

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет кораблебудування імені адмірала Макарова
Навчально-науковий інститут комп'ютерних наук та управління проєктами
Кафедра інформаційних управляючих систем та технологій

"Допущений до захисту"

Завідувач кафедри ІУСТ



к.т.н, доц. Михелєв І.Л.

" ____ " _____ 2025 р.

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття ступеня вищої освіти "бакалавр"

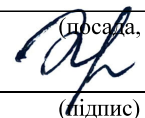
на тему:

«Розроблення інформаційної системи центру кінного спорту»

Виконав: студент групи 4147ст

 *Карай Б.А.*
(підпис)

Керівник роботи:

 (посада, науковий ступінь вчене звання)
Книрик К.О..
(підпис)

м. Миколаїв – 2025 р.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет кораблебудування імені адмірала Макарова
Навчально-науковий інститут комп'ютерних наук та управління проектами
Кафедра інформаційних управляючих систем та технологій
Спеціальність 122 "Комп'ютерні науки"
Освітня програма "Комп'ютерні науки"

"ЗАТВЕРДЖУЮ"
Гарант освітньої програми

к.т.н, доц. Гайда А.Ю.

" ____ " _____ 2022 р.

ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ на здобуття ступеня вищої освіти "бакалавр"

Студенту Карай Богдану Андрійовичу

(прізвище, ім'я, по батькові)

1. Тема роботи: Розроблення інформаційної системи центру кінного спорту

Керівник роботи Книрик К.О.

Затверджені наказом ректора № 295-уч від "27" березня 2025 року.

2. Термін подання роботи: 20 травня 2025 р.

3. Вихідні дані до проекту (роботи) ДСТУ щодо обробки інформації, літературні джерела, технічна документація на існуючі аналоги систем,

4. Перелік питань, що належать до розробки (найменування розділів) Дослідження і аналіз предметної області; Розробка технічного проекту АС; Реалізація проекту веб-додатку; Охорона праці та техніка безпеки під час неї;

5. Перелік презентаційних матеріалів Організаційна структуру центру «Верховина», Схема роботи організації, Схема вирішення запропонованої системи, Діаграма прецедентів, Діаграма діяльності обробки заявки, Концептуальна модель даних, Логічна модель даних, фізична модель даних, Математичне забезпечення, Використані технології та програмне забезпечення, Діаграма класів, Діаграма стану; Результати впровадження

6. Консультанти розділів роботи

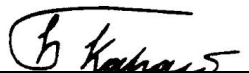
Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1	Книрік К.О.	03.02.2025	22.02.2025
2	Книрік К.О.	22.02.2025	21.03.2025
3	Книрік К.О.	21.03.2025	25.04.2025
4	Книрік К.О.	25.04.2025	07.06.2025

7. Дата видачі завдання 03 лютого 2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів роботи	Примітка
1	Опис та аналіз предметної області	13.02.2025	
2	Постановка задачі	15.02.2025	
3	Розробка концепції системи	22.02.2025	
4	Розробка проектних рішень по системі (та її частинах)	21.03.2025	
5	Реалізація проекту та розробка робочої документації	25.04.2025	
6	Вирішення питань охорони праці	07.06.2025	
7	Виконання графічних документів	12.06.2025	
8	Подання роботи на перевірку на плагіат	10.06.2025	
9	Подання ДР рецензенту	15.06.2025	
10	Подання ДР на захист	20.06.2025	

Студент


(підпис)

Карай Б.А.

(ПІБ)

Керівник роботи


(підпис)

Книрік К.О.

(ПІБ)

АНОТАЦІЯ

У даній дипломній роботі розглядається завдання щодо набуття навичок у розробці й проектуванні інформаційних систем. Розробляється інформаційна система Центру кінного спорту «Верховина».

Робота викладена на 90 сторінках, містить 4 розділи, 33 рисунків, 4 додатки та список використаної літератури з 20 найменувань.

Робота виконана українською мовою.

Ключові слова: інформаційна система, АРМ тренера, АРМ оператора, рекламний сайт, СУБД.

ABSTRACT

This degree work have the task of acquiring skills in the development and design of information systems. The information system is being developed of the «Verkhovina» equestrian center.

The work is laid out on 90 pages, contains 4 subsections, 33 drawing, 4 applications and 20 referes.

The work is done in Ukrainian.

Keywords: Information system, Trainer's Workstation, Operator's Workstation, Advertising Site, DBMS.

ЗМІСТ

ВСТУП	
РОЗДІЛ 1. ДОСЛІДЖЕННЯ І АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ.....	
1.1 Дослідження та аналіз предметної області.....	
1.2 Аналіз заданого підприємства	
1.3 Формування вимог до ІС	
РОЗДІЛ 2. РОЗРОБКА ТЕХНІЧНОГО ПРОЕКТУ АС	
2.1 Інформаційне забезпечення.....	
2.1.1 Розробка та аналіз системних рішень	
2.1.2 Розробка концептуальної моделі даних	
2.1.3 Розробка логічної моделі даних.....	
2.1.4 Розробка фізичної моделі даних	
2.2 Рішення з математичного забезпечення.....	
2.3 Програмне забезпечення	
2.3.1 Рішення з програмного забезпечення	
2.3.2 Структура програмного забезпечення веб-додатку	
2.4 Технічне забезпечення проекту	
РОЗДІЛ 3. РЕАЛІЗАЦІЯ ПРОЕКТУ ВЕБ-ДОДАТКУ	
РОЗДІЛ 4. ОХОРОНА ПРАЦІ ТА ТЕХНІКА БЕЗПЕКИ ПІД ЧАС НЕЇ	
4.1 Поняття охорона праці на державному рівні.....	
4.2 Аналіз небезпечних та шкідливих факторів центрі кінного спорту «Верховина» та шляхи їх усунення	
ВИСНОВОК.....	
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	
Додаток А – Технічне завдання	
Додаток Б – Код створення таблиць mysql.....	
Додаток В – Код серверної частини програми.....	
Додаток Г – Код клієнтської частини програми	

ВСТУП

Кожного дня ми стискаємося з комп'ютерними системами та технологіями, дивимося, як технічний прогрес все більше і більше набирає обертів. З комп'ютерами в нас пов'язано все: навчання, робота, спілкування з іншими людьми. Жодне виробництво зараз не обходиться без втручання інформаційних технологій. В тому числі, без веб-застосунків та веб-сторінок.

Тема дипломної роботи – «Розробка інформаційної системи центру кінного спорту «Верховина»». Спроектowana інформаційна система для центру «Верховина» підвищить рентабельність та актуальність робочих процесів у компанії, створить стержень для подальшого розвитку автоматизації всередині школи.

Мета роботи: визначити проблемні сторони роботи центру кінного спорту «Верховина».

Завдання: проаналізувати компанію, визначитись з проблемами її не ефективної роботи, надати шляхи вирішення цих проблем, визначитись з актуальністю, розробити концепт майбутньої інформаційної системи (ІС далі) - технічні, математичні, програмні сторони, застосувати програмне забезпечення, для реалізації складових інформаційної системи.

Розробка ІС спростить роботу деяких відділів компанії, а реалізація рекламного сайту покращить рентабельність та зацікавить нових клієнтів.

РОЗДІЛ 1. ДОСЛІДЖЕННЯ І АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ.

ПОСТАНОВКА ЗАДАЧІ ЩОДО ЦЕНТРУ «ВЕРХОВИНА»

1.1 Дослідження та аналіз предметної області

Спорт є важливою частиною нашого життя. Згідно з опитуванням GfK Ukraine, проведеним у 2015 році за підтримки ООН, 24% респондентів вважають спорт ключовим для здоров'я. У 2016 році міністерство молоді та спорту повідомило про 659 493 осіб, що займаються літніми олімпійськими видами спорту, і 15 907 осіб, які займаються зимовими видами спорту [1]. До першої категорії входить безліч видів, одним з яких є кінний спорт.

Люди почали взаємодіяти з конями приблизно 5-6 тисяч років тому. Вперше використання коней для їзди відбулося в Азії, а потім поширилося по всьому світу. Спочатку люди їздили на конях без особливої амуніції, що не сприяло розвитку та навчання коня. Однак пізніше, у зв'язку зі зростанням військових та тактичних потреб, з'явилося систематичне навчання людей і коней у манежах.

У 1990 році кінний спорт входить до Олімпійських ігор, а вимоги до юних і не дуже вершників зростає. Міжнародна федерація кінного спорту офіційно визнає такі підвиди кінного спорту: виїздка, конкур, рейтинг, драйвінг тощо[2]. І як у будь-якому іншому спорті розвиток та навчання відіграє важливу роль. Тому їй доводиться звертатися до різних центрів кінного спорту.

Центр кінного спорту з боку огляду клієнтом має такі пункти:

1. Коні мають бути добре доглянуті, проживати в належних умовах, мати доступ до виходу, ветеринарної допомоги, їжі та води. Породистість бажана, але не обов'язкова. Кінь повинен відповідати рівню вмінь учня та характеру тварини, щоб тренер міг збалансувати навички обох.

2. Свіжа їжа. Клуб кінного спорту повинен забезпечувати різноманітний корм для коней: зерно, фрукти і овочі, вітаміни, особливо взимку. Хороше

сіно має бути доступним постійно, оскільки воно утворює основу раціону для здорового коня.

3. Забезпечення належного догляду за конями значною мірою залежить від професійної компетентності конюхів. Вони повинні володіти необхідними знаннями щодо утримання коней, мати відповідне спорядження та дотримуватися регулярності в очищенні денників з метою запобігання погіршенню санітарних умов.

4. Кваліфіковані тренери є необхідною складовою навчального процесу у кінному спорті. Самостійне опанування теоретичних аспектів може бути недостатнім, оскільки помилки часто залишаються непоміченими або виявляються надто пізно. Тренер, який володіє відповідною кваліфікацією, здатен коригувати помилки, демонструвати правильні техніки та сприяти всебічному розвитку учня як вершника.

5. Ветеринари та лікарі для наїзників. Коні можуть хворіти або отримувати травми, тому ветеринар повинен бути доступним для консультацій і вказівок. Кінний спорт небезпечний, оскільки падіння з сідла звичне явище. Лікар має бути завжди доступний для надання першої допомоги.

6. Додаткові послуги. Додаткові послуги включають не тільки оренду коня або укладання договору з тренером, а також оренду амуніції, аксесуарів, предметів догляду та іншого для зручного використання комплексу.

7. Вартість послуг. Ціни повинні бути врегульованими, точними та відповідати рівню послуг. Клієнти повинні мати доступ до всієї специфіки та тонкощів, які включаються у вартість.

Ці дані знадобляться будуть використані у роботі далі.

Розглянемо приклад. Суб'єкт А, який проявляв інтерес до коней ще в ранньому дитинстві, ухвалює рішення про інтеграцію цього інтересу в своє

професійне життя. Для реалізації даної мети він розпочинає пошук оптимальних шляхів взаємодії з кіньми. У його розпорядженні три основні варіанти: консультування з особами, які мають досвід у даній сфері; пошук закладів, пов'язаних із заняттями кіньми, шляхом відвідування локальних установ; або використання можливостей мережі Інтернет для пошуку рекламних сайтів відповідних комплексів, які пропонують широкий спектр послуг.

Проаналізуємо ці шляхи:

1. Запитувати у знайомих, надалі суб'єкт Б. Думка суб'єкта Б, без аргументації та якогось ґрунту, дуже суб'єктивна. Те, що йому подобається, може просто не сподобатися суб'єкту А. Пізніше це може призвести до того, що за підсумком дія «за рекомендаціями» не принесе жодного результату. Суб'єкт Б образиться на щирі невдоволення А, а потім може просто втратити спілкування через таку дурість. Але й важко шукати людину з потрібної сфери, якщо в найближчому оточенні її немає. Тобто цей шлях має місце, проте фінал може принести жодних результатів і дуже загальмувати процес.

Плюси першого шляху:

- Можна дізнатись інформацію від довіреної особи;
- Відсівається велика кількість непотрібної інформації. Мінуси першого шляху:
- Суб'єктивність думки;
- При відсіюванні інформації може бути втрачена та, що потрібна суб'єкту.

2. Ходити містом у пошуках чогось пов'язаного з кінною їздою. По-перше, все, що пов'язане з кіньми, – це величезний спектр питань. Це може бути кінний магазин амуніції, господарського приладдя для догляду; центр для реабілітації інвалідів контактом із тваринами; підприємства оренди для, припустимо, прокату коней для поїздок у ліс. Тобто складно зорієнтуватися і знайти відразу те, що потрібно.

По-друге, найчастіше центри верхової їзди знаходяться не в центрі міста через великі розміри. Для коней потрібен простір, для учнів – місце для навчання. Тобто нашому суб'єкту А доведеться їздити в різні точки міста.

Це займе дуже багато часу та сил, що не доцільно. Цей шлях взагалі є слабким, порівняно з першим та третім пунктом.

Плюси другого шляху:

- Можна на власні очі побачити комплекс відразу ж, оцінити всі послуги, що надаються віч-на-віч з представником компанії.
- Одразу ж укласти договір, якщо все сподобалося. Мінуси другого шляху:
- Довгий пошук, який може дуже затягнутися за відсутності результату.
- Великий спектр заданого питання, що додасть ще більше часу і витрат людських ресурсів на порожнечу.

3. Пошук комплексу в Інтернеті. Найкращий варіант для суб'єкта А. Він зможе без жодних труднощів знайти те, що йому необхідно, проаналізувати всі комплекси та центри, що знаходяться в його місті, при цьому не виходячи з дому. Найчастіше на сайтах вже є вся потрібна інформація, а в розділі зв'язку суб'єкт зможе знайти телефони, пошту чи інші шляхи зв'язку з підприємством. А навіть якщо нічого не сподобається з перших кількох запитів у пошуковій системі, таргетована реклама вишикується в браузері телефону/комп'ютера так, що тепер у рекламних банерах інших сайтів будуть виникати комплекси, які так необхідні суб'єкту.

Плюси:

- Усі послуги, переваги та недоліки за допомогою лише кількох натискань без виходу з дому.
- Великий діапазон вибору суб'єкта А.
- Підстроювана реклама, яка зможе вивести нові пункти вирішення проблеми.

- Інтерактивність та краса допоможе швидше визначитися з вибором.

Мінуси:

- Пошук також може бути тривалим, якщо варіантів дуже багато, однак вони в рази менші за часом, ніж перші два шляхи.
- Якщо сайт побудований правильно, у ньому є всі розділи та відповіді на запитання, які можуть виникнути у користувача, то мінусів у даного шляху може не бути.

Тобто робимо висновок, що останній шлях є найоптимальнішим.

Визначилися із вибором шляхи пошуку необхідного для суб'єкта А.

Отже, у нас є приклад пунктів кінного комплексу, а також ми вибрали шлях, яким може йти клієнт при пошуку та виборі послуг. Як було зазначено вище, шлях вибору за допомогою веб-доступу є найоптимальнішим та найактуальнішим на даний момент.

При організації центру використовувати документацію на руках досить важко, до того ж при розробці рекламного сайту для цього продукту буде дуже складно орієнтуватися на текстову документацію і вести облік змін у реальному часі. Це дуже ускладнює процес не лише рекламної кампанії центру, а також погіршує продуктивність роботи, наприклад, тренерів, яким доведеться довго зіставляти свої графіки розкладів та затверджувати їх з іншими колегами. Тому як ніколи актуально автоматизувати цей процес, щоб спростити багато процедур:

- Облік студентів та узгодження його вибору з приводу виду тренувань. З цього впливає облік тренерів, який зіставлятиметься з учнями комплексу;
- Облік коней, їх кваліфікації, породи, віку та інших параметрів. Додатково оновлення інформації, що стосується орендодавця тварини;
- Облік загальної суми зі списку цін, що має виплачувати суб'єкт користування послуг комплексу;

- Облік зарплати працівників комплексу, розподіл обов'язків різних груп штату;
- Облік фінансових маніпуляцій: закупівля харчування, господарської та іншої продукції для коней та їх оточення.

1.1 Аналіз заданого підприємства

Центр кінного спорту «Верховина» - це заклад, який навчає всіх бажаючих не лише різним категоріям кінного спорту, а правильній взаємодії з тваринами, соціальній адаптивності учня за допомогою тренера до труднощів, наприклад, до змагань.

«Верховина» має у своєму доступі такі основні завдання: можливість давати учням брати участь у змаганнях на рівні міста, області і навіть країни; для молодих учнів - повний курс навчання поведінки з конем; для старших та досвідчених – прогрес у своєму професіоналізмі; є коні різних категорій та порід; навчання спеціалістами своєї справи від імені тренерів різних видів цього спорту: конкуру, виїздки; лікування коней ветеринарами.

Організаційна структура школи «Верховина» представлено на рисунку 1.1.

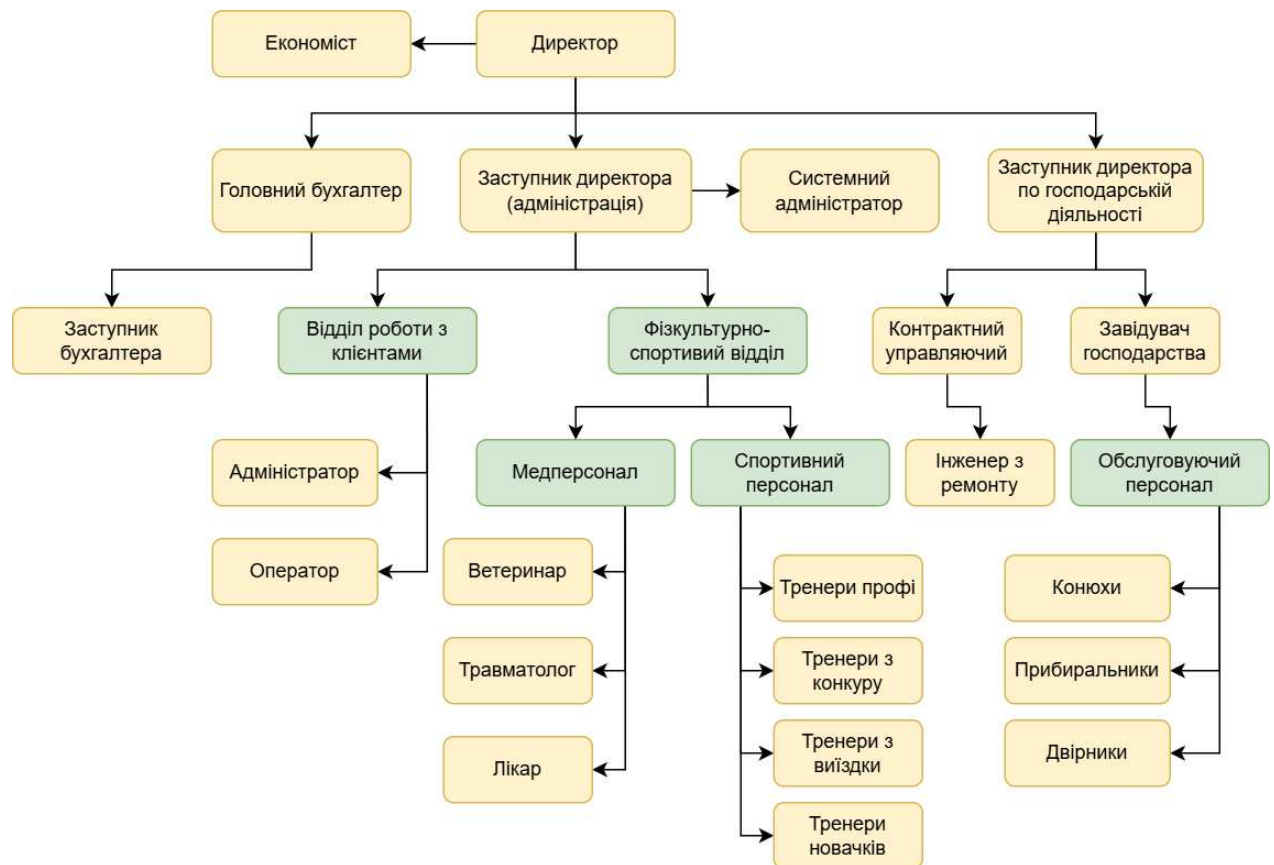


Рис. 1.1. Організаційна структура школи «Верховина»

Пояснимо структуру підприємства:

Директор – контролює роботу інших частин структури організації, перевіряє документацію та підписує договори, затверджує проектні рішення та керує системою в цілому.

Економіст та головний бухгалтер – займаються економічною частиною компанії, перевіряють та узгоджують фінанси з директором. При своїй роботі використовують АРМ економіста та бухгалтера.

Заступник директора з адміністрації – погоджують основну діяльність комплексу із директором, спостерігає за роботою системи; системний адміністратор – помічник з технічної частини заступника адміністратора.

Фізкультурно-спортивний відділ включає медперсонал (ветеринар, травматолог, доктор) – перевірка фізичного стану тварин та їх власників, а також спортивний персонал (тренера з різних видів кінної їзди) – тренування та підготовка студентів школи. Використовують державні АРМ для ведення звітів щодо своїх пацієнтів.

Відділ роботи з клієнтами взаємодіє з останніми, затверджує заявки, відправляє їх на розгляд, наприклад, тренерам, щоб у майбутньому приймали рішення про укладення контракту з майбутнім учнем.

Заступник директора з господарської частини – врегулює питання щодо ремонту та модернізації комплексу. Завідувач стежить над поставками предметів діяльності (продуктів, засобів для догляду, устаткування персоналу госпчастини). Вони використовуються АРМ для ведення звітів про використання.

Молодший обслуговуючий персонал – конюхи (стежать за кінями та його денниками), прибиральники (прибирають внутрішні частини комплексу), двірники (прибирають зовнішні ділянки, які також включають манеж і загони).

При цьому, порівнявши наведені раніше пункти школи, більшість частин «Верховина» виконує и вдосконалює, що є плюсом системи. Процеси з економіки, фінансів, господарської діяльності та мед персоналу виконані та автоматизовані.

Але проаналізувавши структуру заданої системи далі, робимо висновок, що більше всього від не автоматизованості страждають тренери та учні, а останні є запорукою успішної роботи системи в цілому; відділ роботи з клієнтами не автоматизований та немає конкретного доступу до клієнто орієнтованої частини проекту. уся документація цих частин виконана у паперовому та усному формулюванні. З цього випливають такі проблеми:

1. Повільне узгодження розкладу між тренерами, що призупиняє його активізацію та представлення для студентів школи;

2. Можуть виникнути нестикування, що призведе до скасування занять, як приклад;

3. При подальшій розробці рекламного сайту неможливо буде показати точний розклад тренерів та його зміни, що може збентежити та відкинути нових клієнтів, які зацікавлені тим чи іншим тренером;

4. Клієнт, який потрапляє на майбутній сайт, знаходить контакти операторів, дзвонить та не може отримати точну інформацію від них, бо немає швидкого доступу до, наприклад знову, розкладу;

5. Втрата даних та неправильне його трактування;
6. Несвоєчасне встановлення зайнятості тренерів;
7. Не автоматизованість встановлення рейтингу учнів;
8. Відсутність сповіщення існуючим клієнтам щодо сплачення аренди коня чи інших послуг.

Схему роботи з клієнтами організації без застосування АС тим часом представлено на рисунку 1.2.

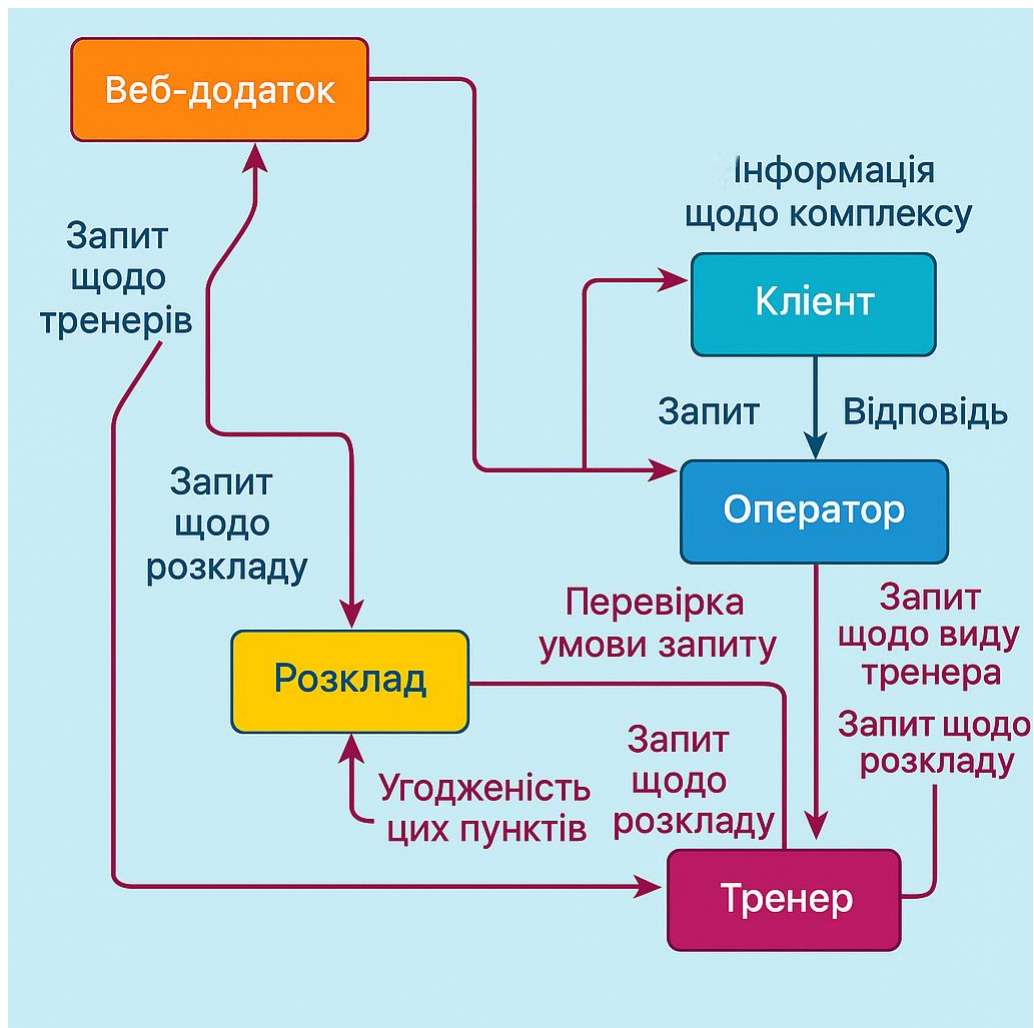


Рис. 1.2 Схема роботи організації

Система працює дуже повільно, клієнт може чекати на відповідь від оператора довго і втратити терпіння, що призводить до того, що підприємство позбудеться нового студента.

1.3. Формування вимог до ІС

Шляхом вирішення цих поставлених питань та проблем може бути модель запропонованої ІС, що представлена на рисунку 1.3.

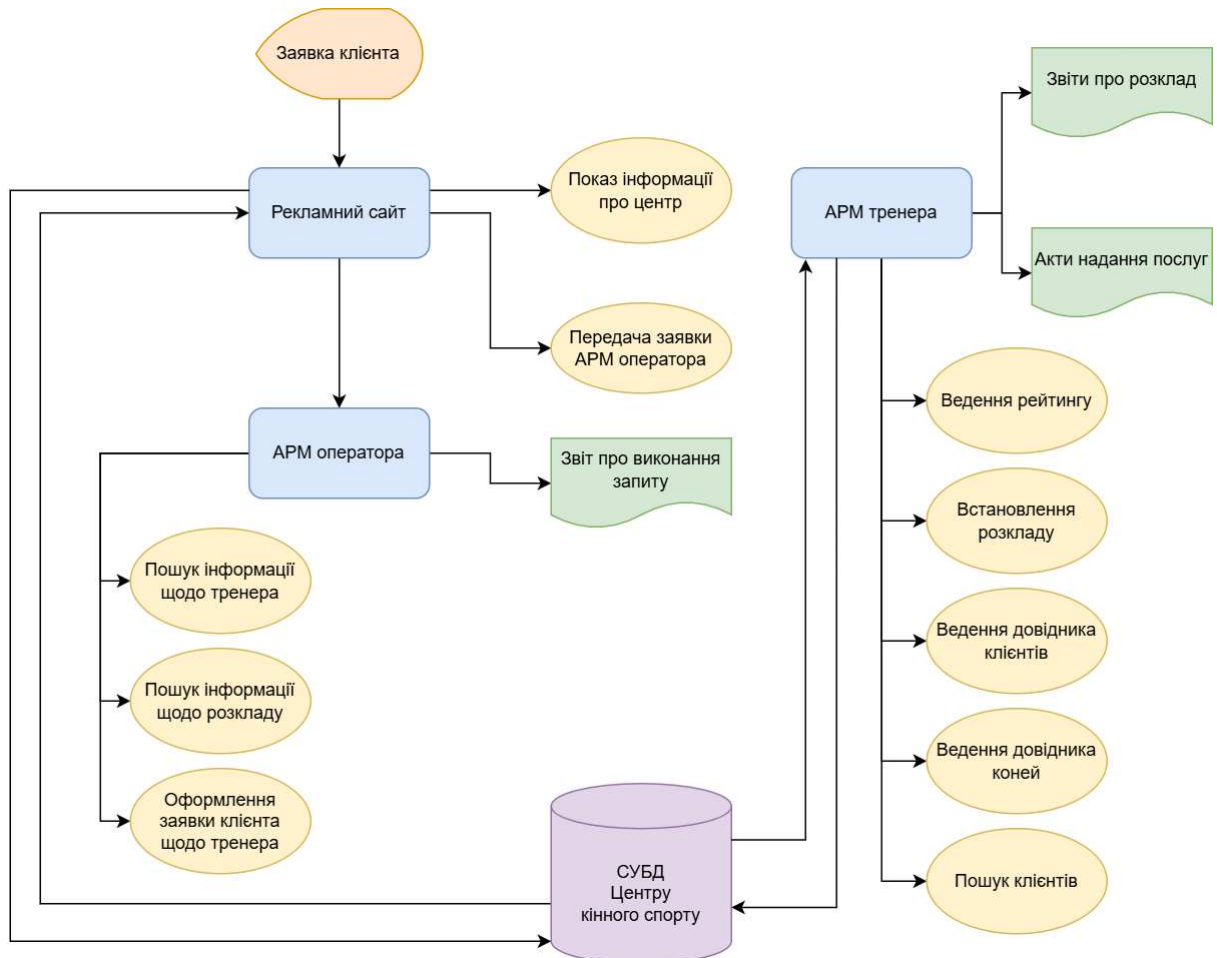


Рис. 1.3 Модель запропонованої ІС

Клієнт спочатку розглядає сайт, на якому висвітлюється частини системи (який динамічно оновлюється, бо пов'язаний з системою). Він вирішує відіслати запит, щоб задати якісь питання або використатися послугами комплексу. Оператор отримує запит та вирішує його, користаючи своїм АРМ, в якому вже динамічно погоджено вимоги, що надалися: тренер вільний, розклад вистроєний та має структуру (тренер впливає на ці структурні частини за допомогою свого АРМ), тому відповідь має бути задовільною. Коли система відповідає, оператор передає цю інформацію клієнту. За допомогою цієї системи зменшується час обробки запиту клієнта, що покращує рентабельність підприємства, підвищує її

актуальності, клієнт задоволений та вже може порекомендувати комплекс іншим із-за швидкості роботи підприємства.

За основу беремо розробку ІС з подальшим спрямуванням розвитку рекламного веб-додатку.

Вимоги, які виконуються, щодо ІС:

1. Усі дані, що входять в ІС, взаємопов'язані, мають структуру та шаблони, втрат не може бути;
2. Узгодженість клієнта з його майбутнім тренером та його розкладом за часом зменшується;
3. При розробці майбутнього рекламного сайту, усі дані, що змінюються в системі, будуть також динамічно змінюватися і у веб-додатку;
4. Оператор швидко реагує на різні питання, дивлячись в додаток АС;
5. Автоматична зайнятість тренера або, навпаки, його визволення;
6. Клієнт завжди оповіщений щодо сплати послуг школи;
7. Тренер підраховує рейтинг, а учень його бачить.

Функціонування системи передбачає використання таких програмних засобів:

1. Реляційна СУБД;
2. Мова програмування сервісної частини;
3. Мова розмітки та формальна мова опису зовнішнього вигляду документа клієнтської частини;
4. Браузер;
5. Середовище розробки програмного забезпечення.;
6. Графічний редактор для створення діаграм.

Функціонування системи передбачає використання таких сервісних засобів:

Постійний Інтернет зв'язок для динамічного виведення інформації на рекламний сайт;

1. Захист інформації від зовнішніх подразників та загроз;
2. Почтове програмне забезпечення для автоматизації відділу роботи з клієнтами.

Програмні засоби

Для того, щоб керувати базою даних, вибираємо реляційну систему MySQL. Розроблятиметься веб-додаток у середовищі розробки JetBrains WebStorm.

Серверна частина: розробляється на JavaScript з використанням веб-фреймворка Express для обробки запитів Http. Доступ до БД відбувається за допомогою драйвера mysql2.

Клієнтська частина: розробляється на JavaScript з використанням веб-фреймворку Vue. Отримання даних від сервера відбувається за допомогою axios, що базуються на Promise HTTP-клієнта. Також у клієнтській частині є відправлення даних на пошту. Для цього використовується сервіс EmailJS. Мова розмітки – HTML5 із використанням SASS.

Веб-додаток є кросплатформним, тому для його доступу та роботи необхідна операційна система. Для комп'ютерів ОС – Linux, браузер Chrome, смартфонів – Android, IOS, браузер Chrome.

Для створення діаграм у процесі розробки використовуватиметься сервіс Draw.io.

Для посилання даних на пошту використовуємо безкоштовний сервіс Google – Gmail.

Інтернет підключення – використовується технологія Ethernet. З'єднання серверу та робочого комп'ютера здійснюється за допомогою бездротової мережі і роутеру, який підключений до Ethernet Інтернет кабелем.

Для захисту системи від зовнішніх загроз використовується антивірус Avast.

Технічні засоби

Системні вимоги щодо клієнтської частини реалізації:

- 1) Процесор – починаючи з Intel® Core™ i3-10100F CPU @ 3.60 GHz;
- 2) Тип системи – 64-розрядна;
- 3) Оперативна пам'ять – від 4гб;
- 4) Відеокарта – від 256 mb;
- 5) Монітор – 1920 x 1080;
- 6) Клавіатура;
- 7) Миша;
- 8) Місце на жорсткому диску – 4 гб;
- 9) Мережевий адаптер – Intel® Ethernet Connection (12) I219-V.

Системні вимоги щодо серверній частини реалізації:

- 1) Процесор – починаючи з Intel® Core™ i3-10100F CPU @ 3.60 GHz;
- 2) Тип системи – 64-розрядна;
- 3) Оперативна пам'ять – від 4гб;
- 4) Відеокарта – від 256 mb;
- 5) Монітор – 1920 x 1080;
- 6) Клавіатура;
- 7) Миша;
- 8) Місце на жорсткому диску – 35гб;
- 9) Мережевий адаптер – Intel® Ethernet Connection (12) I219-V.

При цьому для перевірки адаптивності веб-додатку потрібен телефон, бо не завжди віртуальне відображення розмірів є коректним.

Висновок

При реалізації цієї ІС ми більш актуалізуємо підприємство, а особливо частину контакту клієнта та робітників комплексу. За допомогою розробки веб-додатку ми прорекламуємо комплекс, а її інтерактивністю та

динамічним відновленням даних зацікавимо нових клієнтів. При цьому робота щодо оновлення цих самих даних автоматизується, збільшиться продуктивність роботи та майже зникнуть проблеми з втратою даних.

Висновок

У першому розділі було визначено технічні характеристики та основні переваги запровадження нової інформаційної системи для комплексу. З розвитком веб-додатку підприємство отримує додатковий інструмент для залучення клієнтів та автоматизації роботи з даними. Такий підхід дозволяє підвищити продуктивність, мінімізувати втрати інформації та створити інтерактивний досвід для користувачів, сприяючи зростанню популярності комплексу серед нових клієнтів.

РОЗДІЛ 2. РОЗРОБКА ТЕХНІЧНОГО ПРОЕКТУ АС

2.1 Інформаційне забезпечення

2.1.1 Розробка та аналіз системних рішень

Розробка будь-якої інформаційної системи включає себе проектування її заздалегідь, щоб уникнути майбутніх проблем та помилок, які можуть виникнути в подальшій розробці програмної частини. Так, наприклад, для проектування майбутніх баз даних використовують моделі на кожному етапі, на яких БД змінюється або деталізується: концептуальна, логічна, фізична. При цьому можна використовувати різні алгоритми та технології для подібного. Про це трохи згодом.

Проведемо аналіз. Користуватися даною системою будуть: відділ роботи з клієнтами (оператори), тренери та користувачі. Користувачі будуть переглядати веб-сторінку з інформацією про кінну школу, дізнаватися про послуги, що надаються, рейтинги та розклади. Тренери зможуть оновлювати інформацію щодо розкладу, встановлювати рейтинги учнів. Оператори ж переглядають деяку інформацію (розклад, наприклад) і можуть надавати більш докладну версію, коли клієнт із нею зв'язується. Подібне надано у діаграмі прецедентів ІС на рисунку 2.1.

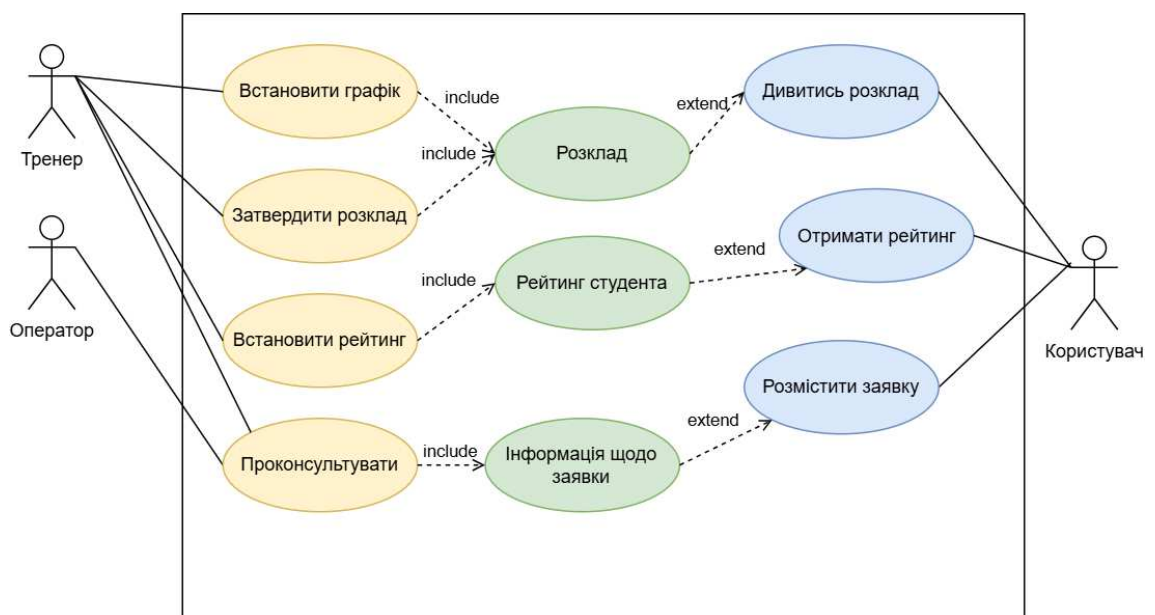


Рисунок 2.1 Діаграма прецедентів

Тренер виконує «встановити рейтинг», який, логічно, включає рейтинг

студента, а студент цей рейтинг бачить через «отримання рейтингу». Так само при консультації оператором клієнта тренер також може вступати в консультацію.

Для допомоги у сприйнятті роботи надаємо діаграму діяльності обробки заявки, коли користувач хоче дізнатися інформацію про тренера. Діаграма показана на рисунку 2.2.

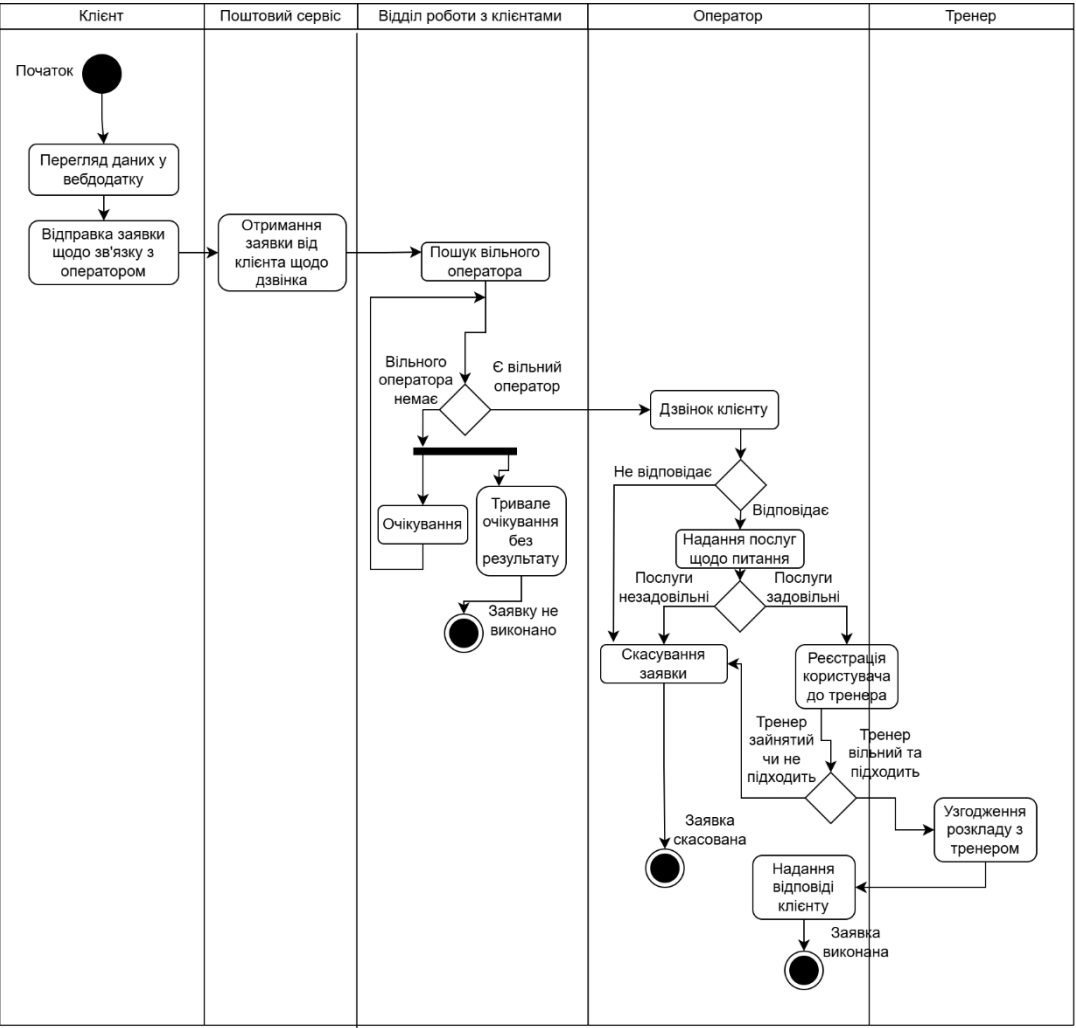


Рис. 2.2. Діаграма діяльності обробки заявки

Спочатку клієнт переглядає рекламний сайт, оглядається ситуація, коли йому щось сподобалось, і він залишає заявку на дзвінок. Після цього ця заявка відправляється на пошту відділу роботи з клієнтами, вони самостійно обирають вільного оператора. Якщо ж вільних операторів немає, заявка ставиться в очікування, поки хтось не звільниться. При довгому очікуванні

заявка не виконується, а на пошту клієнта надсилається лист із проханням про очікування. Якщо оператор знаходиться, виконується наступні дії: оператор дзвонить клієнту, якщо той не відповідає довгий час, заявку повністю відхиляють. В іншому випадку клієнт відповідає, оператор представляється, спочатку коротко визначає ту послугу, про яку запитував клієнт у заявці. Послуги задовольняють клієнта, він відповідає згодою пропозицію оператора пошуку потрібного тренера. Оператор із допомогою ІС оглядає необхідного тренера. Якщо він не вільний, заявку відмовляють та пропонують послуги іншого тренера. При цьому може вийти так, що клієнт повністю відмовляється від послуг, заявку також скасовано. Якщо ж тренер вільний, клієнту він подобається, далі йде погодження часу та розкладу тренера (наприклад, особиста зустріч або пробне тренування), заявка виконана.

Основними характеристики гарної системи є:

1. Правильна архітектура та гнучкість системи - є найважливішою частиною ІС. При правильному побудові робота буде йти динамічно, зменшиться кількість помилок, час роботи та обробки запитів будуть швидше. Гнучкість системи зможе допомогти у розширенні її можливостей у майбутньому. Як приклад, створювана база даних зможе використовуватися поза межами користування тренерів та операторів;

2. Надійність у використанні бере також велику роль у своєму призначенні. ІС потрібно проектувати так, щоб захист був як від внутрішніх, так і від зовнішніх загроз. Встановлення антивірусних програм забезпечить безпеку від вірусів, як приклад;

3. Способи захисту від випадкових і не випадкових технічних збоїв – потрібно встановити, щоб там, де буде встановлено сервер, увімкнуто безпеку від раптових відключень від мережі, які можуть призвести до втрати даних;

4. Пошук інформації та комфортність у її перегляді – при проектуванні,

наприклад, сайту, потрібно не забувати про всі пункти роботи з графікою.

Анімації впадають у вічі, клієнту сподобається, якщо все крутиться, підсвічується при наведенні, перемикання між якимись компонентами відбувається плавно. Також потрібно враховувати те, що людина може заходити на сайт із телефону. Дані на сторінці мають бути адаптовані. У цьому випадку, можна використовувати чуйно адаптивну верстку, при якій компоненти сторінки будуть вибудовуватися в залежності від розмірів екрану (ширина і висота буде змінюватися, блоки можуть вишиковуватися один під один, і головне, що на відміну від звичайно адаптивної верстки, все не буде скакати по сторінці, а плавно змінюватись).

2.1.2 Розробка концептуальної моделі даних

Розробка концептуальної моделі відбувається на підставі тих вимог та завдань, що були поставлені вище за проектом. При цьому необхідно враховувати свої особливості побудови, щоб досягти необхідного результату.

Концептуальна модель – це відображення предметної області бази даних, що розробляється. Як ми знаємо, база даних складається з сутностей та її атрибутів, а також зв'язків між цими сутностями. Розробляючи концептуальну модель, слід пам'ятати, що сутності відображаються у вигляді прямокутника, атрибути – овалів, а зв'язки – лініями, які посередині з'єднуються в один ромб, який визначає, що робить цей зв'язок.

Використовуємо при розробці концептуальної моделі модель «сутність-зв'язок» або ERD діаграма (Entity–relationship model). Вона є однією з найзручніших інструментів уніфікованого представлення даних, незалежного від програмного забезпечення, що його реалізує[3]. Проектуючи, потрібно перетворити ERD модель у конкретну схему вибраної СУБД.

При аналізі функціональної моделі ІС, бачимо, що необхідно створити таку концептуальну модель, щоб покрити всі задані завдання. Наведемо

конкретні сутності, їх атрибути та зв'язки.

Сутність Користувач має такі атрибути (незалежна сутність):

- Ім'я;
- По-батькові;
- Прізвище – ці три атрибути мають ПІБ користувача;
- Телефон – має мобільний телефон користувача, має бути разом із телефонним кодом, оскільки може бути таке, що користувач приїхав із-за кордону і ще немає номера мобільного оператора країни;
- День народження – має день народження користувача;
- Рейтинг користувача – має дані, необхідні для обліку на додаткові послуги (здебільшого тренерів);
- Стать – має стать користувача;
- Дані по зображенню користувача – опис даного атрибуту так само перебуватиме і в інших сутностях, тому опишемо його трохи пізніше;
- Тип коня – має зв'язок з сутністю Кінь. Про це трохи згодом.

Сутність Працівник має такі атрибути (незалежна сутність):

- Ім'я;
- По-батькові;
- Прізвище – ці три атрибути мають ПІБ робітника;
- День народження – має день народження робітника;
- Стать – має стать працівника;
- Телефон – має мобільний телефон, так само вказується з телефонним кодом, як і у користувача;
- Тип посади – пов'язаний з сутністю Штат;
- Дані щодо зображення працівника.

Сутність Штат має такі атрибути (незалежна сутність):

- Назва посади – має назву посади;
- Оклад – має суму зарплати працівника для статистики.

Сутність Сервіс має такі атрибути (залежна сутність):

- Тип користувача – пов’язаний з сутністю Користувач;
- Тип працівника – пов’язаний з сутністю Працівник;
- Назва послуги – має назву послуги, що надається.

Сутність Розклад має такі атрибути (залежна сутність):

- Тип розкладу – пов’язаний з сутністю Тип розкладу;
- Тип часу розкладу – пов’язаний з сутністю Час розкладу;
- Тип тижня розкладу – пов’язаний з сутністю День тижня.

Сутність День тижня має такі атрибути (характеристична сутність):

- Назви дня тижня – має дні тижня для розкладу.

Сутність Тип розкладу має такі атрибути (характеристична сутність):

- Назва типу розкладу – має тип розкладу.

Час розкладу має такі атрибути (характеристична сутність):

- Початок заняття – має час початку заняття;
- Кінець заняття – має час кінця заняття.

Сутність Кінь має такі атрибути (незалежна сутність):

- Ім'я – має ім'я коня;
- Вік – має вік коня;
- Стать – має стать коня;
- Порода – має породу коня;
- Клас – має категорію коня за статусом фізичної підготовки;
- Дані щодо зображення коня.

Сутність Оренда має такі атрибути (залежна сутність):

- Тип користувача – пов’язаний з сутністю Користувач.

- Тип ціни – пов’язаний з сутністю Ціна.

Ціни мають такі атрибути (незалежна сутність):

- Ціна оренди – має ціну орендованого товару/послуги;
- Назва оренди – має назву оренду.

Зв’язки між сутностями та їх типи представлені у таблиці 2.1.

Назва сутностей	Їх зв’язок	Тип зв’язку
Сервіс $\rightleftharpoons$ Користувач	Використовує	Один-до-багатьох
Сервіс $\rightleftharpoons$ Робітник	Обслуговує	Один-до-багатьох
Штат $\rightleftharpoons$ Працівник	Працює	Один-до-багатьох
Користувач $\rightleftharpoons$ Кінь	Орендує	Один-до-одного
Розклад $\rightleftharpoons$ Користувач	Слідкує	Один-до-багатьох
День тижня $\rightleftharpoons$ Розклад	Має	Один-до-багатьох
Тип розкладу $\rightleftharpoons$ Розклад	Має	Один-до-багатьох
Час розкладу $\rightleftharpoons$ Розклад	Має	Один-до-багатьох
Оренда $\rightleftharpoons$ Користувач	Сплачує	Один-до-багатьох
Оренда $\rightleftharpoons$ Ціна	Отримує	Один-до-багатьох

Табл.2.1. Зв’язки між сутностями

Щодо даних, що стосуються зображень. У самій базі буде інформація про назву самого файлу, щоб пізніше використовувати її в подальшій розробці сайту. При цьому потрібно розуміти, що на сервері необхідно виділити якусь кількість пам'яті для цього.

Концептуальну модель даних будемо за допомогою діаграми «сутність- зв'язок». Модель представлена рисунку 2.3.

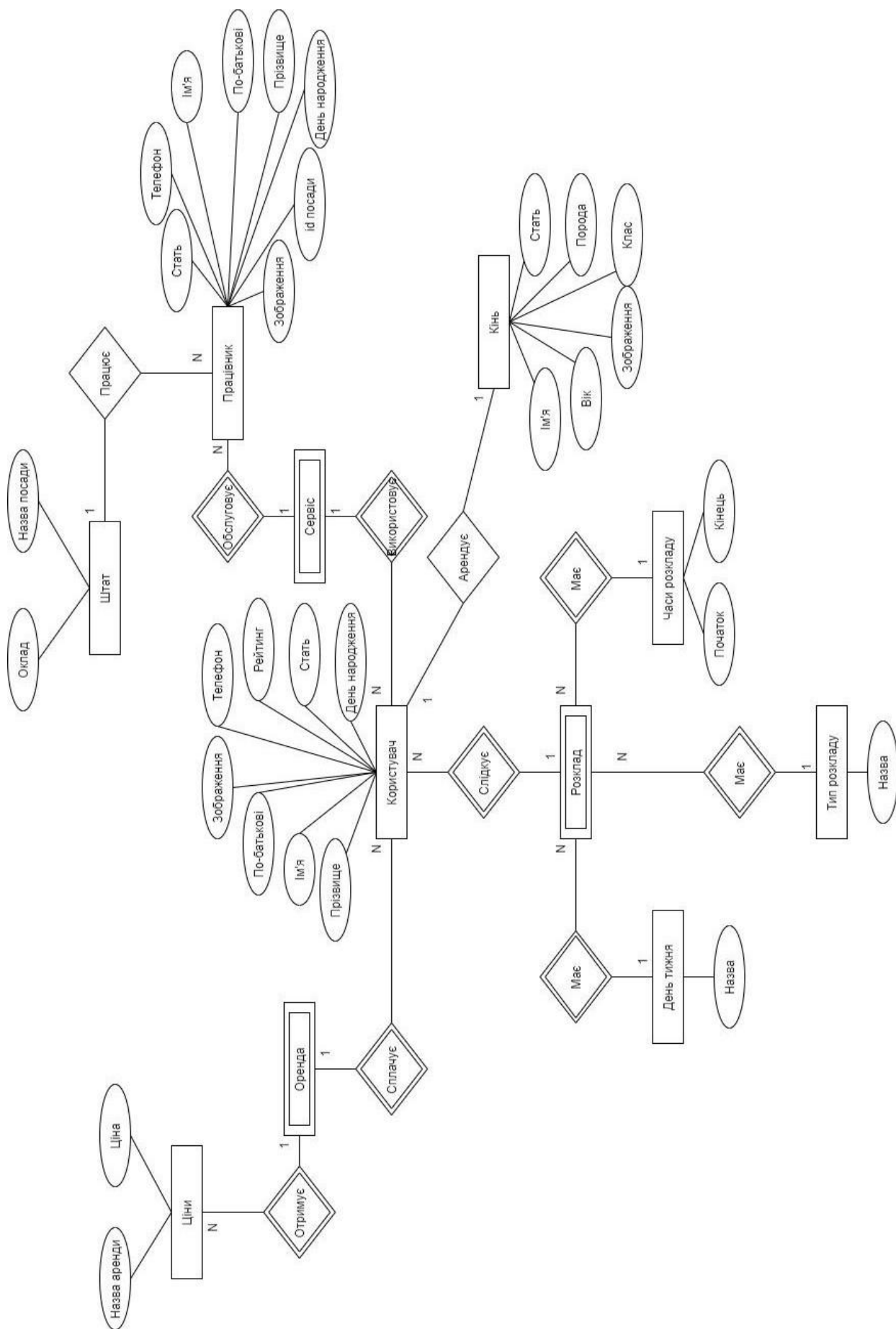


Рис.2.3. Концептуальна модель даних

2.1.3 Розробка логічної моделі даних

При розробці логічної моделі даних необхідно базуватися на концептуальній моделі, яка надана вище. У цьому логічна модель даних зазвичай розширює минулу модель, деталізує її, трансформує деякі сутності та його атрибути. На цьому етапі також вказуються первинні та зовнішні ключі кожного запиту. В першу чергу, логічна модель даних створюється для користувача, щоб формально показати структуру БД перед визначенням фізичної моделі.

Логічна модель даних зазвичай є прототипом майбутньої бази даних. Прототип бази даних, яка буде використовуватися при розробці веб-додатку, у вигляді логічної моделі представлено на рисунку 2.4.

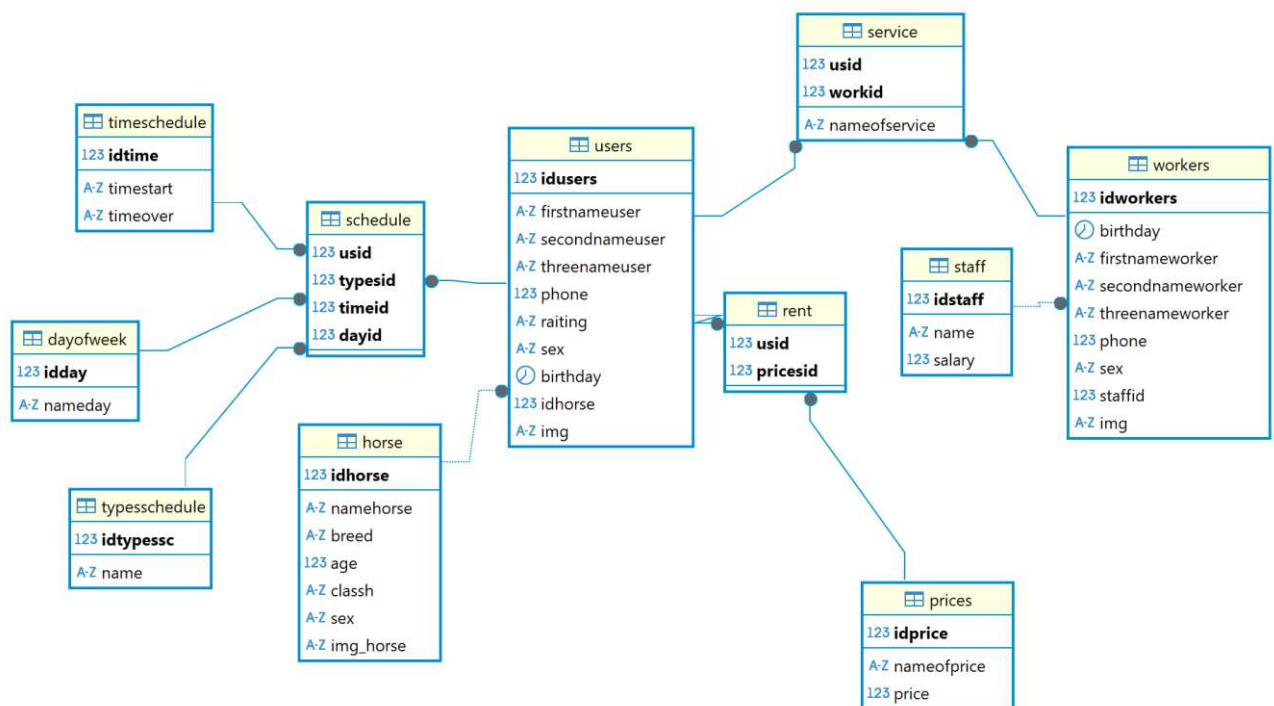


Рис. 2.4. Логічна модель даних

2.1.4 Розробка фізичної моделі даних

Фізична модель даних один із останніх етапів створення власної бази даних. І тут вже описуються дані (сутності, атрибути, зв'язку) засобами конкретної СУБД. За визначенням атрибути стають стовпцями таблиць, для ключових атрибутів створюються унікальні індекси, домени перетворюються на типи даних, прийняті конкретної СУБД. В своєму сенсі, фізична модель даних створюється вже для комп'ютера.

Фізична модель майбутньої БД представлена на рисунку 2.5.

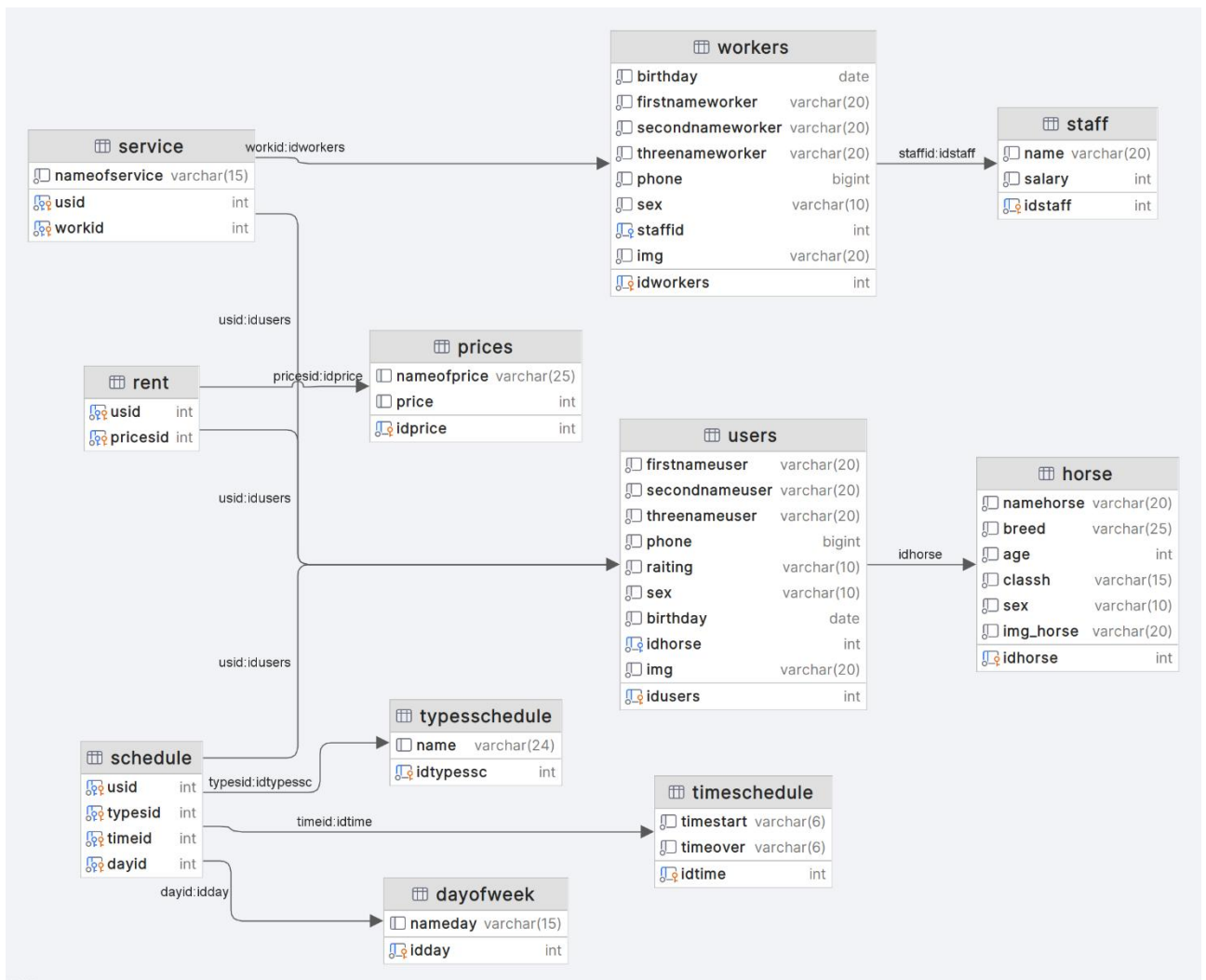


Рис. 2.5. Фізична модель даних

2.2 Рішення з математичного забезпечення

Тренер, коли користуватиметься інформаційною системою, повинен встановлювати рейтинг учня залежно від його здібностей. Розрахунок вестиметься за стандартною процедурою оцінки даних студента при змаганнях та стартах. Результат розрахунку цих даних від тренера може не збігатися в майбутньому при розрахунку результатів пройдених стартів учнем. Розрахунок буде братися в залежності від виду занять, що проводяться. При цьому заняття підготовки до змагань не враховуються у базовому рейтингу. Рейтинг для всіх є загальним, однак рейтинг для новачків не обов'язковий. Рейтинг встановлюється за 100-бальною шкалою і округляється до повного числа.

За кожне тренування тренер повинен виставляти певну кількість балів.

Розрахунок поточного рейтингу представлено за формулою 1.

$$R = \frac{(r_d + r_n)}{2} \quad (1)$$

де R – це загальний рейтинг, r_d – це рейтинг до тренування, r_p – рейтинг тренування, яке сталося.

Рейтинг тренування, що сталося, вважається залежно від тренування, яке відбулося.

Розрахунки щодо виїздки використовують систему розрахунку оцінок за національними правилами виїздки за 2018 рік [4]. Подано вона формулою 2.

$$r_n = r_a + r_p + 2 * r_l + 2 * r_o$$

r_c – оцінка посадки (12,5 балів) и r_o – оцінка за послух (12,5 балів).

Бали виставляються з огляду на помилки.

Розрахунки з конкуру використовують систему розрахунків оцінок протоколу суддівства з конкуру [5]. Подано вона формулою 3.

$$r_n = r_b + r_s + r_r$$

де r_b - це оцінка за правильність відпрацювання перешкод (50 балів), r_s - оцінка за посадку (25 балів), r_r – оцінка за ритмічність їзди (25 балів).

Розрахунок для тренувань з новачками використовують загальний бал, на розсуд тренера за 100-бальною шкалою, і є суто суб'єктивним.

Розрахунок для тренувань із профі залежить від типу тренування, який проводить тренер зі студентом. Тренер сам обирає формулу (виїздка, конкур чи загальний бал загалом).

Умовний алгоритм внесення нового рейтингу студента наведено на рисунку 2.6.

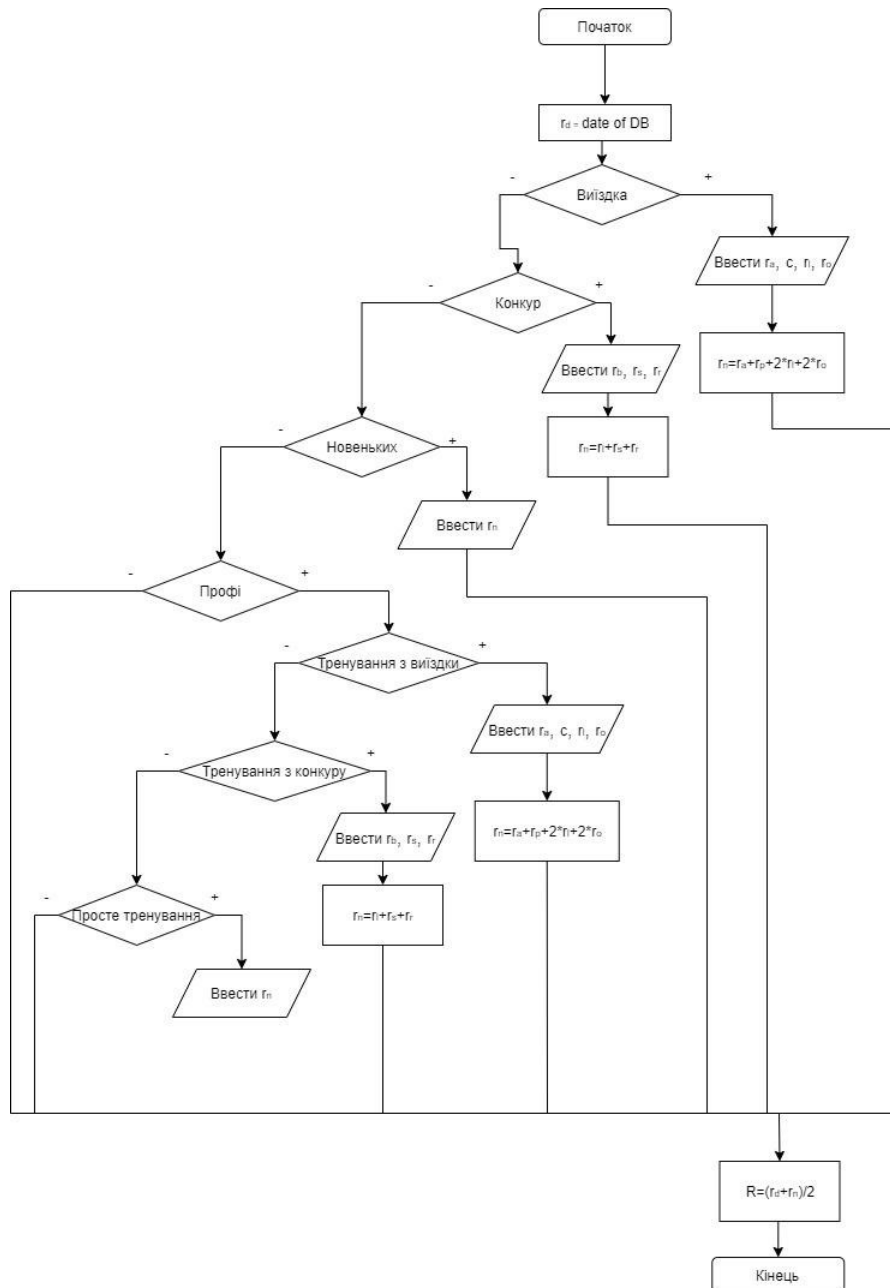


Рис. 2.6. Алгоритм

Даний алгоритм є зразковим, він може змінюватися при розробці і бути використаним у майбутньому при створенні програми для тренерів.

2.3 Програмне забезпечення

2.3.1 Рішення з програмного забезпечення

Для розробки ІС та заданих вище параметрів програмних засобів обираємо рішення з програмного забезпечення.

Розробка рекламного одно сторінкового сайту ІС.

Починаємо з вибору СУБД. MySQL — це вільна система керування реляційними базами даних, яка була розроблена компанією «ТсХ» для підвищення швидкодії обробки великих баз даних[6]. Воно є компактним багато потоковим сервером баз даних. Можливостями СУБД MySQL є такі характеристики:

1. безкоштовне ПО;
2. висока швидкість виконання запитів та команд, що надаються, багатопоточність;
3. легка в розумінні інтерфейсу та можливостей цього інтерфейсу;
4. має необмежену підтримку кількості користувачів, що працюють із БД;
5. оптимізація операцій JOIN;
6. підтримка напівсинхронного механізму реплікації;
7. ODBC драйвер.

Для роботи з цією СУБД використовуємо DBeaver 25. В першу чергу це ПО є комфортним для візуальної роботи з базами даних, в ньому можливе створення схем щодо майбутньої БД, інтегрована генерація SQL скриптів, що полегшую роботу з MySQL.

Код створення таблиць подань у додатку А.

Із базою визначилися. Тепер приступимо до реалізації програмного коду клієнтської та серверної частини. Так як ми створюємо веб-додаток, основна функція якого реалізувати якнайбільше інтерактивностей, щоб залучити клієнтську базу, а так само динамічно оновлюваною інформацією,

що зберігатися в БД, потрібно знайти ті технології, які допоможуть все це створити.

Найперше про що необхідно поговорити, це стандартна для всіх використань мова розмітки HTML та мова опису зовнішнього уявлення CSS. Ці дві складові – це основа основ, за допомогою якого будуватиметься сторінка, а в майбутньому DOM (Document Object Model), про яку ми поговоримо трохи пізніше. HTML позначить зміст сторінки, а CSS врегулює його зовнішній вигляд (у нашому випадку скриптова мова SASS, яка знадобиться для анімацій у Vue).

Для розробки проекту використовуємо середовище розробки JetBrains WebStorm. WebStorm – це інтегроване середовище розробки для JavaScript та пов'язаних із ним технологій. Робота в цьому середовищі полегшується великою кількістю автоматизації рутинної роботи, більш приємним інтерфейсом та врегульованою системою співпраці серверу та клієнту [7]. При цьому ще однією перевагою WebStorm є інтегрована NodeJS, що є полегшеним середовищем виконання JavaScript поза браузером, тобто на сервері чи командному рядку.

Так як ми вибрали середовище розробки WebStorm, переходимо до мови програмування – JavaScript, якою ми будемо користуватися як при розробці сервера, так і клієнтської частини проекту (при спільній співпраці деяких інших технологій). Декілька слів про JavaScript. JavaScript – це динамічна та об'єктно-орієнтована прототипна мова програмування. Вона допомагає створювати динамічно поновлювальний контент, керувати мультимедією, роботи анімації та інше [8].

Клієнтська частина. Функціональність JS полягає в тому, що можна створювати Інтерфейси прикладного програмування або API, які додають ще більше можливостей регуляції структурної частини коду. Наприклад, API інтерфейс DOM, про який говорилося раніше, дозволяє регулювати роботу HTML, CSS, керувати та змінювати їх, коли користувач взаємодіє з

якоюсь частиною сторінки.

Головна проблема DOM в тому, що він практично не розрахований на створення динамічного інтерфейсу користувача. Виникають проблеми з продуктивністю, якщо елементів на сторінці дуже багато. Користувач почне перегортати сторінку, повну інформацію, і завантаження контенту буде досить довгим для сучасного Інтернету, що вплине на ставлення клієнта, і він може втратити інтерес до проекту. І тому є рішення. Так, наприклад, щоб не працювати з DOM з «віч-на-віч» використовується вдосконалений його Virtual DOM. Відбувається порівняння основного DOM та його копії, а потім сторінка перемальовується. Ця технологія є в деяких фреймворках JS. Що таке фреймворк? Фреймворк – це програмна платформа, що визначає структуру програмної системи. Вона відрізняється від бібліотек тим, що впливає на архітектуру програми, сама проектує та розширює програму.

Одним із таких популярних фреймворків є VueJS. VueJS використовує Virtual DOM, про який говорилося вище, надає реактивність та компонентність програми. З офіційної документації можна дізнатися [9], що його ядро насамперед вирішує завдання рівня представлення (view), що спрощує інтеграцію з іншими бібліотеками та існуючими проектами.

Основною перевагою Vue є його компонентний підхід. Тобто, практично весь додаток складається з деяких компонентів, які пов'язані між собою, і кожен такий компонент має свої методи, дані, стилі. Така організація називається дерево вкладених компонентів. Так, наприклад, якщо якась кнопка використовується на сторінці багато разів, то переписувати щоразу HTML-код або копіювати його не потрібно, а використовувати як окремий свій компонент цієї кнопки, який на основній сторінці буде просто оголошуватися в самому коді. При цьому зазвичай такі компоненти будуть називатися дочірніми, а в свою чергу основні – батьківські. Про це необхідно пам'ятати у майбутній реалізації проектного

коду. Також за допомогою Vue відбувається адаптивність верстки (наприклад, зміна звичайного меню на «бургер»-меню, коли користувач використовує мобільний телефон для перегляду сайту). Також використовуємо технологію Vue з анімаціями.

Взаємодія клієнта та сервера є важливою частиною процесу. Про сервер нижче, а зараз вибираємо для отримання та відправлення даних на клієнтській стороні програми бібліотеку Axios. Ця бібліотека з відкритим вихідним кодом, яка надає основні методи передачі через HTTP-запити – `get()`, `post()`. При роботі з Axios ми маємо прямий доступ до JSON(javascript object notation), який ми отримали в результаті прийому інформації з сервера. Так само перевагою Axios над іншими бібліотеками (наприклад, Fetch) є те, що при помилці отримання або віддачі інформації, Axios видаватиме мережеву помилку, що полегшить роботу при розробці сайту.

Одна з частин клієнтської сторони програми – це робота з поштою, тому що клієнт вводитиме потрібну інформацію, а потім вона буде надсилатися на поштову скриньку оператора, який пізніше зможе зв'язатися з користувачем послуг. І тому ми використовуємо сервіс EmailJS. Цей сервіс надає послуги, яка трохи спростить код і зробить його лаконічнішим. Клієнт отримує дані від клієнта, потім відправляє їх на сервіс EmailJS, а той, у свою чергу, за допомогою деяких маніпуляцій та шаблону відправляє готовий лист на пошту оператора.

Серверна частина. Серверна частина також розробляється на JS, проте використовує трохи інші технології. Спочатку доступ до БД. Для доступу до створеної бази даних використовуємо драйвер `mysql2`. Драйвер підключається до бази даних за допомогою заданих параметрів (`host`, `port`, `user`, `password`, `database` тощо), з методів драйвера та використанням запитів витягується потрібна інформація.

Для передачі клієнту інформації використовуємо веб-фреймворк Express. Програма запускає сервер і слухає з'єднання на порті,

встановленому для серверної частини проекту. При цьому воно видає необхідні дані за допомогою методів `get()`, `post()`, що адресовані маршруту. Тут у справу вступає клас `express.Router`, які можуть створювати модульні, монтуючи обробники маршрутів. Якщо не виникають помилки, то перетворюється результат у формат JSON. При цьому пізніше URL маршрут використовується в клієнтській частині за допомогою `Axios`. Таким чином, відбувається динамічний зв'язок між клієнтом та сервером, отримання та відправлення даних між ними. Зразкова робота такого процесу надана на рисунку 2.7.

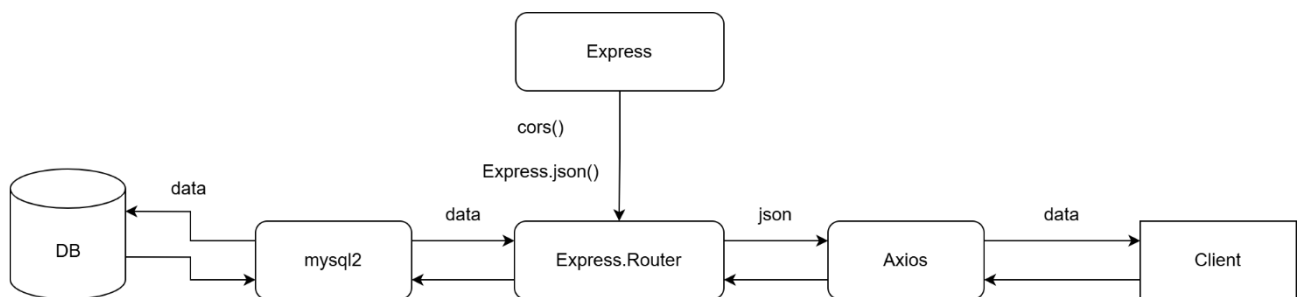


Рис. 2.7. Процес взаємодії сервера та клієнта

При цьому серверну частину проекту можна буде використовувати для подальшого розвитку системи.

Операційні системи

Для розробки нам знадобляться деякі ОС. Оскільки перегляд сайту відбуватиметься і з телефону, і з комп'ютера, вибираємо такі ОС для них – Android (телефон), для комп'ютера – Linux.

Linux – це назва UNIX-подібних операційних систем, основою якого є однойменне ядро. При цьому ми будемо використовувати дистрибутив Mint, який існує на основі Ubuntu та Debian. Основними перевагами Linux Mint є такі характеристики:

- Легка у розумінні та застосуванні;
- Повна мультимедійність;
- Безкоштовна у використанні;

- зручний інтерфейс;
- Встановлені заздалегідь та підтримка зазвичай ПЗ;
- Повна готовність до роботи із самого початку [10].

Android – це операційна система та платформа для телефонів і планшетів, розроблена компанією Google на основі ядра Linux. Основною перевагою цієї системи є її відкритість для користувачів. Більшість телефонів підтримують цю ОС, що сприяє подальшому розвитку в області розробки.

Типові програмні засоби

Для перегляду веб-програми потрібен браузер. Вибираємо Chrome - безкоштовний браузер від Google на основі Chromium. На відміну від інших браузерів, в Chrome кожна вкладка є окремим процесом, що дозволяє закрити завислу вкладку без втрати даних. Браузер включає JavaScript інтерпретатор, консоль і менеджер завдань для перевірки використання пам'яті кожною вкладкою [11].

Консоль JavaScript може знадобитися у майбутній реалізації, бо ми розробляємо веб-додаток на javascript.

Draw.io – це інструмент проектування діаграм, блок-схем, інтелект-карт, відносини сутностей, програмних блоків та інше. Ми використовуємо даний сервіс для полегшення роботи, так як він включає безліч заздалегідь створених шаблонів, володіє багатим набором функцій для візуалізації більшості завдань.

Сервісне програмне забезпечення

Захист інформації відбувається за допомогою антивірусного ПЗ (програмного забезпечення) Avast. Avast може працювати на більшості ОС (Windows, macOS, Linux, а так на Android та IOS, що нам і потрібно). Можливості програми включають в себе перевірку комп'ютера на віруси,

блокування шкідливих скриптів, резидентний захист незалежними модулями (тобто сканер файлової систем, сканер пошти, сканер відвідування вебсайтів, сканер мережевого доступ тощо), видалення шпигунських ПЗ, підтримка голосового повідомлення, якщо виявляється вірус та багато інших [12]. Тобто, avast виконає більшість функцій по захисту системи від зовнішніх та внутрішніх загроз.

Однією з основних частин, пов'язаних із роботою системи, є електронна пошта (зв'язок клієнта та оператора). Тому використовуємо для таких цілей Gmail. Gmail – безкоштовна служба електронної пошти від тієї самої компанії Google, яка створила браузер Chrome. Вона надає доступ до поштових скриньок через вебінтерфейс і за протоколами POP3, SMTP, IMAP. Головною перевагою цієї служби є зберігання 15 гігабайтів простору. Gmail побудований на JS, застосовується технологія AJAX (оновлення сторінок без перезавантаження). Автозбереження, «гарячі клавіші», фільтрація від спаму, підтримка різних мов та багато іншого виділяє Gmail серед інших поштових сервісів [13]. Окрім відділу роботи з клієнтами, цим сервісом також користуються усі в кінноспортивній школі для полегшення зв'язку між відділами.

Підключення у комплексі відбувається за допомогою технології Ethernet, розповсюдження сигналу – за допомогою Wi-Fi. Комутатор – PowerLink SP401F. Роутер зразка Tp-Link. Більшість комп'ютерів підключено до роутерів, проте до деяких немає доступу за допомогою мережних дротів. Тому деякі з них використовують двухдіпазонний Wi-Fi адаптер U&P AC600AC. Швидкість підключення (перевірено за допомогою сервісу 2ip.ua) – 47.37 Мбіт/сек.

2.3.2 Структура програмного забезпечення веб-додатку

Для візуалізації компонентів та об'єктів проекту можемо використовувати технологію UML. Об'єктно-орієнтований підхід представляється з допомогою діаграми класів. Оскільки JS заснований на

прототипах (тобто, це стиль об'єктно-орієнтованої мови, однак у ньому відсутнє як таке поняття класу, а успадкування відбуватиметься шляхом клонування (у більшості випадків)), то структурні діаграми не є особливо привабливими для представлення, проте можна отримати вигоду з концепції діаграми класів, щоб показати наміри проектування або прототип, за допомогою якого відбуватиметься майбутнє створення об'єктів. Виберемо концепт прототипу, тобто покажемо приблизно як виглядатиме майбутній проект у реалізації діаграми класів.

Опишемо дві частини програмного забезпечення рекламного сайту. Серверна частина проекту.

Index – відбувається запуск сервера;

Database – підключення сервера до бази MySQL;

HorseModel, TrainerModel, PricesModel – виконують запити для БД та отримують від неї дані (або помилку, що виникла при виконанні) ;

HorseController, TrainersController, PricesController – приймають дані, перетворюють дані на JSON;

Routes – налаштування маршрутів, які приймає дані в JSON; Серверна частина проекту представлена на рисунку 2.8.

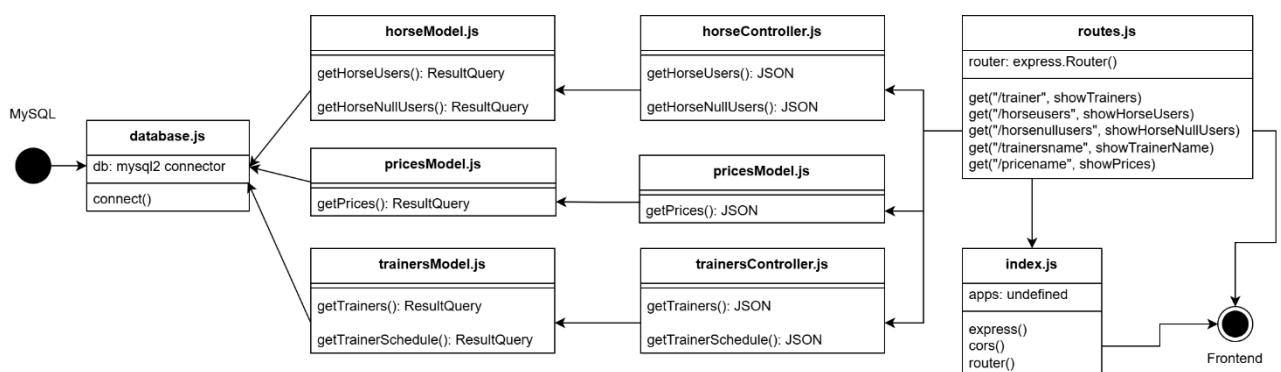


Рис. 2.8. Сервер

Клієнтська частина проекту.

Main.js – зберігає у собі інформацію про маршрути, запускає сервер клієнта;

App.vue – екземпляр, який ловить маршрути для відображення сторінки;

MainPage.vue – компонент, який відображає важливі та не дуже інші компоненти, використовується для відображення меню та скролінгу по сторінці;

FooterPart.vue – нижня частина сторінки;

HorseStage.vue – компонент, який зберігає в собі інформацію про вільних (BlockHorseNull.vue) та зайнятих коней (BlockHorseNotNull.vue), а також студентів, які їх використовують;

AboutStage.vue – компонент, що зберігає стартову, презентуючи інформацію комплексу;

PeculStage.vue – компонент, який відображає переваги комплексу у вигляді інтерактивного меню (PeculPart.vue). При цьому в цьому компоненті є зв'язок із компонентним за допомогою кнопки, в якому прораховується, користувач зможе прорахувати суму вибраних послуг (PricesCalculate.vue) ;

TrainerStage.vue – компонент, який включає всіх тренерів комплексу (TrainerNew.vue – приймає інформацію щодо тренерів новеньких, TrainerProfi.vue – приймає інформацію щодо тренерів профі, TrainerOne.vue – приймає інформацію щодо тренерів виїздки чи конкуру (в залежності від необхідного). При цьому цей компонент має зв'язок з компонентним ModalTrainer.vue за допомогою кнопки, яка в модальному вікні відображає повну інформацію обраного тренера;

ScheduleStage.vue – компонент, який відображає актуальний графік тренерів. При цьому має зв'язок з PartSchedule.vue, в якому відображаються тренери, а далі ще один компонент, який вже відображає сам розклад за

вказаними днями тижня (ScheduleWeek.vue);

FeedbackStage.vue – компонент, який необхідний для відображення форм зв'язку клієнта та сервісу. Email.vue – компонент-форма, до якої користувач вводить дані.

AlertTools.vue – пов'язаний з минулим компонентом шляхом відображення помилок та успіху виконання відправки даних (при цьому цей компонент, як прототип, можна використовувати в майбутньому розширеному веб-додатку або за інших умов). Email.vue також пов'язаний з бібліотекою emailjs;

HorseStage.vue, Trainer.vue, PeculStage.vue, ScheduleStage.vue – компоненти, які мають зв'язок із axios. Тобто, за допомогою axios клієнт пов'язаний із сервером.

Клієнтська частина проекту представлена рисунку 2.9.

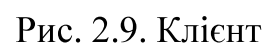


Рис. 2.9. Клієнт

Ще однією візуалізацією роботи проекту – це діаграма стану (поведінкова діаграма UML). Вона вже більше підходить для проектів, написаних на JS, тому що ілюструє стани та переходи.

Розпишемо більш детально деякі з процесів на клієнтській стороні проекту.

Користувач заходить на сторінку, відбувається анімація Vue у елементів (поява, переміщення до кінцевої точки). Користувач звертає увагу на меню і вирішує натиснути кнопку, яка переміщає його на частину сторінки, яка пов'язана з конями. Позиція користувача стає плавно на цій частині, при цьому анімації вже відпрацьовані. Клієнт помічає кнопки, які перемикають обидва поля – коней зі своїми власниками та вільних коней. Він натискає кнопку перегортання вправо. Відбувається плавна анімація зникнення та появи того поля, на якому розташовувалася кнопка. При цьому інше поле залишається в тому ж стані, дії кнопки не поширюється. При цьому під час перемикавання діє метод, який витягує потрібне зображення коня чи користувача. Таким чином, одна з частин сторінки змінює свій стан від натискання кнопки.

Якщо, наприклад, користувач вибрав частину з тренерами, його позиція змінюється на позицію частини сторінки тренерів. Він натискає на іконку тренера, відбувається плавне відкриття вікна (змінюється віртуальне DOM дерево, модальне вікно стає видимим для користувача). При цьому, коли користувач натискає кнопку закриття вікна, побудова дерева повертається до кінцевої точки, яка була.

Таким чином працюють процеси на нашій сторінці. Залежно від вибору користувача будуватиметься сторінка, змінюватиметься, анімуватиметься та інше.

Приклад роботи веб-додатку в діаграмі стану та переходів представлений на рисунку 2.10.



Цей стан підлягає і до верстки для телефону (оскільки всі функції зберігаються), однак є доповнення до контекстного меню, яке представлене на рисунку 2.11.

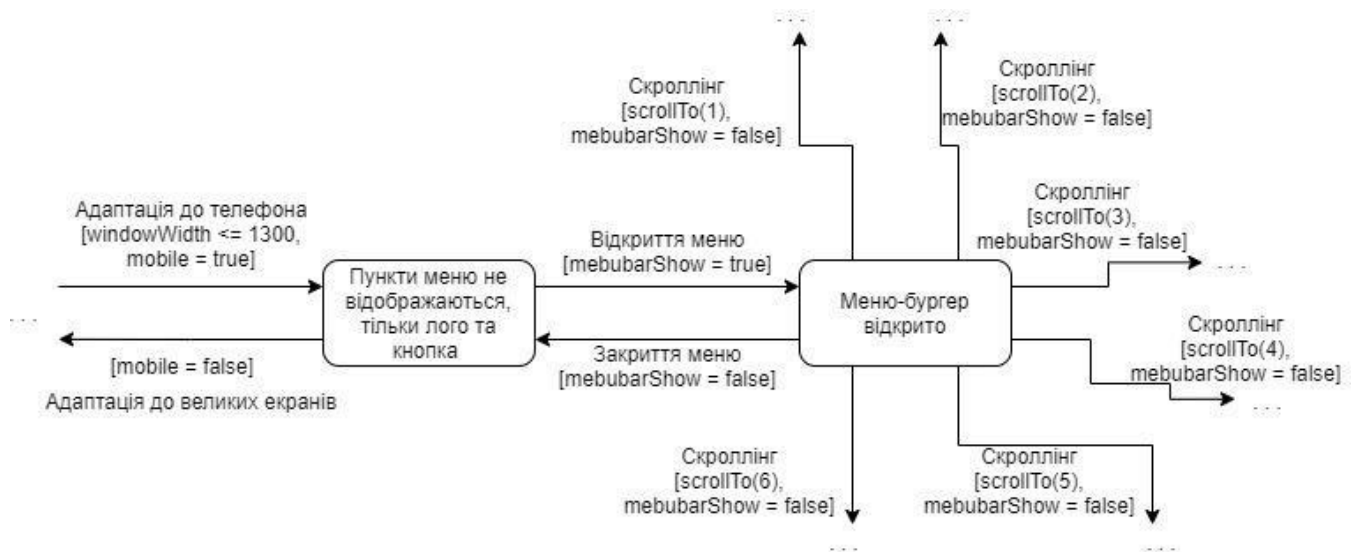


Рис. 2.11. Доповнення до діаграми станів

Процес полягає в тому, що при впливі на розміри сторінки користувачем (розширення браузера, захід через телефон) верхня шапка змінюється. Користувач натискає кнопку відображення меню в цій шапці, плавно з'являється меню. При натисканні на один з пунктів відбувається переміщення користувача по сторінці, а при цьому меню так само плавно закривається.

2.4 Технічне забезпечення проекту

Оглядаючи системні вимоги майбутньої реалізації, наведемо технічне забезпечення. Так само не варто забувати, що на клієнтській частині додатка можна зберігати дуже багато зображень (які пов'язані або з базою даних сутностей або ж для простих декоративних елементів)

- Системний блок: QUBE QB Miditower 500Вт;
- Процесор: Intel Core i3-10100F 4 ядра 3,6 (4,3) ГГц;
- Пам'ять: DDR4 SSD 1Тб 8 гб 2666 МГц;
- Відеокарта: Nvidia GeForce GTX 1650 4 гб;
- Миша: MSI Cultch GM11;

- Клавіатура: Genius LuxeMate i220; Монітор: HP M27fd;
- Мережевий адаптер – Intel® Ethernet Connection (12) I1219-V.
- Смартфон для перевірки верстки: Xiaomi Mi Note 10 Lite.

Так само для роботи комп'ютерів при стрибках напруги (щоб техніка не виходила з ладу через це), кожен з них має бути оснащений стабілізатором потужності – Logic Power.

Висновок

У даному розділі розглянуто та реалізовано компоненти та технології що є важливими для роботи системи, а саме як апаратні, так і програмні аспекти. Розроблені та описані компоненти обладнання забезпечують стабільну роботу створеного застосунку, включаючи захист від стрибків напруги та багатофункціональність. Вибір техніки та оптимізація її використання дозволяють створити умови для розробки якісного продукту, які будуть надалі використані під час реалізації проекту, детально описаного у розділі 3.

РОЗДІЛ 3. РЕАЛІЗАЦІЯ ПРОЕКТУ ВЕБ-ДОДАТКУ

У цій дипломній роботі була розроблена ІС центру кінного спорту, яка призначена для подальшого розвитку компанії з автоматизованим направленням.

Реалізуємо частину інформаційної системи, а саме веб-додаток рекламного призначення. Вона включає в собі такі модулі:

- Верхній колонтитул, який містить меню;
- Нижній колонтитул;
- Модуль сторінки, який презентує та вітає користувача (представлено на рисунку 3.1).



Центр кінного спорту ВЕРХОВИНА

ПРО НАС

ПОСЛУГИ

ГАЛЕРЕЯ

КОНТАКТИ



Рис. 3.1. Привітаюча клієнта частина

- Модуль сторінки особливостей кінного центру (представлено на рисунку 3.2);

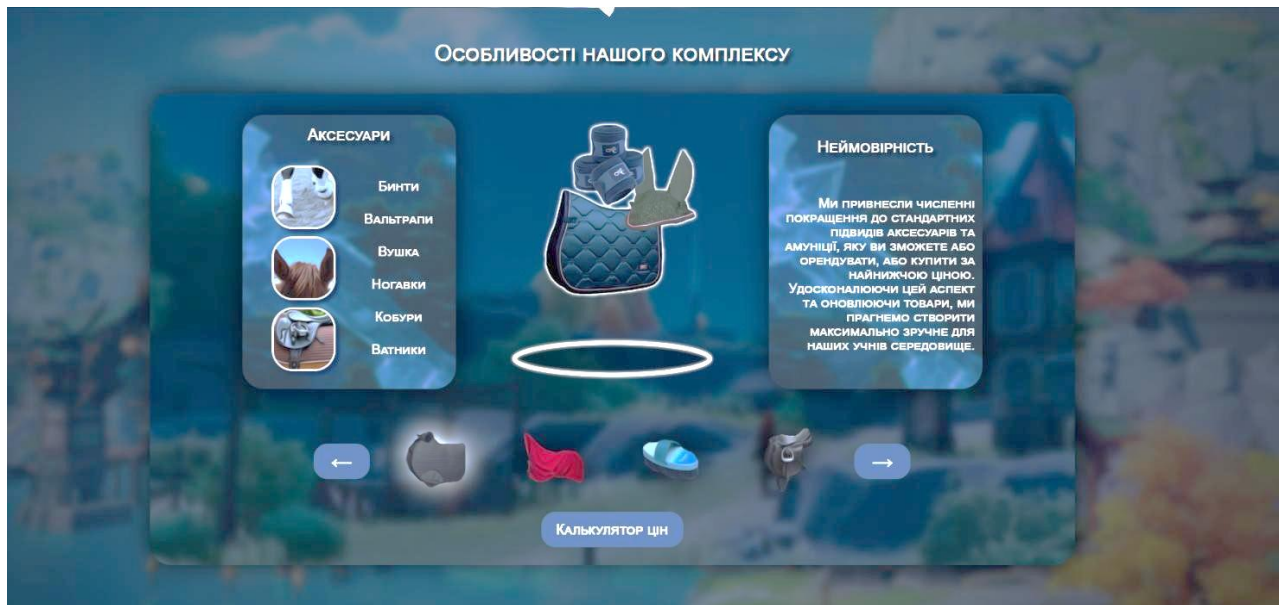


Рис. 3.2. Частина про особливості

- Модуль сторінки, який відображає коней, з їх власниками чи вільних (представлено на рисунку 3.3);



Рис. 3.3. Частина про коней

- Модуль сторінки, який відображає тренерів комплексу (представлено на рисунку 3.4);

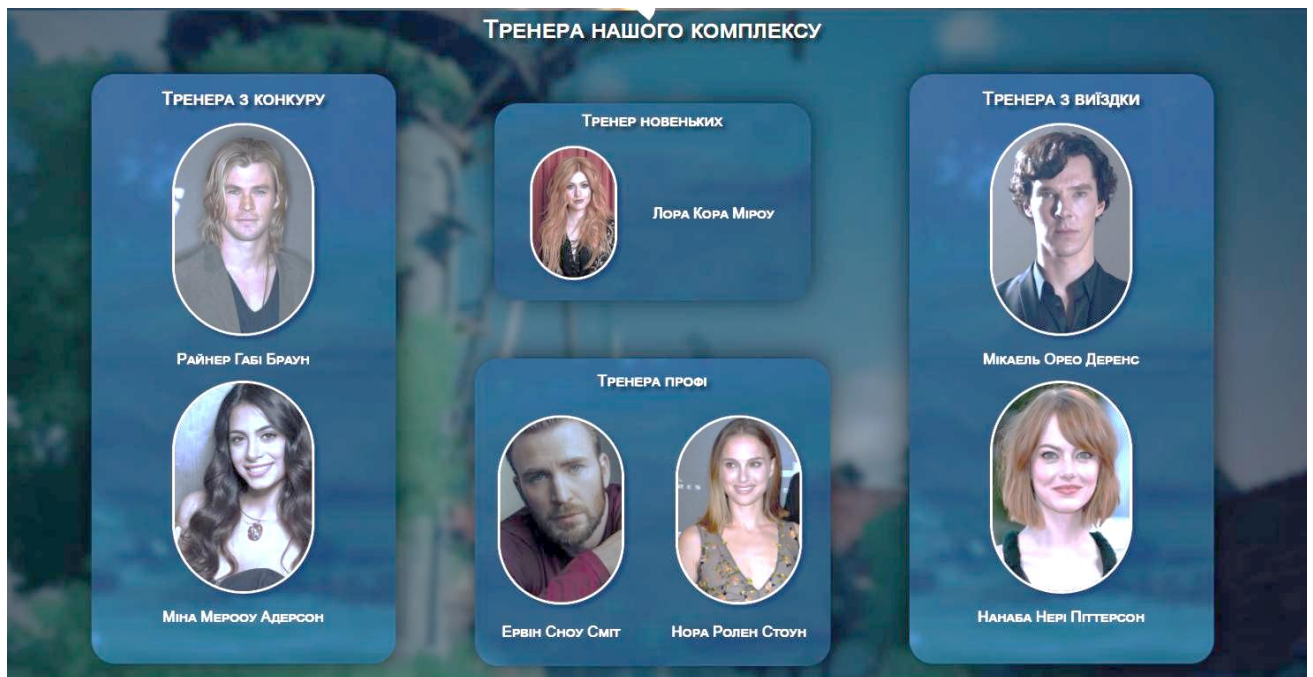


Рис. 3.4. Частина про тренерів

- Модуль сторінки, який відображає актуальний розклад тренерів в залежності від дня тижня (представлено на рисунку 3.5);

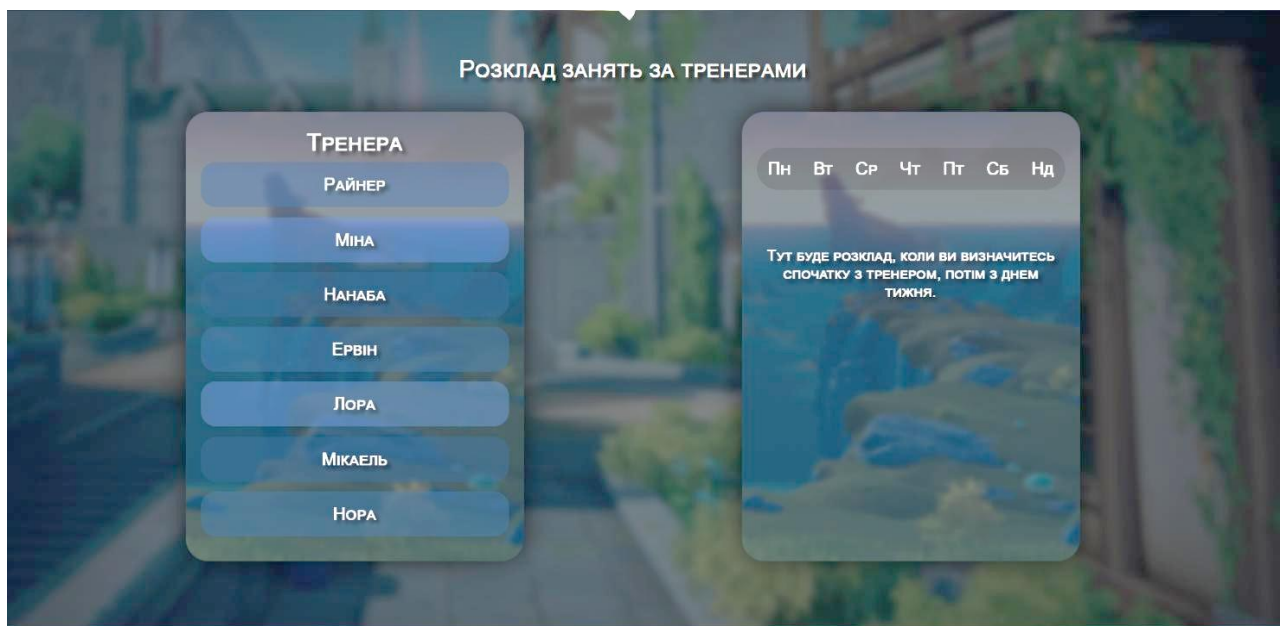


Рис. 3.5. Частина про розклад

- Модуль сторінки зворотного зв'язку (представлено на рисунку 3.6).

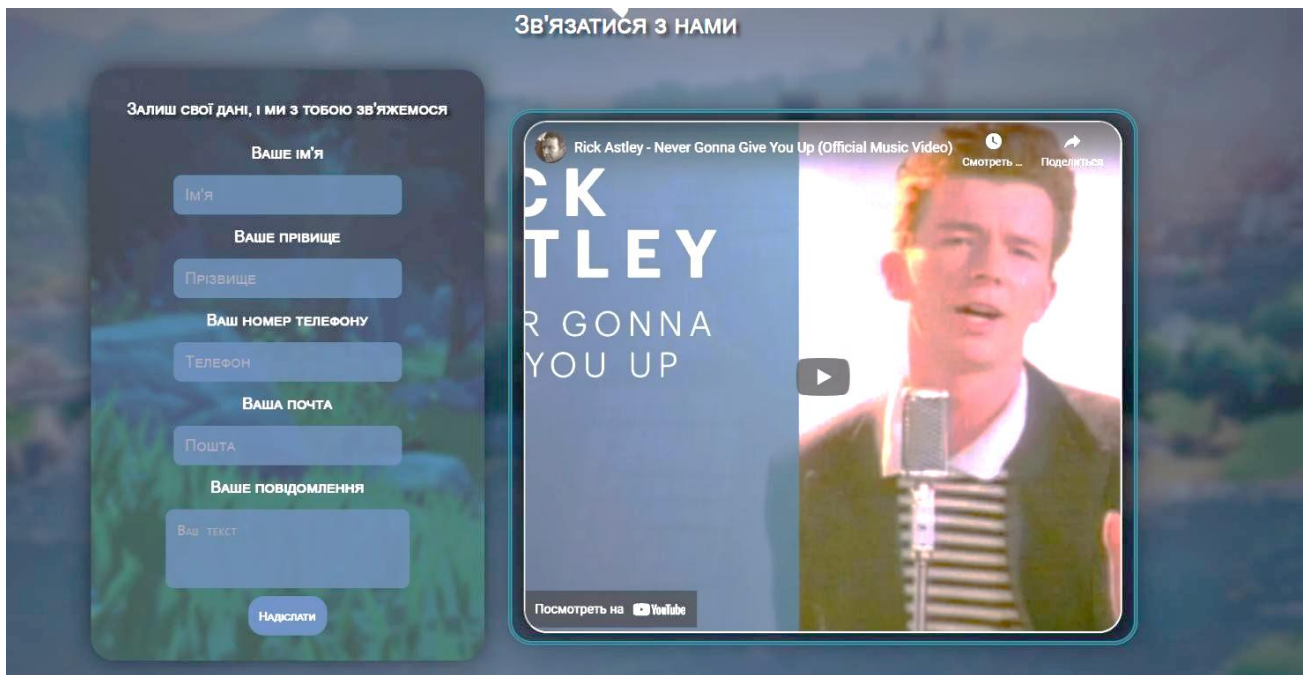


Рис. 3.6. Частина зворотного зв'язку

Ця сторінка повністю генерується при вході на неї та може змінюватися від дій користувача. Всі частини розташовані одна під одною. Прикладами змін є:

- Зміна елементів у особливостях школи (представлено на рисунку 3.7);



Рис. 3.7. Зміна особливостей

- Модальне вікно калькулятора цін (представлено на рисунку 3.8);

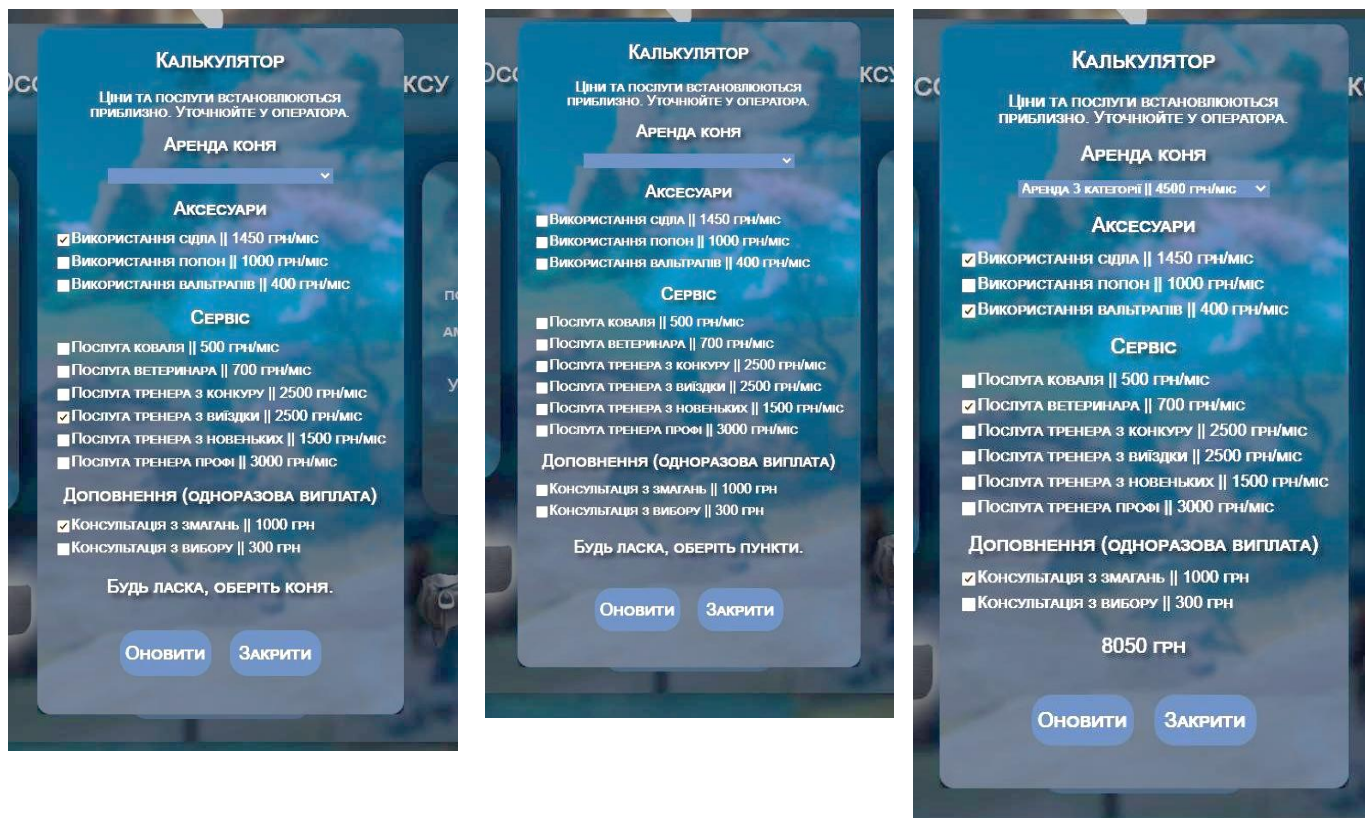


Рис. 3.8. Калькулятор

Модалне вікно з повною інформацією щодо тренера (представлено на рисунку 3.15);

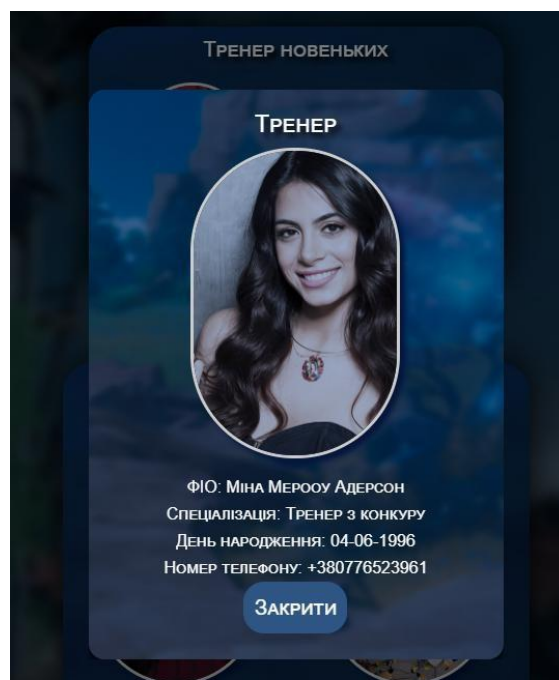


Рис. 3.9. Модалне вікно тренера

Зміна інформації при виборі тренера та дня тижня (представлено на рисунку 3.10-3.11);

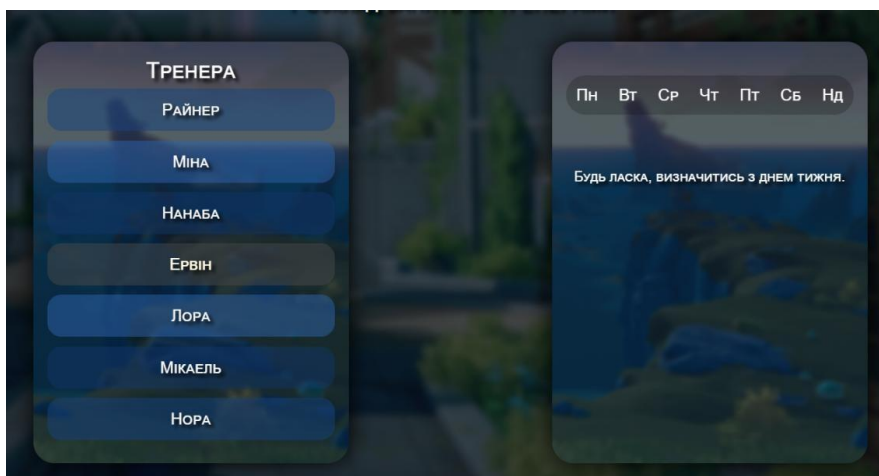


Рис. 3.10. Відображення розкладу без вибраного дня



Рис. 3.11. Відображення розкладу з днем

Інструкція користувача

Інтерфейс сайту інтуїтивно зрозумілий для будь-якого користувача, який заходитиме на нього. Кольорове вирішення виконані в холодних тонах, на всій сторінці використовується один шрифт, всі зображення мають схожі стилі (обведення білим контуром, тіні). Сторінка, як у технічному описі, має кілька розділів: навігаційне меню (під комп'ютер і телефон), розділ опису, розділ коней, розділ тренерів, розділ особливостей, розділ розкладу, розділ зворотного зв'язку, а так само нижній колонтитул. На сторінці також реалізований скролінг до певних частин сторінки для швидкого та комфортного переміщення позиції користувача на ній. Дані, які витягуються з бази даних за допомогою сервера та передаються клієнту, відображаються на сторінці коректно, динамічно. Деякі структурні частини мають плавну анімацію появи, зникнення, переміщення, зміни. Є компоненти сайту, які виконують такі функції:



При натисканні на кнопку змінюється положення користувача (залежить від вибраної теми);



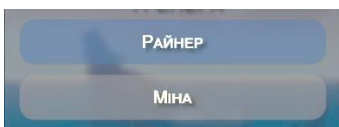
При натисканні на кнопку відбувається перегортання ліворуч або праворуч, залежно від стрілки в назві;



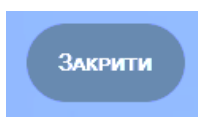
При натисканні на подібні кнопки із зображенням змінюється інформація на панелі особливостей (вибрана кнопка підсвічується);



При натисканні кнопки відбувається відкриття модального вікна з калькулятором цін;



При натисканні кнопки з логотипом сторінка оновлюється;



При натисканні на кнопку у багатьох модальних вікнах відбувається закриття вікна;

Інструкція формування, введення БД та запуску роботи сайту

За допомогою документації школи (наприклад, повний список студентів, тренерів, список цін за послуги, розклад тренерів та студентів) вводяться дані до СУБД MySQL за допомогою MySQL WorkBench. У майбутньому, при реалізації додатка для тренерів, дані про розклад та учнів можуть запроваджуватися тренерами, а також їм зміняться. При видаленні даних з бази, інформація заноситиметься в архів, після чого зберігатиметься невизначений термін (директор встановлює час зберігання інформації залежно від його виду). Розклад оглядається кожного тижня, за потреби додається, оновлюється та видаляється. Перевірку даних на актуальність проводить системний адміністратор щосуботи кожного тижня, звіряється з архівом. Він же стежить за зберіганням, використанням, а також пам'яттю.

Запуск сервера провадиться командою `node index` у терміналі. Запуск клієнта виконується командою `npm run serve`.

Перезапуск сервера та клієнта відбувається в останню неділю місяця опівночі і триватиме до трьох ночі понеділка. В цей час системний адміністратор вносить якісь зміни.

Підготовка до роботи

На початку роботи системи системний адміністратор має познайомитися із системою загалом, провести аналіз технічної документації проекту. Тренери та оператори, які використовуватимуть інформаційну систему, також мають бути ознайомлені з технікою роботи з сайтом, знати вади роботи з нею. При створенні в майбутньому розширеного функціоналу, що включає вхід в обліковий запис, передбачає реєстрацію тренерів, операторів, щоб вони входили в свій обліковий запис з повною функціональністю, необхідною для роботи (при цьому обліковий запис користувача буде відрізнятися).

Аварійні ситуації

Можуть бути незаплановані роботи та вимкнення сервера з клієнтом, перед цим користувачів попереджають рядком, що біжить. Це так само може стати в нагоді в майбутньому розширенні функціональності рекламного сайту і перетворення його на повноцінне веб-додаток.

При відключенні електроживлення, використовується джерело безперебійного живлення. При нестабільному електричному напрузі використовується стабілізатор. При порушенні роботи комп'ютер викликається майстер, що перевіряє роботу всіх складових комп'ютера, а при виявленні помилок або поломок, надає послуги ремонту або купівлю нових частин, що зламані. При проблемах з роботою інтернету викликається майстер провайдера, який або документує, що необхідно зробити для ремонту (лагодженням займається системний адміністратор), або прибуває в школу для подальшого ремонту спорядження.

Висновок

У розділі було детально описано процес формування бази даних, порядок введення даних, підготовку до запуску системи та її функціонування, а також заходи реагування на аварійні ситуації. Розглянуто основні технічні аспекти роботи сайту, включаючи періодичне оновлення розкладу, роль системного адміністратора та методи захисту від непередбачуваних перебоїв. Описані механізми підтримки стабільності системи та забезпечення її ефективної роботи в умовах реальних викликів.

РОЗДІЛ 4. ОХОРОНА ПРАЦІ ТА ТЕХНІКА БЕЗПЕКИ ПІД ЧАС НЕЇ

4.1 Поняття охорона праці на державному рівні

Охорона праці - це система правових, соціально-економічних, організаційно-технічних, санітарно-гігієнічних і лікувально-профілактичних заходів та засобів, спрямованих на збереження життя, здоров'я і працездатності людини у процесі трудової діяльності [14].

Охорона праці характеризується об'єктом, а саме дослідженням праці людини та середовища, де вона працює (виробництво, організація). З приводу цього дослідження проводяться заходи та засоби, які зберігають здоров'я, працездатність та життя людини у середовищі праці.

Конституція України та Верховна Рада України визначають умови охорони праці та здійснюють здорові та незалежні умови при роботі працівника (тобто запобігання нещасних випадків та професійних захворювань). При встановленні документів Незалежності України, 14 жовтня 1992 року Верховна Рада України встановила закон «Про охорону праці» за постановою ВР №2695- XII.

Закон «Про охорону праці» встановлює, що при порушенні законодавчих чи інших нормативних актів про охорону праці, а саме розділ VIII, будуть визначатися такі види відповідальності: дисциплінарні, адміністративні, матеріальні чи кримінальні. При цьому, відповідальність мають нести як і підприємство, так і працівники.

Основними видами порушень можуть бути такі характеристики:

- Не проведення відповідних інструктажів перед початком роботи;
- Відсутність спеціальних засобів безпеки працівників;
- Незнання техніки безпеки працівником чи погане його освоєння тощо.

У розділі IV – «Стимулювання охорони праці» говориться, що

відшкодування збитків за порушення правил охорони праці згідно з діючим законодавством, витрати на рятування потерпілих під час аварії та ліквідації її наслідків, на розслідування її причин, відшкодовує роботодавець[15].

Актуальність охорони праці має бути в тому, щоб звести до мінімуму поразки робітників у процесі праці, під дією шкідливого виробничого впливу, забезпечити найкращі умови для гарної продуктивності робочих процесів.

Держава вносить свої корективи у впроваджені охорони праці: пріоритетність життя та здоров'я працівника, відповідальності роботодавця за створення безпечних умов праці; забезпечення технічного контролю за станом виробничої, технологічної сфери; розв'язання задач щодо охорони праці за загальнодержавними програмами з цього питання, при цьому враховуючі економічні, соціальні та екологічні напрямки політики; соціального захисту працівника, відшкодування шкоди особам, що потрапили у нещасний випадок під час роботи та професійних захворювань; проведення навчання та професійної підготовки з підвищенням кваліфікації працівників; підвищення безпеки праці на основі міжнародного співробітництва.

Охорона праці має неабияку важливість у суспільстві. Здоров'я та життя – це найголовніше, що існує у людини, тому потрібно слідкувати, щоб ці два пункти завжди були в безпеці. Ніякі досягнення, гроші та здобутки не можуть повернути життя. Працівники – це насамперед люди, у яких окрім роботи є ще родина, тому потрібно підтримувати захист та підтримку в цьому, не зважаючи на витрачені на безпеку гроші.

Рентабельність у проблемі охорони праці та її успіх завжди залежатиме від якості проведених заздалегідь технік безпеки працівників, підготовки фахівців так, щоб пізніше вони змогли приймати швидкі та дієві рішення при небезпечних для життя робочих моментах.

А, у зв'язку з тим, що технологічний і науковий прогрес пішов далеко вперед, різних факторів, які можуть загрожувати життю стало ще більше, ніж раніше, потрібно завжди враховувати ці моменти при складанні графіка – працівник не повинен перетруджуватися і тим більше загрожувати своєму життю якимись робочими ситуаціями. Режим, правильність і встановлення правил – запорука успішної охорони праці.

4.2 Аналіз небезпечних та шкідливих факторів у Центрі кінного спорту «Верховина» та шляхи їх усунення

Основними законодавчими актами, якими керується центр кінного спорту «Верховина», коли організовує охорону праці та забезпечує безпеку життєдіяльності своїх працівників, є Конституція України, Кодекс законів про працю та Закон України «Про охорону праці».

З урахуванням того, що кінний спорт вважається одним із самих небезпечних видів спорту, потрібно враховувати будь-які небезпечні чинники, з якими може мати справу працівник підприємства.

Беремо за основу роботу системного адміністратора, який працює за комп'ютером в приміщенні.

Небезпечними, шкідливими виробничими факторами та її санітарними нормами при цьому є (їх може бути більше, але ми продивимось тільки ті, з якими може зіткнутися адміністратор при роботі):

1. Хімічні:

– по характеру впливу на людину: токсичні (викликають отруєння організму, у школі можуть виникнути при роботі з різними подразниками, коли навіть системний адміністратор може проходити біля,) дратівні (викликають алергію);

– по шляху проникнення у організм людини: проникаючі через органи дихання, шлунковокишковий тракт, шкіру та слизисті оболонки (при близькій роботі з тваринами це легко може статися);

2. Фізичні:

– Температура, вологість, вентиляція повітря, випромінювання, що в цілому називають мікроклімат. При ненормативній температурі можуть виникати процеси у організмі людини (збереження чи віддача тепла), що поглинає дуже багато енергії, а це вже вплине на самопочуття, а за ним і на її роботу. Як приклад, висока температура повітря - адміністратор дуже швидко стомлюється, його продуктивність падає під час короткого проміжку часу, може призвести до перегрівання організму, теплового удару. Низька температура повітря - викликає місцеве або загальне охолодження організму, може стати причиною простудного захворювання або обмороження. Дуже висока вологість може призвести до замикання прилад, що стоять у офісі. Параметри мікроклімат для приміщень: у холодний період року нормою температури є 22-24 градуси Цельсія, вологість при цьому 40-60%; у теплий період року – температурі є 23-25 градуси Цельсія, вологість така сама, як і в холодний період, які визначені за постановою ДСН 3.3.6.042-99 Міністерством Охорони Здоров'я України.

– Постійність електричних полів та випромінювання погіршують самопочуття, у працівника можуть виникнути головні болі, а при довгому контакті навіть викликати ракові пухлини. Небезпека електричного струму - людина не може самотійно без спеціальних приладів виявити напругу дистанційно. Робота комп'ютерів та периферійних пристроїв можлива завдяки електричному струмові, до джерел якого вони підключаються, тому існує небезпека ураження користувача струмом. Щоб уникнути нещасних випадків варто строго дотримуватися правил електробезпеки. Ураження струмом може призводити до різних порушень в організмі. Параметри електричного випромінювання: гранично допустимі рівні напруженості електричного поля визначаються в залежності від частоти (довжини хвилі) і режиму випромінювання за наказом №239 Міністерством охорони Здоров'я України;

– Рівень шумів, звуків та вібрацій. В залежності від часових

характеристик, шум підрозділяється на постійний та непостійний. Постійним вважається шум, рівень звуку якого за 8-годинний робочий день змінюється у часі не більше ніж на 5 дБ. Непостійний – більше ніж на 5 дБ. Нормуванням постійного широкосмугового шуму на робочих місцях проводиться по шкалі шумоміра А, яка відповідає середньої чутливості вуха людини. При постійному шумі погіршується гострота слуху, підвищується кров'яний тиск, дуже знижується увага. Параметри шумів, звуків та вібрацій: допустимі рівні звуків тиски згідно ГОСТ 12.1.001-83 на робочих місцях не повинні перевищувати таких значень: 12кГц – ≤ 80 дБ; 20кГц – ≤ 90 дБ; 25кГц – ≤ 105 дБ; 31,5 –100кГц – ≤ 110 дБ. Допустимі рівні ультразвуку у зонах контакту рук або інших частин тіла людини з робочими органами приладів та установок не повинні перевищувати 110 дБ. Згідно до СН 22.74-80 нормуються рівні звукового тиску у октавних смугах середньо геометричних частот: 2, 4, 8, 16 Гц. Допустимі рівні інфразвуку для даних частот не повинні перевищувати 105 дБ.

3. Освітлення

– Недостатність природного та технічного освітлення приміщень, де триває робота. Інформація, яку людина отримує під час життя, у дев'яносто відсотках випадків має за допомогою органів зору. При поганій освітленості чи навпаки дуже яскравого світла виникають такі процеси: стомлення очей; розлад центральної нервової системи; зниження розумовою та фізичною працездатності; травматизм при роботі; деформація глазного яблука. Причинами такого можуть бути – сильна блискучість поверхонь, яркі кольори (або навпаки дуже темні), швидко змінювана освітленість. При поганому освітленні людина швидко втомлюється, працює менш продуктивно, зростає потенційна небезпека помилкових дій і нещасних випадків. Згідно із статистичними даними, до 5% травм можна пояснити недостатнім або нераціональним освітленням, а в 20% воно сприяло виникненню травм. Врешті, погане освітлення може призвести до професійних захворювань, наприклад: короткозорість, спазм акомодатії.

При надмірній яскравості джерел світла та оточуючих предметів може відбуватись засліплення працівника. Нерівномірність освітлення та неоднакова яскравість оточуючих предметів призводять до частішої переадаптації очей під час виконання роботи і, як наслідок цього – до швидкого стомлення органів зору.

Також до небезпечних факторів відносять психофізіологічні фактори. Емоційна напруга, динамічні перевантаження, тяжке положення тіла під час виконання робочих процесів, надмірно швидкий темп роботи. А тривале знаходження в одній позі, при сидячій праці, може призвести до захворювань с хребтом (Повний перелік небезпечних та шкідливих виробничих факторів дається у ГОСТ 12.0.003-74).

Між шкідливими і небезпечними виробничими факторами спостерігається визначений взаємозв'язок. У багатьох випадках наявність шкідливих факторів сприяє прояву травмонебезпечних факторів.

4.3 Розробка заходів, які зменшують вплив шкідливих факторів

Проаналізувавши негативні фактори, що впливають на працівників, встановимо шляхи вирішення та послаблення проблем, які можуть виникнути:

Освітлення. Кабінети та робочі кімнати повинні мати природне освітлення з застосуванням штучного освітлення. Люмінесцентні лампи типу ЛБ повинні застосовуватися в якості джерела світла при штучному висвітленні. При світлому фарбуванні інтер'єру завдяки збільшенню кількості відбитого світла рівень освітленості підвищується на 20 – 40%, різкість тіней зменшується, покращується рівномірність освітлення. Поверхні, що добре освітлюються і знаходяться в полі зору, краще фарбувати в кольори середньої світлості, коефіцієнт відбивання яких знаходиться в межах 0,3-0,6, і бажано, щоб вони мали матову або напівматову поверхню. Для забезпечення нормованих значень освітленості

в приміщеннях використання ЕОМ варто проводити чищення стекол віконних рам і світильників не рідше двох разів у рік і проводити своєчасну заміну перегорілих ламп. Зальне освітлення потрібно робити за допомогою суцільних або переривчастих ліній світильників, які потрібно розташовувати збоку від робочого місця, на рівні з зором працівника.

Мікроклімат. Потрібно провітрювати приміщення. Найкраще це робити перед стартом робочого дня, як приклад, на десять хвилин відчиняти вікна. Вентиляція приміщення повинна бути і подавати чисте зовнішнє повітря. При спекотній погоді необхідно включати кондиціонери, щоб стабілізувати температуру в приміщенні та зробити її оптимальною для роботи. Кондиціонування повітря досягається підтримкою у виробничих або побутових приміщеннях незалежно від зовнішніх метеорологічних умов відповідної температури, вологості, чистоти і швидкості руху повітря. У холодну пору року необхідно підтримувати температуру за допомогою водної, парової, повітряної або комбінованої системи опалення.

Шум. При рутинній роботі системного адміністратора має бути тиша. До початку будівництва приміщення потрібно проектувати кімнату так, щоб, закриваючи двері, зовнішні звукові показники були мінімальними. Якщо ж приміщення орендується, то найкраще розташовувати кабінет подалі від зон, де можуть бути гучні робочі процеси (як варіант, подалі від плаців, загонів, стайні). Раціональне планування приміщення і розміщення устаткування є важливим чинником, що дозволяє знизити шум.

Електрична безпека. Це дуже важливий фактор, оскільки системний адміністратор має справу з електронними приладами. Необхідно, щоб були такі засоби захисту:

- електрична ізоляція;
- захисне заземлення приладів;
- контроль за аварійним станом приладів;

- електричний поділ мереж;
- безпечне вимкнення приладів.

Алергія, отруєння. Якщо працівник є алергіком, йому необхідно заздалегідь про це повідомити, коли його беруть на роботу, щоб у майбутньому він міг уникати якихось зовнішніх подразників. При отруєнні на робочому місці (як харчовим, так і іншим) спочатку необхідно викликати лікаря, який є у школі, який дасть працівникові першу медичну допомогу, а після цього викликати швидку(за телефоном 103), незалежно від стану робітника.

Електромагнітне випромінювання. Зменшення впливу такого випромінювання призводить до правильного вибору моніторів при початковому оснащенні приміщення ними. Монітори, які відповідають останнім стандартам TCO'99 та MPR II є найбезпечнішим для здоров'я людини.

У процесі роботи з комп'ютером необхідно дотримуватися правильного режиму праці і відпочинку. В протилежному випадку у працівника може відзначатися значна напруга зорового апарату з появою скарг на незадоволеність роботою, головні болі, дратівливість, порушення сну, утому і хворобливі відчуття в очах, у попереку, в області шиї і руках.

Забороняється у будь-якому вигляді ремонтувати зламані прилади, комп'ютери та інші електронні прилади. Цим мають займатися спеціалізовані фахівці, які викликаються заздалегідь. Якщо виникають проблеми з підключенням до мережі, можливий варіант дзвінка провайдеру, щоб системний адміністратор і тільки він зміг за допомогою оператора провести невеликі ремонти в роботі мережі. При складних поломках це має робити спеціаліст.

Вимоги безпеки роботи:

- Якщо адміністратор заходить до приміщення його робочого місця після

перебування у зоні роботи з тваринами, йому необхідно ретельно вимити руки;

- Для того, щоб не нести бруд та інші мікроби в приміщення, у кожного працівника, який знаходиться в цьому помешканні, має бути змінна пара взуття;
- Планове прибирання приміщення має проводитися щосуботи, проте працівник повинен протягом робочого тижня стежити за порядком і не запускати свій стіл (протирати пил, не смітити);
- Системний адміністратор повинен раз на три місяці повністю проходить по всіх системних блоках комп'ютерів у приміщенні, щоб очистити їх від забруднень та пилу;
- Під час роботи потрібно робити перерви. У ці перерви працівник може пройти до кімнати відпочинку, де є все необхідне обладнання (дивани та інше). При цьому працівник повинен не забувати проводити тренування для очей: підійти до вікна та сфокусувати свій погляд на далекі предмети протягом сорока секунд. Так радять робити офтальмологи. Зарядка також може не пошкодити, а навіть покращити самопочуття.
- Наприкінці кожного робочого дня працівник повинен вимкнути свій персональний комп'ютер (перед цим перевіривши всі робочі файли, які були в роботі в цей день), залишити за собою порядок на робочому столі, відключити усі принтери та сканери, якщо він останній залишає робоче місце. При включеному кондиціонері його необхідно вимкнути. Світло так само має бути вимкнене.

Робоче місце і взаємне розташування всіх його елементів повинне відповідати антропометричним, фізичним і психофізіологічним характеристикам людини. Її організація має велику роль у подальшій роботі працівника. На робочому місці має бути нічого зайвого; предмети, які використовуються найчастіше, мають бути поряд, які менш – в інших частинах столу; організація робочого місця має бути коректно та

забезпечувати необхідну оглядовість; предмети, що потрібні для роботи лівою рукою, мають бути зліва, правою рукою – з правого боку тощо.

Для комфортної роботи стіл повинний задовольняти наступним умовам:

- Стіл має бути висотою з урахуванням можливостей адміністратора сидіти у зручній позі, вільно, при цьому, щоб була можливість опиратися на підлокітники;
- Нижня частина столу має бути такою, щоб адміністратор міг зручно сидіти, щоб його ноги доставали до підлоги, він міг вільно рухати ними;
- Стіл повинен бути таким, щоби зменшити або зовсім викоренити відблиски у полі зору адміністратора;
- Стіл має бути з великим арсеналом шухляд, для повноцінної роботи та збереження якихось речей (документації, канцелярії тощо);
- Висота робочої поверхні рекомендується в межах 680-760 мм. Вимоги до пози працівника під час роботи:
- Голова не повинна бути нахилена більше ніж на 20°;
- Плечі повинні бути розслаблені;
- Лікті під кутом 80...100°;
- Передпліччя і кисті рук – у горизонтальному положенні.

Працівник може неправильно сидіти за своїм робочим місцем, якщо його стіл не відповідає якостям, його розміри дуже великі або дуже маленькі, клавіатура та монітор знаходяться на неправильній відстані, а адміністратору приходиться вічно змінювати пози, щоб найти комфортну для роботи. Потрібно це враховувати при проектуванні та закупівлі обладнань для роботи.

Велике значення надається характеристикам робочого крісла. Так, рекомендована висота сидіння над рівнем підлоги знаходиться в межах 420-550 мм. Поверхня сидіння повинна бути м'якою, передній край закруглений, а кут нахилу спинки – регульований.

Вимоги під час роботи:

- Повинні бути передбачені спеціальні пристосування для регулювання висоти столу, клавіатури, екрану, а також підставки для рук;
- Істотне значення для продуктивної і якісної роботи за комп'ютером мають розміри знаків, щільність їхнього розміщення, контраст і співвідношення яркості, символів і фону екрану;
- Співвідношення яскравості фону екрану і символів – від 1:2 до 1:15;
- Монітор має бути встановлено на відстані 50-60 см від очей;
- Верхня частина дисплея повинна бути на рівні очей.

Створення сприятливих умов праці і правильне естетичне оформлення робочих місць на виробництві має велике значення як для полегшення праці, так і для підвищення його привабливості, що позитивно впливає на продуктивність праці.

Працівники центру кінного спорту «Верховина» згідно з існуючими нормами в повному обсязі забезпечені спецодягом та спецвзуттям, нормативно- правовими актами з охорони праці та інструкціями з охорони праці. Первинні та періодичні медичні та наркологічні огляди працівників проводяться своєчасно.

Питання з охорони праці у центрі кінного спорту «Верховина» є одним з найважливіших, з допомогою якого підприємство успішно функціонує.

ВИСНОВОК

У дипломній роботі було розроблено інформаційну систему для центру кінного спорту «Верховина». Було сформовано предметну область, визначені вимоги та задачі, концепції, що стосувалися автоматизації та прискорення роботи відділу роботи з клієнтами та роботу тренерів у школі. Також розроблено технічний опис проекту, який включав в себе інформаційну, математичні та програмні рішення. На основі концепту цієї ІС був реалізований рекламний сайт, програмне забезпечення якого було створено за допомогою мови програмування JavaScript, веб-фреймворків Express та VueJS із використанням засобів СУБД MySQL. Його можна буде розширити в майбутньому (до реалізовувати повноцінність сайту з входом у особистий аккаунт користувачем, тренером чи оператором; пов'язати дану інформаційну систему з іншими складовими школи, наприклад, розробити інтернет-крамницю, де користувачі зможуть додавати товари в кошик та дистанційно замовляти послуги). При цьому також було проведено аналіз охорони праці на підприємстві, вказані вимоги до робітників, шляхи уникнення небезпечних ситуацій.

Дана розробка має великий потенціал у майбутньому, для подальшого розвитку та розширення можливостей ІС.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Статистика видів спорту [Електронний ресурс]: Веб-журнал «Український тиждень». Режим звернення: https://tyzhden.ua/Society/212256?utm_source=yafavorites – дата звернення: 10.04.2025
2. Про кінний спорт [Електронний ресурс]: Вікіпедія. Вільна енциклопедія. – Режим звернення: https://uk.wikipedia.org/wiki/Кінний_спорт – дата звернення: 10.04.2025
3. ERD діаграма [Електронний ресурс]: Вікіпедія. Вільна енциклопедія. – Режим звернення: https://uk.wikipedia.org/wiki/Модель_«сутність_—_зв'язок» – дата звернення: 10.04.2025
4. Національні Правила змагань з виїздки, 2018. – 26 с
5. Олімпійській StartUp. Протокол суддівства з конкуру. – 1 с
6. MySQL [Електронний ресурс]: Вікіпедія. Вільна енциклопедія. – Режим звернення: <https://uk.wikipedia.org/wiki/MySQL#:~:text=MySQL%20—%20вільна%20система%20керування%20реляційними,створена%20як%20альтернатива%20комерційним%20системам.> – дата звернення: 10.04.2025
7. WebStorm [Електронний ресурс]: Вікіпедія. Вільна енциклопедія. – Режим звернення: <https://uk.wikipedia.org/wiki/WebStorm> – дата звернення: 10.04.2025
8. JavaScript [Електронний ресурс]: Вікіпедія. Вільна енциклопедія. – Режим звернення: <https://uk.wikipedia.org/wiki/JavaScript> – дата звернення: 10.04.2025
9. VueJS [Електронний ресурс]: Офіційна документація веб-фреймворку VueJS. Режим звернення: <https://v3.ru.vuejs.org/ru/guide/introduction.html> – дата звернення: 10.03.2025

10. Linux Mint [Електронний ресурс]: Вікіпедія. Вільна енциклопедія. – Режим звернення: https://uk.wikipedia.org/wiki/Linux_Mint – дата звернення: 10.04.2025
11. Chrome [Електронний ресурс]: Вікіпедія. Вільна енциклопедія. – Режим звернення: https://uk.wikipedia.org/wiki/Google_Chrome – дата звернення: 10.04.2022
12. Avast [Електронний ресурс]: Вікіпедія. Вільна енциклопедія. – Режим звернення: https://uk.wikipedia.org/wiki/Avast_Antivirus – дата звернення: 10.04.2022
13. Gmail [Електронний ресурс]: Вікіпедія. Вільна енциклопедія. – Режим звернення: <https://uk.wikipedia.org/wiki/Gmail> – дата звернення: 10.03.2022
14. Закон України Про охорону праці [Електронний ресурс]: – Режим звернення: <https://zakon.rada.gov.ua/laws/show/2694-12#Text> – дата звернення: 10.04.2022
15. Закон України Про охорону праці розділ IV [Електронний ресурс]: – Режим звернення: <https://zakon.rada.gov.ua/laws/show/2694-12#Text> – дата звернення: 10.04.2022
16. Конспект-матеріали з лекцій пройденого курсу «Охорона праці».
17. В. Ф. Ажищев, К. В. Кошкін, Г. С. Морозова Методичні вказівки до виконання дипломного проектування зі спеціальності "Комп'ютерні науки". М.: Миколаїв : НУК, 2014. – 56 с.

ДОДАТОК А – ТЕХНІЧНЕ ЗАВДАННЯ

Призначення розробки

Назва проекту: «ІС Центру кінного спорту «Верховина». Розробка ІС включає в собі розширення можливостей деяких сфер життя центру кінного спорту. Призначенням є: створення рекламного сайту для запрошення нових клієнтів, автоматизацію робочих процесів відділу роботи з клієнтами, прискорення роботи тренерів за допомогою додатків з тренування та розкладу. Метою створення ІС є прискорення роботи, покращення ефективності та якості, актуалізація багатьох процесів роботи працівників школи.

Вимоги до системи

ІС має бути мобільною та багато структурною, щоб реалізовувати різний функціонал. Система має:

- зберігати актуальну інформацію та динамічно відновлювати її у веб- додатку;
- складатися з трьох рівнів архітектурного рішення, а саме: клієнт (або декілька клієнтів, в залежності від потреб) – сервер – сервер БД;
- необхідно мати читабельний, зручний та зрозумілий інтерфейс.

Вимоги до програмного забезпечення

Вимоги до програмного забезпечення сервера:

- Операційна система: Linux;
- Середовище розробки: WebStorm;
- Мова програмування: JavaScript;
- Веб-фреймворк: Express.

Вимоги до програмного забезпечення клієнту:

- Операційна система: Linux;

- Середовище розробки: WebStorm;
- Мова програмування: JavaScript;
- Веб-фреймворк: Vue.

Вимоги до документації

Потрібно, щоб були наявні такі документи:

- Загальний опис системи;
- Посібник системного адміністратора програмного забезпечення;
- Програми та інструкції виконання;
- Інструкції введення бази даних та її використання;
- Текст програм;
- Посібник користувача.

Порядок контролю і прийому системи

Взаємодія замовника та розробника може бути необмеженою кількістю часу. Це робиться для того, щоб удосконалити систему, розширювати її функціонал.

При цьому на кожному новому етапі відбувається показовий виступ, на якому розробник показує напрацювання, функціональність системи, наводить різні махінації з нею. Замовник затверджує чи не затверджує:

- при не затвердженні розробник виправляє помилки (стверджується акт про помилки та шляхи їх виправлення), який виявив замовник, і знову показує результати;
- при затвердженні співробітництво продовжується до нових завдань, які поставить замовник розробнику.

Весь цей процес затверджується у вигляді акта-договору, які підписує директор компанії та розробник (або розробники з організацій). Отже, на законодавчому рівні буде затверджено співробітництво.

Співпраця може бути розірвана у разі не виконання цілей та послуг з

обох сторін за допомогою спеціальних юридичних процесів.

Вимоги до захисту інформації

Система має забезпечувати повний контроль за інформацією та її збереженням. Доступ до даних ІС має бути встановлено за правом користування. Тобто, певну інформацію користувач може отримувати в межах, встановлених за правом доступу. Як приклад, тренер має доступ до частини ІС, які включають в собі розклад, студенти; оператор має доступ до тренерів, розкладу, студентів, цін; системний адміністратор має доступ до всіх складових.

ДОДАТОК Б – КОД СТВОРЕННЯ ТАБЛИЦЬ MYSQL

Створення таблиці staff

```
CREATE TABLE `staff` (  
    `idstaff` int NOT NULL AUTO_INCREMENT,  
    `name` varchar(20) NOT NULL,  
    `salary` int NOT NULL,  
    PRIMARY KEY (`idstaff`)  
) ENGINE=InnoDB AUTO_INCREMENT=1 DEFAULT CHARSET=utf8mb4  
COLLATE=utf8mb4_0900_ai_ci
```

Створення таблиці workers

```
CREATE TABLE `workers` (  
    `idworkers` int NOT NULL AUTO_INCREMENT,  
    `birthday` date NOT NULL,  
    `firstnameworker` varchar(20) NOT NULL,  
    `secondnameworker` varchar(20) NOT NULL,  
    `threenameworker` varchar(20) NOT NULL,  
    `phone` bigint NOT NULL,  
    `sex` varchar(10) NOT NULL,  
    `staffid` int NOT NULL,  
    `img` varchar(20) NOT NULL,  
    PRIMARY KEY (`idworkers`),  
    KEY `stf_idx` (`staffid`),  
    CONSTRAINT `stf_wr` FOREIGN KEY (`staffid`) REFERENCES  
`staff` (`idstaff`) ON DELETE CASCADE ON UPDATE CASCADE  
) ENGINE=InnoDB AUTO_INCREMENT=1 DEFAULT CHARSET=utf8mb4  
COLLATE=utf8mb4_0900_ai_ci
```

Створення таблиці horse

```
CREATE TABLE `horse` (  
    `idhorse` int NOT NULL AUTO_INCREMENT,  
    `namehorse` varchar(20) NOT NULL,  
    `breed` varchar(25) NOT NULL,  
    `age` int NOT NULL,  
    `classh` varchar(15) NOT NULL,
```

```

        `sex` varchar(10) NOT NULL,
        `img_horse` varchar(20) NOT NULL,
        PRIMARY KEY (`idhorse`)
    ) ENGINE=InnoDB AUTO_INCREMENT=1 DEFAULT CHARSET=utf8mb4
    COLLATE=utf8mb4_0900_ai_ci

```

Створення таблиці users

```

CREATE TABLE `users` (
    `idusers` int NOT NULL AUTO_INCREMENT,
    `firstnameuser` varchar(20) NOT NULL,
    `secondnameuser` varchar(20) NOT NULL,
    `threenameuser` varchar(20) NOT NULL,
    `phone` bigint NOT NULL,
    `raiting` varchar(10) NOT NULL,
    `sex` varchar(10) NOT NULL,
    `birthday` date NOT NULL,
    `idhorse` int NOT NULL,
    `img` varchar(20) NOT NULL,
    PRIMARY KEY (`idusers`),
    KEY `usr_idx` (`idhorse`) /*!80000 INVISIBLE */,
    CONSTRAINT `usr_hs` FOREIGN KEY (`idhorse`) REFERENCES
`horse` (`idhorse`) ON DELETE CASCADE ON UPDATE CASCADE
    ) ENGINE=InnoDB AUTO_INCREMENT=1 DEFAULT CHARSET=utf8mb4
    COLLATE=utf8mb4_0900_ai_ci

```

Створення таблиці service

```

CREATE TABLE `service` (
    `usid` int NOT NULL,
    `workid` int NOT NULL,
    `nameofservice` varchar(15) NOT NULL,
    PRIMARY KEY (`usid`,`workid`),
    KEY `us_idx` (`usid`) /*!80000 INVISIBLE */,
    KEY `wr_idx` (`workid`) /*!80000 INVISIBLE */,
    CONSTRAINT `us_ser` FOREIGN KEY (`usid`) REFERENCES `users`
(`idusers`),
    CONSTRAINT `wr_ser` FOREIGN KEY (`workid`) REFERENCES
`workers` (`idworkers`)

```

```
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4  
COLLATE=utf8mb4_0900_ai_ci
```

Створення таблиці dayofweek

```
CREATE TABLE `dayofweek` (  
  `idday` int NOT NULL AUTO_INCREMENT,  
  `nameday` varchar(15) DEFAULT NULL,  
  PRIMARY KEY (`idday`)  
) ENGINE=InnoDB AUTO_INCREMENT=1 DEFAULT CHARSET=utf8mb4  
COLLATE=utf8mb4_0900_ai_ci
```

Створення таблиці timeschedule

```
CREATE TABLE `timeschedule` (  
  `idtime` int NOT NULL AUTO_INCREMENT,  
  `timestart` varchar(6) NOT NULL,  
  `timeover` varchar(6) NOT NULL,  
  PRIMARY KEY (`idtime`)  
) ENGINE=InnoDB AUTO_INCREMENT=1 DEFAULT CHARSET=utf8mb4  
COLLATE=utf8mb4_0900_ai_ci
```

Створення таблиці typesschedule

```
CREATE TABLE `typesschedule` (  
  `idtypessc` int NOT NULL AUTO_INCREMENT,  
  `name` varchar(24) DEFAULT NULL,  
  PRIMARY KEY (`idtypessc`)  
) ENGINE=InnoDB AUTO_INCREMENT=1 DEFAULT CHARSET=utf8mb4  
COLLATE=utf8mb4_0900_ai_ci
```

Створення таблиці schedule

```
CREATE TABLE `schedule` (  
  `usid` int NOT NULL,  
  `typesid` int NOT NULL,  
  `timeid` int NOT NULL,  
  `dayid` int NOT NULL,  
  PRIMARY KEY (`usid`,`typesid`,`dayid`,`timeid`),  
  KEY `us_idx` (`usid`) /*!80000 INVISIBLE */,  
  KEY `tp_idx` (`typesid`) /*!80000 INVISIBLE */,  
  KEY `tm_idx` (`timeid`) /*!80000 INVISIBLE */,  
  KEY `day_idx` (`dayid`) /*!80000 INVISIBLE */,
```

```

        CONSTRAINT `day_sc` FOREIGN KEY (`dayid`) REFERENCES
`dayofweek` (`idday`),
        CONSTRAINT `tm_sc` FOREIGN KEY (`timeid`) REFERENCES
`timeschedule` (`idtime`),
        CONSTRAINT `tp_sc` FOREIGN KEY (`typesid`) REFERENCES
`typesschedule` (`idtypessc`),
        CONSTRAINT `us_sc` FOREIGN KEY (`usid`) REFERENCES `users`
(`idusers`)
    ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci

```

Створення таблиці prices

```

CREATE TABLE `prices` (
    `idprice` int NOT NULL AUTO_INCREMENT,
    `nameofprice` varchar(25) DEFAULT NULL,
    `price` int DEFAULT NULL,
    PRIMARY KEY (`idprice`)
) ENGINE=InnoDB AUTO_INCREMENT=1 DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci

```

Створення таблиці rent

```

CREATE TABLE `rent` (
    `usid` int NOT NULL,
    `pricesid` int NOT NULL,
    PRIMARY KEY (`usid`,`pricesid`),
    KEY `us_idx` (`usid`) /*!80000 INVISIBLE */ ,
    KEY `pr_idx` (`pricesid`) /*!80000 INVISIBLE */ ,
    CONSTRAINT `pr_rent` FOREIGN KEY (`pricesid`) REFERENCES
`prices` (`idprice`),
    CONSTRAINT `us_rent` FOREIGN KEY (`usid`) REFERENCES
`users` (`idusers`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci

```

ДОДАТОК В – КОД СЕРВЕРНОЇ ЧАСТИНИ ПРОГРАМИ

```
package < config >
```

Листинг 1 – файл <database >

```
import mysql from "mysql2";  
//Підключення к БД.  
const db = mysql.createConnection({  
  host: 'localhost',  
  port: '3307',  
  user: 'root',  
  password: 'ilikejean21',  
  database: 'horsealtair'  
});  
db.connect(function (err) {  
  if(err) {  
    return console.error("Ошибка" + err.message);  
  } else {  
    console.log("Подключение успешно!")  
  }  
})  
export default db;
```

```
package < models >
```

Листинг 2 – файл < horseModel >

```
import db from "../config/database.js";  
//Запит коней та їх власників із БД.  
export const getHorseUsers = (result) => {  
  db.query("select us.img, h.namehorse, h.breed, h.age,  
h.classh, h.sex, h.img_horse, us.firstnameuser, us.secondnameuser,  
us.threenameuser, us.raiting from users as us inner join horse as  
h on h.idhorse=us.idhorse ORDER BY us.raiting desc;", (err,  
results) => {  
  if(err) {  
    console.log(err);  
    result(err, null);  
  }  
})  
}
```

```

        } else {
            result(null, results);
        }
    });
}

//Запит коней без власників із БД.
export const getHorseNullUsers = (result) => {
    db.query("SELECT h.namehorse, h.breed, h.age, h.classh, h.sex,
h.img_horse FROM horse as h WHERE NOT EXISTS (SELECT * FROM users
as us WHERE h.idhorse = us.idhorse) order by h.namehorse;", (err,
results) => {
        if(err) {
            console.log(err);
            result(err, null);
        } else {
            result(null, results);
        }
    });
}

```

package < models >

Листинг 3 - файл < pricesModel >

```

import db from "../config/database.js";
//Запит актуальних цін на послуги із БД.
export const getPrices = (result) => {
    db.query("SELECT pr.nameofprice, pr.price FROM prices as pr",
(err, results) => {
        if(err) {
            console.log(err);
            result(err, null);
        } else {
            result(null, results);
        }
    });
}

```

```

package < models >
Листинг 4 - файл < trainersModel >
import db from "../config/database.js";
//Запит робітників із БД.
export const getTrainers = (result) => {
    db.query("select w.idworkers, w.firstnameworker,
w.secondnameworker, w.threenameworker, w.img,
DATE_FORMAT(w.birthday, '%d-%m-%Y') as birthday, w.phone, s.name,
s.idstaff, w.staffid from workers as w inner join staff as s on
w.staffid=s.idstaff;", (err, results) => {
        if(err) {
            console.log(err);
            result(err, null);
        } else {
            result(null, results);
        }
    });
}
//Запит тренерів та їх розклад із БД.
export const getTrainerSchedule = (result) => {
    db.query("select w.firstnameworker, w.threenameworker,
us.firstnameuser, us.threenameuser, tis.timeover, tis.timestart,
ts.name, dw.nameday from workers as w inner join service as ser on
ser.workid=w.idworkers inner join users as us on us.idusers =
ser.usid inner join schedule as sch on us.idusers = sch.usid inner
join typesschedule as ts on ts.idtypessc=sch.typesid inner join
timeschedule as tis on tis.idtime = sch.timeid inner join
dayofweek as dw on dw.idday=sch.dayid;", (err, results) => {
        if(err) {
            console.log(err);
            result(err, null);
        } else {
            result(null, results);
        }
    });
}

```

```
package < controller >
```

Листинг 5 - файл < horseController >

```
import {getHorseUsers, getHorseNullUsers} from
"../models/horseModel.js";
//Форматування в JSON результату.
export const showHorseUsers = (req, res) => {
  getHorseUsers((err, results) => {
    if (err){
      res.send(err);
    }else{
      res.json(results);
    }
  });
}
//Форматування в JSON результату.
export const showHorseNullUsers = (req, res) => {
  getHorseNullUsers((err, results) => {
    if (err){
      res.send(err);
    }else{
      res.json(results);
    }
  });
}
```

```
package < controller >
```

Листинг 6 - файл < pricesController >

```
import {getPrices} from "../models/pricesModel.js";
//Форматування в JSON результату.
export const showPrices = (req, res) => {
  getPrices((err, results) => {
    if (err){
      res.send(err);
    }else{
      res.json(results);
    }
  });
}
```



```
    }  
  });  
}
```

```
package < controller >
```

Листинг 7 - файл < trainersController >

```
import {getTrainers, getTrainerSchedule} from  
"../models/trainersModel.js";
```

```
//Форматування в JSON результату.
```

```
export const showTrainers = (req, res) => {  
  getTrainers((err, results) => {  
    if (err){  
      res.send(err);  
    }else{  
      res.json(results);  
    }  
  });  
}
```

```
}
```

```
//Форматування в JSON результату.
```

```
export const showTrainerName = (req, res) => {  
  getTrainerSchedule((err, results) => {  
    if (err){  
      res.send(err);  
    }else{  
      res.json(results);  
    }  
  });  
}
```

```
package < routes >
```

Листинг 8 - файл < routes >

```
import express from "express";  
import {showTrainerName, showTrainers} from  
"../controllers/trainersController.js";  
import {showHorseUsers, showHorseNullUsers} from
```

```
"../controllers/horseController.js";
import {showPrices} from "../controllers/pricesController.js";
//Створення маршрутів даних.
const router = express.Router();
router.get('/trainer', showTrainers);
router.get('/horseusers', showHorseUsers);
router.get('/horsenullusers', showHorseNullUsers);
router.get('/trainersname', showTrainerName);
router.get('/pricesname', showPrices);
export default router;
```

Листинг 9 - файл < index >

```
import express from "express";
import cors from "cors";
import router from "../src/routes/routes.js";
//стартовий файл
const app = express();
app.use(express.json());
app.use(cors());
app.use(router);
app.listen(5000, () => console.log('Server running at
http://localhost:5000'));
```

ДОДАТОК Г – КОД КЛІЄНТСЬКОЇ ЧАСТИНИ ПРОГРАМИ

Листинг 1 – файл < main >

```
import { createApp } from 'vue'
import App from './App.vue'
import { createRouter, createWebHistory } from "vue-router";
import MainPage from "@view/MainPage";
import AboutStage from "@view/AboutStage";
import HorseStage from "@view/HorseStage";
import TrainersStage from "@view/TrainersStage";
import ScheduleStage from "@view/ScheduleStage";
import PeculStage from "@view/PeculStage";
import FeedbackStage from "@view/FeedbackStage";
//маршрути сторінки та початок роботи
const routes = [
  {
    path: '/',
    component: MainPage,
    meta: {requiresAuth: true}
  },
  {
    path: '/#about',
    component: AboutStage,
  },
  {
    path: '/#horse',
    component: HorseStage,
  },
  {
    path: '/#trainer',
    component: TrainersStage,
  },
  {
    path: '/#schedule',
    component: ScheduleStage,
```

```

    },
    {
      path: '/#pecul',
      component: PeculStage,
    },
    {
      path: '/#feedback',
      component: FeedbackStage,
    },
  ];
const router = createRouter({
  history: createWebHistory(),
  routes
});
createApp(App).use(router).mount('#app')

```

Скрипти у файлах Vue

```
package < view >
```

Листинг 2 - файл < trainersStage >

```

export default {
  name: "TrainersStage",
  components: {TrainerProfi, TrainerNew, TrainerOne,
ModalTrainers},
  data() {
    return {
      allTrainers: [],
      showModal: false,
      chosenArr: null,
      chosenIndex: null
    }
  },

  created(){
    this.getTrainer();
  },

```

```

methods: {
    //перетворює ім'я файлу із БД у путь до цього файлу для
    відображення
    getImgUrl(img) {
        return require('../assets/image/trainer_stage/photo/' +
img);
    },
    //зв'язок з сервером, отримання даних щодо тренерів
    async getTrainer() {
        try {
            const response = await
axios.get("http://localhost:5000/trainer");
            this.allTrainers = response.data;
        } catch (err) {
            console.log(err);
        }
    },
    //відкриття модального вікна та передача даних щодо вибраного
тренера
    openModalView(arr, index) {
        if (arr !== null) {
            this.chosenArr = arr;
            this.chosenIndex = index;
        }
        this.showModal = true;
    },
    //закриття модального вікна
    closeModalView() {
        this.showModal = false;
    }
},

computed: {
    //фільтрація отриманих даних щодо «Тренера з конкуру»
    findTrainersKonk() {

```

```

        return this.allTrainers.filter((allTrainers) => {
            return allTrainers.name==='Тренер з конкуру';
        })
    },
    //фільтрація отриманих даних щодо «Тренера профі»
    findTrainersProfi() {
        return this.allTrainers.filter((allTrainers) => {
            return allTrainers.name==='Тренер профі';
        })
    },
    //фільтрація отриманих даних щодо «Тренера з виїздки»
    findTrainersVui() {
        return this.allTrainers.filter((allTrainers) => {
            return allTrainers.name==='Тренер з виїздки';
        })
    },
    //фільтрація отриманих даних щодо «Тренера новеньких»
    findTrainersNov() {
        return this.allTrainers.filter((allTrainers) => {
            return allTrainers.name==='Тренер новеньких';
        })
    }
}
}

```

package < view >

Листинг 3 - файл < scheduleStage >

```

name: "ScheduleStage",
components: {PartSchedule},
data() {
    return {
        indexActiveTrainer: null,
        workerDate: [],
        chosenArr: null,
        chosenIndex: null,
    }
},
computed: {

```

```

//сортування отриманих від БД даних щодо першого імені для
подальшої взаємодії
sortedData() {
  return this.workerDate.reduce(
    (massive, object) => {
      const nameworker = object.firstnameworker;
      massive[nameworker] ??= [];
      massive[nameworker].push(object);
      return massive;
    },
    {}
  )
},
created() {
  //відображення даних на сторінці при відкритті сторінки
  this.getDateTrainer();
},
methods: {
  //зв'язок з сервером, отримання даних щодо розкладу
  async getDateTrainer() {
    try {
      const response = await
axios.get("http://localhost:5000/trainersname");
      this.workerDate = response.data;
    } catch (err) {
      console.log(err);
    }
  },
  activeTrainer(arr, index) {
    if (arr !== null) {
      this.chosenArr = arr;
      this.chosenIndex = index;
    }
  },
}
package < view >

Листинг 4 - файл < horseStage >

name: "HorseStage",
components: {BlockHorseNotNull, BlockHorsenNull},
data() {
  return {
    horsenulluser: [],
    horseuser: [],
    indexActiveUser: 0,
    indexActiveNull: 0,
    showModal: false,
    chosenArr: null,
  }
},

created() {

```

```

//відображення даних на сторінці при відкритті сторінки
this.getHorseUser();
this.getHorseNullUser();
},

computed: {
//повернення одного об'єкту масиву
  activeHorseUser() {
    return this.horseuser[this.indexActiveUser];
  },
  activeHorseNull() {
    return this.horsenulluser[this.indexActiveNull];
  }
},

methods: {
//зв'язок з сервером, отримання даних щодо коней без власника
  async getHorseNullUser() {
    try {
      const response = await
axios.get("http://localhost:5000/horsenullusers");
      this.horsenulluser = response.data;
    } catch (err) {
      console.log(err);
    }
  },
//зв'язок з сервером, отримання даних щодо коней з власником
  async getHorseUser() {
    try {
      const response = await
axios.get("http://localhost:5000/horseusers");
      this.horseuser = response.data;
    } catch (err) {
      console.log(err);
    }
  },
}
}

```