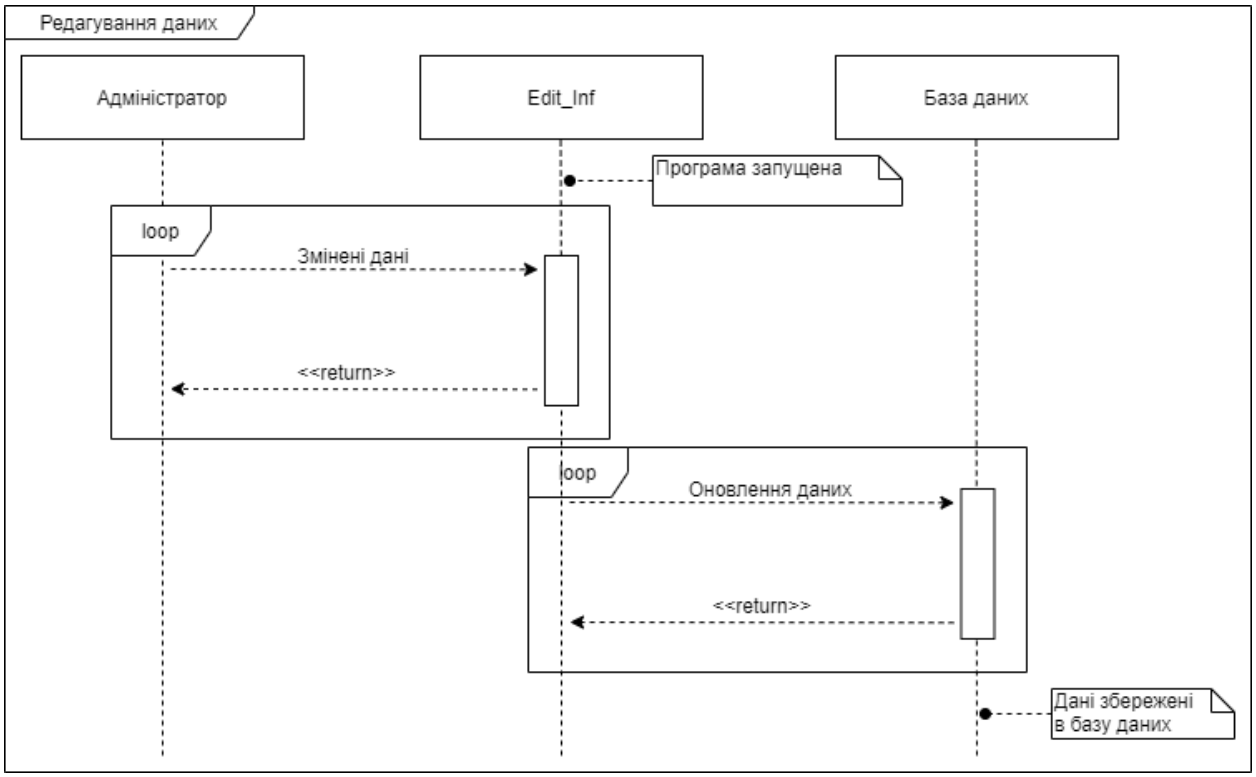

Рисунок 2.9 – Діаграма послідовності редагування інформації

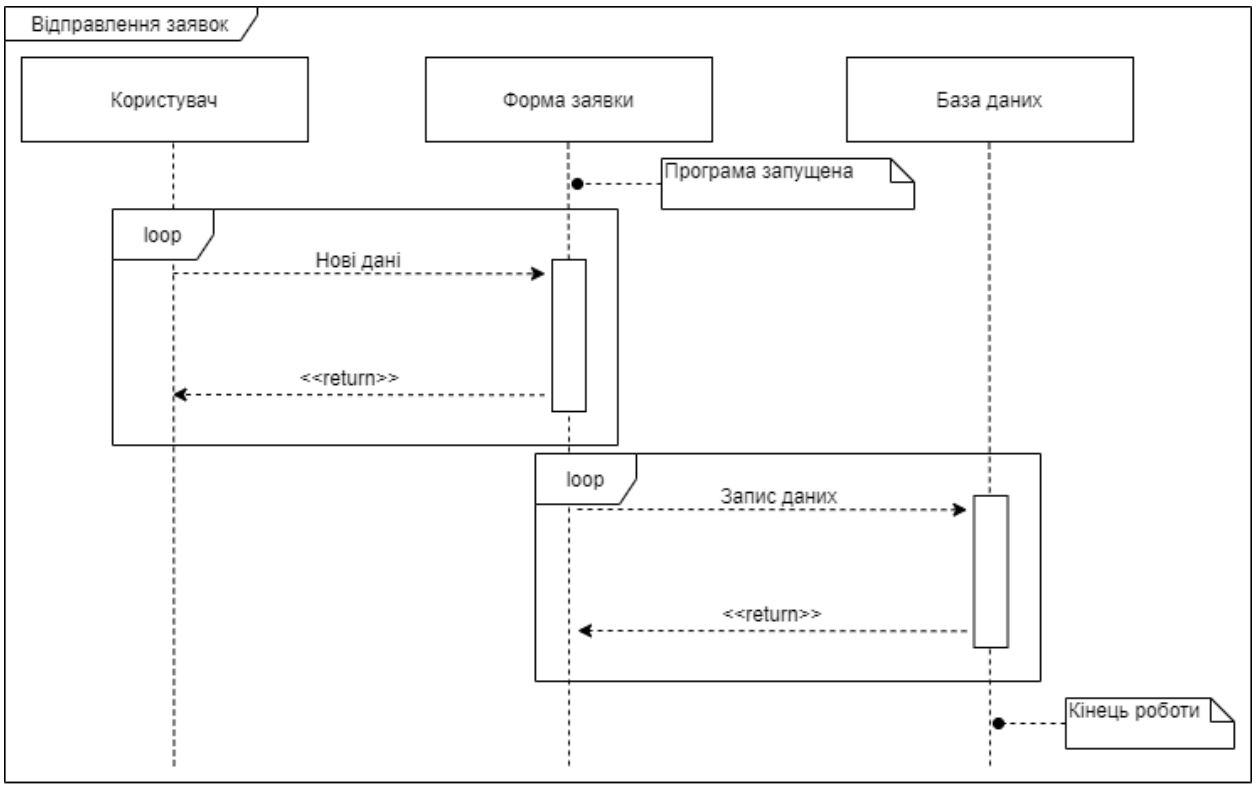

Рисунок 2.10 – Діаграма послідовності додавання заявки

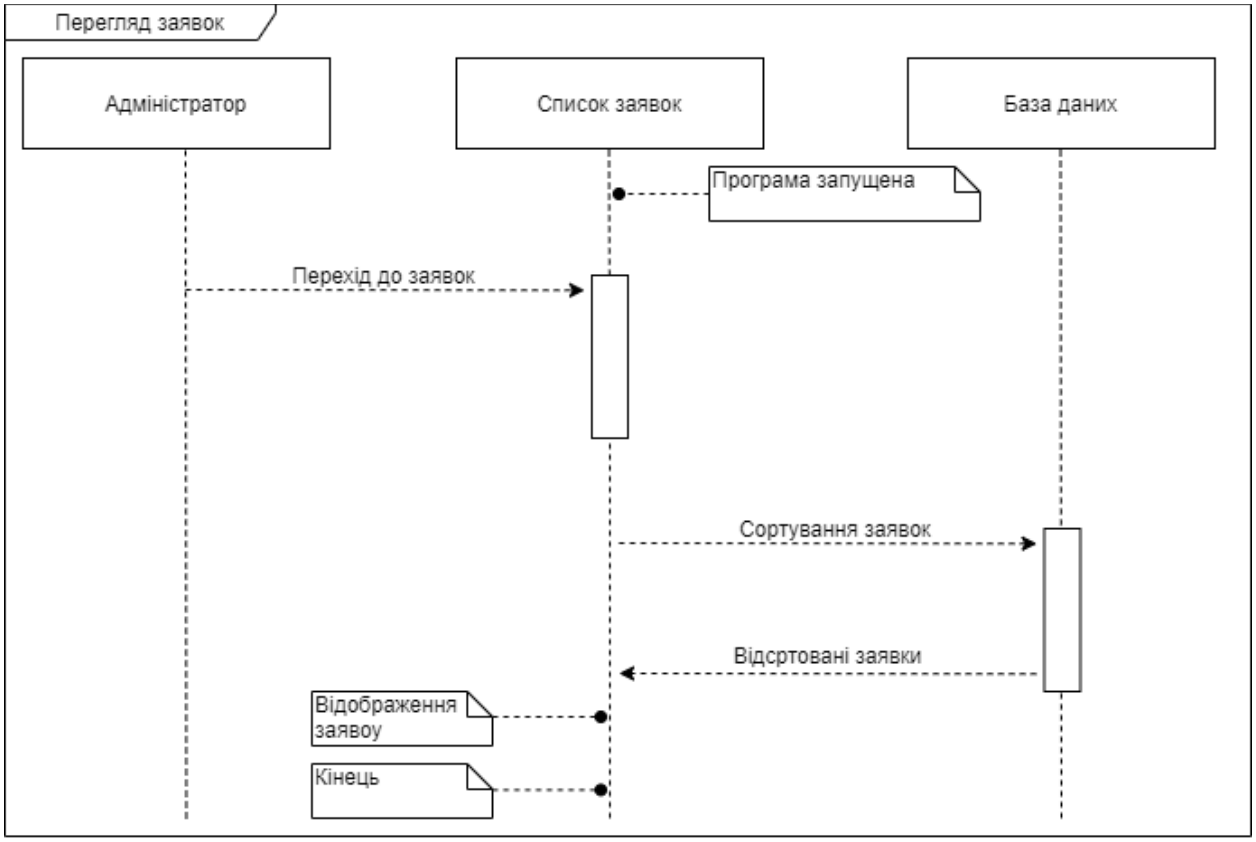

Рисунок 2.11 – Діаграма послідовності перегляд заявки

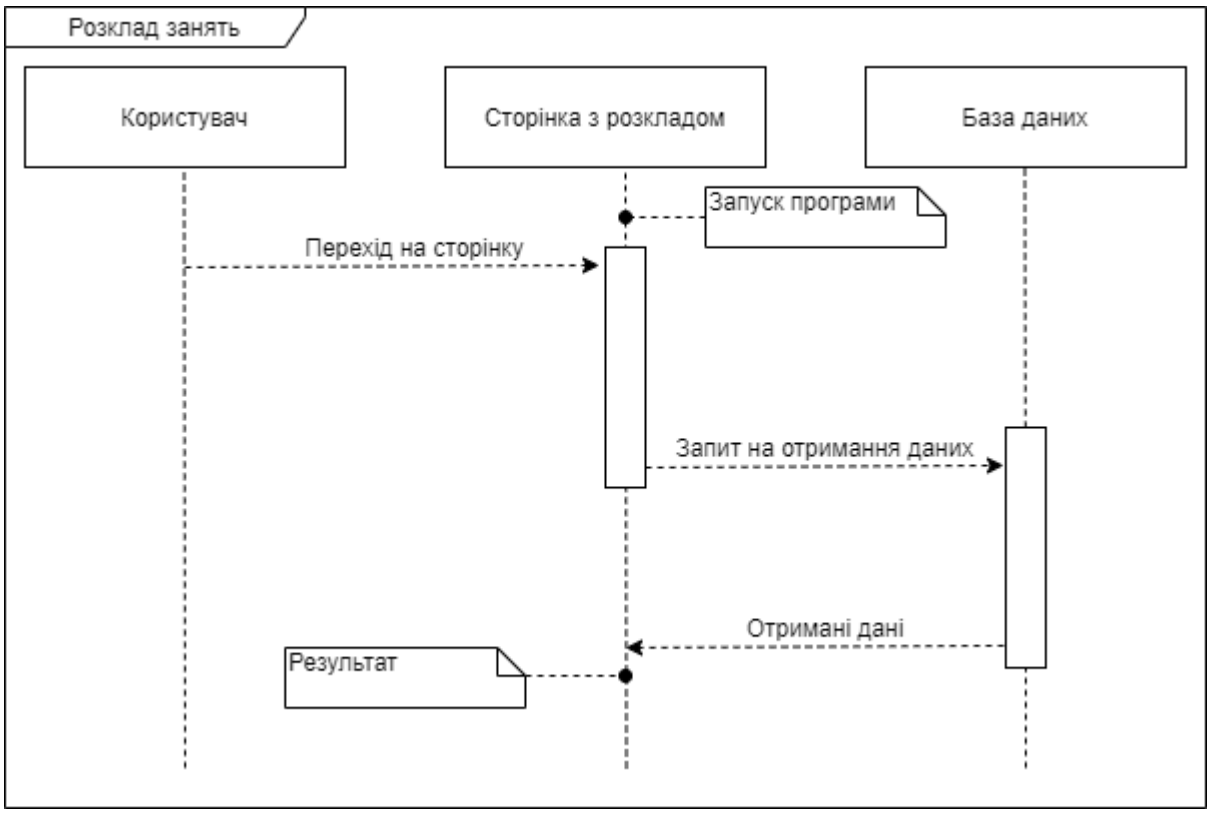

Рисунок 2.12 – Діаграма послідовності перегляд розкладу занять

Діаграма комунікацій – це особливий вид діаграм взаємодії, акцентованих на обміні даними між різними учасниками взаємодії.

Замість того щоб малювати кожного учасника у вигляді лінії життя і показувати послідовність повідомлень, розташовуючи їх по вертикалі, як це робиться в діаграмах послідовності, комунікаційні діаграми допускають довільне розміщення учасників, дозволяючи малювати зв'язку, що показують відносини учасників, і використовувати нумерацію для подання послідовності повідомлень [17].

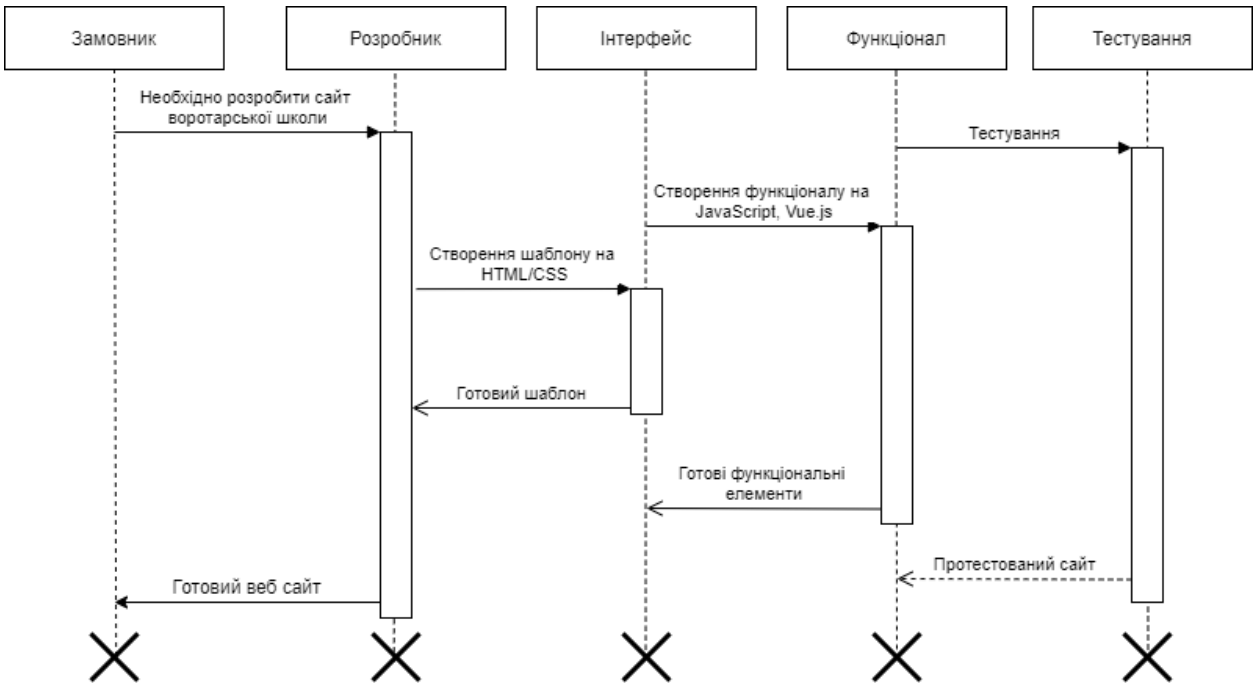

Рисунок 2.13 – Діаграма комунікації

2.2 Проектування інформаційної системи

Після затвердження постановки завдання на розробку веб-сайту починається розробка дизайну.

Прототип веб-дизайну – це проста схема сторінки веб-сайту, що показує структурні елементи майбутнього веб-сайту у вигляді ескізів, або html-документів: меню, кнопки, таблиці тощо. Прототипом може бути статичним зображенням або динамічним html-документом [18].

Орієнтуючись на технічне завдання був розроблений ескіз головної сторінки (рис 2.14).

Рисунок 2.14 – Прототип дизайну сайту

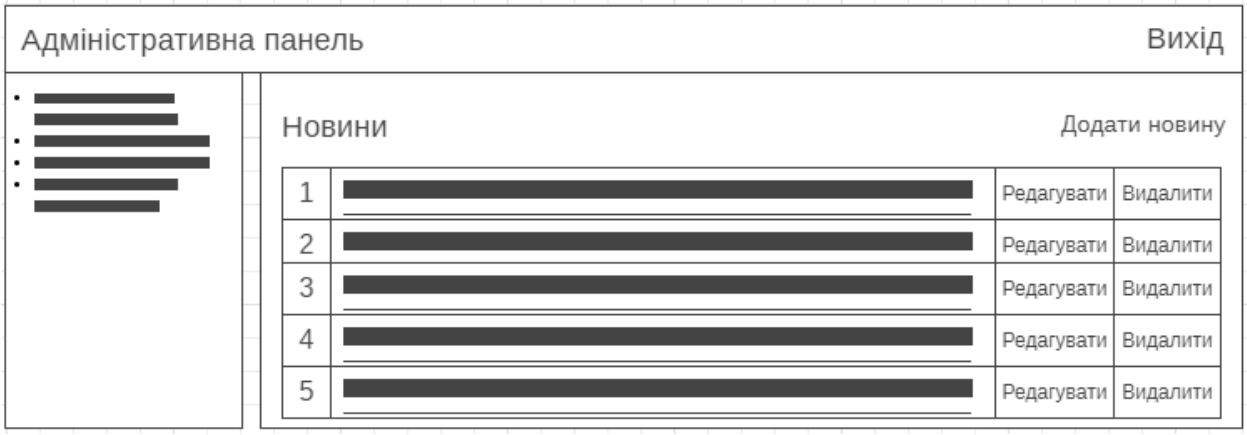

Рисунок 2.15 – Прототип дизайну адміністративної панелі сайту

2.3 Проектування моделі бази даних

Під час створення бази даних необхідно слідкувати за впорядкованістю інформації, що в ній зберігатиметься, для того щоб в процесі подальшої роботи можна було б отримувати чи модифікувати дані в зручному вигляді. Для цього бажано вхідні дані структурувати. Структуризація – це сукупність угод щодо шляхів представлення інформації. Зрозуміло, що структурувати інформацію можна по-різному. Залежно від структури розрізняють ієрархічну, мережеву, реляційну, об'єктно-орієнтовану і гібридну модель баз даних. Найпопулярнішою на сьогоднішній день є реляційна структура [19].

Вхідні дані – це набір даних потрібних для того щоб система коректно працювала. Аналіз вимог вхідних даних полягає в визначенні потреб та умов, які висуваються щодо нового, чи зміненого матеріалу, враховуючи можливо конфліктні вимоги різних замовників.

На основі дослідження предметної області був проведений аналіз, в результаті якого була розроблена схема даних в додатку MySQL Workbench 6.3 CE (рис 2.16).

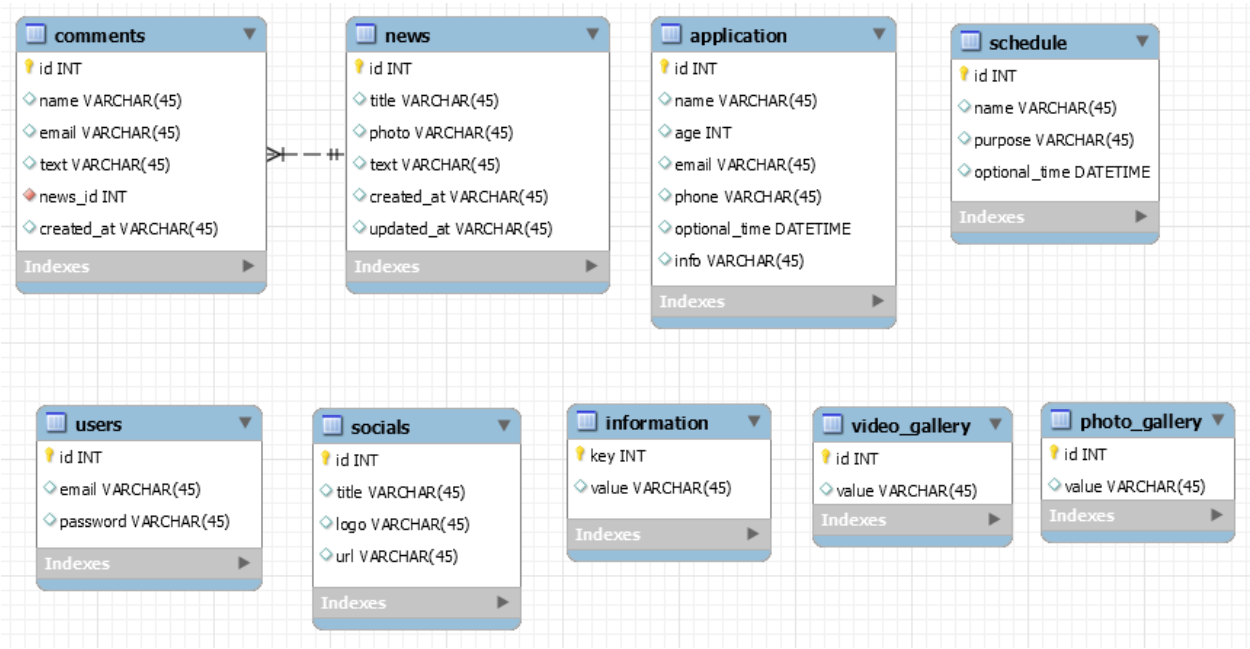

Рисунок 2.16 – Схема даних, використовуваних у веб-додатку

На основі схеми даних було згенеровано базу даних в додатку MySQL, що складається з дев’яти таблиць.

В таблиці «news» (рис. 2.17) містяться записи новин, що публікуються на сайті. Таблиця містить чотири поля в яких зберігається первинний ключ, заголовок новини, інформація, фото, дати створення і оновлення інформації.

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно	Действие
1	id	bigint(20)		UNSIGNED	Нет	Нем		AUTO_INCREMENT	
2	title	varchar(255)	utf8mb4_unicode_ci		Нет	Нем			
3	photo	varchar(255)	utf8mb4_unicode_ci		Да	NULL			
4	text	text	utf8mb4_unicode_ci		Нет	Нем			
5	created_at	timestamp			Да	NULL			
6	updated_at	timestamp			Да	NULL			

Рисунок 2.17 – Таблиця «news»

В таблиці «comments» (рис. 2.18) міститься інформація про коментарі залишені користувачами про новину. Таблиця містить 6 полів в яких зберігається первинний ключ, ім’я користувача, email, текст коментаря, ідентифікатор товару, дату створення.

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно	Действие
1	id	bigint(20)		UNSIGNED	Нет	Нем		AUTO_INCREMENT	
2	name	varchar(255)	utf8mb4_unicode_ci		Нет	Нем			
3	email	varchar(255)	utf8mb4_unicode_ci		Нет	Нем			
4	text	text	utf8mb4_unicode_ci		Нет	Нем			
5	news_id	bigint(20)		UNSIGNED	Нет	Нем			
6	created_at	timestamp			Да	NULL			

Рисунок 2.18 – Таблиця «comments»

Всі заявки на тренування залишені користувачами сайту зберігаються в таблиці «application», яка містить контактні дані користувача, а також додаткову інформацію. Таблиця містить 7 полів в яких зберігається первинний ключ, ім’я користувача, вік, email, контактний телефон, бажаний час проведення заняття, додаткова інформація яка є не обов’язковою.

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно	Действие
1	id	bigint(20)		UNSIGNED	Нет	Нем		AUTO_INCREMENT	
2	name	varchar(255)	utf8mb4_unicode_ci		Нет	Нем			
3	age	int(11)			Нет	Нем			
4	email	varchar(255)	utf8mb4_unicode_ci		Нет	Нем			
5	phone	varchar(255)	utf8mb4_unicode_ci		Нет	Нем			
6	optional_time	varchar(255)	utf8mb4_unicode_ci		Нет	Нем			
7	info	text	utf8mb4_unicode_ci		Да	NULL			

Рисунок 2.19 – Таблиця «application»

Таблиця «schedule» призначена для зберігання розкладу занять. Вона містить чотири поля в яких зберігається первинний ключ, ім’я учасника, мета проведення заняття і назначений час.

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно	Действие
1	id	bigint(20)		UNSIGNED	Нет	Нем		AUTO_INCREMENT	
2	name	varchar(255)	utf8mb4_unicode_ci		Нет	Нем			
3	purpose	text	utf8mb4_unicode_ci		Нет	Нем			
4	optional_time	varchar(255)	utf8mb4_unicode_ci		Нет	Нем			

Рисунок 2.20 – Таблиця «schedule»

В таблиці «users» (рис. 2.21) міститься інформація про користувачів зареєстрованих адміністраторів сайту. Таблиця містить чотири поля в яких зберігається первинний ключ, email, пароль, токен для збереження сесії авторизації.

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно	Действие
1	id	bigint(20)		UNSIGNED	Нет	Нем		AUTO_INCREMENT	
2	email	varchar(255)	utf8mb4_unicode_ci		Нет	Нем			
3	password	varchar(255)	utf8mb4_unicode_ci		Нет	Нем			
4	remember_token	varchar(100)	utf8mb4_unicode_ci		Да	NULL			

Рисунок 2.21 – Таблиця «users»

В таблиці «socials» (рис. 2.22) містяться посилання на соціальні мережі. Таблиця містить чотири поля в яких зберігається первинний ключ, назва соціальної мережі, посилання та логотип.

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно	Действие
1	id	bigint(20)		UNSIGNED	Нет	Нем		AUTO_INCREMENT	
2	title	varchar(255)	utf8mb4_unicode_ci		Нет	Нем			
3	url	varchar(255)	utf8mb4_unicode_ci		Нет	Нем			
4	logo	varchar(255)	utf8mb4_unicode_ci		Нет	Нем			

Рисунок 2.22 – Таблиця «socials»

Фото та відео матеріали, що публікуються на сайті зберігаються у відповідних таблицях «photo_gallery» та «video_gallery». Кожна з який складається з двох полів в яких міститься первинний ключ та посилання на файл. Приклад однієї з таблиць наведено на рис. 2.23.

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно	Действие
1	id	bigint(20)		UNSIGNED	Нет	Нем		AUTO_INCREMENT	
2	value	varchar(255)	utf8mb4_unicode_ci		Нет	Нем			

Рисунок 2.23 – Таблиця «photo_gallery»

Також було створено додаткову таблицю «information» для збереження загальної інформації, що відноситься до сайту, а саме опис сайту, email, контактний телефон та адресу (рис. 2.24).

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно	Действие
1	key	varchar(255)	utf8mb4_unicode_ci		Нет	Нем			
2	value	text	utf8mb4_unicode_ci		Нет	Нем			

Рисунок 2.24 – Таблиця «information»

3 ПРОГРАМНА РЕАЛІЗАЦІЯ

3.1 Архітектура веб додатку

Для побудови проекту буде використано фреймворк laravel, який дозволяє створювати веб-додатки на основі архітектури MVC. Архитектуру веб-додатку наведено на рис.3.1.

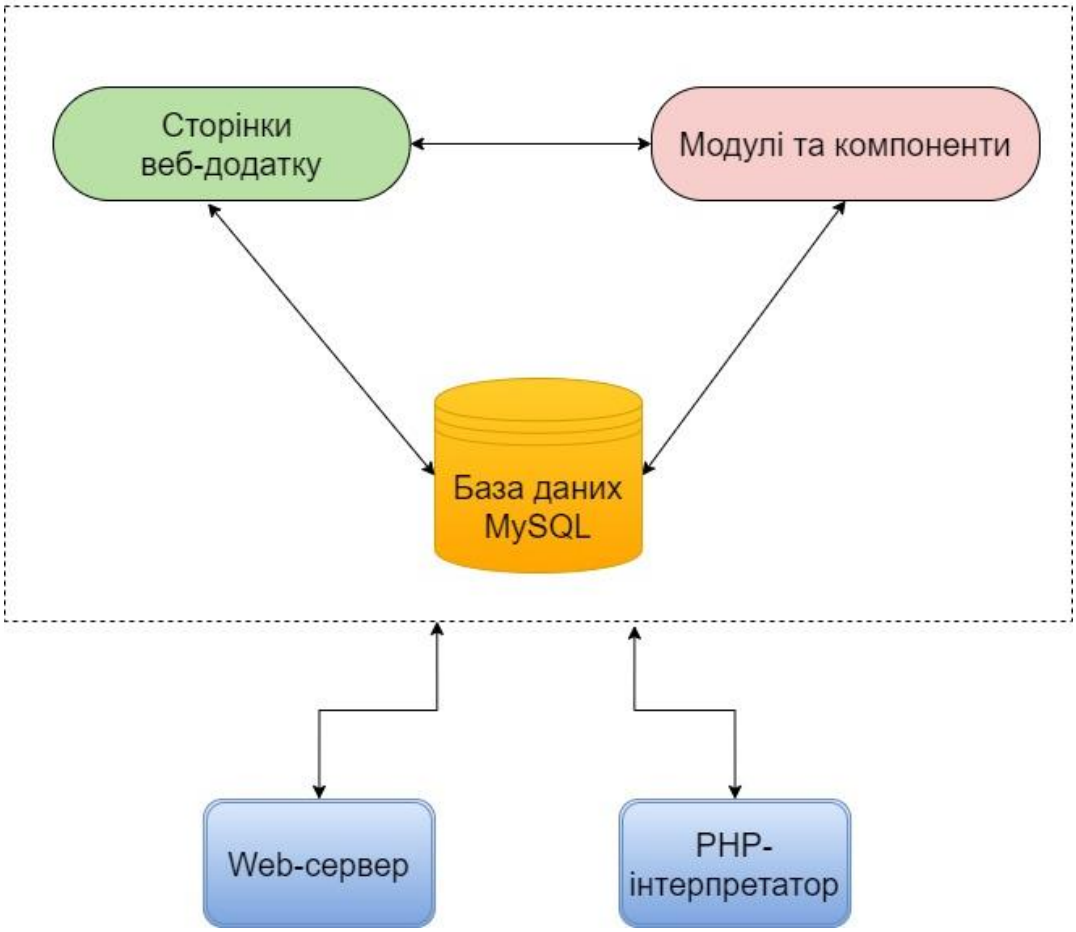

Рисунок 3.1 – Архітектура веб-додатку

3.2 Програмна реалізація

Сама розробка буде відбуватись у безкоштовному текстовому редакторі VS Code. Visual Studio Code – це крос-платформенний редактор скриптів, створений корпорацією Майкрософт. Поряд з розширенням

PowerShell він надає широкі інтерактивні можливості редагування скриптів, спрощуючи написання надійних скриптів PowerShell.

Також для збереження вихідного коду, а також відслідковування версій сайту було використано систему контролю версій Git. Системи контролю версій дозволяють розробникам зберігати всі зміни, внесені в код [20]. Тому в разі, помилки, вони можуть просто відкотити код до робочого стану замість того, щоб витрачати години на пошуки маленької помилки або помилок, які ламають весь код.

Системи контролю версій також дають можливість декільком розробникам працювати над одним проектом і зберігати внесені зміни, щоб переконатися, що всі можуть стежити за тим, над чим вони працюють.

Проект буде розроблятись на віртуальному веб сервері OpenServer. Open Server Panel – це портативна серверна платформа і програмне середовище, створена спеціально для веб-розробників з урахуванням їх рекомендацій і побажань. Також буде використаний менеджер модулів для PHP – Composer.

Розробка сайту починається з створення нового проекту з використанням фреймворку laravel. Для цього в терміналі Composer необхідно написати наступну команду:

```
composer create-project --prefer-dist laravel/laravel football
```

Де «football» – ім'я проекту і може бути змінено на будь-яке. Після чого буде створено базову архітектуру проекту (рис. 3.2).

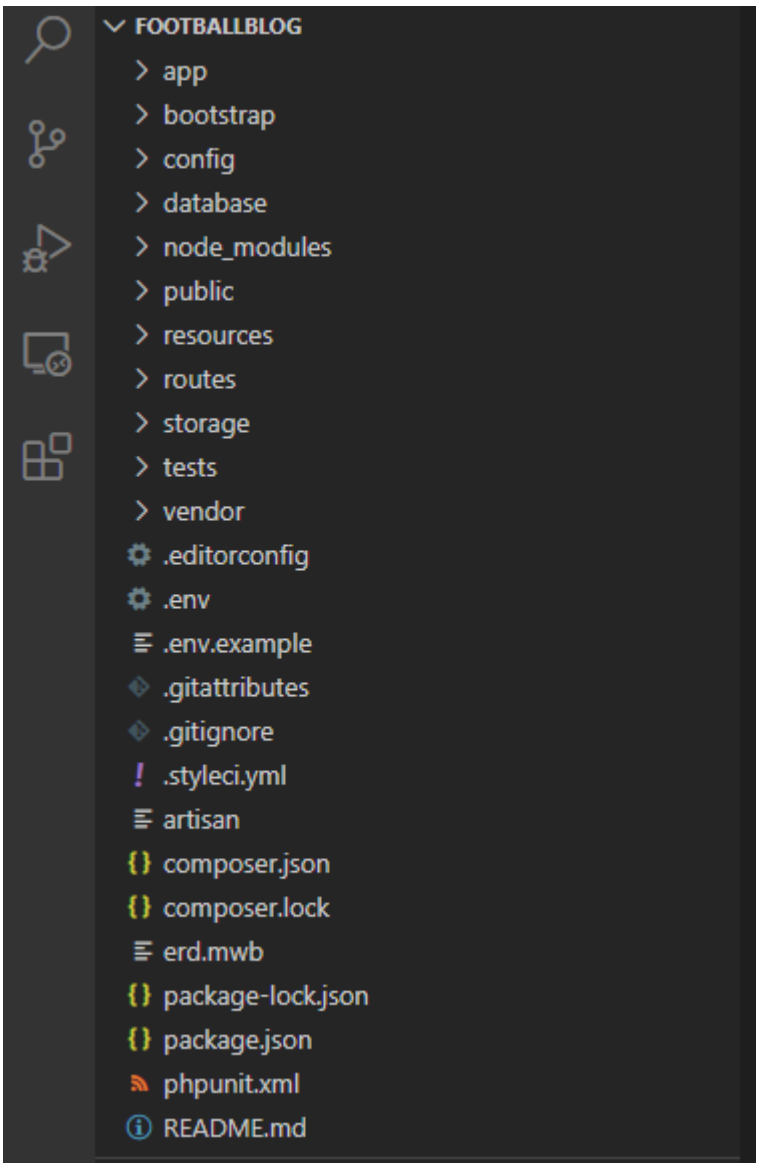

Рисунок 3.2 – Архітектура проекту

Потім починається розробка клієнтської частини сайту, а саме створення всіх необхідних компонентів, налаштування маршрутів доступу і верстка шаблонів та розробка модулів серверної частини сайту, створення всіх необхідних контролерів, моделей та налаштування API роутеру.

3.2.1 Розробка клієнтської частини

Написання сайту починається із створення «скелету» з використанням мови розмітки HTML. Після чого за допомогою мови каскадних стилів CSS формується зовнішній вигляд веб-сайту і надається візуальна форма елементів для зручного користування.

Потім створений шаблон розбивається на окремі компоненти, що надає можливість багаторазово використовувати один компонент на різних сторінках сайту. Приклад компоненту головної сторінки наведено в додатку В. Всі компоненти знаходяться в папці resources/js/.

Наступним кроком створюється єдина точка входу яка підключає всі необхідні компоненти і модулі JavaScript.

```
import router from "./router";
import Vuetify from "vuetify";

import VueSweetalert2 from 'vue-sweetalert2';
import 'sweetalert2/dist/sweetalert2.min.css';

window.Vue = require('vue');
window.Vue.use(Vuetify);
window.Vue.use(VueSweetalert2);

import AppComponent from "./components/AppComponent";

const app = new Vue({
    el: '#app',
    components: {
        AppComponent
    },
    router
});
```

Створивши всі необхідні шаблони і компоненти. Відбувається налаштування маршрутів для навігації між сторінками сайту. Далі наведено приклад коду маршруту для головної сторінки.

```
export default new Router({
    mode: "history",
```

```
    routes: [
      {
        path: '/',
        name: 'home',
        component: HomeComponent
      }
    ]
});
```

В результаті при кліку на посилання відбувається перехід на необхідний компонент вказаний в маршрутизаторі.

3.2.1 Розробка серверної частини

Розробка серверної частини починається зі створення та підключення бази даних. Для цього необхідно перейти до налаштувань підключення до бази даних. Необхідно змінити файл (.env), він знаходиться в кореневій папці проекту. Необхідно змінити 4 рядки: DB_HOST = localhost, DB_DATABASE = football, DB_USERNAME = root, DB_PASSWORD = secret. Де DB_HOST – ім'я хоста, DB_DATABASE – ім'я бази даних, DB_USERNAME – ім'я користувача для доступу до бази даних, DB_PASSWORD – пароль користувача для доступу до бази даних.

Потім створюються міграції для того щоб автоматично було створено базу даних. Міграції – це система контролю версій для бази даних. Вони дозволяють команді програмістів змінювати структуру БД, в той же час залишаючись в курсі змін інших учасників. Далі наведено приклад міграції для створення таблиці користувачів.

```
    public function up()
    {
        Schema::create('users', function (Blueprint $table) {
            $table->id();
            $table->string('email')->unique();
```

```
            $table->string('password');
            $table->rememberToken();
            $table->timestamps();
        });
    }
```

Після створення всіх необхідних міграцій потрібно написати в термінали Composer наступну команду:

```
php artisan:migrate
```

Далі створюються моделі для зручної роботи з базою даних, налаштувань залежностей між таблицями і підключення додаткових модулів. Далі наведено приклад моделі користувачів.

```
class User extends Authenticatable
{
    use HasApiTokens,Notifiable;

    protected $fillable = [
        'email', 'password',
    ];

    protected $hidden = [
        'password', 'remember_token',
    ];
}
```

Всі запити що надходять на сервер обробляються API маршрутизатором, що знаходиться в файлі routes/api.php. Маршрутизація визначає, як додаток відповідає на клієнтський запит до конкретної адреси.

Потім в залежності від типу запиту підключається необхідний контролер з необхідним методом який опрацює дані отримані з бази даних.

Контролер – з'єднує моделі, види і інші компоненти необхідні для роботи системи. А також відповідає за обробку запитів користувача, а саме отримання і обробку даних та відправку готового результату клієнтській частині системи. Приклад такого контролеру для отримання списку всіх новин наведено нижче:

```
class NewsController extends Controller
{
   function get() {
      $data = News::get();
      return response()->json($data);
   }
}
```

Детальнішу інформацію можна знайти у Додатку В.

3.3 Використання програмного додатку

Даний дипломний проект, а точніше веб-сайт, що розроблений для спортивної секції по футболу, складається з двох частин. А саме головна сторінка для перегляду інформації на сайті, вона доступна всім користувачам та адміністративної панелі з можливістю додання і редагування інформації яка доступна лише для адміністраторів сайту.

3.3.1 Головна сторінка сайту

Головна сторінка веб-сайту складається з шапки де знаходиться назва та посилання на соціальні мережі, навігаційного меню, слайд шоу, блоку з інформацією про секцію, блоку з останніми новинами спорту та розкладом занять (рис. 3.2).

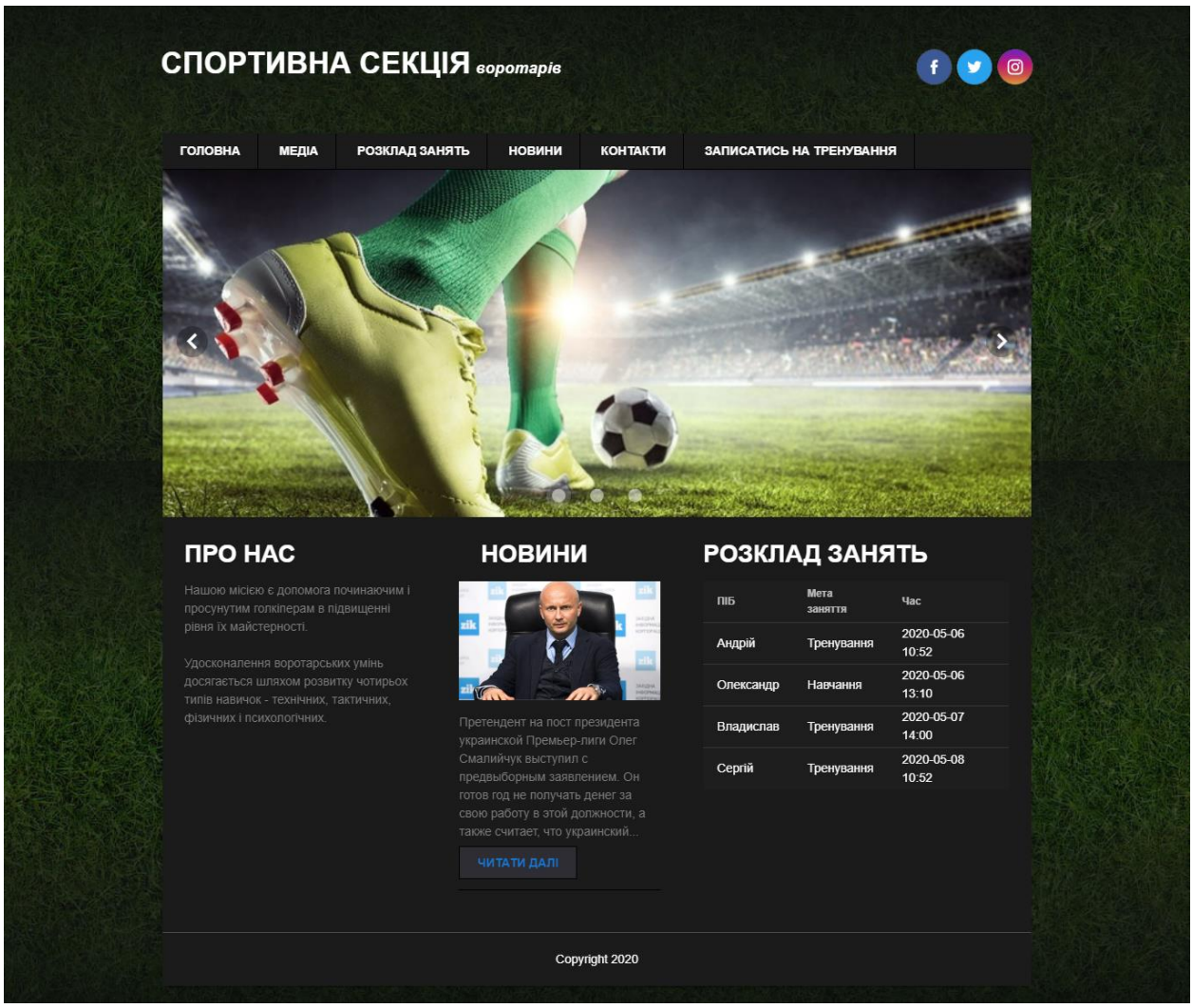

Рисунок 3.2 – Головна сторінка сайту

Пункт головного меню містить в собі посилання на головну сторінку, розкладу занять, новини спорту, контактні дані, форму для запису на тренування, при наведенні на пункт «Медіа» відображаються посилання на фото-галерею та відео-галерею (рис 3.3).

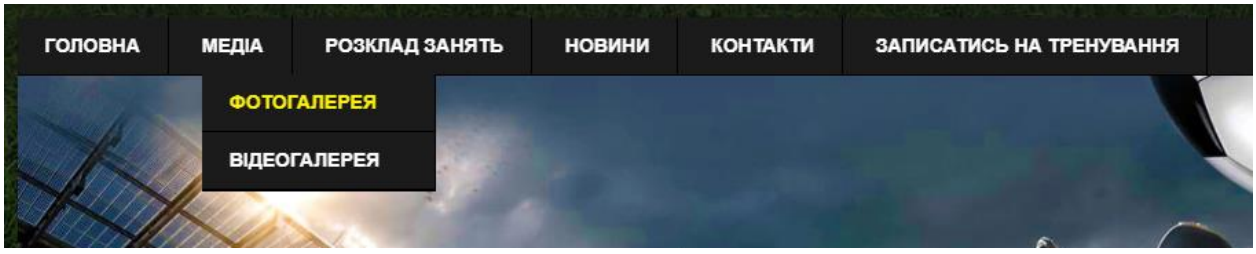

Рисунок 3.3 – Головне меню

Сторінка розладу занять містить таблицю з розладом проведення спортивних тренувань в секції. Вона мастить ім’я та прізвище учасника, мету тренування та назначений час.

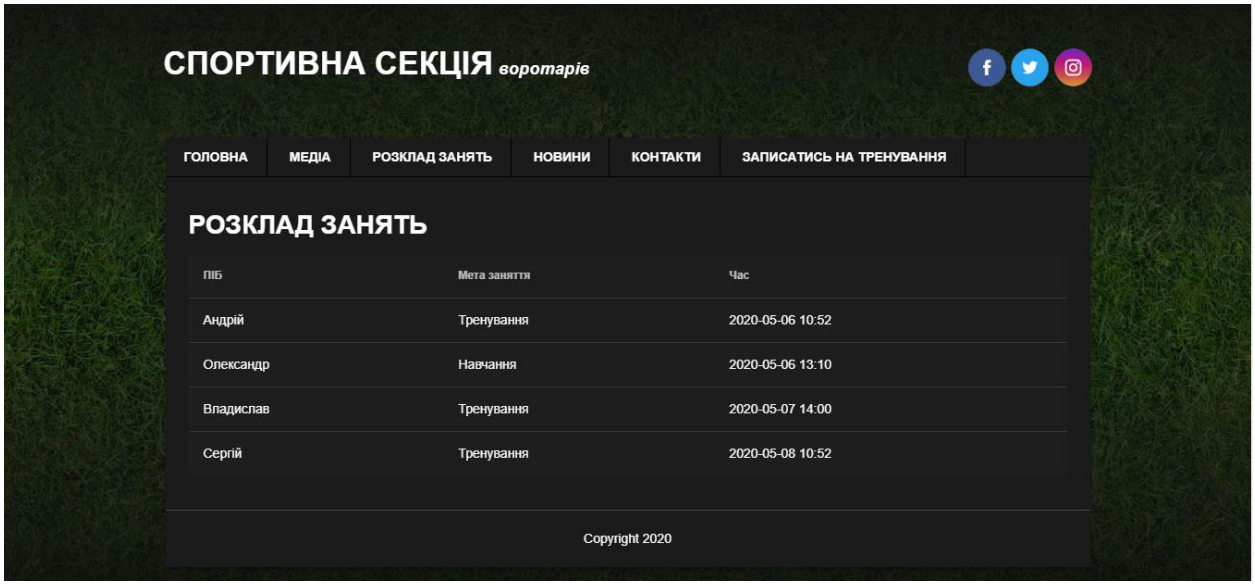

Рисунок 3.4 – Сторінка розкладу занять

Сторінка «Новини» містить останні новини спорту та футболу (рис. 3.5). Кожна картка з новиною містить зображення, заголовок, дату публікації та короткий текст новини. Також можна відкрити детальну інформацію натиснувши кнопку «Читати далі». Після чого буде відкрито сторінку з детальною інформацією та формою для залишання коментарів (рис. 3.6).

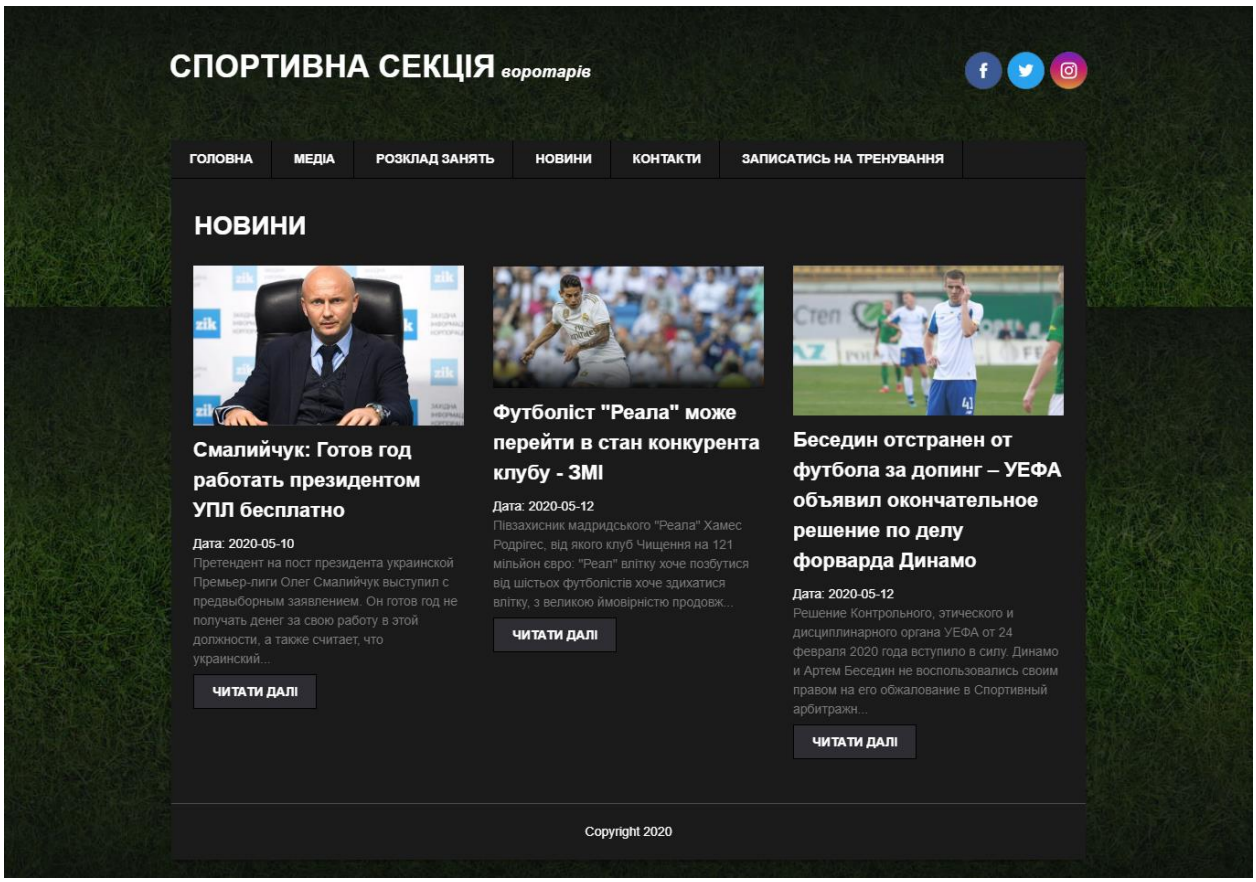

Рисунок 3.5 – Сторінка новин

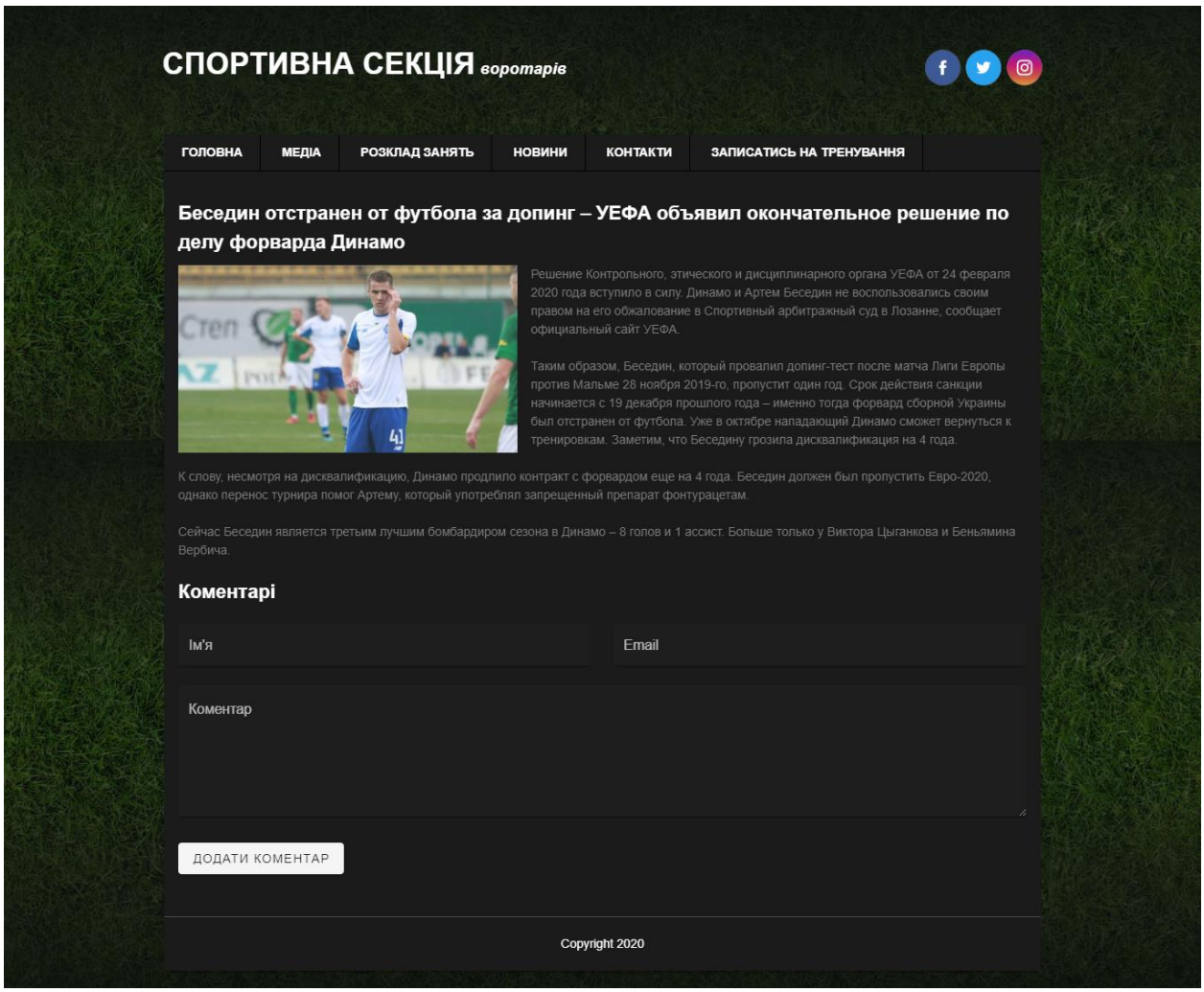

Рисунок 3.6 – Сторінка з новиною

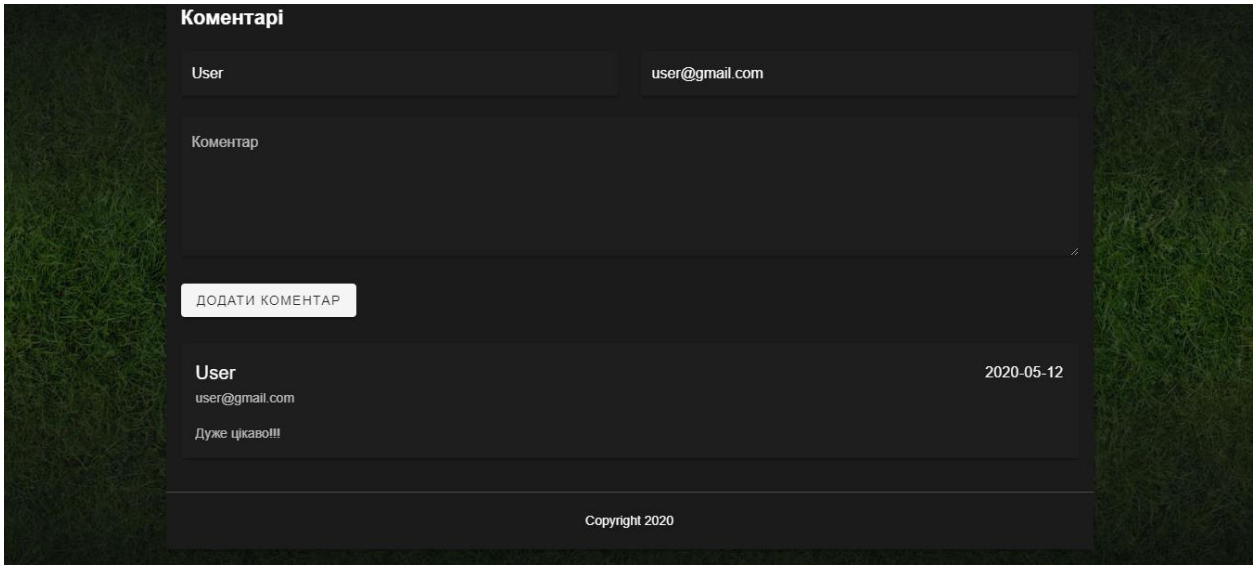

Рисунок 3.7 – Коментарі

Сторінка «Контакти» складається з двох блоків, а саме блоку контактної інформації та мапи з вказаним адресом секції.

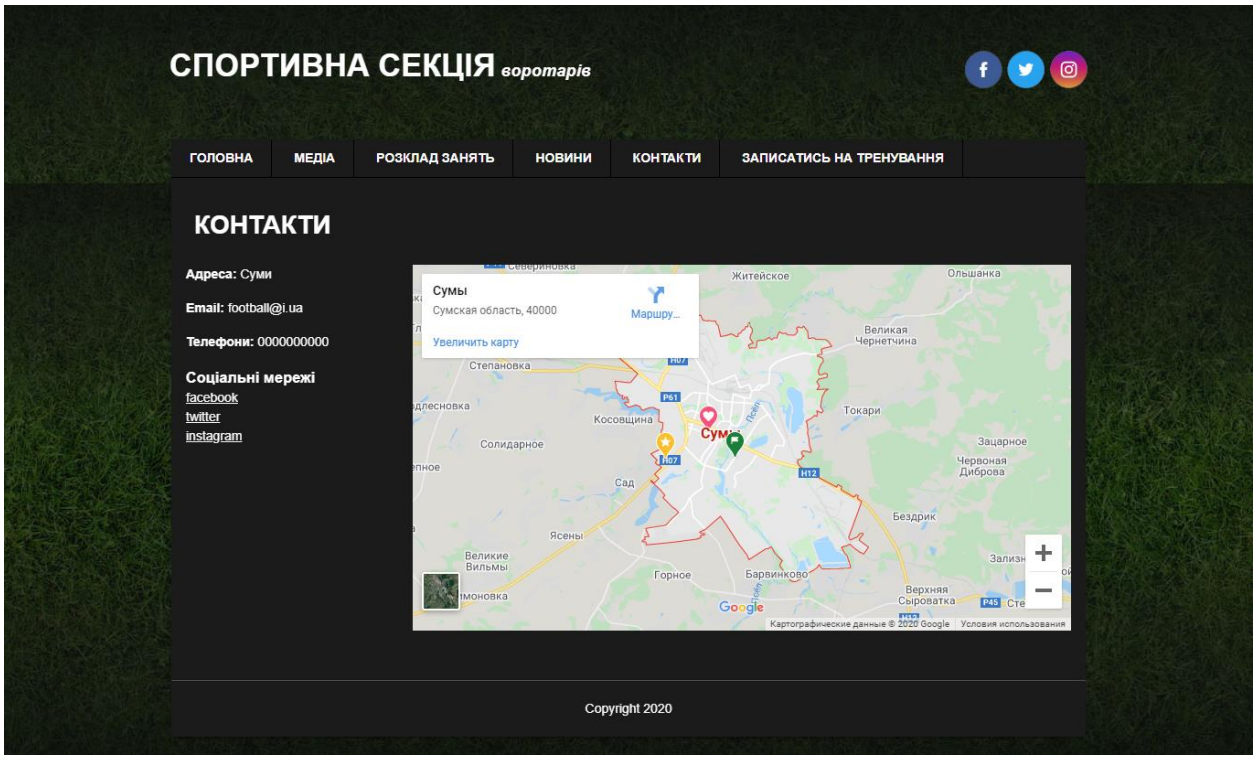

Рисунок 3.8 – Сторінка контактів

Для реєстрації учасників на тренування було створено сторінку «Записатись на тренування» (рис. 3.9). Вона містить форму з полями для заповнення контактних даних користувача, а саме ім’я, email, телефон, вік учасника, бажаний час проведення тренування і поле для додаткової інформації.

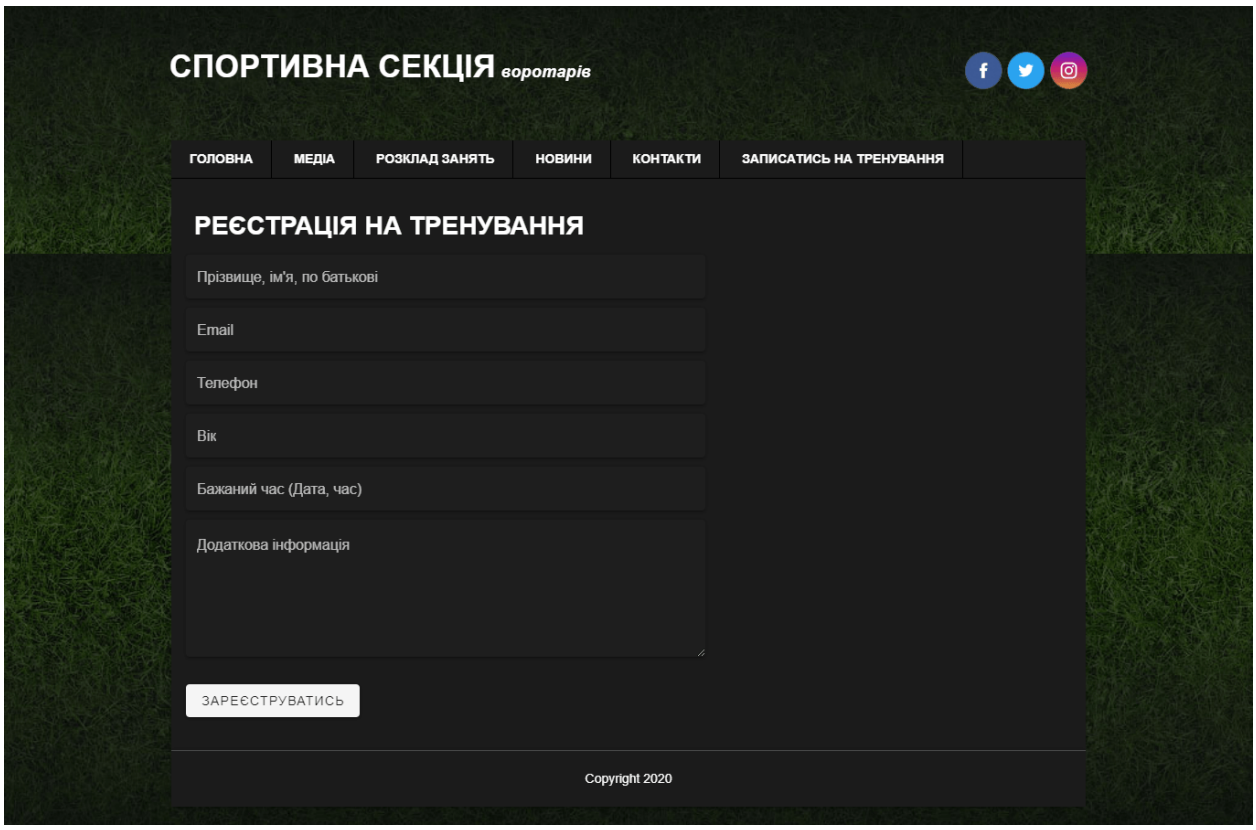

Рисунок 3.9 – Сторінка запису на тренування

Після внесення необхідних даних для реєстрації потрібно натиснути кнопку «Зареєструватись» після чого з’явиться повідомлення про успішну реєстрацію.

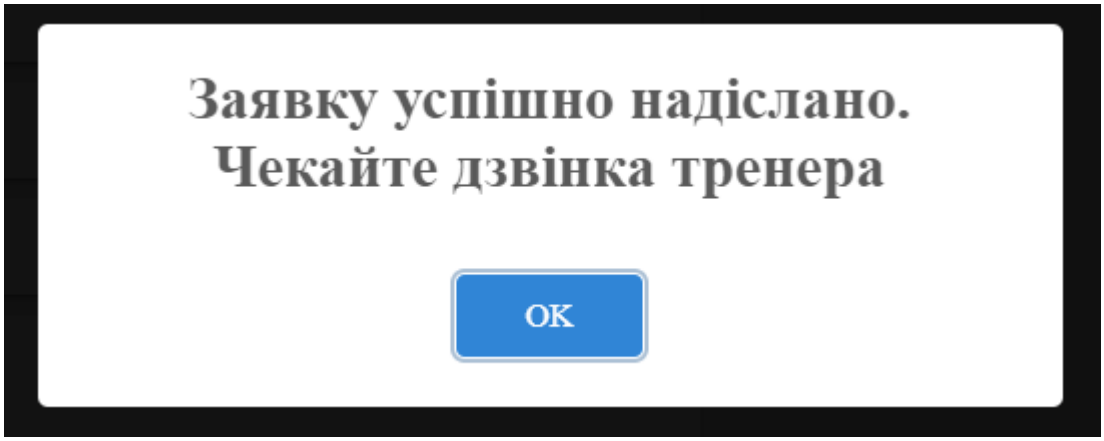

Рисунок 3.10 – Повідомлення про успішну реєстрацію

Сторінки фото-галереї та відео-галереї мають схожу структуру. В них знаходяться зображення, або відео при клікі на які вони збільшяться (рис. 3.11).

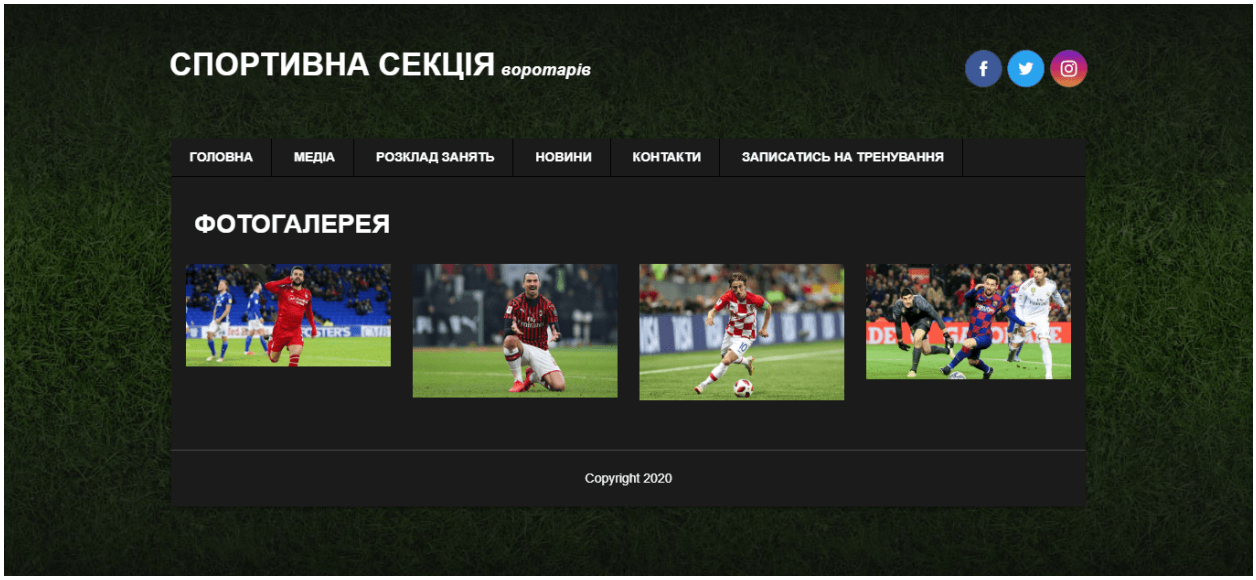

Рисунок 3.11 – Сторінка фото-галереї

3.3.2 Адміністративна панель

Для управління сайтом, додання інформації і обробки заявок на тренування сайт має адміністративну панель. Доступ до неї мають лише адміністратори. Щоб до неї потрапити необхідно авторизуватись. Для авторизації потрібно ввести пошту і пароль користувача у відповідні поля (рис. 3.12).

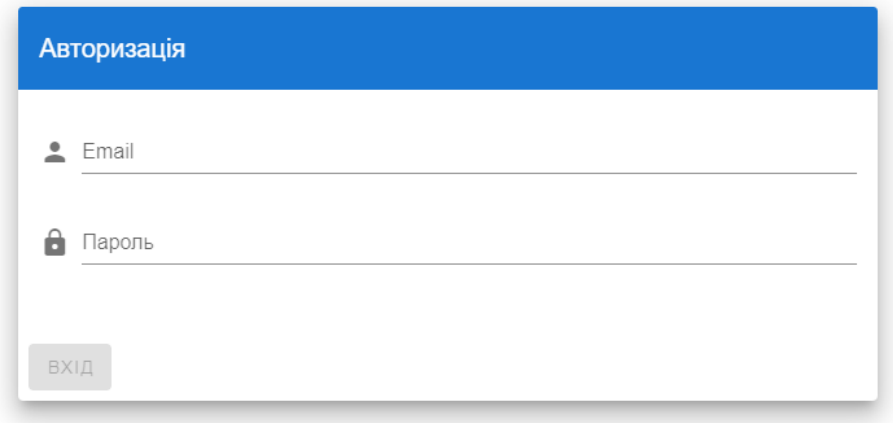

Рисунок 3.12 – Форма авторизації

Після авторизації користувач потравляє на сторінку з загальними налаштуваннями. На якій знаходиться загальна інформація про сайт, а саме опис сайту, адреса, email та телефон. Також є можливість додавати соціальні мережі (рис. 3.13).

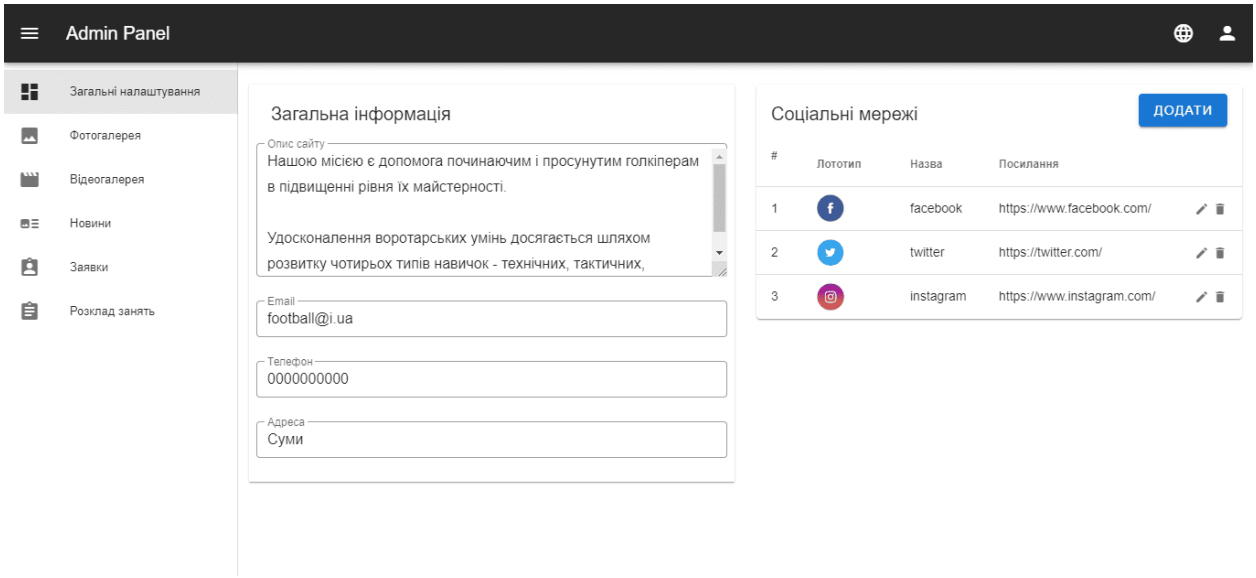

Рисунок 3.13 – Налаштування загальної інформації

На сторінці фото-галереї в адміністративній панелі можна додавати фото. Для цього необхідно натиснути на форму, обрати бажані зображення на комп’ютері і натиснути «Додати». Після чого обрані фото з’являться під формою додання. Також за необхідністю зображення можна видалити натиснувши на корзину під кожним зображенням (рис. 3.14).

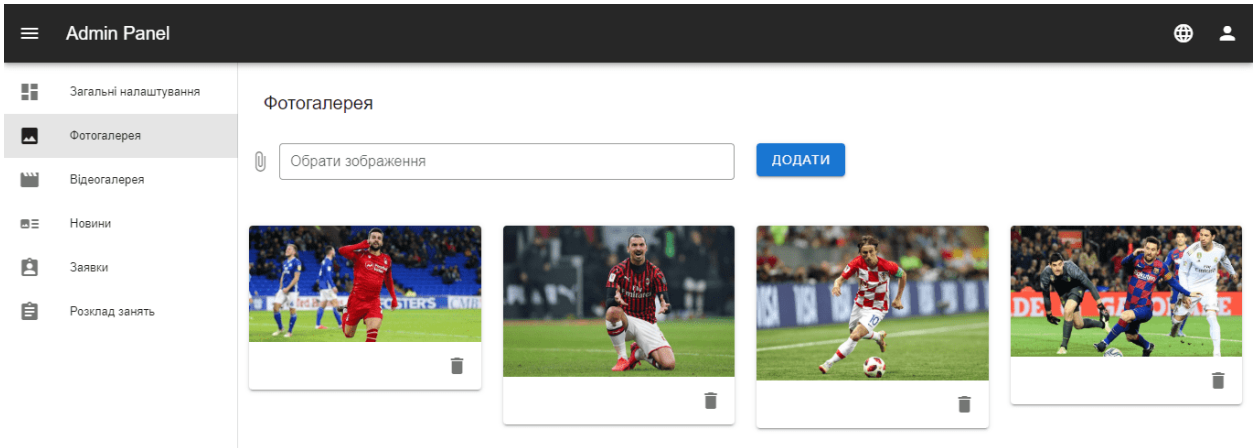

Рисунок 3.14 – Фото-галерея

Сторінка відео-галерея подібна сторінці фото-галереї адміністративної панелі. Проте вона має певні відмінності. Для додання відео необхідно вставити в форму посилання на відео з відео хостингу YouTube. Потім натиснути кнопку «Додати» після чого відео з'явиться під формою додання. Також за необхідністю зображення можна видалити натиснувши на корзину під кожним відео (рис. 3.15).

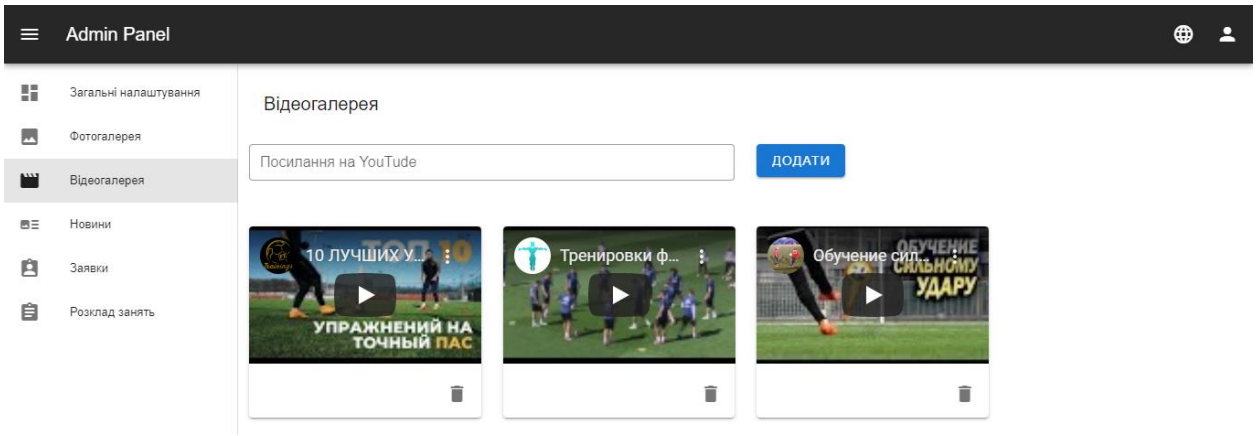

Рисунок 3.15 – Відео-галерея

Для додання новин на сайт необхідно перейти на сторінку «Новини» в адміністративній панелі. На ній знаходиться таблиця зі списком новин, кожну з яких можна видалити або відредагувати. Також є кнопка для додання новини (рис. 3.16).

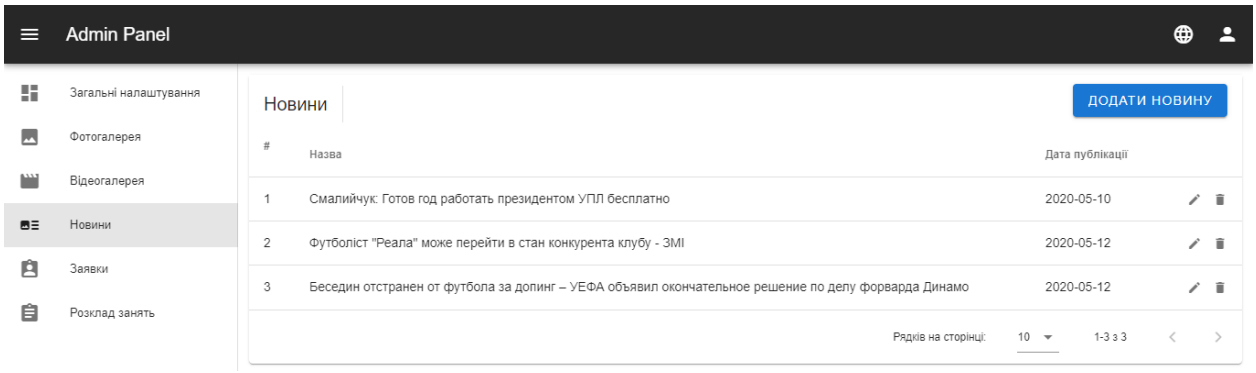

Рисунок 3.16 – Новини

При переході на сторінку додання новини з’являється форма з полями для заповнення назви новини, тексту та зображення (рис. 3.17).

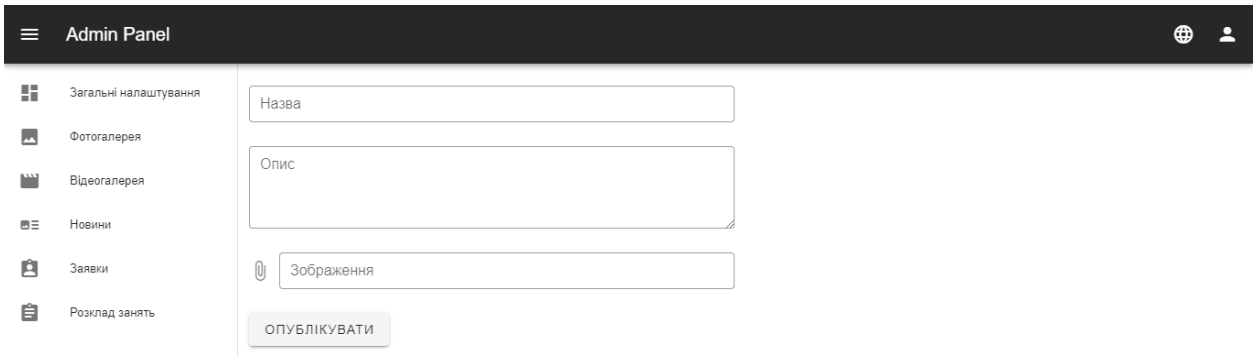

Рисунок 3.17 – Форма додання новини

Після успішної публікації новини з’явиться відповідне повідомлення (рис. 3.18).

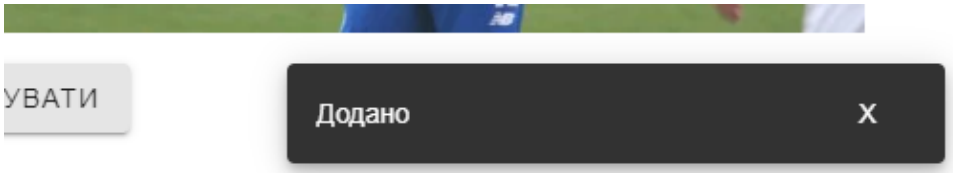

Рисунок 3.18 – Повідомлення про успішну публікацію новини

Сторінка «Заявки» містить в собі таблицю зі списком заявок, що надійшли від користувачів сайту (рис. 3.19).

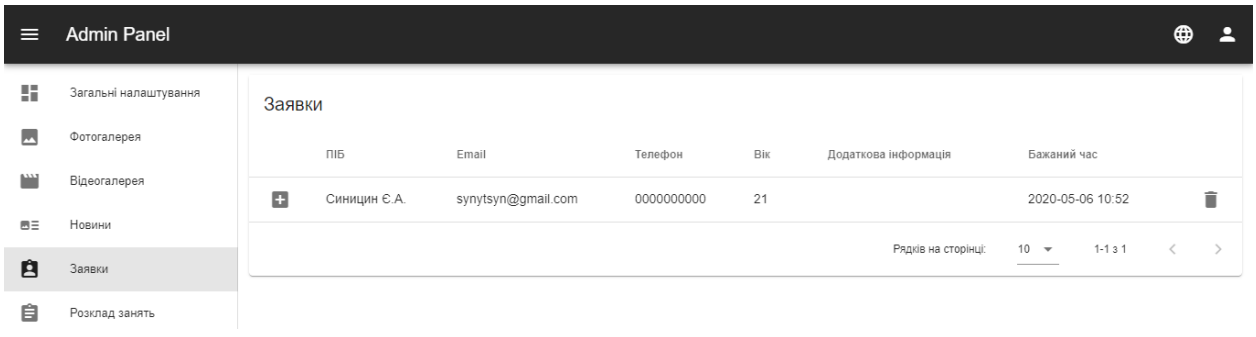

Рисунок 3.19 – Сторінка з заявками

Щоб підтвердити заявку необхідно натиснути кнопку «+», після чого з’явиться модальне вікно з детальною інформацією про учасника і формою для написання мети заняття (рис. 3.20).

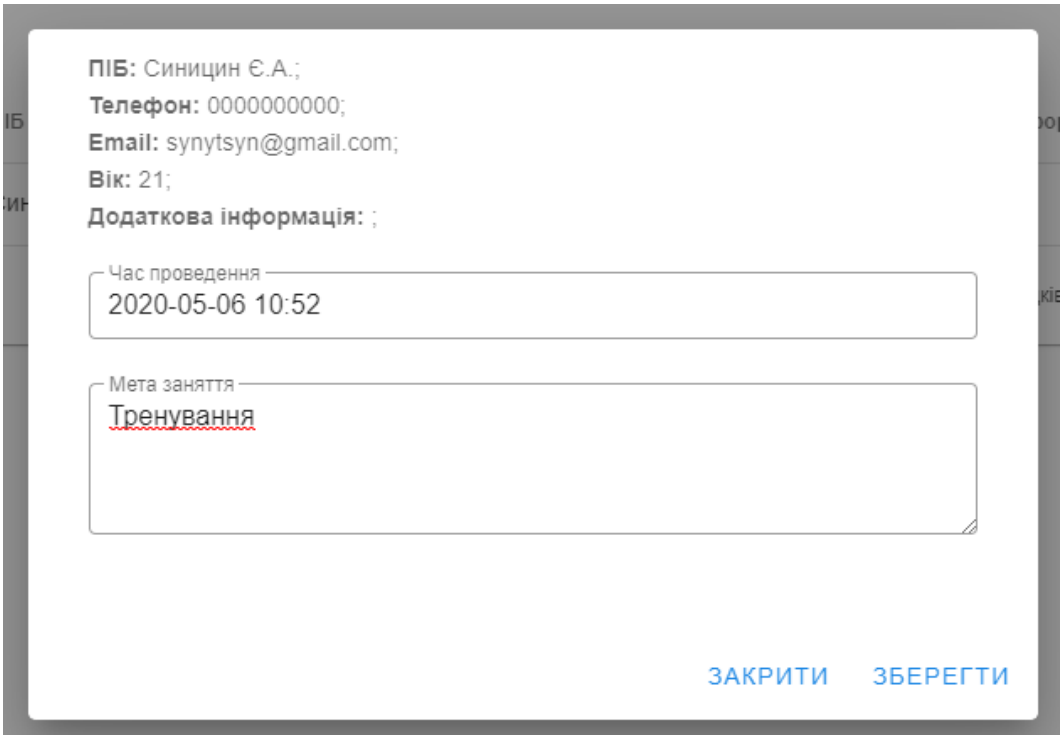

Рисунок 3.20 – Вікно з даними учасника

Написавши мети заняття потрібно затиснути кнопку «Зберегти». Після чого вікно буде закрито, дані учасника будуть додані до таблиці розкладу занять. А в таблиці заявок вона буде видалена. За бажанням заявку можна видалити вручну натиснувши на значок корзини справа таблиці.

Сторінка розкладу занять також має таблицю. Вона містить в собі інформацію про кожне з занять. При необхідності їх можна видалити або відредагувати натиснувши відповідні кнопки справа в таблиці (рис. 3.21).

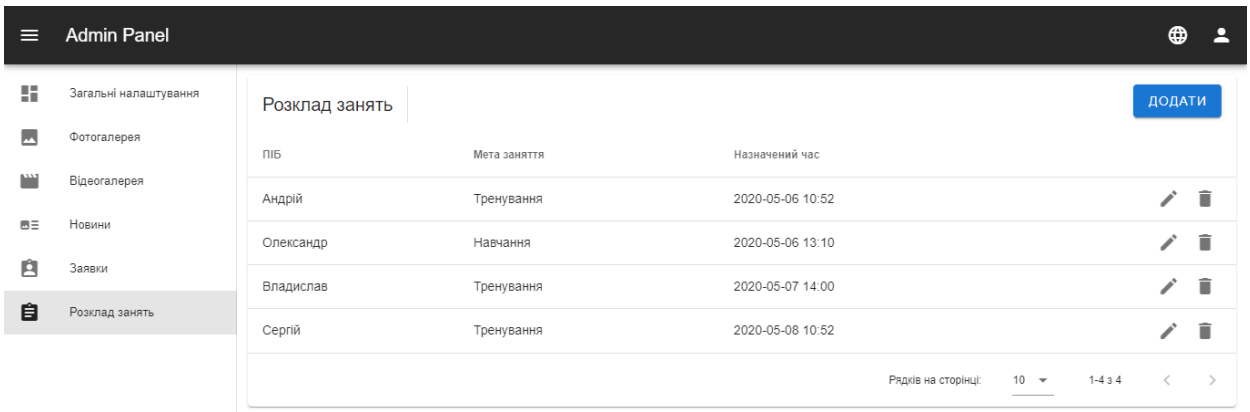

Рисунок 3.21 – Розклад занять

Також при необхідності можна додати заняття натиснувши кнопку «Додати» (рис. 3.22). Після чого з’явиться модальне вікно з формою для заповнення інформації про заняття.

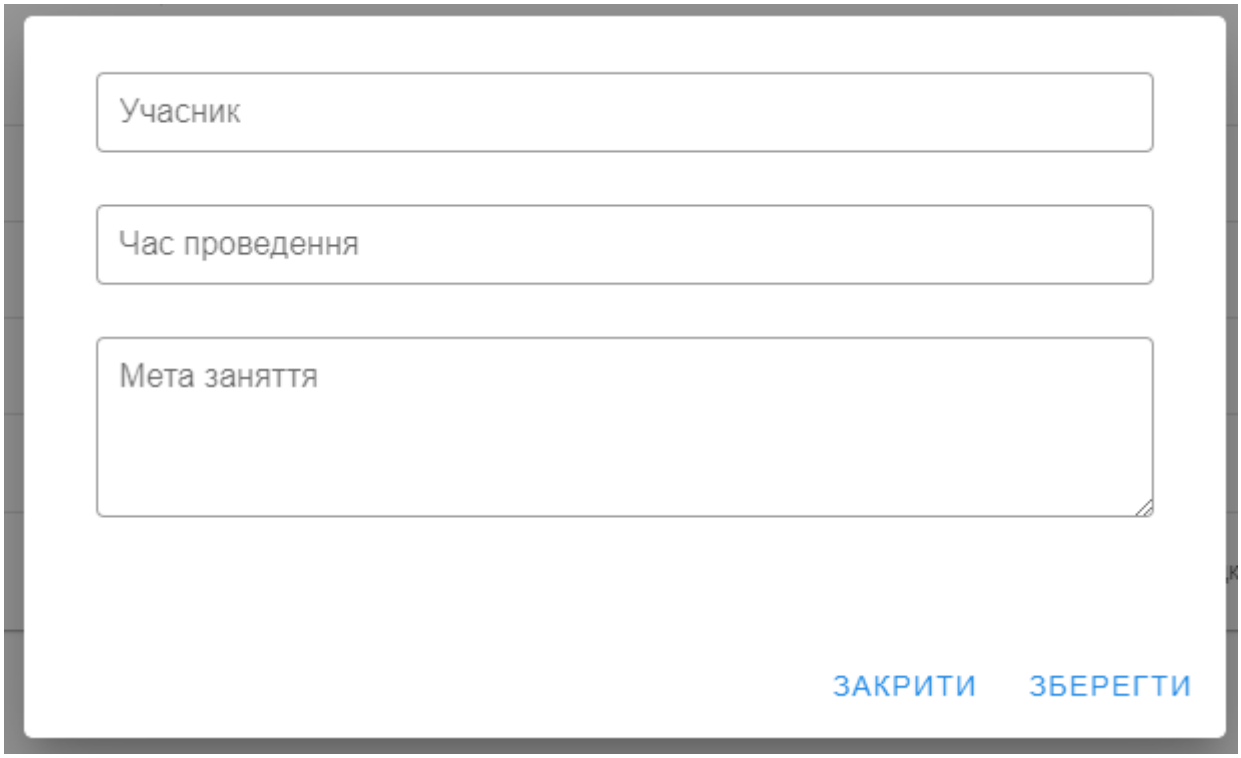

Рисунок 3.22 – Форма додання занять

ВИСНОВКИ

У дипломній роботі вирішена актуальна проблема забезпечення спортивної секції по футболу засобом інформування її учасників, як наявних, так і потенційних, щодо організації роботи секції. Мета дипломного проекту розробка веб сайту для спортивної секції по футболу. Мета досягнута, поставлені завдання виконані:

1. Досліджена предметна область.
2. Проведено аналіз аналогів.
3. Обрана стратегію розробки.
4. Створення моделі процесу роботи секції.
5. Розробка прототипу сайту.
6. Програмна реалізація.

В процесі роботи досліджена предметна область. Оптимізовано web-інтерфейс і навігація сайту, для того щоб користувачеві було зручніше орієнтуватися в віртуальному просторі.

Проект реалізований з використанням фреймворку Laravel, мов програмування PHP, JavaScript використавши бібліотеку Vue.js.

Всі цілі і завдання були виконані.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Що таке сайт? Інтернет сайт? Види сайтів [Електронний ресурс] – Режим доступу до ресурсу: http://moolkin.ru/chto-takoe-sayt-internet-sayt-vidy-saytov/. (дата звернення: 02.05.2020).
2. The 2020 Roadmap To Fullstack Web Development [Електронний ресурс] – Режим доступу до ресурсу: https://codingthesmartway.com/the-2020-roadmap-to-fullstack-web-development/. (дата звернення: 22.04.2020).
3. How To Become A Web Developer — Everything You Need To Know [Електронний ресурс] – Режим доступу до ресурсу: https://careerfoundry.com/en/blog/web-development/what-does-it-take-to-become-a-web-developer-everything-you-need-to-know-before-getting-started/. (дата звернення: 22.04.2020).
4. Крокфорд Дуглас. JavaScript. Сильні сторони, 2016.— 176 с.
5. Laravel – Функції безпеки [Електронний ресурс] – Режим доступу до ресурсу: https://webformyself.com/laravel-funkcii-bezopasnosti/. (дата звернення: 24.04.2020).
6. Марк Сафронов: Разработка веб-приложений в Yii 2, 2015. - 392 с.
7. Створення базового додатку MVC Laravel 5 Види сайтів [Електронний ресурс] – Режим доступу до ресурсу: https://selftaughtcoders.com/from-idea-to-launch/lesson-17/laravel-5-mvc-application-in-10-minutes/. (дата звернення: 21.04.2020).
8. Best Personal and Niche Blogs (30+ Real Examples) [Електронний ресурс] – Режим доступу до ресурсу: https://firstsiteguide.com/examples-of-blogs/. (дата звернення: 22.04.2020).
9. The Pros and Cons of 8 Popular Databases [Електронний ресурс] – Режим доступу до ресурсу: https://www.keycdn.com/blog/popular-databases. (дата звернення: 23.04.2020).

10. MySQL Stored Procedure Advantages [Електронний ресурс] – Режим доступу до ресурсу: https://www.tutorialspoint.com/What-are-the-advantages-and-disadvantages-of-using-MySQL-stored-procedures. (дата звернення: 23.04.2020).

11. Діаграми UML [Електронний ресурс] – Режим доступу до ресурсу: https://planerka.info/item/diagrammy-kommunikacij-uml/. (дата звернення: 23.04.2020).

12. Теорія та практика UML [Електронний ресурс] – Режим доступу до ресурсу: http://www.it-gost.ru/articles/view_articles/94. (дата звернення: 23.04.2020).

13. Документація Git [Електронний ресурс] – Режим доступу до ресурсу: https://git-scm.com/. (дата звернення: 24.04.2020).

14. Що таке PHP? [Електронний ресурс] – Режим доступу до ресурсу: https://www.php.net/manual/en/intro-whatis.php. (дата звернення: 22.04.2020).

15. Хавербеке Марейн. Виразний JavaScript. 2 видання, 2015. – 745 с.

16. Анді Гутманс, Стіг Баккен, Дерік Ретханс. Програмування живлення PHP5, 2015. — 704 с.

17. Документація Vue.js [Електронний ресурс] – Режим доступу до ресурсу: https://ru.vuejs.org/v2/guide/. (дата звернення: 23.04.2020).

18. Хеслоп П. HTML самого початку. С.-Пб: Санкт-Петербург, 2014. – 450с.

19. Що таке CSS [Електронний ресурс] – Режим доступу до ресурсу: http://phpist.com.ua/css/5-whatcss/. (дата звернення: 21.04.2020).

20. Документація Laravel [Електронний ресурс] – Режим доступу до ресурсу: https://laravel.com/. (дата звернення: 22.04.2020).

ДОДАТОК А. ТЕХНІЧНЕ ЗАВДАННЯ

ТЕХНІЧНЕ ЗАВДАННЯ

на розробку web-додатоку підтримки роботи спортивної секції "Goalkeeper society"

Суми 2020

1 ПРИЗНАЧЕННЯ Й МЕТА СТВОРЕННЯ ДОДАТКУ

1.1 Призначення веб-додатку

Сайт призначений для того, щоб кожен бажаючий міг побачити себе в грі футбол та розвинути свої навики в якості голкіпера.

1.2 Мета створення веб-додатку

Метою даного сайту є створення веб-додатку, який зможе проінформувати та допомога починаючим та просунутим голкіперам в підвищенні рівня їх майстерності.

1.3 Цільова аудиторія

Цільова аудиторія – це група людей, якій потенційно потрібна дана послуга або наш товар. У всіх членів цієї групи є спільні ознаки.

Цільова аудиторія даного блогу:

- Початківці гри в футбол.
- Професійні гравці гри в футбол.
- Кожен бажаючий стати професійним голкіпером.
- Кожен хто цікавиться спортом.

2 ВИМОГИ ДО ВЕБ-ДОДАТКУ

Перелік головних вимог до сайту:

1. Сайт повинен складатись з двох частин, а саме користувацької та адміністративної.

2. Користувацька частина призначена для не зареєстрованих користувачів які маю можливість користуватись сайтом не виконуючи авторизацію.

Користувацька частина повинна складатися із наступних сторінок:

– Головна.
– Розклад занять.
– Новини футболу.
– Новини школи.
– Фотогалерея.
– Відео-галерея.
– Контакти.
– Заповнити заявку на тренування.

Доступ до адміністративної частини матимуть лише адміністратори сайту. Для них повинен бути розроблений функціонал який надає можливість додавати публікації, змінювати інформацію, переглядати список заявок на тренування.

Адміністративна частина повинна складатися із наступних сторінок:

– Головна.
– Загальна інформація.
– Галерея.
– Новини.
– Заявки користувачів.
– Графік занять.

3 СКЛАД І ЗМІСТ РОБІТ ЗІ СТВОРЕННЯ ДОДАТКУ

Докладний опис етапів роботи наведено в табл. А.3.1.

Таблиця А.3.1 – Етапи створення моделей

№	Склад і зміст робіт	Строк розробки (у робочих днях)
1	Ініціалізація ідей	5
2	Проектування	10
3	Розробка	14
4	Тестування	3
5	Здача проекту	3

ДОДАТОК Б. ПЛАНУВАННЯ РОБІТ

1 ІДЕНТИФІКАЦІЯ ІДЕЇ ПРОЕКТУ

Метою даного проекту є пропаганда здорового способу життя, ознайомлення аудиторії з різними напрямками видів спорту та удосконалення організації роботи спортивних гуртків із використанням інформаційних технологій.

Веб додаток повинен бути реалізований у вигляді сайту, доступного в мережі Інтернет. Сайт повинен складатися із взаємозалежних розділів із чітко розділеними функціями.

2 ДЕТАЛІЗАЦІЯ МЕТИ МЕТОДОМ SMART

Технологія SMART (СМАРТ) – сучасний підхід до постановки працюючих цілей. Система постановки smart – цілей дозволяє на етапі визначення мети узагальнити всю наявну інформацію, встановити прийнятні терміни роботи, визначити достатність ресурсів, надати всім учасникам процесу ясні, точні, конкретні завдання.

SMART є абревіатурою, розшифровка якої: Specific, Measurable, Achievable, Relevant, Time bound. Кожна буква абревіатури SMART означає критерій ефективності поставлених цілей. Розглянемо кожен критерій smart мети більш докладно.

Результати деталізації методом SMART розміщені у табл. Б.2.1.

Таблиця Б.2.1 − Деталізація мети методом SMART

Specific (конкретна)	Створити сайт для спортивної секції з урахуванням поставлених цілей і вимог замовника.
Measurable (вимірювана)	Використовуючи мінімум ресурсів розробити якісний програмний продукт.
Achievable (досяжна)	Реалізація сайту здійснюється за допомогою фреймворку Laravel.
Relevant (реалістична)	У наявності є всі необхідні технічні та програмні засоби. Розробники достатньо кваліфіковані для виконання поставлених задач.
Time-framed (обмежена у часі)	Ціль має часове обмеження. Робота повинна бути виконана у терміни, що були оговорені замовником проекту. Проект повинен бути виконаний згідно з календарним планом.

3 ОПИСАННЯ ФАЗИ РОЗРОБКИ ІТ—ПРОЕКТУ

3.1 Планування змісту структури робіт ІТ—проекту (WBS)

Ієрархічна структура робіт (Work Breakdown Structure) є інструментом, який дозволяє розділити проект на частини. Вона встановлює ієрархічно структурований поділ праці з реалізації проекту для всіх залучених до нього працівників.

Під час побудови WBS відбувається послідовне розбиття проекту на підпроекти, роботи різних рівнів, детальні робочі пакети. Розподіл - це розподіл результатів проекту на менші, простіші для керування компоненти пакетів. Пакети робіт зазвичай відповідають найнижчому рівню деталізації та складаються з окремих робіт. Декомпозиція повинна бути коректною, тобто елементи будь-якого рівня WBS повинні бути необхідними та достатніми для створення відповідного елемента верхнього рівня.

Ієрархічна структура робіт в основному являє собою перелік завдань проекту. Вона може бути представлена графічно або у формі опису, який показує включення робіт. Ієрархічна структура робіт організовує і визначає весь зміст проекту. Роботи, не включені в WBS, не є роботами проекту.

Виконаємо побудову WBS структури, у якій зазначимо всі виконувані роботи в залежності від головних етапів. Діаграма WBS зображена на рис. 3.1.:

1. Формування технічного завдання — розробка технічного завдання, що встановлює основне призначення, показники якості, техніко-економічні та спеціальні вимоги до розроблюваного інструментального засобу. Формування технічного завдання включає в себе підпункти (визначення предметної області, призначення програмного продукту, визначення мови написання. визначення цільової аудиторії, визначення вимог до програмного продукту).

2. Розробка програмного продукту - написання відповідних модулів, що забезпечують функціонування програмного продукту.

2.1 Визначення проблеми.

2.2. Розробка інтерфейсу.

2.3. Розробка стилю засобами CSS.

2.4. Розробка адміністративної частини сайту.

2.5. Наповнення контентом.

3. Тестування – перевірка роботи, виявлення помилок.

WBS-структура для даного проекту представлена на рисунку Б.3.1.

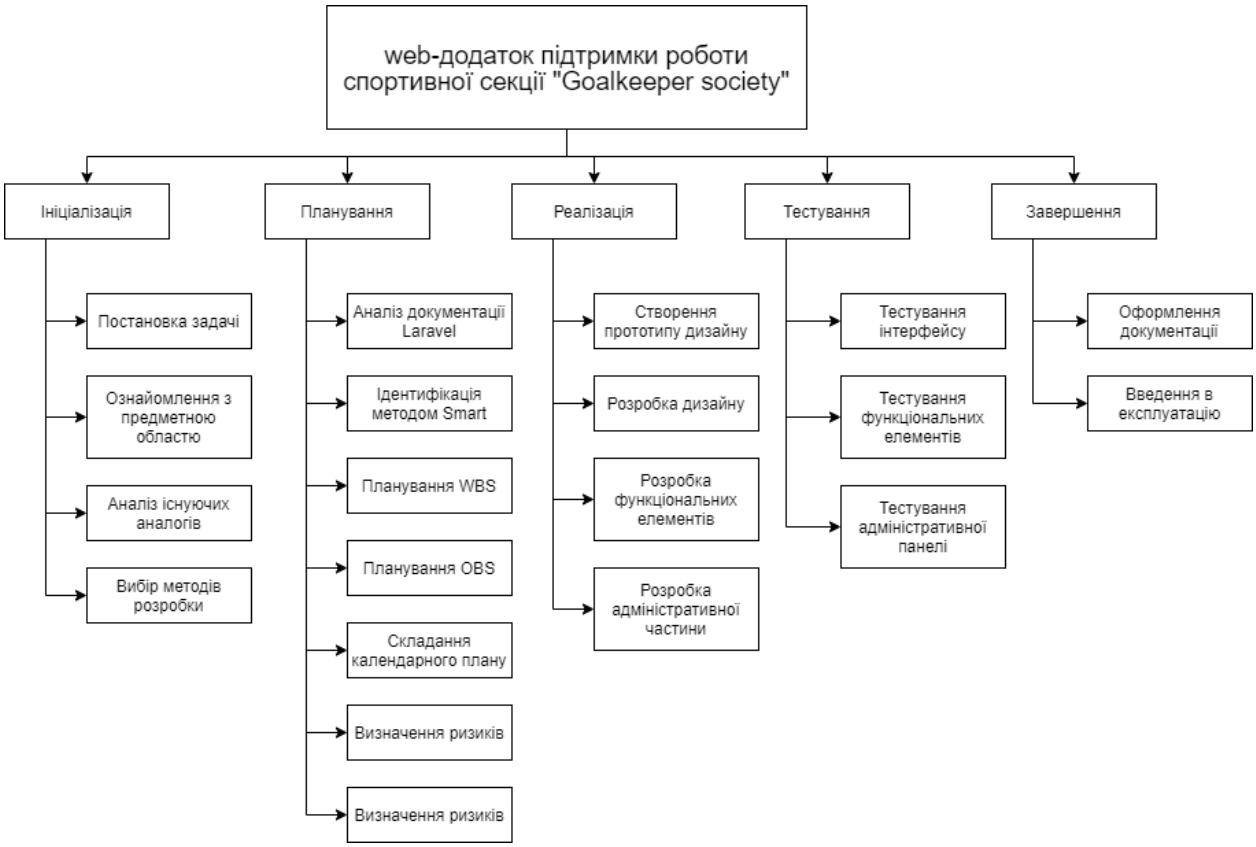

Рисунок Б.3.1 – WBS-структура Веб додатку для замовлення одягу

3.2 Планування структури організації готового проекту

OBS-структура проекту – організаційна структура виконавців (організацій) проекту. Визначається за переліком пакетів робіт нижнього рівня кожної гілки WBS—структури. Представляється відповідальними

(відповідальні – це не обов’язково керівники організацій (відділів), а ті люди які безпосередньо організують виконання робіт) за виконання пакетів робіт.

Організаційна структура представляє собою графічне відображення учасників проекту та їх відповідальних осіб, які задіяні в реалізації проекту. На верхньому рівні OBS розташована команда проекту.

На наступному рівні фіксуються виконавці: організації, відділи тощо. Потім, рівнем нижче, для кожного виконавця вказують прізвища конкретних осіб, які будуть відповідати за виконання елементарних робіт WBS. Потрібно пам’ятати, що відповідальні – це не обов’язково керівники, а ті співробітники, які безпосередньо організовують і відповідають у виконавця за виконання елементарної роботи, зазначеної у WBS. Для них ця елементарна робота також є проектом (у порівнянні з загальним проектом). Для себе вони також можуть побудувати WBS— структуру й застосовувати інші інструменти планування.

Після побудови WBS розробимо організаційну структуру виконавців OBS. Організаційна структура проекту стосується тільки внутрішньої організаційної структури проекту і не стосується відносин проектних груп чи учасників з батьківськими організаціями. Діаграма OBS зображена на рис. Б.3.2.

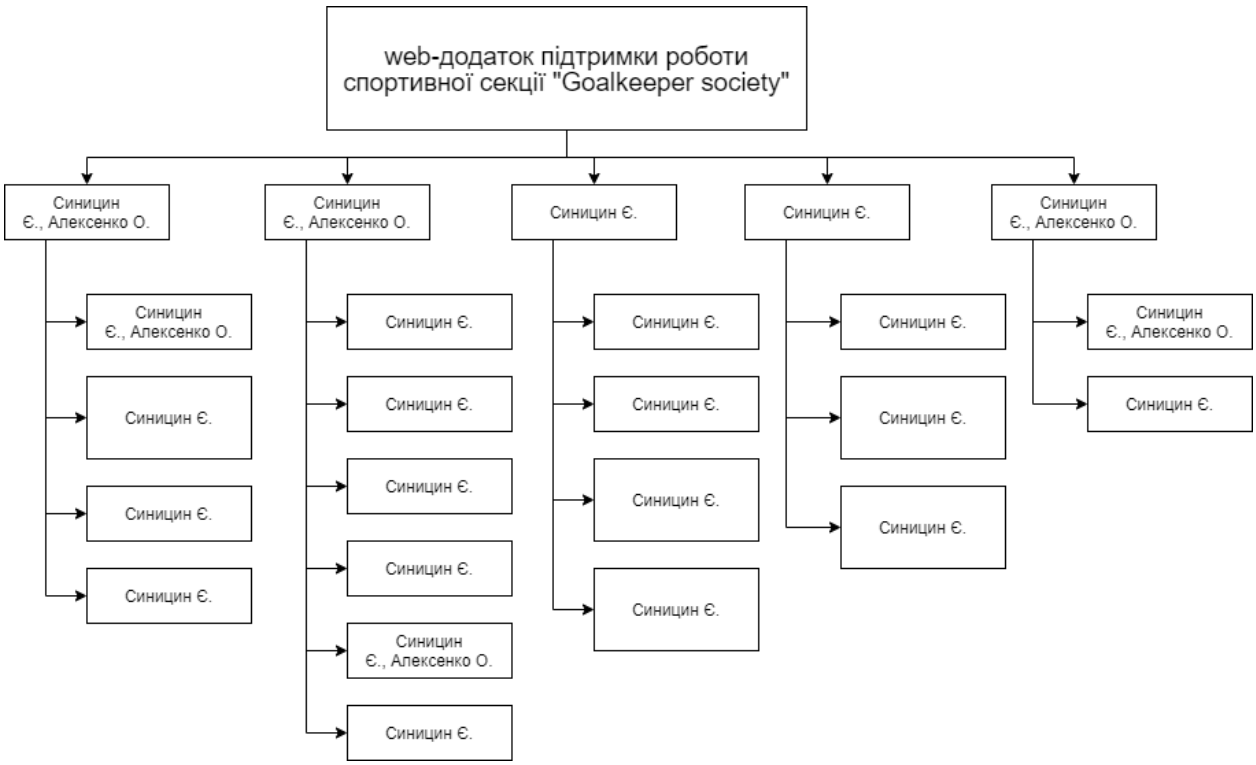

Рисунок Б.3.2 - Організаційна структура виконавців

3.3 Побудова матриці виконавців пакетів робіт

На підставі OBS та WBS структур було створено список виконавців, що функціонують в проекті (табл. Б.3.1).

Таблиця Б.3.1 – Виконавці проекту

Фази	Виконавці	
	Синицин Є.	Алексенко О.
Аналіз предметної області		+
Формування ТЗ	+	+
Мета проекту		+
WBS-структура	+	
Календарний план проекту	+	
Управління ресурсами	+	
Управління ризиками	+	

Таблиця Б.3.1 – Виконавці проекту

Фази	Виконавці	
	Синицин Є.	Алексенко О.
Управління якістю	+	
Проектування дизайну блогу	+	
Розробка прототипу сайту	+	
Розробка дизайну сайту	+	
Розробка адміністративної панелі	+	
Тестування	+	
Інструкція користувача	+	
Здача в експлуатацію		+

4 ПОБУДОВА КАЛЕНДАРНОГО ГРАФІКУ ВИКОНАННЯ ПРОЕКТУ

Діаграма Ганта – горизонтальна лінійна діаграма, на якій задачі проекту представляються протяжними в часі відрізками, що характеризуються датами початку та закінчення, затримками і, можливо, іншими тимчасовими параметрами. Для отримання реального уявлення про тривалість виконання робіт з урахуванням обмеженості у використанні ресурсів, на підставі часткової мережевої моделі, а також, проекту в цілому з урахуванням вихідних та святкових днів, було побудовано календарний графік робіт.

Графік виконання дипломного проекту представлено у вигляді Діаграми Ганта на рисунках Б.4.1-2.

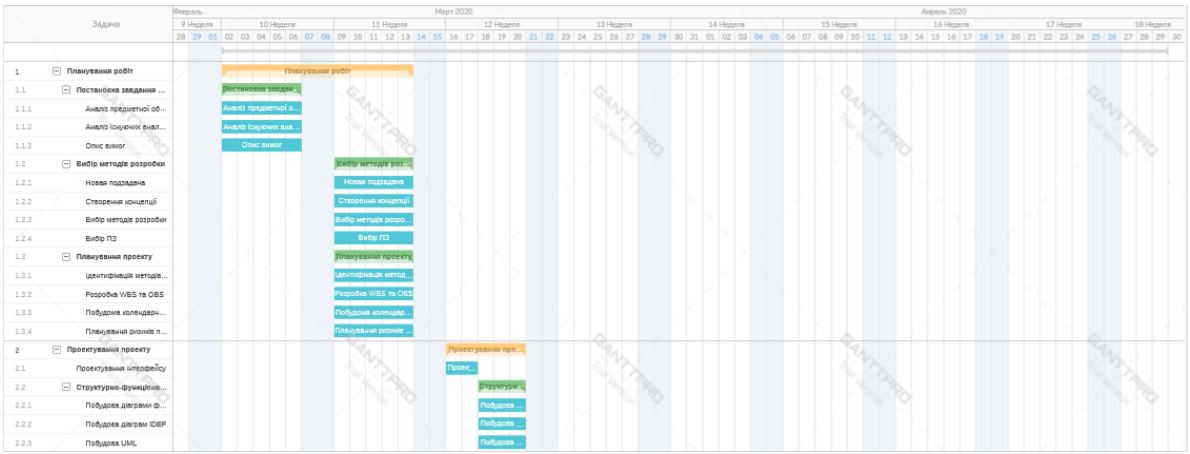

Рисунок Б.4.1 - Діаграма Ганта

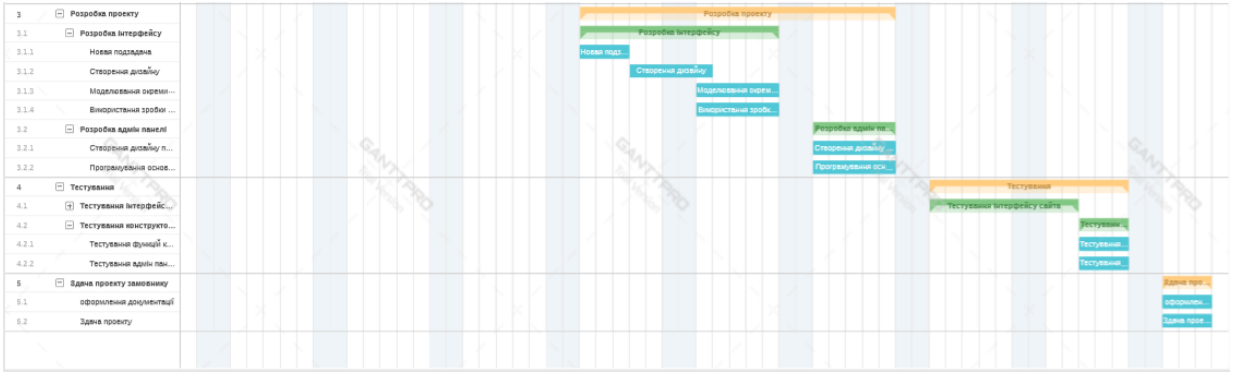

Рисунок Б.4.2 – Продовження діаграми Ганта

5 ІДЕНТИФІКАЦІЯ РИЗИКІВ

Виконаємо якісну і кількісну оцінку ризиків роботи. При якісній оцінці визначимо ризики, що потребують швидкого реагування. Така оцінка визначить ступінь важливості ризику і дозволить вибрати спосіб реагування. Кількісна оцінка ризиків буде виконана для більш повної ідентифікації ризиків та ступеня їхнього впливу на виконання проекту. Кількісна і якісна оцінка ризиків можуть використовуватися окремо або разом, залежно від наявного часу і бюджету, необхідності в кількісній або якісній оцінці ризиків.

Ризик – це ймовірнісна подія, яка у випадку своєї появи негативно або позитивно впливає на проект.

Управління ризиком – це процес реагування на події та зміни ризиків у процесі виконання проекту. При цьому важливим є проведення моніторингу ризиків.

Моніторинг ризиків включає контроль ризиків протягом усього життєвого циклу проекту. Якісний моніторинг ризиків забезпечує управління інформацією, яка допомагає приймати ефективні рішення до настання ризикових подій.

Найбільш розповсюдженою характеристикою ризику є загроза або небезпека виникнення невдач у тій чи іншій діяльності, небезпека виникнення несприятливих наслідків, змін зовнішнього середовища, які можуть викликати втрати ресурсів, збитки, а також небезпеку, від якої слід застрахуватися.

Процес управління ризиками включає в себе такі пункти:

1) Ідентифікація ризиків (виявлення ризиків).
2) Оцінювання ризиків (оцінка ймовірності та впливу).
3) Заходи реагування на ризики.
4) Моніторинг ризиків.

Ідентифікація ризиків – виявлення ризиків здатних вплинути на проект, документальне оформлення їх характеристик. Ризики також поділяються на декілька видів (табл. Б.5.1).

Таблиця Б.5.1 – Види ризику

№	Назва ризику	Опис ризику
1	Зовнішні непередбачувані ризики	Природні катастрофи: повені; землетруси; шторми; кліматичні катаклізми тощо
		Злочини: вандалізм; саботаж; тероризм
		Неочікувані зовнішні ефекти: екологічні; соціальні
		Зриви: у створенні необхідної інфраструктури; через банкрутство підрядників; у фінансуванні: через помилки у визначенні цілей проекту; через неочікувані політичні зміни
2	Зовнішні передбачувані (проте не визначені) ризики	Ринковий ризик у зв'язку із: зміною вимог споживачів; економічними змінами; посиленням конкуренції; втратою позицій на ринку.
		Операційні ризики: порушення безпеки; відступ від цілей проекту; неможливість підтримання робочого стану елементів проекту; неприпустимий екологічний вплив; негативні соціальні післядії; зміни валютних курсів, нерозраховувана інфляція.

Продовження таблиці Б.5.1

	Назва ризику	Опис ризику
	Внутрішні ризики. Внутрішні організаційні ризики.	Зриви планів робіт через: недостачу робочої сили; нестачу часу; помилки проектування; помилки планування; недоліки координації робіт; зміни керівництва; конфлікти та саботаж; зміну можливостей замовника проекту; недостатнє управління. Перевитрати коштів через: зриви планів робіт; невірну стратегію; некваліфікований персонал; переплати за роботу/матеріали; неузгодження частин проекту; невірний кошторис.
	Внутрішні технічні ризики	Зміни технології
		Специфічні ризики технології, що закладаються до проекту
		Помилки в проектно-кошторисній документації
	Інші ризики	Прямі втрати майна: по транспортних спорах; обладнання і т.д.
		Непрямі збитки: перестановка обладнання; неотримання орендного прибутку; порушення запланованого ритму діяльності;
		Ризики, що підлягають обов'язковому страхуванню: нещасні випадки на виробництві;

Продовження таблиці Б.5.1

№	Назва ризику	Опис ризику
6	За джерелами виникнення	Неправильна оцінка ринкової ситуації (ємність ринку, рівень конкуренції).
		Невизначеність мети, інтересів і поведінки учасників проекту, проблеми управління командою;
		Виробничо-технологічне устаткування, виробничі ризики (аварії, брак).
		Неточність проектної документації (витрати, строки реалізації проекту, технічні і технологічні параметри);
		Ризик зміни пріоритетів розвитку підприємства і підтримки з боку керівництва;
		Неповнота або неточність інформації про фінансовий стан та ділову репутацію учасників.

Систематичний ризик – визначається зовнішніми обставинами, не залежить від суб'єкта та не регулюється ним:

– Ризики, пов'язані з нестабільністю нормативного поля режиму інвестування (табл. Б.5.1).

– Можливість змін природно-кліматичних умов, стихійного лиха.

– Можливість погіршення політичної ситуації, несприятливі соціально-політичні зміни.

– Коливання ринкової кон'юнктури, валютного курсу.

Таблиця Б.5.2 – Види прояву ризику

№	Назва ризику	Опис ризику
1	За сферами прояву	Економічні, пов'язані із зміною економічних факторів.
		Політичні, пов'язані із зміною політичного курсу країни.
		Соціальні, пов'язані з соціальними складностями.
		Екологічні, пов'язані з екологічними катастрофами.
2	За видами втрат	Трудові втрати – втрати фондів робочого часу
		Перевитрати часу – уповільнення процесу реалізації проекту у порівнянні з планом
		Фінансові втрати – пов'язані з непередбаченими виплатами (штрафи, уплата додаткових податків), неотриманням коштів з передбачених джерел
		Перевищення витрат – виникає внаслідок зміни початкового плану реалізації проекту або заниження розрахункових витрат на будівництво, затримання строків будівництва
		Ризик нежиттєздатності проекту – передбачені доходи від проекту можуть бути недостатніми для покриття усіх видів витрат
		Ризик недоплати заборгованостей – тимчасове зниження доходу через короткострокове падіння попиту на продукт, або зниження ціни на нього

Продовження таблиці Б.5.2

2	За видами втрат	Виробничі ризики – виникають після завершення впровадження проекту; пов'язані з проблемами технічної підготовки і розробки проекту, поганого інжинірингу, незадовільного навчання персоналу, нестачі сировини і зростання витрат виробництва

В процесі аналізу для визначення числових значень ймовірності виникнення ступеня впливу, зазвичай застосовується метод експертних оцінок. На їх основі визначається ранг ризику, як потенційний вплив ризику на проект, який оцінюється як добуток ймовірності виникнення та ступеню впливу.

Шкала оцінки ризику може відповідати емпіричній шкалі оцінки ризику:

– 5 балів - критичний ризик.
– 4 бали - максимальний ризик.
– 3 бали - високий ризик.
– 2 бали - нормальний ризик.
– 1 бал - малий ризик.

RV = P * I, де RV – ранг ризику; P – ймовірність виникнення; I – ступінь впливу.

В даному випадку на першому етапі, в процесі виявлення ризиків можна виділити ряд ризиків (табл. Б.5.3).

Таблиця Б.5.3 – Ризики проекту

№	Опис ризику	Вплив	Ймовірність	RV	Пом'якшення наслідків
1	Нестабільність Програмного забезпечення	5	3	C	Використовувати перевірене та якісне Програмне забезпечення
2	Некоректний функціонал веб-додатку	4	2	M	Тестування веб-додатку
3	Неактуальність веб-додатку	3	2	M	Використовувати найновіші тренди та сучасний підхід
4	Додаткові вимоги	4	3	M	Обговорити всі вимоги з замовником перед початком проекту
5	Затримка фінансування	3	1	L	Обговорити план фінансування проекту

- Зелений колір – прийнятні ризики.
- Жовтий колір – виправданні ризики.
- Червоний колір – недопустимі ризики.

На підставі отриманого значення індексу ризики класифікують: за рівнем ризику, що знаходиться в табл. Б.5.4.

Таблиця Б.5.4 − Шкала оцінювання за рівнем ризику

№	Назва	Межі	Ризики, які входять (номера)
1	Прийнятні	$1 \leq R \leq 2$	1,8,11,12,13
2	Виправдані	$3 \leq R \leq 4$	2,4,6,10,15
3	Недопустимі	$6 \leq R \leq 9$	3,5,7,9,14

ДОДАТОК В. ПРОГРАМНА РЕАЛІЗАЦІЯ

1. Фрагмент коду з файлу AppComponent.vue: відповідає за відображення базової структури додатку і підключення всіх необхідних компонентів.

```
<template>
  <div id="wrapper">
    <header-component></header-component>
    <div class="page elevation-3">
      <menu-component></menu-component>
      <div class="pb-4">
        <router-view></router-view>
      </div>
    </div>
    <footer-component></footer-component>
  </div>
</template>

<script>
  import MenuComponent from './MenuComponent.vue'
  import HeaderComponent from './HeaderComponent.vue'
  import FooterComponent from './FooterComponent.vue'
  export default {
    name: 'app',
      data: () => ({}),
    components: {
      MenuComponent,
      HeaderComponent,
      FooterComponent
    }
  }
</script>
```

```
<style lang="css">
   .page {
      width: 100%;
      float: left;
      padding-bottom: 20px;
      border: 5px;
      background: #1b1b1b;
      color: #fff;
   }
   .page .title-block {
      font-size: 200%;
      padding: 10px 25px;
      text-transform: uppercase;
      font-weight: bold;
   }
</style>
```

2. Фрагмент коду з файлу HomeComponent.vue: відповідає за відображення головної сторінки сайту.

```
<template>
<div>
      <v-carousel
         cycle
         height="400"
         hide-delimiter-background
         show-arrows-on-hover
         class="slider"
      >
         <v-carousel-item src="/img/slide-1.jpg"></v-carousel-item>
         <v-carousel-item src="/img/slide-2.jpg"></v-carousel-item>
         <v-carousel-item src="/img/slide-3.jpg"></v-carousel-item>
      </v-carousel>
      <v-row>
```

```
            <v-col class="about pa-3" md="4">
              <div class="title-block">Про нас</div>
              <p>{{ about }}</p>
            </v-col>
            <v-col md="3">
              <div class="title-block">Новини</div>
              <div class="news" v-for="item in news" :key="item.id">
                <v-img :src="item.photo"></v-img>
                <p>{{ item.text.slice(0, 200) }}...</p>
                <a :href="'/news/'+item.id">Читати далі</a>
              </div>
            </v-col>
            <v-col md="5">
              <div class="title-block">Розклад занять</div>
              <v-simple-table dark class="schedule">
                <template v-slot:default>
                <thead>
                  <tr>
                  <th class="text-left">ПІБ</th>
                  <th class="text-left">Мета заняття</th>
                  <th class="text-left">Час</th>
                  </tr>
                </thead>
                <tbody>
                  <tr v-for="(item, index) in schedule" :key="index">
                  <td>{{ item.name }}</td>
                  <td>{{ item.purpose }}</td>
                  <td>{{ item.optional_time }}</td>
                  </tr>
                </tbody>
                </template>
              </v-simple-table>
            </v-col>
          </v-row>
        </div>
```

```
</template>
<script>
    export default {
        name: 'home',
        data: () => ({
            schedule: [],
            about: "",
            news: []
        }),
         created() {
            this.getSchedule();
            this.getInfo();
            this.getNews();
        },
        methods: {
            getSchedule() {
                axios.get('/api/schedule')
                    .then((response) => {
                        this.schedule = response.data;
                    })
            },
            getInfo() {
                axios.get('/api/info')
                    .then((response) => {
                        this.about = response.data[0].value;
                    })
            },
            getNews() {
                axios.get('/api/news-home')
                    .then((response) => {
                        this.news = response.data;
                    })
            },
        }
    }
```

```
</script>
<style lang="css">
    .about p {
        padding: 0 25px;
        color: #7f7f7f;
        white-space: pre-wrap;
    }
    .news {
        padding-bottom: 20px;
        border-bottom: 1px solid #000;
    }
    .news img {
        width: 100%;
        margin: 25px 15px;
    }
    .news p {
        margin: 15px 0;
        color: #7f7f7f;
    }
    .news a {
        background: #2e2d33;
        text-transform: uppercase;
        font-weight: bold;
        color: #fff;
        text-decoration: none;
        padding: 10px 20px;
        border: 1px solid #000;
    }
    .news a:hover {
        color: #fbef0d;
    }
    .schedule {
        margin: 0 25px;
    }
    .slider {
```

```
        width: 100%;
    }
</style>
```

3. Код з файлу ApplicationsController.php: відповідає за прийняття, обробку і збереження в базу даних заявок на тренування.

```
<?php
namespace App\Http\Controllers;
use Illuminate\Http\Request;
use App\Models\Applications;
class ApplicationsController extends Controller
{
    function get() {
        $data = Applications::get();
        return response()->json($data);
    }
    function post(Request $request) {
        $model = new Applications();
        $data = $request->all();
        $model->create($data);
        return response('ok', 200);
    }
    function delete($id) {
        Applications::find($id)->delete();
    }
}
```

4. Код з файлу NewsController.php: відповідає за обробку і збереження новин в базу даних.

```
<?php
namespace App\Http\Controllers;
```

```
use Illuminate\Http\Request;
use App\Models\News;
use App\Models\Comments;

class NewsController extends Controller
{
    function get() {
        $data = News::get();
        return response()->json($data);
    }
    function getHome() {
        $data = News::limit(1)->get();
        return response()->json($data);
    }
    function getId($id) {
        $data = news::with('comments')->find($id);
        return response()->json($data);
    }
    function post(Request $request) {
        $news = new News();
        $data = $request->all();
        if($request['photo']) {
            $name = "/img/news/".uniqid().'.'.$request['photo']-
>getClientOriginalExtension();
            $request['photo']->move(public_path()."/img/news/", $name);
            $data['photo'] = $name;
        }
        $news->create($data);
        return response('ok', 200);
    }
    function edit(Request $request, $id) {
        $news = News::find($id);
        $data = $request->all();
        if(gettype($request['photo']) == "object") {
```

```
                $name = "/img/news/".uniqid().'.'.$request['photo']-
>getClientOriginalExtension();
                $request['photo']->move(public_path()."/img/news/", $name);
                $data['photo'] = $name;
            } else {
                $data['photo'] = $news->photo;
            }
            $news->update($data);
            return response('ok', 200);
        }
        function delete($id) {
            News::find($id)->delete();
        }
        // comment
        function postComments(Request $request) {
            $model = new Comments();
            $data = $request->all();
            $request = $model->create($data);
            return response()->json($request);
        }
        function deleteComment($id) {
            Comments::find($id)->delete();
        }
    }
```