

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

ПОЯСНЮВАЛЬНА ЗАПИСКА

до кваліфікаційної роботи освітнього ступеня «бакалавр»
за спеціальністю 121 «Інженерія програмного забезпечення»
(освітня програма «Веб-технології»)

на тему:

«Вебпортал футбольного клубу з інтегрованим фаншопом»

Виконав студент групи ВТ-22-1
ЦИВІНСЬКИЙ Мар'ян Володимирович

Керівник роботи:
НЕРОДА Сергій Іванович

Рецензент:
ЛОКТИКОВА Тамара Миколаївна

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

«ЗАТВЕРДЖУЮ»

Зав. кафедри інженерії
програмного забезпечення
Тетяна Вакалюк

«28» лютого 2026 р.

ЗАВДАННЯ
на кваліфікаційну роботу

Здобувач вищої освіти: **ЦИВІНСЬКИЙ Мар'ян Володимирович**

Керівник роботи: **НЕРОДА Сергій Іванович**

Тема роботи: **«Вебпортал футбольного клубу з інтегрованим фаншопом»**,
затверджена Наказом закладу вищої освіти від **«27» лютого 2026 р., №92/с**

Вихідні дані для роботи:

- об'єкт дослідження: процеси розробки та розгортання вебпорталів для спортивних організацій з інтегрованим модулем електронної комерції.
- предмет дослідження: методи та технології повностекової розробки вебпорталу футбольного клубу з інтегрованим фаншопом на основі фреймворку Next.js з використанням TypeScript та Prisma ORM.

Консультанти з бакалаврської кваліфікаційної роботи із зазначенням розділів, що їх стосуються:

Розділ	Консультант	Завдання видав	Завдання прийняв
1	Нерода С.І.	01.03.2026р.	01.03.2026р.
2	Нерода С.І.	01.03.2026р.	01.03.2026р.
3	Нерода С.І.	01.03.2026р.	01.03.2026р.

РЕФЕРАТ

Пояснювальна записка до випускної кваліфікаційної роботи бакалавра містить 115 сторінок, 58 ілюстрацій, 6 таблиць, 30 джерел літератури та 7 додатків.

Метою роботи є проектування та розробка повнофункціонального вебпорталу футбольного клубу з інтегрованим фаншопом, що включає клієнтську частину, серверну логіку та базу даних, для забезпечення ефективного інформування спільноти та реалізації комерційного потенціалу клубу через онлайн-продажі.

У роботі застосовано методи системного аналізу предметної області, об'єктно-орієнтованого проектування, реляційного моделювання даних та інтеграційного тестування. За результатами роботи розроблено повностековий застосунок на основі Next.js 16 з п'ятьма функціональними модулями: контент та медіа, команда, матч-центр, електронна комерція та користувачі. Реалізовано реляційну базу даних PostgreSQL через Prisma ORM, сім алгоритмів роботи системи, інтеграцію з платіжним шлюзом Stripe, сервісом Sofascore через RapidAPI та хмарним сховищем UploadThing. Аудит Google Lighthouse підтвердив оцінку продуктивності 98/100 у режимі Комп'ютер.

Розроблений портал рекомендується до впровадження у діяльність футбольних клубів як комплексна вебплатформа для інформування вболівальників та здійснення онлайн-продажів без залежності від мультидоменної архітектури.

КЛЮЧОВІ СЛОВА: ВЕБПОРТАЛ, ЕЛЕКТРОННА КОМЕРЦІЯ, NEXT.JS, POSTGRESQL, TYPESCRIPT.

					ІПЗ.КР.Б – 121 – 26 – ПЗ			
Змн.	Арк.	№ докум.	Підпис	Дата	Вебпортал футбольного клубу з інтегрованим фаншопом	Літ.	Арк.	Аркушів
Розроб.		Цивінський М.В.						
Керівник		Нерода С.І.					3	115
Рецензент		Локтікова Т.М.				ФІКТ Гр. ВТ-22-1		
Зав. каф.		Вакалюк Т.А.						

ABSTRACT

The explanatory note to the bachelor's thesis contains 115 pages, 58 illustrations, 6 tables, 30 references, and 7 appendices.

The purpose of the work is the design and development of a fully functional football club web portal with an integrated fan shop, which includes the client side, server logic, and a database, to ensure effective informing of the community and the realization of the club's commercial potential through online sales.

The study applies methods of system analysis of the subject area, object-oriented design, relational data modeling, and integration testing. As a result of the work, a full-stack application based on Next.js 16 with five functional modules was developed: content and media, team, match center, e-commerce, and users. A PostgreSQL relational database was implemented using Prisma ORM, along with seven system operation algorithms, integration with the Stripe payment gateway, the Sofascore service via RapidAPI, and UploadThing cloud storage. The Google Lighthouse audit confirmed a performance score of 98/100 in Desktop mode.

The developed portal is recommended for implementation in the activities of football clubs as a comprehensive web platform for informing fans and conducting online sales without dependence on a multi-domain architecture.

KEYWORDS: WEB PORTAL, E-COMMERCE, NEXT.JS, POSTGRESQL, TYPESCRIPT.

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		4

ЗМІСТ

РЕФЕРАТ	3
ABSTRACT	4
ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ	6
ВСТУП	7
РОЗДІЛ 1. АНАЛІЗ ТЕХНОЛОГІЧНИХ РІШЕНЬ ТА МЕТОДІВ ПОБУДОВИ ІНФОРМАЦІЙНО-КОМЕРЦІЙНИХ ВЕБПОРТАЛІВ ФУТБОЛЬНИХ КЛУБІВ	10
1.1 Постановка задачі	10
1.2 Аналіз аналогів програмного продукту	10
1.3 Вибір архітектури вебпорталу футбольного клубу з інтегрованим фаншопом	17
1.4 Обґрунтування вибору інструментальних засобів та вимоги до апаратного забезпечення	17
Висновки до першого розділу	20
РОЗДІЛ 2. ПРОЄКТУВАННЯ ТА РЕАЛІЗАЦІЯ ВЕБПОРТАЛУ ФУТБОЛЬНОГО КЛУБУ З ІНТЕГРОВАНІМ ФАНШОПОМ	25
2.1 Визначення варіантів використання та об'єктно-орієнтованої структури системи	25
2.2 Розробка бази даних системи	29
2.3 Проєктування та реалізація алгоритмів роботи системи	32
2.4 Реалізація функціоналу системи вебпорталу футбольного клубу з інтегрованим фаншопом	36
Висновки до другого розділу	39
РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З ВЕБПОРТАЛОМ ФУТБОЛЬНОГО КЛУБУ З ІНТЕГРОВАНІМ ФАНШОПОМ	42
3.1 Порядок встановлення та налаштування параметрів платформи	42
3.2 Структура інтерфейсу вебпорталу футбольного клубу з інтегрованим фаншопом	47
3.3 Тестування вебпорталу футбольного клубу з інтегрованим фаншопом	68
Висновки до третього розділу	75
ВИСНОВКИ	77
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	79
ДОДАТКИ	82

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

API – прикладний програмний інтерфейс

AWS – хмарна платформа Amazon (Amazon Web Services)

CDN – мережа доставки контенту (Content Delivery Network)

CI/CD – безперервна інтеграція та розгортання (Continuous Integration / Continuous Deployment)

CSS – каскадні таблиці стилів (Cascading Style Sheets)

CUID – стійкий до колізій унікальний ідентифікатор (Collision-resistant Unique Identifier)

CTA – заклик до дії (Call to Action)

HTML – мова розмітки гіпертексту (HyperText Markup Language)

HTTPS – захищений протокол передачі гіпертексту (HyperText Transfer Protocol Secure)

JSON – нотація об'єктів JavaScript (JavaScript Object Notation)

JWT – JSON веб-токен (JSON Web Token)

OAuth – відкрита авторизація (Open Authorization)

ORM – об'єктно-реляційне відображення (Object-Relational Mapping)

REST – передача репрезентативного стану (Representational State Transfer)

SEO – пошукова оптимізація (Search Engine Optimization)

SQL – мова структурованих запитів (Structured Query Language)

СУБД – система управління базами даних

UI – користувацький інтерфейс (User Interface)

URL – уніфікований локатор ресурсів (Uniform Resource Locator)

XSS – міжсайтовий скриптинг (Cross-Site Scripting)

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		6

ВСТУП

Сьогодні професійний футбольний клуб – це не лише спортивна організація, а й повноцінний медіабренд, успіх якого безпосередньо залежить від якості цифрової взаємодії з аудиторією. Споживання спортивного контенту перемістилося у площину вебресурсів, де користувач очікує отримати все в одному місці: від оперативних новин та детальної статистики до можливості придбання офіційної атрибутики. Сучасні фреймворки на кшталт Next.js 16 дозволяють реалізовувати повностекові рішення в межах єдиної кодової бази, поєднуючи серверний рендеринг із високою інтерактивністю модулів електронної комерції, а хмарні платформи спрощують інтеграцію складних бізнес-функцій без підтримки власної інфраструктури.

Проте більшість існуючих рішень є розрізненими: інформаційний сайт функціонує окремо від комерційного магазину, що ускладнює адміністрування та погіршує користувацький досвід. Порівняльний аналіз вебпорталів провідних клубів підтверджує, що навіть організації світового рівня мають спільну архітектурну проблему: фаншоп у більшості випадків винесено на окремий піддомен, що розриває цілісність сесії покупця. Актуальність теми підтверджується активним розвитком технологій повностекової веброзробки, дедалі більшим попитом на цифровізацію спортивних організацій та відсутністю готових рішень, що поєднують медіаплатформу й електронну комерцію без мультидоменної архітектури.

Метою випускної кваліфікаційної роботи бакалавра є проєктування та розробка повнофункціонального вебпорталу футбольного клубу, що включає клієнтську частину, серверну логіку та базу даних, для забезпечення ефективного інформування спільноти та реалізації комерційного потенціалу клубу через онлайн-продажі. Реалізація проєкту спрямована на розв'язання проблеми фрагментації фанатського досвіду шляхом інтеграції інформаційно-медійної та комерційної складових у єдиний інтерфейс із централізованою базою даних.

Для досягнення поставленої мети необхідно вирішити комплекс взаємопов'язаних завдань. Насамперед потрібно проаналізувати предметну

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		7

область та існуючі рішення шляхом порівняльного огляду вебпорталів провідних футбольних клубів, що дозволить виявити типові недоліки та сформулювати чіткі функціональні вимоги до створюваної системи. На основі цих вимог необхідно обрати архітектуру та інструментальний стек, обґрунтувавши вибір технологій з урахуванням потреб у високій продуктивності, типобезпеці, підтримці мультимовності та зручності адміністрування.

Наступним кроком є проєктування об'єктно-орієнтованої моделі системи, що передбачає визначення акторів, варіантів використання, а також сутностей бази даних із їхніми атрибутами та зв'язками. Відповідно до створеної моделі має бути розроблена реляційна база даних PostgreSQL, що охоплюватиме п'ять предметних областей системи та використовуватиме архітектурні патерни мультимовності й м'якого видалення.

Практична реалізація проєкту вимагає розробки ключових алгоритмів роботи порталу. До них належать логіка автентифікації, оформлення замовлення, обробки платежів через вебхук Stripe, формування гібридної статистики гравців, а також три окремі алгоритми автоматичної синхронізації спортивних даних із сервісом Sofascore для матчів, їхніх деталей та турнірних таблиць. Ці алгоритми стануть фундаментом для реалізації повностекового застосунку, який об'єднає клієнтську та серверну частини з п'ятьма функціональними модулями, захищеною адміністративною панеллю та повноцінною інтеграцією зовнішніх сервісів.

Завершальними завданнями є розгортання готового застосунку та налаштування процесів безперервної інтеграції і доставки (CI/CD) шляхом організації пайплайну через GitLab із дзеркалюванням коду на GitHub та деплоєм на платформу Vercel. Після розгортання необхідно провести комплексне тестування системи: перевірити роботу ключових підсистем та процесів, обробити граничні випадки алгоритмів, оцінити загальну продуктивність за метриками Google Lighthouse та впевнитися у сумісності платформи з основними браузерними рушіями.

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		8

Об'єктом дослідження є процеси розробки та розгортання вебпорталів для спортивних організацій з інтегрованим модулем електронної комерції.

Предметом дослідження є методи та технології повностекової розробки вебпорталу футбольного клубу з інтегрованим фаншопом на основі фреймворку Next.js 16 з використанням TypeScript та Prisma ORM.

Методологічною основою роботи є комплексний підхід, у межах якого застосовано системний аналіз для дослідження предметної області та порівняльного огляду існуючих вебпорталів. Побудова моделі варіантів використання та діаграм класів здійснювалася за допомогою об'єктно-орієнтованого проєктування. Для проєктування схеми бази даних із нормалізацією та індексуванням використано реляційне моделювання даних, тоді як методи транзакційного програмування дозволили забезпечити цілісність даних при обробці замовлень. Перевірка роботи системи в реальних умовах production-середовища виконувалася шляхом інтеграційного та end-to-end тестування.

Практичне значення роботи полягає у створенні готового програмного комплексу, що дозволяє централізовано керувати інформаційними потоками клубу та здійснювати онлайн-продажі в межах цілісного інформаційного простору. Впровадження платформи дає клубу змогу відмовитися від підтримки розрізнених систем, забезпечити наскрізний кошик покупця та підвищити конверсію фаншопу завдяки контекстним продажам через профілі гравців. Автоматична синхронізація спортивних даних через Vercel Cron знімає рутинне навантаження з адміністраторів, а CI/CD пайплайн забезпечує надійне розгортання оновлень.

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		9

РОЗДІЛ 1. АНАЛІЗ ТЕХНОЛОГІЧНИХ РІШЕНЬ ТА МЕТОДІВ ПОБУДОВИ ІНФОРМАЦІЙНО-КОМЕРЦІЙНИХ ВЕБПОРТАЛІВ ФУТБОЛЬНИХ КЛУБІВ

1.1 Постановка задачі

Вебпортал футбольного клубу – це інформаційна система, яка об'єднує медіаплатформу, спортивну статистику та електронну комерцію в комплексну вебплатформу. Платформи такого рівня відіграють ключову роль у формуванні бренду спортивної організації, забезпечуючи пряму комунікацію з вболівальниками, популяризацію матчів та монетизацію через продаж клубної атрибутики. На відміну від класичних корпоративних сайтів, вебпортал футбольного клубу поєднує в собі риси медіаресурсу, спортивного довідника та інтернет-магазину, що висуває особливі вимоги до архітектури системи: вона повинна однаково ефективно обслуговувати як публічний контент із серверним рендерингом для SEO, так і транзакційні операції з гарантією цілісності даних.

Вебпортал охоплює п'ять функціональних модулів. Модуль контент та медіа дозволяє адміністраторам публікувати матеріали різних типів – новини, інтерв'ю, офіційні заяви – та керувати фотогалереями з прив'язкою до конкретних гравців або матчів; редагування текстового вмісту реалізовано через редактор Tiptap, що генерує структурований HTML з підтримкою форматування та вбудованих медіафайлів [22]. Модуль команди забезпечує ведення даних гравців та тренерів, включаючи біографію, антропометрію та гібридну статистику, що поєднує вручну внесені дані попередніх сезонів із динамічно обчисленою статистикою поточного сезону на основі матчових подій у базі даних – це дозволяє відображати актуальні показники навіть за відсутності повних архівних даних у системі. Модуль матч-центру надає інструменти для відображення календаря ігор, турнірних таблиць та протоколів матчів із фіксацією складу, результативних дій, карток і замін, а турнірні таблиці та деталі матчів синхронізуються із зовнішнім сервісом Sofascore через платформу

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		10

RapidAPI, що звільняє адміністраторів від необхідності вручну вносити статистику після кожного туру. Модуль електронної комерції забезпечує повний цикл онлайн-продажів: управління каталогом товарів із категоріями, розмірною сіткою та системою знижок, обробку кошика та оформлення замовлень з оплатою через платіжний шлюз Stripe. Стан кошика зберігається на клієнті за допомогою Zustand із персистенцією у localStorage. Це дозволяє зберігати вміст між сесіями без потреби в авторизації. Модуль користувачів відповідає за автентифікацію через бібліотеку NextAuth v5 [2] з підтримкою облікових записів та OAuth-провайдерів. Він також забезпечує розмежування ролей адміністратора і звичайного користувача та збереження історії замовлень з можливістю прив'язки гостьових замовлень до акаунту після реєстрації.

Система підтримує дві мови інтерфейсу – українську та англійську – засобами бібліотеки next-intl [1]. Мультимовність реалізована на двох рівнях. Переклади статичного інтерфейсу зберігаються у JSON-файлах. Переклади динамічного контенту – назви гравців, товарів, турнірів, категорій та суперників – зберігаються безпосередньо в базі даних через окремі таблиці перекладів для кожної сутності. Такий підхід забезпечує коректне відображення локалізованого вмісту незалежно від обраної мови без дублювання основних записів.

Архітектура застосунку побудована на базі Next.js 16 App Router. Серверні компоненти відповідають за рендеринг публічного контенту, а Server Actions слугують єдиним шаром мутацій даних, що усуває потребу в окремому REST API. Валідація вхідних даних реалізована через бібліотеку Zod як на клієнті у формах, так і на сервері перед записом у базу даних. Взаємодія з базою даних PostgreSQL організована через Prisma ORM, що забезпечує типобезпечні запити та автоматичне керування міграціями схеми. Зберігання та доставка медіаконтенту реалізовані через хмарний сервіс UploadThing [13], що знімає навантаження з основного сервера. Для проєктування інтерфейсу використано shadcn/ui на базі Tailwind CSS v4, а для табличного відображення даних в адміністративній панелі – TanStack Table [20, 21].

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		11

1.2 Аналіз аналогів програмного продукту

Розробка вебпорталу з інтегрованим фаншопом потребує аналізу існуючих рішень на ринку. У рамках дослідження було проведено порівняльний огляд офіційних вебпорталів провідних футбольних клубів України та Європи.

Одним з провідних українських клубів в Європі зараз є «Шахтар Донецьк» (рис. 1.1).

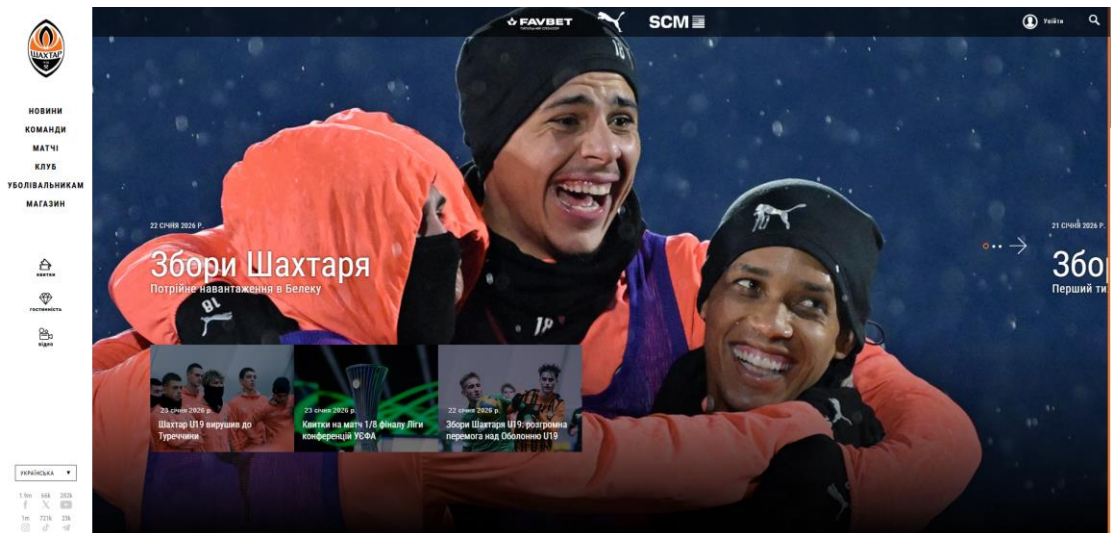


Рис. 1.1 – вебпортал «Шахтар Донецьк»

Серед переваг порталу виділяється реалізована розширена статистика, що включає історію очних зустрічей, стартові склади, текстову трансляцію подій та фотогалерею матчу. Профілі футболістів містять детальну статистику за всі сезони, які гравець провів за «Шахтар», новини та медіаматеріали. Присутній функціонал перехресних продажів, можливість переходу до купівлі ігрової футболки безпосередньо з профілю гравця. Реалізована фільтрація по атрибутіці за конкретним футболістом та інше.

Суттєвим недоліком цієї системи є те, що фаншоп та медіапортал реалізовані на окремих піддоменах або зовнішніх сайтах. Це призводить до розриву користувацького досвіду, оскільки інтерфейс та навігація змінюються при переході між розділами. Через те, що магазин знаходиться на іншому домені, кошик покупця не відображається на основному сайті. Користувач не бачить статус своїх покупок, поки не перейде на сайт магазину. Додавання в кошик

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		12

відбувається з помітною затримкою в секунду, що свідчить про проблеми з обміном даних між різними системами.

Наступним аналогом є житомирський футбольний клуб «Полісся» (рис. 1.2).

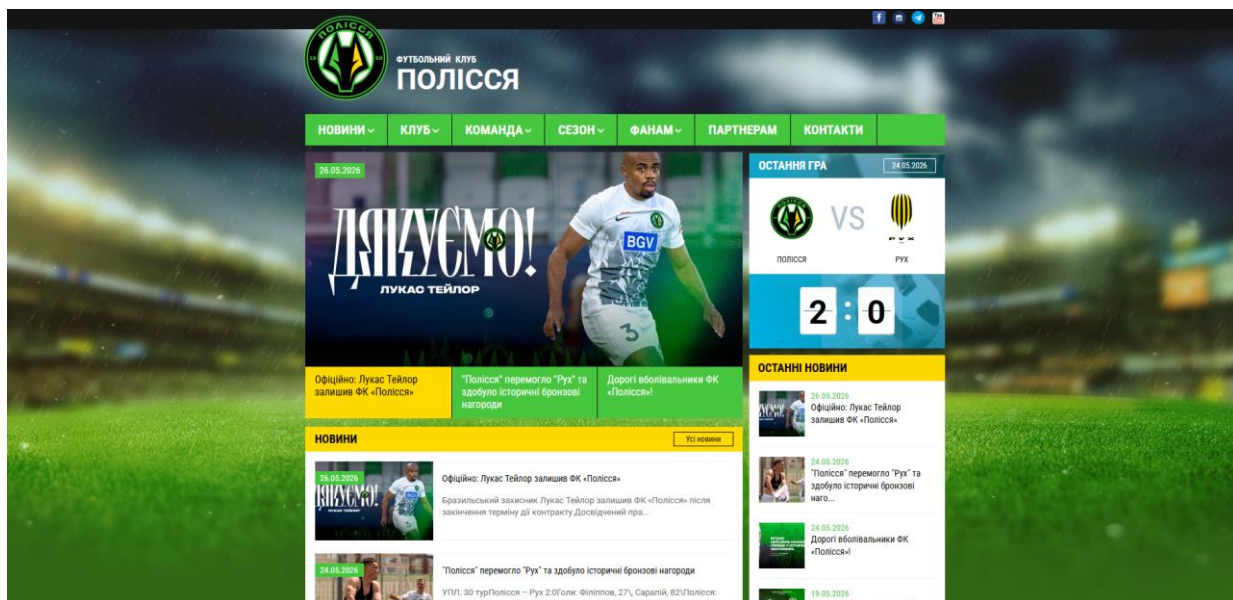


Рис. 1.2 – вебпортал «Полісся»

Сильною стороною вебпорталу є чіткий чіткий розподіл контенту за категоріями команд (Основний склад, u-19, Жіноча команда, СДЮСШОР). Новини наповнені якісним контентом. Присутні окремі розділи для всіх вікових та гендерних категорій команд, а також сторінки тренерського штабу, керівництва та персоналу. Реалізовано доступ до архівних даних за попередні сезони, включаючи результати матчів та інформацію про колишніх гравців клубу всіх категорій.

Водночас система має низку критичних проблем щодо навігації та оптимізації. Як і в попередньому випадку, фаншоп винесено на окремий піддомен, що ламає загальну структуру сайту та робить кошик невидимим на головній сторінці. Крім того, при додаванні товару ресурс примусово перенаправляє користувача до кошика, не даючи змоги плавно продовжити покупки, а зображення екіпірування завантажуються повільно через низьку оптимізацію медіафайлів. Інформативність порталу додатково знижується через відсутність турнірних таблиць для молодіжних та жіночих ліг. Окремою

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		13

архітектурною проблемою є неактуальність контенту: профілі багатьох гравців заповнено лише частково або із застарілою статистикою, трапляються порожні сторінки, а повна відсутність функціоналу глобального пошуку суттєво ускладнює знаходження новин та архівної інформації.

Ще одним аналогом є вебпортал клубу «Реал Мадрид» (рис. 1.3).

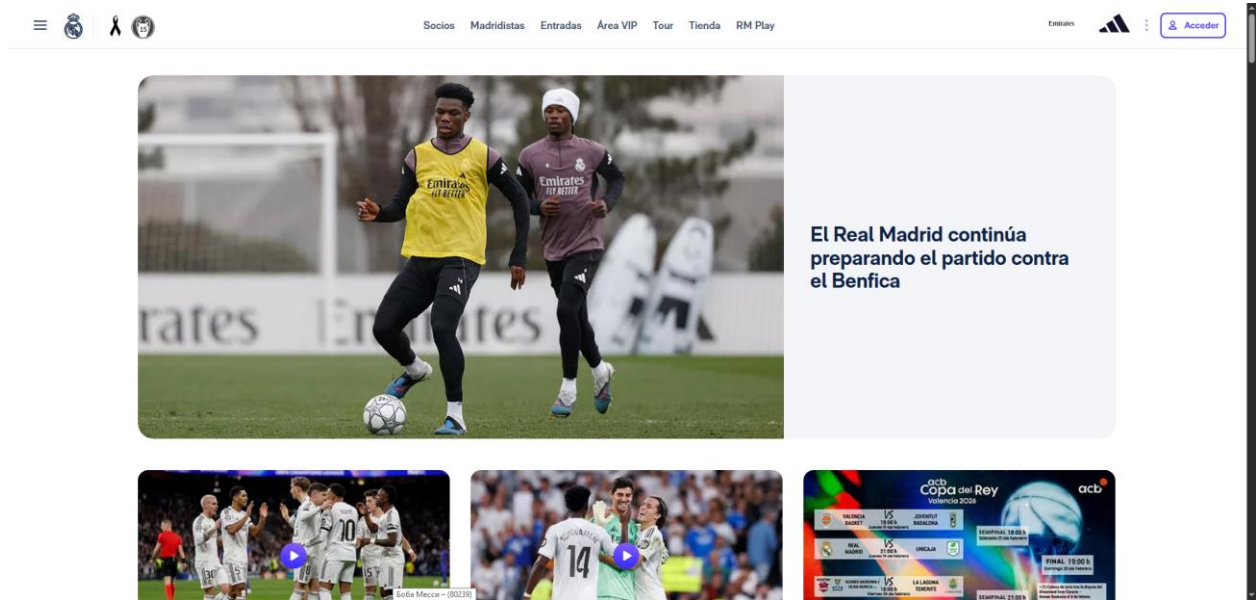


Рис. 1.3 – вебпортал «Реал Мадрид»

Головною перевагою цього ресурсу є глибока інтеграція власної платформи «RM Play», що дозволяє переглянути ефіри та архіви матчів без переходу на зовнішні відеохостинги. Профілі гравців мають детальну статистику, структуровану за амплуа, включно з даними за всі турніри, отримані нагороди та хронологію виступів за попередні клуби до початку дебютного матчу за «Реал Мадрид». Реалізовано механізм залучення користувачів, зокрема можливість залишати реакції на фотографії у галереях матчів. Наявність переходу з профілю гравця на купівлю його іменної футболки. Інтеграція сервісів продажу квитків та турів інфраструктурою клубу. Також представлені детальні склади та турнірні таблиці для всіх вікових категорій, а не тільки для основної команди.

Проте недоліком цього рішення є відсутність індикаторів завантаження при повільному інтернет-з'єднанні, через що користувач не може зрозуміти, чи зависла сторінка, чи виконується запит до сервера. Незважаючи на величезний

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		14

обсяг контенту, на сайті повністю відсутній глобальний пошук, що ускладнює навігацію. Як і в інших аналогах, комерційний модуль винесено на окремий піддомен, і хоча дані користувача успішно зберігаються, кошик стає невидимим при поверненні на головну інформаційну сторінку, порушуючи безперервність сесії покупця.

«Арсенал Лондон» (рис. 1.4) – один із провідних англійських клубів.

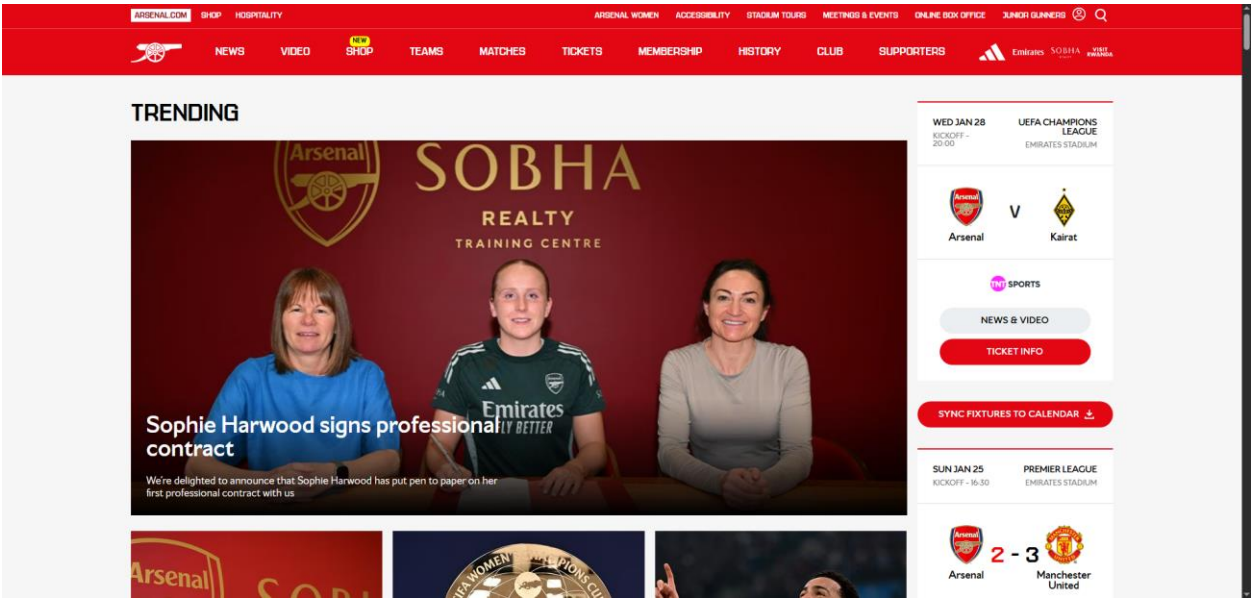


Рис. 1.4 – вебпортал «Арсенал Лондон»

Визначною рисою цієї платформи є потужний пошуковий рушій, який виконує текстовий пошук та категоризує результат, відображаючи точну кількість знайдених статей, відео, профілів гравців та матчів. Профілі гравців оснащені не лише таблицями, а й графічними метриками, що дозволяє аналізувати гру футболіста. Також присутні перехресні посилання на пов’язані новини та медіа. Впроваджено функціонал створення власних плейлістів, користувач може зберігати моменти матчів чи інтерв’ю для перегляду пізніше. Реалізовано інтерактивні вікторини з таймером та підказками, де користувачі перевіряють свої знання про клуб, що значно підвищує час перебування на сайті. Зручна функція синхронізації розкладу матчів, можна обрати також категорію з особистим календарем користувача.

Попри загальну технологічність основного ресурсу, фаншоп винесено на окремий піддомен. При переході з сайту новин до магазину кошик користувача

зникає з навігаційної панелі. Це порушує цілісність сесії та ускладнює швидкі покупки.

Узагальнюючи аналіз існуючих вебпорталів, складемо порівняльну таблицю їхніх функціональних можливостей та розроблюваного вебпорталу (табл. 1.1).

Таблиця 1.1

Порівняння вебпорталів футбольних клубів

Характеристика	Шахтар Донецьк	Полісся	Реал Мадрид	Арсенал	Розроблюваний вебпортал
Сучасний технологічний стек	+-	-	+-	+	+
Інтеграція фаншоп	-	-	-	-	+
Контекстні продажі	+	-	+	+	+
Матч-центр (Деталізація)	+	+-	+	+	+
Керування історією	+	+-	+	+	+
Наскрізний кошик	-	-	-	-	+
Мультимовність	+	-	+	-	+
Підтримка всіх складів	+	+-	+	+	+

Аналіз показав, що більшість сучасних вебпорталів – включно з клубами світового рівня – мають спільну архітектурну проблему: фаншоп і медіаресурс існують як окремі піддомени. Це ламає навігацію, ховає кошик при переході між розділами та ускладнює підтримку двох систем одночасно.

Розроблюваний портал усуває ці проблеми через єдину кодову базу на Next.js 16 з App Router, де фаншоп є повноцінним модулем, а не зовнішнім сайтом. Наскрізний кошик на Zustand доступний із будь-якої сторінки. Гібридна статистика через Prisma ORM автоматично об'єднує архівні дані з подіями поточного сезону. Медіаконтент доставляється через UploadThing без затримок, характерних для аналогів [13].

Таким чином, розроблений портал є актуальним рішенням для футбольного клубу, що прагне створити цифровий простір без залежності від сторонніх CMS або мультидоменної архітектури.

1.3 Вибір архітектури вебпорталу футбольного клубу з інтегрованим фаншопом

Для реалізації вебпорталу футбольного клубу з інтегрованим фаншопом обрано гібридну клієнт-серверну архітектуру на базі фреймворку Next.js 16 [23]. Цей вибір зумовлений необхідністю поєднати серверний рендеринг для медіаконтенту та SEO-оптимізації новин із високою інтерактивністю модулів електронної комерції в межах єдиної кодової бази. Кодова база структурована за допомогою механізму route groups архітектури App Router, що дозволяє чітко розділити застосунок на функціональні зони – публічну частину, фаншоп, автентифікацію та адміністрування – без впливу на маршрутизацію. Такий підхід усуває потребу в окремих репозиторіях або піддоменах для різних частин системи та спрощує спільне використання компонентів, утиліт і типів між усіма зонами застосунку.

Основою платформи слугує Next.js 16 з архітектурою App Router, що дозволяє застосовувати React Server Components для рендерингу контентних сторінок безпосередньо на сервері – HTML надходить до браузера вже з готовим вмістом, що позитивно впливає на швидкість першого відображення та індексацію пошуковими системами. Клієнтська частина реалізована на React 19, а візуальна складова побудована на Tailwind CSS v4 та компонентній бібліотеці

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		17

shadcn/ui, що дозволяє створювати адаптивний інтерфейс для складних форм в модулях користувача та адміністрування без написання окремих CSS-файлів.

Оскільки система керує структурованими даними зі складними зв'язками – між гравцями та їхньою статистикою, матчами та подіями, користувачами та історією замовлень – як сховище даних обрано реляційну СУБД PostgreSQL [15]. Реляційна модель дозволяє ефективно виконувати складні запити з об'єднанням таблиць та гарантує цілісність даних через зовнішні ключі та транзакції. Взаємодія з базою даних здійснюється через ORM Prisma, що забезпечує повну типобезпеку завдяки інтеграції з TypeScript і автоматично генерує клієнт із типами на основі схеми, що унеможливлює невідповідність між структурою бази даних і кодом застосунку [16].

Глобальний клієнтський стан, зокрема вміст кошика у фаншопі, керується бібліотекою Zustand із персистенцією у localStorage – це дозволяє зберігати дані про покупки при переході між сторінками порталу та між сесіями браузера без потреби в авторизації [14]. Наскрізна валідація даних реалізована через бібліотеку Zod: схеми перевіряють коректність даних як на стороні клієнта у формах реєстрації та замовлення, так і на сервері перед записом у базу даних, що гарантує однорідність правил перевірки на обох рівнях і унеможливлює обхід клієнтської валідації [9].

Частина функціоналу делеговано спеціалізованим хмарним сервісам для зниження навантаження на основний сервер. Зберігання та оптимізована доставка медіаконтенту здійснюється через UploadThing, що забезпечує автоматичну обробку завантажених файлів та їх доставку через CDN [13]. Модуль матч-центру інтегровано із зовнішнім сервісом Sofascore через платформу RapidAPI: за допомогою API автоматично синхронізуються матчі поточного сезону з деталями подій, турнірні таблиці, а також дані гравців [17]. Прив'язка сутностей до зовнішнього сервісу здійснюється через ідентифікатори Sofascore, збережені безпосередньо в базі даних, що дозволяє точно зіставляти локальні записи з даними зовнішнього джерела при кожній синхронізації. Обробка фінансових транзакцій у фаншопі реалізована через платіжний шлюз

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		18

Stripe, а автентифікація користувачів – засобами бібліотеки NextAuth v5 з підтримкою протоколу OAuth [2, 19].

Архітектура передбачає розділення системи на чотири логічні зони доступу, що забезпечує чітке розмежування прав та спрощує підтримку middleware-рівня. Усі вхідні запити до застосунку проходять через єдиний файл middleware. Він послідовно виконує дві операції: визначає поточну локаль із першого сегмента URL та перевіряє права доступу до запитуваного маршруту. Такий порядок виконання гарантує, що перенаправлення при несанкціонованому доступі відбуваються з урахуванням активної мови інтерфейсу – користувач потрапляє на локалізовану сторінку входу, а не на сторінку без префікса локалі.

Публічна зона охоплює новини, профілі гравців та тренерів, матч-центр із турнірними таблицями та фаншоп з каталогом товарів. Усі сторінки цієї зони використовують серверний рендеринг для забезпечення актуальності даних та коректної індексації пошуковими системами. Компоненти, що потребують клієнтської інтерактивності – фільтри каталогу, перемикачі вкладок, кошик – позначені директивою "use client" і отримують дані через пропси від серверних батьківських компонентів. Це мінімізує обсяг JavaScript, який надсилається браузеру. Параметри фільтрації та активні вкладки зберігаються в URL через searchParams, завдяки чому стан сторінки відновлюється при поверненні назад і посиланнями можна ділитися з вже застосованими фільтрами.

Зона автентифікації містить сторінки входу та реєстрації з перенаправленням авторизованих користувачів, щоб унеможливити повторний вхід при активній сесії. Приватна зона особистого кабінету відображає персональні дані та історію замовлень користувача і доступна лише після успішної автентифікації. Неавторизовані запити автоматично перенаправляються на сторінку входу на рівні middleware ще до того, як запит досягне серверного компонента сторінки. Це виключає часткове відображення захищеного вмісту. Адміністративна зона захищена системою контролю доступу на основі ролей і надає інтерфейси для управління контентом, командою, матчами та замовленнями магазину. Доступ до неї отримують лише користувачі

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		19

з роллю ADMIN, що перевіряється як на рівні middleware, так і в кожній серверній дії окремо. Це унеможливорює несанкціонований виклик мутацій навіть при прямому зверненні до ендпоінтів в обхід інтерфейсу.

1.4 Обґрунтування вибору інструментальних засобів та вимоги до апаратного забезпечення

Основою платформи слугує Next.js 16 з архітектурою App Router, що поєднує серверні компоненти React 19 для рендерингу контентних сторінок із клієнтськими компонентами для інтерактивних елементів [23, 30]. Server Actions використовуються як основний механізм мутації даних, що усуває потребу в окремому REST API та спрощує організацію бізнес-логіки на стороні сервера. Клієнтську реактивність інтерфейсу забезпечує React 19, вдосконалені механізми конкурентного рендерингу якого дозволяють ефективніше керувати оновленням інтерфейсу при роботі з динамічними даними фаншопу та матч-центру [10, 28]. TypeScript застосовується на всіх рівнях застосунку – від схеми бази даних до клієнтських компонентів: типи, згенеровані Prisma з моделей бази даних, автоматично поширюються на серверні дії та пропси компонентів, що мінімізує кількість помилок на етапі розробки та спрощує рефакторинг при зміні схеми [11, 26].

Стилізацію інтерфейсу реалізовано засобами Tailwind CSS v4, утилітарний підхід якого дозволяє будувати адаптивні макети без написання окремих CSS-файлів [20]. Своєю чергою, компонентна бібліотека shadcn/ui на базі Radix UI надає доступні та легко кастомізовані UI-примітиви (діалоги, розкриті списки, форми, таблиці), що використовуються як у публічній частині, так і в адміністративній панелі [18, 25]. Для редагування публікацій в адмін-панелі застосовано Tiptap – редактор багатого тексту, що підтримує форматування, вставку медіа та генерує структурований HTML-вміст, який перед збереженням очищується бібліотекою sanitize-html для запобігання XSS-атакам [22]. Табличне відображення даних в адміністративній панелі реалізовано через TanStack Table,

що надає серверну пагінацію, сортування та фільтрацію без додаткових запитів на клієнті [21].

Зберігання даних організовано засобами реляційної СУБД PostgreSQL через ORM Prisma, що забезпечує типобезпечні запити, міграції схеми та генерацію клієнта з повною відповідністю TypeScript-типів [15, 16]. Наскрізну валідацію даних реалізує Zod – схеми визначаються один раз і використовуються як на стороні клієнта при введенні даних у форми, так і на сервері в Server Actions перед записом у базу даних, що гарантує однорідність правил перевірки на обох рівнях [9]. Глобальний клієнтський стан кошика фаншопу керується через Zustand із персистенцією у localStorage під ключем emerald-gang-cart – у сховищі зберігається масив позицій кошика, кожна з яких містить ідентифікатори товару та варіанту, розмір, кількість, ціну, поля кастомізації та актуальний залишок на складі для клієнтської перевірки [14]. Стан відкриття міні-кошика зберігається лише в пам'яті та не персистується між сесіями. Автентифікацію з підтримкою облікових записів із паролем та Google OAuth реалізує NextAuth v5 із стратегією JWT – токен містить поля id та role користувача та зберігається в cookies, тоді як таблиці Session і VerificationToken присутні в схемі як технічна вимога PrismaAdapter, але не використовуються [2]. При реєстрації або першому вході через Google спрацьовують події createUser та signIn, які автоматично прив'язують усі раніше оформлені гостьові замовлення з відповідною поштою до новоствореного акаунту. Мультимовність інтерфейсу забезпечує бібліотека next-intl із конфігурацією localePrefix: "as-needed" – українська мова як мова за замовчуванням не отримує префікса в URL, тоді як англійська доступна за префіксом /en/. Переклади статичного інтерфейсу зберігаються у двох JSON-файлах – uk.json та en.json – із розподілом за namespace-ами відповідно до модулів застосунку, а переклади динамічного контенту зберігаються в окремих таблицях бази даних.

Частину функціоналу делеговано зовнішнім хмарним сервісам. Завантаження та зберігання медіафайлів обробляє UploadThing, що знімає навантаження з основного сервера та забезпечує оптимізовану доставку контенту

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		21

через власну CDN-інфраструктуру [13]. Спортивні дані для синхронізації матчів поточного сезону, турнірних таблиць та гравців надходять із зовнішнього сервісу Sofascore через платформу RapidAPI [17]. Карткові платежі у фаншопі обробляє Stripe через механізм Checkout Session – система формує сесію з переліком позицій, success_url та cancel_url і перенаправляє користувача на захищену сторінку Stripe, не передаючи платіжні дані через власні сервери [19]. Результат транзакції повідомляється через webhook-механізм: подія checkout.session.completed оновлює прапор isPaid у замовленні, а подія checkout.session.expired запускає транзакцію скасування із поверненням залишків на склад. Повторна оплата реалізована через окрему Server Action retryPayment, яка формує нову Checkout Session з явно встановленим терміном дії 30 хвилин через параметр expires_at. Автоматизовані завдання синхронізації спортивних даних виконуються через три Vercel Cron Jobs, визначені у файлі vercel.json: оновлення турнірних таблиць запускається о 3:00 щопонеділка, п'ятниці, суботи та неділі, синхронізація матчів – о 3:00 щопонеділка, синхронізація деталей завершених матчів – о 3:30 того ж дня, через 30 хвилин після попереднього завдання.

Для керування пакетами та залежностями проєкту під час розробки застосовується менеджер npm. Розгортання (deployment) готового застосунку здійснюється на хмарній платформі Vercel, що забезпечує автоматичну CI/CD інтеграцію з Git-репозиторієм. Усі конфіденційні ключі доступу до бази даних PostgreSQL на Neon та зовнішніх API (Stripe, UploadThing, RapidAPI) зберігаються виключно як захищені змінні середовища (Environment Variables), що гарантує безпеку конфігурації платформи.

Вимоги до апаратного забезпечення для розробки передбачають процесор Intel Core i5 або AMD Ryzen 5 не нижче 8-го покоління, оперативну пам'ять не менше 16 ГБ, не менше 20 ГБ вільного місця на SSD, операційну систему Windows 10/11, macOS 11+ або Linux Ubuntu 20.04+, Node.js 20+ та стабільне інтернет-з'єднання для взаємодії із зовнішніми сервісами [8]. Вимоги до серверного середовища включають Node.js-сумісний хостинг із не менше 1 ГБ

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		22

оперативної пам'яті, доступом до PostgreSQL-інстансу та налаштованими змінними середовища для підключення до бази даних і зовнішніх сервісів – Stripe, UploadThing та RapidAPI. Для кінцевих користувачів достатньо будь-якого сучасного браузера – Chrome 90+, Firefox 90+, Safari 14+ або Edge 90+ – з увімкненим JavaScript та інтернет-з'єднанням від 2 Мбіт/с.

Висновки до першого розділу

У першому розділі проведено аналіз предметної області та сформульовано технічну основу для розробки вебпорталу футбольного клубу з інтегрованим фаншопом.

У ході постановки задачі визначено п'ять функціональних модулів системи: контент та медіа, команда, матч-центр, електронна комерція та користувачі. Для кожного модуля окреслено конкретний перелік функціональних вимог, включаючи гібридну статистику гравців, синхронізацію даних із зовнішнім сервісом Sofascore та підтримку двох мов інтерфейсу.

Порівняльний аналіз офіційних вебпорталів «Шахтаря», «Полісся», «Реал Мадрида» та «Арсенала» виявив спільну архітектурну проблему: фаншоп у всіх розглянутих випадках винесено на окремий піддомен, що порушує цілісність сесії покупця та ускладнює адміністрування. Додатково зафіксовано проблеми з наскрізним кошиком. Розроблюваний портал усуває ці недоліки через єдину кодову базу, де фаншоп є повноцінним модулем застосунку.

Для реалізації системи обрано гібридну клієнт-серверну архітектуру на базі Next.js 16 з App Router. Кодова база структурована через механізм route groups із розділенням на публічну зону, фаншоп, автентифікацію та адміністрування. Зберігання даних організовано засобами PostgreSQL через Prisma ORM, що забезпечує типобезпеку на рівні бази даних.

Обґрунтовано вибір інструментального стеку: React 19 та TypeScript – для клієнтської частини, Tailwind CSS v4 та shadcn/ui – для інтерфейсу, Zustand для стану кошика, Zod для наскрізної валідації, NextAuth v5 для автентифікації з

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		23

розмежуванням ролей, next-intl для мультимовності. Зовнішні сервіси UploadThing, Sofascore та Stripe делегують відповідно зберігання медіа, спортивні дані та обробку платежів.

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		24

РОЗДІЛ 2. ПРОЄКТУВАННЯ ТА РЕАЛІЗАЦІЯ ВЕБПОРТАЛУ ФУТБОЛЬНОГО КЛУБУ З ІНТЕГРОВАНИМ ФАНШОПОМ

2.1 Визначення варіантів використання та об'єктно-орієнтованої структури системи

На етапі проєктування вебпорталу футбольного клубу першочерговим завданням є формалізація функціональних вимог через визначення акторів системи та сценаріїв їхньої взаємодії. Для цього застосовано методологію варіантів використання Use Case, яка дозволяє описати межі системи та поведінку кожної групи користувачів. Паралельно розроблено об'єктно-орієнтовану модель на основі сутностей бази даних, що слугує фундаментом для реалізації серверної логіки.

У системі виокремлено чотири актори: гість, авторизований користувач, адміністратор та зовнішня система.

Гість – неавторизований відвідувач порталу. Він має доступ до всього публічного контенту: перегляду новин, профілів гравців і тренерів, матчів із турнірними таблицями та каталогу фаншопу. Гість може додавати товари до кошика – стан якого зберігається в localStorage без потреби в авторизації – та оформлювати замовлення, заповнюючи контактні дані. Для карткової оплати гість перенаправляється на Stripe Checkout, де після підтвердження транзакції повертається на сторінку замовлення. Після оформлення замовлення гість автоматично перенаправляється на сторінку замовлення. Якщо він не авторизований, сторінка пропонує увійти через Google для автоматичного прив'язування замовлення до акаунту.

Авторизований користувач отримує все, що доступне гостю, плюс особистий кабінет з історією замовлень. Авторизація підтримує два варіанти: через Google OAuth або через email і пароль. При вході через Google замовлення, оформлені раніше з тією ж поштою, автоматично прив'язуються до акаунту. При

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		25

вході через email/пароль користувач може вручну прив'язати замовлення до акаунту за номером замовлення і телефоном.

Адміністратор має повний доступ до панелі керування. У блоці новин він може створювати, редагувати, архівувати публікації та остаточно видаляти їх з архіву. У блоці команди – керувати профілями гравців і тренерів, а також запускати синхронізацію складу команди з Sofascore вручну для оновлення даних гравців. Блок турнірів та матчів надає інструменти для створення матчів вручну, синхронізації матчів поточного сезону через API, синхронізації деталей матчу (подій та складу), а також управління турнірами, сезонами, суперниками, турнірними таблицями та словником перекладів назв команд. Система захищає від некоректних операцій: турнір неможливо архівувати, якщо до нього прив'язані активні матчі; категорію – якщо до неї прив'язані товари; активний сезон також не можна архівувати. У блоці фаншопу адміністратор керує замовленнями – переглядає деталі, змінює статуси доставки, скасовує оплату, архівує оброблені замовлення, а також товарами з прив'язкою до гравців, розмірною сіткою та медіафайлами, і категоріями. Галерея дозволяє створювати фотозвіти з матчів із опціональною прив'язкою до конкретного матчу.

Зовнішня система представлена двома компонентами. Vercel Cron автоматично запускає три завдання за розкладом: оновлення турнірних таблиць чотири дні на тиждень у дні проведення матчів, синхронізацію матчів поточного сезону щопонеділка та синхронізацію деталей завершених матчів – складів і подій, через 30 хвилин після попереднього завдання. Синхронізація складу команди запускається вручну адміністратором через відповідну кнопку в адмін-панелі. Stripe Webhook обробляє події платіжного шлюзу: при успішній оплаті оновлює поле isPaid у замовленні, а при закінченні сесії оплати – скасовує замовлення та повертає залишки товарів на склад.

Діаграма варіантів використання унаочнює межі системи та функціональні зв'язки між акторами і сценаріями взаємодії (додаток Ж, рис. Ж.1-Ж.2). У зв'язку з обсягом діаграму розміщено на двох частинах: перша відображає публічну

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		26

частину та особистий кабінет (див. додаток Ж, рис. Ж.1), а друга – адміністрування та автоматизацію (див. додаток Ж, рис. Ж.2).

Об'єктно-орієнтована модель системи сформована на основі сутностей бази даних Prisma. Кожна сутність відповідає окремій таблиці PostgreSQL і реалізує конкретну частину предметної області. У зв'язку з великим обсягом моделі діаграму класів розділено на три частини за функціональними підсистемами: підсистеми користувачів, контенту та медіа (рис. 2.1), команди та матч-центру (рис. 2.2) і електронної комерції (додаток К, рис. К.1).

Діаграма класів підсистем користувачів, контенту та медіа відображає дві логічні групи сутностей (див. рис. 2.1). У пакеті «Користувачі» центральною є клас User із полями ідентифікатора, електронної пошти, хешу пароля та ролі (перелік UserRole: ADMIN, USER). Клас Account зберігає OAuth-акаунти зовнішніх провайдерів із унікальним складеним ключем (provider, providerAccountId); класи Session та VerificationToken присутні як технічна вимога адаптера PrismaAdapter і не беруть участі в бізнес-логіці застосунку. У пакеті «Контент та медіа» клас Post описує публікацію з типом (перелік PostType: NEWS, INTERVIEW, STATEMENT) та контекстом команди (перелік TeamContext: MAIN_TEAM, U19, ACADEMY, GENERAL). Локалізований вміст публікацій зберігається у класі PostTranslation через відношення композиції «один до багатьох». Клас Gallery із перекладами GalleryTranslation слугує контейнером фотозвітів. Клас Media є спільним ресурсом, пов'язаним із Post та Gallery відношеннями агрегації «багато до багатьох» через неявні зведені таблиці Prisma.

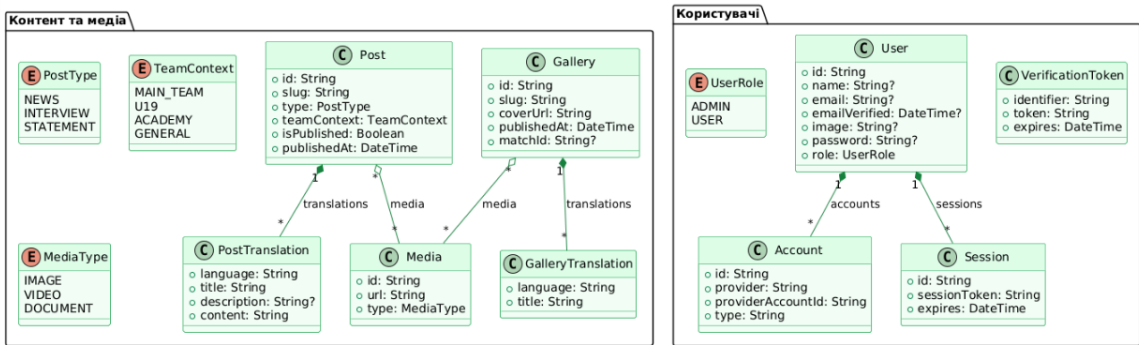


Рис. 2.1 – Діаграма класів підсистем користувачів, контенту та медіа

```
classDiagram
    class MatchStatus {
        SCHEDULED
        LIVE
        FINISHED
        POSTPONED
        CANCELED
    }
    class EventType {
        GOAL
        ASSIST
        YELLOW_CARD
        RED_CARD
        SUBSTITUTION_IN
        SUBSTITUTION_OUT
    }
    class Tournament {
        id: String
        slug: String
        sofascored: Int?
        hasStandings: Boolean
    }
    class Season {
        id: String
        slug: String
        sofascored: Int?
        isActive: Boolean
        startDate: DateTime
        endDate: DateTime
    }
    class Opponent {
        id: String
        slug: String
        sofascored: Int?
        logoUrl: String?
    }
    class TeamDictionary {
        id: String
        sofascored: Int
        originalName: String
    }
    class Standing {
        id: String
        rank: Int
        teamName: String
        teamLogo: String?
        points: Int
        played: Int
        win: Int
        draw: Int
        lose: Int
        goalsFor: Int
        goalsAgainst: Int
        goalsDiff: Int
        teamContext: TeamContext
    }
    class Match {
        id: String
        slug: String
        sofascored: Int?
        date: DateTime
        startTimeStamp: Int?
        status: MatchStatus
        teamContext: TeamContext
        isHomeGame: Boolean
        homeScore: Int?
        awayScore: Int?
        opponentLineup: json?
        isDetailsSynced: Boolean
        round: Int?
        stadium: String?
        homeCoachName: String?
        awayCoachName: String?
        highlightOut: String?
        postMatchUrl: String?
    }
    class MatchLineup {
        id: String
        isStarter: Boolean
        played: Boolean
    }
    class MatchEvent {
        type: EventType
        minute: Int
        isOpponent: Boolean
        customPlayerName: String?
    }
    class TeamContext {
        MAIN_TEAM
        U19
        ACADEMY
        GENERAL
    }
    class TournamentTranslation {
        language: String
        name: String
    }
    class OpponentTranslation {
        language: String
        name: String
    }
    class TeamDictionaryTranslation {
        language: String
        name: String
    }
    class Player {
        id: String
        slug: String
        sofascored: Int?
        number: Int
        position: PlayerPosition
        avatar: String?
        isManualAvatar: Boolean
        birthDate: DateTime?
        height: Int?
        weight: Int?
        nationality: String?
        initialMatches: Int
        initialGoals: Int
        initialAssists: Int
        initialCleanSheets: Int
        initialGoalsConceded: Int
    }
    class PlayerPosition {
        GOALKEEPER
        DEFENDER
        MIDFIELDER
        FORWARD
    }
    class Coach {
        id: String
        slug: String
        avatar: String?
        teamContext: TeamContext
        nationality: String?
        birthDate: DateTime?
    }
    class PlayerTranslation {
        language: String
        name: String
        bio: String?
    }
    class CoachTranslation {
        language: String
        name: String
        role: String
        bio: String?
    }

    MatchStatus --> Match
    EventType --> MatchEvent
    Tournament --> Match : tournament
    Season --> Match : season
    Opponent --> Match : opponent
    TeamDictionary --> Match : translations
    Standing --> Match : translations
    Match --> MatchLineup : lineup
    Match --> MatchEvent : events
    MatchLineup --> MatchEvent : lineupEntries
    MatchEvent --> Match : events
    TeamContext --> Match : translations
    TournamentTranslation --> Match : translations
    OpponentTranslation --> Match : translations
    TeamDictionaryTranslation --> Match : translations
    Player --> Match : translations
    Coach --> Match : translations
    PlayerTranslation --> Match : translations
    CoachTranslation --> Match : translations
```

Діаграма класів підсистеми електронної комерції відображає повний цикл
 йн-продажів (див. додаток К, рис. К.1). Клас Order є центральним: він
 ь контактні дані покупця, метод оплати (перелік PaymentMethodEnum:

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				28
Змн.	Арк.	№ докум.	Підпис	Дата		

CARD, COD), прапор isPaid та статус (перелік OrderStatusEnum: PENDING, PAID, SHIPPED, DELIVERED, CANCELLED). Зв'язок із User через поле userId є необов'язковим – замовлення може існувати без акаунту. Клас OrderItem фіксує рядок замовлення з ціною на момент купівлі (fixedPrice) та полями кастомізації customName і customNumber для нанесення на форму. Клас Product зберігає каталожні дані товару, включаючи ціну, ціну розпродажу, цільову аудиторію (перелік Demographic: UNISEX, MEN, WOMEN, KIDS) та атрибути фільтрації – колір, тип одягу, рік сезону. Клас ProductVariant деталізує товар за розміром із залишком на складі та унікальним артикулом. Клас Category із перекладами CategoryTranslation забезпечує ієрархічну організацію каталогу. Локалізовані назви та описи товарів зберігаються у класі ProductTranslation через відношення композиції «один до багатьох».

2.2 Розробка бази даних системи

Для зберігання даних вебпорталу обрано реляційну СУБД PostgreSQL. Вибір обумовлений необхідністю підтримки складних зв'язків між сутностями системи, транзакційної цілісності при обробці замовлень та ефективного індексування для фільтрації каталогу фаншопу. Взаємодія з базою даних реалізована через ORM Prisma, що забезпечує типобезпечні запити та автоматичне керування міграціями схеми.

База даних організована за модульним принципом і охоплює п'ять предметних областей: користувачів, контент та медіа, команду, матч-центр та електронну комерцію (додаток Д, рис. Д.1-Д.3). У зв'язку з великим обсягом схеми, її розділено на три частини та винесено в додатки: користувачі, контент та медіа (див. додаток Д, рис. Д.1), команда та матч-центр (див. додаток Д, рис. Д.2), а також електронна комерція (див. додаток Д, рис. Д.3).

У схемі застосовано два наскрізні патерни проєктування. Перший – мультимовність через таблиці перекладів. Кожна сутність з локалізованим вмістом має окрему таблицю перекладів із полем language та складеним

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		29

унікальним ключем (entityId, language). Цей патерн реалізовано для Post, Player, Coach, Product, Category, Tournament, Opponent, Gallery та TeamDictionary. Це дозволяє зберігати українську та англійську версії контенту в одній таблиці без дублювання основних записів.

Другий патерн – м'яке видалення через поле deletedAt. Замість фізичного видалення запис отримує мітку часу в цьому полі, що дозволяє зберігати історичні дані та відновлювати видалені записи. Патерн застосовано для User, Post, Player, Coach, Match, Tournament, Opponent, Product, Order, Gallery, Category, Media та Season.

Модуль користувачів побудований навколо таблиці User, яка є центральною сутністю системи. Вона зберігає ідентифікатор у форматі CUID, електронну пошту з унікальним обмеженням, хеш пароля для облікових записів із паролем автентифікацією, роль (ADMIN/USER) та мітку м'якого видалення. Таблиця Account зберігає токени доступу OAuth-провайдера. Таблиці Session та VerificationToken присутні як технічна вимога PrismaAdapter і не містять бізнес-логіки – автентифікаційний стан зберігається через JWT у cookies.

Модуль контенту та медіа побудований навколо таблиці Post, яка зберігає публікації з типом (NEWS, INTERVIEW, STATEMENT), контекстом команди та статусом публікації. Індеси на полях publishedAt, isPublished, type та teamContext забезпечують швидку фільтрацію на сторінках новин. Локалізований вміст – заголовок, опис та тіло публікації – зберігається в PostTranslation.

Таблиця Media є спільним сховищем медіафайлів для всіх модулів системи. Зв'язки між Media та публікаціями, гравцями, тренерами, товарами й галереями реалізовані через неявні зведені таблиці Prisma. Таблиця Gallery зберігає фотогалереї з опціональним зовнішнім ключем matchId для прив'язки до конкретного матчу.

Модуль команди охоплює дві таблиці – Player та Coach. Поле sofascoreId у таблиці Player має унікальне обмеження, що унеможливорює прив'язку двох профілів до одного ідентифікатора зовнішнього сервісу. Індекс на полі deletedAt

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		30

забезпечує ефективну вибірку лише активних гравців без сканування всієї таблиці. Таблиця Coach має аналогічну структуру без полів статистики та поля sofascoreId.

Модуль матч-центру побудований навколо таблиці Match, яка є центральною сутністю. Вона пов'язана з Tournament, Season та Opponent через зовнішні ключі. Поле isDetailsSynced фіксує стан синхронізації деталей матчу. Склад суперника зберігається у полі opponentLineup типу JSON без нормалізації, оскільки ці дані використовуються лише для відображення і не потребують реляційних запитів.

Таблиця MatchLineup реалізує склад команди для конкретного матчу. Складений унікальний ключ (matchId, playerId) унеможливлює дублювання гравців у складі. Прапор isStarter розділяє стартовий склад і запасних. Таблиця MatchEvent фіксує всі події матчу: голи, асисті, картки та заміни з хвилиною та прив'язкою до гравця. Поле isOpponent дозволяє фіксувати події суперника без окремих таблиць.

Таблиця Standing зберігає рядок турнірної таблиці з прив'язкою до Tournament та Season. Оновлення відбувається при синхронізації через Sofascore API. Таблиця TeamDictionary забезпечує переклад назв команд суперників, отриманих із зовнішнього API, на українську та англійську мови.

Таблиця Season описує ігровий сезон із датами початку та завершення і прапором isActive – лише один сезон може бути активним одночасно, що унеможливлює конфліктне прив'язування матчів і таблиць.

Таблиця Opponent зберігає команди-суперники з ідентифікатором Sofascore для синхронізації та перекладами назв через OpponentTranslation.

Модуль електронної комерції реалізований через таблиці Product, ProductVariant, Order та OrderItem. Таблиця Product зберігає товари каталогу з ціною, ціною розпродажу, цільовою аудиторією (UNISEX, MEN, WOMEN, KIDS) та атрибутами фільтрації – кольором, типом одягу, роком сезону. Набір індексів на цих полях, зокрема композитний (categoryId, isArchived, deletedAt), забезпечує ефективну фільтрацію в каталозі. Таблиця ProductVariant зберігає

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		31

варіанти товару за розміром із залишком на складі. Складений унікальний ключ (productId, size) унеможливорює дублювання розмірів.

Таблиця Order зберігає повні контактні дані покупця, метод оплати (CARD/COD) та прапор isPaid, який оновлюється вебхуком Stripe при підтвердженні транзакції. Зв'язок із User через userId є необов'язковим – замовлення може існувати без акаунту. Таблиця OrderItem фіксує рядок замовлення з ціною на момент покупки (fixedPrice) та полями кастомізації – іменем і номером для нанесення на форму.

Між модулями системи існують зв'язки типу «багато до багатьох», реалізовані через неявні зведені таблиці Prisma. Публікація (Post) пов'язана з гравцями (Player) для позначки згаданих футболістів та з матчами (Match) для прив'язки матеріалів до конкретних ігор. Товари (Product) пов'язані з гравцями (Player) для реалізації контекстних продажів – переходу з профілю гравця до купівлі його іменної форми. Медіафайли (Media) є спільним ресурсом для публікацій, гравців, тренерів, товарів та галерей.

2.3 Проєктування та реалізація алгоритмів роботи системи

Функціонування вебпорталу забезпечується комплексом алгоритмів, що реалізують ключові бізнес-процеси: автентифікацію користувачів, оформлення замовлень, обробку платежів, синхронізацію спортивних даних та обчислення статистики гравців. Кожен алгоритм спроектовано з урахуванням вимог до безпеки, цілісності даних та стійкості до помилок. У межах підрозділу розроблено та описано сім алгоритмів. Перший реалізує автентифікацію користувачів із підтримкою двох сценаріїв входу. Другий забезпечує транзакційне оформлення замовлення з перевіркою залишків на складі. Третій обробляє результати платежів через Stripe Webhook. Четвертий, п'ятий та шостий алгоритми автоматизують синхронізацію матчів поточного сезону, деталей завершених матчів і турнірних таблиць із зовнішнім сервісом Sofascore. Сьомий

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		32

алгоритм обчислює гібридну статистику гравців, поєднуючи архівні дані з показниками поточного сезону.

Загальна схема взаємодії акторів із системою ілюструє ці процеси (рис. 2.3).

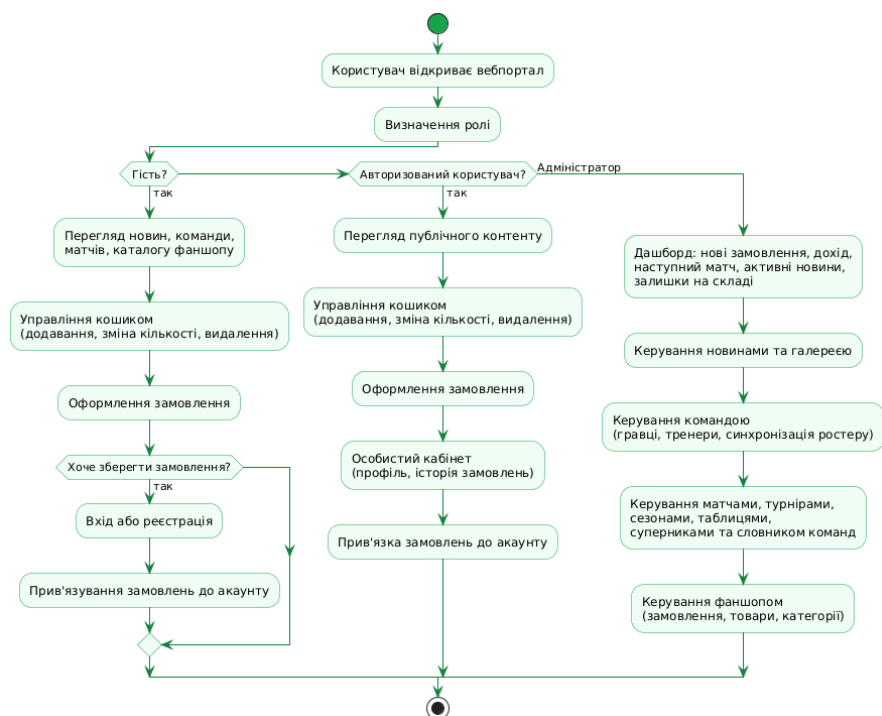


Рис. 2.3 – Загальний алгоритм роботи з порталом

Алгоритм автентифікації реалізує два сценарії входу: через форму з email і паролем та через Google OAuth (додаток II, рис. II.1).

При вході через форму дані валідуються схемою Zod, після чого NextAuth v5 викликає провайдер credentials. Система знаходить користувача за email у базі даних. Якщо обліковий запис не знайдено або пароль не збігається з хешем bcrypt – повертається узагальнена помилка, що не дозволяє визначити, чи зареєстрований конкретний email. При успішній перевірці формується JWT-токен із полями id та role, що зберігається в cookies.

При вході через Google OAuth-провайдер передає профіль PrismaAdapter, який шукає обліковий запис за provider та providerAccountId. Якщо акаунт не знайдено – створюється новий запис User та прив'язується Account. Після цього спрацьовує подія createUser, яка знаходить усі замовлення з відповідною поштою та userId = null і автоматично прив'язує їх до новоствореного акаунту. Аналогічна

логіка реалізована в події signIn для повторних входів через Google – вона забезпечує прив'язку замовлень, оформлених гостем до реєстрації.

Алгоритм оформлення замовлення є найкритичнішим з точки зору цілісності даних, оскільки одночасно перевіряє залишки, списує товари зі складу та створює запис замовлення (додаток И, рис. И.2).

Після валідації форми Zod та перевірки непорожності кошика алгоритм запускає транзакцію Prisma. У межах транзакції для кожного товару перевіряється наявність варіанту та достатність залишку на складі. Якщо залишок менший за замовлену кількість – транзакція переривається з поверненням помилки OutOfStock із точною кількістю доступного товару. При успішній перевірці всіх позицій залишки зменшуються, після чого одним запитом створюється замовлення з усіма OrderItem.

Подальший сценарій залежить від методу оплати. При виборі COD (накладений платіж) виконується перенаправлення на сторінку замовлення – транзакція завершена. При виборі CARD система формує Stripe Checkout Session із line_items, success_url, cancel_url та metadata.orderId, після чого перенаправляє користувача на захищену сторінку Stripe. Платіжні дані картки ніколи не проходять через сервери порталу.

Алгоритм обробки результату платежу реалізований через Stripe Webhook. При отриманні події checkout.session.completed система оновлює поле isPaid = true у відповідному замовленні. При події checkout.session.expired (термін сесії вичерпано) запускається транзакція: замовлення отримує статус CANCELLED, а залишки всіх позицій повертаються на склад. Повторна оплата реалізована через окрему Server Action retryPayment, яка перевіряє права доступу, статус замовлення та формує нову Stripe Checkout Session з терміном дії 30 хвилин.

Алгоритм синхронізації матчів запускається як автоматично через Vercel Cron щопонеділка, так і вручну адміністратором з адмін-панелі (додаток И, рис. И.3).

Спочатку система знаходить активний сезон у базі даних. За відсутності активного сезону синхронізація переривається з відповідним повідомленням.

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		34

Далі виконуються два паралельні запити до Sofascore API – get-last-matches та get-next-matches. Отримані події об'єднуються та дедуплікуються за sofascoreId. Кожен матч перевіряється на відповідність датам активного сезону – події поза межами сезону пропускаються.

Для кожного матчу алгоритм визначає турнір і суперника: якщо вони відсутні в базі – створюються нові записи з перекладами. Якщо матч із таким sofascoreId вже існує – оновлюються рахунок, статус та дата. Якщо не існує – створюється новий запис із прив'язкою до активного сезону, турніру та суперника. Уся робота виконується в межах однієї транзакції з таймаутом 30 секунд.

Алгоритм синхронізації деталей запускається через Cron через 30 хвилин після синхронізації матчів. За один запуск обробляється максимум 3 завершені матчі з isDetailsSynced = false – це запобігає перевищенню ліміту API. Між обробкою матчів витримується затримка 500 мс (додаток II, рис. II.4).

Для кожного матчу виконуються три паралельні запити до Sofascore: склади (get-lineups), події (get-incidents) та деталі (detail). Перед записом нових даних видаляються всі існуючі записи MatchLineup та MatchEvent для цього матчу. Склад суперника зберігається у полі opponentLineup у форматі JSON. Для гравців власної команди виконується пошук у базі даних за sofascoreId – знайдені гравці додаються до MatchLineup.

При обробці інцидентів алгоритм розпізнає голи, асисті, картки та заміни. Для власних голів (ownGoal) логіка інвертує приналежність події: гол зараховується суперникові, хоча м'яч забив гравець власної команди. При замінах алгоритм додатково оновлює прапор played = true для гравця, що вийшов на заміну. Після успішної обробки встановлюється isDetailsSynced = true.

Алгоритм синхронізації турнірних таблиць запускається автоматично чотири дні на тиждень. Система знаходить активний сезон із sofascoreId та всі турніри з прапором hasStandings = true (додаток II, рис. II.5).

Для кожного турніру виконується запит до Sofascore, отримані рядки замінюють наявні записи Standing у базі даних. При відображенні назв команд

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		35

алгоритм звертається до TeamDictionary – словника перекладів. Якщо команда зустрічається вперше, новий запис автоматично створюється в словнику з оригінальною назвою для обох мов, яку адміністратор може перекласти пізніше вручну.

Алгоритм гібридної статистики гравців поєднує два джерела даних: вручну введені показники попередніх сезонів (initialGoals, initialMatches тощо) та динамічно обчислені показники поточного сезону на основі матчових подій у базі даних (додаток И, рис. И.6).

При завантаженні профілю гравця сторінка отримує з бази даних lineupEntries (зіграні матчі зі статусом FINISHED) та events (матчові події поточного сезону). Функція calculateHybridStats визначає кількість зіграних матчів як довжину lineupEntries, голи та асисті – як кількість відповідних подій із isOpponent = false, що виключає події суперника з підрахунку. Для воротарів додатково обчислюються пропущені голи і «сухі» матчі: для кожного зіграного матчу визначається кількість голів суперника, а нульові значення рахуються як cleanSheet. Підсумкові показники є сумою ручних і динамічних даних, що забезпечує коректне відображення статистики незалежно від повноти наповнення бази даних. Результат відображається в компоненті PlayerQuickStats, що показує три ключові метрики залежно від позиції гравця: для польових гравців – матчі, голи та асисті, для воротарів – матчі, пропущені голи та «сухі» матчі.

2.4 Реалізація функціоналу системи вебпорталу футбольного клубу з інтегрованим фаншопом

У цьому підрозділі розглянуто технічну реалізацію вебпорталу: організацію кодової бази, ключові архітектурні рішення та інтеграцію зовнішніх сервісів (рис. 2.4).

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		36

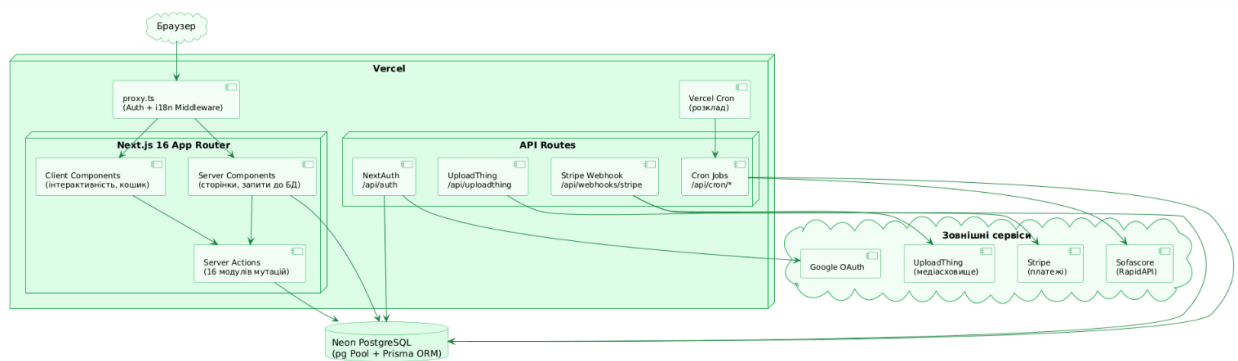


Рис. 2.4 – Діаграма системної архітектури

Проект організований за стандартною структурою Next.js 16 з App Router. Кореневий каталог `src/` містить такі директорії: `app/` – маршрутизація та сторінки, `actions/` – серверні дії, `components/` – компоненти інтерфейсу, `hooks/` – клієнтські хуки, `i18n/` – конфігурація інтернаціоналізації, `lib/` – утиліти та сервіси, `messages/` – файли перекладів, `store/` – клієнтський стан.

Маршрутизація застосунку організована через механізм `route groups` усередині каталогу `app/[locale]/`: `(main)` – публічна частина порталу, `(shop)` – фаншоп, `(auth)` – сторінки входу та реєстрації, `(admin)` – адміністративна панель. `Route groups` не впливають на URL-адреси – вони служать виключно для логічного розмежування частин застосунку та застосування різних `layout`-компонентів до кожної зони. Окремий каталог `app/api/` містить маршрути для автентифікації NextAuth, Cron-завдань, завантаження файлів через UploadThing та обробки Stripe Webhook [13].

Бізнес-логіка мутацій даних реалізована через 16 файлів Server Actions у каталозі `src/actions/`: `auth.ts`, `category.ts`, `checkout.ts`, `dictionary.ts`, `gallery.ts`, `match.ts`, `news.ts`, `opponent.ts`, `order.ts`, `payment.ts`, `product.ts`, `profile.ts`, `season.ts`, `standings.ts`, `team.ts`, `tournament.ts`. Такий підхід усуває потребу в окремому REST API – клієнтські компоненти викликають серверні функції напряму, а Next.js 16 автоматично формує захищені POST-ендпоінти.

Клієнтські форми адмін-панелі реалізовані за єдиним патерном на базі хука `useActionState` з React 19. Оскільки Server Actions не приймають довільні аргументи окрім `FormData`, динамічний стан форми (вміст редактора, обрані медіафайли, прив'язані сутності) передається через метод `.bind()` – перед

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		37

передачею в `useActionState` action зв'язується з поточним станом клієнтського компонента. Хук повертає три значення: поточний стан відповіді, функцію для передачі у `form action` та прапор `isPending` для блокування UI під час виконання запиту. Результат обробляється через `useEffect` – при `state.success` відбувається перенаправлення, при помилці – відображається `toast`-сповіщення через бібліотеку `Sonner`.

Підключення до PostgreSQL на платформі Neon реалізоване через `@prisma/adapters-pg`. Замість стандартного TCP-підключення використовується пул з'єднань через `Pool` із бібліотеки `pg`, що є обов'язковою умовою для коректної роботи в безсерверному середовищі `Vercel` – кожен запит виконується в окремому ізольованому контексті без підтримки постійних з'єднань. Для запобігання створенню кількох екземплярів `PrismaClient` під час гарячого перезавантаження в режимі розробки використовується патерн `singleton` через `globalThis` [27].

Файл `proху.ts` реалізує комбінований `middleware`, що поєднує логіку автентифікації `NextAuth` та інтернаціоналізацію `next-intl` в одному обробнику. Порядок виконання такий: спочатку визначається поточна локаль із першого сегмента URL, потім нормалізується шлях без префікса локалі. Далі застосовуються три правила захисту: авторизовані користувачі перенаправляються з сторінок входу та реєстрації до профілю; неавторизовані – з захищених сторінок (`/profile`, `/admin`) до сторінки входу; користувачі без ролі `ADMIN` – з адміністративних маршрутів до профілю. Після перевірки всіх умов запит передається обробнику `next-intl` для визначення локалі. `Matcher` у конфігурації виключає статичні файли, маршрути `_next` та `api`.

Мультимовність реалізована через бібліотеку `next-intl` із конфігурацією `localePrefix: "as-needed"`. Це означає, що українська мова (мова за замовчуванням) не має префікса в URL, тоді як англійська доступна за префіксом `/en/`. Переклади інтерфейсу зберігаються у JSON-файлах у каталозі `messages/` – по одному файлу на локаль. Переклади динамічного контенту (назви гравців, товарів, турнірів тощо) зберігаються у відповідних таблицях бази даних через

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		38

патерн окремих таблиць перекладів. Навігаційні компоненти `Link`, `redirect`, `usePathname` та `useRouter` імпортуються з обгортки `il8n/navigation.ts`, що автоматично враховує поточну локаль при формуванні URL.

У `next.config.ts` підключено плагін `next-intl/plugin` та визначено список дозволених зовнішніх доменів для компонента `next/image`: `lh3.googleusercontent.com` для аватарів Google OAuth, домени `UploadThing` (`utfs.io`, `uploadthing.com`, `buinwmyrpxz.ufs.sh`) для медіаконтенту, `img.sofascore.com` для логотипів команд та аватарів гравців, `img.youtube.com` для обкладинок відео.

Типова сторінка реалізована як асинхронний серверний компонент із двома експортованими функціями: `generateMetadata` для формування SEO-метаданих та `Open Graph` тегів і основний компонент сторінки. Параметри фільтрації передаються через `searchParams`, що дозволяє зберігати стан фільтрів в URL і ділитися посиланнями на відфільтровані результати. Компоненти, що потребують клієнтської інтерактивності (фільтри, перемикачі вкладок, кошик), позначені директивою `"use client"` і отримують дані через пропси від серверних батьківських компонентів.

Застосунок розгорнуто на платформі Vercel із автоматичним CI/CD [29]. База даних PostgreSQL розміщена на платформі Neon у безсерверному режимі – підключення через `@prisma/adapters-pg` забезпечує сумісність із середовищем Vercel [12]. Vercel Cron Jobs виконують три заплановані завдання для синхронізації спортивних даних без потреби в окремому сервері.

Висновки до другого розділу

У другому розділі виконано повний цикл проєктування та реалізації вебпорталу футбольного клубу з інтегрованим фаншопом – від формалізації функціональних вимог до технічного опису готового застосунку.

У ході визначення варіантів використання виокремлено чотири актори системи: гість, авторизований користувач, адміністратор та зовнішня система.

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		39

Побудовано діаграму варіантів використання, що охоплює публічний контент, фаншоп (як модуль електронної комерції), особистий кабінет, адміністрування та автоматизовану синхронізацію спортивних даних. Об'єктно-орієнтована модель сформована на основі сутностей бази даних із визначеними атрибутами, зв'язками та обмеженнями цілісності.

Спроектовано реляційну базу даних PostgreSQL, організовану за п'ятьма предметними областями: користувачі, контент та медіа, команда, матч-центр та електронна комерція. У схемі реалізовано два наскрізні патерни – мультимовність через окремі таблиці перекладів для дев'яти сутностей та м'яке видалення через поле deletedAt для тринадцяти таблиць, що забезпечує збереження історичних даних без фізичного знищення записів. Описано всі ключові таблиці, їхні індекси та міжмодульні зв'язки типу «багато до багатьох», реалізовані через неявні зведені таблиці Prisma.

Спроектовано та описано сім алгоритмів роботи системи з відповідними діаграмами активності. Алгоритм автентифікації реалізує два сценарії входу – через email/пароль та Google OAuth – з автоматичним прив'язуванням гостей до замовлень до акаунту через події NextAuth. Алгоритм оформлення замовлення забезпечує транзакційну перевірку залишків на складі та інтеграцію зі Stripe Checkout. Алгоритм обробки результатів платежу реалізований через Stripe Webhook з автоматичним поверненням товарів на склад при закінченні сесії. Три алгоритми синхронізації з Sofascore через RapidAPI автоматизують оновлення матчів поточного сезону, деталей завершених матчів та турнірних таблиць за допомогою Vercel Cron. Алгоритм гібридної статистики поєднує вручну введені архівні показники попередніх сезонів із динамічно обчисленими даними поточного.

Описано технічну реалізацію застосунку. Кодова база організована через механізм route groups App Router із розділенням на чотири функціональні зони без впливу на URL-адреси. Бізнес-логіка мутацій реалізована через 16 Server Actions, що усуває потребу в окремому REST API. Клієнтські форми адмін-панелі побудовані за єдиним патерном на базі useActionState з React 19 із

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		40

передачею стану через `.bind()`. Підключення до Neon PostgreSQL реалізоване через `@prisma/adapters-pg` з пулом з'єднань для сумісності з безсерверним середовищем. Забезпечено безперервну інтеграцію та доставку коду з автоматичним розгортанням готового застосунку на платформі Vercel.

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		41

РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З ВЕБПОРТАЛОМ ФУТБОЛЬНОГО КЛУБУ З ІНТЕГРОВАНИМ ФАНШОПОМ

3.1 Порядок встановлення та налаштування параметрів платформи

Для забезпечення публічного доступу до порталу та перевірки його роботи у реальних умовах виконано розгортання у хмарних сервісах. Production-середовище побудоване на безкоштовних тарифах двох незалежних провайдерів і не потребує власного серверного обладнання, що робить його придатним для демонстраційного проєкту з мінімальними витратами на інфраструктуру.

Платформа розділена на два рівні. Застосунок Next.js розгорнуто на платформі Vercel, яка забезпечує безсерверне виконання, глобальний CDN та автоматичне розгортання [24]. База даних PostgreSQL розміщена на платформі Neon [12] у безсерверному режимі з підтримкою пулу з'єднань. Кожен із провайдерів відповідає за свою задачу та може масштабуватися незалежно. Такий поділ відповідальності спрощує діагностику збоїв та дозволяє замінювати окремі компоненти інфраструктури без впливу на решту системи.

Процес розробки організовано через GitLab-репозиторій із налаштованим CI/CD пайплайном [3]. При кожному push до будь-якої гілки автоматично запускаються три стадії перевірки: lint – статичний аналіз коду через ESLint, type-check – перевірка типів TypeScript через tsc --noEmit та build – повна збірка Next.js 16 із попередньою генерацією Prisma-клієнта [7]. Після успішного проходження всіх стадій на гілці main виконується четверта стадія mirror – автоматичне дзеркалювання коду до приватного GitHub-репозиторію. Vercel відстежує цей GitHub-репозиторій і ініціює розгортання при кожному новому коміті. Такий підхід забезпечує безперервну інтеграцію та унеможливорює розгортання коду, що не пройшов автоматичні перевірки (рис. 3.1).

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		42

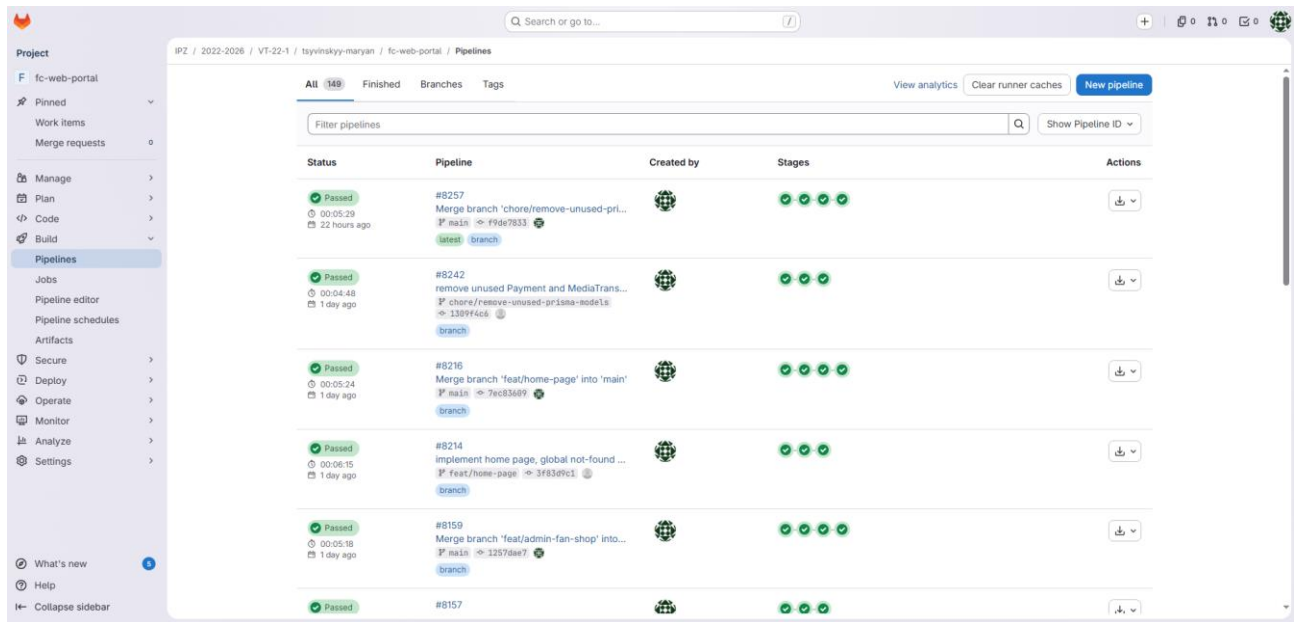


Рис. 3.1 – Пайплайни GitLab CI/CD із пройденими стадіями

На платформі neon.tech створюється новий проєкт. При створенні обирається регіон AWS Europe Central 1 (Frankfurt) для мінімальної затримки відповіді. Після створення проєкту платформа надає два рядки підключення: DATABASE_URL через pgBouncer для запитів застосунку та DATABASE_URL_UNPOOLED для прямого підключення, яке використовується при виконанні міграцій Prisma (рис. 3.2).

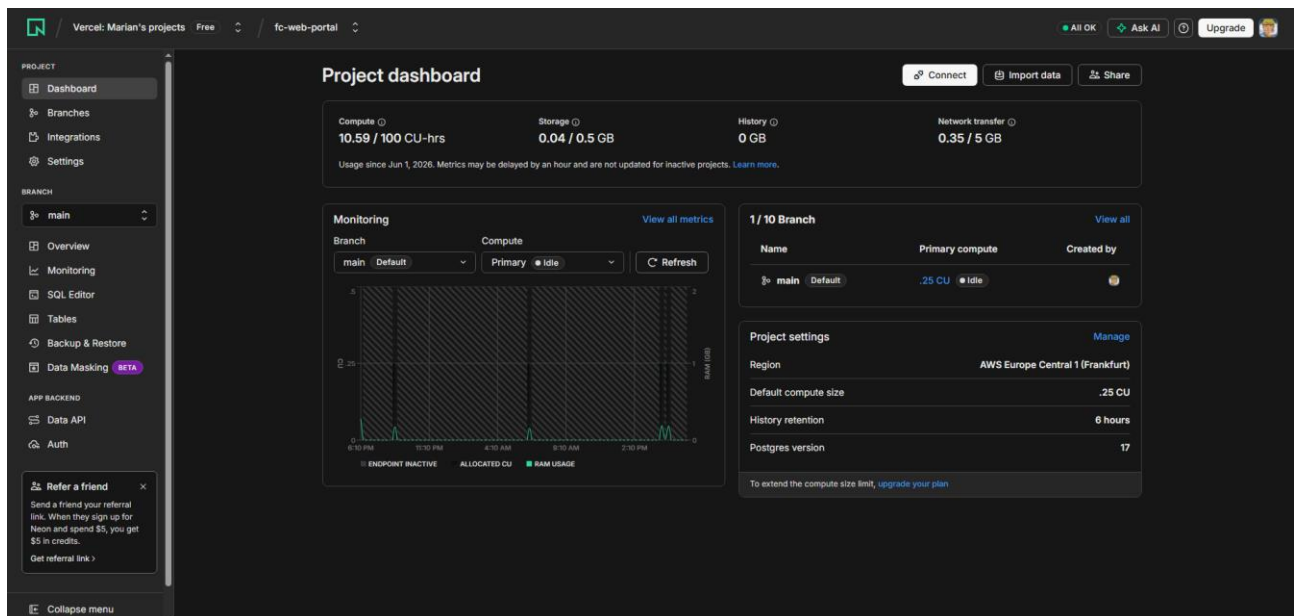


Рис. 3.2 – Дашборд Neon із параметрами бази даних

У дашборді Vercel імпортується GitHub-репозиторій fc-web-portal. Після підключення репозиторію у розділі Storage підключається база даних Neon через

інтеграцію – платформа автоматично заповнює змінні підключення до PostgreSQL у змінних середовища проєкту. Решта змінних вводиться вручну у розділі Environment Variables (табл. 3.1).

Таблиця 3.1

Змінні середовища застосунку

Змінна	Призначення
DATABASE_URL	Рядок підключення до PostgreSQL через pgBouncer
DATABASE_URL_UNPOOLED	Пряме підключення без pgBouncer для міграцій
AUTH_SECRET	Секретний ключ для підпису JWT-токенів NextAuth
AUTH_URL	Базова URL-адреса застосунку
AUTH_TRUST_HOST	Дозвіл довіри хосту (значення true)
AUTH_GOOGLE_ID	Client ID OAuth-додатку Google
AUTH_GOOGLE_SECRET	Client Secret OAuth-додатку Google
UPLOADTHING_TOKEN	Токен для роботи з UploadThing API
UPLOADTHING_SECRET	Секретний ключ UploadThing
CRON_SECRET	Секретний ключ для авторизації Cron-запитів
RAPIDAPI_KEY	Ключ доступу до Sofascore через RapidAPI

NEXT_PUBLIC_STRIPE_PUBLISHABLE_KEY	Публічний ключ Stripe
STRIPE_SECRET_KEY	Секретний ключ Stripe для серверних запитів
STRIPE_WEBHOOK_SECRET	Секрет для верифікації Stripe Webhook подій

Зовнішній вигляд налаштованих змінних середовища у дашборді Vercel проілюстровано нижче (рис. 3.3).

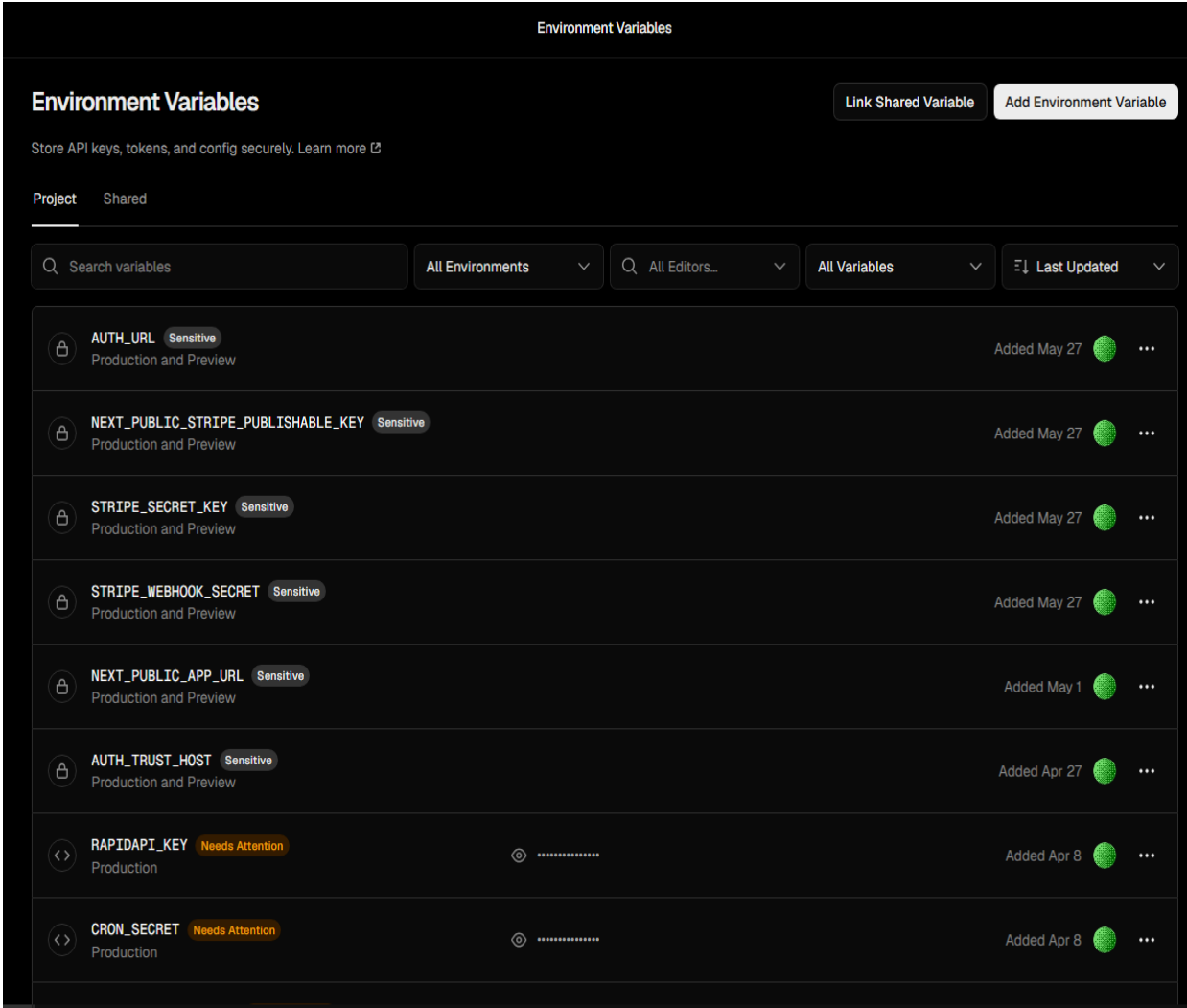


Рис. 3.3 – Змінні середовища проєкту у дашборді Vercel

Після збереження змінних Vercel автоматично запускає першу збірку та розгортання. Застосунок стає доступним за адресою <https://fc-web-portal.vercel.app>. Три Cron Jobs, визначені у файлі `vercel.json`, реєструються

платформою автоматично після першого розгортання та виконуються за налаштованим розкладом: оновлення турнірних таблиць чотири дні на тиждень, синхронізація матчів щопонеділка та синхронізація деталей матчів через 30 хвилин після попереднього завдання. Через обмеження безкоштовного тарифу Vercel Hobby вкладка Cron Jobs у розділі Observability відображає лише завдання, що виконувались протягом останніх 12 годин. Активне завдання `/api/cron/update-standings`, що спрацювало у поточний день (рис. 3.4).

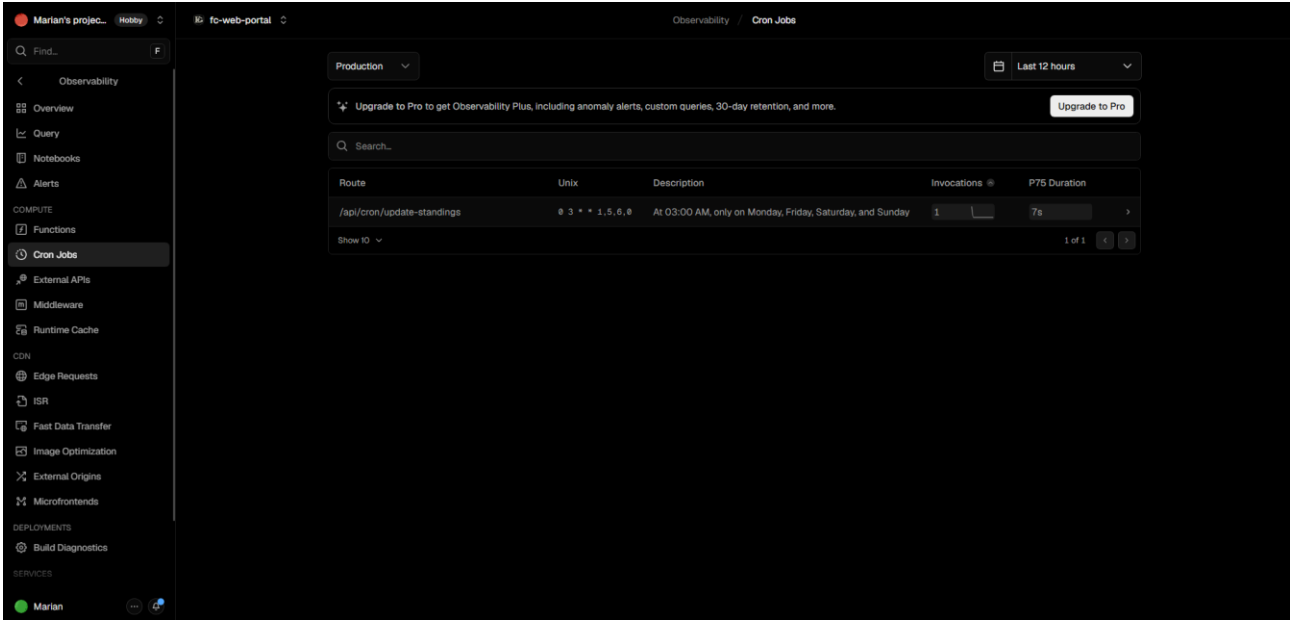


Рис. 3.4 – Активні Cron Jobs у дашборді Vercel

Після першого розгортання перший адміністратор системи створюється вручну – через форму реєстрації на сторінці `/register`, після чого у консолі Neon SQL Editor виконується SQL-запит для надання облікового запису ролі ADMIN.

Подальші оновлення поширюються автоматично завдяки налаштованому CI/CD пайплайну, який було детально розглянуто раніше. Після успішного проходження автоматичних перевірок у GitLab та дзеркалювання коду [4] (рис. 3.5), Vercel відстежує ці зміни й ініціює нове розгортання, включаючи генерацію Prisma-клієнта та застосування змін схеми бази даних. Увесь процес від відправки коду до появи змін у production-середовищі займає близько двох хвилин, що підтверджується журналом розгортань (рис. 3.6).

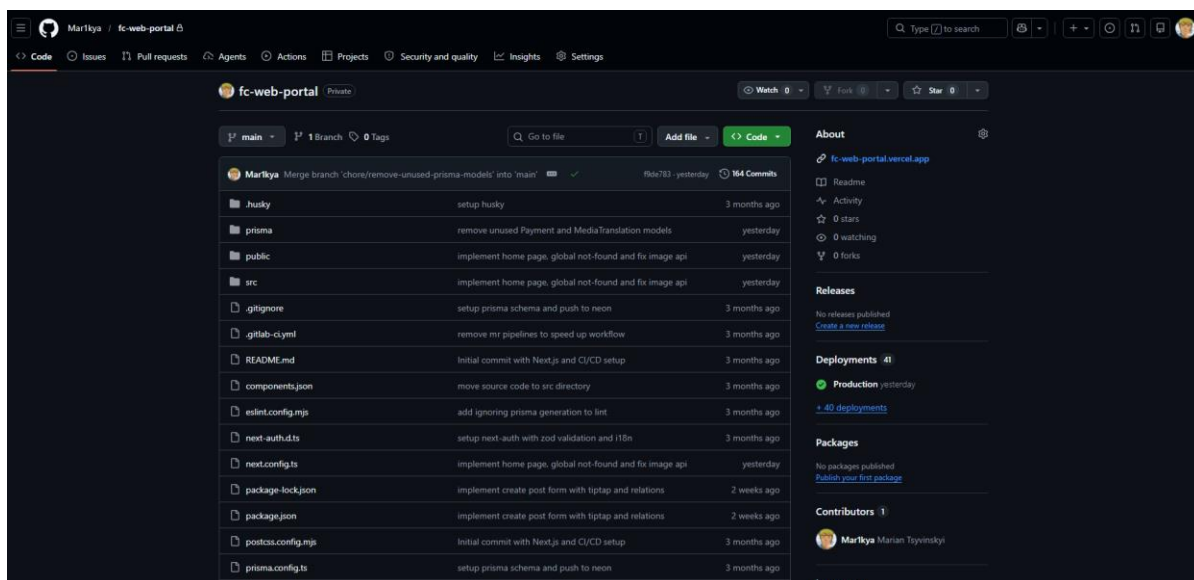


Рис. 3.5 – Приватний GitHub-репозиторій

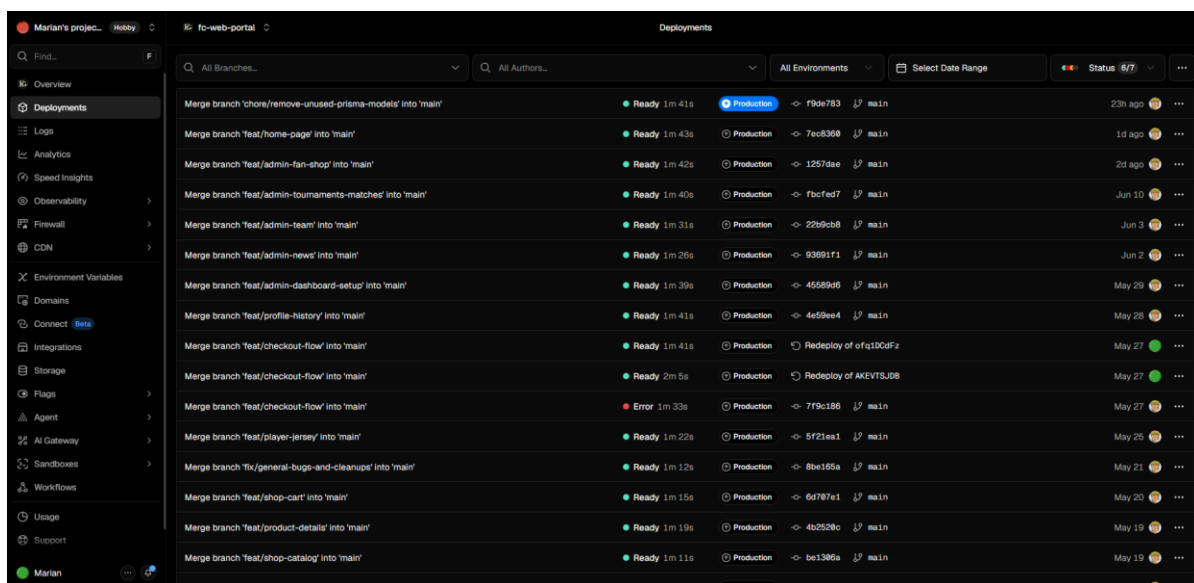


Рис. 3.6 – Журнал розгортань у дашборді Vercel зі статусами Ready

3.2 Структура інтерфейсу вебпорталу футбольного клубу з інтегрованим фаншопом

Інтерфейс порталу організований через чотири незалежні layout-каркаси, що відповідають route groups Next.js. Публічна частина (main) та фаншоп (shop) використовують спільний layout із навігаційною панеллю угорі, спонсорською панеллю та футером знизу. Навігаційна панель містить логотип клубу, головне меню з спадним підменю – Новини, Команда, Матчі, Магазин, Клуб – іконку

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		47

кошика з лічильником товарів, аватар авторизованого користувача та перемикач мови між українською та англійською. Спонсорська панель відображає логотипи партнерів клубу. Футер організований у чотири колонки: логотип із коротким описом порталу, навігація по розділу «Клуб», навігація по розділу «Команда» та контактна інформація з посиланнями на соціальні мережі. Зона автентифікації (auth) має мінімалістичний layout без навігації та футера – лише темний фон. Адміністративна панель (admin) має власний layout з темним сайдбаром та логотипом клубу. Кольорова схема базується на темно-зелених відтінках у форматі OKLCH з єдиною темною темою без перемикача. Акцентний смарагдовий колір використовується для інтерактивних елементів, бейджів та виділення важливих дій. Типографіка побудована на шрифті Geist із жирними заголовками у верхньому регістрі.

Головна сторінка є точкою входу до порталу і агрегує ключовий контент з усіх модулів системи. Него-секція займає верхню частину сторінки та відображає найближчий запланований або останній зіграний матч – логотипи команд, дату, час та стадіон, а також два СТА-елементи для переходу до матч-центру або розкладу. Нижче послідовно розміщені секції: матчі з виділеними попереднім, наступним та через один, три останні публікації, турнірні таблиці активного сезону та рядок товарів фаншопу (рис. 3.7).

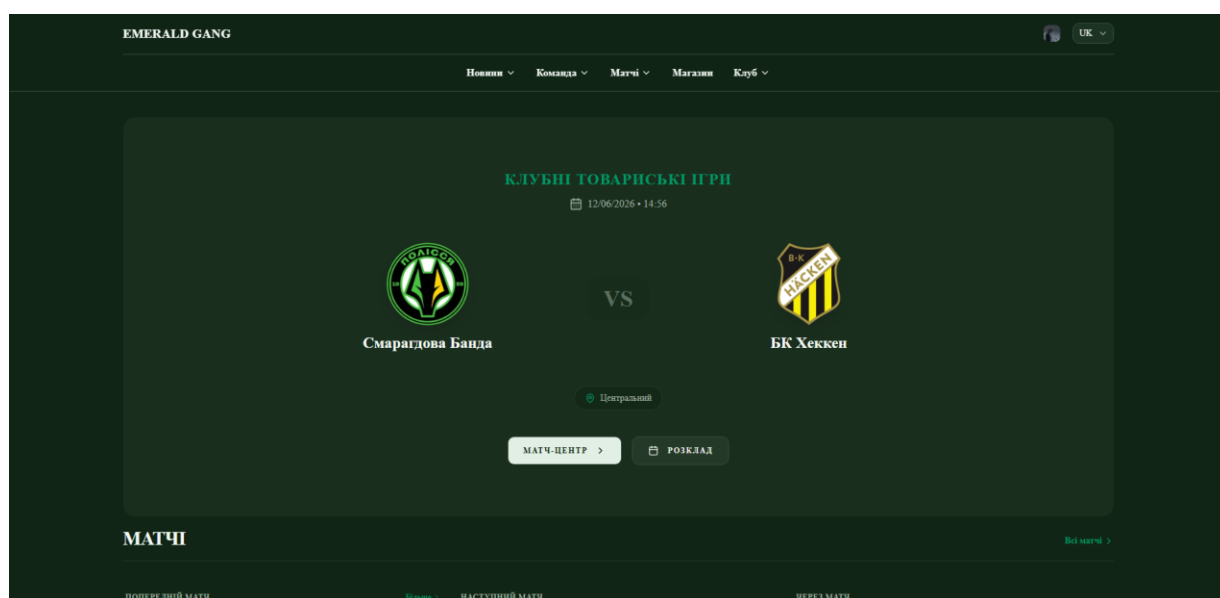


Рис. 3.7 – Головна сторінка вебпорталу

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		48

Сторінка новин відображає публікації у вигляді адаптивної чотириколонкової сітки карток із фільтрацією за типом контенту – Всі новини, Основна команда, U-19, Академія, Новини, Інтерв'ю та Офіційно. Бічна панель містить інтерактивний календар публікацій для навігації за датою та компактну турнірну таблицю. Кожна картка відображає обкладинку або плейсхолдер, тип публікації, контекст команди, заголовок та дату (рис. 3.8).

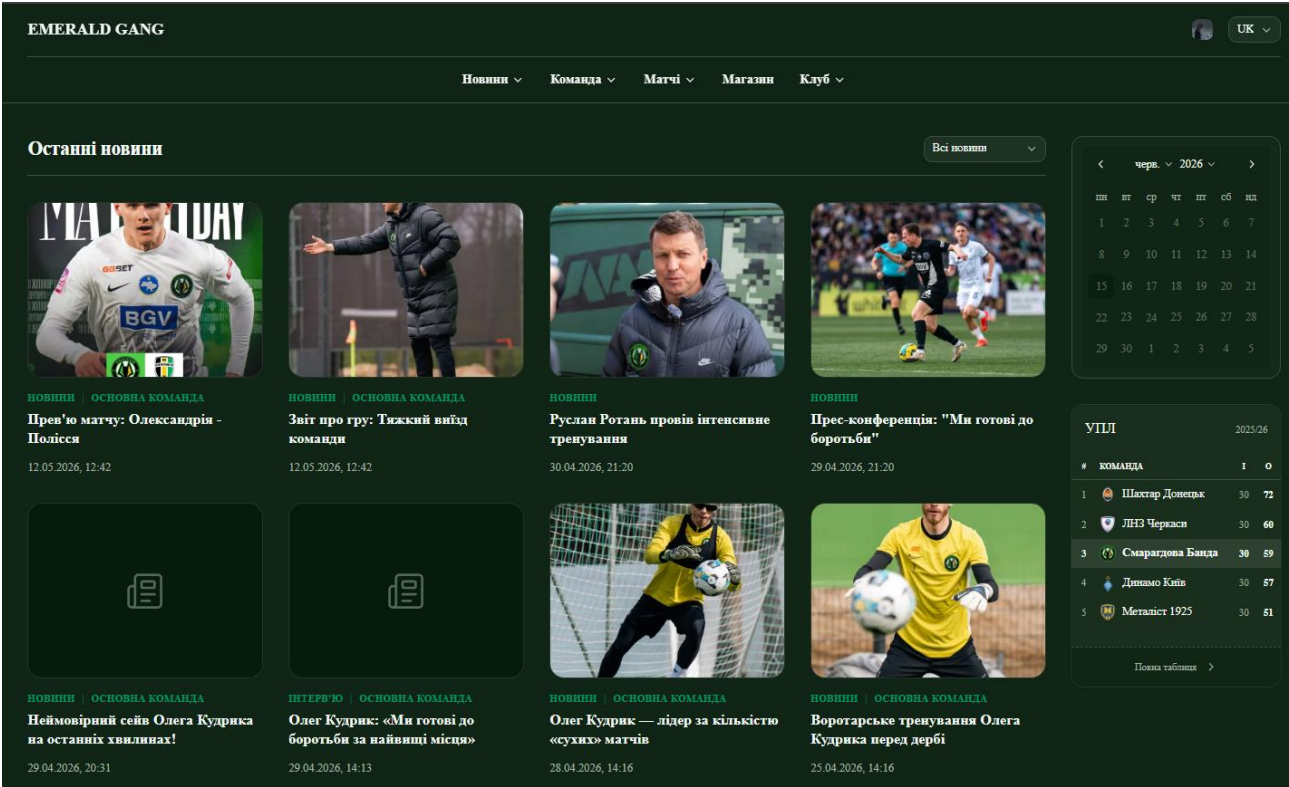


Рис. 3.8 – Сторінка новин із сіткою публікацій та бічною панеллю

Сторінка окремої публікації відображає повний HTML-вміст статті, відрендерений із редактора Tiptap та очищений через sanitize-html. Вміст стилізований через плагін @tailwindcss/typography для правильного відображення заголовків, абзаців та зображень. Над текстом відображаються мітки типу публікації та контексту команди, дата і час публікації. Головне зображення статті виводиться повною шириною контентної зони одразу під заголовком. Права колонка зберігає бічну панель із календарем подій та турнірною таблицею УПЛ із актуальними позиціями команд. Знизу виводяться три пов'язані публікації у вигляді карток із обкладинкою, категорією та датою для продовження перегляду (рис. 3.9).



Рис. 3.9 – Сторінка окремої публікації з вмістом та рекомендаціями

Сторінка команди відображає повний склад клубу із фільтрацією за позицією через вкладки: всі, воротарі, захисники, півзахисники, нападники та тренерський штаб. Гравці автоматично групуються за позицією з відповідними заголовками секцій. Стан активної вкладки зберігається в URL-параметрі, що дозволяє ділитися посиланням на конкретну позицію (рис. 3.10).

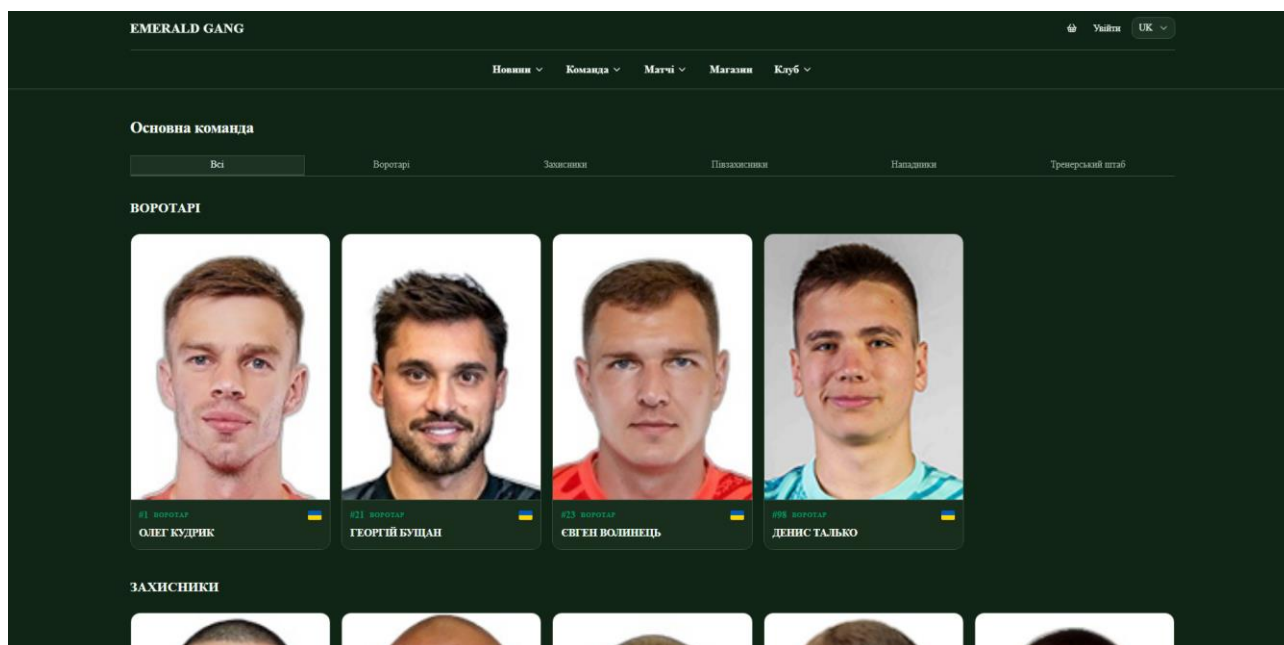


Рис. 3.10 – Сторінка команди з фільтрацією за позицією

Профіль гравця розпочинається з hero-блоку, що займає повну ширину сторінки. Ліва частина містить ім'я з номером, амплуа, громадянство, дату народження, зріст та вагу. Одразу під hero-блоком розміщено рядок із трьома ключовими метриками гібридної статистики – для воротаря це матчі, сухі матчі та пропущені голи; для польових – матчі, голи та асисті. При наявності пов'язаних товарів з'являється кнопка переходу до купівлі іменної форми. Нижче розміщені вкладки біографії, новин та медіагалереї (рис. 3.11).

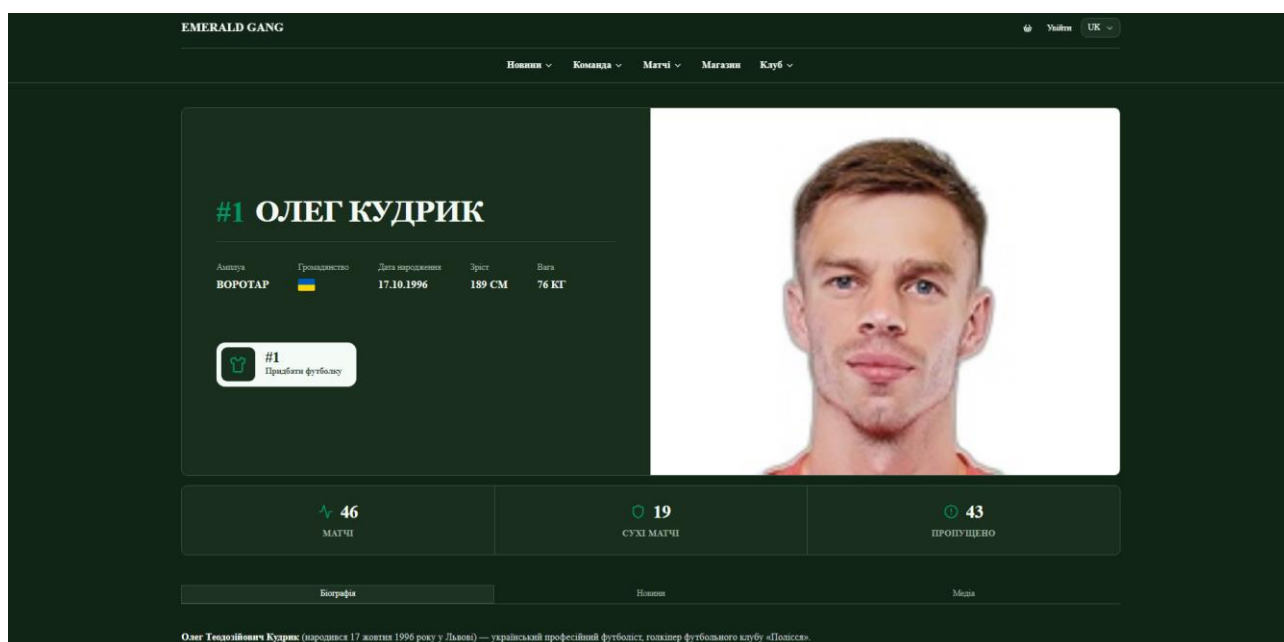


Рис. 3.11 – Профіль гравця зі статистикою та кнопкою купівлі форми

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		51

Профіль тренера має аналогічну структуру без блоку статистики – його блок із посадою, громадянством та датою народження, а також вкладки з біографією, новинами та медіа (рис. 3.12).

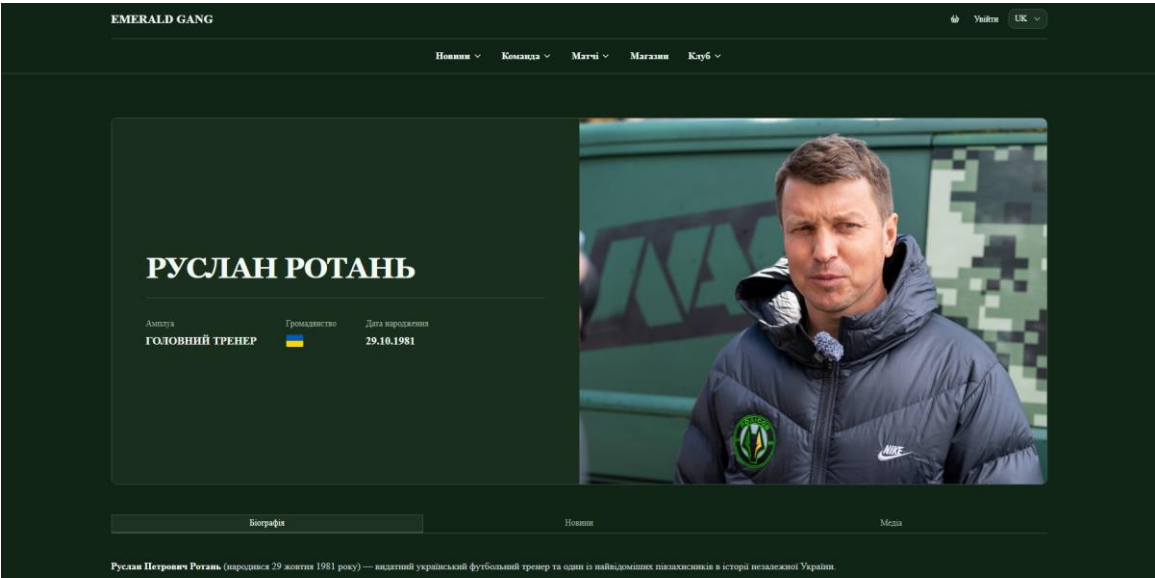


Рис. 3.12 – Сторінка профілю тренера

Сторінка матчів відображає три виділені матчі у горизонтальному блоці – попередній з результатом, наступний та через один – кожен з логотипами команд, датою та посиланням до матч-центру. Нижче розміщено хронологічний список усіх матчів обраного сезону, згрупований по місяцях з відповідними заголовками. Фільтрація за сезоном виконується через спадний список у верхньому правому куті (рис. 3.13).

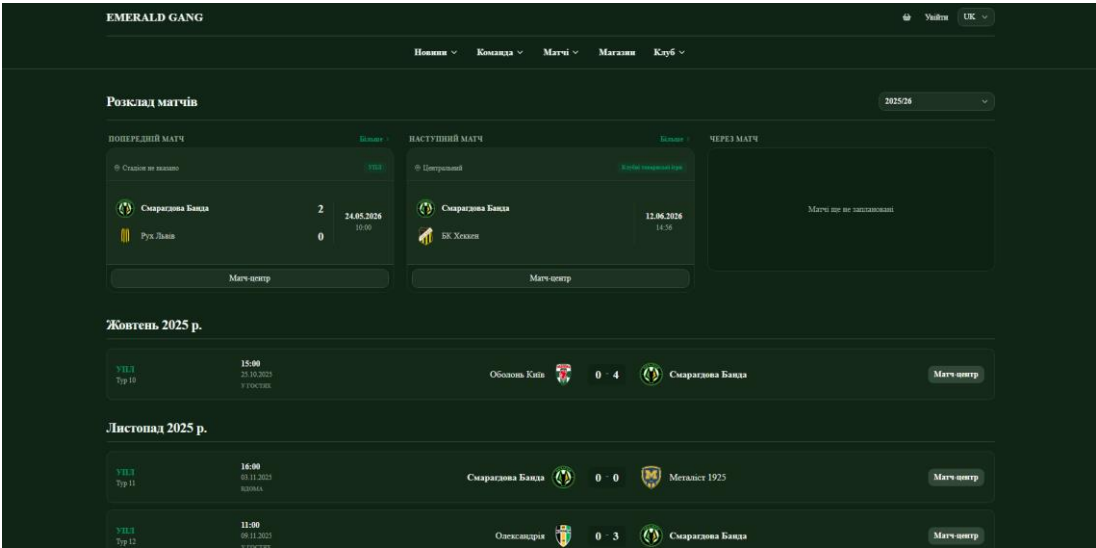


Рис. 3.13 – Розклад матчів із виділеними матчами та групуванням по місяцях

Сторінка окремого матчу відкривається з великим hero-блоком. У ньому вказано турнір з туром, дату та час, логотипи та назви команд із рахунком по центру, імена тренерів та стадіон. Для завершених матчів відображаються результативні події – голи з хвилинами та іменами авторів. Нижче розміщені вкладки зі складами команд, новинами, фотогалереєю та відео. Склад власної команди формується з таблиці MatchLineup. Склад суперника отримується з поля opponentLineup у форматі JSON (рис. 3.14).

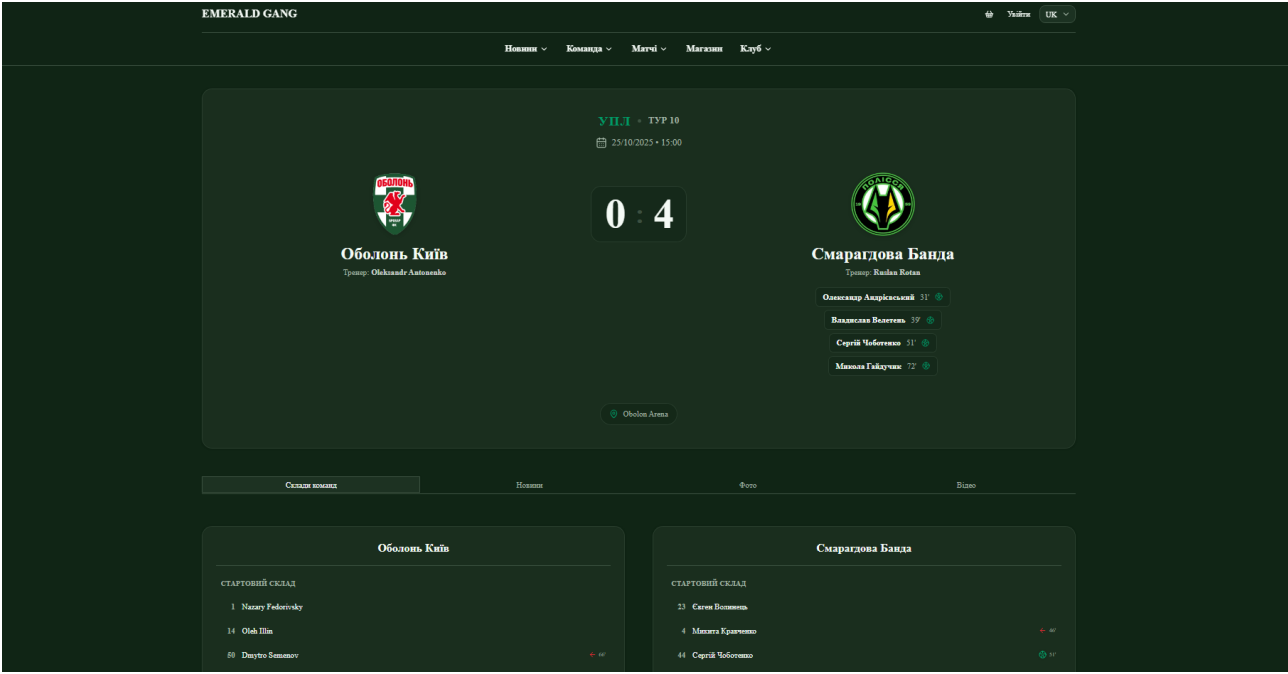


Рис. 3.14 – Сторінка матчу з рахунком, подіями та складами команд

Сторінка турнірної таблиці відображає повну таблицю з логотипами команд. Рядок власної команди в ній виділено фоном для миттєвого візуального орієнтування. Колонки таблиці містять місце, назву команди, кількість зіграних ігор, перемоги, нічий, поразки, різницю забитих і пропущених голів та загальну кількість очок. Логотипи команд підвантажуються із зовнішнього джерела Sofascore та відображаються поруч із назвою для кращої ідентифікації. Таблиця формується на основі даних, синхронізованих із Sofascore через автоматичний Cron-алгоритм. Це забезпечує актуальність показників після кожного туру. Назви команд у таблиці відображаються відповідно до обраної мови інтерфейсу завдяки словнику перекладів TeamDictionary. Фільтрація за сезоном доступна через спадний список у правому верхньому куті сторінки (рис. 3.15).

EMERALD GANG						
УПЛ						
2025/26						
#	Команда	І	В	Н	П	О
1	Шахтар Донецька	30	22	6	2	71-21
2	ДНЗ Черкаси	30	18	6	6	39-17
3	Спартаківська Багача	30	18	5	7	51-21
4	Динамо Київ	30	17	6	7	66-36
5	Металіст 1925	30	13	12	5	36-19
6	Колос Ковалівка	30	13	10	7	30-25
7	Кривбас Кривий Ріг	30	13	9	8	53-46
8	Зоря Луганськ	30	12	10	8	42-36
9	Карпати Львів	30	10	11	9	40-31
10	Елісентур	30	8	8	14	36-45
11	Верес Рашів	30	7	10	13	26-40
12	Оболонь Київ	30	7	10	13	25-49
13	Куріпка	30	7	7	16	32-48
14	Рос-Дніп	30	6	3	21	30-51

Рис. 3.15 – Турнірна таблиця УПЛ сезону 2025/26

Сторінки входу та реєстрації навмисно відокремлені від основного layout-каркасу і відображаються по центру темного фону без навігаційного меню. Сторінка входу містить кнопку «Увійти через Google» як пріоритетний варіант та форму з полями електронної пошти і пароля як альтернативу. Валідація виконується через Zod – при некоректних даних повертається узагальнена помилка без розкриття деталей (рис. 3.16).

Рис. 3.16 – Сторінка входу з підтримкою Google OAuth

Сторінка реєстрації містить форму з іменем та прізвищем, поштою та паролем, а також кнопку реєстрації через Google. Після успішної реєстрації

виконується автоматичне перенаправлення на сторінку входу. Обидві сторінки містять посилання для переходу між собою (рис. 3.17).

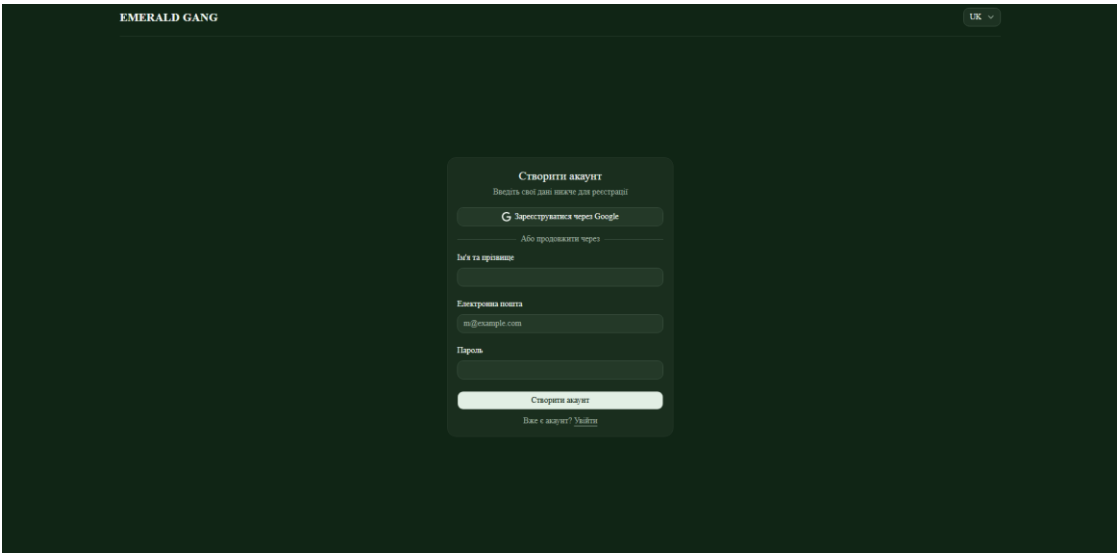


Рис. 3.17 – Сторінка реєстрації

Фаншоп має власний layout з додатковим підменю категорій безпосередньо під основною навігацією: вітрина, ігрова форма, одяг, аксесуари, футболки гравців та розпродаж – виділений червоним кольором. Вітрина фаншопу містить hero-банер із колажем гравців у клубній формі та закликом до дії, карусель хітів продажу, новинки та блок акційних товарів зі знижками (рис. 3.18).

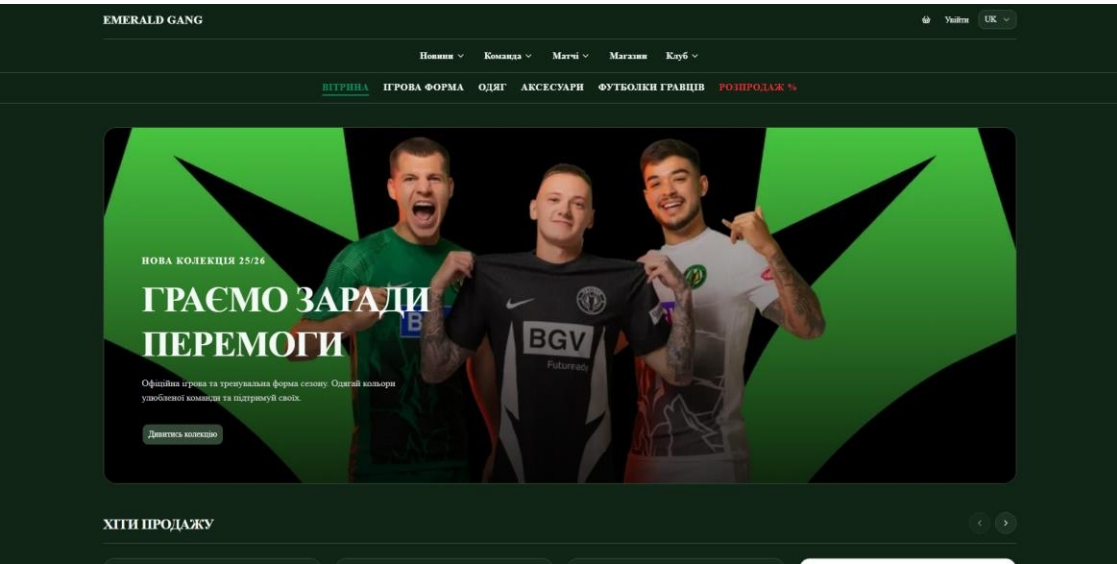


Рис. 3.18 – Вітрина фаншопу

Каталог підтримує багатокритеріальну фільтрацію: діапазон ціни через двоточковий слайдер із ручним введенням граничних значень, цільову аудиторію, тип одягу, доступні розміри та колір. Усі активні фільтри

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				55
Змн.	Арк.	№ докум.	Підпис	Дата		

синхронізуються з URL-параметрами сторінки, що дозволяє зберігати та ділитися посиланням із вже застосованими умовами пошуку. Навігація між категоріями реалізована через підменю у верхній частині сторінки з виділенням активної категорії. Сортування доступне за датою додавання та ціною через спадний список у правому верхньому куті. Картка товару відображає статус наявності, бейджи «Новинка» та «Топ» для акцентування уваги, а також знижену ціну поруч із закресленою оригінальною при активній акції. Кнопка швидкого додавання до кошика доступна безпосередньо з картки без переходу на сторінку товару (рис. 3.19).

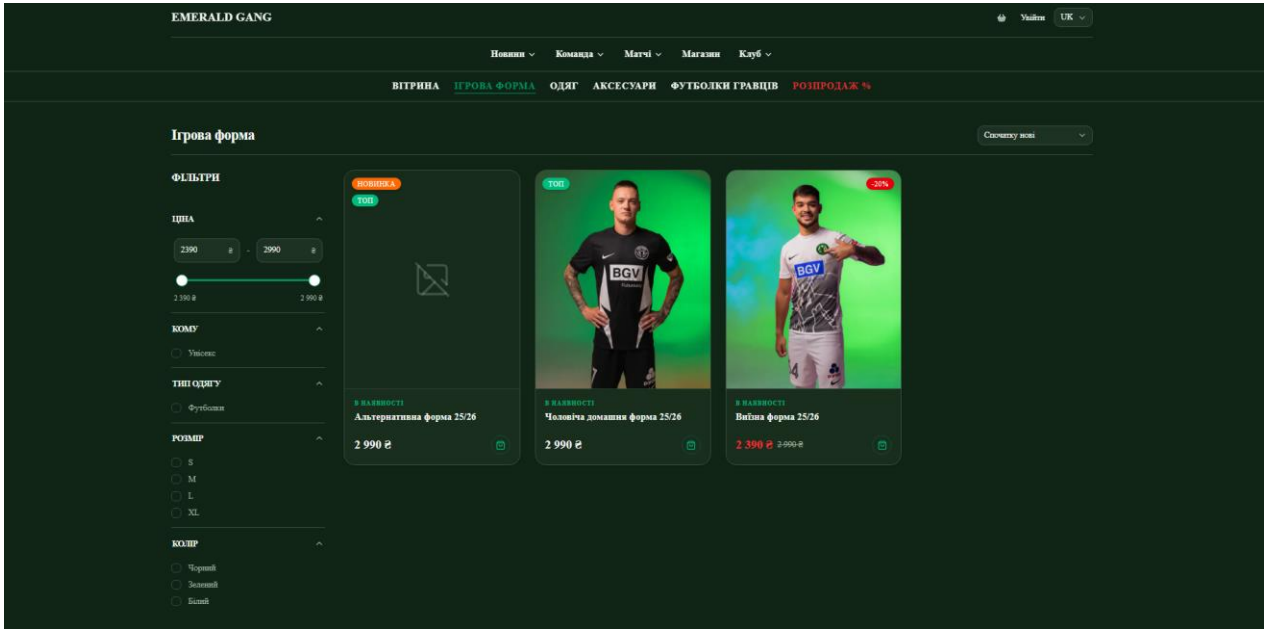


Рис. 3.19 – Каталог фаншопу з панеллю фільтрів

Сторінка товару має двоколонкову структуру: ліворуч – галерея зображень з мініатюрами для перемикання, праворуч – назва, статус, артикул, ціна, вибір розміру через кнопки-тоглери та вибір кількості через спадний список. При виборі розміру, якого немає в наявності, кнопка «В кошик» блокується. Кількість одного товару одного розміру обмежена максимум 10 одиницями за одне замовлення – це обмеження діє як на рівні інтерфейсу, так і на рівні логіки кошика у Zustand. Нижче виводиться опис товару та карусель схожих товарів (рис. 3.20).

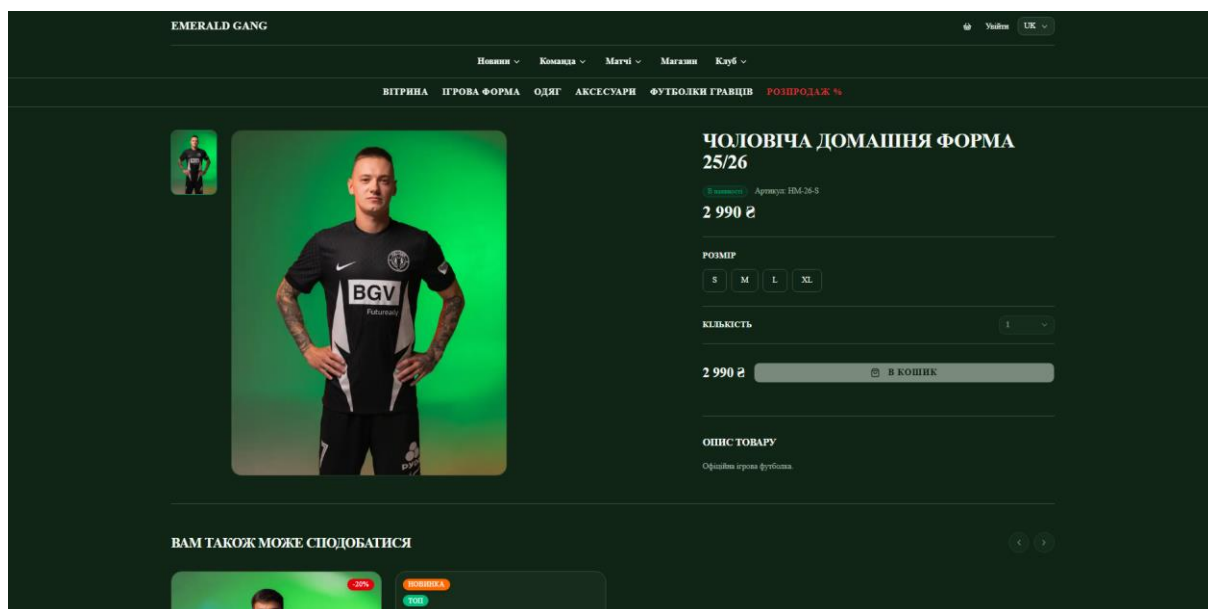


Рис. 3.20 – Сторінка товару з вибором розміру та кількості

Розділ «Футболки гравців» відображає всіх гравців основного складу у вигляді сітки портретних карток. Після вибору гравця відкривається сторінка з перемикачем типу форми – домашня, виїзна або альтернативна – та автоматичною фільтрацією товарів, прив'язаних до цього гравця. Це реалізує механізм контекстних продажів через зв'язок `relatedPlayers` у схемі бази даних (рис. 3.21, 3.22).

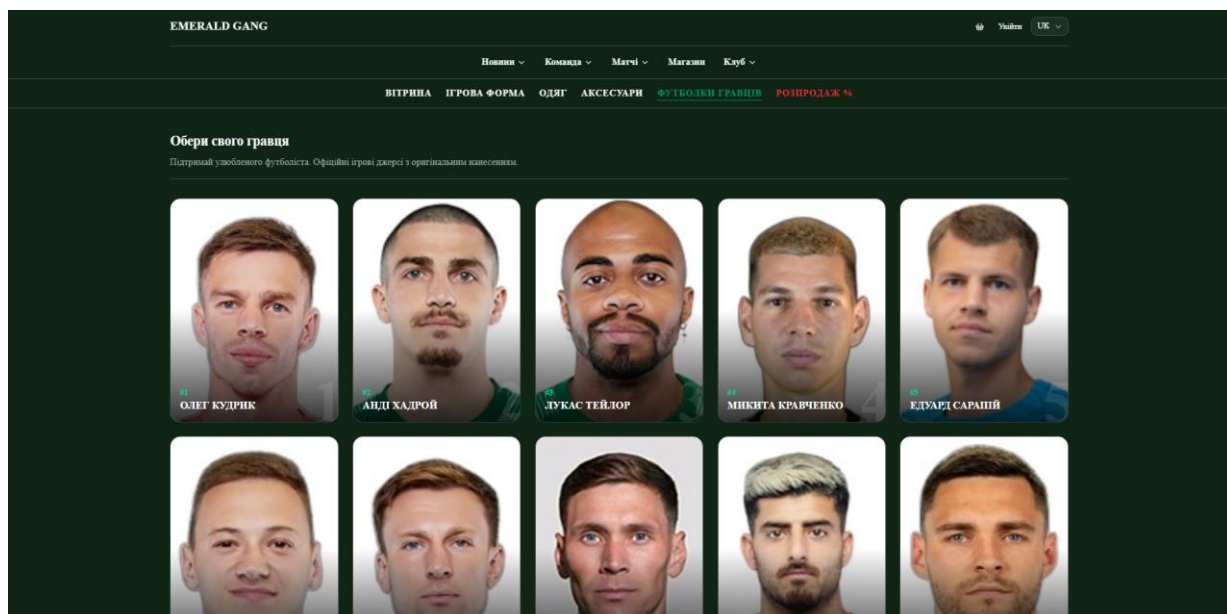


Рис. 3.21 – Сторінка вибору гравця для замовлення іменної форми

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		57

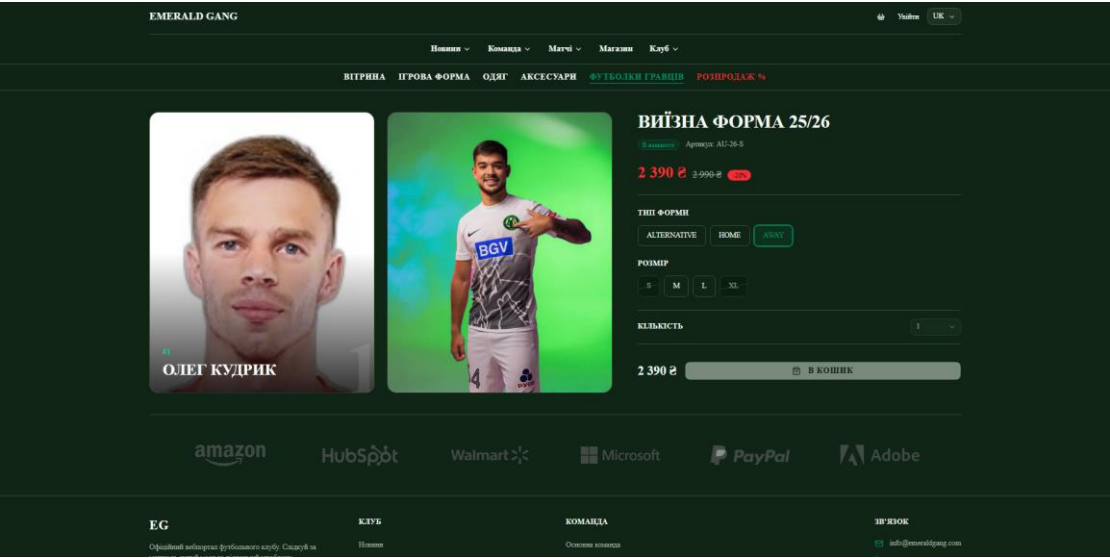


Рис. 3.22 – Сторінка форми конкретного гравця з вибором типу та розміру

Кошик реалізований у два рівні взаємодії. При натисканні на іконку кошика у шапці відкривається спадне міні-меню з переліком доданих товарів, загальною сумою та двома кнопками – «Перейти в кошик» та «Оформити замовлення». Це дозволяє перейти одразу до оформлення без додаткових кліків. Повна сторінка кошика відображає детальний перелік товарів з управління кількістю через кнопки «-» та «+» з перевіркою на доступний залишок. У правій колонці відображається підсумок із загальною сумою та двома кнопками – оформлення замовлення та продовження покупок. Стан кошика зберігається у localStorage через Zustand middleware persist, тому не зникає між сесіями (рис. 3.23).

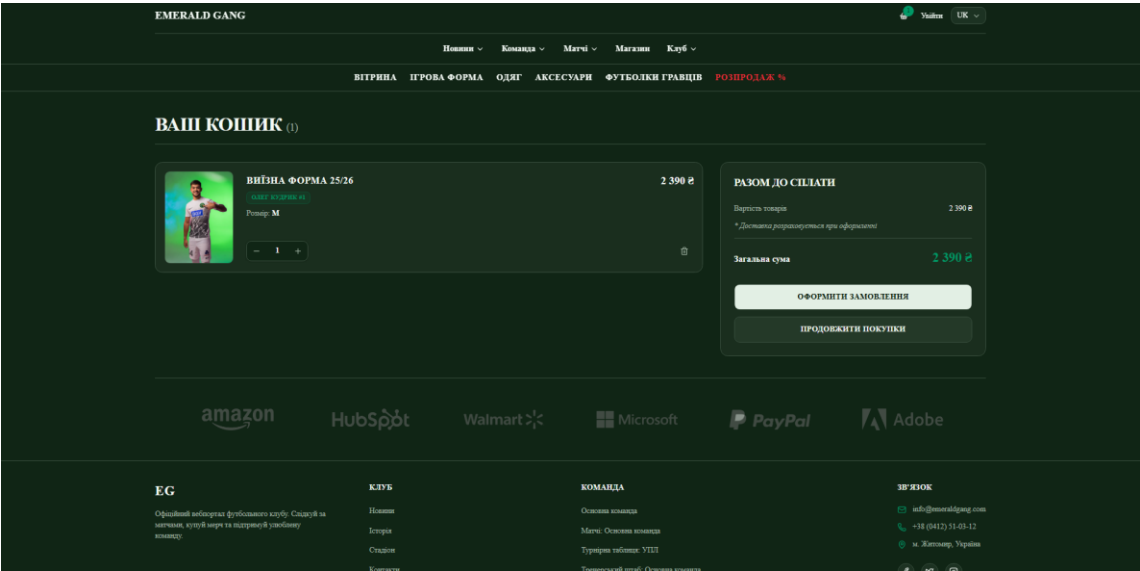


Рис. 3.23 – Сторінка кошика із переліком товарів

EMERALD GANG

👤 Увійти

UK

Новини

Команда

Матчі

Магазин

Клуб

ВІТРИНАІГРОВА ФОРМАОДЯГАКСЕСУАРИФУТБОЛКИ ГРАВЦІВРОЗПРОДАЖ %

ОФОРМЛЕННЯ ЗАМОВЛЕННЯ

1

ДАНІ ОТРИМУВАЧА

Ім'я

Май'я

Прізвище

Целінський

Електронна пошта

vt221_bumv@student.ztu.edu.ua

Номер телефону

0972248245

2

СПОСІБ ДОСТАВКИ

ОБЕРІТЬ СПОСІБ ДОСТАВКИ

📦 Відділення

📮 Поштамат

🚚 Кур'єр

Місто / Населений пункт

Житомир

Номер відділення

1

3

МЕТОД ОПЛАТИ

ОБЕРІТЬ МЕТОД ОПЛАТИ

💳 Карткою онлайн

📄 Готівкою при отриманні

ВАШЕ ЗАМОВЛЕННЯ



Військова форма 25/26

Розмір: М • Колір: 1

0.125 КІЛОГРАМ • 1

2 390 ₴

Сума товарів

2 390 ₴

* Доставка за тарифом прискіпки

Всього

2 390 ₴

🛒 ОФОРМИТИ ЗАМОВЛЕННЯ

При виборі карткової оплати користувач перенаправляється на захищену сторінку Stripe Checkout з підтримкою Visa, Mastercard, Apple Pay та Google Pay. Платіжні дані картки ніколи не проходять через сервери порталу (рис. 3.25).

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				59
Змн.	Арк.	№ докум.	Підпис	Дата		

Рис. 3.25 – Сторінка оплати Stripe Checkout

Після оформлення відкривається сторінка замовлення з унікальним коротким номером, статусами доставки та оплати у вигляді кольорових бейджів, даними отримувача, адресою та методом оплати. Для оплачених замовлень статус оплати відображається зеленим бейджем «Оплачено» (рис. 3.26).

Рис. 3.26 – Сторінка оплаченого замовлення

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				60
Змн.	Арк.	№ докум.	Підпис	Дата		

Для замовлень з карткою, що очікують підтвердження, відображається таймер зворотного відліку 30 хвилин та кнопка «Оплатити зараз» для повторної спроби. Для неавторизованих гостей сторінка пропонує увійти через Google для автоматичного прив'язування – персональні дані при цьому маскуються (рис. 3.27).

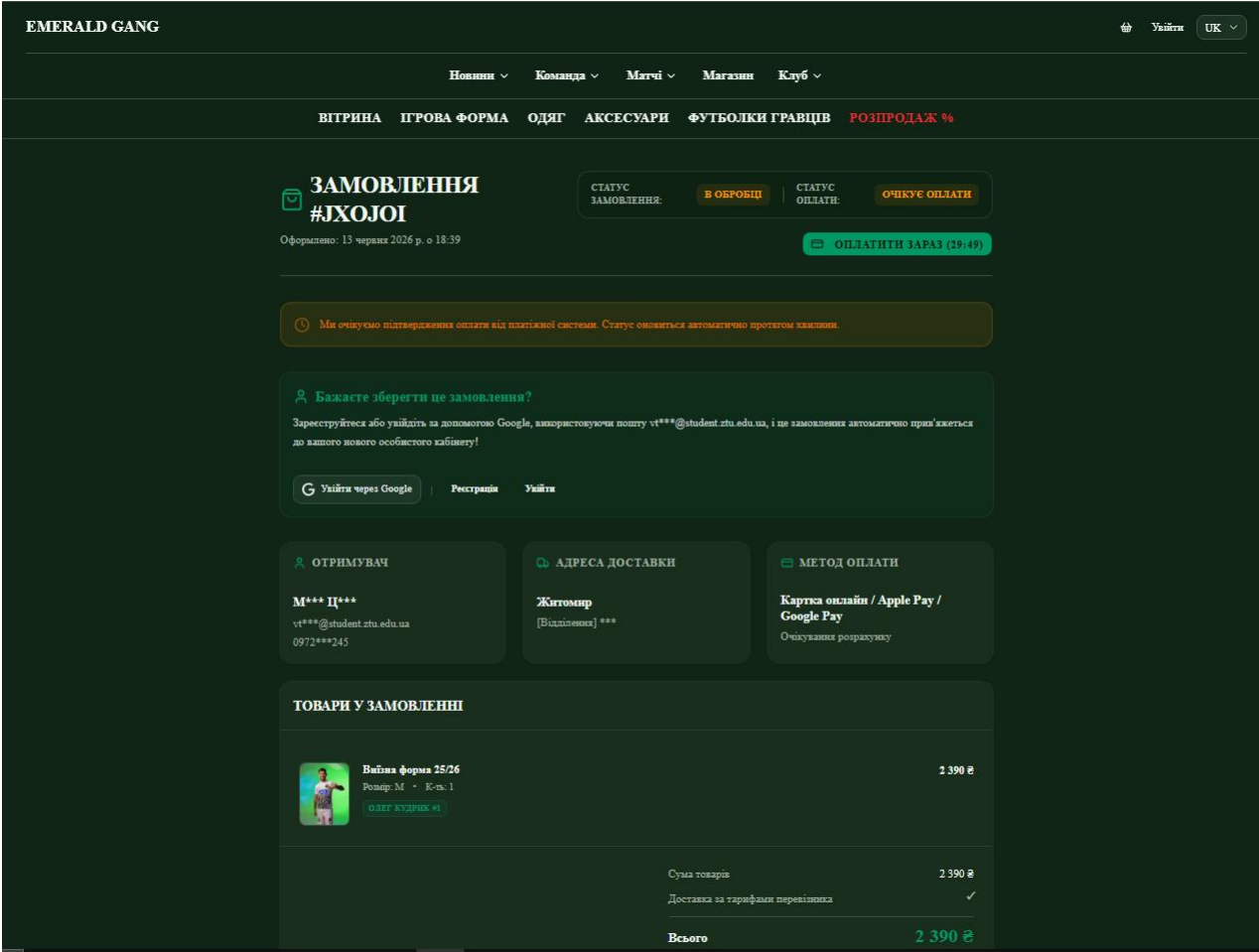


Рис. 3.27 – Сторінка замовлення для гостя з таймером та пропозицією авторизації

Особистий кабінет має власний layout з бічним меню – два розділи та кнопка виходу. Сторінка профілю дозволяє змінити ім'я та прізвище, завантажити фото через UploadThing та змінити пароль через форму з трьома полями. Електронна пошта відображається як незмінне поле для ідентифікації (рис. 3.28).

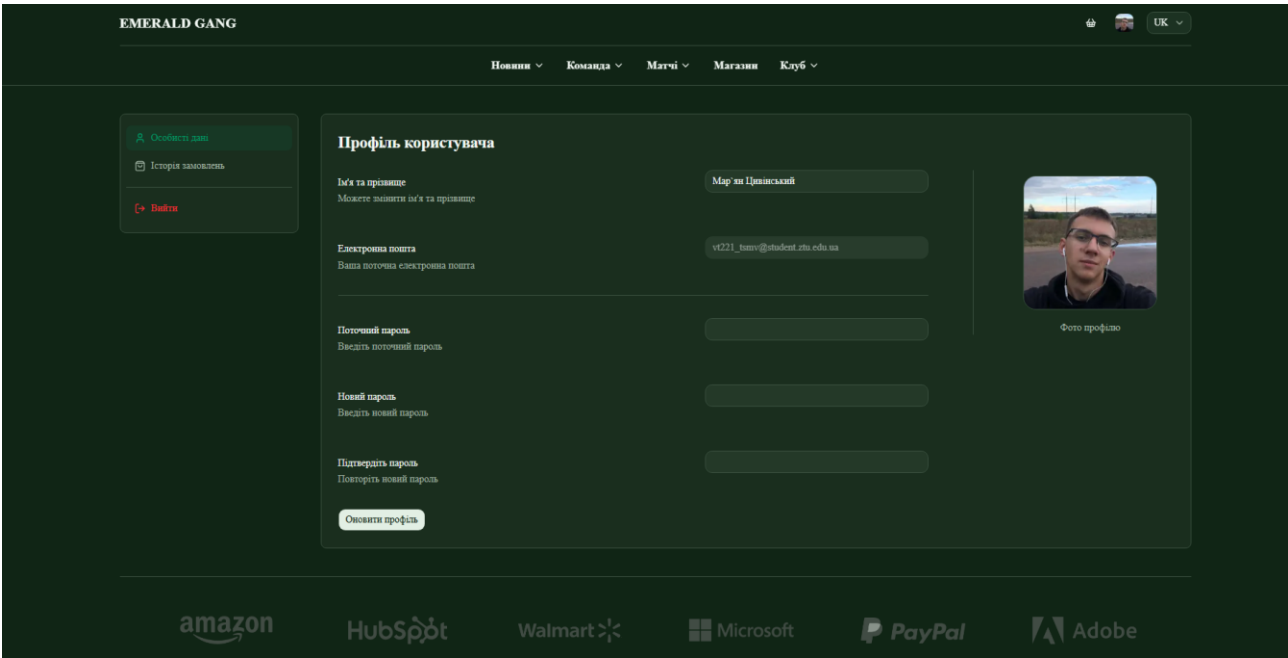


Рис. 3.28 – Особистий кабінет: профіль користувача

Сторінка історії замовлень відображає всі замовлення з фільтрацією за статусом через вкладки та сортуванням. Кожна картка замовлення показує номер, дату, мініатюри товарів, суму та статуси з кольоровими бейджами (рис. 3.29).

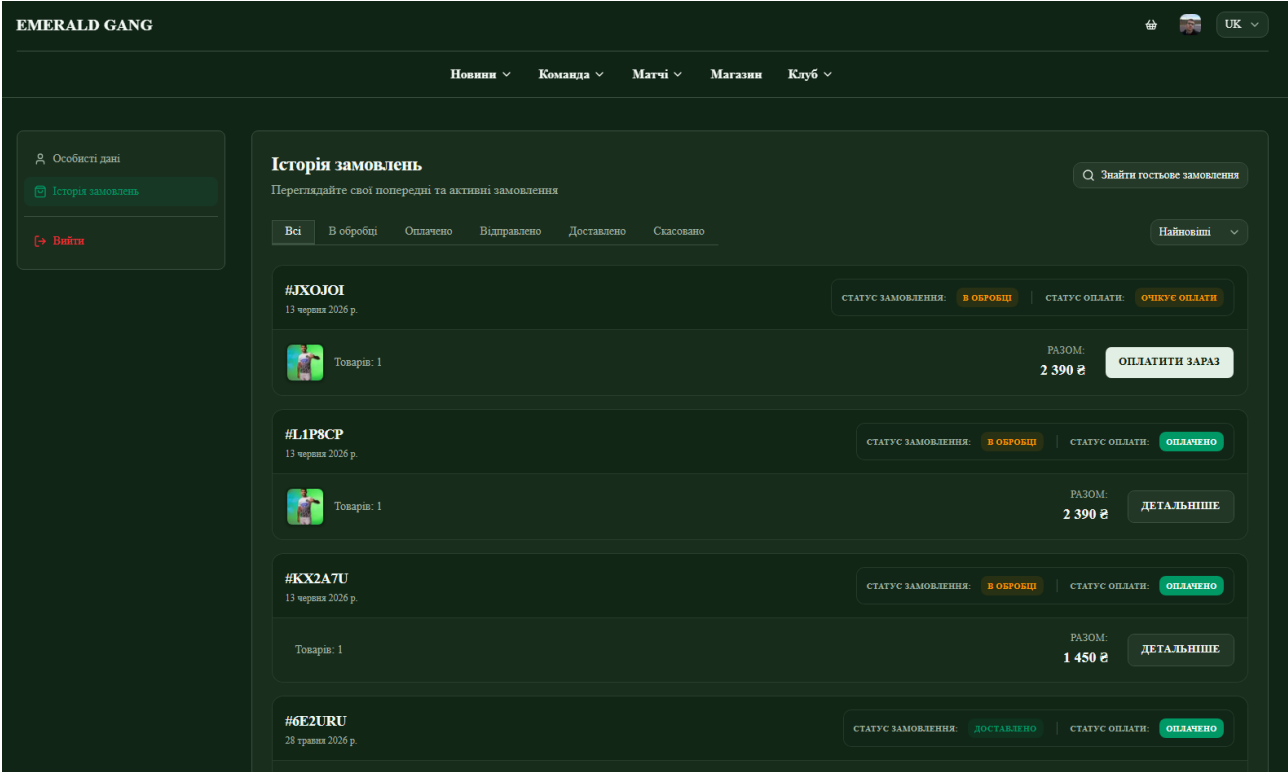


Рис. 3.29 – Особистий кабінет: історія замовлень

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		62

Кнопка «Знайти гостьове замовлення» відкриває модальне вікно з двома полями номер замовлення та номер телефону – для прив'язки замовлення, оформленого до реєстрації (рис. 3.30).

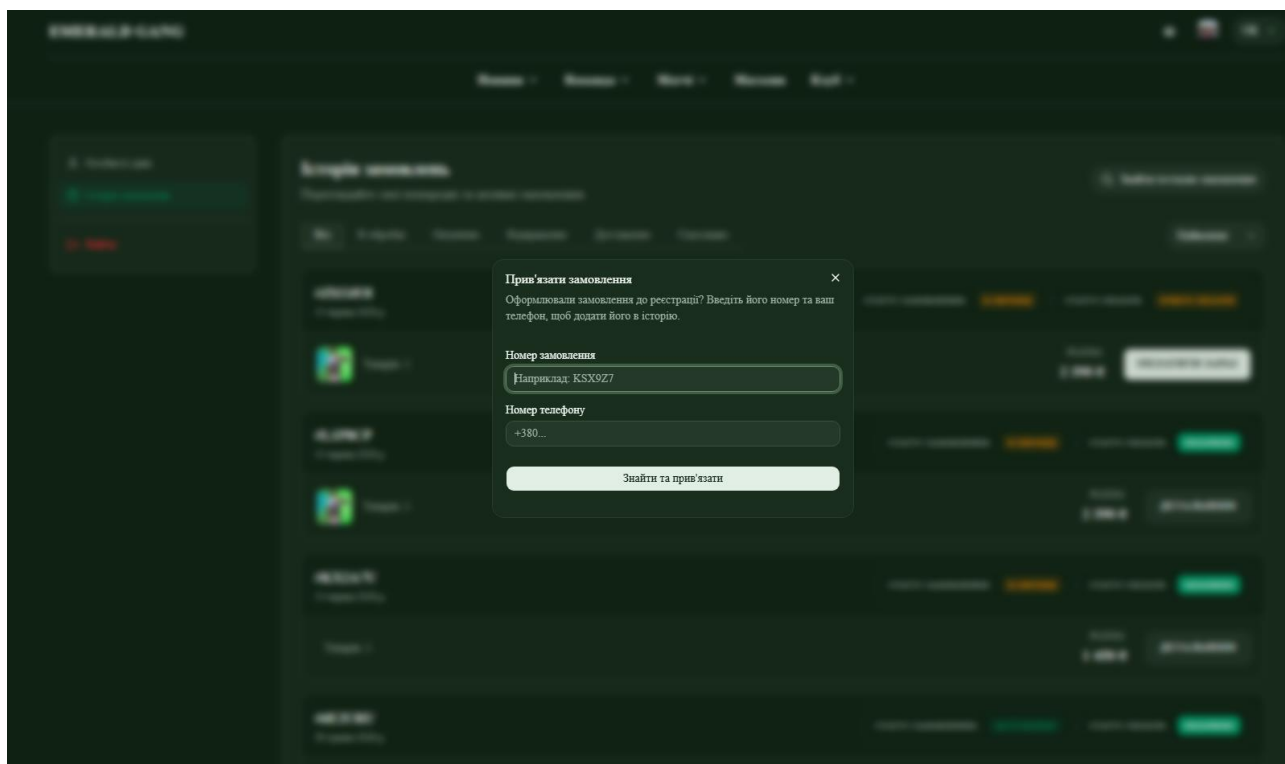


Рис. 3.30 – Модальне вікно прив'язки гостьового замовлення

Адміністративна панель доступна лише користувачам з роллю ADMIN і має власний layout з темним сайдбаром та логотипом клубу. Навігація містить шість пунктів: Дашборд, Новини, Команда, Турніри та Матчі, Фаншоп та Галерея. Перехід до адмін-панелі для неадміністраторів блокується на рівні middleware з перенаправленням до профілю.

Дашборд відображає чотири інформаційні віджети: нові замовлення зі статусом «В обробці», загальний дохід з оплачених замовлень, дані наступного матчу клубу та кількість активних публікацій. У правій частині розміщені попередження про залишки товарів менше 5 одиниць та список завершених матчів, що очікують синхронізації деталей із кнопкою «Оновити» для кожного (рис. 3.31).

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		63

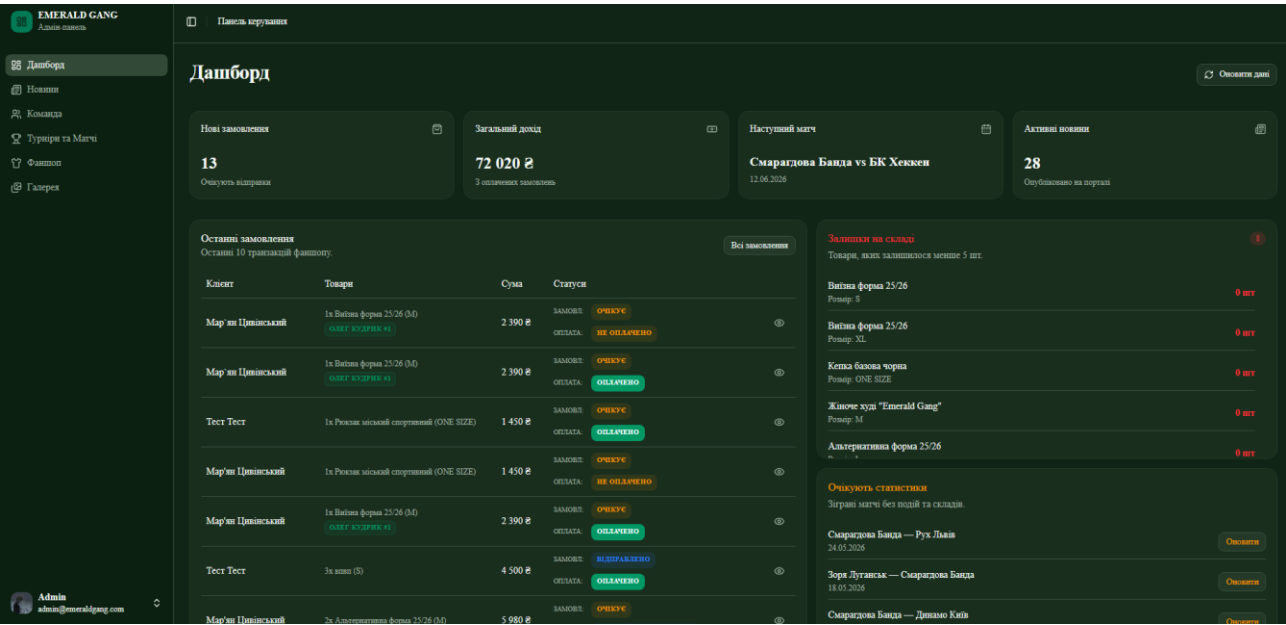


Рис. 3.31 – Дашборд адміністративної панелі

Розділ новин відображає таблицю публікацій на базі TanStack Table із пошуком та фільтрацією за командою, категорією та статусом. Форма створення публікації підтримує мультимовний вміст через вкладки, завантаження медіафайлів через UploadThing з вибором обкладинки та прив'язку до гравців, тренерів і матчів для крос-відображення на відповідних профілях (рис. 3.32).

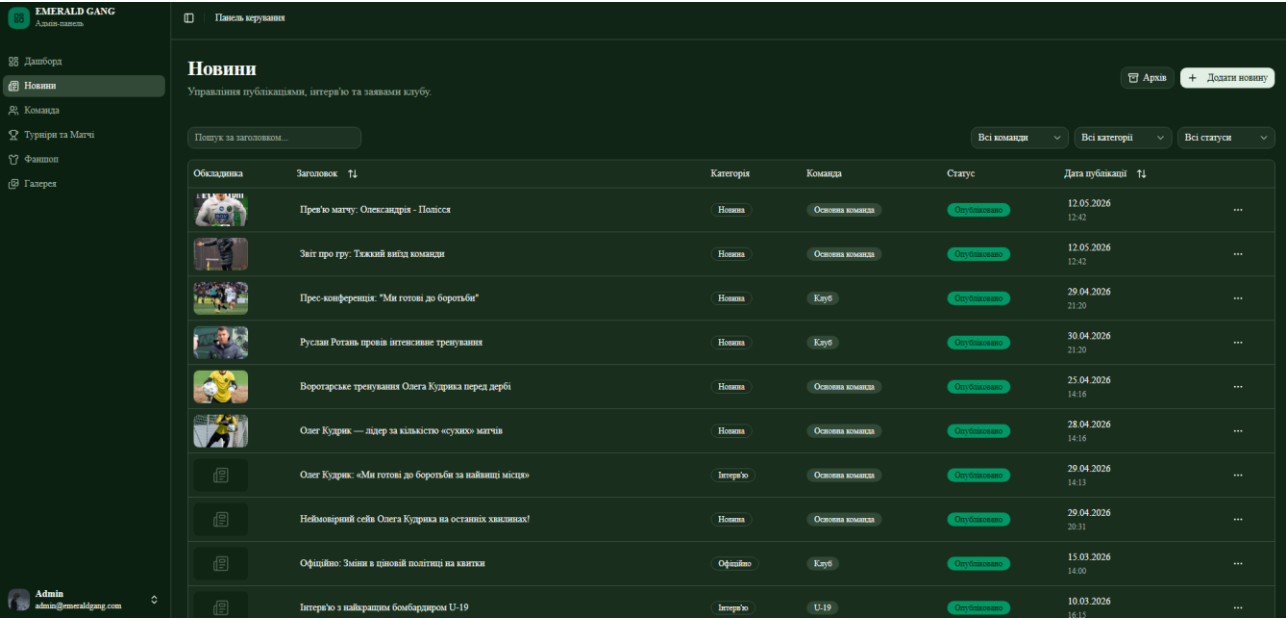


Рис. 3.32 – Адмін-панель: список публікацій

У розділі команди відображається її склад у вигляді таблиці з фотографіями гравців, номерами, позиціями та поточною гібридною статистикою. Кнопка «Синхронізувати» запускає оновлення складу через

інтеграцію з Sofascore – автоматично оновлюються номери, позиції, антропометричні дані та аватари (рис. 3.33).

#	Гравець	Позиція	Статистика
1	Олег Кірич	Воротар	М: 25 СМ: 9 ПГ: 29
2	Аки Хадрой	Захисник	М: 0 Г: 0 А: 0
3	Лукас Теклор	Захисник	М: 0 Г: 0 А: 0
4	Махита Крамченко	Захисник	М: 0 Г: 0 А: 0
5	Едурд Саратій	Захисник	М: 0 Г: 0 А: 0
7	Олександр Назаренко	Півзахисник	М: 0 Г: 0 А: 0
8	Руслан Бабенко	Півзахисник	М: 0 Г: 0 А: 0
9	Олександр Фалішов	Нападник	М: 0 Г: 0 А: 0
10	Томер Йосефі	Півзахисник	М: 0 Г: 0 А: 0

Рис. 3.33 – Адмін-панель: склад команди зі статистикою

Розділ «Турніри та Матчі» організований через шість вкладок: Матчі, Турніри, Сезони, Таблиці, Суперники та Переклад команд. Таблиця матчів відображає дату, пару команд з рахунком, статус та індикатор синхронізації деталей. Форма редагування матчу організована у три вкладки: головне, склад та події (рис. 3.34).

Дата матчу	Матч	Статус	Деталі (Склади/Події)
12.06.2026 17:56	Смарагдова Банда - БК Хевен	Заплановано	---
24.05.2026 13:00	Смарагдова Банда - Рух Львів	Завершено	Очікує скану
18.05.2026 13:00	Зоря Луганськ - Смарагдова Банда	Завершено	Очікує скану
12.05.2026 17:26	Смарагдова Банда - Динамо Київ	Завершено	Дані для ручного матчу
12.05.2026 13:30	ФК Еліцентр Ду... - Смарагдова Банда	Завершено	Синхронізовано
08.05.2026 18:00	Смарагдова Банда - Олександрія	Завершено	Синхронізовано
08.05.2026	Смарагдова Банда - Хевен Сіті	Завершено	Синхронізовано

Рис. 3.34 – Адмін-панель: матч-центр із статусами синхронізації

Розділ фаншопу містить три вкладки – «Замовлення», «Товари» та «Категорії». Таблиця замовлень відображає клієнта з контактними даними, товари з кастомізацією, де ім'я та номер гравця виводяться зеленим бейджем, роздільні статуси доставки та оплати та дату. Кнопка «Архів» переміщує оброблені замовлення без їхнього видалення (рис. 3.35).

Клієнт	Товари	Сума	Статус	Дата
Mar'ni Pivnitskiy +021...@emerald-gang.com	1x Футболка форма 25/26 (M)	2390 ₴	ЗАМОВЛ. ОПЛАЧЕНО ОПЛАТА НЕ ОПЛАЧЕНО	13.06.2026 21:39
Mar'ni Pivnitskiy +021...@emerald-gang.com	1x Футболка форма 25/26 (M)	2390 ₴	ЗАМОВЛ. ОПЛАЧЕНО ОПЛАТА ОПЛАЧЕНО	13.06.2026 21:36
Test Test test@mail.com	1x Рюкзак місткий спортивний (ON...)	1450 ₴	ЗАМОВЛ. ОПЛАЧЕНО ОПЛАТА ОПЛАЧЕНО	13.06.2026 19:05
Mar'ni Pivnitskiy mar'ni041210101@gmail.com	1x Рюкзак місткий спортивний (ON...)	1450 ₴	ЗАМОВЛ. ОПЛАЧЕНО ОПЛАТА НЕ ОПЛАЧЕНО	13.06.2026 18:36
Mar'ni Pivnitskiy mar'ni041210101@gmail.com	1x Футболка форма 25/26 (M)	2390 ₴	ЗАМОВЛ. ОПЛАЧЕНО ОПЛАТА ОПЛАЧЕНО	13.06.2026 18:34
Mar'ni Pivnitskiy mar'ni041210101@gmail.com	2x Чоловіча довгошия футболка 25/26 (L)	5980 ₴	ЗАМОВЛ. ОПЛАЧЕНО ОПЛАТА ОПЛАЧЕНО	29.05.2026 14:31
Mar'ni Pivnitskiy		2000 ₴	ЗАМОВЛ. ОПЛАЧЕНО ОПЛАТА ОПЛАЧЕНО	29.05.2026

Рис. 3.35 – Адмін-панель: управління замовленнями фаншопу

Розділ галереї відображає таблицю фотозвітів з обкладинкою, назвою, кількістю фотографій та назвою прив'язаного матчу. Кожна галерея може бути прив'язана до конкретного матчу або існувати незалежно як окрема публікація клубних подій (рис. 3.36).

Обкладинка	Назва	Фото	Матч	Дата публікації
	Фотозвіт: Олександрія - Полісся	3 фото	Олександрія Банда - Олександрія 08.05.2026 2:1	12.05.2026 12:42

Рис. 3.36 – Адмін-панель: управління галереями

Інтерфейс адаптований для мобільних пристроїв через утилітарні класи Tailwind CSS v4 без окремих CSS-файлів. На мобільному розширенні навігаційне меню згортається у бургер-кнопку, сітки публікацій та товарів переходять в одно- або двоколонковий режим, бічні панелі прибираються, а таблиці отримують горизонтальне прокручування. Форма оформлення замовлення перебудовується з двоколонкової на одноколонкову. Кошик у шапці відображається у вигляді іконки з лічильником на всіх розширеннях (рис. 3.37).

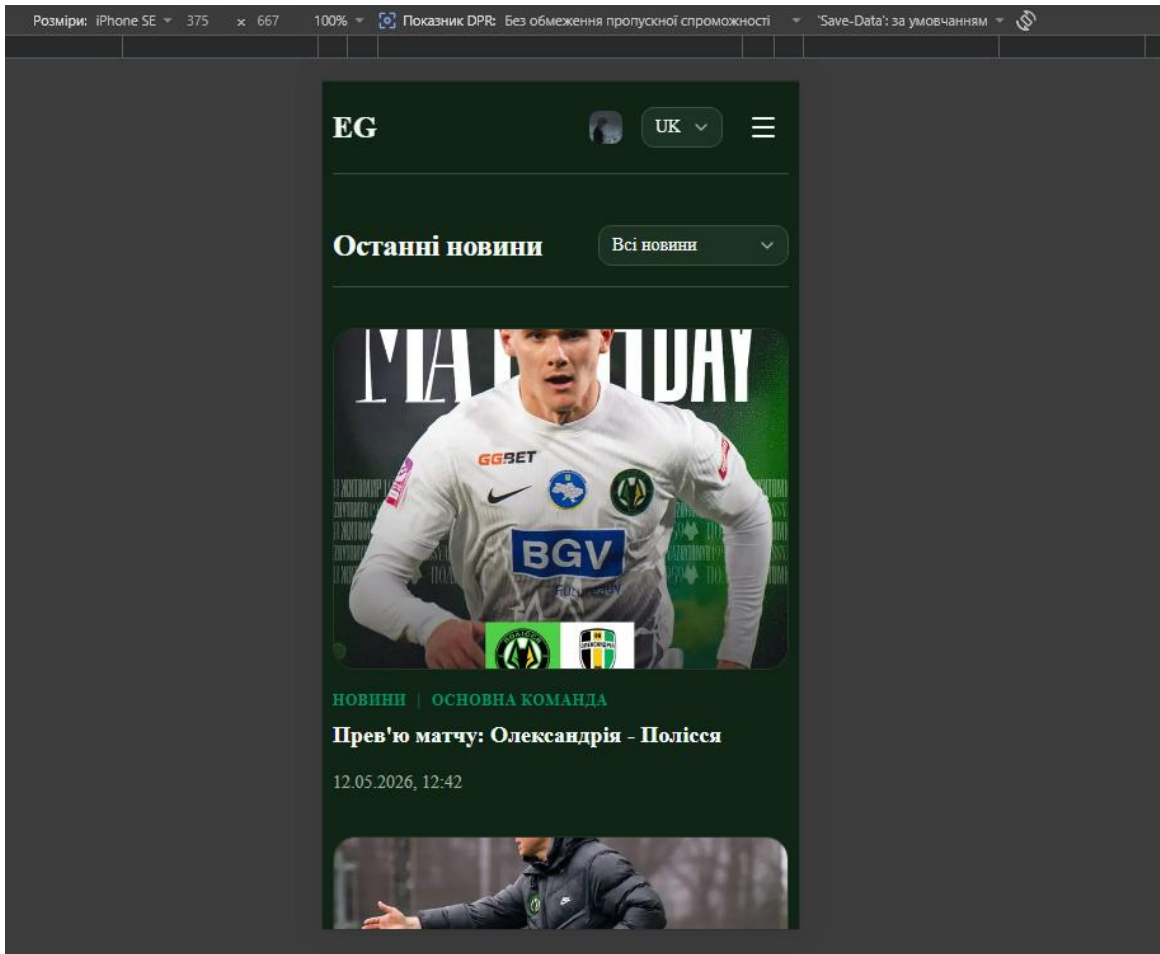


Рис. 3.37 – Мобільна версія порталу

Описаний інтерфейс охоплює всі ключові сценарії використання для трьох категорій користувачів – гостя, авторизованого вболівальника та адміністратора. Дотримання єдиної кольорової схеми, типографіки та патернів взаємодії на всіх сторінках знижує когнітивне навантаження та пришвидшує освоєння системи новим користувачем.

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		67

3.3 Тестування вебпорталу футбольного клубу з інтегрованим фаншопом

Тестування розробленого вебпорталу проводилося на production-розгортанні за адресою <https://fc-web-portal.vercel.app> з реальною базою даних Neon та реальними мережевими затримками, що відображає умови експлуатації застосунку кінцевими користувачами. Обрано підхід інтеграційного та end-to-end тестування замість виключно модульного – для порталу такого типу перевірка коректності окремих функцій у відриві від їхнього контексту не дає реальної гарантії того, що система працює як єдине ціле.

Перевірка порталу розділена на чотири блоки. Перший блок перевіряє коректність роботи ключових підсистем та процесів – автентифікації, фаншопу, синхронізації з Sofascore та адміністративної панелі. Другий блок верифікує алгоритм гібридної статистики гравців, а також алгоритми оформлення замовлення та обробки результатів платежу на граничних випадках. Третій блок оцінює продуктивність застосунку у production-середовищі за допомогою аудиту Google Lighthouse. Четвертий блок підтверджує сумісність із основними браузерними рушіями.

Перевірка застосунку розпочалася з тестування ключових функціональних сценаріїв порталу. Для кожної з ключових підсистем розроблено набір тестових сценаріїв, що перевіряють поведінку системи як на коректних вхідних даних, так і на граничних випадках (табл. 3.2).

Таблиця 3.2

Результати тестування функціональних сценаріїв

Підсистема / Процес	Тестовий сценарій	Очікуваний результат	Статус
Автентифікація	Реєстрація через email і пароль	Акаунт створено	+
Автентифікація	Вхід через Google OAuth	JWT-токен видано	+

продовж. табл. 3.2

Автентифікація	Автоприв'язка замовлень при вході через Google	Замовлення прив'язано до userId	+
Автентифікація	Спроба доступу до /admin без ролі ADMIN	Перенаправлення до /profile	+
Автентифікація	Перехід на /login при активній сесії	Перенаправлення до /profile	+
Фаншоп	Додавання товару до кошика без авторизації	Товар збережено у localStorage	+
Фаншоп	Перевірка залишку при оформленні замовлення	Транзакція перервана з помилкою OutOfStock	+
Фаншоп	Оформлення замовлення з методом COD	Замовлення створено, перенаправлення на сторінку	+
Фаншоп	Оформлення замовлення з карткою (Stripe)	Перенаправлення на Stripe Checkout	+
Фаншоп	Webhook: checkout.session.completed	isPaid = true у замовленні	+
Фаншоп	Webhook: checkout.session.expired	Статус CANCELLED, залишки повернено на склад	+

Синхронізація	Синхронізація матчів поточного сезону	Матчі створено/оновлено в БД	+
Синхронізація	Синхронізація деталей: автогол	isOpponent інвертовано	+
Синхронізація	Синхронізація складу: гравця не знайдено за sofascoreId	Запис пропущено без помилки	+
Адмін-панель	Оновлення таблиці для нової команди	Запис створено у TeamDictionary	+
Адмін-панель	Архівування турніру з прив'язаними матчами	Помилка: неможливо архівувати	+
Адмін-панель	Архівування активного сезону	Помилка: неможливо архівувати	+
Адмін-панель	Архівування категорії з прив'язаними товарами	Помилка: неможливо архівувати	+

Усі вісімнадцять сценаріїв пройшли успішно. Особливої уваги заслуговує тестування платіжної інтеграції: Stripe Webhook коректно оновлює статус оплати при успішній транзакції та повертає залишки товарів на склад при закінченні сесії. Захист адміністративних маршрутів через middleware коректно відпрацьовує всі варіанти несанкціонованого доступу. Логіка інверсії автогола при синхронізації подій матчу правильно зараховує м'яч суперникові незалежно від того, хто його забив.

Окрему увагу під час тестування було приділено верифікації алгоритму гібридної статистики гравців, а також алгоритмів оформлення замовлення та обробки результатів платежу – трьох найбільш критичних з точки зору бізнес-логіки алгоритмів системи. Для кожного розроблено набір граничних випадків, що охоплюють як типові сценарії, так і крайні стани даних (табл. 3.3, 3.4).

Таблиця 3.3

Граничні випадки алгоритму гібридної статистики

Код	Вхідні умови	Очікуваний результат	Фактичний результат	Статус
STAT-1	initialGoals=5, liveGoals=0	goals=5	goals=5	+
STAT-2	initialGoals=0, liveGoals=3	goals=3	goals=3	+
STAT-3	initialGoals=10, liveGoals=7	goals=17	goals=17	+
STAT-4	Воротар: матч завершено 0:0	cleanSheets+=1	cleanSheets+=1	+
STAT-5	Воротар: матч завершено 2:1	conceded+=1, cleanSheets без змін	conceded+=1	+
STAT-6	Гравець не брав участі в матчах поточного сезону	Показуються лише initialStats	Показуються лише initialStats	+

Таблиця 3.4

Граничні випадки алгоритмів оформлення замовлення та обробки платежів

Код	Вхідні умови	Очікуваний результат	Фактичний результат	Статус
STK-1	Замовлення товару з залишком 0	Помилка OutOfStock з кількістю 0	Помилка OutOfStock	+

STK-2	Замовлення 3 одиниць при залишку 2	Помилка з доступною кількістю 2	Помилка, availableStock=2	+
STK-3	Webhook expired: повернення залишків	stock += quantity для кожної позиції	stock збільшено коректно	+
STK-4	Додавання більше 10 одиниць одного товару	Кількість обмежена 10	Кількість=10	+
STK-5	Паралельне оформлення одного товару	Транзакція Prisma гарантує цілісність	Один запит успішний	+

Усі одинадцять граничних випадків пройшли з нульовою розбіжністю між очікуваним і фактичним результатом. Особливо важливим є STK-5, який підтверджує коректність використання транзакцій Prisma при одночасному оформленні – атомарна операція гарантує, що залишки не можуть стати від'ємними навіть при паралельних запитах.

Для підтвердження відповідності технічним стандартам було проведено оцінку продуктивності застосунку у production-середовищі. Для виміру використано інструмент Google PageSpeed Insights на базі Lighthouse API – галузевий стандарт оцінки якості вебзастосунків [5]. Тестування проводилось у двох режимах – Комп'ютер та Мобільний – проти production-домену <https://fc-web-portal.vercel.app> за реальних мережевих умов (рис. 3.38).

Звіт від 14 черв. 2026 р., 13:03:34

<https://fc-web-portal.vercel.app/>

Аналізувати



Мобільний



Комп'ютер

Результати пошуку для URL-адреси: <https://fc-web-portal.vercel.app/en>
Запустити з оригінальною URL-адресою



Дізнайтеся, як відбувається взаємодія в реальних користувачів

Немає даних



Виявляйте проблеми з продуктивністю

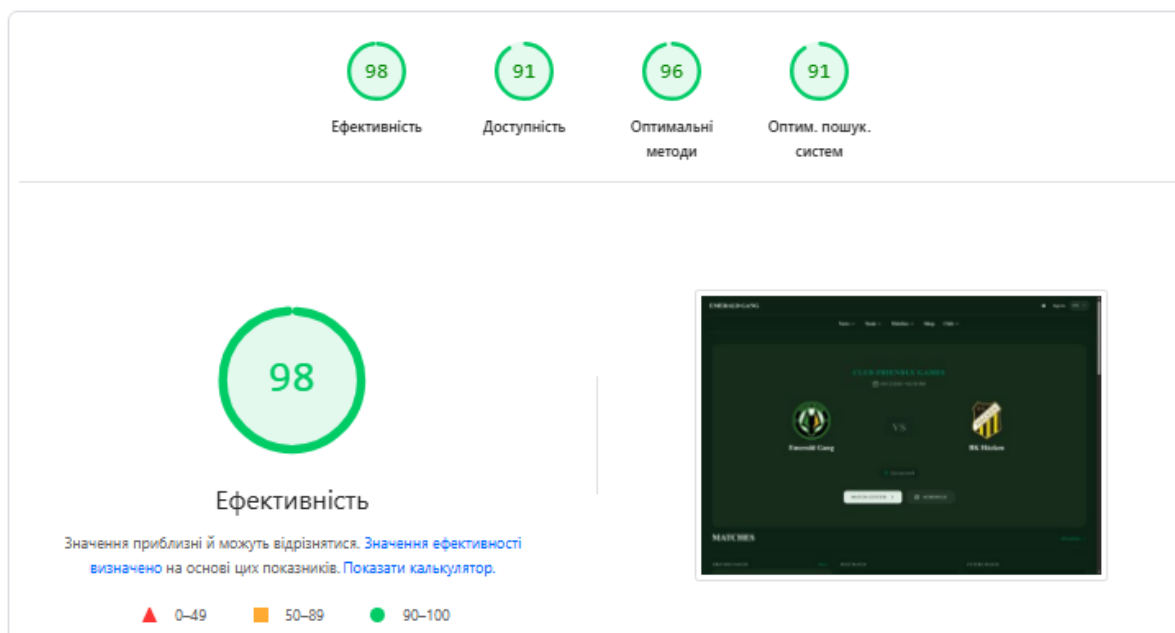


Рис. 3.38 – Підсумкові оцінки Google Lighthouse для production-розгортання

У режимі Комп'ютер портал отримав 98/100 за Ефективністю, 91/100 за Доступністю, 96/100 за Оптимальними методами та 91/100 за SEO. У режимі Мобільний показники склали відповідно 96/100, 91/100, 96/100 та 91/100. Всі чотири категорії в обох режимах потрапили до зеленої зони (90–100), що свідчить про відповідність сучасним стандартам якості вебзастосунків [6].

Оцінка Ефективності 98 для десктопу та 96 для мобільних пристроїв досягнута завдяки кільком архітектурним рішенням. Серверний рендеринг через Next.js Server Components забезпечує швидке перше відображення – HTML надходить вже з готовим вмістом без очікування виконання клієнтського JavaScript. Оптимізація зображень через компонент next/image з автоматичною конвертацією у WebP та lazy loading позитивно впливає на ці показники.

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		73

Категорія Оптимальні методи з оцінкою 96/100 підтверджує дотримання сучасних стандартів – коректне налаштування HTTPS, відсутність застарілих API та відповідність вимогам безпеки браузера. Стабільно висока оцінка SEO 91/100 в обох режимах забезпечується динамічно генерованими мета-тегами та Open Graph даними для кожної сторінки через функцію generateMetadata.

На завершальному етапі здійснювалася перевірка кросбраузерності порталу та його сумісності з основними рушіями. З цією метою було розроблено smoke-тест, що послідовно проходить сім ключових сценаріїв – завантаження головної сторінки, перегляд новин, перегляд профілів команди, навігація матч-центром, взаємодія з фаншопом (каталог та кошик), автентифікація (логін/реєстрація) та перехід до адмін-панелі (табл. 3.5).

Таблиця 3.5

Результати кросбраузерного тестування

Рушій	Браузер	Головна	Новини	Команда	Матчі	Фаншоп	Логін / Реєстрація	Адмін-панель
Chromium 141	Chrome, Edge	+	+	+	+	+	+	+
WebKit 26.4	Safari 17+	+	+	+	+	+	+	+
Gecko 131	Firefox	+	+	+	+	+	+	+

Усі сім сценаріїв успішно пройшли в усіх трьох браузерних рушіях, що охоплюють практично весь сучасний ринок десктопних і мобільних браузерів. Збереження кошика через localStorage та Zustand однаково працює в усіх рушіях. Мультимовна маршрутизація через next-intl коректно відпрацьовує перемикання між українською та англійською в усіх браузерах. Завдяки тому, що в кодовій

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				74
Змн.	Арк.	№ докум.	Підпис	Дата		

базі не використовуються пропрієтарні API окремих браузерів, а стилізація реалізована через стандартні утилітарні класи Tailwind CSS v4, додаткова робота з адаптацією виявилася непотрібною [20].

Підсумовуючи результати чотирьох блоків тестування: функціональні сценарії коректно відпрацьовують за всіма ключовими бізнес-процесами, алгоритми гібридної статистики, оформлення замовлення та обробки платежів працюють згідно з документованою логікою на всіх граничних випадках, продуктивність відповідає сучасним вимогам за метриками Google Lighthouse, а сумісність охоплює усі актуальні браузерні рушії. Це підтверджує готовність порталу до реального впровадження у діяльність футбольного клубу.

Висновки до третього розділу

У третьому розділі описано порядок встановлення, структуру інтерфейсу та результати тестування вебпорталу футбольного клубу з інтегрованим фаншопом.

Описано процес розгортання застосунку у хмарному середовищі на двох незалежних платформах – Vercel для Next.js-застосунку та Neon для PostgreSQL-бази даних. Налаштовано CI/CD пайплайн через GitLab із чотирма стадіями – перевіркою коду, типів, збіркою та дзеркалюванням на GitHub – що забезпечує автоматичне розгортання при кожному push до гілки main. Наведено перелік чотирнадцяти змінних середовища та описано порядок налаштування зовнішніх сервісів: Google OAuth, UploadThing, Stripe та RapidAPI.

Описано структуру інтерфейсу порталу, організованого через чотири незалежні layout-каркаси для публічної частини, фаншопу, автентифікації та адміністрування. Розглянуто тридцять сім екранів застосунку, що охоплюють усі ключові сценарії для трьох категорій користувачів – гостя, авторизованого вболівальника та адміністратора. Показано реалізацію адаптивного інтерфейсу через утилітарні класи Tailwind CSS v4 без окремих CSS-файлів.

За результатами тестування усі вісімнадцять функціональних сценаріїв пройшли успішно, підтвердивши коректність роботи підсистем автентифікації та

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		75

фаншопу, процесів синхронізації з Sofascore та захисту адміністративних маршрутів. Одинадцять граничних випадків алгоритмів гібридної статистики, оформлення замовлення та обробки платежів пройшли з нульовою розбіжністю між очікуваним і фактичним результатом. Аудит Google PageSpeed Insights підтвердив високу продуктивність – 98/100 за Ефективністю у режимі Комп'ютер та 96/100 у режимі Мобільний, усі чотири категорії потрапили до зеленої зони. Кросбраузерне тестування у трьох рушіях – Chromium, WebKit та Gecko – підтвердило коректну роботу всіх семи перевірених сценаріїв без необхідності додаткової адаптації.

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		76

ВИСНОВКИ

У випускній кваліфікаційній роботі бакалавра вирішено актуальну задачу розробки повнофункціонального вебпорталу футбольного клубу з інтегрованим фаншопом як комплексної вебплатформи на базі сучасного повностекового фреймворку Next.js 16.

Проведено аналіз предметної області та існуючих рішень. Порівняльний огляд офіційних вебпорталів «Шахтаря», «Полісся», «Реал Мадрида» та «Арсенала» виявив спільну архітектурну проблему, характерну навіть для клубів світового рівня: фаншоп у всіх випадках реалізовано на окремому піддомені, що порушує цілісність сесії покупця, приховує кошик при переходах між розділами та ускладнює адміністрування двох незалежних систем. Визначено п'ять функціональних модулів системи – контент та медіа, команда, матч-центр, електронна комерція та користувачі – та обґрунтовано вибір гібридної клієнт-серверної архітектури на базі Next.js 16 App Router як оптимального рішення для поєднання серверного рендерингу з інтерактивністю електронної комерції.

Спроектовано та реалізовано систему. Розроблено об'єктно-орієнтовану модель із 31 сутністю та побудовано реляційну базу даних PostgreSQL, організовану за п'ятьма предметними областями з двома наскрізними патернами – мультимовністю через таблиці перекладів та м'яким видаленням через поле deletedAt. Спроектовано та реалізовано сім алгоритмів роботи системи: автентифікації з автоматичним прив'язуванням гостей замовлень, транзакційного оформлення замовлення з перевіркою залишків, обробки платежів через Stripe Webhook, трьох алгоритмів синхронізації спортивних даних з Sofascore через Vercel Cron та обчислення гібридної статистики гравців. Технічну реалізацію засновано на 16 Server Actions як єдиному шарі мутацій без окремого REST API, патерні useActionState з React 19 для клієнтських форм адмін-панелі та підключенні до Neon PostgreSQL через @prisma/adapters-pg з пулом з'єднань для сумісності з безсерверним середовищем Vercel.

Виконано розгортання, описано інтерфейс та проведено тестування.

Застосунок розгорнуто на платформі Vercel із автоматичним CI/CD пайплайном

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		77

через GitLab – чотири стадії перевірки завершуються дзеркалюванням на GitHub та автоматичним деплоєм. Інтерфейс організовано через чотири незалежні layout-каркаси для публічної частини, фаншопу, автентифікації та адміністрування, адаптований для мобільних пристроїв засобами Tailwind CSS v4. За результатами тестування всі 18 функціональних сценаріїв та 11 граничних випадків пройшли успішно. Аудит Google PageSpeed Insights підтвердив оцінку 98/100 за Ефективністю у режимі Комп'ютер та 96/100 у режимі Мобільний – усі чотири категорії в зеленій зоні. Кросбраузерне тестування у рушіях Chromium, WebKit та Gecko підтвердило повну сумісність без додаткової адаптації.

Практична цінність роботи полягає в тому, що розроблений портал усуває ключовий недолік аналогів – архітектурний розрив між медіаплатформою та магазином – шляхом реалізації фаншопу як повноцінного модуля єдиної кодової бази. Наскрізний кошик на Zustand доступний з будь-якої сторінки, контекстні продажі через профілі гравців збільшують конверсію, а автоматична синхронізація з Sofascore знімає рутинне навантаження з адміністраторів клубу.

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		78

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. next-intl – Internationalization (i18n) for Next.js. URL: <https://next-intl.dev/> (дата звернення: 14.06.2026).
2. NextAuth.js. URL: <https://next-auth.js.org/> (дата звернення: 14.06.2026).
3. GitLab Docs. URL: <https://docs.gitlab.com/ci/> (дата звернення: 14.06.2026).
4. GitHub.com Help Documentation. *GitHub Docs*. URL: <https://docs.github.com/en> (дата звернення: 14.06.2026).
5. About PageSpeed Insights | Google for Developers. *Google for Developers*. URL: <https://developers.google.com/speed/docs/insights/v5/about> (дата звернення: 14.06.2026).
6. Web vitals | articles | web.dev. *web.dev*. URL: <https://web.dev/articles/vitals> (дата звернення: 14.06.2026).
7. Documentation - ESLint - Pluggable JavaScript Linter. *Find and fix problems in your JavaScript code - ESLint - Pluggable JavaScript Linter*. URL: <https://eslint.org/docs/latest/> (дата звернення: 14.06.2026).
8. Index | node.js v26.3.0 documentation. *Node.js – Run JavaScript Everywhere*. URL: <https://nodejs.org/docs/latest/api/> (дата звернення: 14.06.2026).
9. Intro | Zod. *Zod*. URL: <https://zod.dev/> (дата звернення: 14.06.2026).
10. *React*. URL: <https://react.dev/> (дата звернення: 14.06.2026).
11. The starting point for learning TypeScript. *TypeScript: JavaScript With Syntax For Types*. URL: <https://www.typescriptlang.org/docs/> (дата звернення: 14.06.2026).
12. Neon. Neon documentation - Neon Docs. *Neon*. URL: <https://neon.com/docs/introduction> (дата звернення: 14.06.2026).
13. UploadThing docs - uploadthing docs. URL: <https://docs.uploadthing.com/> (дата звернення: 14.06.2026).
14. Introduction - Zustand. *Poimandres documentation*. URL: <https://zustand.docs.pmnd.rs/learn/getting-started/introduction> (дата звернення: 14.06.2026).

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		79

15. PostgreSQL: Documentation. *PostgreSQL: The world's most advanced open source database*. URL: <https://www.postgresql.org/docs/> (дата звернення: 14.06.2026).
16. Get started with Prisma. *Prisma | Integrated TypeScript Infrastructure for Agentic Software Development*. URL: <https://www.prisma.io/docs> (дата звернення: 14.06.2026).
17. RapidAPI. URL: <https://docs.rapidapi.com/> (дата звернення: 14.06.2026).
18. Introduction. *The Foundation for your Design System - shadcn/ui*. URL: <https://ui.shadcn.com/docs> (дата звернення: 14.06.2026).
19. Dokumentation. *Stripe-Dokumentation*. URL: <https://docs.stripe.com/> (дата звернення: 14.06.2026).
20. Install Tailwind CSS with Next.js. *Tailwind CSS - Rapidly build modern websites without ever leaving your HTML*. URL: <https://tailwindcss.com/docs/installation/framework-guides/nextjs> (дата звернення: 14.06.2026).
21. Introduction | TanStack Table Docs. *TanStack | The open-source application stack for the web*. URL: <https://tanstack.com/table/latest/docs/introduction> (дата звернення: 14.06.2026).
22. Get started | Tiptap Editor Docs. *Tiptap - Dev Toolkit Editor Suite*. URL: <https://tiptap.dev/docs/editor/getting-started/overview> (дата звернення: 14.06.2026).
23. Vercel. Next.js Docs | Next.js. *Next.js by Vercel - The React Framework*. URL: <https://nextjs.org/docs> (дата звернення: 14.06.2026).
24. Vercel Documentation. *Vercel: Build and deploy the best web experiences with the AI Cloud*. URL: <https://vercel.com/docs> (дата звернення: 14.06.2026).
25. Introduction – radix primitives. *Radix UI*. URL: <https://www.radix-ui.com/primitives/docs/overview/introduction> (дата звернення: 14.06.2026).
26. Halvorsen J. *Learning TypeScript : Enhance your web development skills using type-safe javascript*. O'Reilly Media, Incorporated, 2022. 318 с.

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				80
Змн.	Арк.	№ докум.	Підпис	Дата		

27. Osmani A. Learning JavaScript Design Patterns : A JavaScript and React Developer's Guide. O'Reilly Media, Incorporated, 2023. 286 с.
28. Schwarzmüller M. React Key Concepts : Consolidate Your Knowledge of React's Core Features. Packt Publishing, Limited, 2022. 590 с.
29. Smith B. DevOps for the Desperate : A Hands-On Survival Guide. No Starch Press, Incorporated, 2022. 176 с.
30. Riva M. Real-World Next.js : Build scalable, high-performance, and modern web applications using Next.js, the React framework for production. Packt Publishing, 2022. 366 с.

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		81

ДОДАТКИ

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		82

Схема бази даних вебпорталу футбольного клубу

```

generator client {
  provider = "prisma-client-js"
  output   = "../generated/prisma"
}

datasource db {
  provider = "postgresql"
}

enum UserRole {
  ADMIN
  USER
}

model User {
  id          String    @id @default(cuid())
  name        String?
  email       String?   @unique
  emailVerified DateTime?
  image       String?
  password    String?
  role        UserRole  @default(USER)
  accounts    Account[]
  sessions    Session[]
  orders      Order[]

  createdAt DateTime @default(now())
  updatedAt DateTime @updatedAt
  deletedAt DateTime?

  @@index([deletedAt])
}

model Account {
  id          String    @id @default(cuid())
  userId      String
  type        String
  provider    String
  providerAccountId String
  refresh_token String? @db.Text
  access_token String? @db.Text
  expires_at  Int?
  token_type  String?
  scope       String?
  id_token    String? @db.Text
  session_state String?

```

```

    user User @relation(fields: [userId], references: [id], onDelete: Cascade)

    @@unique([provider, providerAccountId])
}

model Session {
  id          String    @id @default(cuid())
  sessionToken String    @unique
  userId      String
  expires     DateTime
  user        User      @relation(fields: [userId], references: [id], onDelete:
Cascade)
}

model VerificationToken {
  identifier String
  token       String    @unique
  expires     DateTime

  @@unique([identifier, token])
}

enum MediaType {
  IMAGE
  VIDEO
  DOCUMENT
}

model Media {
  id          String    @id @default(cuid())
  url         String
  type        MediaType @default(IMAGE)
  posts       Post[]
  players     Player[]
  coaches     Coach[]
  products    Product[]
  galleries   Gallery[]

  createdAt DateTime @default(now())
  deletedAt DateTime?

  @@index([deletedAt])
}

enum PostType {
  NEWS
  INTERVIEW
  STATEMENT
}

```

```

enum TeamContext {
    MAIN_TEAM
    U19
    ACADEMY
    GENERAL
}

model Post {
    id                String                @id @default(cuid())
    slug              String                @unique
    translations       PostTranslation[]
    media              Media[]
    type               PostType             @default(NEWS)
    teamContext        TeamContext          @default(GENERAL)
    isPublished        Boolean              @default(true)
    publishedAt        DateTime             @default(now())
    mentionedPlayers   Player[]
    mentionedCoaches   Coach[]
    relatedMatches     Match[]

    createdAt DateTime @default(now())
    updatedAt DateTime @updatedAt
    deletedAt DateTime?

    @@index([deletedAt])
    @@index([publishedAt])
    @@index([isPublished])
    @@index([type])
    @@index([teamContext])
}

model PostTranslation {
    id                String @id @default(cuid())
    postId            String
    post              Post   @relation(fields: [postId], references: [id], onDelete:
Cascade)
    language          String @db.VarChar(5)
    title             String
    description        String?
    content            String @db.Text

    @@unique([postId, language])
    @@index([language])
}

model Gallery {
    id                String                @id @default(cuid())
    slug              String                @unique
    translations       GalleryTranslation[]
    coverUrl          String

```

```

media          Media[]
matchId        String?
match          Match? @relation(fields: [matchId], references: [id])

publishedAt    DateTime @default(now())
createdAt      DateTime @default(now())
updatedAt      DateTime @updatedAt
deletedAt      DateTime?

@@index([deletedAt])
}

model GalleryTranslation {
  id          String @id @default(cuid())
  galleryId   String
  gallery     Gallery @relation(fields: [galleryId], references: [id], onDelete: Cascade)

  language    String @db.VarChar(5)
  title       String

  @@unique([galleryId, language])
}

enum PlayerPosition {
  GOALKEEPER
  DEFENDER
  MIDFIELDER
  FORWARD
}

model Player {
  id          String @id @default(cuid())
  slug        String @unique
  sofascoreId Int? @unique
  translations PlayerTranslation[]
  number      Int
  position    PlayerPosition
  teamContext TeamContext @default(MAIN_TEAM)
  avatar      String?
  isManualAvatar Boolean @default(false)
  media       Media[]
  birthDate   DateTime?
  height      Int?
  weight      Int?
  nationality  String? @db.VarChar(10)
  initialMatches Int @default(0)
  initialGoals Int @default(0)
  initialAssists Int @default(0)
  initialCleanSheets Int @default(0)
}

```

```

    initialGoalsConceded Int @default(0)
    lineupEntries MatchLineup[]
    events MatchEvent[]
    mentionedInPosts Post[]
    relatedProducts Product[]

    createdAt DateTime @default(now())
    updatedAt DateTime @updatedAt
    deletedAt DateTime?

    @@index([deletedAt])
}

model PlayerTranslation {
  id String @id @default(cuid())
  playerId String
  player Player @relation(fields: [playerId], references: [id], onDelete: Cascade)
  language String @db.VarChar(5)
  name String
  bio String? @db.Text

  @@unique([playerId, language])
}

model Coach {
  id String @id @default(cuid())
  slug String @unique
  translations CoachTranslation[]
  avatar String?
  media Media[]
  teamContext TeamContext @default(MAIN_TEAM)
  nationality String? @db.VarChar(10)
  birthDate DateTime?
  mentionedInPosts Post[]

  createdAt DateTime @default(now())
  updatedAt DateTime @updatedAt
  deletedAt DateTime?

  @@index([deletedAt])
}

model CoachTranslation {
  id String @id @default(cuid())
  coachId String
  coach Coach @relation(fields: [coachId], references: [id], onDelete: Cascade)
  language String @db.VarChar(5)
  name String

```

```

    role      String
    bio        String? @db.Text

    @@unique([coachId, language])
}

enum MatchStatus {
    SCHEDULED
    LIVE
    FINISHED
    POSTPONED
    CANCELED
}

model Opponent {
    id          String          @id @default(cuid())
    slug         String          @unique
    sofascoreId Int?            @unique
    logoUrl      String?
    translations OpponentTranslation[]
    matches      Match[]
    deletedAt    DateTime?
}

model OpponentTranslation {
    id          String @id @default(cuid())
    opponentId  String
    opponent    Opponent @relation(fields: [opponentId], references: [id], onDelete: Cascade)
    language    String @db.VarChar(5)
    name        String

    @@unique([opponentId, language])
}

model Tournament {
    id          String          @id @default(cuid())
    slug         String          @unique
    sofascoreId Int?            @unique
    hasStandings Boolean          @default(false)
    translations TournamentTranslation[]
    matches      Match[]
    standings     Standing[]
    deletedAt    DateTime?

    @@index([deletedAt])
}

model TournamentTranslation {
    id          String @id @default(cuid())

```

```

    tournamentId String
    tournament Tournament @relation(fields: [tournamentId], references: [id],
onDelete: Cascade)
    language String @db.VarChar(5)
    name String

    @@unique([tournamentId, language])
}

model Standing {
    id String @id @default(cuid())
    rank Int
    teamName String
    teamLogo String?
    points Int
    played Int
    win Int
    draw Int
    lose Int
    goalsFor Int @default(0)
    goalsAgainst Int @default(0)
    goalsDiff Int
    tournamentId String?
    tournament Tournament? @relation(fields: [tournamentId], references: [id])
    seasonId String?
    season Season? @relation(fields: [seasonId], references: [id])
    teamContext TeamContext @default(MAIN_TEAM)

    updatedAt DateTime @updatedAt
}

model TeamDictionary {
    id String @id @default(cuid())
    sofascoreId Int @unique
    originalName String
    translations TeamDictionaryTranslation[]

    createdAt DateTime @default(now())
    updatedAt DateTime @updatedAt
}

model TeamDictionaryTranslation {
    id String @id @default(cuid())
    teamDictionaryId String
    teamDictionary TeamDictionary @relation(fields: [teamDictionaryId], references:
[id], onDelete: Cascade)
    language String @db.VarChar(5)
    name String

    @@unique([teamDictionaryId, language])
}

```

```

}

model Match {
  id          String      @id @default(cuid())
  slug        String      @unique
  sofascoreId Int?        @unique
  date        DateTime
  startTimestamp Int?
  status       MatchStatus @default(SCHEDULED)
  isDetailsSynced Boolean  @default(false)
  round        Int?
  teamContext  TeamContext @default(MAIN_TEAM)
  tournamentId String?
  tournament   Tournament? @relation(fields: [tournamentId], references: [id])
  seasonId     String?
  season       Season?     @relation(fields: [seasonId], references: [id])
  opponentId   String
  opponent     Opponent    @relation(fields: [opponentId], references: [id])
  isHomeGame   Boolean     @default(true)
  homeScore    Int?
  awayScore    Int?
  opponentLineup Json?
  stadium      String?
  homeCoachName String?
  awayCoachName String?
  highlightsUrl String?
  postMatchUrl String?
  galleries     Gallery[]
  lineup        MatchLineup[]
  events        MatchEvent[]
  relatedPosts  Post[]

  createdAt DateTime @default(now())
  updatedAt DateTime @updatedAt
  deletedAt DateTime?

  @@index([deletedAt])
  @@index([seasonId])
  @@index([teamContext])
}

model MatchLineup {
  id          String @id @default(cuid())
  matchId     String
  match       Match  @relation(fields: [matchId], references: [id], onDelete: Cascade)
  playerId    String
  player      Player @relation(fields: [playerId], references: [id], onDelete: Cascade)
  isStarter   Boolean @default(false)

```

```

    played Boolean @default(false)

    createdAt DateTime @default(now())

    @@unique([matchId, playerId])
}

enum EventType {
    GOAL
    ASSIST
    YELLOW_CARD
    RED_CARD
    SUBSTITUTION_IN
    SUBSTITUTION_OUT
}

model MatchEvent {
    id String @id @default(cuid())
    matchId String
    match Match @relation(fields: [matchId], references: [id],
onDelete: Cascade)
    playerId String?
    player Player? @relation(fields: [playerId], references: [id],
onDelete: Cascade)
    customPlayerName String?
    isOpponent Boolean @default(false)
    type EventType
    minute Int

    createdAt DateTime @default(now())

    @@index([playerId])
    @@index([type])
}

model Category {
    id String @id @default(cuid())
    slug String @unique
    translations CategoryTranslation[]

    products Product[]
    deletedAt DateTime?

    @@index([deletedAt])
}

model CategoryTranslation {
    id String @id @default(cuid())
    categoryId String

```

```

    category    Category @relation(fields: [categoryId], references: [id], onDelete:
Cascade)
    language    String   @db.VarChar(5)
    name        String

    @@unique([categoryId, language])
}

enum Demographic {
    UNISEX
    MEN
    WOMEN
    KIDS
}

model Product {
    id          String      @id @default(cuid())
    slug        String      @unique
    translations ProductTranslation[]
    price       Decimal     @db.Decimal(10, 2)
    salePrice   Decimal?    @db.Decimal(10, 2)
    isOnSale    Boolean     @default(false)
    categoryId  String
    category    Category    @relation(fields: [categoryId], references:
[id])
    media       Media[]
    demographic Demographic @default(UNISEX)
    color       String?
    apparelType String?
    seasonYear  String?
    matchType   String?
    variants    ProductVariant[]
    isArchived  Boolean     @default(false)
    isFeatured  Boolean     @default(false)
    orderItems  OrderItem[]
    relatedPlayers Player[]

    createdAt DateTime @default(now())
    updatedAt DateTime @updatedAt
    deletedAt DateTime?

    @@index([categoryId, isArchived, deletedAt])
    @@index([demographic])
    @@index([color])
    @@index([apparelType])
    @@index([isOnSale])
    @@index([createdAt])
    @@index([deletedAt])
}

```

```

model ProductTranslation {
  id          String @id @default(cuid())
  productId   String
  product     Product @relation(fields: [productId], references: [id], onDelete:
Cascade)
  language    String @db.VarChar(5)
  name        String
  description String @db.Text

  @@unique([productId, language])
}

model ProductVariant {
  id          String @id @default(cuid())
  productId   String
  product     Product @relation(fields: [productId], references: [id], onDelete:
Cascade)
  sku         String? @unique
  size        String
  stock       Int      @default(0)
  position    Int      @default(0)

  orderItems OrderItem[]

  @@unique([productId, size])
  @@index([productId, stock])
  @@index([size, stock])
}

enum OrderStatusEnum {
  PENDING
  PAID
  SHIPPED
  DELIVERED
  CANCELLED
}

enum PaymentMethodEnum {
  CARD
  COD
}

model Order {
  id          String @id @default(cuid())
  userId      String?
  user        User?   @relation(fields: [userId], references: [id])
  status      OrderStatusEnum @default(PENDING)
  isPaid      Boolean  @default(false)
  totalPrice  Decimal  @db.Decimal(10, 2)
  firstName   String

```

```

        lastName      String
        email          String
        phone          String
        city           String
        postalCode     String?
        address        String?
        orderItems     OrderItem[]
        paymentMethod  PaymentMethodEnum @default(CARD)

        createdAt DateTime @default(now())
        updatedAt DateTime @updatedAt
        deletedAt DateTime?

        @@index([deletedAt])
    }

    model OrderItem {
        id            String          @id @default(cuid())
        orderId       String
        order         Order           @relation(fields: [orderId], references: [id],
onDelete: Cascade)
        productId     String
        product       Product         @relation(fields: [productId], references: [id])
        variantId     String?
        variant       ProductVariant? @relation(fields: [variantId], references: [id],
onDelete: SetNull)
        size          String?
        quantity      Int
        fixedPrice    Decimal          @db.Decimal(10, 2)
        customName    String?
        customNumber  String?
    }

    model Season {
        id            String          @id @default(cuid())
        slug          String          @unique
        name          String          @unique
        sofascoreId   Int?            @unique
        startDate     DateTime
        endDate       DateTime
        isActive      Boolean         @default(false)
        matches       Match[]
        standings     Standing[]

        createdAt DateTime @default(now())
        updatedAt DateTime @updatedAt
        deletedAt DateTime?

        @@index([deletedAt])
    }

```

Комбінований middleware автентифікації та інтернаціоналізації

```

import { auth } from "@auth";
import createMiddleware from "next-intl/middleware";
import { routing } from "@i18n/routing";
import { NextResponse } from "next/server";

const intlMiddleware = createMiddleware(routing);

const authPages = ["/login", "/register"];
const protectedPages = ["/profile", "/admin"];
const adminPages = ["/admin"];

export default auth((req) => {
  const { nextUrl } = req;
  const session = req.auth;
  const isLoggedIn = !!session;
  const userRole = session?.user?.role;

  const firstSegment = nextUrl.pathname.split("/")[1];

  const hasLocale = (routing.locales as readonly string[]).includes(firstSegment);

  const pathname = hasLocale
    ? nextUrl.pathname.replace(`/${firstSegment}`, "") || "/"
    : nextUrl.pathname;

  const redirectTo = (path: string) => {
    const url = nextUrl.clone();
    url.pathname = hasLocale ? `/${firstSegment}${path}` : path;
    return NextResponse.redirect(url);
  };

  if (isLoggedIn && authPages.includes(pathname)) {
    return redirectTo("/profile");
  }

  if (!isLoggedIn && protectedPages.some(route => pathname.startsWith(route))) {
    return redirectTo("/login");
  }

  if (adminPages.some(route => pathname.startsWith(route))) {
    if (userRole !== "ADMIN") {
      return redirectTo("/profile");
    }
  }

  return intlMiddleware(req);
});

```

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Герода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		95

```
});  
  
export const config = {  
  matcher: ["/((?!api|_next|_vercel|.*\\.\\.*)).*"/],  
};
```

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		96

Server Action оформлення замовлення у фаншопі

```

"use server";

import { auth } from "@auth";
import { prisma } from "@lib/prisma";
import { getLocale, getTranslations } from "next-intl/server";
import { redirect } from "@i18n/navigation";
import { redirect as nativeRedirect } from "next/navigation";
import { createCheckoutSchema } from "@lib/schemas";
import { CartItem } from "@store/useCartStore";
import { stripe } from "@lib/stripe";
import { getTranslation } from "@lib/utils/get-translation";

export async function getCheckoutInitialData() {
  const session = await auth();
  if (!session?.user?.id) return null;

  const userData = await prisma.user.findUnique({
    where: { id: session.user.id, deletedAt: null },
    select: {
      name: true,
      email: true,
      orders: {
        where: { deletedAt: null },
        orderBy: { createdAt: "desc" },
        take: 1,
        select: {
          firstName: true,
          lastName: true,
          phone: true,
          email: true,
          city: true,
          postalCode: true,
          address: true,
        },
      },
    },
  });

  if (!userData) return null;
  const lastOrder = userData.orders[0] || null;
  const nameParts = userData.name?.trim().split(" ") || [];

  return {
    firstName: lastOrder?.firstName || nameParts[0] || "",
    lastName: lastOrder?.lastName || nameParts.slice(1).join(" ") || "",
    email: lastOrder?.email || userData.email || "",
  };
}

```

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		97

```

    phone: lastOrder?.phone || "",
    city: lastOrder?.city || "",
    postalCode: lastOrder?.postalCode || "",
    address: lastOrder?.address || "",
  };
}

export type CheckoutState = {
  errors?: {
    firstName?: string[];
    lastName?: string[];
    email?: string[];
    phone?: string[];
    city?: string[];
    postalCode?: string[];
    address?: string[];
  };
  message?: string;
  outOfStockItem?: {
    variantId: string;
    availableStock: number;
  };
};

export async function processCheckout(
  cartItems: CartItem[],
  deliveryMethod: "branch" | "postomat" | "courier",
  paymentMethod: "card" | "cod",
  _prevState: CheckoutState | undefined,
  formData: FormData,
): Promise<CheckoutState | undefined> {
  const tErrors = await getTranslations("Shop.CheckoutErrors");
  const tForm = await getTranslations("Shop.CheckoutPage.Form");
  const tSummary = await getTranslations("Shop.CheckoutPage.Summary");
  const locale = await getLocale();

  const CheckoutSchema = createCheckoutSchema(tErrors);
  const validatedFields = CheckoutSchema.safeParse(
    Object.fromEntries(formData.entries()),
  );

  if (!validatedFields.success) {
    return {
      errors: validatedFields.error.flatten().fieldErrors,
      message: tErrors("invalidData"),
    };
  }

  const { firstName, lastName, email, phone, city, postalCode, address } =
    validatedFields.data;

```

```

if (deliveryMethod === "courier" && (!address || address.length < 2)) {
  return {
    errors: { address: [tErrors("tooShort")] },
    message: tErrors("invalidData"),
  };
}
if (deliveryMethod !== "courier" && (!postalCode || postalCode.length < 1)) {
  return {
    errors: { postalCode: [tErrors("tooShort")] },
    message: tErrors("invalidData"),
  };
}

const session = await auth();

const methodTranslate =
  deliveryMethod === "branch"
    ? tForm("deliveryBranch")
    : deliveryMethod === "postomat"
      ? tForm("deliveryPostomat")
      : tForm("deliveryCourier");

const finalAddress =
  deliveryMethod === "courier"
    ? `[${methodTranslate}] ${address}`
    : `[${methodTranslate}] №${postalCode}`;

try {
  if (!Array.isArray(cartItems) || cartItems.length === 0) {
    return { message: tErrors("emptyCart") };
  }

  const totalPrice = cartItems.reduce(
    (acc, item) => acc + item.price * item.quantity,
    0,
  );

  const order = await prisma.$transaction(async (tx) => {
    for (const item of cartItems) {
      if (!item.variantId) {
        throw new Error(`MissingVariantId:${item.productId}`);
      }

      const variant = await tx.productVariant.findUnique({
        where: { id: item.variantId },
      });

      if (!variant) {
        throw new Error(`VariantNotFound:${item.variantId}`);
      }
    }
  });
}

```

```

    }

    if (variant.stock < item.quantity) {
      throw new Error(`OutOfStock:${item.variantId}:${variant.stock}`);
    }

    await tx.productVariant.update({
      where: { id: variant.id },
      data: {
        stock: variant.stock - item.quantity,
      },
    });
  }

  return await tx.order.create({
    data: {
      userId: session?.user?.id || null,
      status: "PENDING",
      isPaid: false,
      paymentMethod: paymentMethod === "card" ? "CARD" : "COD",
      totalPrice,
      firstName,
      lastName,
      email,
      phone,
      city,
      postalCode: postalCode || "",
      address: finalAddress,
      orderItems: {
        create: cartItems.map((item) => ({
          productId: item.productId,
          variantId: item.variantId,
          size: item.size,
          quantity: item.quantity,
          fixedPrice: item.price,
          customName: item.customName || null,
          customNumber: item.customNumber || null,
        })),
      },
    },
  });
});

console.log("Order created successfully:", order.id);

const appUrl = process.env.AUTH_URL || "http://localhost:3000";

if (paymentMethod === "cod") {
  redirect({ href: `/shop/order/${order.id}`, locale: locale });
} else {

```

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		100

```

const line_items = cartItems.map((item) => {
  const translatedData = getTranslation(
    { translations: item.translations },
    locale,
  );
  const itemName = translatedData?.name || "Emerald Gang Produc";

  const descriptionParts = [];
  if (item.size) descriptionParts.push(`${tSummary("size")}: ${item.size}`);
  if (item.customName) descriptionParts.push(`Name: ${item.customName}`);
  if (item.customNumber) descriptionParts.push(`Number:
${item.customNumber}`);

  const description = descriptionParts.length > 0 ? descriptionParts.join(" •
") : undefined;

  return {
    price_data: {
      currency: "uah",
      product_data: {
        name: itemName,
        description: description,
      },
      unit_amount: Math.round(item.price * 100),
    },
    quantity: item.quantity,
  };
});

const stripeSession = await stripe.checkout.sessions.create({
  payment_method_types: ["card"],
  mode: "payment",
  billing_address_collection: "auto",
  line_items,
  success_url: `${appUrl}/${locale}/shop/order/${order.id}`,
  cancel_url: `${appUrl}/${locale}/shop/order/${order.id}`,
  metadata: {
    orderId: order.id,
  },
});

if (!stripeSession.url) {
  throw new Error("Stripe session URL generation failed");
}

nativeRedirect(stripeSession.url);
}
} catch (error: unknown) {
  const isRedirectError =
    error instanceof Error &&

```

```

        (error.message === "NEXT_REDIRECT" ||
        ("digest" in error && typeof error.digest === "string" &&
error.digest.startsWith("NEXT_REDIRECT")));

    if (isRedirectError) {
        throw error;
    }

    if (error instanceof Error && error.message.startsWith("OutOfStock")) {
        const [, variantId, stockStr] = error.message.split(":");
        const availableStock = parseInt(stockStr, 10);

        return {
            message: tErrors("outOfStock"),
            outOfStockItem: {
                variantId,
                availableStock
            }
        };
    }

    console.error("Checkout error:", error);
    return { message: tErrors("serverError") };
}
}

```

Обробник подій платіжного шлюзу Stripe Webhook

```

import { headers } from "next/headers";
import { NextResponse } from "next/server";
import { stripe } from "@lib/stripe";
import { prisma } from "@lib/prisma";
import Stripe from "stripe";
import { LOCALES } from "@lib/constants";
import { revalidatePath } from "next/cache";

function revalidateOrderPaths(orderId: string) {
  LOCALES.forEach((locale) => {
    revalidatePath(`/${locale}/shop/order/${orderId}`);
    revalidatePath(`/${locale}/admin/shop/orders/${orderId}`);
    revalidatePath(`/${locale}/admin/shop/orders`);
    revalidatePath(`/${locale}/profile/history`);
  });
}

export async function POST(req: Request) {
  const body = await req.text();
  const headersList = await headers();
  const signature = headersList.get("Stripe-Signature") as string;

  let event: Stripe.Event;

  try {
    event = stripe.webhooks.constructEvent(
      body,
      signature,
      process.env.STRIPE_WEBHOOK_SECRET!,
    );
  } catch (error: unknown) {
    const errorMessage =
      error instanceof Error ? error.message : "Unknown error";
    console.error("Webhook signature verification failed:", errorMessage);
    return new NextResponse(`Webhook Error: ${errorMessage}`, { status: 400 });
  }

  if (event.type === "checkout.session.completed") {
    const session = event.data.object as Stripe.Checkout.Session;
    const orderId = session.metadata?.orderId;

    if (orderId) {
      try {
        await prisma.order.update({
          where: { id: orderId },
          data: {

```

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		103

```

        isPaid: true,
      },
    });
    revalidateOrderPaths(orderId);
  } catch {
    return new NextResponse("Database Error", { status: 500 });
  }
}
} else if (event.type === "checkout.session.expired") {
  const session = event.data.object as Stripe.Checkout.Session;
  const orderId = session.metadata?.orderId;

  if (orderId) {
    try {
      await prisma.$transaction(async (tx) => {
        const order = await tx.order.findUnique({
          where: { id: orderId },
          include: { orderItems: true },
        });

        if (!order || order.status === "CANCELLED" || order.isPaid) {
          return;
        }

        await tx.order.update({
          where: { id: orderId },
          data: { status: "CANCELLED" },
        });

        for (const item of order.orderItems) {
          if (item.variantId) {
            await tx.productVariant.update({
              where: { id: item.variantId },
              data: {
                stock: {
                  increment: item.quantity,
                },
              },
            });
          }
        }
      });
    } catch {
      return new NextResponse("Database Error", { status: 500 });
    }
    revalidateOrderPaths(orderId);
  }
}
return new NextResponse(null, { status: 200 });
}

```

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		104

ER-діаграми бази даних

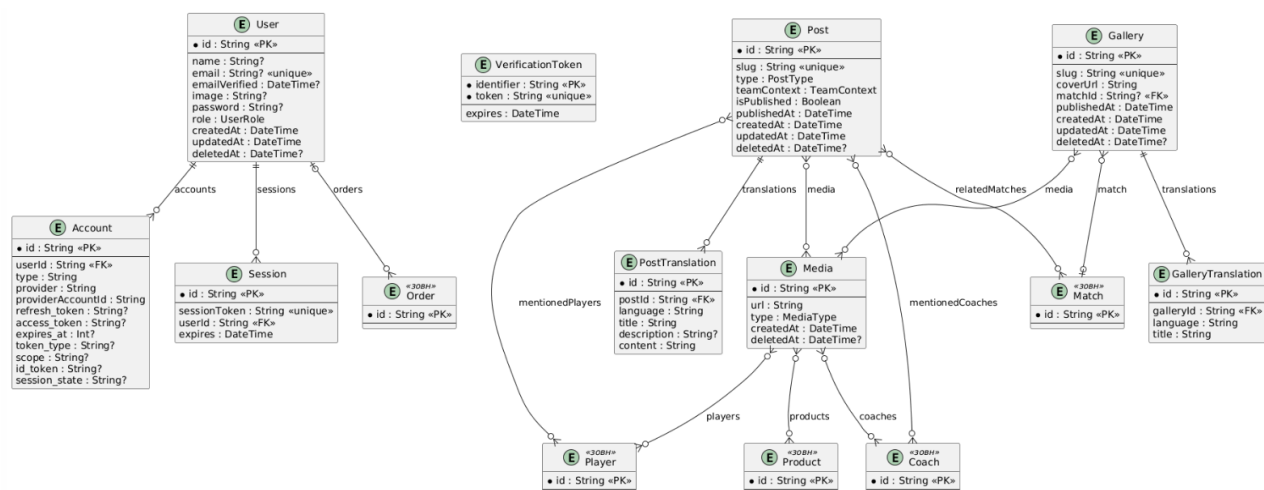


Рис. Д.1 – ER-діаграма бази даних (користувачі, контент та медіа)

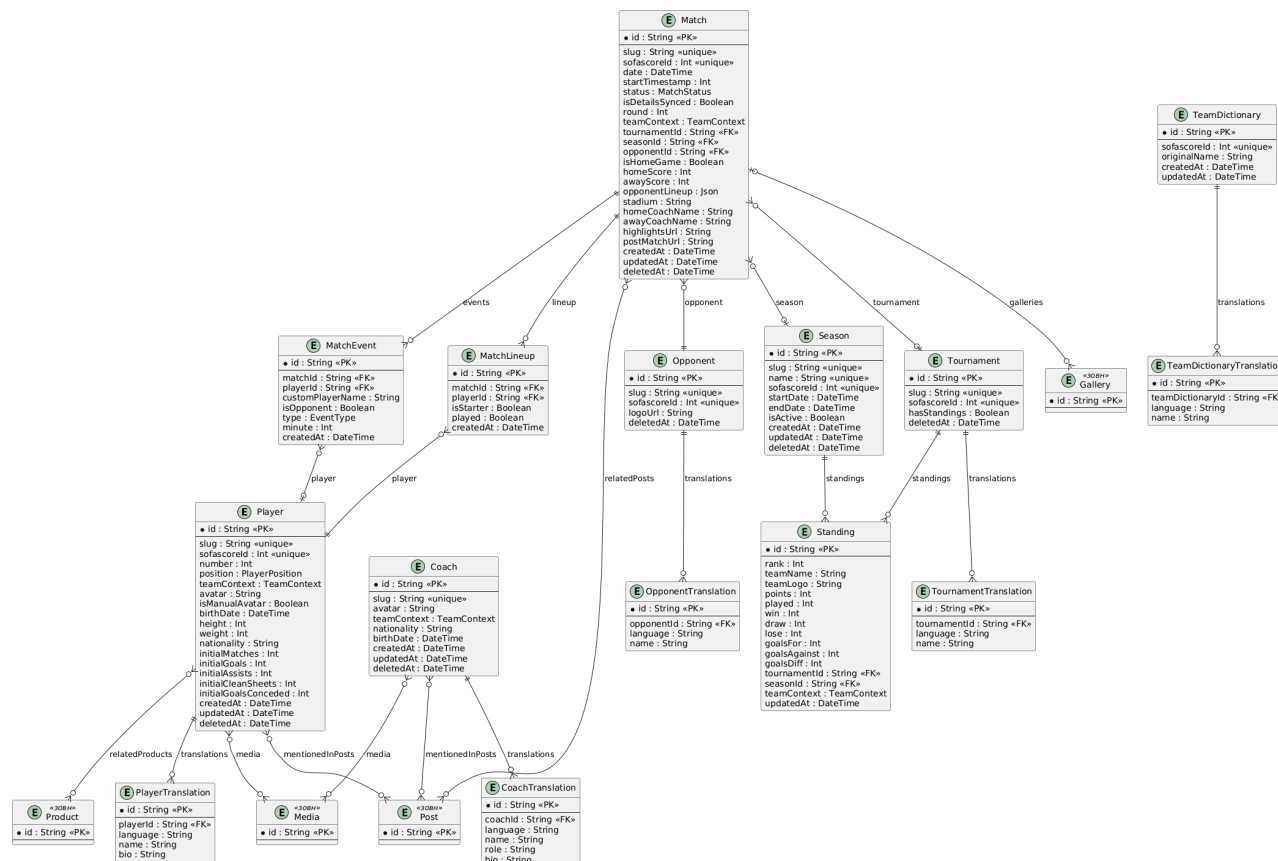


Рис. Д.2 – ER-діаграма бази даних (команда та матч-центр)

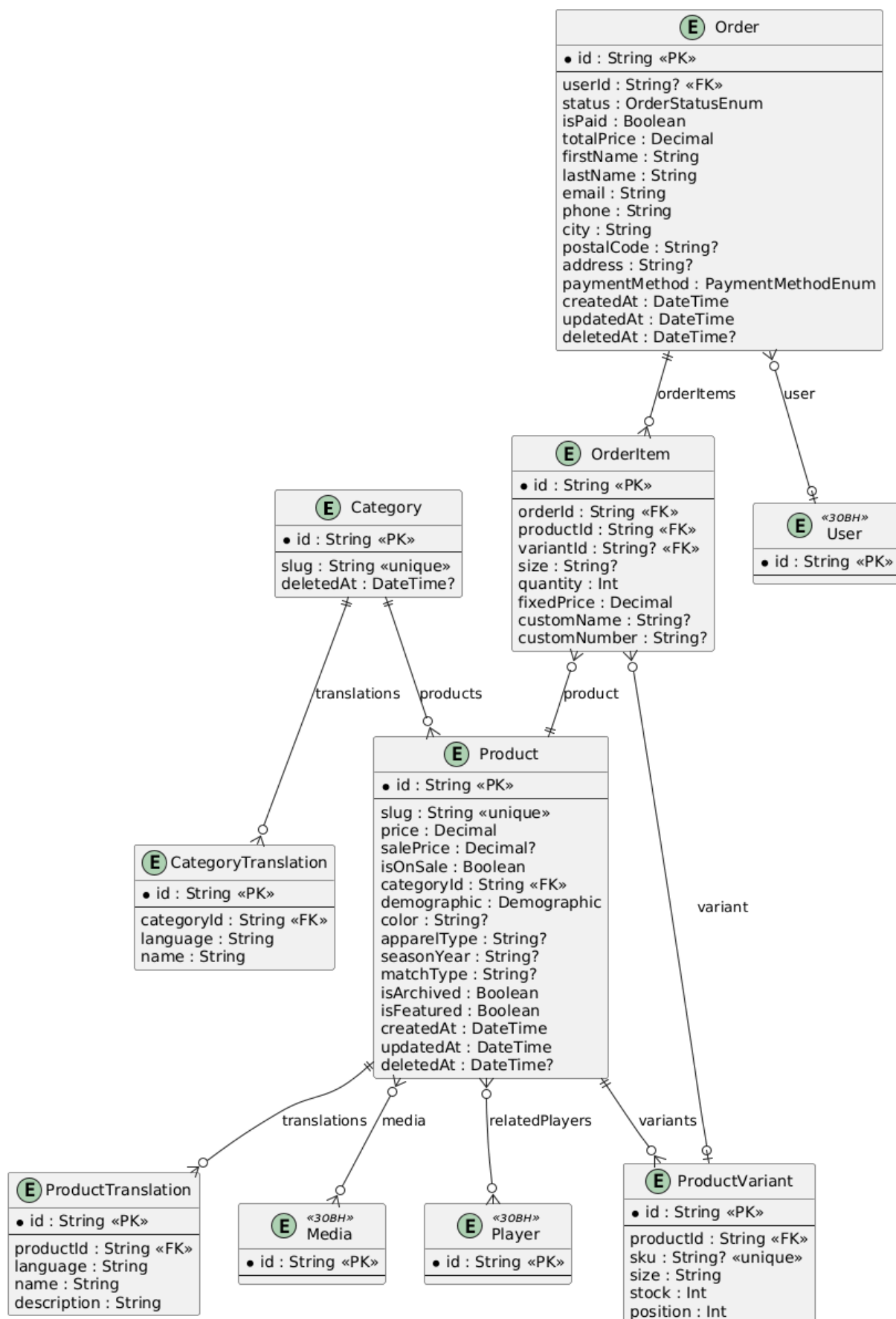


Рис. Д.3 – ER-діаграма бази даних (електрона комерція)

Діаграми варіантів використання системи

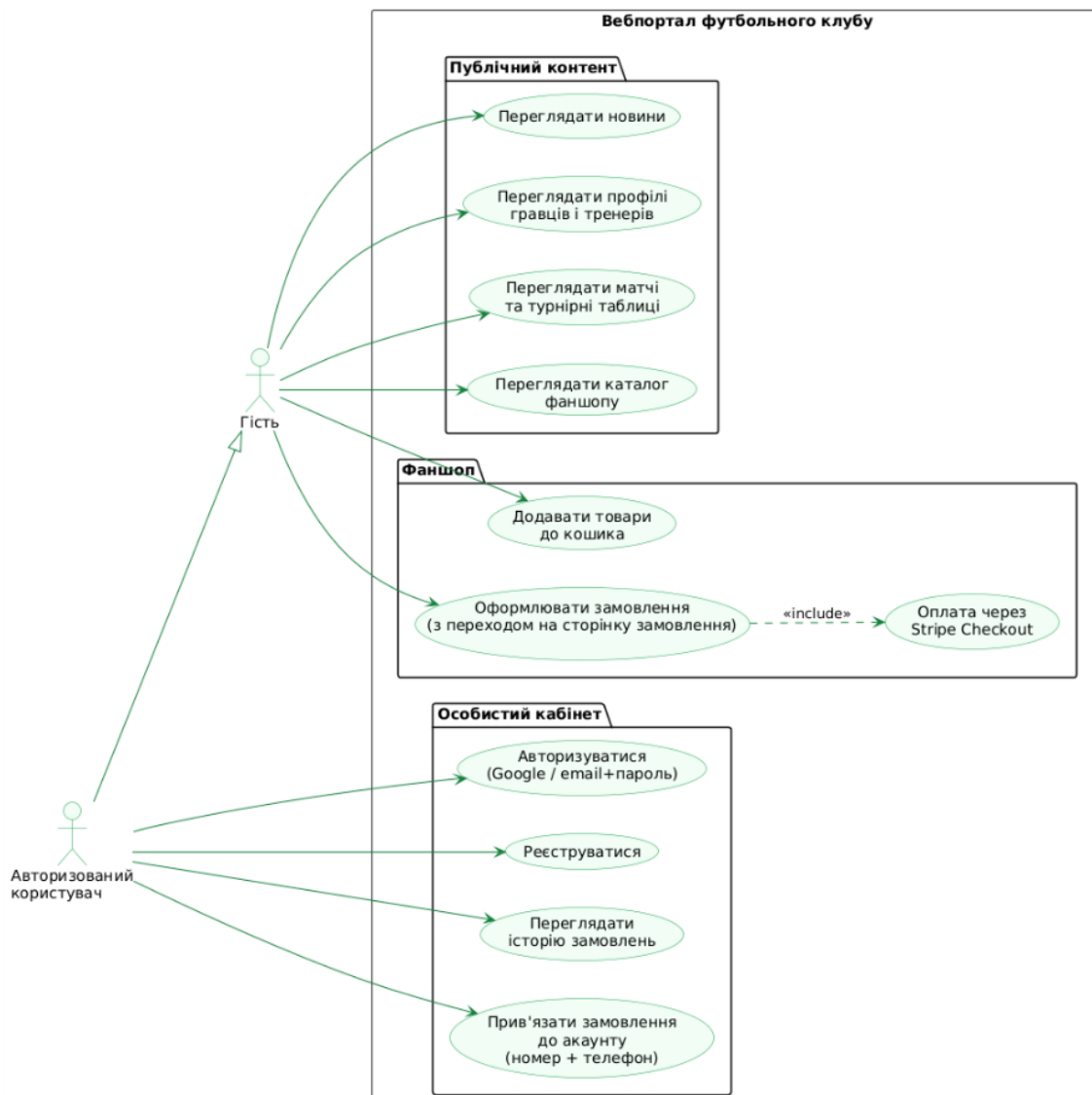


Рис. Ж.1 – Діаграма варіантів використання системи (публічна частина та особистий кабінет)

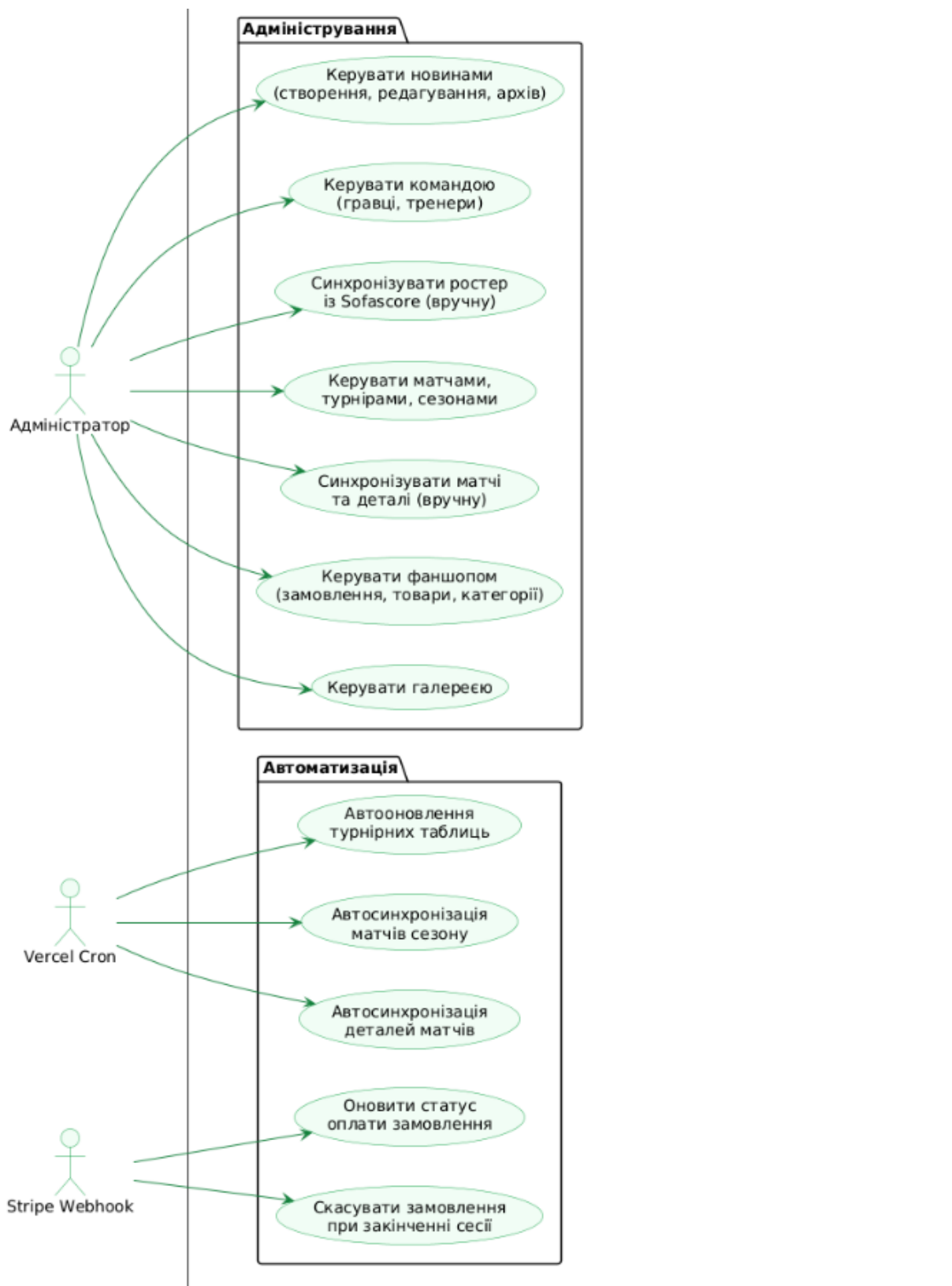


Рис. Ж.2 – Діаграма варіантів використання системи (адміністрування та автоматизація)

Діаграми активності

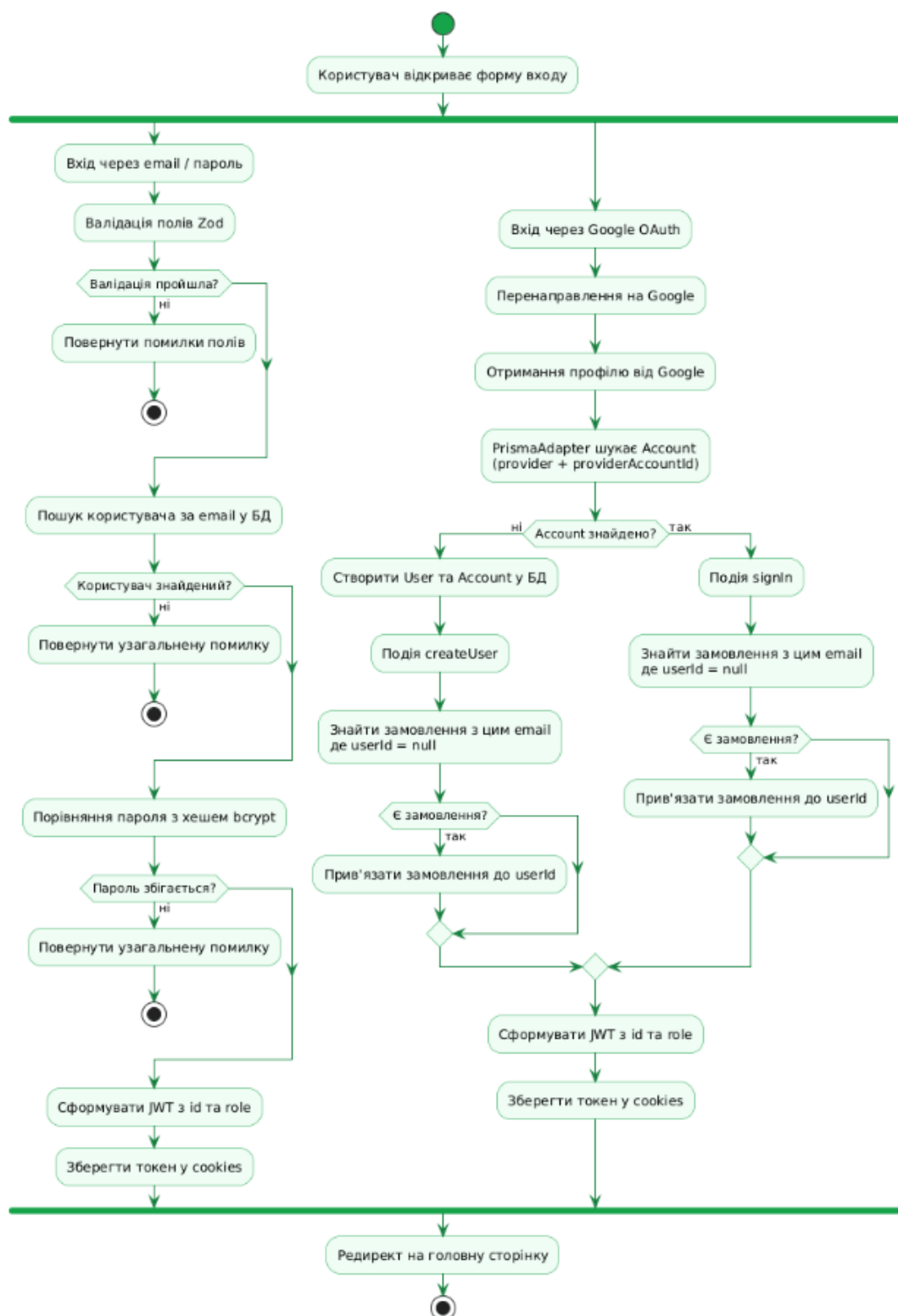


Рис. И.1 – Діаграма активності процесу автентифікації

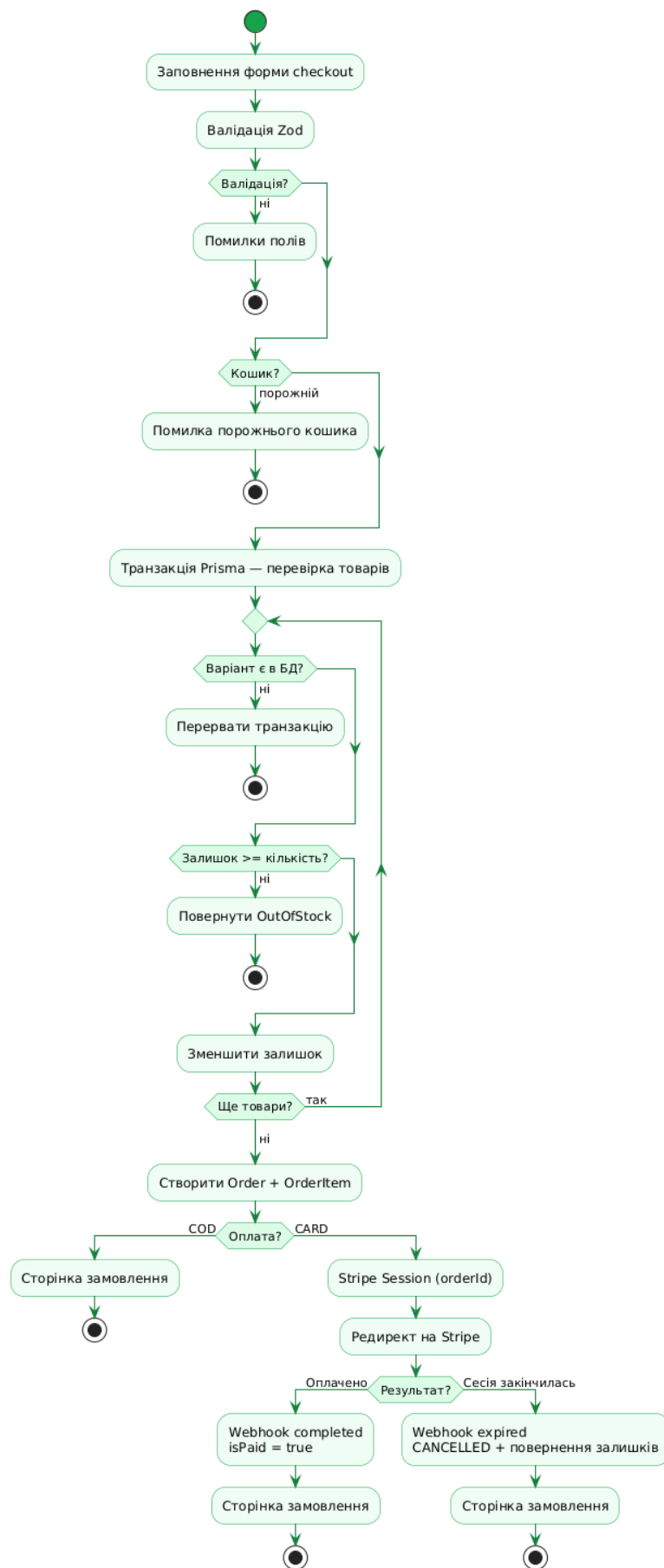


Рис. И.2 – Діаграма активності процесу оформлення замовлення

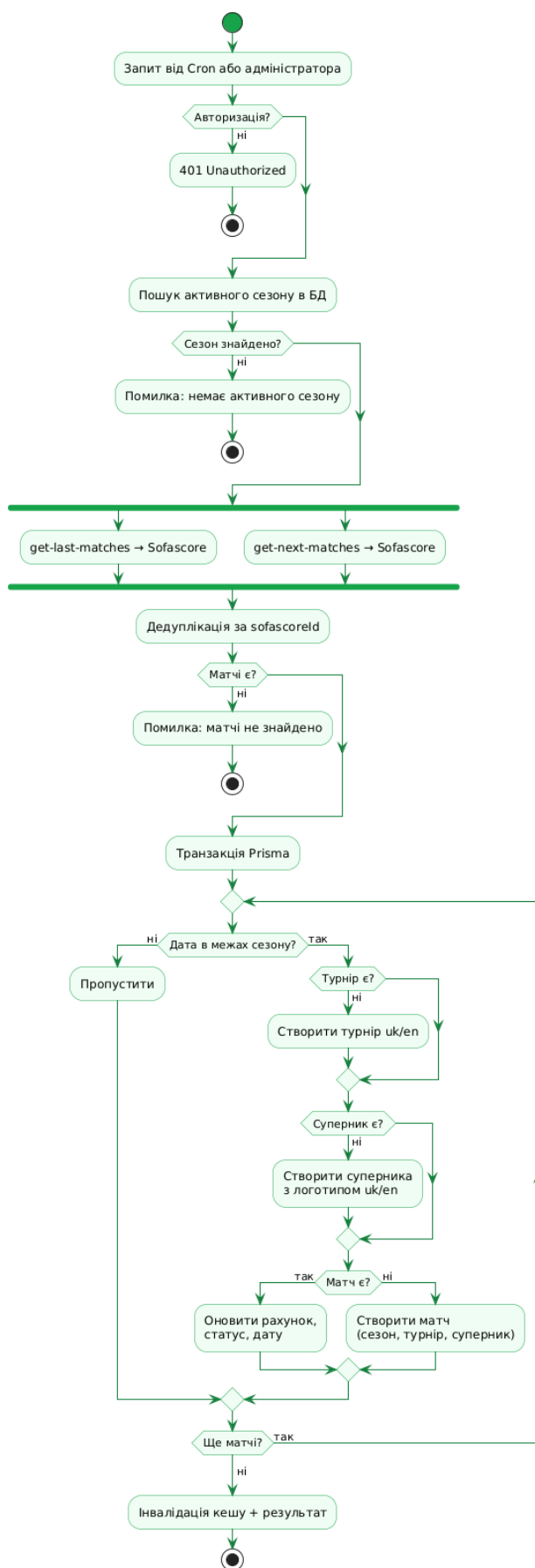


Рис. И.3 – Діаграма активності синхронізації матчів

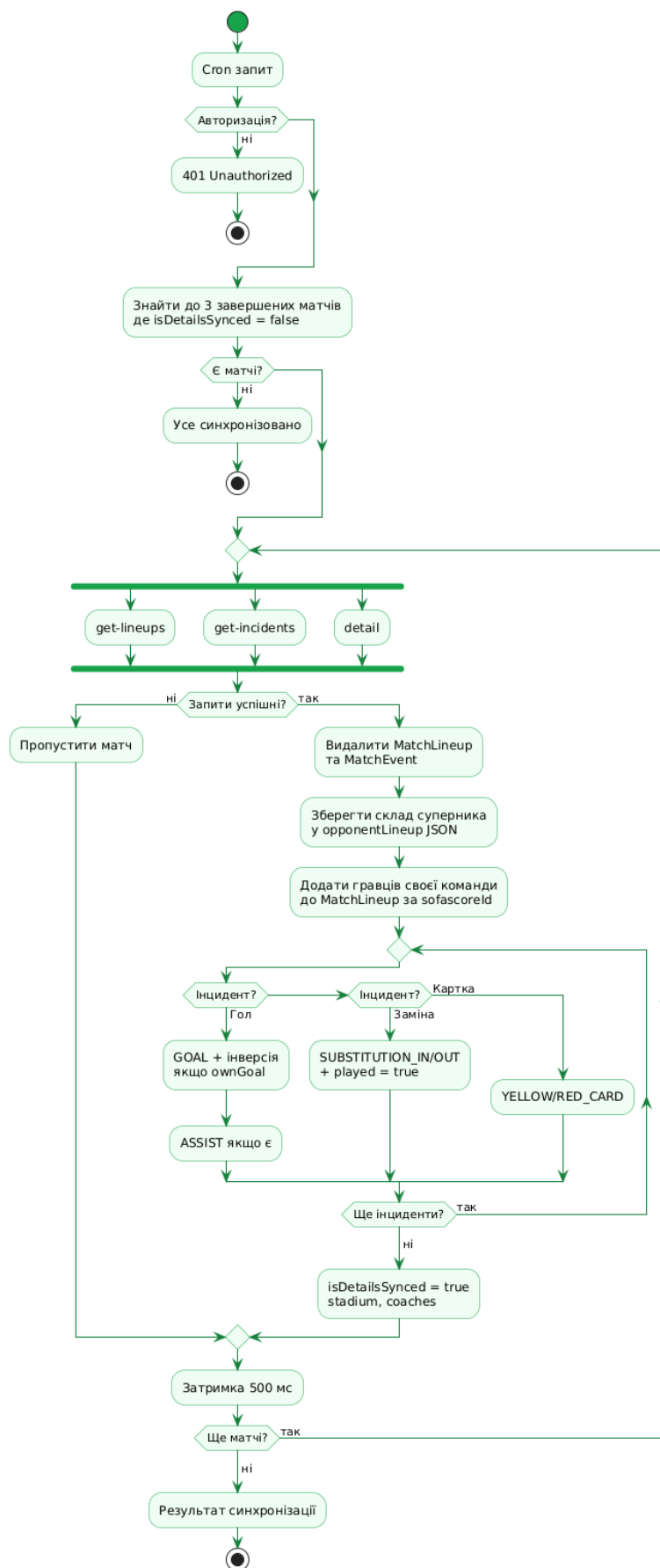


Рис. И.4 – Діаграма активності синхронізації деталей матчу

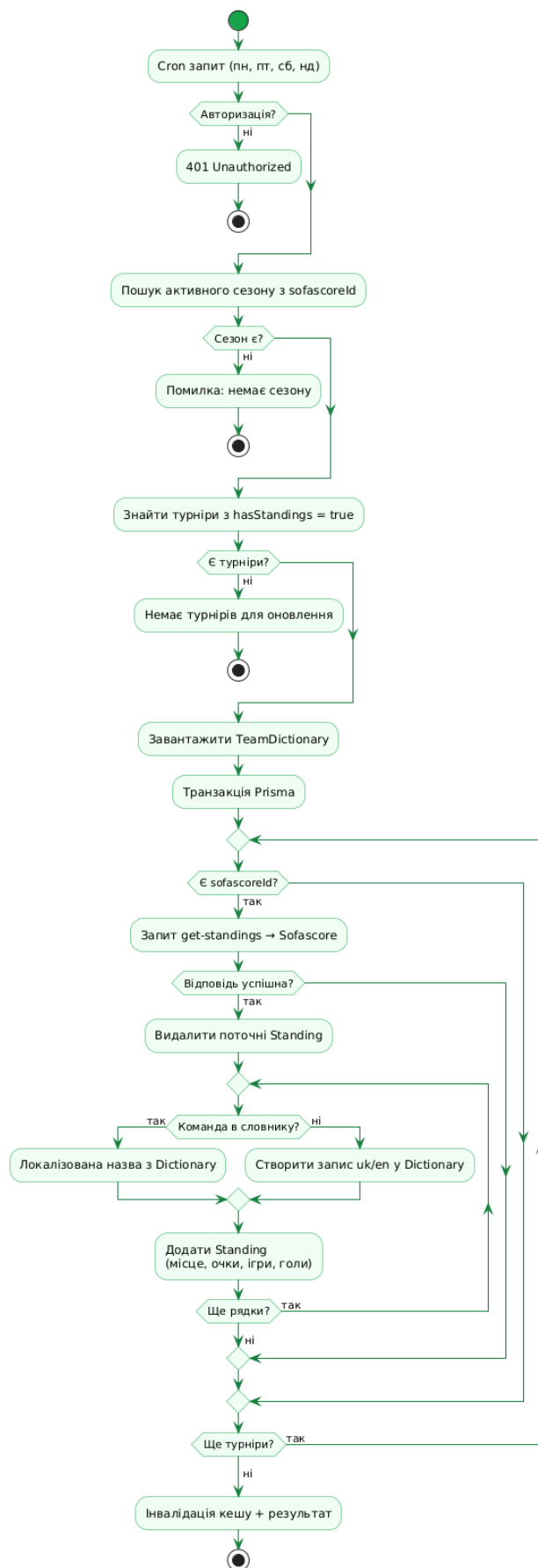


Рис. И.5 – Діаграма активності синхронізації турнірних таблиць

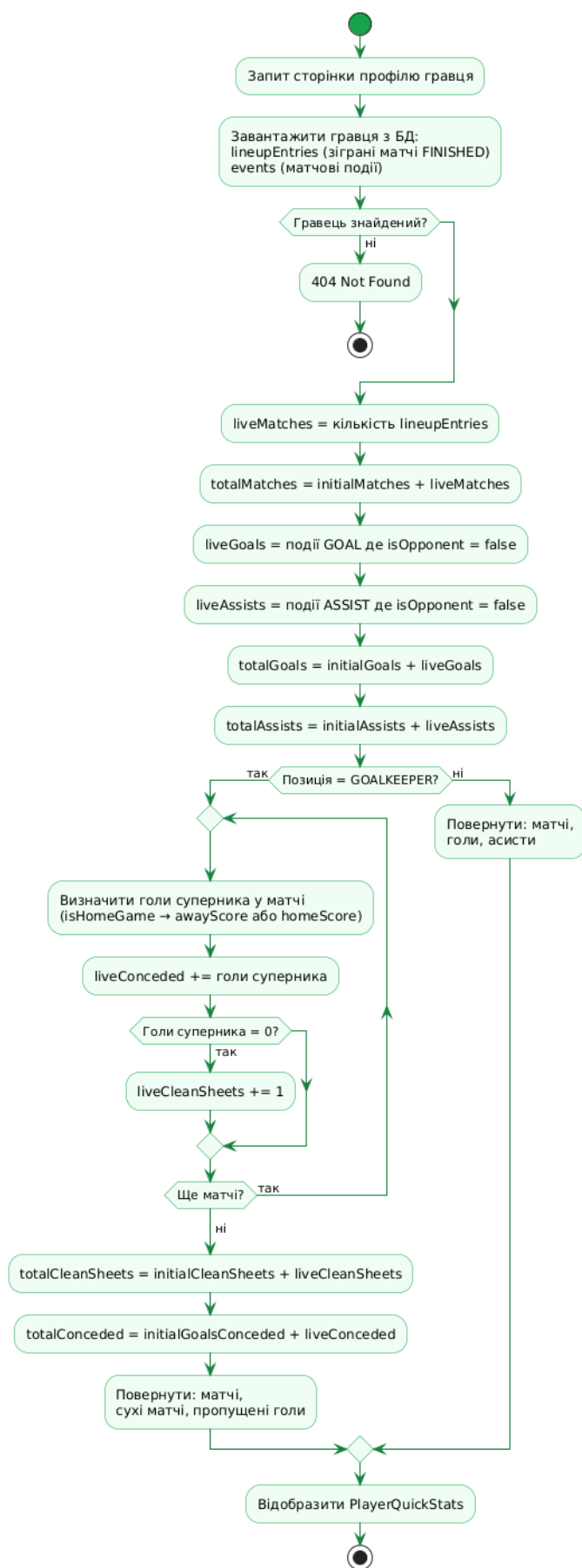


Рис. И.6 – Діаграма активності обчислення гібридної статистики

		Цивінський М.В.			ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
		Нерода С.І.				
Змн.	Арк.	№ докум.	Підпис	Дата		114

Діаграма класів підсистеми електронної комерції

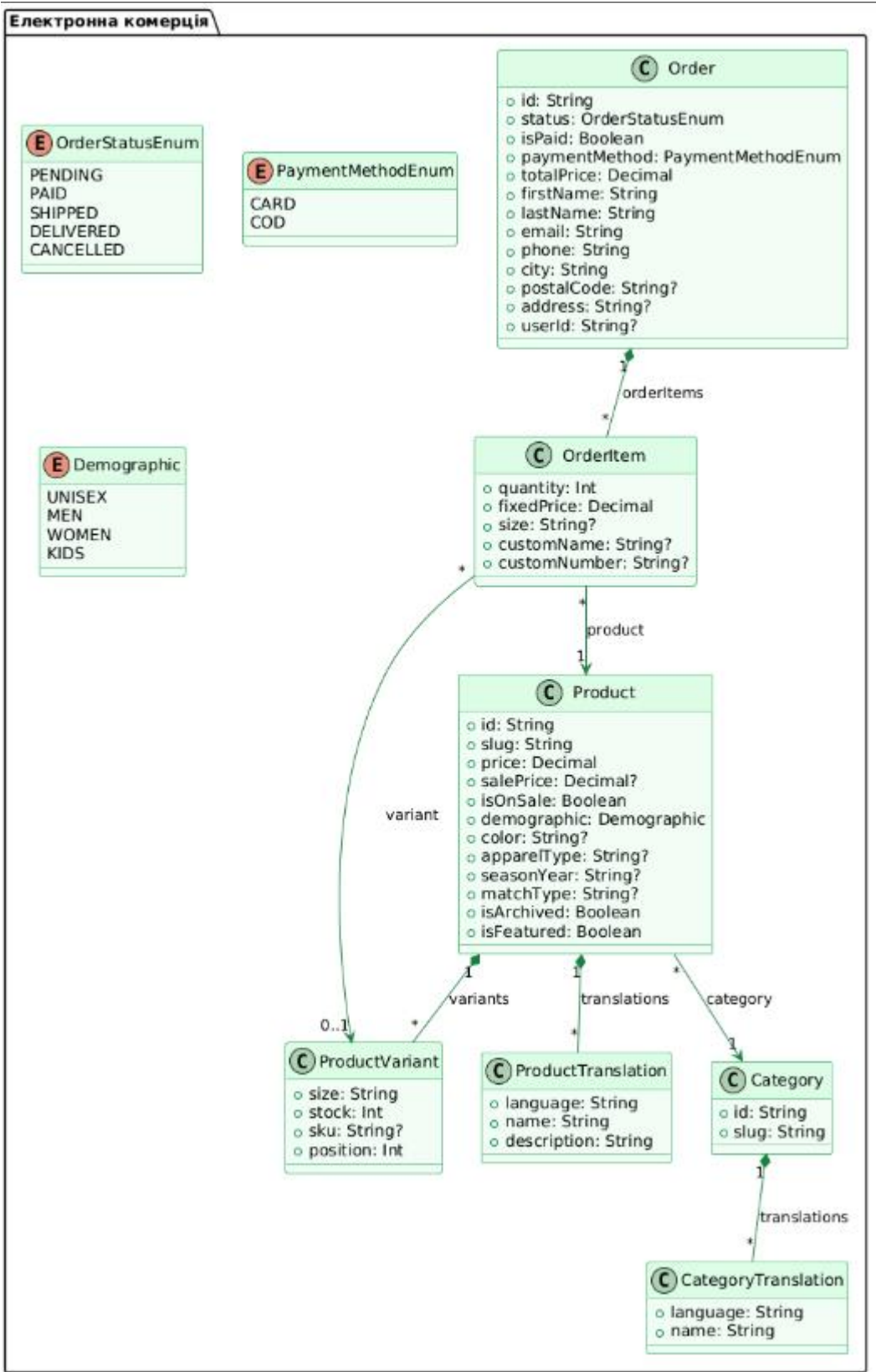


Рис. К.1 – Діаграма класів підсистеми електронної комерції