МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

МАТЕМАТИЧНИЙ ФАКУЛЬТЕТ
Кафедра програмної інженерії

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

на тему: **«ПРОЄКТУВАННЯ ТА РОЗРОБКА ВЕБСАЙТУ ФУТБОЛЬНОГО КЛУБУ»**

Виконав: студент 3 курсу, групи 6.1211-пі-с
спеціальності 121 інженерія програмного забезпечення
(шифр і назва спеціальності)
освітньої програми програмна інженерія (зі скороченим терміном навчання)
(назва освітньої програми)
Я.В. Шмата
(ініціали та прізвище)
Керівник доцент кафедри програмної інженерії, доцент, к.т.н. Лимаренко Ю.О.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)
Рецензент доцент кафедри електроніки, інформаційних систем та програмного забезпечення ІННІ ЗНУ, доцент, к.т.н. Заяц В.І.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Запоріжжя – 2024

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет математичний

Кафедра програмної інженерії

Рівень вищої освіти бакалавр

Спеціальність 121 інженерія програмного забезпечення
(шифр і назва)

Освітня програма програмна інженерія (зі скороченим терміном навчання)

ЗАТВЕРДЖУЮ
Завідувач кафедри програмної інженерії, к.ф.-м.н., доцент

_____ Лісняк А.О.
(підпис)

“ ____ ” _____ 2023 р.

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТОВІ

Шматі Ярославу Віталійовичу
(прізвище, ім’я та по-батькові)

1. Тема роботи Проєктування та розробка вебсайту футбольного клубу

керівник роботи Лимаренко Юлія Олексіївна , к.т.н., доцент
(прізвище, ім’я та по-батькові, науковий ступінь, вчене звання)

затверджені наказом ЗНУ від « 21 » грудня 2023 року № 2180-с

2. Строк подання студентом роботи 03.06.2024 р.

3. Вихідні дані до роботи 1. Постановка задачі.
2. Перелік літератури.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Дослідження та аналіз предметної області.
2. Проєктування вебсайту.
3. Розробка вебсайту.

5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)
презентація до захисту

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання 25.12.2023 р.

К А Л Е Н Д А Р Н И Й П Л А Н

№	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	Розробка плану роботи.	09.01.2024	
2.	Збір вихідних даних.	14.02.2024	
3.	Обробка методичних та теоретичних	28.02.2024	
	джерел.		
4.	Розробка першого та другого розділу.	11.04.2024	
5.	Розробка третього розділу.	20.05.2024	
6.	Оформлення та нормоконтроль	27.05.2024	
	кваліфікаційної роботи бакалавра.		
7.	Захист кваліфікаційної роботи.	21.06.2024	

Студент _____ Я.В. Шмата
(підпис) (ініціали та прізвище)

Керівник роботи _____ Ю.О. Лимаренко
(підпис) (ініціали та прізвище)

Нормоконтроль пройдено

Нормоконтролер _____ А.В. Столярова
(підпис) (ініціали та прізвище)

РЕФЕРАТ

Кваліфікаційна робота бакалавра «Проєктування та розробка вебсайту футбольного клубу»: 62 с., 44 рис., 12 джерел.

АДАПТИВНІСТЬ ВЕБСАЙТУ, ВЕБДИЗАЙН, ВЕБПРОГРАМУВАННЯ, ВЕБСАЙТІВ ХОСТИНГ, ВЕБСТОРІНОК СТИЛІЗАЦІЯ, ВІЗУАЛЬНЕ ОФОРМЛЕННЯ, ГІПЕРТЕКСТУ МОВА РОЗМІТКИ, ІНФОРМАЦІЙНИЙ САЙТ, КОНТЕНТОМ УПРАВЛІННЯ, КРОСБРАУЗЕРНА СУМІСНІСТЬ, ФРОНТ-ЕНД РОЗРОБКА.

Об’єкт дослідження – процес розробки вебсайту футбольного клубу

Предмет дослідження – розробка вебсайту футбольного клубу.

Мета дослідження – розробка фан вебсайту футбольного клубу.

У кваліфікаційній роботі було наведено аналіз предметної області, огляд існуючих систем з предметної області та описано засоби для створення вебсайтів. На основі даних теоретичних відомостей розроблено проєкт вебсайту для відслідковування новин улюбленого клубу та можливості вивчити історію походження клубу. Результатом виконання кваліфікаційної роботи є вебсайт, створений за допомогою html, css та javascript.

SUMMARY

Bachelor’s qualifying paper “Design and Development of a Football Club Website”: 62 p., 44 figures, 12 references.

ADAPTABILITY WEBSITE, BROWSER COMPATIBILITY CROSS, CONTENT MANAGEMENT, DESIGN VISUAL, DEVELOPMENT FRONT-END, HOSTING WEBSITE, HYPERTEXT MARKUP LANGUAGE, INFORMATION WEBSITE, PAGE STYLING WEB, PROGRAMMING WEB, WEB DESIGN.

Object of research – the process of developing a football club website

Subject of research – development of a football club website.

Purpose of the study – development of a football club fan website.

The qualification work provided an analysis of the subject area, an overview of existing systems in the subject area and described the tools for creating websites. Based on this theoretical information, a website project was developed to track the news of your favourite club and the opportunity to purchase branded goods. The result of the qualification work is a website created using the html, css and javascript.

ЗМІСТ

ВСТУП

За останнє десятиріччя значно зросла кількість користувачів Інтернет не лише у світі, але й в Україні. Разом із її зростанням розширилася сфера використання мережі у соціально-економічному житті суспільства, зокрема: побутове використання, у навчання та економічні діяльності. Кількість Інтернет-користувачів в Україні невпинно збільшується, що призводить до зростання інфраструктури та її обсягів.

На фоні стрімкого розвитку Інтернету, все більше підприємств і організацій усвідомлюють необхідність створення вебсайтів для представлення своєї діяльності та залучення нових споживачів. Не є виключенням і спортивні клуби, які активно використовують Інтернет для взаємодії зі своїми фанатами, поширення інформації про клуб. Ця дипломна робота присвячена розробці вебсайту футбольного клубу.

На сайті буде представлена актуальна інформація про результати команди, її склад, найближчі матчі та її історії. Метою даної роботи є створення функціонального та зручного для користувачів вебсайту, який стане важливим інструментом для підтримки взаємодії між клубом та його прихильниками, а також сприятиме розвитку кількості фанатів.

Для досягнення цієї мети будуть розглянуті та використані сучасні технології веброзробки, а також враховані кращі практики дизайну та користувацького досвіду. Розробка сайту включатиме кілька етапів: аналіз вимог, проєктування, програмування, тестування та впровадження.

Таким чином, дана робота має не лише теоретичну цінність, а й практичне застосування, оскільки результати дослідження можуть бути використані для подальшого розвитку та вдосконалення вебсайтів інших спортивних організацій.

Об’єкт дослідження – процес розробки фан вебсайту футбольного клубу

Предмет дослідження – розробка фан вебсайту футбольного клубу

засобами HTML, CSS, JavaScript.

Мета дослідження – розробка фан вебсайту футбольного клубу.

Для реалізації поставленої мети було сформульовано наступні завдання:

– проаналізувати вебсайти футбольних клубів;
– дослідити та обрати засоби розробки;
– розробити проект вебсайту футбольного клубу;
– розробити вебсайт відповідно до проекту.

Розроблена система задовольняє всім вимогам, що були зазначені на етапі постановки завдання.

Кваліфікаційна робота бакалавра складається зі вступу та трьох розділів.

Перший розділ містить основні відомості про вебсайти футбольних клубів, а саме розглянуто сайти-аналоги та сформульовано основні вимоги до розроблюваного сайту, досліджено інструменти розробки.

Другий розділ містить технічне завдання, опис структури подібних вебсайтів, діаграму варіантів використання, макети для створення вебсайту.

Третій розділ містить опис розробленого вебсайту, особливості реалізації, опис веб-сторінок розробленого вебсайту.

1 ДОСЛІДЖЕННЯ ТА АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Опис об’єкту автоматизації

Основна мета проектування – розробка вебсайту футбольного клубу для фанатів. Сайт, що розробляється, повинен являти собою вебсайт для слідкування за улюбленим футбольним клубом, його результатами, наступними матчами, складом та вивчанням його історії.

Створюваний сайт повинен являти собою відкриту, розширювану, масштабовану і модифіковану систему.

1.2 Обгрунтування необхідності автоматизації

Користувачів Інтернету з кожним днем стає все більше. А це означає, що поширення вебсайту в мережі відбувається дуже швидко. Тепер за допомоги інтернету можна робити все що завгодно. Тому необхідно розміщувати вебсайти для фанатів футбольних клубів, щоб всі мали змогу зручно мати актуальну інформацію в одному місці.

1.3 Дослідження аналогів сайту

Для створення сайту інтернет-магазину футбольного товару дослідимо сучасні тенденції в цій галузі, аналізуючи сайти аналогічного спрямування. Зокрема, можна виділити декілька сайтів:

– http://www.fcab.com.ua/ [11];
– https://dynamo.kiev.ua/uk/ [12];
– https://fcdynamo.com/.

Представлення головних сторінок та вкладки “Новини” на рисунках 1.1 – 1.4.

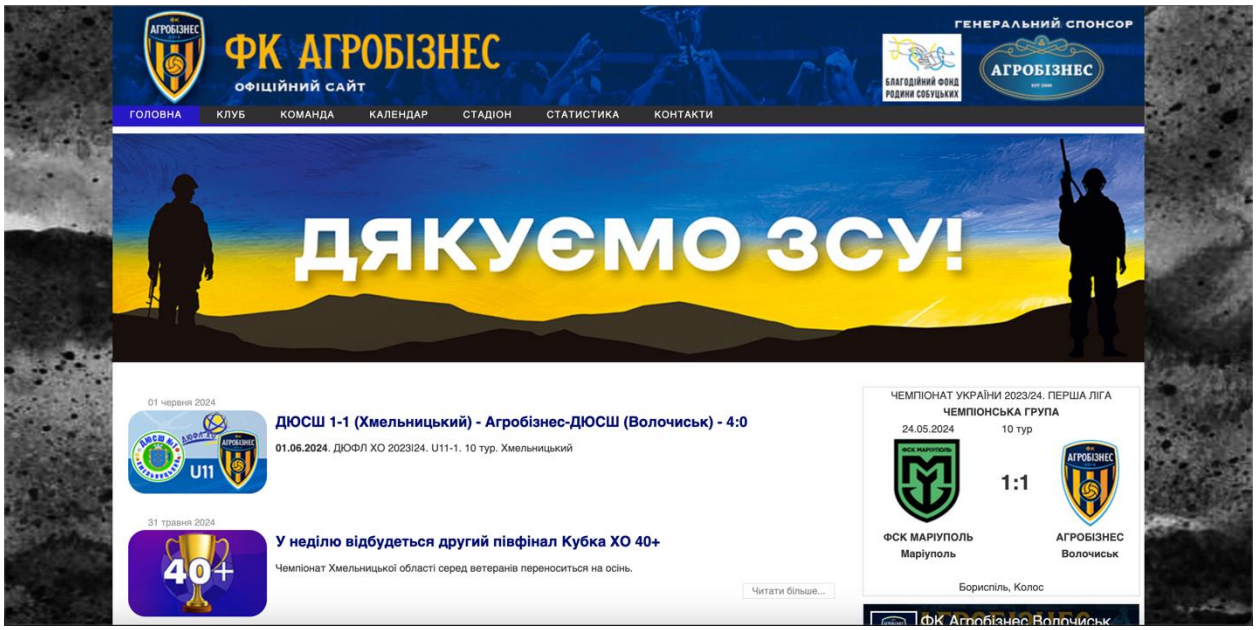

Рисунок 1.1 – Вигляд аналогічного вебсайту ФК Агробізнес

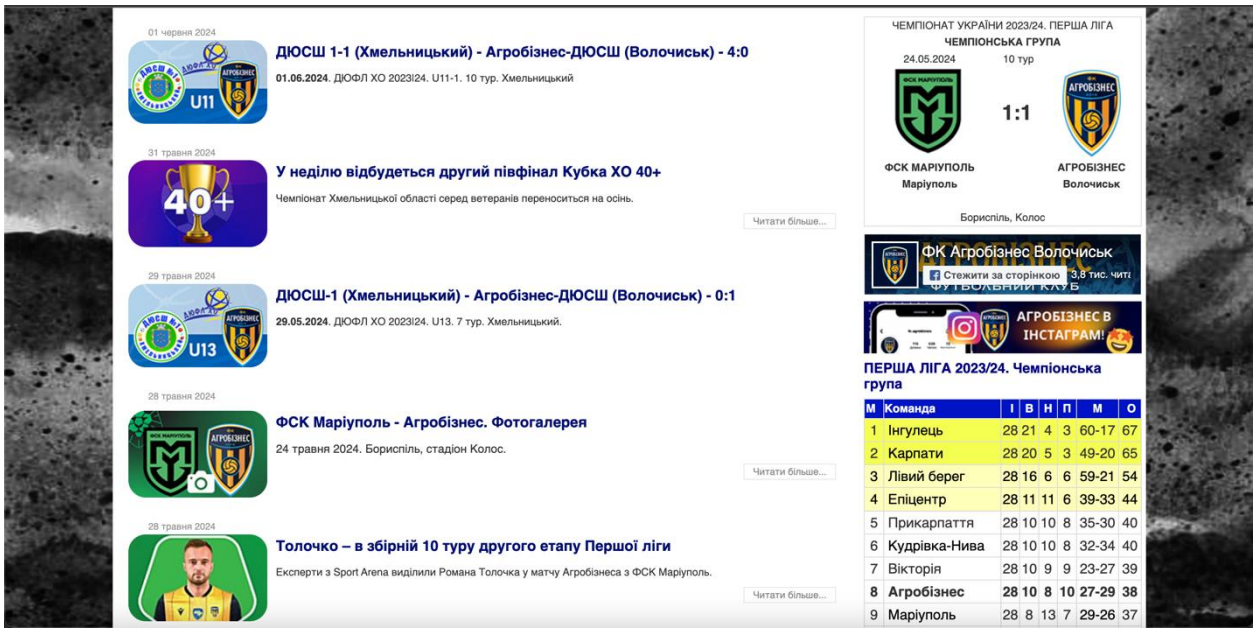

Рисунок 1.2 – Вигляд вкладки “Новини” на сайті ФК Агробізнес

На цьому сайті ми можемо помітити такі недоліки, як відсутність вкладки з новинами та відсутність описаної історії клубу, що може повпливати на зацікавленість споживача, щодо слідкування та вболівання за цей клуб.

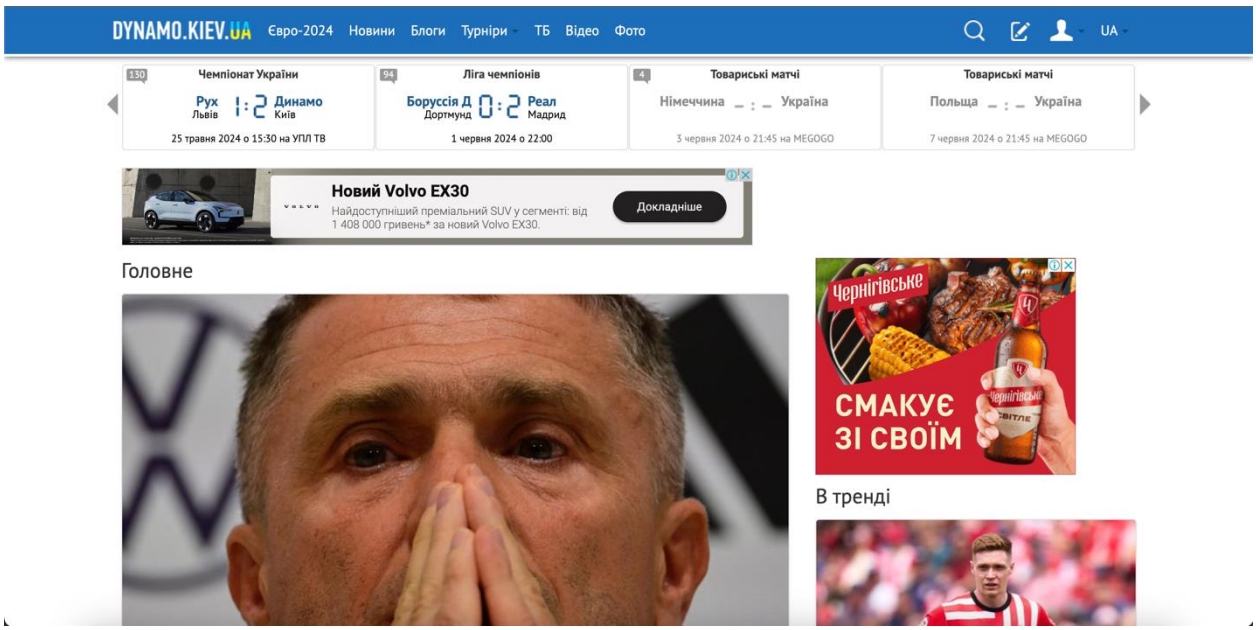

Рисунок 1.3 – Вигляд аналогічного вебсайту “Dynamo.kiev.ua”

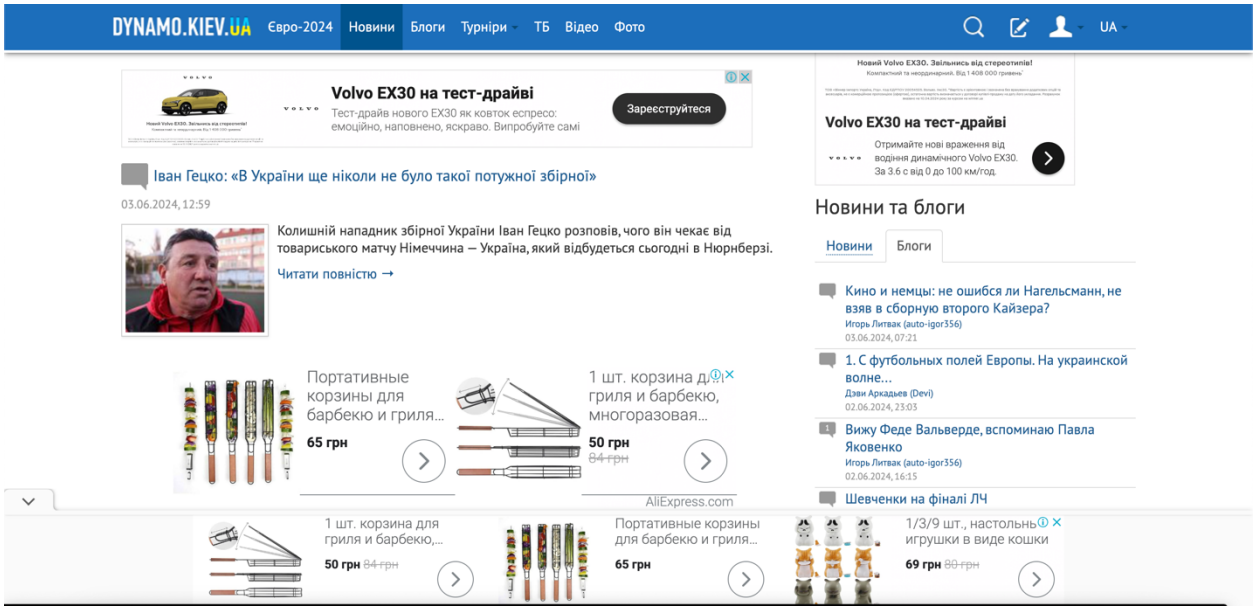

Рисунок 1.4 – Вигляд вкладки “Новини” на сайті “Dynamo.kiev.ua”

На цьому аналозі, ми можемо помітити, що візуальне оформлення сайту не дуже приваблює нас, як споживача, також присутня дуже велика кількість реклами, яка заважає переглядати будь-яку інформацію на сайті.

1.3.1 Переваги сайтів

Дослідивши дизайн та функціонал наведених сайтів, у них можна виділити наступні переваги:

– легкість в роботі;
– зрозумілість;
– підтримка декількох мов;
– зручне використання;
– перемикання світлого та темного режиму.

1.3.2 Недоліки сайтів

Розглянуті сайти мають також певні недоліки:

– відсутня можливість користувачам залишати коментарі;
– недостатня мобільна оптимізація;
– велика кількість реклами;
– відсутність вкладки з новинами;
– візуальна не привабливість.

1.4 Постановка завдання

Основними функціями і завданнями створюваного програмного продукту є наступні:

– організація зрозумілого і простого в управлінні користувацького інтерфейсу;
– забезпечення більш зручного доступу до даних;
– актуальна та якісна інформація.

1.5 Формування вимог до сайту

Таким чином, з урахуванням аналізу аналогів, можна сформулювати наступні вимоги до сайту:

- швидке формування, просте, зручне користування;
- мобільна версія;
- легкість пошуку інформації;
- відсутність зайвої інформації;
- відсутність реклами;
- легкість використання;
- зрозумілість сайту;
- візуальна привабливість;
- актуальна інформація;
- присутність всіх потрібних вкладок.

1.6 Засоби розробки сайту

Для створення інтернет-магазину були використані HTML, CSS та JavaScript. HTML забезпечує структуру сторінок, CSS дозволяє стилізувати їх, надаючи привабливий вигляд, а JavaScript забезпечує інтерактивність і функціональність сайту. HTML (HyperText Markup Language) є основною мовою розмітки для створення вебсторінок. Вона визначає структуру контенту за допомогою таких елементів, як заголовки, абзаци, списки, таблиці, зображення та посилання. Завдяки HTML можна легко організувати інформацію на сторінці, зробити її зрозумілою та доступною для користувачів.

CSS (Cascading Style Sheets) використовується для оформлення HTML-елементів. З його допомогою можна налаштовувати кольори, шрифти, відступи, розташування елементів на сторінці та багато іншого. CSS дозволяє створити привабливий і послідовний дизайн для всього сайту, що покращує користувацький досвід та робить сайт більш професійним.

JavaScript – це мова програмування, яка додає інтерактивність до вебсторінок. З його допомогою можна створювати динамічні елементи, такі як випадаючі меню, слайдери зображень, форми з перевіркою введення даних та багато іншого. JavaScript дозволяє реагувати на дії користувачів у реальному часі, що робить сайт більш живим та зручним у використанні.

Використання HTML, CSS та JavaScript має безліч переваг. По-перше, ці технології забезпечують гнучкість і адаптивність, дозволяючи створювати адаптивні дизайни, що виглядають добре на будь-яких пристроях – від мобільних телефонів до настільних комп'ютерів [3]. По-друге, JavaScript надає можливість створювати інтерактивні функції, що покращують взаємодію користувачів із сайтом. По-третє, завдяки оптимізації CSS та JavaScript, вебсторінки можуть швидко завантажуватись, що є важливим фактором для зручності користувачів та SEO. І, нарешті, ефективне поєднання HTML, CSS та JavaScript дозволяє створювати зручні та інтуїтивно зрозумілі інтерфейси, які полегшують навігацію по сайту та здійснення покупок.

Завдяки використанню цих технологій вебсайт отримав професійний та сучасний вигляд, високу функціональність та можливість ефективно збільшувати популярність та кількість фанатів клубу.

1.6.1 JavaScript

JavaScript – це високорівнева, інтерпретована мова програмування, яка використовується для створення динамічного контенту на вебсайтах. Вона є однією з найпопулярніших мов програмування серед розробників вебдодатків.

JavaScript широко використовується для реалізації різноманітних функцій, таких як валідація форм, анімація, динамічне оновлення сторінок без перезавантаження, обробка подій користувача та багато іншого. Вона може бути використана для розробки як клієнтської, так і серверної частини вебдодатків [8].

JavaScript є мовою з відкритим вихідним кодом, що сприяє постійному розвитку та підтримці спільнотою розробників. Існують безліч фреймворків і бібліотек, таких як React, Angular, Vue.js тощо, які роблять розробку вебдодатків на JavaScript більш ефективною та зручною.

Завдяки своїй широкій підтримці в браузерах, JavaScript став необхідним інструментом для будь-якого веброзробника. Вона є важливою складовою сучасного веброзвитку і продовжує зростати в популярності завдяки своїй потужності та гнучкості.

1.6.2 GNU

GNU General Public License (Загальна публічна ліцензія GNU або Загальна громадська ліцензія GNU) – одна з найпопулярніших ліцензій на вільне програмне забезпечення, створена Річардом Столменом для проекту GNU.

Мета GNU GPL – надання користувачеві прав на копіювання, зміни й розповсюдження програми та зобов’язань, згідно з якими користувачі всіх похідних від неї програм теж отримають ці права. Принцип «спадковості» таких прав називають «копілефт», такий термін запропонував Річард Столмен. На відміну від GPL, ліцензії на власницьке програмне забезпечення дуже рідко надають користувачеві такі права й, переважно, намагаються, навпаки, обмежити їх, наприклад, встановивши заборону на відновлення початкового коду.

GPL – приклад сильної копілефт-ліцензії, яка вимагає, щоб усі похідні роботи були доступні на тих же умовах, що й оригінал. GPL надає одержувачам комп’ютерної програми права відповідно до визначення вільного програмного забезпечення й використовує копілефт, щоб гарантувати, що ці права будуть збережені навіть тоді, коли робота буде значно змінена чи до неї будуть додані будь-які частини. Це відрізняє її від

дóзвільних ліцензій на безплатне програмне забезпечення, прикладом яких є ліцензія BSD або ліцензія Apache.

1.6.3 CSS

CSS (абревіатура від Cascading Style Sheets, що в перекладі означає каскадні таблиці стилів) є спеціальною мовою стилів, яка використовується для опису вигляду документів, написаних мовами розмітки даних, такими як HTML, XHTML та XML. Вона дозволяє визначати, які елементи вебсторінки будуть відображені і як вони будуть відображені [9].

Однією з головних переваг CSS є здатність розділити зміст сторінки від її оформлення. Це дозволяє покращити сприйняття та доступність змісту, забезпечити більшу гнучкість та контроль за відображенням змісту в різних умовах.

CSS дозволяє розмітці сторінки бути більш структурованою та простішою, оскільки вона дозволяє уникнути повторення коду. Це спрощує процес розробки та підтримки вебсайтів, оскільки зміни в стилях можна внести централізовано і вони автоматично застосуються до всіх відповідних елементів.

Крім того, CSS забезпечує можливість створення адаптивного та респонсивного дизайну, що дозволяє сторінці належним чином адаптуватися до різних розмірів екранів та пристроїв. Це робить вебсайти більш динамічними та зручними для користувачів.

Загалом, CSS є важливим інструментом для веброзробників, який дозволяє створювати стильні, привабливі та функціональні вебсторінки з великою ефективністю та гнучкістю.

Що дає використання CSS:

– відображати один і той же документ в різних стилях;

– декілька дизайнів сторінки для різних пристроїв (наприклад, на

екрані дизайн буде розрахований на велику ширину, під час друку меню не виводитиметься, а на смартфоні меню буде внизу, під вмістом);

- зменшення часу завантаження сторінок сайту за рахунок перенесення правил відображення в окремий CSS-файл (в цьому випадку браузер завантажує тільки структуру документа і дані, що зберігаються на сторінці, а стильові правила цих даних завантажуються браузером тільки один раз і кешуються);
- простота подальшої зміни дизайну (не потрібно правити кожну сторінку, а лише змінити CSS-файл);
- додаткові можливості оформлення (наприклад, за допомогою CSS-розмітки можна зробити так, щоб меню було завжди видно при скролінгу сторінки, або прибрати підкреслення у посилань);
- дозволяє створювати складну і пропрацьовану техніку дизайну.

1.6.4 HTML

HTML (HyperText Markup Language – мова розмітки гіпертексту) є основною мовою для створення вебсторінок у всесвітній мережі Інтернет. Ця мова використовується для розмітки контенту на вебсторінках і визначає структуру та вигляд цих сторінок [1].

Браузери отримують HTML-документи з вебсерверів або з локальної пам’яті користувача і інтерпретують їх для відображення вебсторінок. HTML дозволяє вбудовувати різноманітні елементи, такі як текст, зображення, відео, аудіо та інші медіа-елементи, а також інтерактивні форми і елементи.

Елементи HTML є будівельними блоками вебсторінок і складаються з тегів, які визначають тип контенту та його відображення. Наприклад, тег `<p>` використовується для визначення абзаців тексту, тег `<img>` – для вставки зображень, тег `<a>` – для створення посилань, та інші.

HTML надає можливість створювати структуровані документи, які відображають ієрархічну семантику тексту, таку як заголовки, абзаци, списки, цитати тощо. Це допомагає забезпечити зрозумілість та доступність контенту для користувачів та пошукових систем.

HTML є основою веброзробки і є важливим інструментом для будь-якого веброзробника. Вона надає можливість створювати візуально привабливі та функціональні вебсторінки для широкого кола користувачів Інтернету.

HTML впроваджує засоби для:

– створення структурованого документа шляхом позначення структурного складу тексту: заголовки, абзаци, списки, таблиці, цитати та інше;
– отримання інформації із Всесвітньої мережі через гіперпосилання;
– створення інтерактивних форм;
– включення зображень, звуку, відео, та інших об’єктів до тексту.

1.6.5 Порівняння HTML, CSS, JS з іншими мовами програмування

У порівнянні з іншими мовами програмування, такими як Python, Ruby або PHP, HTML, CSS та JavaScript мають низку унікальних переваг. Наприклад, Python та Ruby є чудовими для розробки бекенд-частини вебдодатків, але вони не можуть забезпечити такий рівень інтерактивності та візуальної привабливості, як JavaScript у поєднанні з HTML та CSS. HTML забезпечує структуру вебсторінок, визначаючи їхній зміст за допомогою тегів, таких як заголовки, абзаци, списки, таблиці, зображення та посилання.

Завдяки своїй простоті та зрозумілості HTML є стандартом в індустрії веброзробки, забезпечуючи сумісність з усіма веббраузерами та платформами. CSS використовується для стилізації HTML-елементів, дозволяючи налаштовувати кольори, шрифти, відступи, розміри та розташування

елементів на сторінці. CSS також дозволяє створювати адаптивні дизайни, які автоматично підлаштовуються під різні розміри екранів та пристроїв, що є особливо важливим у сучасному світі, де користувачі часто переглядають вебсайти з мобільних пристроїв.

JavaScript – це мова програмування, яка додає інтерактивність до вебсторінок. Однією з найбільших переваг JavaScript є його здатність виконуватися безпосередньо у веббраузері, що забезпечує швидку та інтерактивну взаємодію з користувачами. JavaScript дозволяє створювати динамічні елементи, такі як випадаючі меню, слайдери зображень, форми з перевіркою введення даних та багато іншого. Крім того, JavaScript підтримує роботу з великим числом бібліотек та фреймворків, таких як React, Angular та Vue.js, які значно спрощують розробку складних вебдодатків.

Використання HTML, CSS та JavaScript разом забезпечує низку переваг, які роблять їх незамінними для веброзробки. Ця комбінація дозволяє створювати повноцінні вебдодатки з привабливим дизайном та високою функціональністю. Всі три мови є добре документованими та мають велику спільноту розробників, що забезпечує доступ до численних ресурсів, навчальних матеріалів та підтримки. HTML, CSS та JavaScript є стандартами в індустрії веброзробки, що забезпечує їхню сумісність з усіма сучасними веббраузерами та платформами. Це означає, що вебсайти та додатки, створені за допомогою цих технологій, будуть коректно відображатися та працювати на будь-якому пристрої.

У порівнянні з іншими мовами програмування, такими як Python, Ruby або PHP, HTML, CSS та JavaScript мають низку унікальних переваг. Python та Ruby є чудовими для розробки бекенд-частини вебдодатків, але вони не можуть забезпечити такий рівень інтерактивності та візуальної привабливості, як JavaScript у поєднанні з HTML та CSS. PHP часто використовується для серверної частини вебдодатків, але він не має можливостей для роботи на стороні клієнта, як JavaScript [10].

Крім того, комбінація HTML, CSS та JavaScript дозволяє створювати

односторінкові додатки (SPA), які забезпечують більш швидкий та плавний користувацький досвід у порівнянні з традиційними багатосторінковими додатками [4].

У підсумку, HTML, CSS та JavaScript є потужними та незамінними інструментами для веброзробки, які забезпечують високу гнучкість, сумісність, інтерактивність та привабливий дизайн вебсайтів та додатків.

Використання цих технологій дозволяє створювати сучасні, адаптивні та ефективні вебрішення, що відповідають вимогам сучасного інтернету та потребам користувачів.

2 ПРОЄКТУВАННЯ ВЕБСАЙТУ

2.1 Створення технічного завдання

Технічне завдання (ТЗ) – це вихідний документ для проєктування споруди чи промислового комплексу, конструювання технічного пристрою, розробки автоматизованої системи, створення програмного продукту або проведення науково-дослідних робіт (НДР) [2].

ТЗ є ключовим документом, який регулює всі етапи реалізації проєкту, від початкового планування до введення в експлуатацію та подальшого обслуговування [2].

ТЗ містить в собі всі необхідні вимоги та параметри, які повинні бути виконані для успішного завершення проєкту. Це включає в себе детальний опис мети проєкту, основних завдань, які потрібно вирішити, і обґрунтування необхідності проведення робіт.

У ТЗ також визначаються технічні характеристики, функціональні вимоги, параметри якості, терміни виконання та інші критичні аспекти проєкту.

Вступна частина ТЗ описує мету проєкту і завдання, які необхідно вирішити для її досягнення. В розділі обґрунтування необхідності надається аналіз поточної ситуації, виявляються проблеми або потреби, які мають бути вирішені за допомогою проєкту, і пояснюється, чому даний проєкт є важливим та доцільним.

Також ТЗ включає детальний технічний опис, що охоплює вимоги до матеріалів, технологій, інструментів і обладнання, які будуть використовуватися. Крім того, в документі можуть бути вказані методи і засоби контролю якості, вимоги до безпеки та екологічності, а також особливі умови експлуатації і технічного обслуговування об’єкту.

ТЗ також регламентує процес виготовлення та приймання об’єкта, встановлює критерії оцінки відповідності результатів виконаних робіт вимогам завдання, а також визначає порядок введення об’єкта в експлуатацію. Важливою частиною ТЗ є розділ, який описує порядок взаємодії між замовником і виконавцем, а також умови і порядок внесення змін до завдання в ході його виконання.

Таким чином, ТЗ є основним документом, що забезпечує ефективну організацію та контроль за виконанням проєкту, дозволяючи уникнути непорозумінь між сторонами, знизити ризики і забезпечити досягнення поставлених цілей в установлені терміни та з дотриманням необхідних стандартів якості.

2.1.1 Найменування і область застосування

Програмний продукт, що буде розробляться матиме назву «FunFcShakhtar». Програма призначена для слідкування за улюбленим футбольним клубом та вивченням його історії.

2.1.2 Призначення розробки

Даний вебсайт повинен забезпечувати користувача такими можливостями:

- перегляд актуальних новин про футбольний клуб;
- перегляд результатів матчів та розкладу наступних;
- вивчення історії походження клубу;
- перегляд актуального складу команди;
- короткий фотозвіт.

2.1.3 Вимоги до функціональних характеристик

Програма повинна забезпечити можливість виконання наступних функцій:

– забезпечення коректного вводу даних параметрів пошуку;
– забезпечення пошуку, додавання та видалення новин та товару;
– корегувати ціни.

2.1.4 Вхідні та вихідні дані

Вхідними даними є:

– головна сторінка;
– новини;
– склад клубу;
– результати матчів;
– історія клубу.

2.1.5 Етапи розробки сайту

На сьогоднішній день існують такі етапи розробки вебсайту:

– проєктування сайту;
– розробка креативної концепції сайту;
– створення дизайну сайту;
– створення макетів сторінок;
– верстка сторінок і шаблонів;
– розміщення матеріалів сайту;
– тестування і внесення корегувань;
– відкриття проекту (хостинг та викладання в мережу);

– обслуговування працюючого сайту або його програмної основи.

Залежно від поточного завдання, якісь з етапів можуть бути відсутніми, або бути тісно пов’язані один з одним.

Створення технічного завдання. Складанням технічного завдання для фахівців займається менеджер проєкту. Робота з замовником починається з заповнення брифу, в якому замовник викладає свої побажання щодо візуального представлення і структури сайту, наводить приклади сайтів конкурентів. Виходячи з брифу, менеджер складає технічне завдання, враховуючи можливості програмних і дизайнерських засобів. Етап закінчується після затвердження технічного завдання замовником.

Після затвердження технічного завдання, команда розробників приступає до виконання проєкту, орієнтуючись на визначені в ТЗ параметри та вимоги. В процесі розробки можуть виникати питання або потреби в уточненнях, тому важливо, щоб замовник і менеджер проєкту підтримували тісний зв’язок та оперативно вирішували всі питання.

Технічне завдання є основою для всіх подальших дій. Воно визначає не лише візуальне оформлення та функціональність сайту, але й вимоги до продуктивності, безпеки, сумісності з різними пристроями та браузерами. Також у ТЗ можуть бути прописані вимоги щодо майбутнього технічного обслуговування та оновлення сайту.

Менеджер проєкту повинен уважно слідкувати за тим, щоб всі етапи виконувалися у відповідності до технічного завдання, оскільки будь-яке відхилення може призвести до незадовільного результату та додаткових витрат часу та ресурсів. Тому затверджене ТЗ є ключовим документом, який допомагає уникнути непорозумінь і забезпечити високу якість кінцевого продукту.

Після завершення робіт згідно з ТЗ, замовник проводить приймання виконаного проєкту. Якщо всі умови технічного завдання виконані належним чином, проєкт вважається успішно завершеним. У разі необхідності можуть бути проведені додаткові коригування, щоб привести проєкт у повну

відповідність до очікувань замовника.

Дизайн основний і типових сторінок сайту. Починається робота зі створення дизайну, зазвичай в графічному редакторі. Дизайнер створює один або кілька варіантів дизайну відповідно до технічного завдання. Окремо створюється дизайн головної сторінки і дизайни типових сторінок.

Дизайн сторінки представляє собою графічний файл, листковий малюнок. Кількість ескізів і порядок їх надання обговорюється з проєкт-менеджером. Менеджер проєкту здійснює контроль термінів. У великих вебстудіях в процесі бере участь арт-директор, який контролює якість графіки. Етап також закінчується затвердженням ескізу замовником.

Створення дизайну починається з детального аналізу вимог, викладених у технічному завданні. Дизайнер враховує всі побажання замовника щодо кольорової гами, стилю, шрифтів та розташування елементів на сторінці. Перший крок – створення кількох концептуальних ескізів, які відображають загальну ідею та стиль майбутнього сайту. Ці ескізи можуть бути представлені у вигляді швидких начерків або більш детальних візуалізацій.

Після узгодження концепції з замовником, дизайнер переходить до більш детального опрацювання кожного елементу сторінки. Він створює макети головної сторінки та типових внутрішніх сторінок, які включають різні варіанти розташування тексту, зображень, кнопок та інших елементів. Важливою частиною цього етапу є створення графічних елементів, таких як логотипи, іконки, фони та інші декоративні елементи, які надають сайту індивідуальність.

Процес створення дизайну також включає тестування різних варіантів макетів та елементів, щоб знайти найкраще рішення, яке відповідатиме вимогам користувачів і забезпечить зручність навігації. Дизайнер може створювати інтерактивні прототипи, які дозволяють оцінити зручність користування сайтом та зробити необхідні корективи.

Менеджер проєкту контролює дотримання термінів виконання робіт, забезпечуючи своєчасне надання ескізів для затвердження замовником. У

великих вебстудіях в процесі створення дизайну бере участь арт-директор, який відповідає за відповідність графіки високим стандартам якості. Він оцінює роботу дизайнера, надає рекомендації щодо покращення та допомагає вирішувати складні завдання.

Завершальний етап створення дизайну включає остаточне затвердження ескізів замовником. Після отримання схвалення, дизайнер підготовлює усі необхідні файли для подальшої розробки сайту, включаючи графічні елементи у високій роздільній здатності та специфікації для розробників. Це дозволяє забезпечити плавний перехід до наступного етапу – верстки та програмування, де дизайн перетворюється у функціональний вебсайт.

Завершальним етапом розробки сайту є тестування. Процес тестування є важливою складовою розробки вебсайтів та програмного забезпечення. Він включає в себе найрізноманітніші перевірки, спрямовані на виявлення і виправлення помилок, а також на забезпечення високої якості кінцевого продукту.

Тестування може включати перевірку вигляду сторінки зі збільшеними шрифтами, що особливо важливо для користувачів з вадами зору. Це дозволяє переконатися, що текст залишається читабельним та зручним для сприйняття навіть при зміні розміру шрифту.

Також здійснюються перевірки при різних розмірах вікна браузера. Це тестування адаптивності сайту, яке дозволяє забезпечити коректне відображення контенту на різних пристроях, включаючи настільні комп’ютери, планшети та мобільні телефони. Така перевірка дозволяє виявити проблеми з версткою та забезпечити зручність користування незалежно від розміру екрану.

Крім того, процес тестування включає перевірки на кросбраузерну сумісність. Це означає тестування сайту в різних браузерах, таких як Google Chrome, Mozilla Firefox, Safari, Microsoft Edge та інші. Мета такого тестування – переконатися, що сайт виглядає та працює однаково добре у всіх популярних браузерах, оскільки кожен з них може інтерпретувати код по-різному.

Ще один важливий аспект – тестування функціональності. Воно включає перевірку всіх інтерактивних елементів сайту, таких як форми, кнопки, посилання, навігація та інші. Це дозволяє виявити і виправити можливі помилки, що можуть впливати на роботу сайту та зручність користування ним.

Значну увагу приділяється також перевірці швидкодії сайту. Це тестування дозволяє оцінити, наскільки швидко завантажуються сторінки, як працюють скрипти та інші елементи. Висока швидкість завантаження є критично важливою для забезпечення позитивного користувацького досвіду і може впливати на рейтинг сайту в пошукових системах.

Не менш важливим є тестування безпеки. Воно включає перевірку сайту на вразливості, які можуть бути використані зловмисниками для несанкціонованого доступу або втручання в роботу сайту. Це можуть бути перевірки на SQL-ін’єкції, XSS-атаки, CSRF та інші типи атак. Мета такого тестування – забезпечити захист даних користувачів та цілісність роботи сайту.

Підсумовуючи, процес тестування є комплексним і включає безліч різних перевірок. Це дозволяє забезпечити високу якість кінцевого продукту, його зручність, швидкодію та безпеку. Тільки після успішного проходження всіх етапів тестування можна впевнено вводити сайт або програмне забезпечення в експлуатацію.

2.2 Діаграма прецедентів

UML (Universal Modeling Language) – універсальна мова моделювання, розроблена компанією Rational Software з метою створення найбільш оптимальної та універсальної мови для опису як предметної області, так і конкретного завдання в програмуванні. Будь-яке завдання моделюється за допомогою певного набору ієрархічних діаграм, кожна з яких представляє собою певну проекцію системи.

Діаграми прецедентів застосовуються для моделювання виду системи з точки зору зовнішнього спостерігача. На діаграмі прецедентів графічно показана сукупність прецедентів та суб'єктів, а також відносини між ними.

Кожна діаграма UML надає різні аспекти розгляду системи, що дозволяє охопити всі можливі аспекти функціонування та взаємодії компонентів системи. Одним із головних аспектів використання UML є його здатність забезпечити наочне відображення системи, що сприяє кращому розумінню та комунікації між учасниками проєкту. Це досягається через стандартизацію елементів моделювання, що робить UML універсальною мовою для розробників з різних компаній та команд.

Використання UML дозволяє створювати детальні моделі програмного забезпечення, що включають в себе різні рівні абстракції, від високорівневих діаграм вимог до детальних діаграм класів, об'єктів та послідовностей. Це забезпечує чітке розуміння структури та поведінки системи, полегшує процес розробки, тестування та обслуговування програмного забезпечення.

UML також підтримує моделювання динамічних аспектів системи, таких як потоки даних, зміни станів та взаємодії між об'єктами в реальному часі. Це дозволяє ефективно описувати складні бізнес-процеси та сценарії використання, що є критично важливим для великих та розподілених систем.

Завдяки своїй гнучкості та масштабованості UML підходить для моделювання як малих, так і великих проєктів, забезпечуючи можливість поступового вдосконалення та деталізації моделей у процесі розробки.

UML також сприяє повторному використанню модулів та компонентів, що підвищує ефективність розробки та знижує витрати на підтримку та розвиток програмного забезпечення.

Розглянемо основні елементи діаграми прецедентів.

Суб'єкт (actor) – будь-яка сутність, що взаємодіє з системою ззовні або безліч логічно пов'язаних ролей, виконуваних при взаємодії з прецедентами. Стандартним графічним позначенням суб'єкта на діаграмах є фігурка «чоловічка», під якою записується конкретне ім'я суб'єкта, проте суб'єктом

може бути не тільки людина, але і технічне пристроїв о, програма або будь-яка інша система, яка може служити джерелом впливу на моделюючу систему так, як визначить сам розробник (рис. 2.1).

Прецеденти (use case) – це опис безлічі послідовностей дій (включаючи їх варіанти), які виконуються системою для того, щоб актор отримав результат, який має для нього певне значення. При цьому нічого не говориться про те, яким чином буде реалізовано взаємодію суб'єктів з системою, це одна з найважливіших особливостей розробки прецедентів. Стандартним графічним позначенням прецеденту на діаграмах є еліпс (рис. 2.1), всередині якого міститься коротка назва прецеденту або ім'я у формі дієслова з пояснювальними словами.

Діаграма прецедентів демонструє, що для сайта, що проектується, передбачено 2 види користувачів: користувачі та адміністратор.

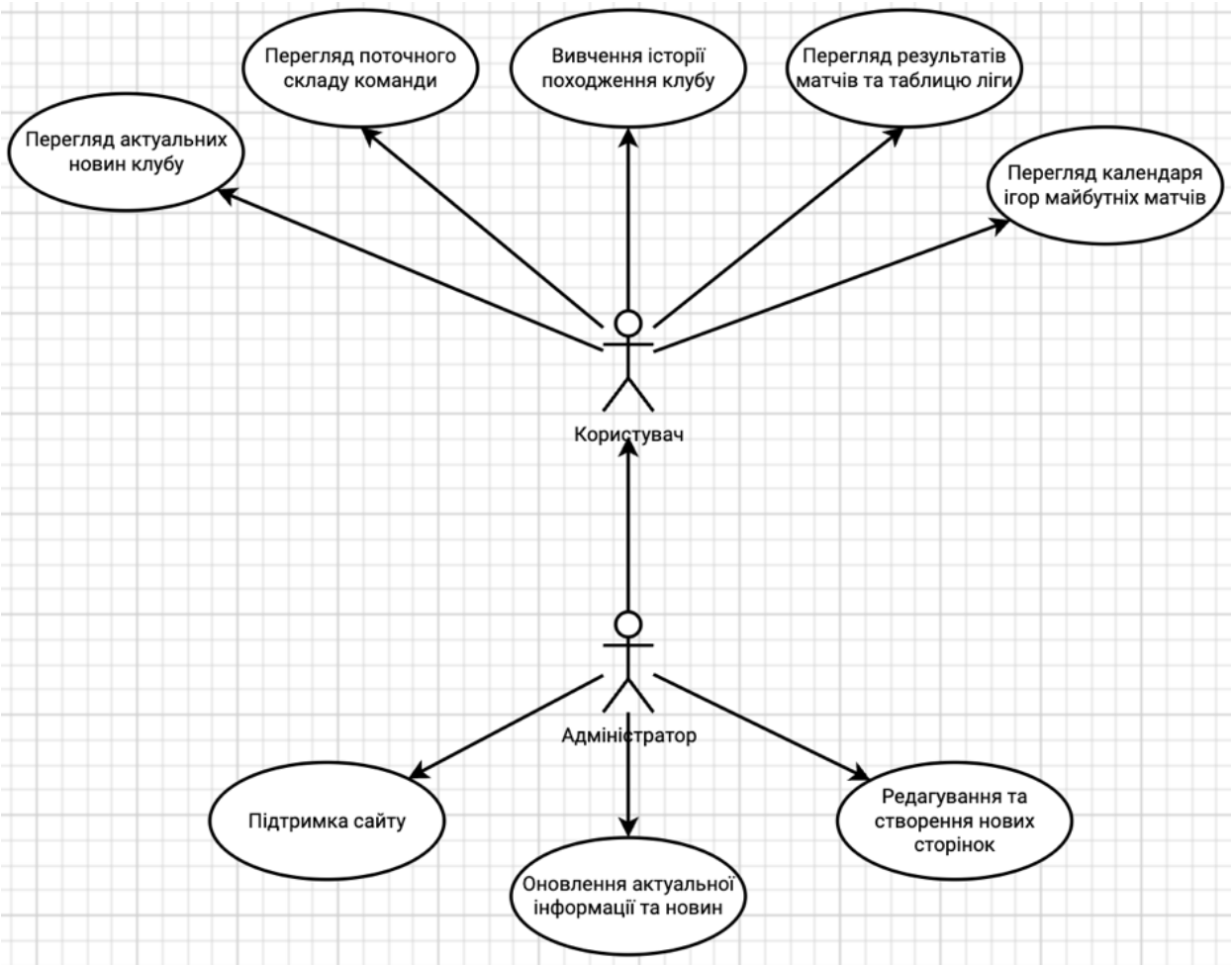

Рисунок 2.1 – Діаграма прецедентів

2.3 Макет сайту

Дизайн-макет сайту – це візуальний образ майбутнього сайту, розроблений з урахуванням технічних можливостей HTML верстки. Такий макет є демонстрацією того, як візуально буде виглядати ваш сайт після верстки і наповнення [5]. Дизайн-макет дозволяє побачити загальну структуру сторінок, розташування основних елементів, таких як заголовки, тексти, зображення, форми та навігаційні панелі. Це важливий етап у процесі створення сайту, оскільки він допомагає визначити, чи відповідає дизайн вимогам замовника і чи зручною буде взаємодія користувачів із сайтом. На етапі розробки дизайн-макету важливо враховувати принципи юзабіліті та доступності, щоб забезпечити зручність користування сайтом для всіх категорій користувачів, включаючи людей з обмеженими можливостями. Дизайн-макет також допомагає визначити кольорову гамму, шрифти, розміри та стилі текстових елементів, що в свою чергу впливає на загальне враження від сайту. Правильно підібрана кольорова гамма та шрифти можуть значно покращити читабельність та привабливість сайту, а також створити певну атмосферу, що відповідає тематиці та меті сайту.

Процес створення дизайн-макету включає кілька етапів. Спочатку дизайнер збирає вимоги та побажання замовника, аналізує цільову аудиторію та конкурентів. Після цього створюється кілька чорнових варіантів макету, які проходять обговорення та затвердження з замовником. Після затвердження основного концепту дизайнер приступає до деталізації макету, додаючи всі необхідні елементи та підготовлюючи його до верстки. На цьому етапі також можуть вноситися корективи та покращення, що забезпечить відповідність макету технічним можливостям верстки та вимогам замовника.

Важливою частиною процесу є тестування макету на різних пристроях та роздільних здатностях екрану. Це дозволяє переконатися, що сайт буде коректно відображатися та працювати на різних платформах, включаючи

настільні комп’ютери, планшети та смартфони. Адаптивний дизайн стає все більш важливим у сучасному вебдизайні, оскільки кількість користувачів, що переглядають сайти з мобільних пристроїв, постійно зростає. Дизайн-макет також має враховувати технічні можливості та обмеження сучасних браузерів, щоб уникнути проблем із сумісністю та забезпечити стабільну роботу сайту у різних умовах. Це може включати перевірку сумісності з різними версіями браузерів, використання стандартних шрифтів та графічних форматів, а також оптимізацію зображень та інших медіа-елементів для швидкого завантаження.

Після завершення розробки дизайн-макету його передають верстальнику, який на основі макету створює HTML-код сайту. Це включає розміщення всіх елементів на сторінці, застосування стилів CSS для оформлення та використання JavaScript для додавання інтерактивних елементів. Верстка повинна точно відповідати дизайн-макету, щоб зберегти задуманий зовнішній вигляд та функціональність сайту. У процесі верстки також можуть виникати необхідність внесення незначних змін або оптимізацій, що забезпечить коректну роботу сайту у різних умовах [6].

Таким чином, дизайн-макет сайту є важливим етапом у процесі розробки вебресурсу, що дозволяє візуалізувати кінцевий результат, узгодити всі деталі з замовником та забезпечити відповідність сайту всім вимогам і стандартам. Він служить основою для подальшої верстки та програмування, допомагаючи створити якісний та привабливий продукт, що відповідає очікуванням користувачів.

Макет представляється у вигляді картинки, яка буде відображена в інтернет браузері, без активних кнопок і інших динамічних елементів.

На даному макеті (рис. 2.2) представлено головну сторінку вебсайту, як можна помітити, головна сторінка має хедер, в моєму випадку це емблема клубу, футер, навігацію, центральну частину з чотирма інформаційними блоками, три бічних блоки. Головна сторінка стилізована так само, як і всі інші сторінки вебсайту, задля гарного візуального вигляду вебсайту.

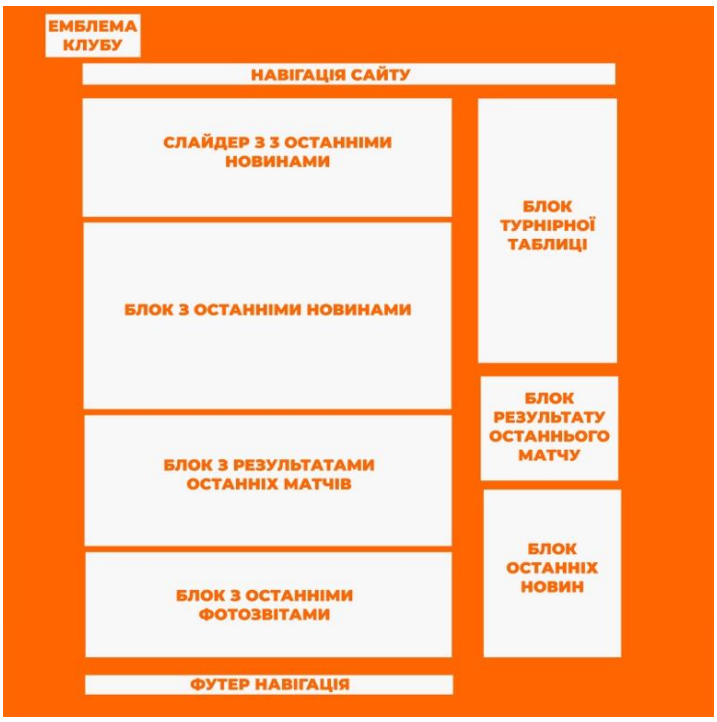

Рисунок 2.2 – Макет головної сторінки

На макеті сторінки “Клуб” ми можемо побачити меншу кількість блоків, так як на цій сторінці представлено лише один центральний блок з історичним походженням клубу (рис. 2.3).

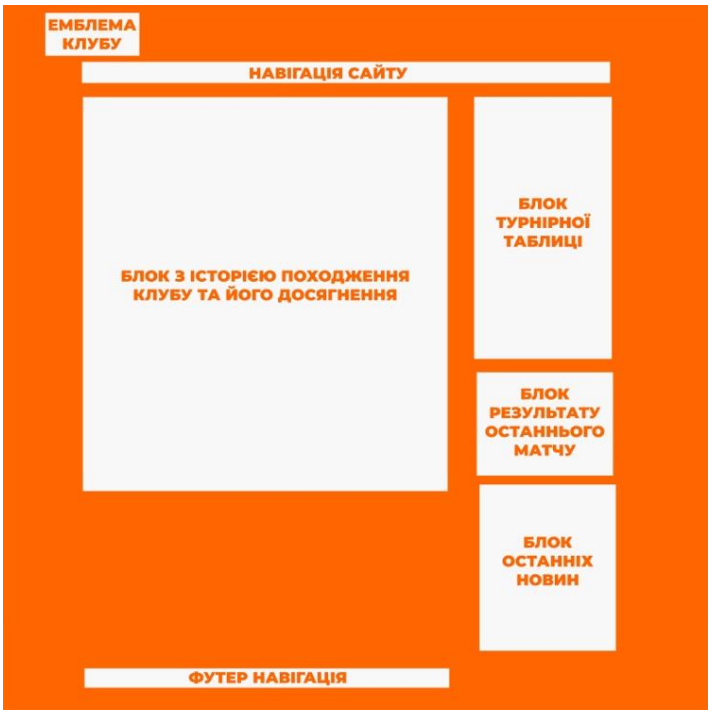

Рисунок 2.3 – Макет сторінки “Клуб”

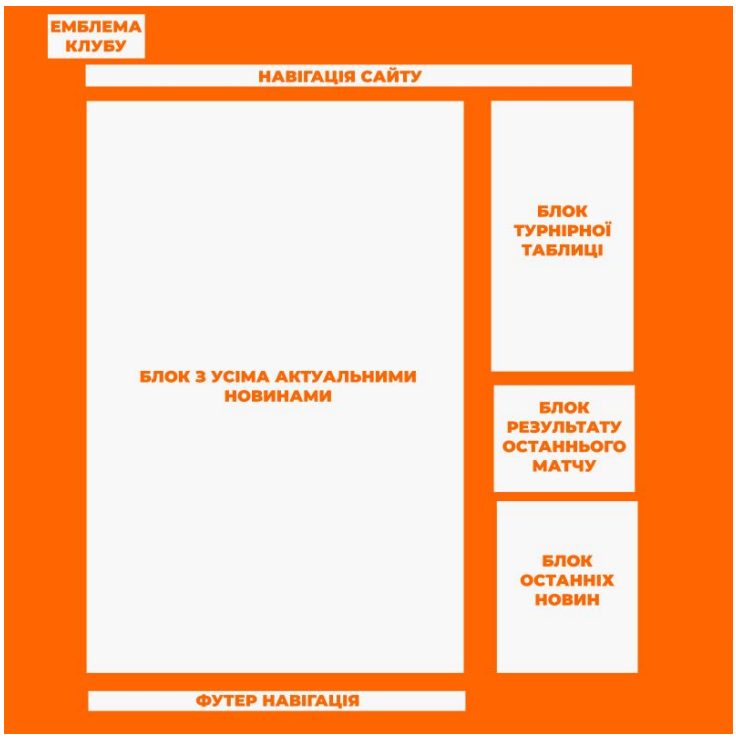

Рисунок 2.4 – Макет сторінки “Новини”

На макеті сторінки “Команда” представлено повний тренерський та основний склад футбольного клубу (рис. 2.5). Кожному тренеру та гравцю присвячена своя окрема картка.

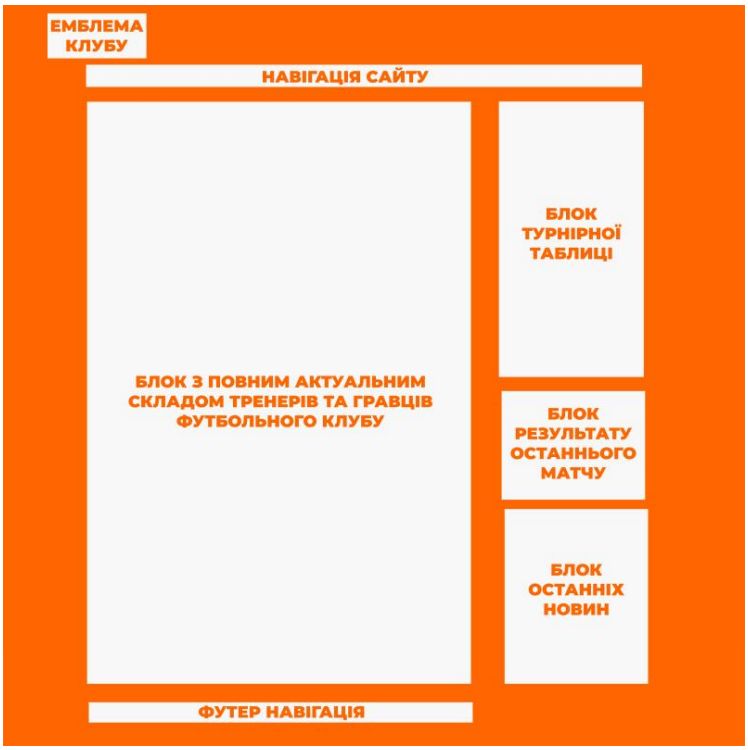

Рисунок 2.5 – Макет сторінки “Команда”

На сторінці “Сезон” представлено турнірну таблицю у повному розмірі з усіма статистиками та повний календар ігор за весь сезон (рис. 2.6).

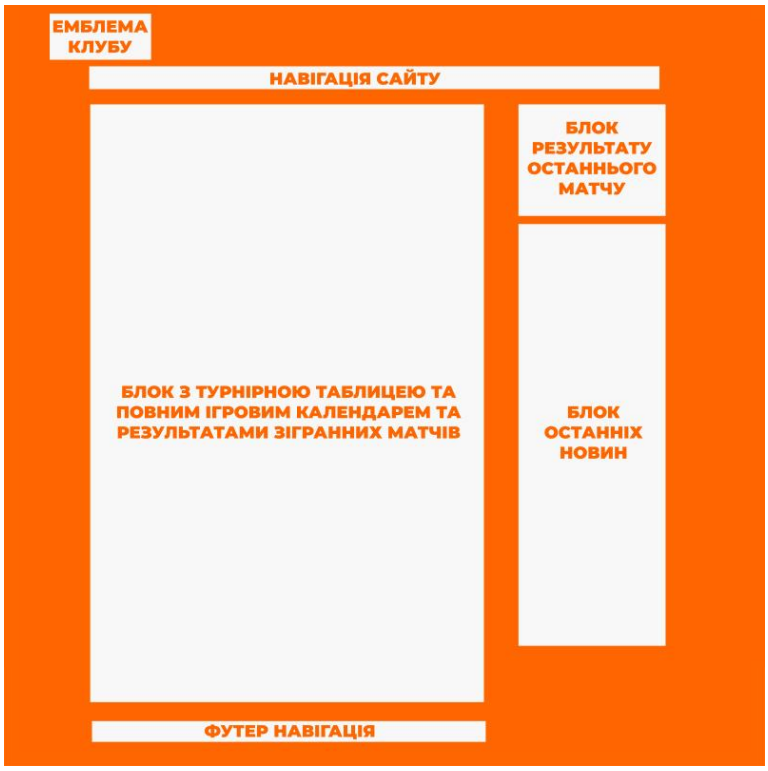

Рисунок 2.6 – Макет сторінки “Сезон”

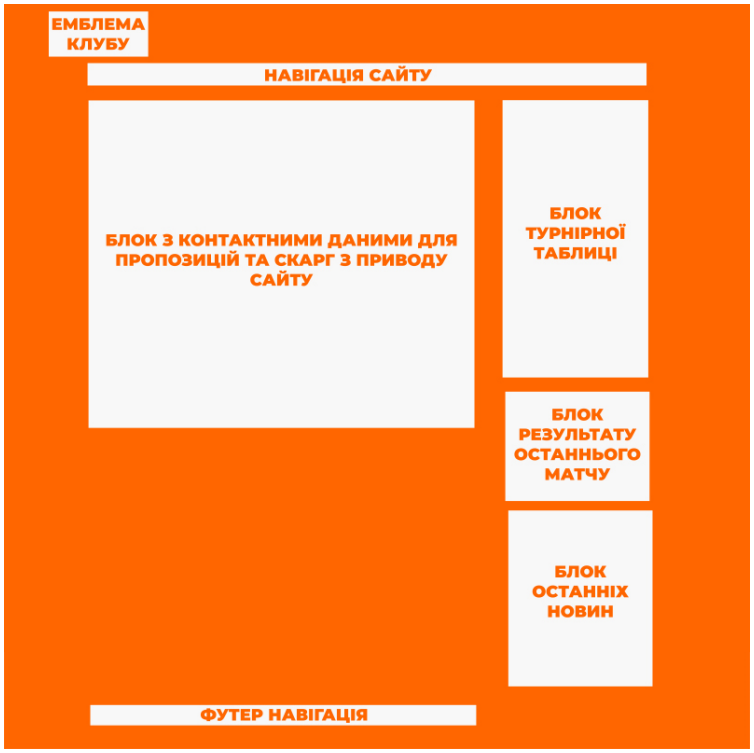

Рисунок 2.7 – Макет сторінки “Контакти”

В другому розділі ми розглянули проектування веб-додатку. Було створено технічне завдання, діаграма прецедентів, макети сайту та описано етапи розробки сайту. Сайт був оформлений в одному стилі , задля приємного користування.

3 РОЗРОБКА ВЕБСАЙТУ

3.1 Середовище та методи розробки

HTML є основою будь-якого вебсайту. Це мова розмітки, яка використовується для створення структури вебсторінок. Кожна вебсторінка складається з різних елементів, таких як заголовки, абзаци, посилання, зображення, таблиці тощо, і саме HTML визначає, як ці елементи організовані на сторінці.

Переваги HTML:

- простота та доступність: HTML є легкою для розуміння та використання мовою – навчитися базовим принципам HTML можна за короткий час, що робить його доступним навіть для початківців;
- універсальність: HTML підтримується всіма веббраузерами, що забезпечує сумісність вебсторінок на різних платформах та пристроях;
- стандартизація: HTML є стандартом, який підтримується World Wide Web Consortium (W3C), що гарантує стабільність та надійність;
- розширюваність: HTML можна розширювати за допомогою інших мов розмітки, таких як XML, а також інтегрувати з іншими технологіями, такими як CSS та JavaScript.

CSS використовується для стилізації вебсторінок. За допомогою CSS можна визначити зовнішній вигляд та розташування HTML-елементів на сторінці. CSS дозволяє змінювати кольори, шрифти, відступи, розміри, анімації тощо.

Переваги CSS:

- поділ змісту та стилю: CSS дозволяє відокремити візуальне оформлення від змісту вебсторінки – це робить код більш організованим та легшим для підтримки;

- гнучкість та контроль: CSS надає детальний контроль над виглядом сторінки – ви можете точно налаштувати кожен елемент відповідно до своїх потреб;
- повторне використання стилів: CSS дозволяє використовувати одні й ті самі стилі на різних сторінках, що зменшує обсяг коду та спрощує його підтримку;
- адаптивний дизайн: CSS медіа-запити дозволяють створювати адаптивні дизайни, що коректно відображаються на різних пристроях та екранах різних розмірів [7].

JavaScript є мовою програмування, яка дозволяє додати інтерактивність та динамічні елементи на вебсторінки. За допомогою JavaScript можна створювати функції, які реагують на дії користувача, змінюють вміст сторінки без перезавантаження, верифікують дані форм, а також багато інших можливостей.

Переваги JavaScript:

- інтерактивність: JavaScript дозволяє створювати багаті, інтерактивні вебдодатки, що покращують взаємодію користувача зі сторінкою;
- динамічний контент: за допомогою JavaScript можна динамічно змінювати контент на сторінці, що дозволяє створювати більш сучасні та зручні для користувача вебдодатки;
- широка підтримка бібліотек та фреймворків: JavaScript має безліч бібліотек (наприклад, jQuery, React, Angular, Vue.js), які спрощують розробку та додають додаткові можливості;
- взаємодія з сервером: JavaScript дозволяє виконувати асинхронні запити до сервера (AJAX), що забезпечує швидше та ефективніше завантаження даних без необхідності перезавантаження сторінки.

Visual Studio Code – це потужне середовище розробки (IDE), яке надає розробникам інструменти для створення вебдодатків. Використання Visual Studio для розробки вебсайтів має безліч переваг.

Переваги Visual Studio Code:

- інтегроване середовище розробки: Visual Studio Code надає всі необхідні інструменти для написання, тестування та налагодження коду в одному місці;
- підтримка кількох мов програмування: Visual Studio Code підтримує HTML, CSS, JavaScript, а також інші мови, що робить його універсальним інструментом для веброзробки;
- автоматичне завершення коду: функція IntelliSense в Visual Studio Code допомагає автоматично завершувати код, що прискорює процес розробки та зменшує кількість помилок;
- інструменти для налагодження: Visual Studio Code надає потужні інструменти для налагодження, які допомагають знайти та виправити помилки у коді;
- інтеграція з системами контролю версій: Visual Studio Code легко інтегрується з Git та іншими системами контролю версій, що полегшує спільну роботу над проєктами.

Розробка вебсайтів за допомогою HTML, CSS та JavaScript у середовищі Visual Studio Code є ефективним та потужним підходом. HTML забезпечує структуру сторінки, CSS – її стилізацію, а JavaScript – інтерактивність. Visual Studio Code, у свою чергу, надає всі необхідні інструменти для продуктивної роботи та полегшує процес розробки завдяки своїм багатофункціональним можливостям.

Дизайн, управління системою та інші можливості:

- простота встановлення, простота налаштувань;
- підтримка вебстандартів (XHTML, CSS);
- модулі для підключення (плагіни) з унікально простою системою їх взаємодії з кодом; можливість автоматичного встановлення та оновлення версії безпосередньо з панелі адміністратора;
- підтримка тем ,з допомогою яких легко змінюється як зовнішній

вигляд, так і способи виведення даних;

– можливість редагувати шаблони одразу в панелі адміністратора;
– теми реалізовані як набори файлів-шаблонів на PHP;
– велика кількість бібліотек тем і плагінів;
– СЕО-оптимізована система.

3.2 Реалізація функціоналу вебсайту

У шапці сайту відображено емблему футбольного клубу та його назву (рис. 3.1).

Рисунок 3.1 – Шапка сайту

У навігації можна побачити такі сторінки: «Новини», «Клуб», «Команда», «Сезон» та «Контакти» (рис. 3.2).

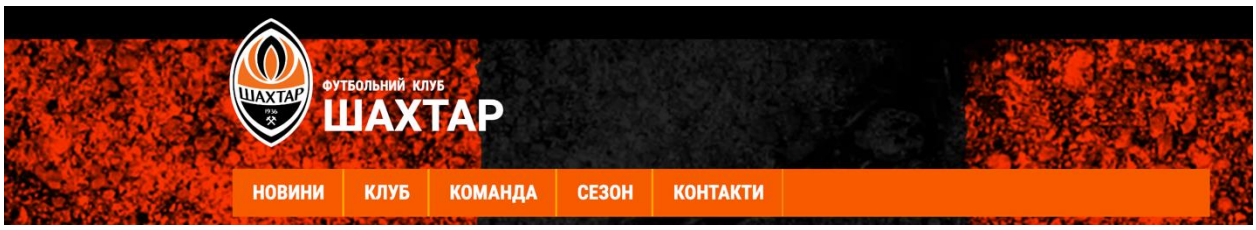

Рисунок 3.2 – Навігація сайту

Програмна реалізація навігації виконана та стилізована наступним чином (рис. 3.3 – 3.4).

На головній сторінці сайту є інформаційний слайдер, де будуть відображатися останні 3 новини футбольного клубу (рис. 3.5).

```html
<nav>
    <ul class="main-menu">
        <li class="sub">
            <a href="./pages/news.html">Новини</a>
        </li>
        <li class="sub">
            <a href="./pages/club.html">Клуб</a>
        </li>
        <li class="sub">
            <a href="./pages/team.html">Команда</a>
        </li>
        <li class="sub">
            <a href="./pages/season.html">Сезон</a>
        </li>
        <li class="sub">
            <a href="./pages/contacts.html">Контакти</a>
        </li>
    </ul>
</nav>
```

Рисунок 3.3 – HTML код навігації

```css
nav {
    background: #f75b00;
    clear: both;
    height: 48px;
}
.main-menu {
    list-style: none;
    display: block;
}
.main-menu > li {
    float: left;
    position: relative;
    border-right: 1px solid orange;
    border-left: 1px solid orange;
}
.main-menu > li > a {
    padding: 12px 20px 12px 20px;
    font-size: 20px;
    font-weight: 700;
    font-family: 'Roboto Condensed', sans-serif;
    color: white;
    text-decoration: none;
    text-transform: uppercase;
    display: block;
}
```

Рисунок 3.4 – CSS код навігації сайту

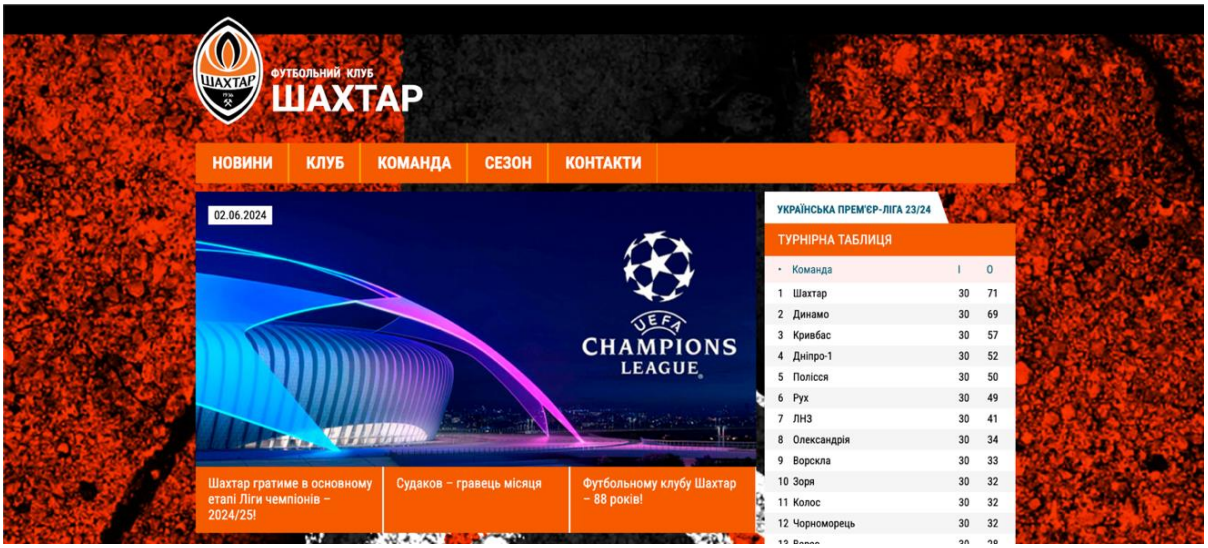

Рисунок 3.5 – Слайдер з новинами

Для цього слайдеру та його функціональності було написано скрипт на JavaScript (рис. 3.6).

```
document.addEventListener('DOMContentLoaded', function () {
    const sliderItems = document.querySelectorAll('.slider-item')
    const sliderImg = document.querySelector('.slider-img img')
    const sliderLink = document.querySelector('.slider-img')
    const sliderDate = document.querySelector('.slider-date')

    function updateSlider(item) {
        const imgSrc = item.getAttribute('data-img')
        const linkHref = item.getAttribute('data-link')
        const dateText = item.getAttribute('data-date')

        sliderImg.src = imgSrc
        sliderLink.href = linkHref
        sliderDate.textContent = dateText
    }

    sliderItems.forEach(item => {
        item.addEventListener('click', function () {
            // Remove 'active' class from all items
            sliderItems.forEach(i => i.classList.remove('active'))

            // Add 'active' class to the clicked item
            item.classList.add('active')

            // Update the main image, link, and date
            updateSlider(item)
        })
    })

    // Display the first news item by default
    updateSlider(document.querySelector('.slider-item.active'))
})
```

Рисунок 3.6 – Скрипт роботи слайдеру на головній сторінці

На цьому слайдері можна обрати одну з новин до перегляду. Його програмна реалізація та стилізація виконана за допомогою наступного коду (рис. 3.7 – 3.8).

```
<div class="slider">
    <a href="#" class="slider-img">
        <img src="" alt="" />
        <div class="slider-date"></div>
    </a>
    <div class="slider-items">
        <div
            class="slider-item active"
            data-img="./assets/img/new2.jpg"
            data-date="02.06.2024"
            data-link=""
        >
            <a href="#"
                >Шахтар гратиме в основному етапі Ліги чемпіонів – 2024/25!</a
            >
        </div>
        <div
            class="slider-item"
            data-img="./assets/img/new1.jpg"
            data-date="31.05.2024"
            data-link=""
        >
            <a href="#"
                >Судаков – гравець місяця</a
            >
        </div>
        <div
            class="slider-item"
            data-img="./assets/img/new3.jpg"
            data-date="24.05.2024"
            data-link=""
        >
            <a
                >Футбольному клубу Шахтар – 88 років!</a
            >
        </div>
    </div>
</div>
```

Рисунок 3.7 – HTML реалізація слайдеру

```css
.slider {
    height: 410px;
}
.slider-img {
    display: block;
    height: 329px;
}
.slider-img img {
    vertical-align: bottom;
    width: 100%;
    height: 329px;
    object-fit: cover;
}
.slider-item {
    width: 195px;
    float: left;
    background: #f75b00;
    padding: 10px 11px 10px 15px;
    border-right: 1px solid #fff;
    border-left: 1px solid #fff;
    overflow: hidden;
}
.slider-item a {
    color: white;
    font-family: 'Roboto Condensed', sans-serif;
    font-size: 16px;
    text-decoration: none;
    display: inline-block;
    max-height: 55px;
    min-height: 55px;
    overflow: hidden;
}
.slider-items div.active {
    background: #f75b00;
    border-right: 1px solid #fff;
    min-height: 57px;
}
.slider-items div:first-child {
    border-left: none;
    width: 197px;
}
.slider-img {
    position: relative;
}
.slider-date {
    position: absolute;
    top: 17px;
    left: 15px;
    background: white;
    font-size: 14px;
    font-family: 'Roboto Condensed', sans-serif;
    color: black;
    padding: 3px 7px;
```

Рисунок 3.8 – CSS реалізація слайдеру

Також на головній сторінці можемо побачити блок з трьома останніми матчами футбольного клубу та його фінальним рахунком, з додатковою кнопкою переходу до вкладки усіх матчів та турнірної таблиці (рис. 3.9).

Рисунок 3.9 – Блок з останніми матчами

Надається html та css коди реалізації цього блоку (рис. 3.10 – 3.11).

```
<div class="news">
    <div class="news-title-wrapper">
        <div class="news-title">Останні матчі</div>
        <div class="news-all"><a href="./pages/season.html">Усі матчі</a></div>
    </div>
    <div class="match-item">
        <a>
            <h2 class="match-header">
                Шахтар – Полісся | Українська Прем'єр-Ліга 23/24 | Тур 30 |
                2024-05-25 | 15:30 | Житомир | Центральний стадіон
            </h2>
            <div class="match-content">
                <div class="last-match-logo">
                    <img
                        src="./assets/img/fcshakh.svg.png"
                        alt=""
                    />
                </div>
                <div class="match-result">
                    <div class="goal-count">0</div>
                    <div class="goal-count">2</div>
                </div>
                <div class="last-match-logo">
                    <img
                        src="./assets/img/pol.png"
                        alt=""
                    />
                </div>
            </div>
        </a>
    </div>
    <div class="match-item">
        <a>
            <h2 class="match-header">
                Шахтар – Дніпро-1 | Українська Прем'єр-Ліга 23/24 | Тур
                29 | 2024-05-19 | 15:30 | Дніпропетровська обл. |  Арена Дніпро
            </h2>
            <div class="match-content">
                <div class="last-match-logo">
                    <img
                        src="./assets/img/fcshakh.svg.png"
                        alt=""
                    />
                </div>
                <div class="match-result">
                    <div class="goal-count">1</div>
                    <div class="goal-count">1</div>
                </div>
                <div class="last-match-logo">
                    <img
                        src="./assets/img/dnipr1.png"
                        alt=""
                    />
                </div>
            </div>
        </a>
```

Рисунок 3.10 – HTML код блоку “Останні матчі”

```
.match-item {
    padding: 0 0 20px 15px;
    overflow: hidden;
}
.match-item h2.match-header {
    font-size: 14px;
    margin: 5px 0;
    font-family: 'Roboto Condensed', sans-serif;
    color: #f75b00;
    text-transform: uppercase;
    position: relative;
    padding-left: 13px;
}
.match-item h2.match-header:before {
    content: '';
    position: absolute;
    height: 16px;
    width: 3px;
    background: black;
    left: 0;
    top: 3px;
}
.match-item a {
    text-decoration: none;
}
.match-item .match-content {
    height: 80px;
    max-height: 80px;
    overflow: hidden;
    border-top: 1px solid #f0f0f0;
    border-bottom: 1px solid #f0f0f0;
    margin: 0 auto;
    width: 382px;
    padding: 16px 0 0 0;
}
.match-item .match-content .last-match-logo,
.match-item .match-content .match-result {
    float: left;
}
.match-item .match-content .last-match-logo {
    width: 120px;
    text-align: center;
}
.match-item .match-content .last-match-logo img {
    vertical-align: top;
    max-width: 100%;
    max-height: 78px;
}
.match-item .match-content .match-result {
    padding: 0;
    width: 142px;
}
```

Рисунок 3.11 – CSS код блоку “Останні матчі”

Далі йде блок з фотографіями останніх подій, який буде оновлюватися приблизно раз у 2 тижні (рис. 3.12).

Рисунок 3.12 – Блок Фотоновин

Його програмна реалізація та стилізація виконана за допомогою наступного коду (рис. 3.13 – 3.14).

З правого боку від цих блоків, розташовуються додаткові три блоки, перший з них це турнірна таблиця, яку можна буде побачити на кожній сторінці сайту, вона відображає поточну статистику матчів та очок кожної команди (рис. 3.15).

```
<div class="photo-news">
    <div class="photo-news-title">Фотоновини</div>
    <div class="news-all">
    </div>
    <a
    >
        <img
            src="./assets/img/photonew1.jpg"
            alt=""
    /></a>
    <a
        ><img
            src="./assets/img/photonew2.jpg"
            alt=""
    /></a>
    <a
        ><img
            src="./assets/img/photonew3.jpg"
            alt=""
    /></a>
    <a
        ><img
            src="./assets/img/photonew4.jpg"
            alt=""
    /></a>
    <a
        ><img
            src="./assets/img/photonew5.jpg"
            alt=""
    /></a>
</div>
```

Рисунок 3.13 – HTML код блоку “Фотоновини”

```css
.photo-news {
    position: relative;
    margin-top: 10px;
    margin-bottom: 10px;
    overflow: hidden;
}
.photo-news-title {
    display: inline-block;
    font-size: 18px;
    font-family: 'Roboto Condensed', sans-serif;
    font-weight: 700;
    color: white;
    text-transform: uppercase;
    line-height: 38px;
    padding: 0 15px;
    background: #f75b00;
    position: absolute;
    top: 15px;
    left: 0;
    z-index: 10;
}
.photo-news img {
    vertical-align: bottom;
}
.photo-news > a {
    display: block;
    float: left;
    width: 223px;
    position: relative;
    overflow: hidden;
}
.photo-news > a.photo-news-main {
    width: 446px;
}
.photo-news .news-all {
    position: absolute;
    right: 0px;
    top: 7px;
    z-index: 10;
}
.news-item:last-child {
    padding: 0 0 15px 15px;
}
.photo-news > a:before {
    content: '';
    position: absolute;
    left: -100%;
    top: 0;
    width: 100%;
    height: 100%;
    background: rgba(87, 176, 227, 0.75) url('../img/photo-h.png') center
        no-repeat;
    z-index: 2;
```

Рисунок 3.14 – CSS код блоку “Фотоновини”

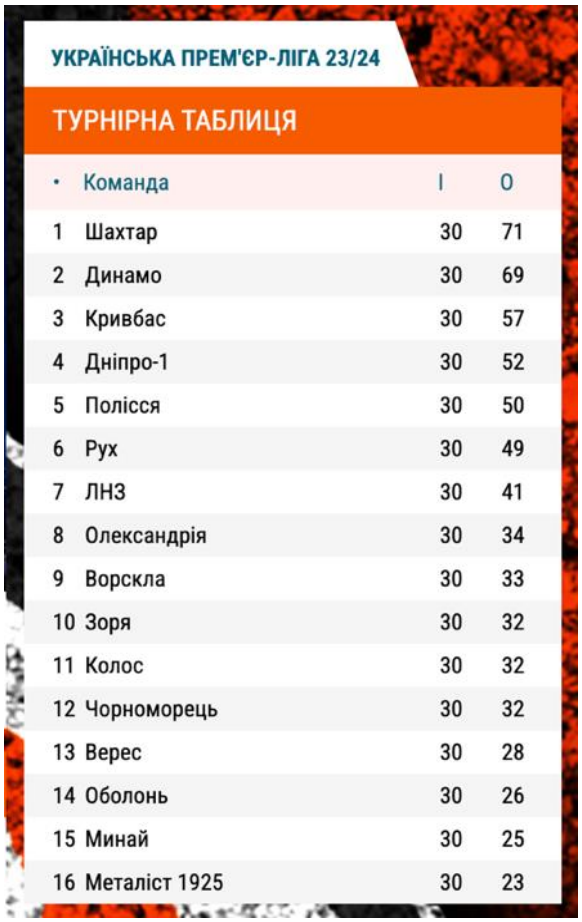

Рисунок 3.15 – Блок з турнірною таблицею

Його стилізація та реалізація на сторінці буде представлена далі (рис. 3.16 – 3.17).

```
<div class="sidebar">
    <div class="championships table-main">
        <div class="tab championship-active" data-id="17">
            Українська Прем'єр-Ліга 23/24
        </div>

        <table id="t17" class="show-table">
            <caption>
                Турнірна таблиця
            </caption>
            <thead>
                <tr>
                    <th>•</th>
                    <th>Команда</th>
                    <th>І</th>
                    <th>О</th>
                </tr>
            </thead>
            <tbody>
                <tr>
                    <td>1</td>
                    <td>Шахтар</td>
                    <td>30</td>
                    <td>71</td>
                </tr>
                <tr>
                    <td>2</td>
                    <td>Динамо</td>
                    <td>30</td>
                    <td>69</td>
                </tr>
                <tr>
                    <td>3</td>
                    <td>Кривбас</td>
                    <td>30</td>
                    <td>57</td>
                </tr>
                <tr>
                    <td>4</td>
                    <td>Дніпро-1</td>
                    <td>30</td>
                    <td>52</td>
                </tr>
                <tr>
                    <td>5</td>
                    <td>Полісся</td>
                    <td>30</td>
                    <td>50</td>
                </tr>
                <tr>
                    <td>6</td>
                    <td>Рух</td>
                    <td>30</td>
```

Рисунок 3.16 – Приклад html коду блоку “Турнірна таблиця”

```
                <tr>
                    <td>7</td>
                    <td>ЛНЗ</td>
                    <td>30</td>
                    <td>41</td>
                </tr>
                <tr>
                    <td>8</td>
                    <td>Олександрія</td>
                    <td>30</td>
                    <td>34</td>
                </tr>
                <tr>
                    <td>9</td>
                    <td>Ворскла</td>
                    <td>30</td>
                    <td>33</td>
                </tr>
                <tr>
                    <td>10</td>
                    <td>Зоря</td>
                    <td>30</td>
                    <td>32</td>
                </tr>
                <tr>
                    <td>11</td>
                    <td>Колос</td>
                    <td>30</td>
                    <td>32</td>
                </tr>
                <tr>
                    <td>12</td>
                    <td>Чорноморець</td>
                    <td>30</td>
                    <td>32</td>
                </tr>
                <tr>
                    <td>13</td>
                    <td>Верес</td>
                    <td>30</td>
                    <td>28</td>
                </tr>
                <tr>
                    <td>14</td>
                    <td>Оболонь</td>
                    <td>30</td>
                    <td>26</td>
                </tr>
                <tr>
                    <td>15</td>
                    <td>Минай</td>
                    <td>30</td>
                    <td>25</td>
                </tr>
                <tr>
                    <td>16</td>
                    <td>Металіст 1925</td>
                    <td>30</td>
                    <td>23</td>
                </tr>
            </tbody>
        </table>
    </div>
```

Рисунок 3.17 – Приклад html коду блоку “Турнірна таблиця”

Нижче турнірної таблиці розташовується блок під назвою “Остання гра”, на якому відтворюється результат останнього матчу футбольного клубу (рис. 3.18).

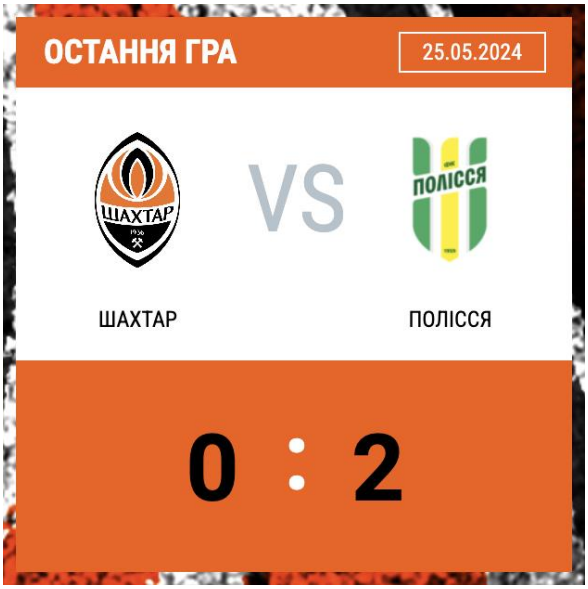

Рисунок 3.18 – Блок “Остання гра”

Далі буде наведено приклад html та css кодів реалізації цього блоку (рис. 3.19 – 3.20).

```
<div class="last-match">
    <div class="last-match-title">
        <span class="match-title">Остання гра</span>
        <span class="time">25.05.2024</span>
    </div>
    <div class="last-match-teams">
        <div class="last-match-team">
            <div class="last-match-logo">
                <img
                    src="./assets/img/fcshakh.svg.png"
                    alt=""
                />
            </div>
            <div class="last-match-name">Шахтар</div>
        </div>
        <div class="last-match-team">
            <div class="last-match-logo">
                <img
                    src="./assets/img/pol.png"
                    alt=""
                />
            </div>
            <div class="last-match-name">Полісся</div>
        </div>
    </div>
    <div class="match-result">
        <div class="goal-count">0</div>
        <div class="goal-count">2</div>
    </div>
</div>
```

Рисунок 3.19 – Приклад html коду блоку “Остання гра”

```css
.last-match {
    background: #f75b00 url('../img/last-match.jpg') left no-repeat;
    margin: 12px 0 0 0;
    min-height: 299px;
}
.last-match-title span.match-title {
    font-size: 18px;
    font-family: 'Roboto Condensed', sans-serif;
    font-weight: 700;
    color: white;
    text-transform: uppercase;
    line-height: 38px;
    padding: 0 15px;
    display: inline-block;
    float: left;
    text-decoration: none;
}
.last-match-title span.time {
    display: inline-block;
    float: right;
    font-size: 12px;
    font-family: 'Roboto Condensed', sans-serif;
    color: #ffffff;
    text-decoration: none;
    padding: 3px 12px;
    border: 1px solid white;
    background: transparent;
    margin-right: 14px;
    margin-top: 8px;
}
.last-match-title {
    clear: both;
    overflow: hidden;
}
.last-match-teams {
    background: #ffffff;
    overflow: hidden;
    padding: 22px 0 0 0;
}
.last-match-name {
    text-transform: uppercase;
    font-size: 12px;
    font-family: 'Roboto Condensed', sans-serif;
    color: #000000;
}
```

Рисунок 3.20 – Приклад css коду блоку “Остання гра”

Далі переходимо на сторінку “Новини”, вона відокремлюється від головної сторінки тим, що на ній зібрані всі новини футбольного клубу. На кожну з новин в подальшому буде розроблена окрема сторінка для детального опису новини (рис. 3.21).

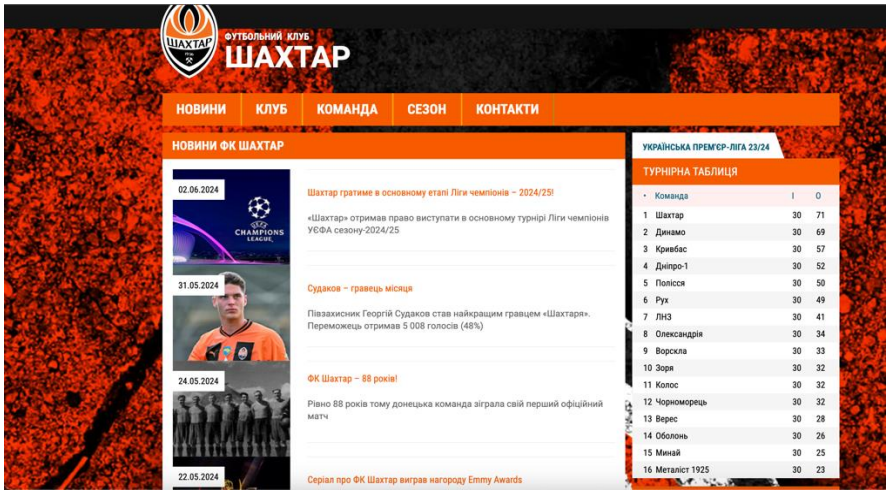

Рисунок 3.21 – Сторінка “Новини”

Наступною нас зустрічає сторінка “Клуб” на якій відтворена історія походження футбольного клубу “Шахтар” в текстовому форматі (рис. 3.22).

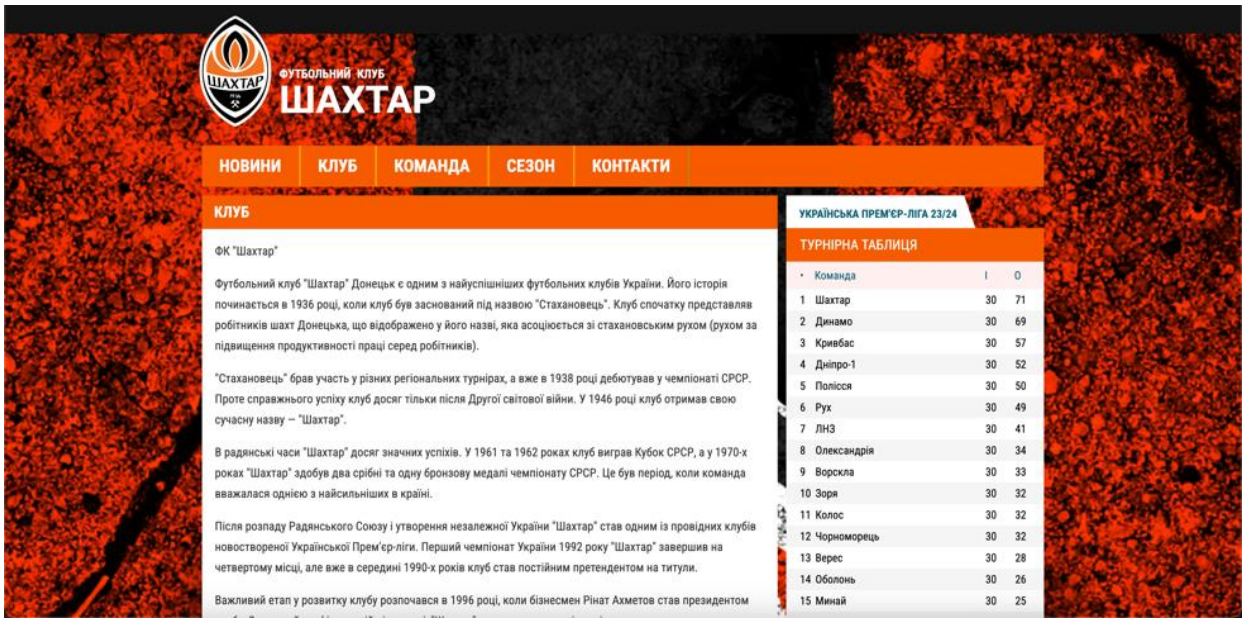

Рисунок 3.22 – Деталі товару та схожі товари

Переходимо далі до сторінки під назвою “Команда”, де показано весь актуальний тренерський та основний склад футбольного клубу, кожному тренеру та футболісту присвячена окрема картка (рис. 3.23 – 3.25).

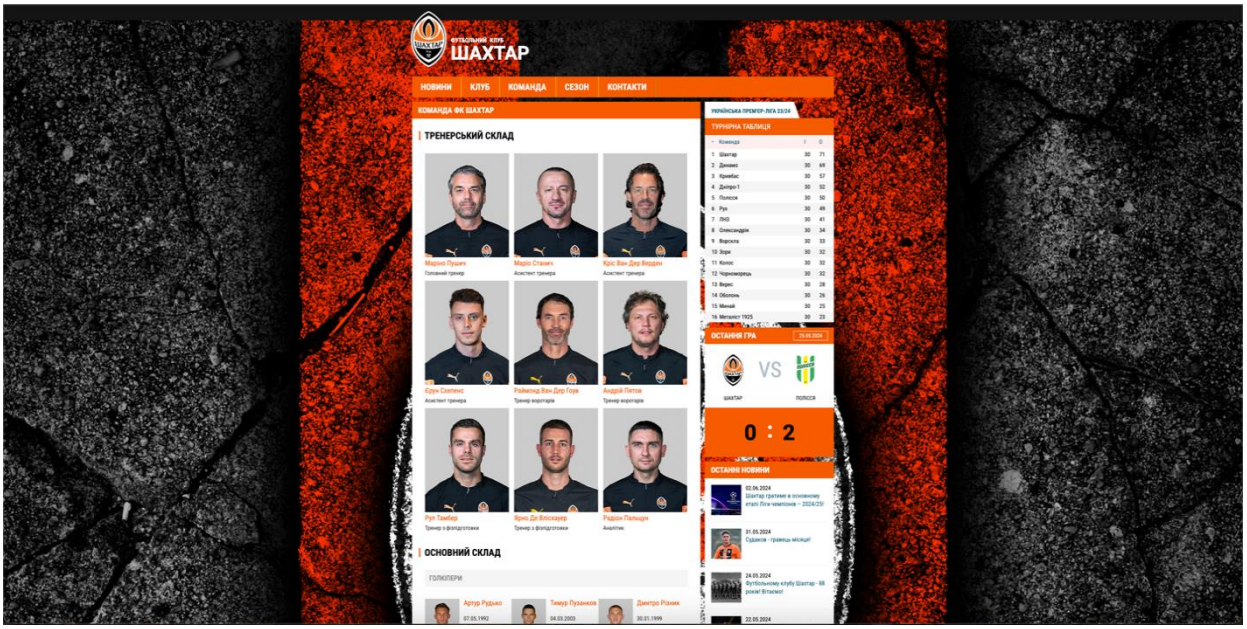

Рисунок 3.23 – Склад тренерів на сторінці “Команда”

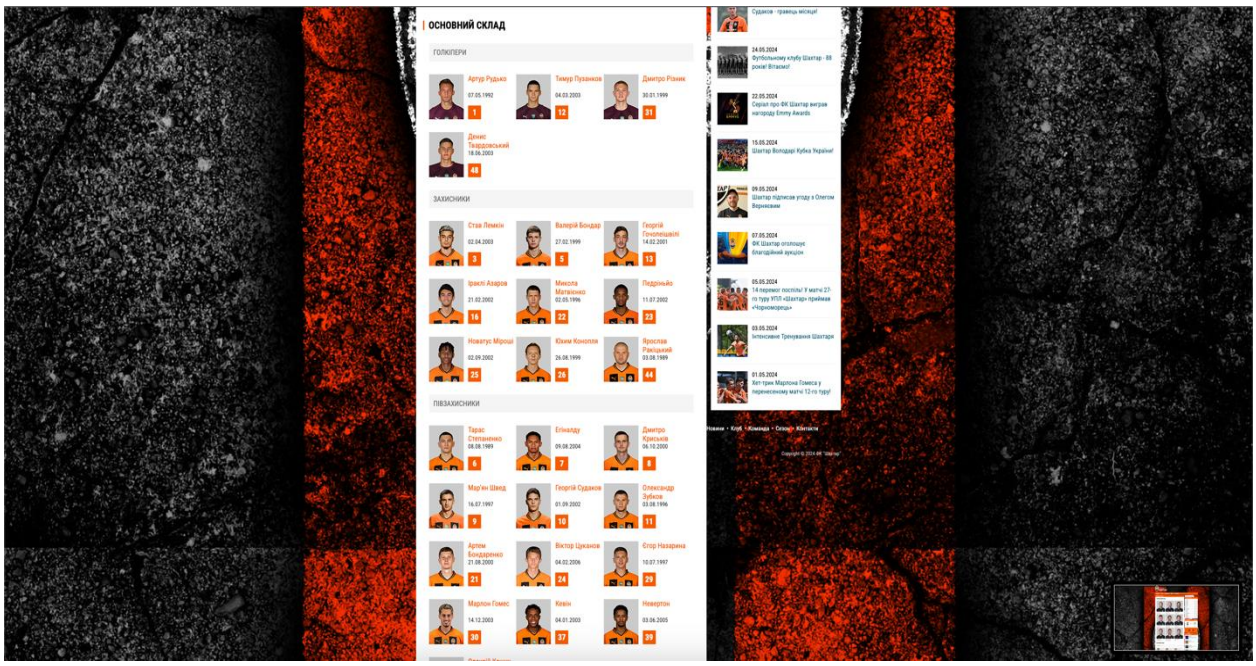

Рисунок 3.24 – Основний склад гравців ФК “Шахтар”

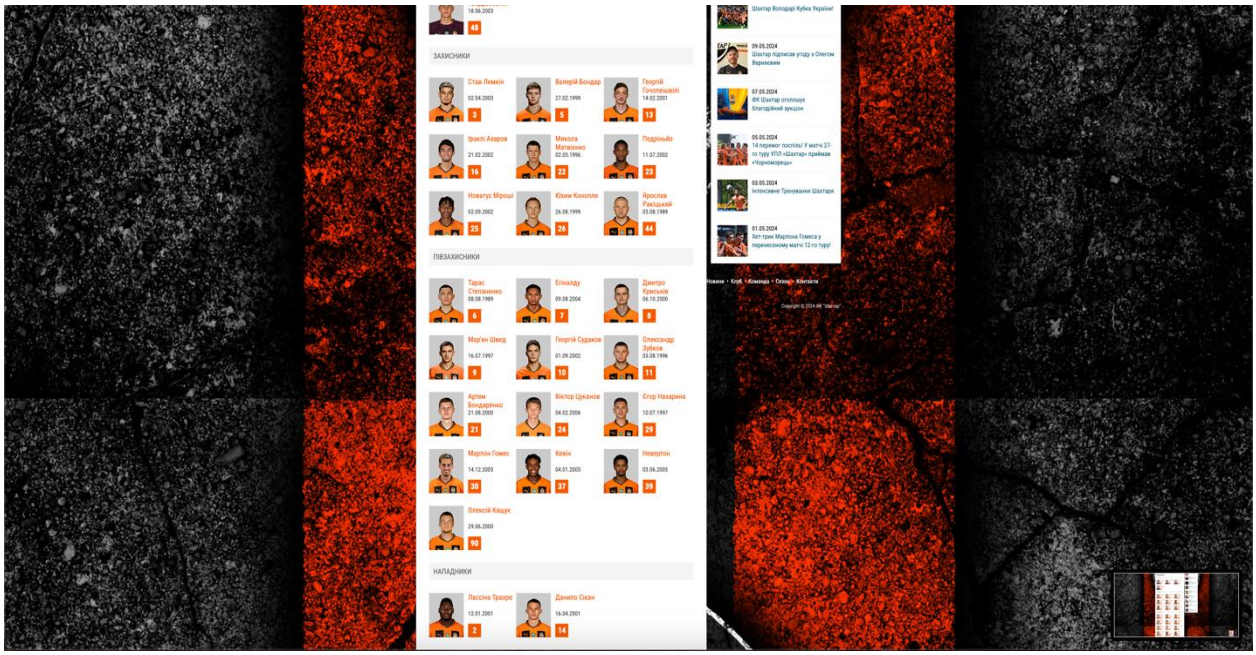

Рисунок 3.25 – Основний склад гравців ФК “Шахтар”

Далі буде наведено приклад реалізації даної сторінки та карток тренерів та гравців (рис. 3.26 – 3.30).

```
<div class="wrapper">
    <div class="content">
        <div class="page-content team">
            <h1>Команда ФК Шахтар</h1>
            <div class="team-wrapper">
                <h2>Тренерський склад</h2>
            </div>
            <div class="coach-wrapper">
                <div class="coach-item">
                    <a
                        ><img
                            src="../assets/img/pusic.jpg"
                            alt=""
                        /></a>
                    <a class="name-coach"
                        >Маріно Пушич</a
                    >
                    <div class="position-coach">
                        <span>Головний тренер </span>
                    </div>
                </div>
                <div class="coach-item">
                    <a
                        ><img
                            src="../assets/img/stanic.jpg"
                            alt=""
                        /></a>
                    <a class="name-coach"
                        >Маріо Станич</a
                    >
                    <div class="position-coach">
                        <span>Асистент тренера</span>
                    </div>
                </div>
                <div class="coach-item">
                    <a
                        ><img
                            src="../assets/img/weerden.jpg"
                            alt=""
                        /></a>
                    <a class="name-coach"
                        >Кріс Ван Дер Верден</a
                    >
                    <div class="position-coach">
                        <span>Асистент тренера</span>
                    </div>
```

Рисунок 3.26 – Приклад html реалізації сторінки “Команда”

```
        </div>
        <div class="team-wrapper">
            <h2>Основний склад</h2>
        </div>
        <div id="main-team" class="coach-wrapper">
            <h3>Голкілери</h3>
            <div class="player-item">
                <div class="player-photo">
                    <a
                        ><img
                            src="../assets/img/rudko.jpg"
                            alt=""
                        /></a>
                </div>
                <div class="player-text">
                    <div class="player-name">
                        <a>Артур Рудько </a>
                    </div>
                    <div class="player-birthday">07.05.1992</div>
                    <div class="player-number">1</div>
                </div>
            </div>
            <div class="player-item">
                <div class="player-photo">
                    <a
                        ><img
                            src="../assets/img/puzankov.jpg"
                            alt=""
                        /></a>
                </div>
                <div class="player-text">
                    <div class="player-name">
                        <a>Тимур Пузанков </a>
                    </div>
                    <div class="player-birthday">04.03.2003</div>
                    <div class="player-number">12</div>
                </div>
            </div>
            <div class="player-item">
            <div class="player-photo">
                <a
                    ><img
                        src="../assets/img/riznuk.jpg"
                        alt=""
                    /></a>
                </div>
```

Рисунок 3.27 – Приклад html реалізації сторінки “Команда”

```
<h3>Півзахисники</h3>
<div class="player-item">
    <div class="player-photo">
        <a
            ><img
                src="../assets/img/stepanenko.jpg"
                alt=""
            /></a>
    </div>
    <div class="player-text">
        <div class="player-name">
            <a>Тарас Степаненко </a>
        </div>
        <div class="player-birthday">08.08.1989</div>
        <div class="player-number">6</div>
    </div>
</div>
<div class="player-item">
    <div class="player-photo">
        <a
            ><img
                src="../assets/img/eginaldo.jpg"
                alt=""
            /></a>
    </div>
    <div class="player-text">
        <div class="player-name">
            <a>Егіналду </a>
        </div>
        <div class="player-birthday">09.08.2004</div>
        <div class="player-number">7</div>
    </div>
</div>
<div class="player-item">
    <div class="player-photo">
        <a
            ><img
                src="../assets/img/kruskiv.jpg"
                alt=""
            /></a>
    </div>
    <div class="player-text">
        <div class="player-name">
            <a>Дмитро Криськів </a>
        </div>
```

Рисунок 3.28 – Приклад html реалізації сторінки “Команда”

```
.coach-item {
    width: 195px;
    float: left;
    overflow: hidden;
    padding-bottom: 10px;
    padding-right: 12px;
}
.coach-item:nth-child(3n) {
    padding-right: 0px;
}
.team {
    overflow: hidden;
}
.coach-wrapper {
    max-width: 610px;
    margin: 0 auto;
}
.name-coach {
    font-size: 16px;
    font-family: 'Roboto Condensed', sans-serif;
    color: #f75b00;
    text-decoration: none;
}
.position-coach {
    font-size: 13px;
    font-family: 'Roboto Condensed', sans-serif;
    color: #333;
    padding-top: 6px;
}
.coach-full {
    padding: 30px 15px;
    overflow: hidden;
}
.coach-full img {
    float: left;
    padding: 0 20px 33px 0;
}
.coach-info {
    border-top: 1px solid #f0f0f0;
    border-bottom: 1px solid #f0f0f0;
    float: right;
```

Рисунок 3.29 – Приклад css реалізації сторінки “Команда”

```css
.player-name a {
    color: #f75b00;
    font-size: 16px;
    font-family: 'Roboto Condensed', sans-serif;
    text-decoration: none;
}
.player-birthday {
    color: #333333;
    font-family: 'Roboto Condensed', sans-serif;
    font-size: 13px;
}
.player-number {
    font-family: 'Roboto Condensed', sans-serif;
    font-size: 18px;
    font-weight: 700;
    background: #f75b00;
    display: inline-block;
    height: 30px;
    width: 30px;
    color: #ffffff;
    text-align: center;
    line-height: 30px;
    margin-top: 15px;
}
.player-item {
    width: 33%;
    float: left;
    overflow: hidden;
    padding-bottom: 28px;
}
.player-photo {
    width: 80px;
    float: left;
    padding-right: 10px;
}
.player-photo img {
    vertical-align: top;
}
.player-name {
    max-height: 38px;
    min-height: 38px;
    overflow: hidden;
}
```

Рисунок 3.30 – Приклад css реалізації сторінки “Команда”

Наступною маємо сторінку “Сезон” на якій відображена детальна турнірна таблиця зі статистиками забитих та пропущених голів кожної команди, кількістю виграних та програних матчів, різницею забитих та пропущених м’ячів та кількістю очок, а нижче відображений календар ігор, на якому всі матчі футбольного клубу “Шахтар” та їх результати (рис. 3.31 – 3.32).

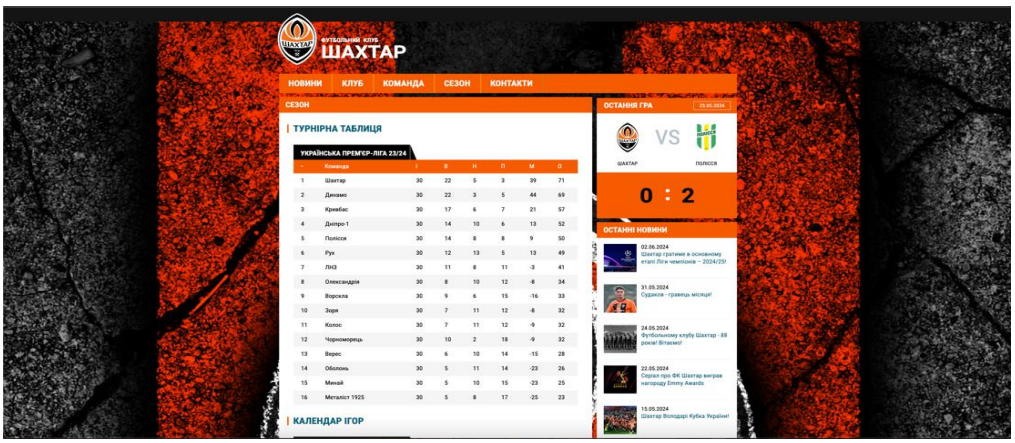

Рисунок 3.31 – Турнірна таблиця на сторінці “Сезон”

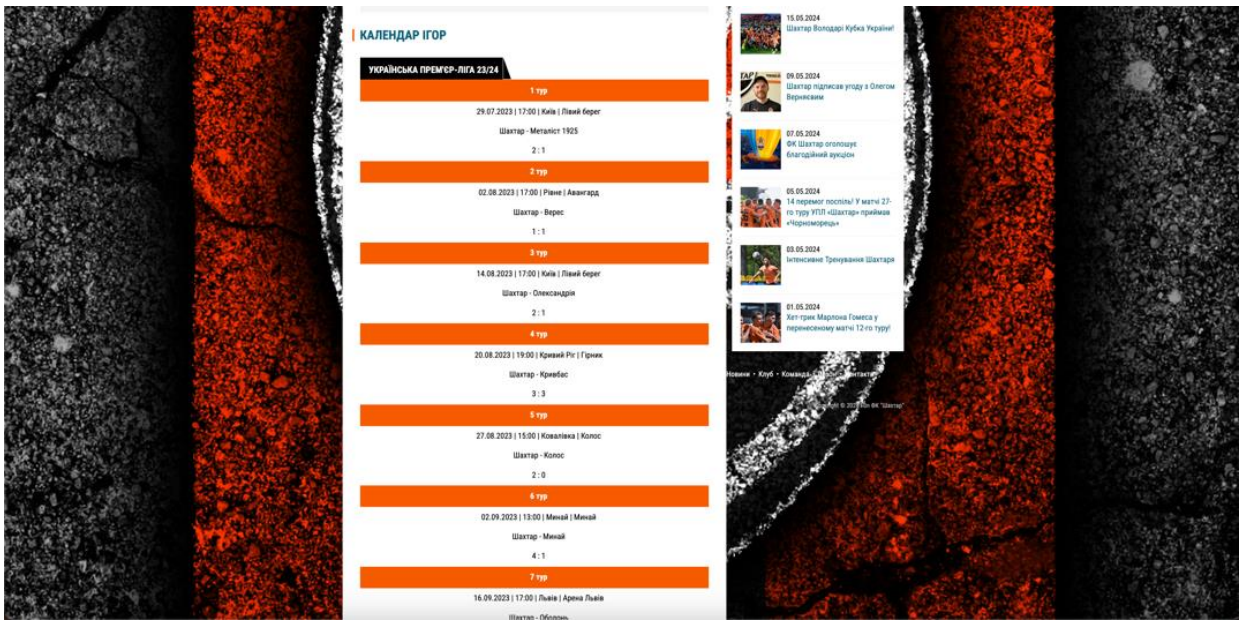

Рисунок 3.32 – Календар ігор на сторінці “Сезон”

Навожу приклади реалізації данної сторінки та її блоків (рис. 3.33 – 3.38).

```
<div class="page-content team">
    <h1>Сезон</h1>
    <div class="table-wrapper">
        <h2>Турнірна таблиця</h2>
        <div class="table-wrapper-item">
            <div class="championships table-main">
                <div class="tab championship-active" data-id="17">
                    Українська Прем'єр-Ліга 23/24
                </div>

                <table id="t17" class="show-table">
                    <thead>
                        <tr>
                            <th>•</th>
                            <th>Команда</th>
                            <th>І</th>
                            <th>В</th>
                            <th>Н</th>
                            <th>П</th>
                            <th>М</th>
                            <th>О</th>
                        </tr>
                    </thead>
                    <tbody>
                        <tr>
                            <td>1</td>
                            <td>Шахтар</td>
                            <td>30</td>
                            <td>22</td>
                            <td>5</td>
                            <td>3</td>
                            <td>39</td>
                            <td>71</td>
                        </tr>
                        <tr>
                            <td>2</td>
                            <td>Динамо</td>
                            <td>30</td>
                            <td>22</td>
                            <td>3</td>
                            <td>5</td>
                            <td>44</td>
                            <td>69</td>
                        </tr>
                        <tr>
                            <td>3</td>
```

Рисунок 3.33 – Приклад html реалізації турнірної таблиці

```
                        </tr>
                        <tr>
                            <td>13</td>
                            <td>Верес</td>
                            <td>30</td>
                            <td>6</td>
                            <td>10</td>
                            <td>14</td>
                            <td>−15</td>
                            <td>28</td>
                        </tr>
                        <tr>
                            <td>14</td>
                            <td>Оболонь</td>
                            <td>30</td>
                            <td>5</td>
                            <td>11</td>
                            <td>14</td>
                            <td>−23</td>
                            <td>26</td>
                        </tr>
                        <tr>
                            <td>15</td>
                            <td>Минай</td>
                            <td>30</td>
                            <td>5</td>
                            <td>10</td>
                            <td>15</td>
                            <td>−23</td>
                            <td>25</td>
                        </tr>
                        <tr>
                            <td>16</td>
                            <td>Металіст 1925</td>
                            <td>30</td>
                            <td>5</td>
                            <td>8</td>
                            <td>17</td>
                            <td>−25</td>
                            <td>23</td>
                        </tr>
                    </tbody>
                </table>
            </div>
        </div>
```

Рисунок 3.34 – Приклад html реалізації турнірної таблиці

```
<table id="tc17" class="show-table">
    <caption></caption>
    <tbody>
        <tr class="tour-line">
            <td>1 тур</td>
        </tr>
        <tr>
            <td>
                <a>
                    29.07.2023 | 17:00 | Київ | Лівий берег
                    <br />
                    Шахтар – Металіст 1925 <br />2 : 1
                </a>
            </td>
        </tr>
        <tr class="tour-line">
            <td>2 тур</td>
        </tr>
        <tr>
            <td>
                <a>
                    02.08.2023 | 17:00 | Рівне | Авангард <br />
                    Шахтар – Верес <br />1 : 1
                </a>
            </td>
        </tr>
        <tr class="tour-line">
            <td>3 тур</td>
        </tr>
        <tr>
            <td>
                <a>
                    14.08.2023 | 17:00 | Київ | Лівий берег
                    <br />
                    Шахтар – Олександрія <br />2 : 1
                </a>
            </td>
        </tr>
        <tr class="tour-line">
            <td>4 тур</td>
        </tr>
        <tr>
            <td>
                <a>
                    20.08.2023 | 19:00 | Кривий Ріг | Гірник <br />
```

Рисунок 3.35 – Приклад html реалізації календаря ігор

```
<tr class="tour-line">
    <td>27 тур</td>
</tr>
<tr>
    <td>
        <a>
            05.05.2024 | 18:00 | Київ | Лівий берег
            <br />
            Шахтар – Чорноморець <br />3 : 0
        </a>
    </td>
</tr>
<tr class="tour-line">
    <td>28 тур</td>
</tr>
<tr>
    <td>
        <a>
            11.05.2024 | 18:00 | Львів | Арена Львів
            <br />
            Шахтар – Динамо <br />1 : 0
        </a>
    </td>
</tr>
<tr class="tour-line">
    <td>29 тур</td>
</tr>
<tr>
    <td>
        <a>
            19.05.2024 | 15:30 | Дніпро | Дніпро Арена<br />
            Шахтар – Дніпро-1 <br />1 : 1
        </a>
    </td>
</tr>
<tr class="tour-line">
    <td>30 тур</td>
</tr>
<tr>
    <td>
        <a>
            25.05.2024 | 15:30 | Житомир | Центральний стадіон
            <br />
            Шахтар – Полісся <br />0 : 2
        </a>
```

Рисунок 3.36 – Приклад html реалізації календаря ігор

```
.calendar-match tr:nth-child(odd) {
    background: transparent;
}
.calendar-match tr:nth-child(odd) {
    background: transparent;
}
.calendar-match tr:nth-child(even) {
    background: transparent;
}

.calendar-match table tr.tour-line {
    background: #f75b00;
    font-weight: 700;
}
.calendar-match tr td {
    border-bottom: 1px solid #f75b00;
    text-align: center;
    color: white;
}
.calendar-match caption {
    background: #f75b00;
    height: 2px;
}
.calendar-match {
    padding-bottom: 20px;
}
.calendar-match a {
    color: black
}
```

Рисунок 3.37 – Приклад css реалізації календаря ігор

```css
.championships caption {
    text-align: left;
    position: relative;
    z-index: 38;
}
.tab,
.ctab {
    background: #f0f0f0;
    padding: 14px 8px 9px 15px;
    color: #b0b0b0;
    text-transform: uppercase;
    font-size: 13px;
    font-weight: 700;
    font-family: 'Roboto Condensed', sans-serif;
    position: relative;
    margin-right: 7px;
    cursor: pointer;
}
.championship-active {
    background: #ffffff;
    color: #046077;
}
.tab:before,
.ctab:before {
    content: '';
    position: absolute;
    top: 0;
    right: -14px;
    width: 0;
    height: 0;
    border-left: 0px solid transparent;
    border-right: 14px solid transparent;
    border-bottom: 38px solid #f0f0f0;
    z-index: 10;
}
```

Рисунок 3.38 – Приклад css реалізації турнірної таблиці

Адаптація сайту грає важливу роль, адже багато користувачів використовують лише телефон для слідкування за новинами, тому і треба зробити так щоб всім було зручно переглядати сайт з будь-якого пристрою (рис. 3.39 – 3.40).

Рисунок 3.39 – Приклад адаптивності сайту

Рисунок 3.40 – Приклад адаптивності сайту

Третій розділ містить опис процесу створення сайту та всіх окремих сторінок які зображені на картинках. Порівняння методів та середовище розробки з аналогами.

ВИСНОВКИ

Під час виконання дипломної роботи виконано наступні завдання:

– проаналізовано інформаційні сайти футбольних клубів;
– проаналізовано сучасні технології у створенні вебсайтів та досліджено засоби розробки;
– розроблено проєкт вебсайту футбольного клубу;
– розроблено інформаційний фан вебсайт футбольного клубу ФК “Шахтар” “FunFcShakhtar”.

Основні розділи розробленого вебсайту: Головна сторінка, Новини, Клуб, Команда, Сезон та Контакти.

Розроблено дизайн вебсайту згідно з основними принципами макет-дизайну. Увага була зусереджена на розробці легкого та приємного для використання інформаційного вебсайту. Створений програмний продукт задовольняє всім вимогам, що були зазначені на етапі постановки завдання.

Отже, в ході виконання кваліфікаційної роботи було продемонстровано практичні навички створення та розробки вебсайтів.

ПЕРЕЛІК ПОСИЛАНЬ

1. HTML. URL: https://css.in.ua/article/shcho-take-html_10 (дата звернення: 11.04.2024).
2. ТЗ. URL: https://uk.infoaboutlawyers.com/4006406-technical-task-what-is-it-the-concept-and-content-how-to-make-tk (дата звернення: 11.04.2024).
3. Лисенко Д. HTML5 і CSS3: сучасні веб-технології. Київ : Видавництво «Академія», 2015. 300 с.
4. Коваленко І. Веб-розробка для початківців: основи HTML, CSS та JavaScript. Львів : Видавництво «Бібліотека», 2017. 250 с.
5. Шевченко В. Сучасний веб-дизайн: теорія та практика. Харків : Видавництво «Освіта», 2019. 320 с.
6. Кравченко М. Професійний підхід до веб-розробки. Одеса : Видавництво «Наука», 2018. 280 с.
7. Гончар Ю. Створення адаптивних веб-сторінок. Дніпро : Видавництво «Техніка», 2020. 210 с.
8. Дакетт Дж. JavaScript та JQuery: інтерактивна розробка веб-сторінок / перекл. з англ. Київ : Видавництво «Книголюб», 2016. 400 с.
9. CSS. URL: https://css.in.ua/ (дата звернення: 11.04.2024).
10. МакДональд М. HTML, CSS та JavaScript. Вивчаємо разом. Київ : Видавництво «Навчальна книга», 2021. 350 с.
11. Вебсайт аналог. URL: http://www.fcab.com.ua/ (дата звернення: 11.04.2024).
12. Вебсайт аналог. URL: https://dynamo.kiev.ua/uk/ (дата звернення: 11.04.2024).