МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
КАФЕДРА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА

на тему: «Інформаційна система для кіберспортивних турнірів з дисципліни CS GO»

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

Виконавець роботи: студент групи ІТ.мз-11с Галич Антон Олексійович

Кваліфікаційну роботу захищено на засіданні ЕК з оцінкою _____ «___» грудня 2022 р.

Науковий керівник _____ к.т.н., доц., Парфененко Ю. В.
(підпис)

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

Студент _____
(підпис)

Суми-2022

Сумський державний університет
Центр заочної, дистанційної та вечірньої форм навчання
Кафедра інформаційних технологій
Спеціальність 122 «Комп'ютерні науки»
Освітньо-професійна програма «Інформаційні технології проектування»

ЗАТВЕРДЖУЮ
В.о. зав. кафедри ІТ

_____ С. М. Ващенко
«___» _____ 2022 р.

ЗАВДАННЯ

на кваліфікаційну роботу магістра студентові

Галич Антон Олексійович
(прізвище, ім'я, по батькові)

1 Тема проекту Інформаційна система для кіберспортивних турнірів з дисципліни CS GO
затверджена наказом по університету від «04» листопада 2022 р. № 1015-VI

2 Термін здачі студентом закінченого проекту «10» грудня 2022 р.

3 Вхідні дані до проекту інформація про турніри, інформація про користувачів та гравців у обрану дисципліну, додатковий графічний матеріал, загальна інформація за грою

4 Зміст розрахунково-пояснювальної записки (перелік питань, що їх належить розробити) вступ, аналіз предметної області, огляд останніх досліджень і публікацій, аналіз програмних продуктів кіберспортивної тематики, постановка задачі та методи дослідження,мета та задачі дослідження, розробка візуальної частини інформаційної системи, розробка функціональної частини інформаційної системи, моделювання та проектування, структура інформаційної системи для кіберспортивних турнірів, структурно-функціональне моделювання процес, моделювання діаграми варіантів використання інформаційної системи для кіберспортивних турнірів, проектування бази даних, практична реалізація інформаційної системи, програмна реалізація, використання інформаційної системи звичайним користувач, адміністрування інформаційної системи, висновки, список використаних джерел.

5 Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) IDEF0 системи для кіберспортивних турнірів, IDEF1 системи для кіберспортивних турнірів, діаграма варіантів використання, Пакети та модулі інформаційної системи для кіберспортивних турнірів з дисципліни CS GO, ERD-діаграмма

6. Консультанти випускної роботи із зазначенням розділів, що їх стосуються:

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
Аналіз кіберспортивної області	Парфененко Ю.В.		
Постановка задачі та вибір розробки	Парфененко Ю.В.		
Реалізація інформаційної системи	Парфененко Ю.В.		

Дата видачі завдання _____.

Керівник _____
(підпис)

Завдання прийняв до виконання _____
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів випускної проекту	Термін виконання етапів проекту	Примітка
1	Аналіз предметної області	10.09.22-15.09.22	
2	Постановку задачі	15.09.22-27.09.22	
3	Планування робіт	27.09.22-10.10.22	
4	Аналіз та виріб інструментів для роботи	10.10.22-22.10.22	
5	Проектування діаграм та моделей використання	22.10.22-27.10.22	
6	Розробка інформаційної системи для кіберспортивних турнірів з дисципліни CS GO	27.10.22-30.11.22	
7	Викладення інформаційної системи для кіберспортивних турнірів з дисципліни CS GO на хостинг	30.11.22-10.12.22	

Магістрант _____ Галич А.О.

Керівник роботи _____ к.т.н., доц. Парфененко Ю.В.

РЕФЕРАТ

Тема кваліфікаційної роботи магістра «Інформаційна система для кіберспортивних турнірів з дисципліни CS GO».

Практична значення інформаційної системи для кіберспортивних турнірів з дисципліни CS GO є спрощення організації та проведення турнірі. Це дозволить користувачам створювати власні команди та брати участь у турнірах.

В першому розділі проаналізовано кіберспортивну область та проведено розбір актуальних досліджень даної тематики. Також було проаналізовано публікацій та актуальних продуктів-конкурентів.

В другому розділі було поставлено головну мету розробки інформаційної системи по кіберспортивних турнірах. Після аналізу було обрано програмне забезпечення та інструментарій.

В третьому розділі було розроблено структуру та модель інформаційної системи, структурно-функціональне моделювання процесу, а також моделювання діаграми варіантів використання. Додатково було розроблено діаграму бази даних для взаємодії із даними.

В четвертому розділі було продемонстровано приклади використання інформаційної системи для кіберспортивних турнірів з дисципліни CS GO зі сторони звичайного користувача та адміністратора. Весь функціонал був продемонстрований й детальній варіації.

Результатом кваліфікаційної роботи магістра є розроблена інформаційна система для кіберспортивних турнірів з дисципліни CS GO.

Кваліфікаційна робота містить 108 сторінок, 6 таблиць, 98 рисунків, список літератури 23 найменувань, 2 додатки.

Ключові слова: інформаційна система, турніри, кіберспорт, команда, новини, комунікація, нотифікація, виграш, cs go, дисципліна, моделювання.

ЗМІСТ

ВСТУП 6
1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ 8
1.1 Огляд останніх досліджень і публікацій 8
1.2 Аналіз програмних продуктів кіберспортивної тематики 10
2 ПОСТАНОВКА ЗАДАЧІ ТА МЕТОДИ ДОСЛІДЖЕННЯ 23
2.1. Мета та задачі дослідження 23
2.2. Розробка візуальної частини інформаційної системи 24
2.3. Розробка функціональної частини інформаційної системи 26
3 МОДЕЛЮВАННЯ ТА ПРОЕКТУВАННЯ 28
3.1 Структура інформаційної системи для кіберспортивних турнірів 28
3.2 Структурно-функціональне моделювання процесу 29
3.3 Моделювання діаграми варіантів використання інформаційної системи для кіберспортивних турнірів 31
3.4. Проектування бази даних 35
4 ПРАКТИЧНА РЕАЛІЗАЦІЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ 43
4.1 Програмна реалізація 43
4.2 Використання інформаційної системи звичайним користувачем 49
4.3 Адміністрування інформаційної системи 65
ВИСНОВКИ 72
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 73
ДОДАТОК А 76
ДОДАТОК Б 85

ВСТУП

У сучасному світі технології становляться популярними в багатьох сферах. Також значних змін та розвитку зазнала сфера комп’ютерних ігор. Разом з цим з’явився новий популярний напрямок – кіберспорт.

Кіберспорт із досить маргінальної розваги кінця 90-х років. минулого століття перетворився на широковідомий прибутковий медіапродукт. Даний напрямок приваблює мільйони глядачів та є популярним у всіх країнах світу.

Наростаюча комерціалізація кіберспорту привернула до цієї галузі багато фірм. Технологічний прогрес сприяв зростанню кіберспорту протягом багатьох років, зокрема таких технологічних підприємств, як Weaver, які використовували AI, 8K, VR, AR та інші технології, щоб надати фанатам кіберспорту нові способи перегляду.

Розвиток технологій, як-от зростання потокової передачі та покращення з’єднань, був критично важливим елементом зростання сектора. Однак інвестиції компаній також були рушійною силою поширення конкурентних ігор. Багато найкращих платформ онлайн-казино співпрацюють із кіберспортивними командами та додають їхні матчі до спортивних букмекерів, дозволяючи фанатам ігор грати разом зі своїми улюбленими командами.

В топ кіберспортивних напрямків входить й командний шутер від першої особи, а саме Counter-Strike: Global Offensive. Найкращі команди, найсильніші гравці та більшість уболівальників даного напрямку перебувають у Європі. Даний факт спрощує комунікацію та взаємодію людей в рамках кіберспортивних турнірів з дисципліни CS GO.

Даний напрям також є розповсюдженим серед комп'ютерних гравців України. Для спрощення організації та проведені кіберспортивних турнірів з дисципліни CS GO в Україні було прийнято рішення по розробці інформаційної системи.

Отже, сформовано тему роботи бакалавра, мету, об’єкти та предмети дослідження.

Об’єкт дослідження: процес інформаційної підтримки проведення кіберспортивних змагань з дисципліни CS GO.

Предмет дослідження: методи організації взаємодії користувачів один з одним за допомогою інформаційної системи організації кіберспортивних турнірів.

Метою роботи є розробка інформаційної системи для спрощення організації та проведення турнірів CS GO. Реалізація всього необхідного переліку функціонала забезпечить полегшення пошуку членів команди для участі в турнірах, розповсюдження та популяризацію даної кіберспортивної дисципліни.

Для досягнення мети потрібно виконати такі задачі:

– провести аналіз предметної області;

– виконати постановку задачі;

– виконати проектування всіх необідних діаграм та моделей використання;

– розробити інформаційну систему для кіберспортивних турнірів з дисципліни CS GO та провести її тестування.

Практичне значення: Даний напрям також є розповсюдженим серед комп’ютерних гравців України. Реалізація дипломної роботи та розробка інформаційної системи для кіберспортивних турнірів з дисципліни CS GO допоможе в спрощенні організації та проведенні турнірів в Україні. Також, вагомим фактором є те, що користувачі системи отримують можливість взаємодіяти один з одним.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Огляд останніх досліджень і публікацій

Кіберспорт, або як ще називають цей напрямок – електронні види спорту, є логічною еволюцією змагальних онлайн-ігор. Дисципліна починала розвиватися лише як нішева індустрія, але за останнє десятиліття зайняла не останнє місце в індустрії спорту та розваг [1-2]. Кіберспорт також став частиною основної культури. Глядачі можуть спостерігати, як гравці відеоігор змагаються один проти одного у віртуальному середовищі, як вони дивляться професійні спортивні змагання.

Потенціал заробляти на іграх стає все більш спокусливим, оскільки існує кілька способів зробити це. Найбільш очевидним є призовий фонд від турнірів [4]. Крім того, кіберспортсмени також можуть заробити мільйони доларів, транслюючи свій ігровий процес у прямому ефірі, підписуючи контракти з такими великими організаціями чи спонсорські угоди. з організаціями. Ці угоди можуть бути, як у сфері ігор, так й суміжні з цим [3, 5-7]. Від даної дисципліни можуть отримати вигоду гравці, глядачі та популярні корпорації.

До теми кіберспорту можна віднести й інші пов’язані із цим технологічні напрямки. Їх можна розділити на:

1. Платформи та їх інфраструктура. Прикладом таких, що використовуються й для проведення турнірів є Battlefy. До комунікаційних інструментів відносять популярні додатки TeamSpeak та Discord, які значно полегшують планування та проведення організаційних заходів пов’язаних з кіберспортом.

2. Ресурси прямої трансляції. До таких відносять перш за все платформи для прямої трансляції, які включають й аналітичні дані. Також, даний ресурс використовується стримерами для монетизації відеоінформації. Зокрема, Twitch є одним із найпопулярніших потокових сайтів для кіберспорту.

3. Ринки предметів. Фантастичні ліги з професійними спортсменами допомогли ринкам предметів, в залежності від гри, значно зросли за останні кілька років. Хорошим прикладом сервісу, що продає скіни, ігрові елементи є dMarket.

4. Ресурси для початківців професійних гравців і шанувальників: це може бути все, починаючи від джерел новин, відстеження галузевих статистичних даних, коучингів й інструментів для вдосконалення навичок для професійних і звичайних гравців. Такі веб-сайти, як Game Rant і PC Gamer, допомагають людям бути в курсі останніх новин ігрової індустрії. Такі сайти, як metafy.gg, пропонують тренінги від найкращих професіоналів гри для тих, хто хоче вдосконалюватися.

В основі розвитку кіберспорту лежить технологічний прогрес [8,9]. Даний напрямок став стимулятором в розвитку таких напрямків як оновлення ігрового обладнання, віртуальна реальність та штучний інтелект (табл.1.1).

Таблиця 1.1 – Детальна інформація по напрямках

Назва	Детальний опис за напрямками
Оновлення ігрового обладнання	Процесори, монітори, пристрої введення, аудіосистеми, мобільні телефони, потокове обладнання та інші спеціалізовані апаратні пристрої є частиною еволюції апаратного забезпечення [10].
Віртуальна реальність (VR) і доповнена реальність (AR)	Завдяки першокласним удосконаленням для захоплюючих і захоплюючих ігрових вражень, AR і VR мають намір внести кардинальні зміни в сектор кіберспорту [11-13]. Ці нові технологічні досягнення допомогли в розробці 3D ігрових ефектів.
Штучний інтелект	Люди ніколи б не припустили, що ШІ відіграватиме таку важливу роль в ігровій індустрії. Ця передова технологія зараз починає справляти вплив. Рішення на основі штучного інтелекту дозволяють приймати рішення на основі даних, переглядаючи та аналізуючи змагання з кіберспорту.

Попри всю риторику про те, що звичайні види спорту запроваджують більше інновацій, кіберспортивні технології можна використовувати так, як мало хто з інших форм розваг [14]. Це також один із небагатьох видів спорту, у яких більшість людей виросла з Інтернетом і очікує персоналізованих матеріалів у своїх руках. Використання технологій є зв'язком між досягненням очікувань шанувальників і забезпеченням зв'язку між компаніями та фанатами.

1.2 Аналіз програмних продуктів кіберспортивної тематики

Для аналізу кіберспортивної тематики, а саме веб-сайтів для організації кіберспортивних турнірів з дисципліни CS GO було обрано: «Gameinside.ua» [15], «Upea.com.ua» [16] та «Uesf.org.ua» [17].

Розглянемо веб-додаток «GameInside.ua». Даний веб-додаток являє собою інформаційну сторінку агентства, що надає консультаційні та організаційні послуги у сфері кіберспорту.

На рис. 1.1 представлено головну сторінку. На головній сторінці «GameInside.ua» представлена коротка інформація про найпопулярніші статі та тематики додатку.

Крім того, на кожній сторінці додана реклама. Не дивлячись на те, що так чи інакше належить до кіберспортивної тематики, це створює інформаційний шум на сторінках додатку, що показано на рис. 1.2. Адже коли користувач переходить до сторінки із детальною інформацією щодо обраної новини, реклама праворуч від теми значно відволікає. При розробці інформаційної системи для кіберспортивних турнірів даний момент буде врахований при розробці дизайну.

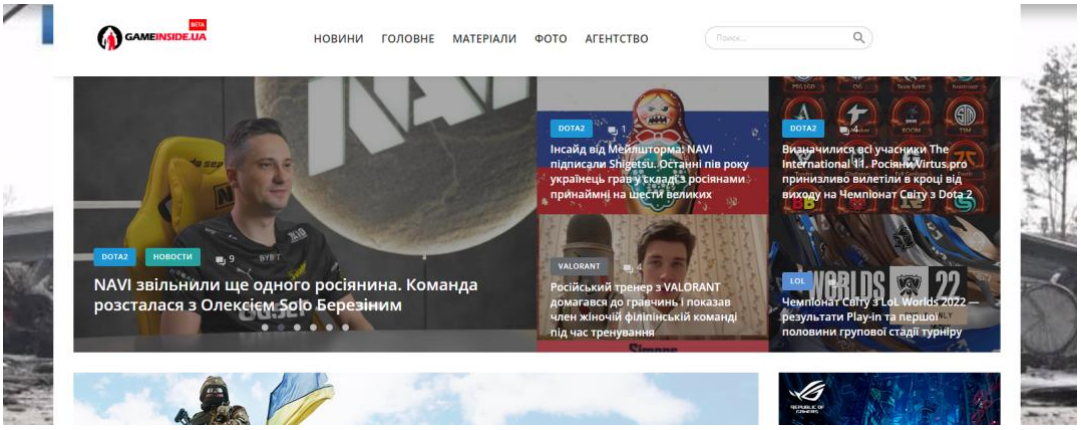

Рисунок 1.1 – «GameInside.ua» головна сторінка

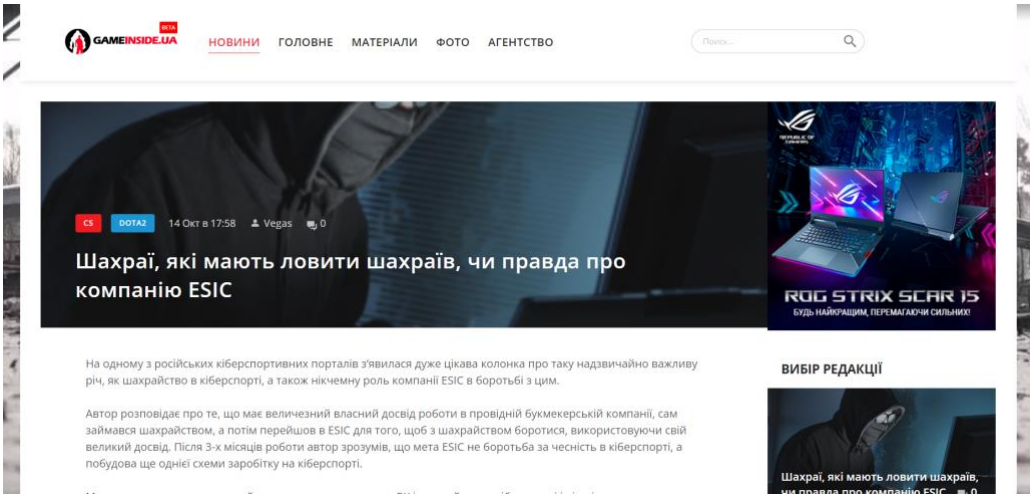

Рисунок 1.2 – «GameInside.ua» детальна інформація статі

До переваг додатку «GameInside.ua» можна віднести можливість додавання коментарів (рис.1.3).

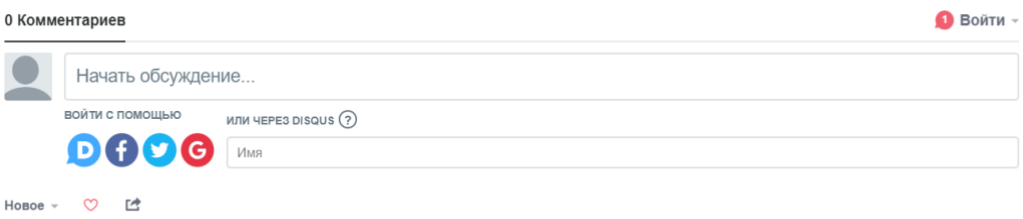

Рисунок 1.3 – «GameInside.ua» форма для коментарів

Не дивлячись на це, коментарі, що вже є в додатку залишаються без відповіді протягом довгого часу (рис.1.4). У результаті можна зробити висновок, що адміністратори додатку «GameInside.ua» не активні.

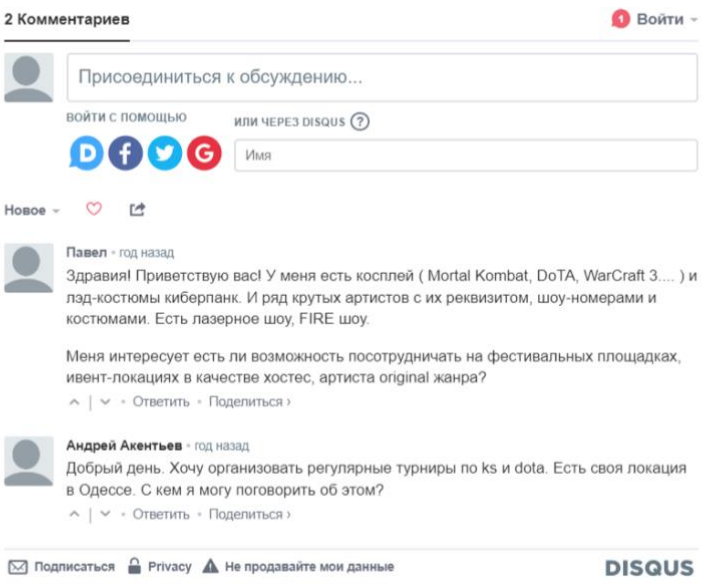

Рисунок 1.4 – «GameInside.ua» приклад коментарів

На рис. 1.5 представлено галерея та розгорнутий формат (рис.1.6) обраного прикладу з галереї. Кожна із обраних відкритих прикладів галереї має також короткий опис фотографій.

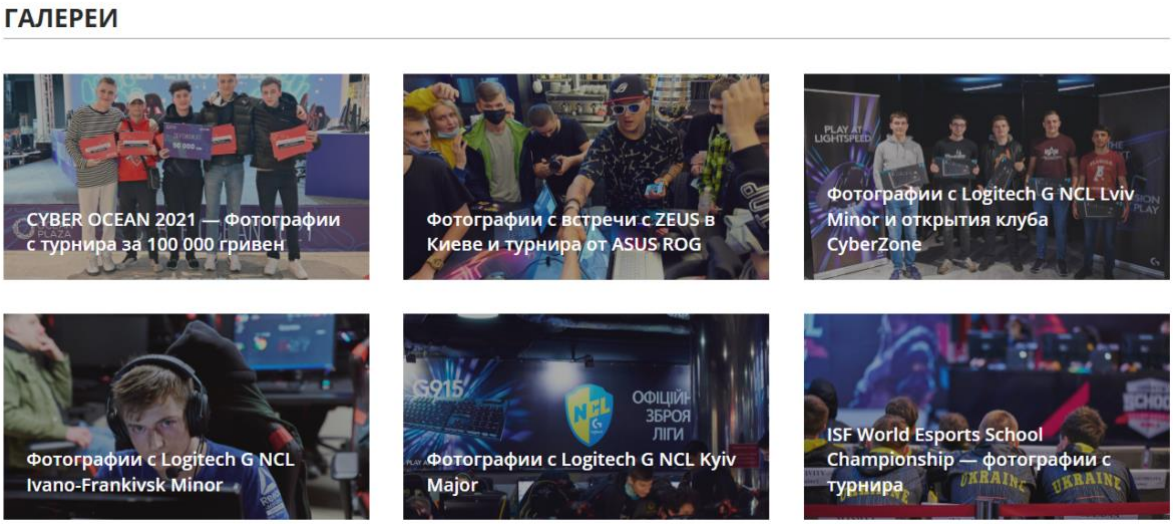

Рисунок 1.5 – «GameInside.ua» галерея

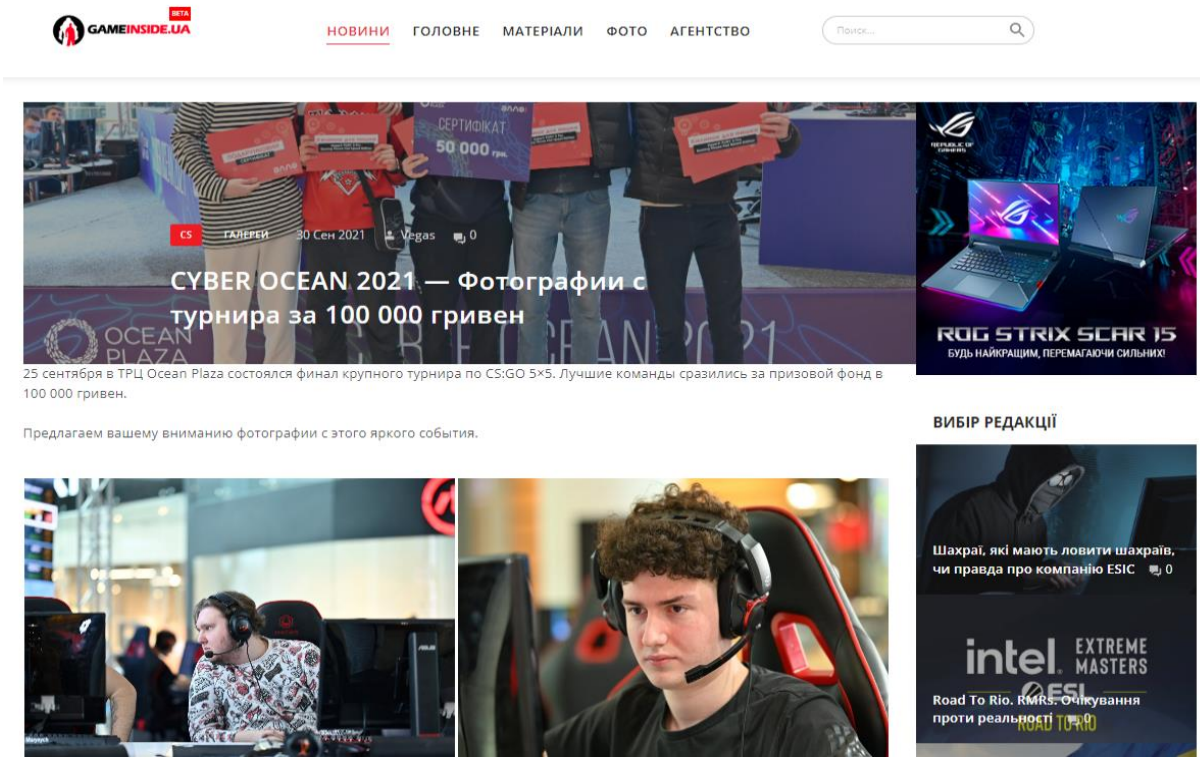

Рисунок 1.6 – «GameInside.ua» відкрита сторінка галереї

На рис. 1.7 представлена загальна інформація по «GameInside.ua». Саме на цій сторінці додатку можна дізнатися детальніше про розробників.

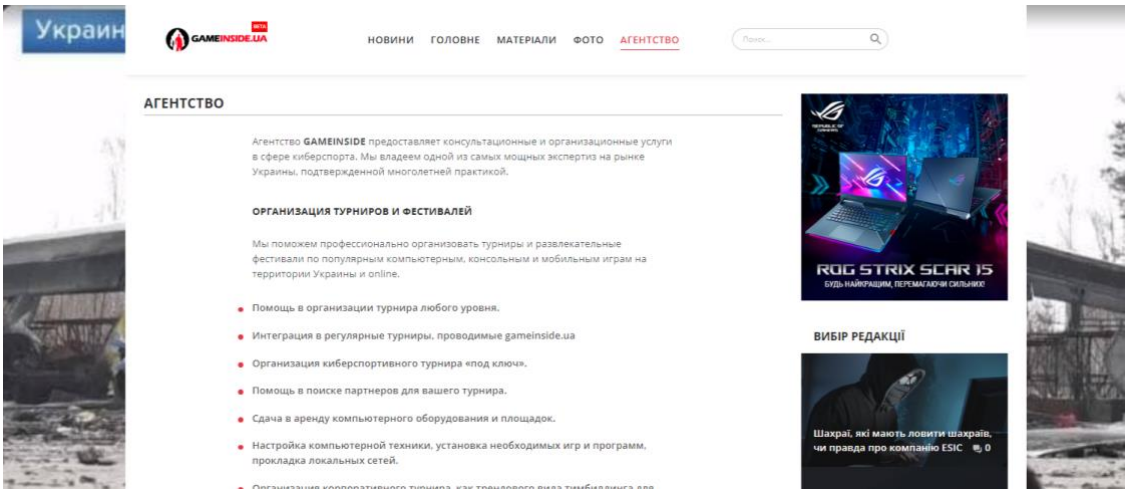

Рисунок 1.7 – «GameInside.ua» контактна інформація

Розглянемо веб-додаток «Upea.com.ua». Даний додаток є більш широкий, оскільки він розрахований на більше кіберспортивних дисциплін. На рис. 1.8 представлена головна сторінка додатку «Upea.com.ua».

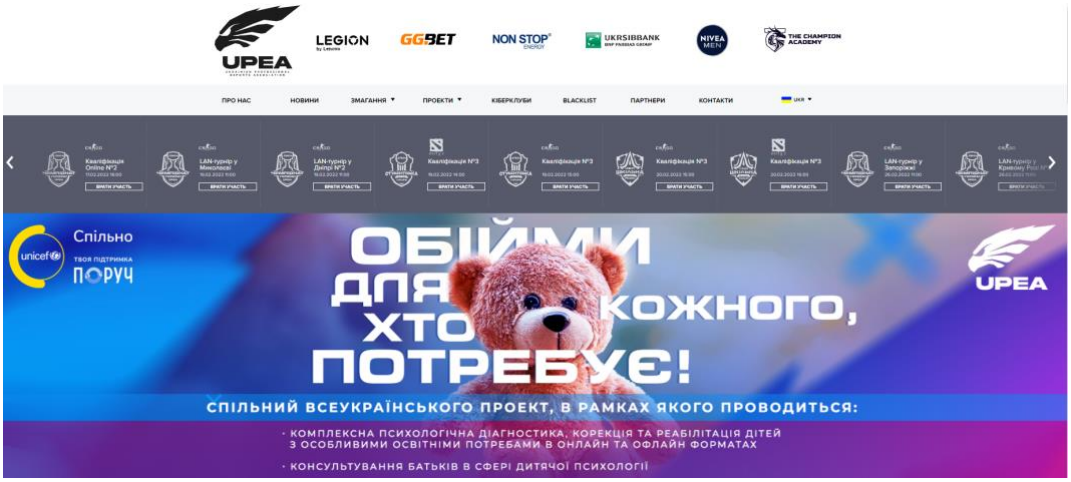

Рисунок 1.8 – «Upea.com.ua» головна сторінка

На ній представлення коротка інформація щодо останніх новин, календар змагань (рис.1.9) та інші проекти власників (рис.1.10).

Рисунок 1.9 – «Upea.com.ua» календар змагань

Рисунок 1.10 – «Upea.com.ua» інформація про проекти

Хотілось би також виділити те, що розробники виділили новини на окрему сторінку (рис.1.11). Дане рішення є досить зручним та допомагає користувачеві швидко зорієнтуватися та віднайти потрібно інформацію по кіберспорту.

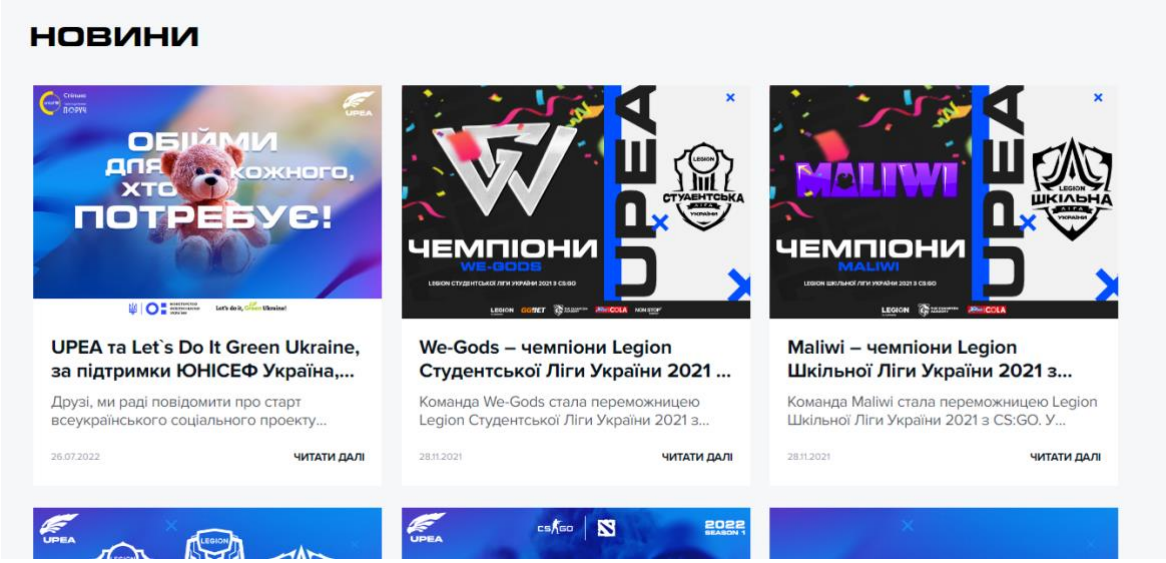

Рисунок 1.11 – «Upea.com.ua» сторінка новин

Оскільки, «Upea.com.ua» проводить змагання з декількох дисциплін, ми можемо перейти на сторінку певних змагань. Прикладом є рис. 1.12, де зображено приклад оформлення змагання з дисципліни CS GO.

Рисунок 1.12 – «Upea.com.ua» головна сторінка змагань з дисципліни CS GO

На даній сторінці зручно розміщена інформація, щодо кроків для реєстрації та участі у змаганнях кіберспортивних турнірів. При реєстрації користувач повинен обрати турнір та заповнити всю необхідну інформацію про себе чи команду (рис.1.13-14).

Рисунок 1.13 – «Upea.com.ua» вибір турніру

Головним недоліком є те, що користувачі для реєстрації повинні заповнити гугл-форму. На мою думку, це досить не зручно, адже виконується вихід з сайту.

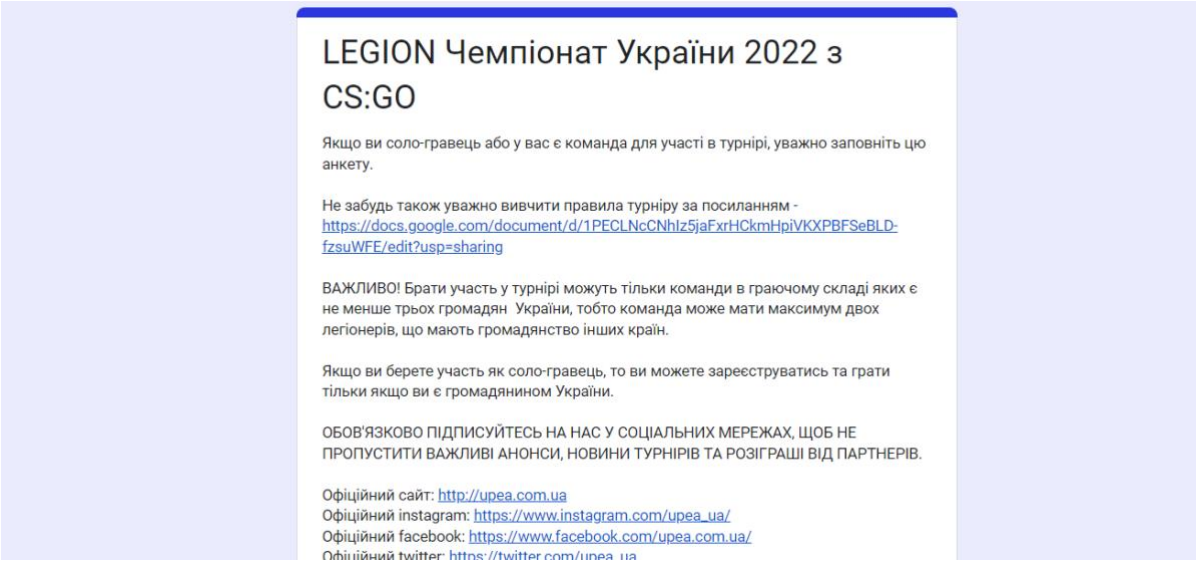

Рисунок 1.14 – «Upea.com.ua» сторінка для реєстрації на турнір

До переваг даного додатку можна віднести інформацію про тренувальні центри з кіберспорту в Україні (рис.1.15). Для пошуку є карта, посилання на соціальні мережі та приблизна вартість за кожним закладом.

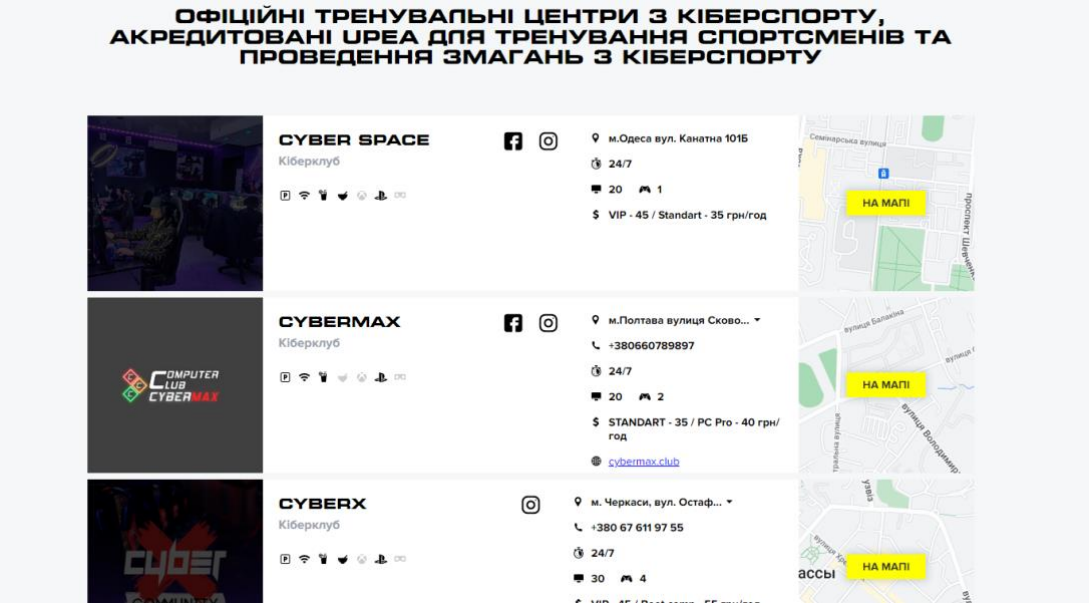

Рисунок 1.15 – «Upea.com.ua» карта закладів для тренування з кіберспорту

«Upea.com.ua» має цікавий функціонал для того, щоб поскаржитися на інших гравців чи команди, що порушують правила спільноти. Для блокування можна подати заявку з інформацією на сторінці «Blacklist» (рис.1.16).

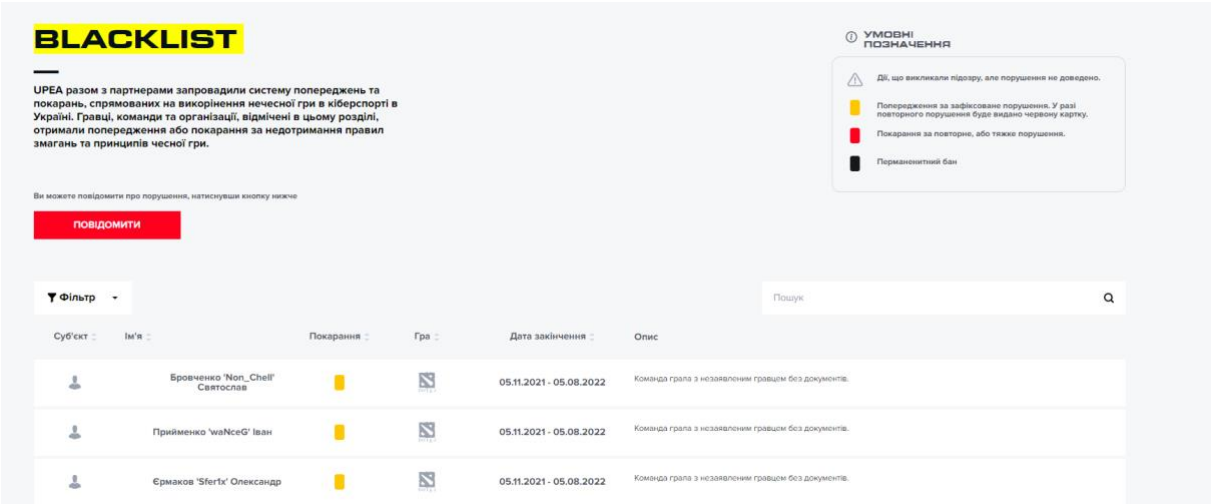

Рисунок 1.16 – «Upea.com.ua» сторінка для подання скарг

Не менш відомим є «Uesf.org.ua». Даний додаток також використовується для перегляду й пошуку кіберспортивних змагань з двох дисциплін. На рис. 1.17 представлено головну сторінку, де користувач має можливість зареєструватися для отримання актуальної інформації першим.

Також на даній сторінці представлено рейтинг команд з кожної дисципліни. На жаль, переглянути детальніше можливості немає.

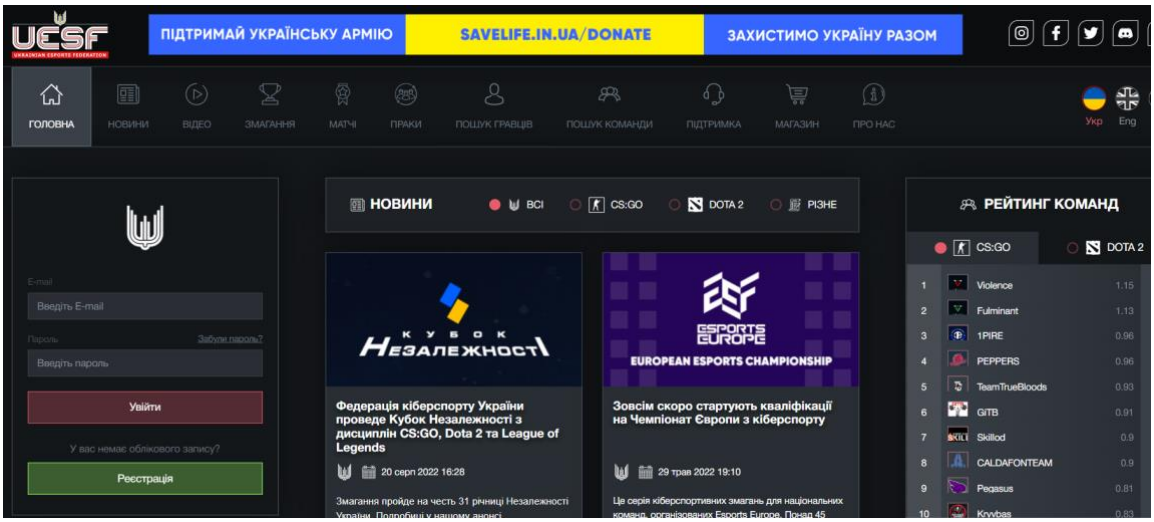

Рисунок 1.17 – «Uesf.org.ua» головна сторінка

Крім того, новини можна переглянути як на головній, так й на окремо виділеній для цього сторінці (рис.1.18). До переваг додатку також хочу віднести – багатомовність. Користувач має можливість перемикатися з однієї мови на іншу.

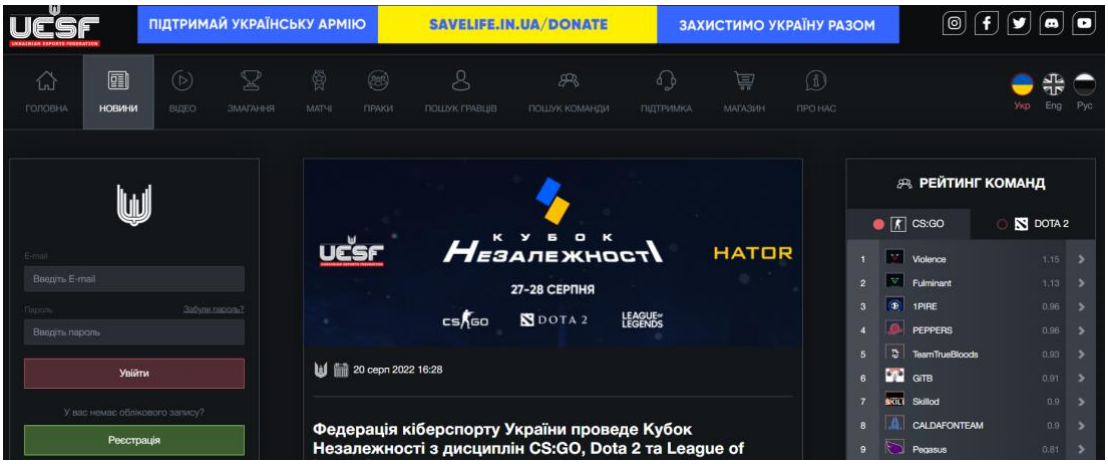

Рисунок 1.18 – «Uesf.org.ua» сторінка новин

Із недоліків, переглядати коментар та залишати їх під новинами чи іншими дописами можуть лише зареєстровані користувачі (рис.1.19). На мою думку, це є не зручним для користувача функціоналом.

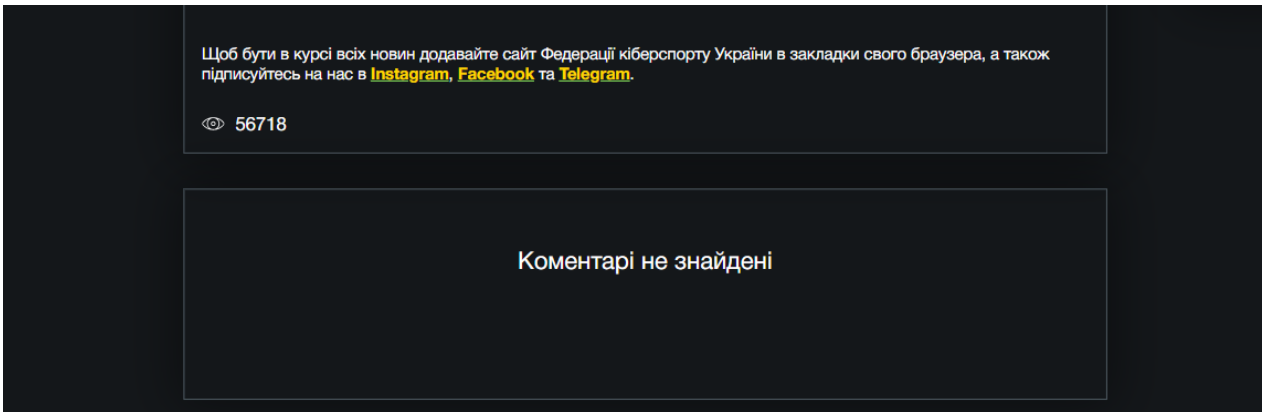

Рисунок 1.19 – «Uesf.org.ua» коментарі під новинами

Переходячи на сторінку зі змаганнями, користувач може переглянути інформацію як про майбутні, так і про минулі кіберспортивні новини. На рис. 1.20 та 1.21 представлені приклади відображення різного результату фільтрації.

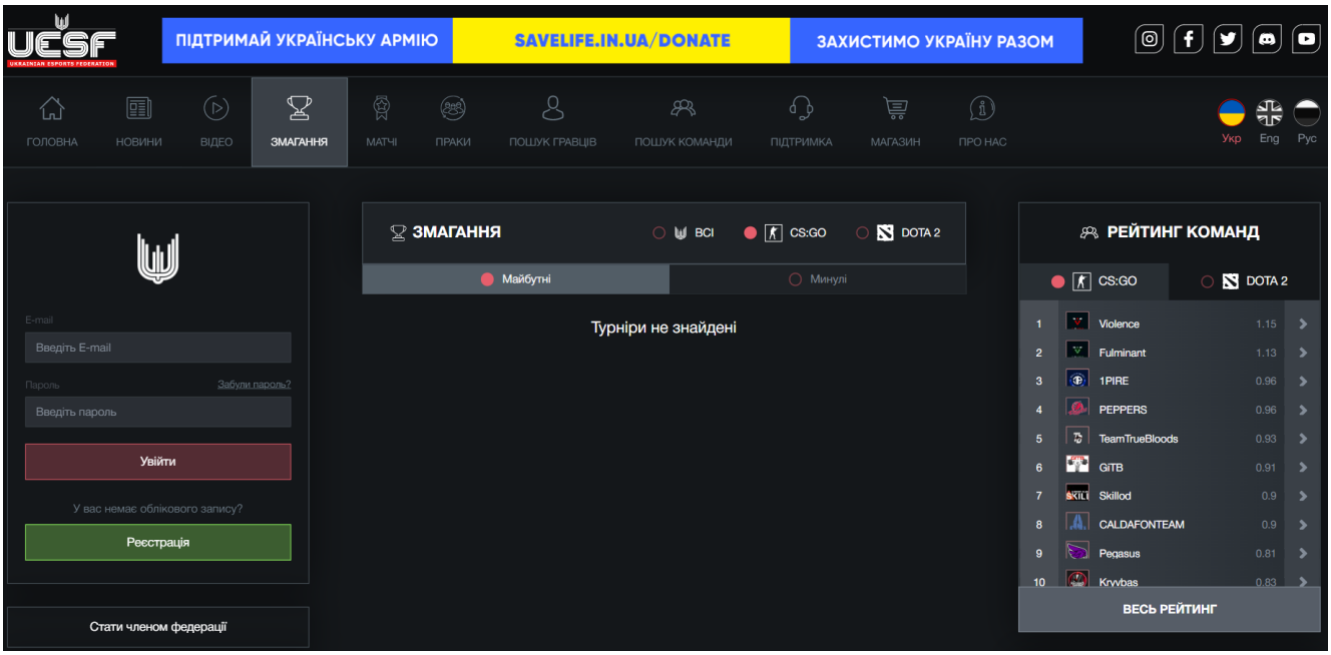

Рисунок 1.20 – «Uesf.org.ua» пошук майбутніх змагань

Як ми бачимо на рис. 1.21, при пошуку відображається детальна інформація про змагання. Така як: призовий фонд, формат, кількість місць та інше.

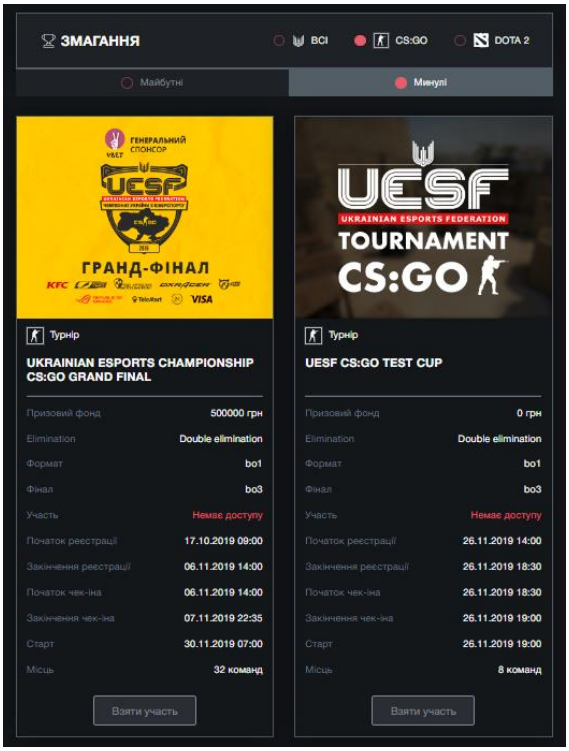

Рисунок 1.21 – «Uesf.org.ua» пошук минулих змагань

Також в додатку «Uesf.org.ua» можна знайти зареєстрованого гравця чи команду (рис.1.22-23). Даний функціонал виконаний з якісним пошуком, а саме: за регіоном, ніком, рейтингом, роллю та іншим.

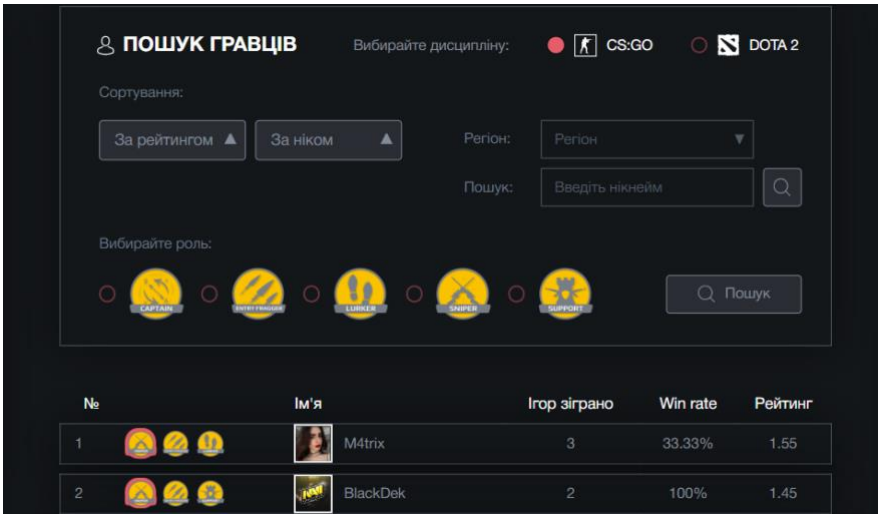

Рисунок 1.22 – «Uesf.org.ua» пошук гравців

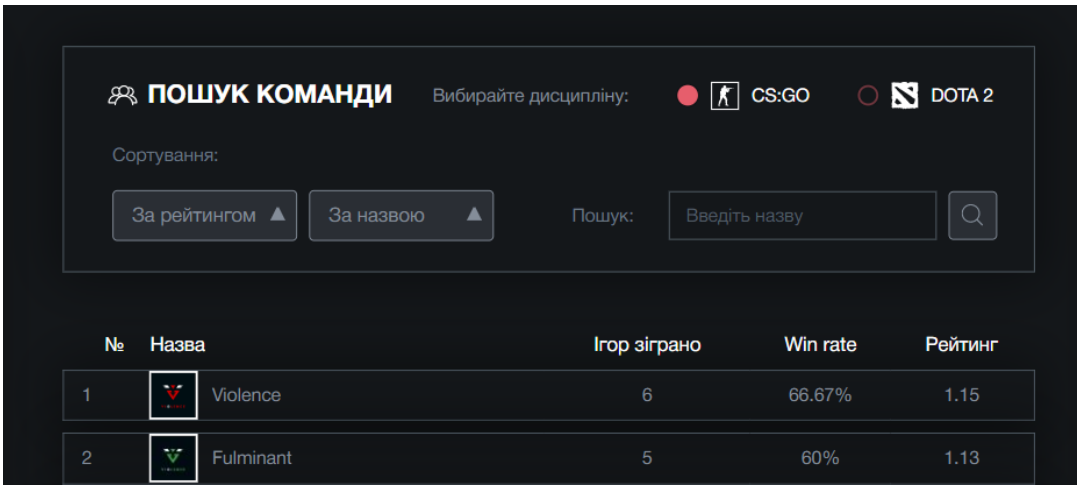

Рисунок 1.23 – «Uesf.org.ua» пошук команд

Для аналізу кіберспортивної тематики, а саме веб-сайтів для організації кіберспортивних турнірів з дисципліни CS GO було обрано: «Gameinside.ua», «Upea.com.ua» та «Uesf.org.ua». Отже, виконаємо порівняння за критеріями, що представлені в таблиці 1.2.

Таблиця 1.2 – Порівняльна характеристика веб-сайтів для організації кіберспортивних турнірів

Критерії	«Gameinside.ua»	«Upea.com.ua»	«Uesf.org.ua»	Інформаційна система для кіберспортивних турнірів з дисципліни CS GO
Багатомовність	–	+	+	+
Можливість зареєструватися	–	–	+	+
Реєстрація на турнір через систему	+	+	+	+
Можливість подати заявку на участь	+	+	+	+
Формування команди в інформаційній системі	–	–	–	+

Продовження таблиці 1.2

Критерії	«Gameinside.ua»	«Upea.com.ua»	«Uesf.org.ua»	Інформаційна система для кіберспортивних турнірів з дисципліни CS GO
Новини	+	+	+	+
Модуль для подання бан-заявки	–	+	–	+
Можливість написати іншому користувачу	–	–	–	+
Перегляд інформацію про минулі турніри	+	+	+	+
Адаптивність під мобільні пристрої	–	+	+	+

Отже, була виконана порівняльна характеристика «Gameinside.ua», «Upea.com.ua» та «Uesf.org.ua» з інформаційною системою для кіберспортивних турнірів з дисципліни CS GO, що буде надалі розроблена.

2 ПОСТАНОВКА ЗАДАЧІ ТА МЕТОДИ ДОСЛІДЖЕННЯ

2.1. Мета та задачі дослідження

Метою інформаційної системи для кіберспортивних турнірів з дисципліни CS GO є спрощення організації та проведення турнірі. Додаткова інформація представлена в Додатку А.

Для досягнення поставленої мети, а саме розробка інформаційної системи для кіберспортивних турнірів з дисципліни CS GO, потрібно реалізувати весь функціонал:

1. Реєстрація в інформаційній системі для кіберспортивних турнірів з дисципліни CS GO;
2. Перегляд новин;
3. Перегляд публікацій, що до проведених турнірів;
4. Перегляд інформації про нові турніри з дисципліни CS GO;
5. Формування команди для участі у турнірові;
6. Подання заявки на участь у змаганні як від команди чи як від особистого лиця;
7. Подання запит іншому користувачеві системи на формування команди;
8. Можливість виходити зі складу команди;
9. Адміністратор команди має можливість видалити гравця з команди;
10. Додання публікацій на сторінку від лиця команди;
11. Додання публікацій на сторінку від особистого лиця;
12. Оформлення оголошення по тренування команди зі сторони адміністратора команди;
13. Можливість написати користувачеві інформаційної системи для кіберспортивних турнірів;
14. Спільний чат команди, що створюється автоматично;
15. Формування турнірів як для команд, так й для поодиноких ігор;
16. Формування результатів проведених змагань;

17. Редагування загальної інформації в інформаційній системі для кіберспортивних турнірів з дисципліни CS GO;
18. Оформлення сповіщення електронною поштою для гравців обраного турніру;
19. Перегляд відправлених сповіщень для гравців обраного турніру;
20. Блокування користувачів інформаційної системи для кіберспортивних турнірів.

Що до цільової аудиторії, це досить великий аспект. Адже до цільових аудиторій з ігрової тематики, а сама дисципліни CS GO можна віднести:

1. Кіберспортсмени;
2. Початківці в комп'ютерній грі;
3. Рекламодавці;
4. Люди, що зацікавлені в обраній дисципліні.

2.2. Розробка візуальної частини інформаційної системи

Звичайно, для розробки візуальної частини інформаційної системи для кіберспортивних турнірів з дисципліни CS GO було обрано HTMSL, CSS та JS. Головний момент на цьому етапі – це обрання фреймворку для розробки.

Існує три основні фреймворки для створення та використання додатків чи інших інформаційних систем, про які чув кожен розробник: React , Vue.js та Angular.

Якщо коротко, то: React — це бібліотека інтерфейсу користувача; Angular — це повноцінний інтерфейсний фреймворк; Vue.js — прогресивний фреймворк.

Їх можна використовувати майже взаємозамінно для створення зовнішніх додатків, але вони не на 100 відсотків однакові. Кожен фреймворк базується на компонентах і дозволяє швидко створювати функції інтерфейсу користувача.

Однак усі вони мають різну структуру та архітектуру — тому спочатку ми розглянемо їхні архітектурні відмінності, щоб зрозуміти філософію, що стоїть за ними.

Розглянемо перш за все саме React. React можна використовувати як бібліотеку інтерфейсу інформаційної системи по проведенню кіберспортивних турнірів та для візуалізації елементів, без застосування конкретної структури проекту, і тому це не суто фреймворк. Елементи React — це найменші будівельні блоки додатків React. Вони потужніші, ніж елементи DOM, тому що React DOM забезпечує їх ефективне оновлення.

React базується на JavaScript, але здебільшого поєднується з JavaScript XML, розширенням синтаксису, яке дозволяє створювати елементи, що містять HTML та навіть JavaScript одночасно.

Наступним прикладом, що можна використовувати для розробки інформаційної системи по проведенню кіберспортивних турнірів є AngularJS. Це оригінальний фреймворк.

Даний фреймворк є орієнтовним Model-View-Controller. Але в Angular 2 немає суворого зв’язку з шаблонами, оскільки він також базується на компонентах.Кожен компонент в Angular містить шаблон, клас, який визначає логіку програми, метадані та декоратори. Метадані для компонента повідомляють Angular, де знайти всі необхідні компоненти.

Останнім прикладом для порівняння є Vue.js. Основна бібліотека Vue.js зосереджена лише на шарі View. Його називають прогресивним фреймворком, тому що ви можете розширити його функціональність за допомогою офіційних і сторонніх пакетів, щоб перетворити його на справжній фреймворк.

Хоча Vue.js не пов’язаний строго з шаблоном Model-View-ViewModel, його дизайн був частково натхненний ним. З Vue.js ви працюватимете здебільшого над шаром ViewModel, щоб переконатися, що дані програми обробляються таким чином, щоб фреймворк відображав оновлене представлення. Головна перевага даного фреймворка – це компоненти у Vue невеликі, самодостатні та можуть повторно використовуватися у всій програмі.

Отже, у результаті аналізу усіх фреймворків було обрано Vue.js. Адже він використовує компоненти, що є легкі в розумінні та в розробці. Vue.js добре також при розширенні коду та проекту. Крім того, даний фреймворк є зрозумілий у

використанні та чудово підійде під невеликий проект по типу інформаційної системи для кіберспортивних турнірів з дисципліни CS GO.

2.3. Розробка функціональної частини інформаційної системи

Попри будь-які інформаційні новини, що PHP вже випередили нові альтернативи, такі як Javascript або Python, дана мова програмування все ще зростає в популярності від року в рік.

Відгалуження PHP є однією з найстаріших мов, яка орієнтована на середовище веб-розробки. PHP вважається дуже доступним для веб-розробки з багатьох причин. З коробки PHP розроблений для бездоганної роботи з найбільш часто використовуваними мовами баз даних, такими як MySQL.

Одна з переваг даної мови полягає в тому, що PHP має величезну спільноту розробників і широкий спектр документації, яка допомогає розробкинам будь-якого рівня почати використовувати його.

Найпопулярніші фреймворки PHP є Laravel та Symfony. Вирішити, який із цих двох буде кращим для інформаційної системи для кіберспортивних турнірів з дисципліни CS GO, може бути важко – обидва вони мають деякі плюси та мінуси, про які розробники повинні враховувати, перш ніж вибрати Laravel або Symfony.

Laravel — це фреймворк із відкритим вихідним кодом, який дотримується шаблону проектування Model-View-Controller. Він повторно використовує чинний компоненти різних фреймворків для створення веб-додатку. Він також містить основні функції фреймворків PHP, таких як Yii, CodeIgniter або Ruby on Rails.

Symfony – цей фреймворк також базується на проектах PHP з відкритим кодом, таких як Propel, Doctrine, PHPUnit, Twig і Swift Mailer. Наперекір тому, що є такі компоненти, як Symfony YAML, Symfony Event Dispatcher, Symfony Dependency Injector і Symfony Templating.

Розглянемо деякі відмінності кожного із фреймворків PHP. Механізм створення шаблонів – Symfony надає Twig, але Laravel надає Blade, що має велику перевагу – ви можете повторно використовувати код. Такі відомі платформи, як Drupal, Magento та

Publish, використовують Symfony, тому це хороша рекомендація для цього фреймворку.

Крім того, Symfony не вимагає серйозних знань SQL, хоча ви повинні створити репозиторій для кожного виклику. Хоча, при роботі із Laravel розробник повинен мати серйозні знання SQL.

У Laravel швидкість програми подібна до інших програм PHP. Він забезпечує належну систему контролю версій, що допомагає пізніше перенести програми. Якщо Symfony правильно реалізовано, швидкість програми покращується. Він регулює швидкість окремої основної функції, тому вся програма може легко визначити, які функції потрібні в цей момент. Це дозволяє інтегрувати програми та найпопулярніші служби електронної пошти через API.

Не дивлячись на це, менше коду в Laravel означає, що розробнику доведеться розробляти коду більше з голови, тоді як у Symfony ви можете просто прочитати код, якого бракує в Laravel. Це потенційно робить Laravel швидшою для роботи, коли користувач працюєте над чимось самостійно.

Отже, для розробки інформаційної системи для кіберспортивних турнірів з дисципліни CS GO було обрано Laravel. Він є зручним у використанні, а також легко інтегрується із бібліотеками вже обраного Vue.js.

3 МОДЕЛЮВАННЯ ТА ПРОЕКТУВАННЯ

3.1 Структура інформаційної системи для кіберспортивних турнірів

Важливим етапом розробки інформаційної системи для кіберспортивних турнірів з дисципліни CS GO є планування саме системи та структури системи. Адже дуже важливо, як користувач буде взаємодіяти з функціоналом.

Для цього було придумано та розроблено повноцінну карту інформаційної системи. Всі головні функції та суміжні з ними було об’єднано у сторінки, для зручного використання (табл.3.1).

Таблиця 3.1 – Структура інформаціної системи для кіберспортивних турнірів

№	Сторінка	Опис
1	Головна	На головні сторінці знаходиться загальна інформація з системи для кіберспортивних турнірів з дисципліни CS GO. Сторінка поділена на три блоки: – остання актуальна новина – активні турніри, – найкращі команди.
2	Новини	На сторінці із новинами представлені останні актуальні новини інформаційної системи для кіберспортивних турнірів.
3	Турніри	На сторінці додаються турніри кіберспортивного напрямку. Всі турніри будуть додані адміністратором.
4	Користувачі	Сторінка із короткою інформацією про користувачів інформаційної системи для кіберспортивних турнірів.
5	Команди	Сторінка із короткою інформацією про створені команди користувачами інформаційної системи для кіберспортивних турнірів.

Продовження таблиці 3.1

№	Сторінка	Опис
6	Власна команда	Користувач інформаційної системи для кіберспортивних турнірів має можливість створити власну команду.
7	Особиста сторінка	Особиста сторінка користувача була розроблена для об’єднання всіх функцій, що відкриті для зареєстрованого користувача.
8	Повідомлення	Користувачі інформаційної системи для кіберспортивних турнірів мають можливість переписуватися повідомленнями.
9	Нотифікації	Користувач отримує повідомлення що до турнірів, на які він зареєструвався, повідомлення від адміністратора системи та оновлень.

Отже, у результаті роботи було сформована інформація, що до структури інформаційної системи та її взаємодії з користувачем інформаційної системи. Інформаційна система для кіберспортивних турнірів з дисципліни CS GO матиме 9 основних сторінок, що об’єднують усі функції у собі.

3.2 Структурно-функціональне моделювання процесу

Розглянемо IDEF0 інформаційної системи для кіберспортивних турнірів з дисципліни CS GO (рис.3.1).

Рисунок 3.1 – IDEF0 системи для кіберспортивних турнірів

Входи моделі: інформація та критерії про турнір, інформація про зареєстрованих на турнір користувачів.

Виходи: проведений турнір, сформований результат турніру.

Наступним кроком буде – декомпозиція на функціональні блоки:

5. Реєстрація користувачів у інформацій системі по проведенню турнірів;
6. Перегляд актуальних турнірів, для реєстрації;
7. Формування команди для участі в актуальних турнірах;
8. Отримання усієї інформації що до турніру від адміністраторів;
9. Проведення турніру;
10. Формування звітності про проведений турнір.

Розглянемо IDEF1 інформаційної системи для кіберспортивних турнірів з дисципліни CS GO (рис.3.2).

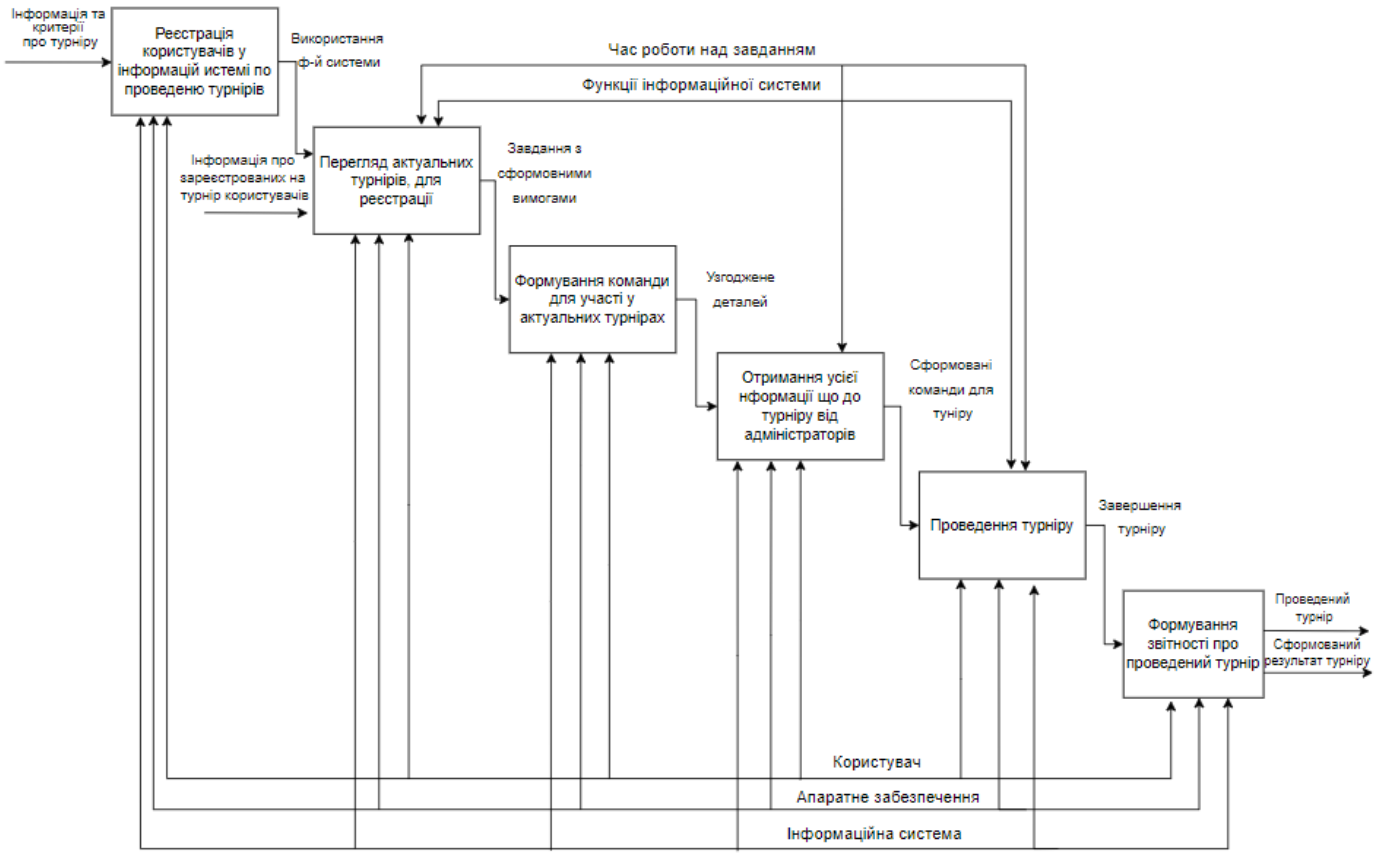

Рисунок 3.2 – IDEF1 системи для кіберспортивних турнірів

3.3 Моделювання діаграми варіантів використання інформаційної системи для кіберспортивних турнірів

Для представлення функціоналу інформаційної системи для кіберспортивних турнірів з дисципліни CS GO було виконано діаграму варіантів використання (табл.3.2-3).

Таблиця 3.2 – Опис акторів

№	Іменування актора	Детальний опис
1	Зареєстрований користувач	Користувач інформаційної системи для кіберспортивних турнірів з дисципліни CS GO, що дає можливість використовувати весь функціонал.

Продовження таблиці 3.2

№	Іменування актора	Детальний опис
2	Не зареєстрований користувач	Користувач інформаційної системи для кіберспортивних турнірів з дисципліни CS GO, що може лише переглядати основну інформацію.
3	Адміністратор	Користувач інформаційної системи для кіберспортивних турнірів з дисципліни CS GO, що має найбільший доступ до системи та має розширений функціонал, що пов’язаний із проведеннями турнірів.

Таблиця 3.3 – Опис варіантів використання

№	Варіанти використання	Детальний опис
1	Авторизація	Зареєстровані користувачі системи для кіберспортивних турнірів можуть авторизуватися в системі.
2	Реєстрація	Не зареєстровані користувачі системи для кіберспортивних турнірів можуть зареєструватися в системі.
3	Написання коментаря	Будь-який користувач системи для кіберспортивних турнірів може написати коментар під статями.
4	Перегляд інформації в інформаційній системі	Будь-який користувач може переглянути інформацію в інформаційній системі.
5	Редагування даних	Зареєстровані користувачі системи для кіберспортивних турнірів можуть редагувати особисту інформацію.

Продовження таблиці 3.3

№	Варіанти використання	Детальний опис
6	Участь у кіберспортивному турнірі	Зареєстровані користувачі системи кіберспортивних турнірів можуть брати участь у змаганнях.
7	Реєстрація на турнір	Зареєстровані користувачі можуть реєструватися у кіберспортивному турнірі.
8	Проведення тренувань команди	Адміністратори команд можуть тренувати команди, для цього назначається час та день тренування.
9	Формування команди для участі у турнірі	Зареєстровані користувачі можуть об’єднуватися у команди для участі у змаганнях.
10	Формування інформації про змагання	Адміністратори системи для кіберспортивних турнірів можуть формувати пости про змагання та публікувати їх.
11	Редагування даних в інформацій системі по кіберзмаганнях	Адміністратори системи для кіберспортивних турнірів можуть редагувати переважну частину даних.
12	Формування результатів проведеного турніру	Адміністратори системи для кіберспортивних турнірів можуть формувати результати проведених турнірів.
13	Надсилання розсилки командам турніру	Адміністратори системи для кіберспортивних турнірів можуть виконувати розсилку для команд, що беруть участь у певному турнірі.
14	Блокування користувачів	Адміністратори системи для кіберспортивних турнірів можуть блокувати користувачів системи.

Сформувавши представлення інформаційної системи для кіберспортивних турнірів з дисципліни CS GO було виконано діаграму варіантів використання на рисунку 3.3-4.

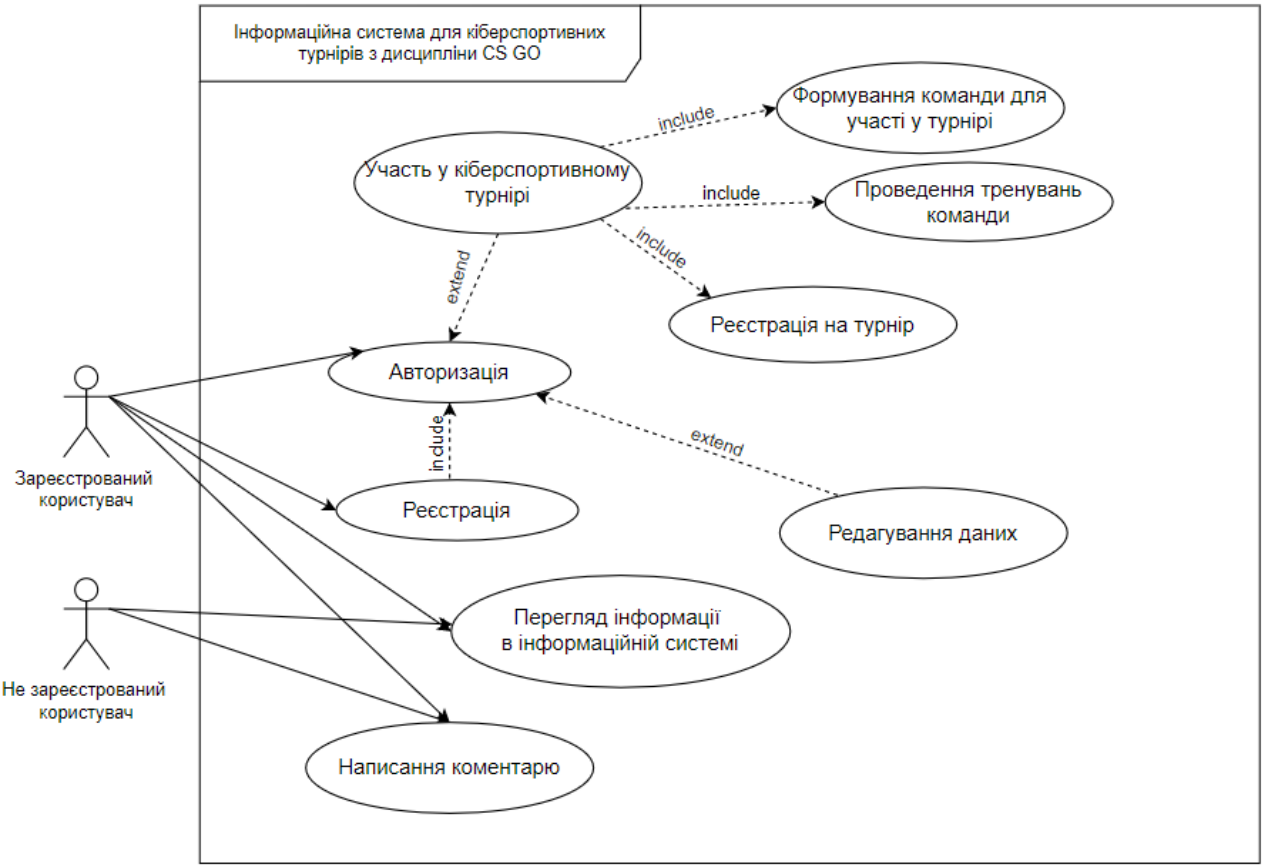

Рисунок 3.3 – Діаграма варіантів використання для користувачів системи

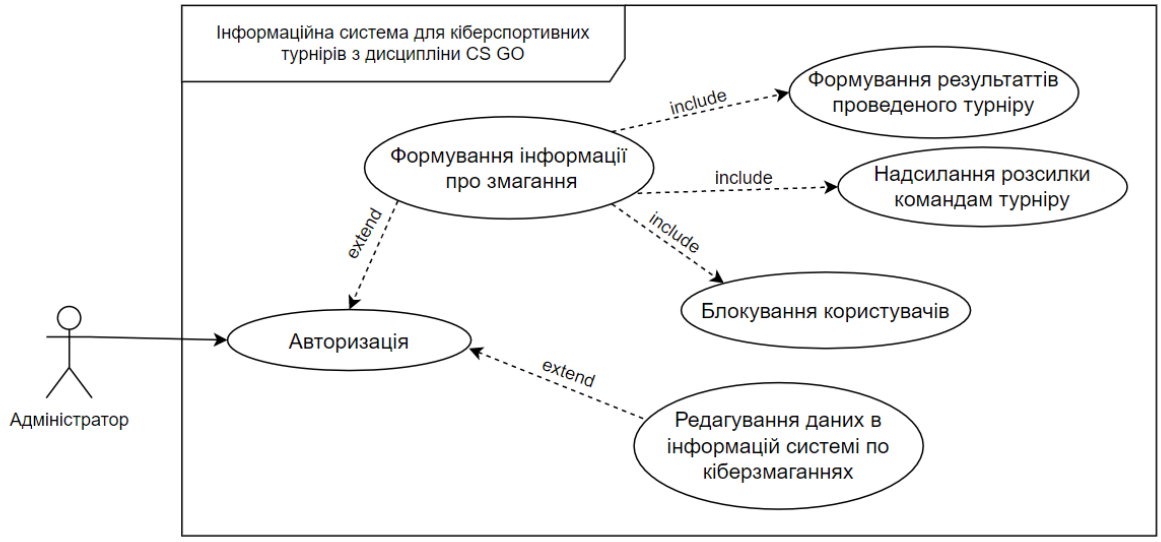

Рисунок 3.4 – Діаграма варіантів використання для адміністратора

Отже, результатом роботи є розроблені діаграми використання для звичайного користувача інформаційної системи для кіберспортивних турнірів з дисципліни CS GO та адміністратора.

3.4. Проектування бази даних

Проектування бази даних повинна виконувати основні поставлені задачі та враховувати всі вимоги до функціоналу інформаційної системи для кіберспортивних турнірів з дисципліни CS GO та адміністратора [18-20]. Для їхнього представлення було виконано розробку ER-діаграми на рисунку 3.5.

Детальний розбір кожного із елементів діаграми представлений в таблиці. 3.4.

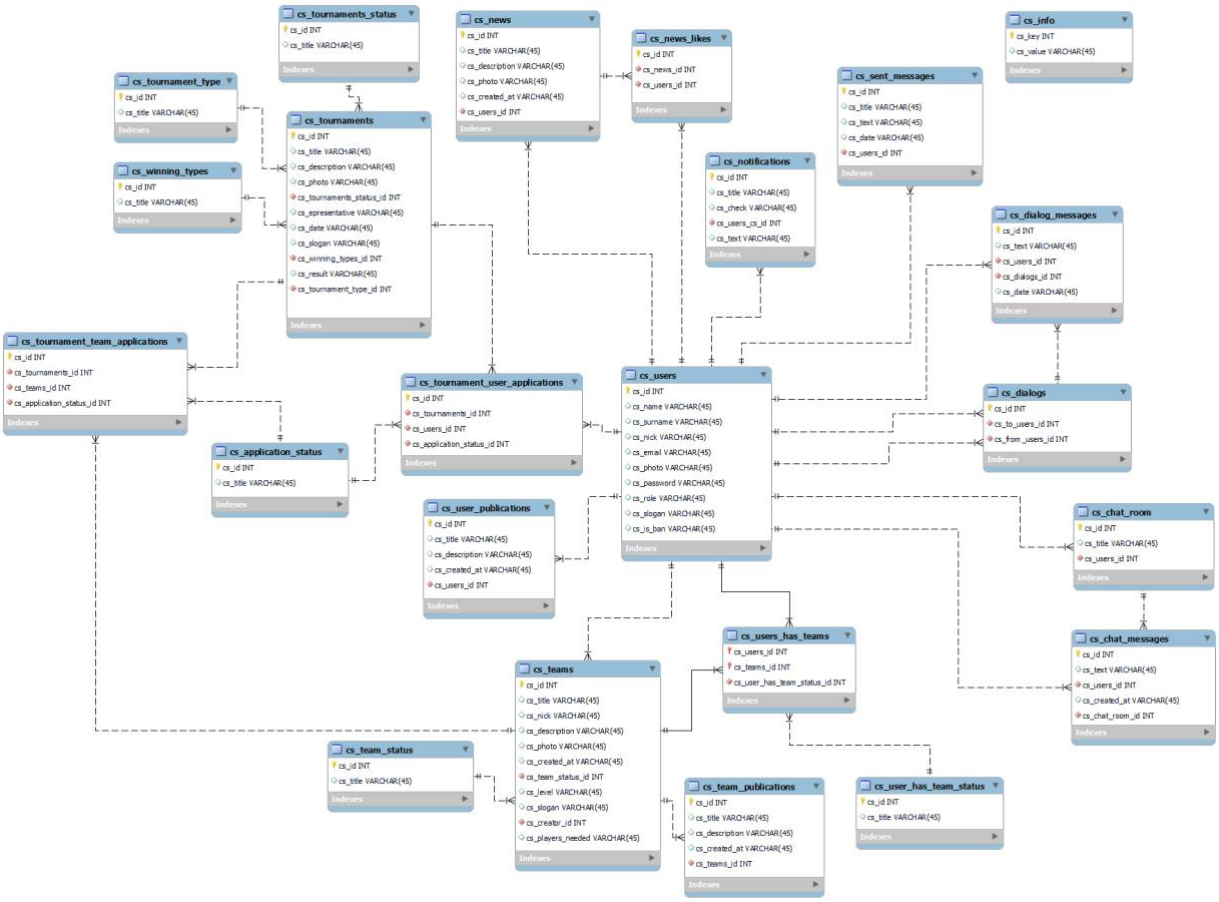

Рисунок 3.5 – ER-діаграма інформаційної системи для кіберспортивних турнірів з дисципліни CS GO

Таблиця 3.4 – Опис ER-діаграми інформаційної системи для кіберспортивних турнірів з дисципліни CS GO

№	Таблиці	Опис	Колонка	Тип	Ключ	Обмеження
1	cs_user	Таблиця, що зберігає в собі інформацію про всіх гравців, що зареєструвалися у інформаційній системі	cs_id	INT	PK	Не порожній
			cs_name	VARCHAR(45)		Не порожній
			cs_surname	VARCHAR(45)		Не порожній
			cs_nick	VARCHAR(45)		Не порожній
			cs_email	VARCHAR(45)		Не порожній
			cs_photo	VARCHAR(45)		Не порожній
			cs_password	VARCHAR(45)		Не порожній
			cs_role	VARCHAR(45)		Не порожній
			cs_slogane	VARCHAR(45)		Не порожній
			cs_is_ban	VARCHAR(45)		Не порожній
2	cs_tournament_type	Інформація про типи турнірів	cs_id	INT	PK	Не порожній
			cs_title	VARCHAR(45)		Не порожній
3	cs_winning_type	Інформація про тип вийграшу й створеному турнірі	cs_id	INT	PK	Не порожній
			cs_title	VARCHAR(45)		Не порожній
4	cs_tournament_status	Інформація про статус турніру	cs_id	INT	PK	Не порожній
			cs_title	VARCHAR(45)		Не порожній

Продовження таблиці 3.4

№	Таблиці	Опис	Колонка	Тип	Ключ	Обмеження
5	cs_tournament	Детальна інформація про створений адміністратором кіберспортивний турнір	cs_id	INT	PK	Не порожній
			cs_title	VARCHAR(45)		Не порожній
			cs_description	VARCHAR(45)		Не порожній
			cs_photo	VARCHAR(45)		Не порожній
			cs_tournament_status_id	INT	FK	Не порожній
			cs_winning_type_id	INT	FK	Не порожній
			cs_tournament_type_id	INT	FK	Не порожній
			cs_epresentative	VARCHAR(45)		Не порожній
			cs_date	VARCHAR(45)		Не порожній
			cs_slogan	VARCHAR(45)		Не порожній
			cs_result	VARCHAR(45)		Не порожній
6	cs_notification	Інформація про нотифікацію	cs_id	INT	PK	Не порожній
			cs_title	VARCHAR(45)		Не порожній
			cs_check	VARCHAR(45)		Не порожній
			cs_user_cs	INT	FK	Не порожній
			cs_text	VARCHAR(45)		Не порожній

Продовження таблиці 3.4

№	Таблиці	Опис	Колонка	Тип	Ключ	Обмеження
7	cs_news	Інформація про сторені новини інформаційної системи	cs_id	INT	PK	Не порожній
			cs_title	VARCHAR(45)		Не порожній
			cs_description	VARCHAR(45)		Не порожній
			cs_photo	VARCHAR(45)		Не порожній
			cs_created_at	VARCHAR(45)		Не порожній
			cs_user_id	INT	FK	Не порожній
8	cs_news_likes	Інформація про вподобання тієї чи іншої новини	cs_id	INT	PK	Не порожній
			cs_news_id	INT	FK	Не порожній
			cs_user_id	INT	FK	Не порожній
9	cs_user_publications	Інформація про публікації користувачів	cs_id	INT	PK	Не порожній
			cs_title	VARCHAR(45)		Не порожній
			cs_description	VARCHAR(45)		
			cs_created_at	DATE		
			cs_user_id	INT	FK	Не порожній
10	cs_application_status	Інформація про статус заявки	cs_id	INT	PK	Не порожній
			cs_title	VARCHAR(45)		Не порожній

Продовження таблиці 3.4

№	Таблиці	Опис	Колонка	Тип	Ключ	Обмеження
11	cs_tournament_ user_application	Детальна інформація про заявку на кіберспортивний турнір від користувача	cs_id	INT	PK	
			cs_tournaments_id	INT	FK	Не порожній
			cs_users_id	INT	FK	Не порожній
			cs_application_status_id	INT	FK	Не порожній
12	cs_tournament_ team_applications	Детальна інформація про заявку на кіберспортивний турнір від команди	cs_id	INT	PK	
			cs_tournaments	INT	FK	Не порожній
			cs_teams_id	INT	FK	Не порожній
			cs_application_status_id	INT	FK	Не порожній
13	cs_team_status	Інформація про статус команди	cs_id	INT	PK	Не порожній
			cs_title	VARCHAR(45)		Не порожній
14	cs_user_has_ team_status	Інформація про взаємозвязок самого користувача та статусу	cs_id	INT	PK	Не порожній
			cs_title	VARCHAR(45)		Не порожній
15	cs_teams	Детальна інформація про команду	cs_id	INT	PK	Не порожній
			cs_title	VARCHAR(45)		Не порожній
			cs_nick	VARCHAR(45)		Не порожній

Продовження таблиці 3.4

№	Таблиці	Опис	Колонка	Тип	Ключ	Обмеження
			cs_description	VARCHAR(45)		Не порожній
			cs_photo	VARCHAR(45)		Не порожній
			cs_created_at	DATE		Не порожній
			cs_team_status_id	INT	FK	Не порожній
			cs_level	VARCHAR(45)		Не порожній
			cs_slogan	VARCHAR(245)		Не порожній
			cs_creator_id	INT	FK	Не порожній
			cs_players_needed	BOOLEAN		Не порожній
16	cs_users_has_teams	Інформація про взаємозвязок самого користувача та його команди	cs_users_id	INT	PK	Не порожній
			cs_teams_id	INT	FK	Не порожній
			cs_user_has_team_status	INT	FK	Не порожній
17	cs_team_publication	Інформація про публікації команди	cs_id	INT	PK	Не порожній
			cs_title	VARCHAR(245)		Не порожній
			cs_description	TEXT		Не порожній
			cs_created_at	VARCHAR(245)		Не порожній
			cs_teams_id	INT	FK	Не порожній

Продовження таблиці 3.4

№	Таблиці	Опис	Колонка	Тип	Ключ	Обмеження
18	cs_sent_messages	Повідомлення користувача інформаційної системи	cs_id	INT	PK	Не порожній
			cs_title	VARCHAR(245)		Не порожній
			cs_text	TEXT		Не порожній
			cs_date	DATE		Не порожній
			cs_users_id	INT	FK	Не порожній
19	cs_info	Загальна інформація інформаційної системи	cs_key	INT	PK	Не порожній
			cs_value	VARCHAR(45)		Не порожній
20	cs_dialog_message	Інформація про діалог та повідомлення	cs_id	INT	PK	Не порожній
			cs_text	VARCHAR(45)		Не порожній
			cs_users_id	INT	FK	Не порожній
			cs_dialogs_id	INT	FK	Не порожній
			cs_date	DATE		Не порожній
21	cs_dialogs	Інформація про діалоги	cs_id	INT	PK	Не порожній
			cs_to_users_id	INT	FK	Не порожній
			cs_from_users_id	INT	FK	Не порожній

Продовження таблиці 3.4

№	Таблиці	Опис	Колонка	Тип	Ключ	Обмеження
22	cs_chat_room	Інформація про чат-кімнати	cs_id	INT	PK	Не порожній
			cs_title	VARCHAR(45)		Не порожній
			cs_users_id	INT	FK	Не порожній
23	cs_chat_messages	Інформація про повідомлення чатів	cs_id	INT	PK	Не порожній
			cs_text	TEXT		Не порожній
			cs_users_id	INT	FK	Не порожній
			cs_ceated_a	DATE		Не порожній
			cs_chat_room_id	INT	FK	Не порожній

4 ПРАКТИЧНА РЕАЛІЗАЦІЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

4.1 Програмна реалізація

Програмна реалізація інформаційної системи для кіберспортивних турнірів з дисципліни CS GO реалізовувалася із використанням Laravel та Vue.js [21-22]. На рисунку 4.1 представлено пакети та модулі, що були створені в ході роботи.

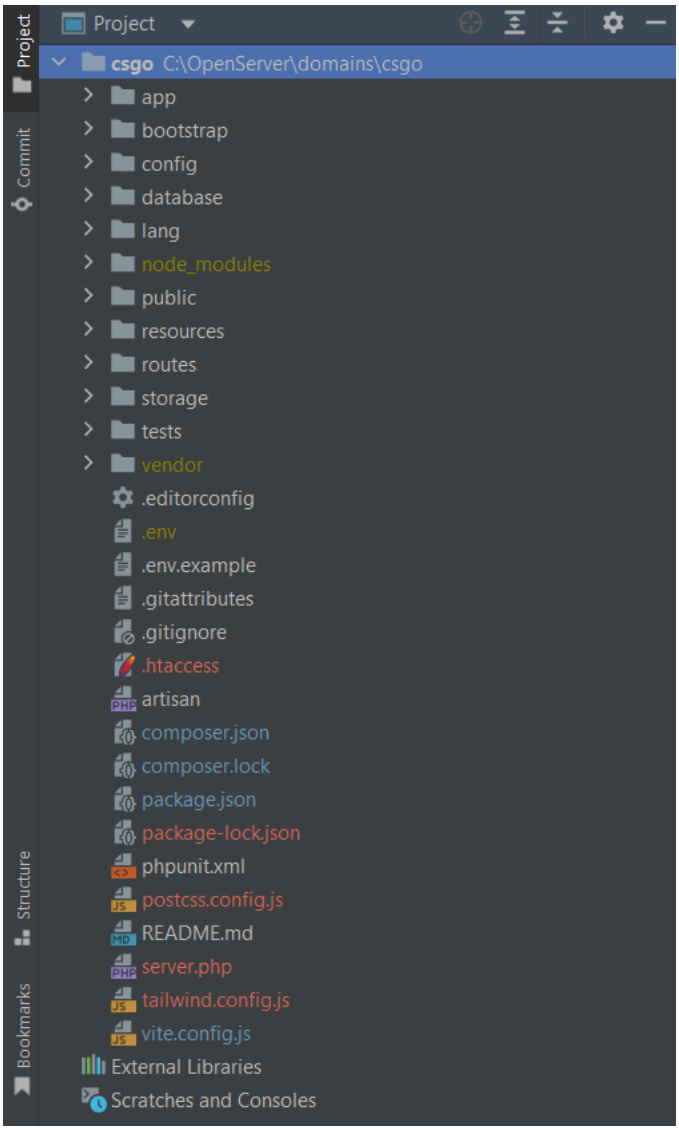

Рисунок 4.1 – Пакети та модулі інформаційної системи для кіберспортивних турнірів з дисципліни CS GO

Також було розроблено базу даних інформаційної системи для кіберспортивних турнірів з дисципліни CS GO. Розглянемо створені таблиці на рис.4.2-24.

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

Рисунок 4.2 – «application_status» таблиця

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно
1	id	bigint(20)		UNSIGNED	Нет	Нем		AUTO_INCREMENT
2	cs_users_id	bigint(20)		UNSIGNED	Нет	Нем		
3	cs_chat_rooms_id	bigint(20)		UNSIGNED	Нет	Нем		
4	cs_text	text	utf8mb4_unicode_ci		Нет	Нем		

Рисунок 4.3 – «chat_messages» таблиця

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

Рисунок 4.4 – «chat_rooms

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно
1	id	bigint(20)		UNSIGNED	Нет	Нем		AUTO_INCREMENT
2	cs_to_users_id	bigint(20)		UNSIGNED	Нет	Нем		
3	cs_from_users_id	bigint(20)		UNSIGNED	Нет	Нем		

Рисунок 4.5 – «dialogs» таблиця

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно
1	id	bigint(20)		UNSIGNED	Нет	Нем		AUTO_INCREMENT
2	cs_users_id	bigint(20)		UNSIGNED	Нет	Нем		
3	cs_dialogs_id	bigint(20)		UNSIGNED	Нет	Нем		
4	cs_text	text	utf8mb4_unicode_ci		Нет	Нем		

Рисунок 4.6 – «dialogs_messages

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно
1	cs_key	varchar(30)	utf8mb4_unicode_ci		Нет	*Нет*		
2	cs_value	varchar(255)	utf8mb4_unicode_ci		Нет	*Нет*		

Рисунок 4.7 – «info» таблиця

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно
1	id	bigint(20)		UNSIGNED	Нет	*Нет*		AUTO_INCREMENT
2	cs_title	varchar(255)	utf8mb4_unicode_ci		Нет	*Нет*		
3	cs_description	longtext	utf8mb4_unicode_ci		Нет	*Нет*		
4	cs_photo	varchar(255)	utf8mb4_unicode_ci		Нет	/img/no-photo-available.png		
5	cs_users_id	bigint(20)		UNSIGNED	Нет	*Нет*		

Рисунок 4.8 – «news» таблиця

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно
1	id	bigint(20)		UNSIGNED	Нет	*Нет*		AUTO_INCREMENT
2	cs_users_id	bigint(20)		UNSIGNED	Нет	*Нет*		
3	cs_news_id	bigint(20)		UNSIGNED	Нет	*Нет*		

Рисунок 4.9 – «news_likes» таблиця

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно
1	id	bigint(20)		UNSIGNED	Нет	*Нет*		AUTO_INCREMENT
2	cs_title	varchar(255)	utf8mb4_unicode_ci		Нет	*Нет*		
3	cs_text	text	utf8mb4_unicode_ci		Нет	*Нет*		
4	cs_check	tinyint(1)			Нет	0		
5	cs_users_id	bigint(20)		UNSIGNED	Нет	*Нет*		

Рисунок 4.10 – «notifications» таблиця

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно
1	id	bigint(20)		UNSIGNED	Нет	*Нет*		AUTO_INCREMENT
2	cs_title	varchar(255)	utf8mb4_unicode_ci		Нет	*Нет*		
3	cs_description	longtext	utf8mb4_unicode_ci		Нет	*Нет*		
4	cs_users_id	bigint(20)		UNSIGNED	Нет	*Нет*		

Рисунок 4.11 – «sent_messages» таблиця

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно
1	id	bigint(20)		UNSIGNED	Нет	*Нет*		AUTO_INCREMENT
2	cs_title	varchar(255)	utf8mb4_unicode_ci		Нет	*Нет*		
3	cs_description	longtext	utf8mb4_unicode_ci		Нет	*Нет*		
4	cs_teams_id	bigint(20)		UNSIGNED	Нет	*Нет*		

Рисунок 4.12 – «team_publications» таблиця

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

Рисунок 4.13 – «team_status» таблиця

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно
1	id	bigint(20)		UNSIGNED	Нет	*Нет*		AUTO_INCREMENT
2	cs_title	varchar(255)	utf8mb4_unicode_ci		Нет	*Нет*		
3	cs_nick	varchar(255)	utf8mb4_unicode_ci		Нет	*Нет*		
4	cs_description	longtext	utf8mb4_unicode_ci		Нет	*Нет*		
5	cs_photo	varchar(255)	utf8mb4_unicode_ci		Нет	/img/no-photo-available.png		
6	cs_team_status_id	bigint(20)		UNSIGNED	Да	*NULL*		
7	cs_level	varchar(255)	utf8mb4_unicode_ci		Нет	1		
8	cs_slogan	text	utf8mb4_unicode_ci		Да	*NULL*		
9	cs_creator_id	bigint(20)		UNSIGNED	Нет	*Нет*		
10	cs_players_needed	tinyint(1)			Нет	0		

Рисунок 4.14 – «teams» таблиця

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно
1	id	bigint(20)		UNSIGNED	Нет	*Нет*		AUTO_INCREMENT
2	cs_title	varchar(255)	utf8mb4_unicode_ci		Нет	*Нет*		
3	cs_description	longtext	utf8mb4_unicode_ci		Нет	*Нет*		
4	cs_photo	varchar(255)	utf8mb4_unicode_ci		Нет	/img/no-photo-available.png		
5	cs_tournament_status_id	bigint(20)		UNSIGNED	Нет	*Нет*		
6	cs_epresentative	varchar(255)	utf8mb4_unicode_ci		Да	*NULL*		
7	cs_slogan	text	utf8mb4_unicode_ci		Да	*NULL*		
8	cs_winning_types_id	bigint(20)		UNSIGNED	Нет	*Нет*		
9	cs_winning	text	utf8mb4_unicode_ci		Да	*NULL*		
10	cs_result	longtext	utf8mb4_unicode_ci		Да	*NULL*		
11	cs_tournament_types_id	bigint(20)		UNSIGNED	Нет	*Нет*		

Рисунок 4.15 – «tournaments» таблиця

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

Рисунок 4.16 – «tournaments_status» таблиця

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно
1	id	bigint(20)		UNSIGNED	Нет	*Нет*		AUTO_INCREMENT
2	cs_tournaments_id	bigint(20)		UNSIGNED	Нет	*Нет*		
3	cs_teams_id	bigint(20)		UNSIGNED	Нет	*Нет*		
4	cs_application_status_id	bigint(20)		UNSIGNED	Нет	*Нет*		

Рисунок 4.17 – «tournaments_team_application» таблиця

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

Рисунок 4.18 – «tournaments_types» таблиця

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно
1	id	bigint(20)		UNSIGNED	Нет	*Нет*		AUTO_INCREMENT
2	cs_tournaments_id	bigint(20)		UNSIGNED	Нет	*Нет*		
3	cs_users_id	bigint(20)		UNSIGNED	Нет	*Нет*		
4	cs_application_status_id	bigint(20)		UNSIGNED	Нет	*Нет*		

Рисунок 4.19 – «tournaments_user_application» таблиця

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно
1	id	bigint(20)		UNSIGNED	Нет	*Нет*		AUTO_INCREMENT
2	cs_users_id	bigint(20)		UNSIGNED	Нет	*Нет*		
3	cs_teams_id	bigint(20)		UNSIGNED	Нет	*Нет*		
4	cs_users_has_teams_status_id	bigint(20)		UNSIGNED	Нет	*Нет*		

Рисунок 4.20 – «users_has_teams

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

Рисунок 4.21 – «users_has_teams_status

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно
1	id	bigint(20)		UNSIGNED	Нет	*Нет*		AUTO_INCREMENT
2	cs_name	varchar(255)	utf8mb4_unicode_ci		Нет	*Нет*		
3	cs_surname	varchar(255)	utf8mb4_unicode_ci		Нет	*Нет*		
4	cs_nick	varchar(255)	utf8mb4_unicode_ci		Нет	*Нет*		
5	cs_email	varchar(255)	utf8mb4_unicode_ci		Нет	*Нет*		
6	cs_photo	varchar(255)	utf8mb4_unicode_ci		Нет	/img/no-photo-available.png		
7	cs_email_verified_at	timestamp			Да	*NULL*		
8	cs_password	varchar(255)	utf8mb4_unicode_ci		Нет	*Нет*		
9	cs_role	varchar(255)	utf8mb4_unicode_ci		Нет	user		
10	cs_slogan	text	utf8mb4_unicode_ci		Да	*NULL*		
11	cs_is_ban	tinyint(1)			Нет	0		
12	cs_need_team	tinyint(1)			Нет	0		
13	cs_level	int(11)			Нет	1		

Рисунок 4.22 – «Users» таблиця

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно
1	id	bigint(20)		UNSIGNED	Нет	*Нет*		AUTO_INCREMENT
2	cs_title	varchar(255)	utf8mb4_unicode_ci		Нет	*Нет*		
3	cs_description	longtext	utf8mb4_unicode_ci		Нет	*Нет*		
4	cs_user_id	bigint(20)		UNSIGNED	Нет	*Нет*		

Рисунок 4.23 – «users_publications» таблиця

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

Рисунок 4.24 – «winning_types» таблиця

Отже, у результаті роботи було розроблено структурно-вірно базу даних, що містить себе двадцять три таблиці. Подалі ці таблиці будуть використовуватися для збереження даних про користувачів, турніри та інше. Додаткова інформація представлена в Додатку Б.

4.2 Використання інформаційної системи звичайним користувачем

Головною сторінкою представляє з себе інформативну сторінку, що лише перенаправляє користувача на сторінки для авторизації чи реєстрації. Для цього, окрім головної кнопки є додаткові, що знаходяться у хедері інформаційної системи.

Зображення для цієї сторінки були обрані відповідно до тематики кваліфікаційної роботи, а саме приклади гравців з гри CS GO. Дана сторінка є головною для всіх неавторизованих користувачів (рис.4.25).

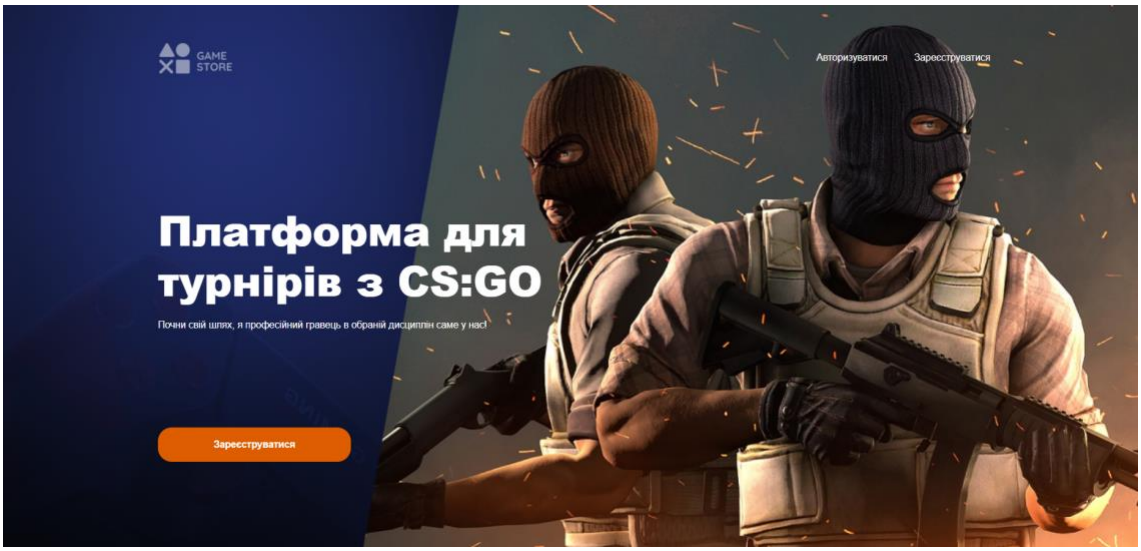

Рисунок 4.25 – Головна сторінка незареєстрованого користувача

При натисканні на кнопку користувач перенаправляється на сторінку для реєстрації. Незареєстрований користувач має ввести особисті данні, такі як прізвище та ім’я, електронну пошту, нік ідентичний з гри (рис.4.26).

Також користувач повинен ввести пароль та продублювати його, для перевірки правильності введення. Також, якщо користувач вже зареєстрований, то він може перейти на сторінку для авторизації для участі у кіберспортивних турнірах.

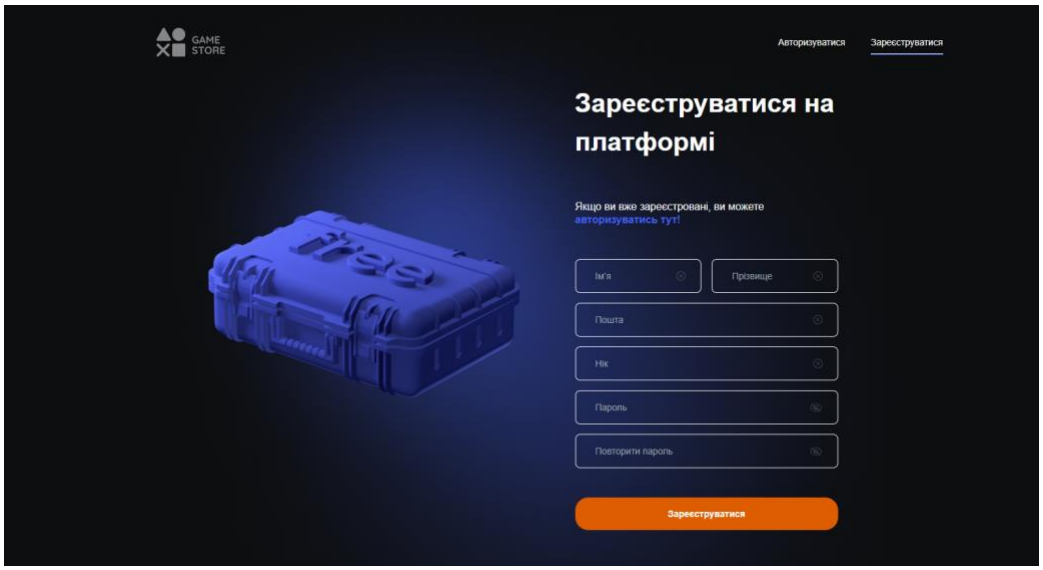

Рисунок 4.26 – Сторінка реєстрації

Для авторизації використовується електронна пошта користувача та вказаний вже раніше, при реєстрації, пароль (рис.4.27). користувач також має можливість використовувати додаткові посилання для переходу на головну сторінку та сторінку для реєстрації.

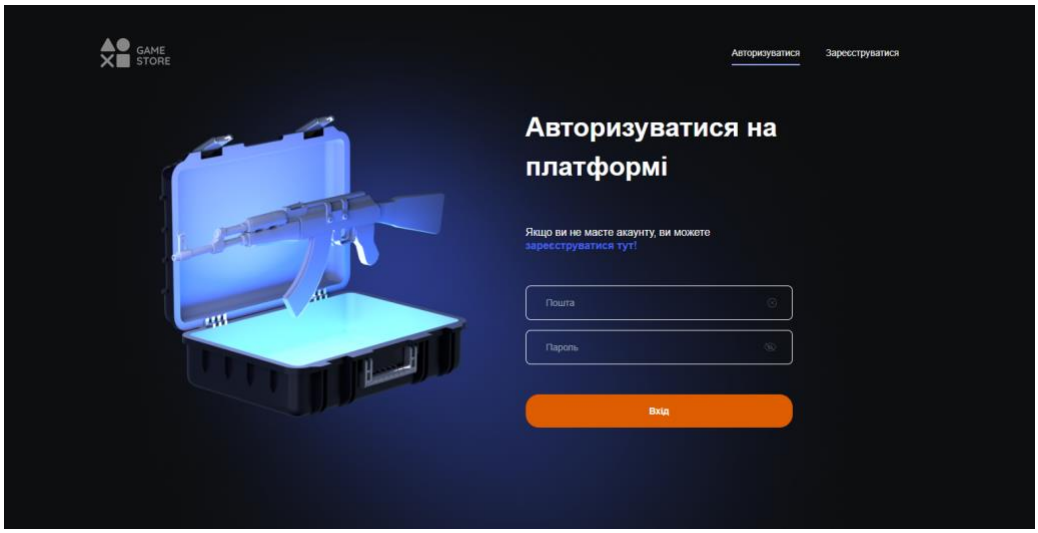

Рисунок 4.27 – Сторінка авторизації

Після авторизації в інформаційній системі для проведення кіберспортивних турнірів системи користувач перенаправляється на головну сторінку зареєстрованого та, водночас, авторизованого користувача (рис.4.28).

Дана сторінка поділяється на три основні блоки, а саме остання новина, найкращі команди та актуальні турніри.

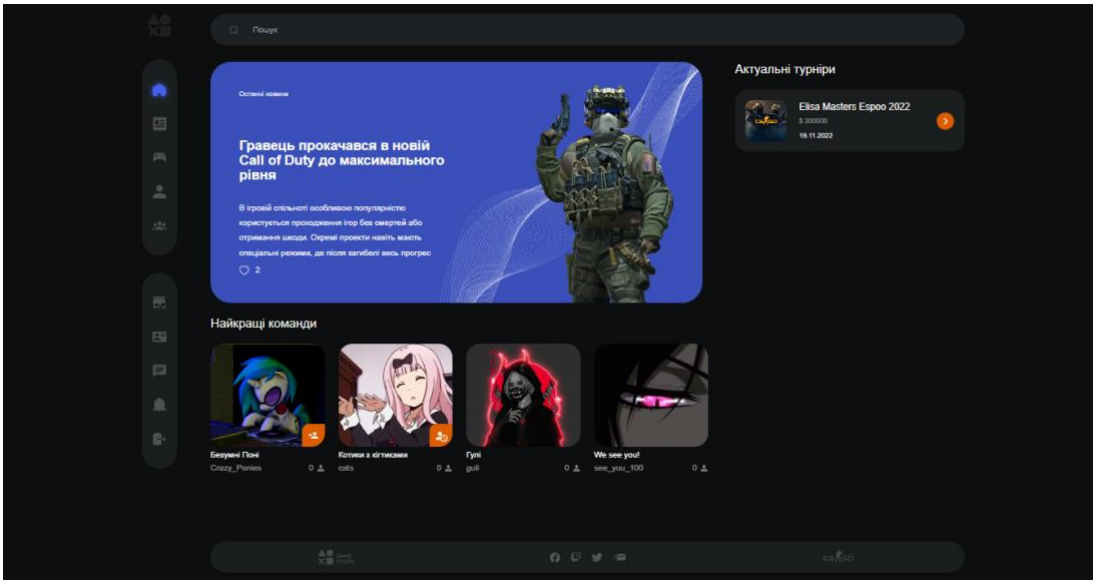

Рисунок 4.28 – Головна сторінка зареєстрованого та авторизованого користувача

Після авторизації користувач має можливість редагувати власні дані. Для цього користувач повинен натиснути на відповідну кнопку в інформаційній системі (рис.4.29).

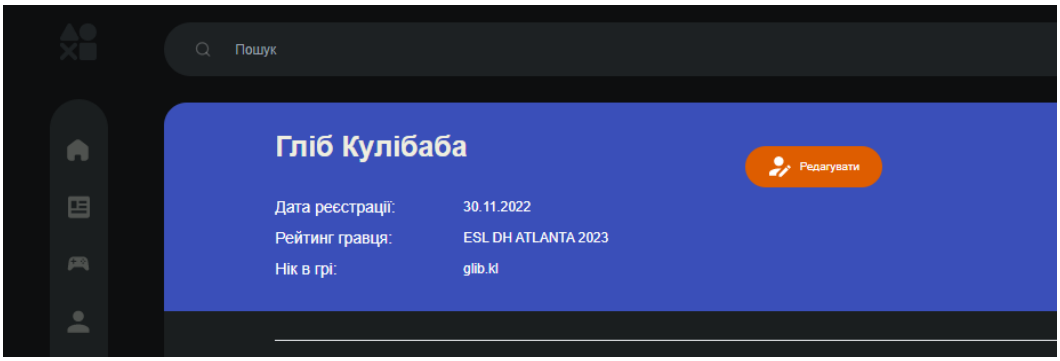

Рисунок 4.29 – Редагування особистої сторінки

Зареєстрований користувач інформаційної системи для проведення кіберспортивних турнірів має можливість змінити ім’я, прізвище, електронну адресу, нік, рейтинг, чи шукає користувач команду, мотивуюче гасло (рис.4.30).

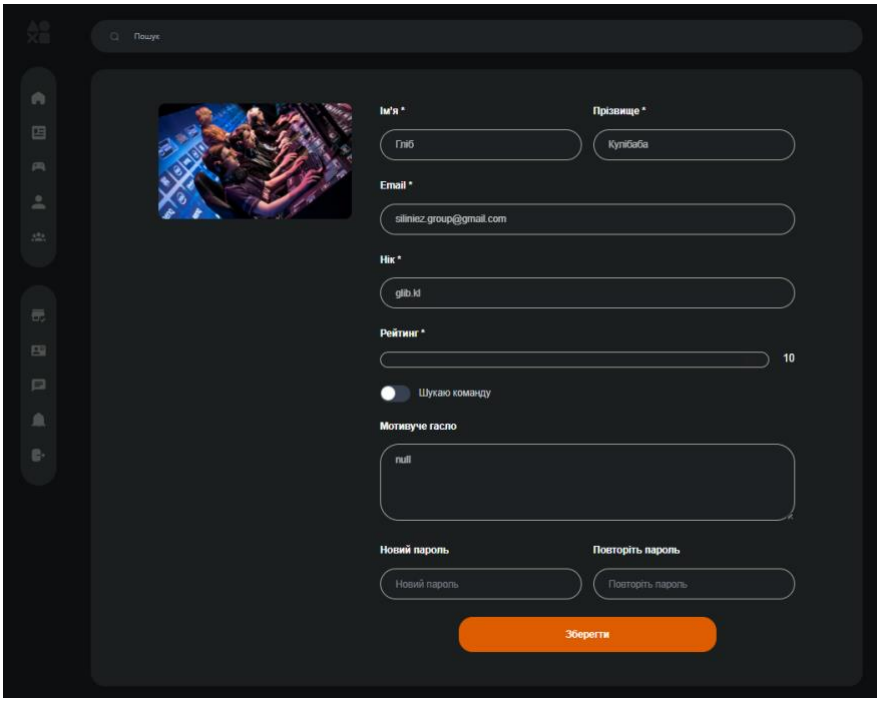

Рисунок 4.30 –

Після збереження інформації користувач отримує повідомлення про вдале оновлення інформації. Прикладом вдалого оновлення інформації є приклад на рисунку 4.31.

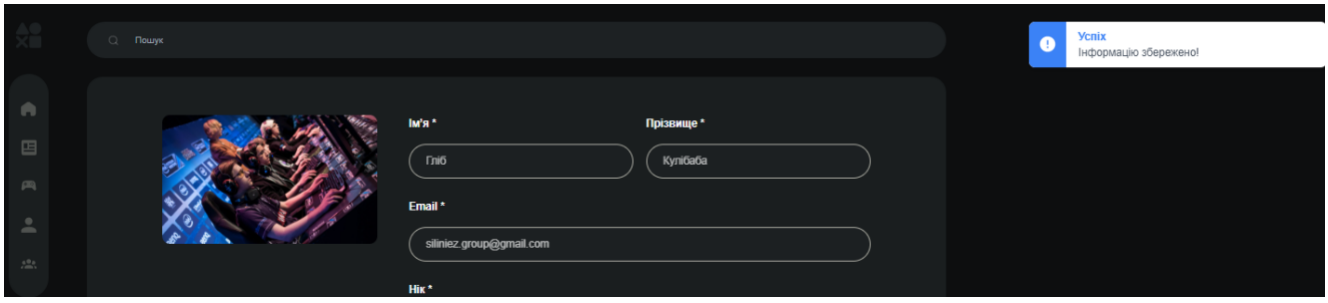

Рисунок 4.31 – Вдале оновлення даних

Після головної сторінки користувач може переглядати новини, що вже були додані до інформаційної системи. На сторінці відображається також заголовок, коротка інформація, кількість вподобань кожної з новини (рис.4.32).

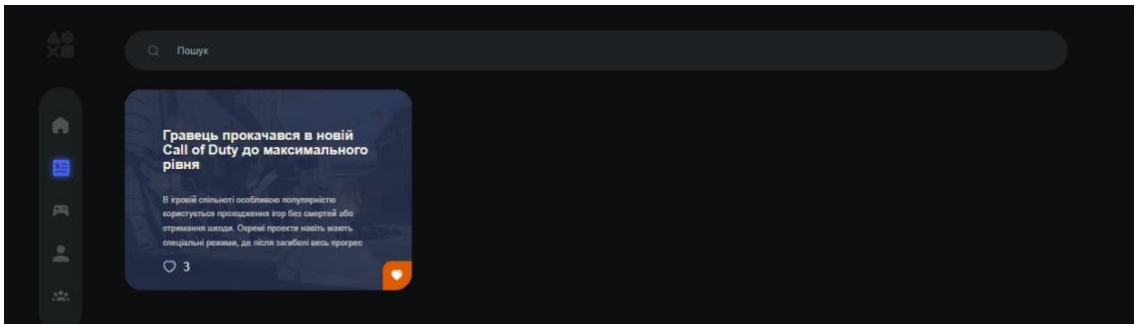

Рисунок 4.32 – Сторінка із кіберспортивними новинами

При натисканні на кнопку «Вподобати новину», що знаходиться у куті блоку, кнопка змінює свій стан. Приклади відображення до та після натискання представлено на рисунку 4.33-34.

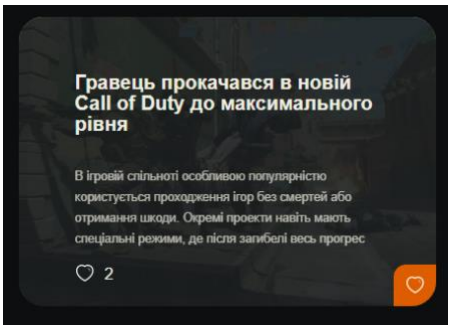

Рисунок 4.33 – Стан до натискання кнопки вподобання

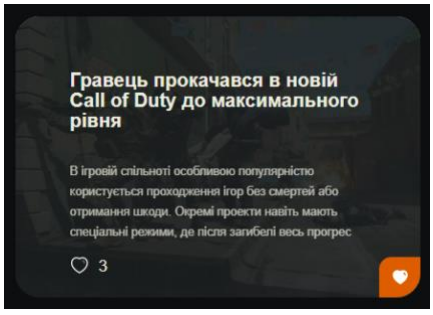

Рисунок 4.34 – Стан після натискання кнопки вподобання

При наведенні також була додана додаткова анімація зміну кольору, щоб користувачеві було легше орієнтуватися, що саме він обирає (рис.4.35). Кольори були обрані відповідно до створеного дизайну.

Рисунок 4.35 – Зміна кольору при наведенні

На рисунку 4.36 представлена новина у повному розмірі. Також, на рисунку 4.37 представлено приклад відображення вподобань обраної кіберспортивної новини.

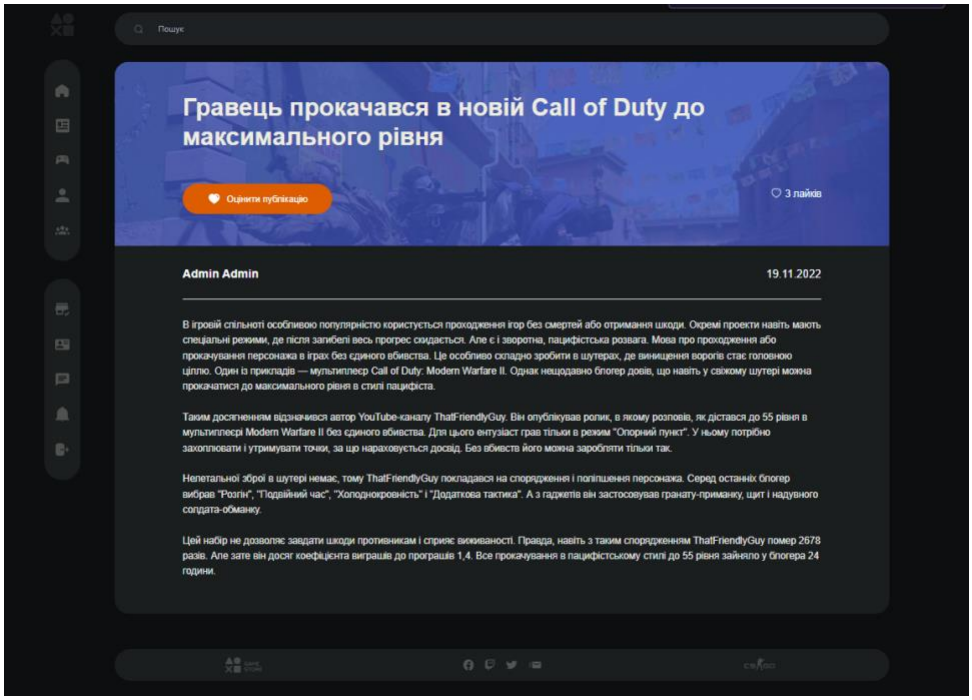

Рисунок 4.36 – Приклад відображення обраної кіберспортивної новини

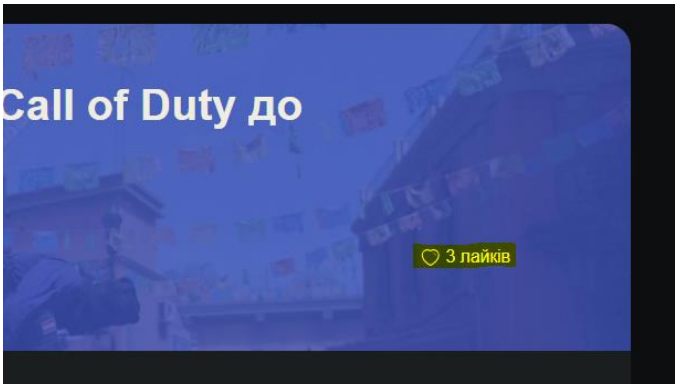

Рисунок 4.37 – Приклад відображення вподобань обраної кіберспортивної новини

Наступною сторінкою є кіберспортивні турніри, де користувач може переглянути як актуальні, так й проведені турніри. Відповідно до кожного із турнірів відображається заголовок, інформація, виграш, кількість поданих заяв та дата створення (рис.4.38).

При наведенні, аналогічно до новин, відбувається зміна кольору (рис.4.39).

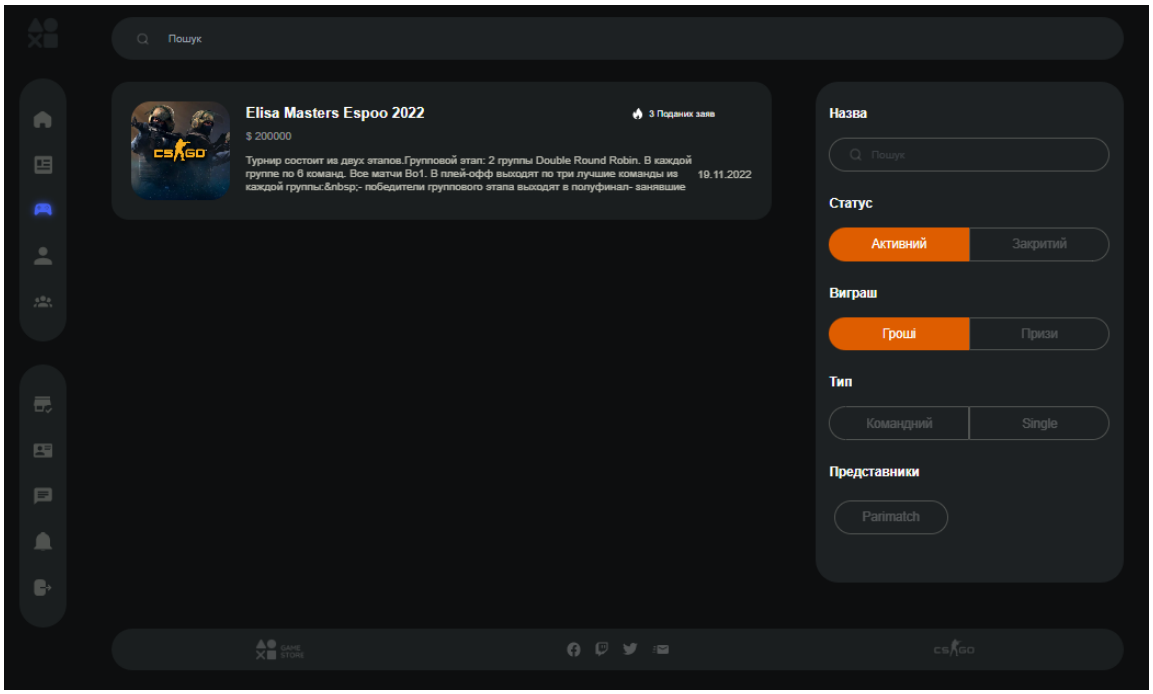

Рисунок 4.38 – Приклад кіберспортивних турнірів

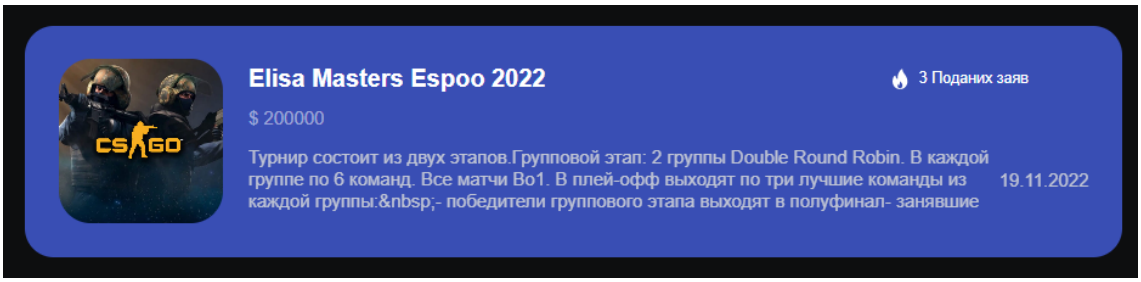

Рисунок 4.39 – Зміна кольору при наведенні

На сторінці також було творено фільтрацію кіберспортивних турнірів за назвою, статусом, типом виграшу, типом турніру та за представником. Після натискання фільтрація відбувається автоматично.

Як приклад, при пошуку вже закритих турнірів в інформаційній системі кіберспортивних турнірів нічого не було знайдено (рис.4.40).

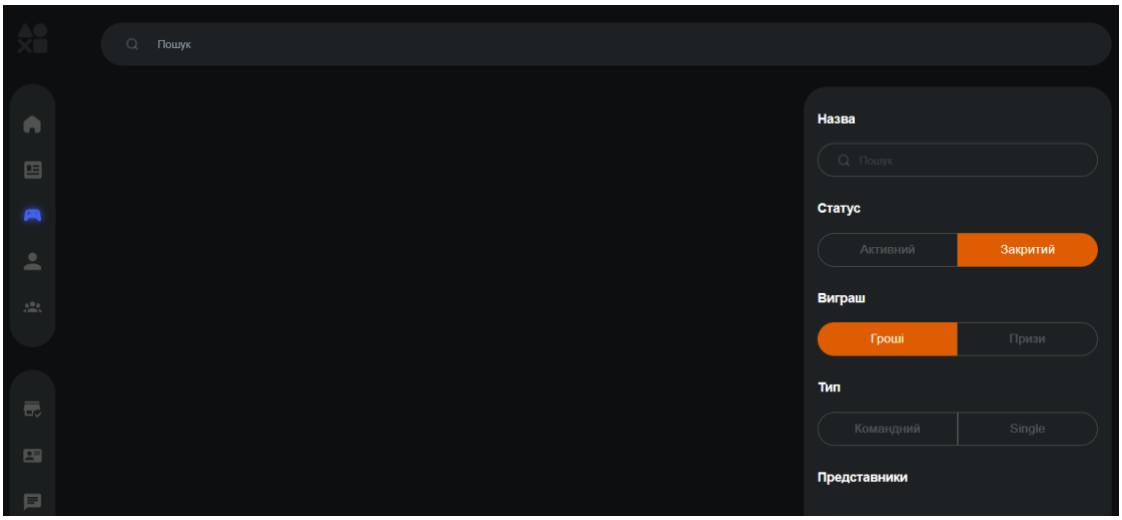

Рисунок 4.40 – Пошук закритих турнірів в інформаційній системі

При переході на сторінку обраного кіберспортивного турніру, користувач може ознайомитися із детальною інформацією про нього (рис.4.41).

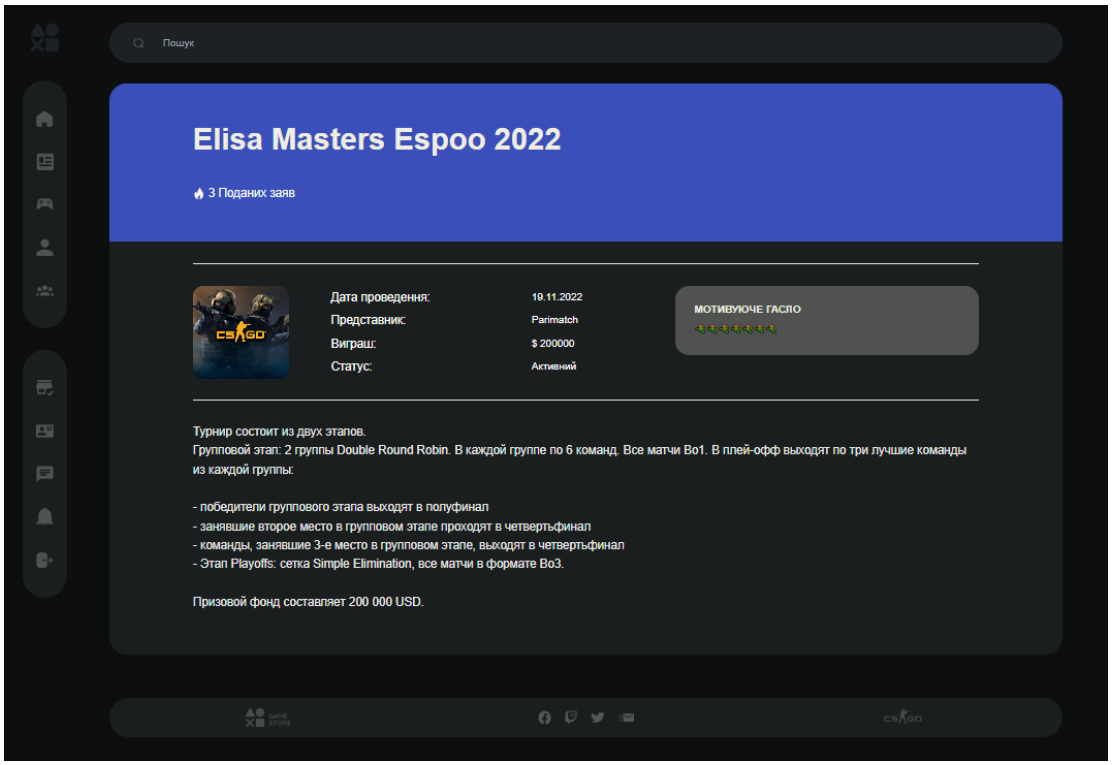

Рисунок 4.41 – Приклад обраного кіберспортивного турніру

Наступною сторінкою є користувачі інформаційної системи для кіберспортивних турнірів з дисципліни CS GO. Відображається фото користувача, прізвище та ім’я, нік з гри (рис.4.42).

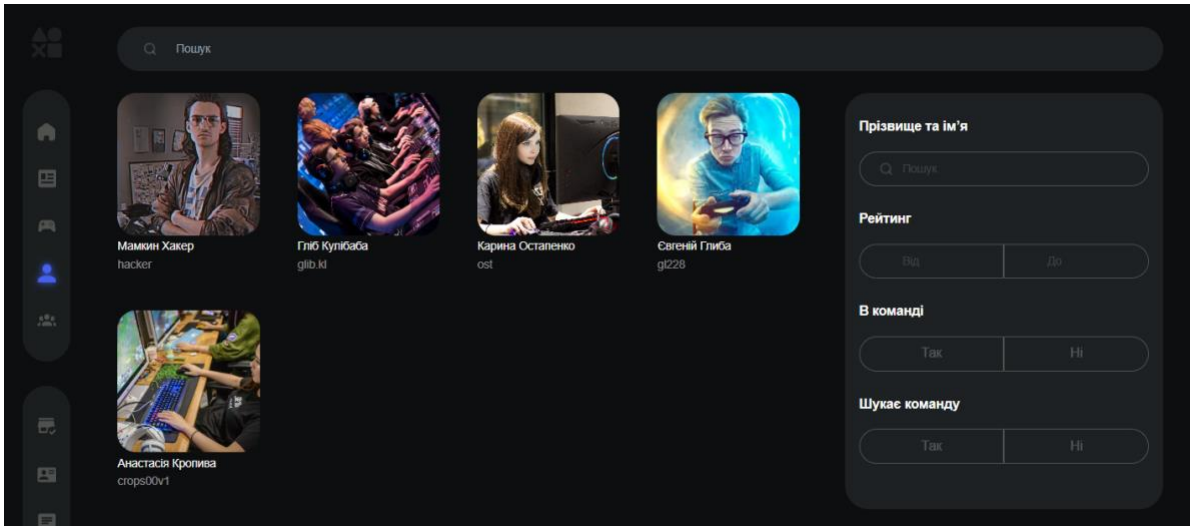

Рисунок 4.42 – Користувачі інформаційної системи для кіберспортивних турнірів

На рисунку 4.43 представлено приклад фільтрації користувачі інформаційної системи для кіберспортивних турнірів. Виконувати фільтрацію можна за прізвищем та ім’ям, рейтингом, чи є в користувача команда, чи ні та, чи шукає він команду.

На прикладі був виконаний пошук користувачів, що не шукають команду, не мають команди та з рейтингом від 1 до 9.

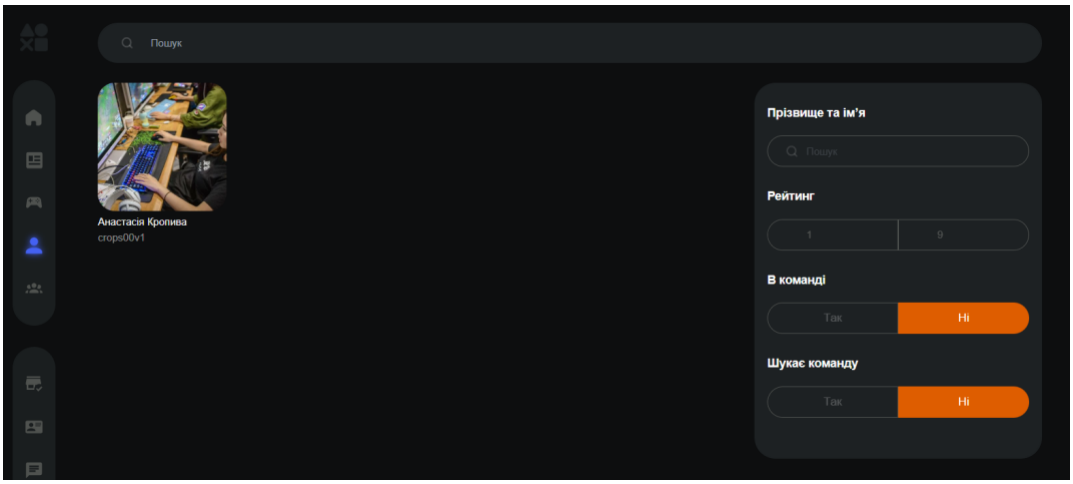

Рисунок 4.43 – Приклад фільтрації користувачів інформаційної системи для кіберспортивних турнірів

Перейдемо на сторінку користувача. Інші користувачі інформаційної системи по кіберспортивних турнірах бачать детальну інформацію про кожного (рис.4.44).

Також на сторінці є інформація про команду, в якій знаходиться користувач. Через цю сторінку можна перейти на сторінку команди та переглянути інформацію про неї.

Крім того, користувачі можуть листуватися саме в інформаційної системи для кіберспортивних турнірів з дисципліни CS GO. Для цього потрібно використати відповідний функціонал та написати іншому обраному користувачеві.

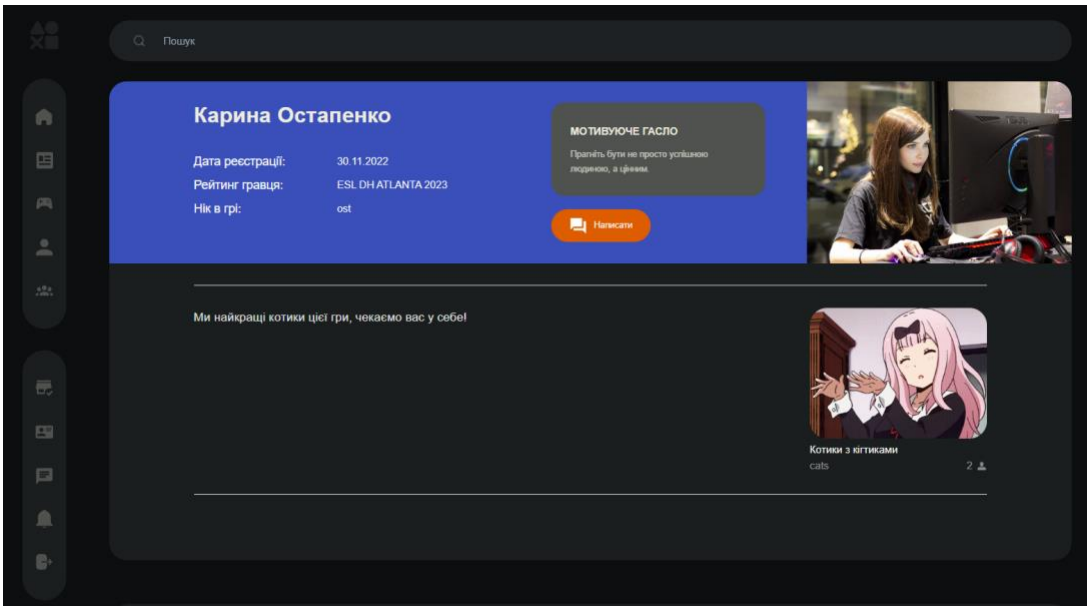

Рисунок 4.44 – Сторінка іншого зареєстрованого користувача інформаційної системи для кіберспортивних турнірів

Прикладом написання повідомлення є рисунок 4.45. Листування є зручним інструментарієм для того, щоб дізнатися актуальну інформацію від іншого користувача чи надати її. Це дуже актуально для інформаційної системи по кіберспортивних турнірів.

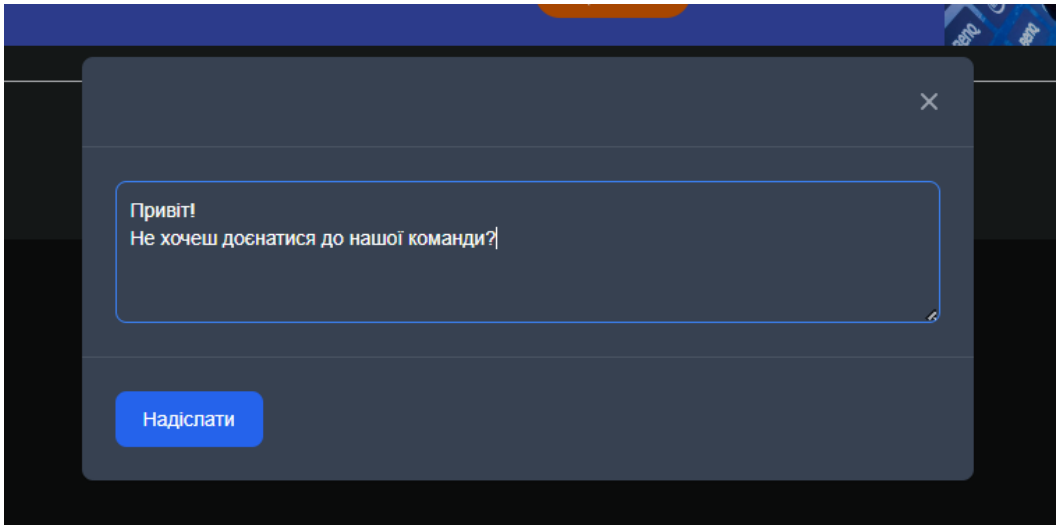

Рисунок 4.45 – Приклад написання повідомлення в інформаційній системі по кіберспортивних турнірів

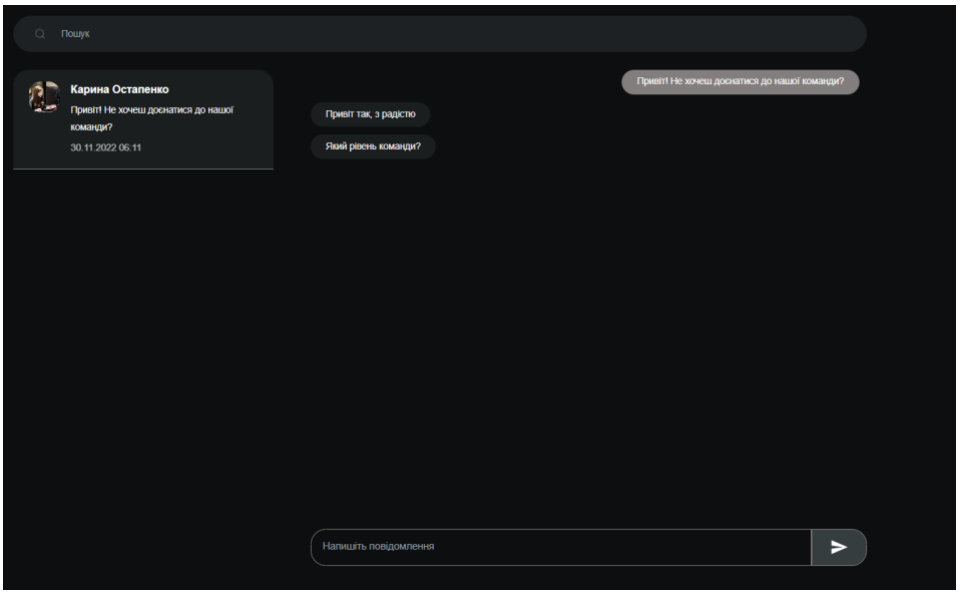

Рисунок 4.46 – Приклад листування в системі по кіберспортивних турнірів

Наступною сторінкою є інформація про команди. Також присутня інформація про кількість учасників у команді, її статус, шукають в команду людей чи ні (рис.4.47).

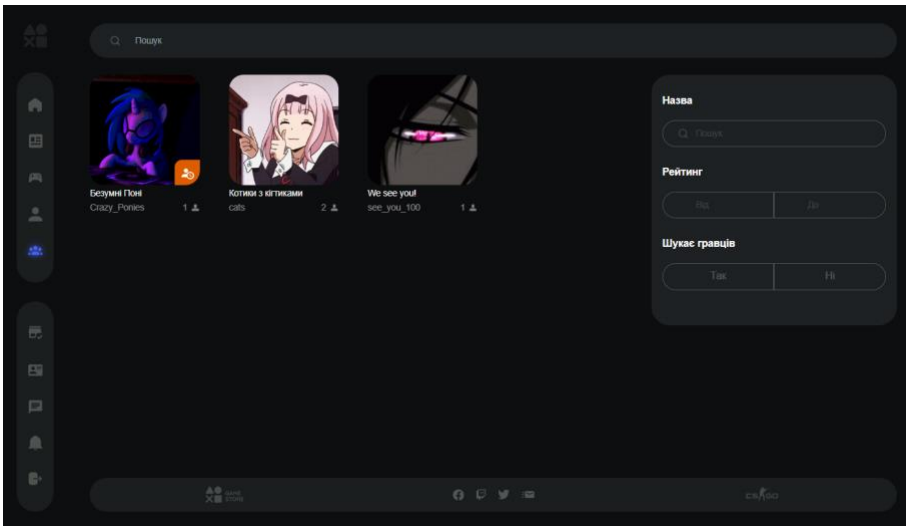

Рисунок 4.47 – Команди в системі по кіберспортивних турнірів

Приклади фільтрації команд користувачів представлена на рисунку 4.48-49. Фільтрація була виконана за рейтингом команди та чи шукає команда гравців.

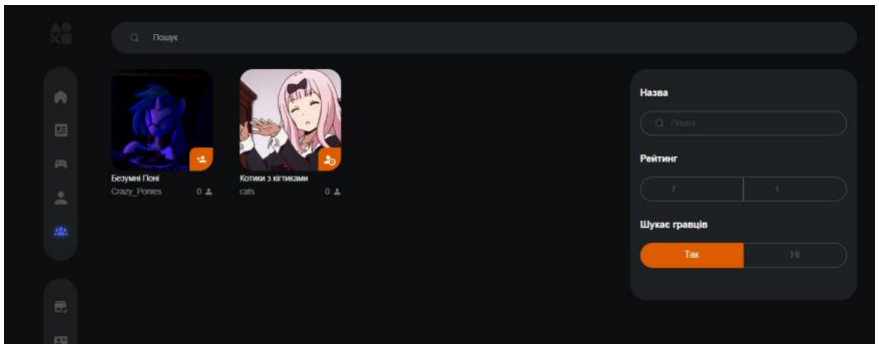

Рисунок 4.48 – Фільтрація команд, що шукають гравців приклад (1)

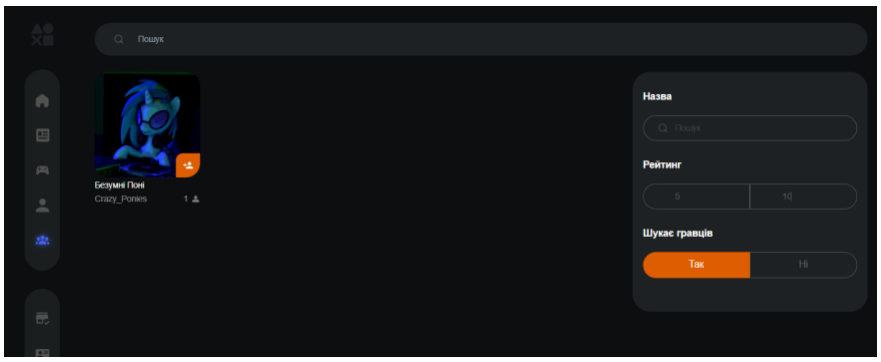

Рисунок 4.49 – Фільтрація команд, що шукають гравців, приклад (2)

Переходячи на сторінку команди, користувач може переглянути інформацію за неї й переглянути учасників команди (рис.4.50).

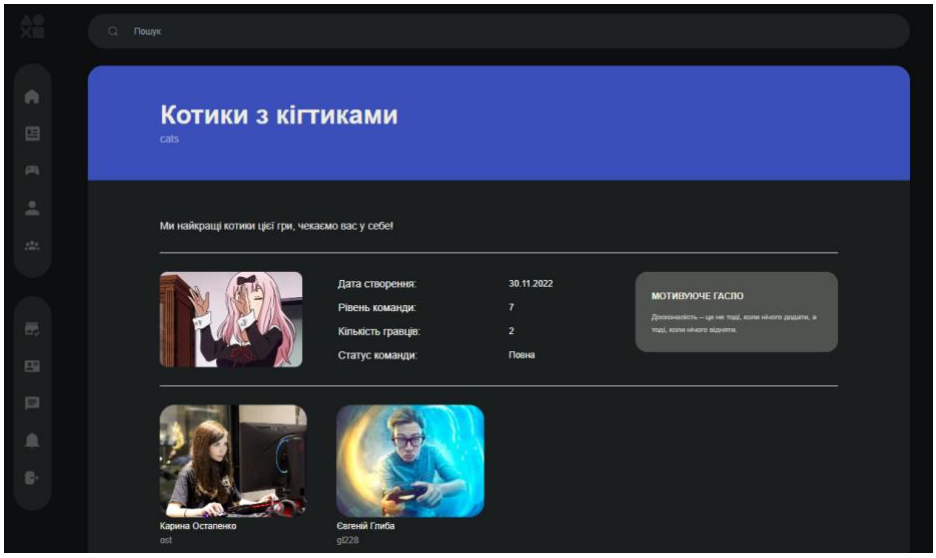

Рисунок 4.50 – Детальна інформація про кіберспортивну команду

Крім того, кожна команда може писати пости на сторінку своєї команди. Це може робити адміністратор команди. Шаблон складає з себе тему та опис (рис.4.51). На рисунку 4.52 представлено приклад створеного та опублікованого посту кіберспортивної команди.

Рисунок 4.51 – Шаблон написання посту від кіберспортивної команди

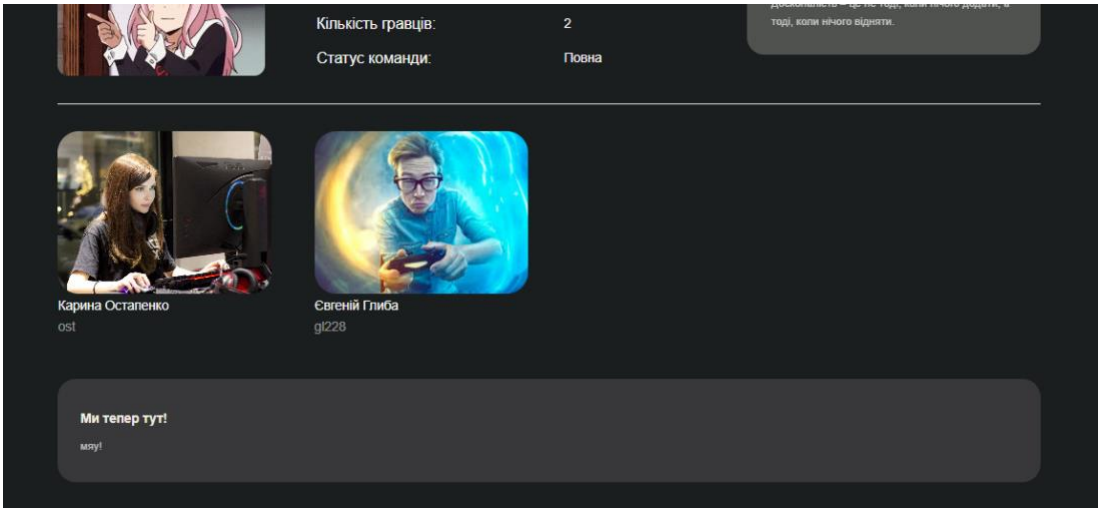

Рисунок 4.52 – Приклад написаного посту від кіберспортивної команди

Наступний функціонал відкриває можливості перед користувачем інформаційної системи можливості по створенню власної команди (рис.4.53). Для створення власної команди потрібно вказати назву, нік команди, орієнтовний рейтинг, чи потрібні гравці, опис та мотивуюче гасло команди.

Також адміністратор команди додає фото команди. Для аватара команди можна використовувати різні формати, від jpeg до gif.

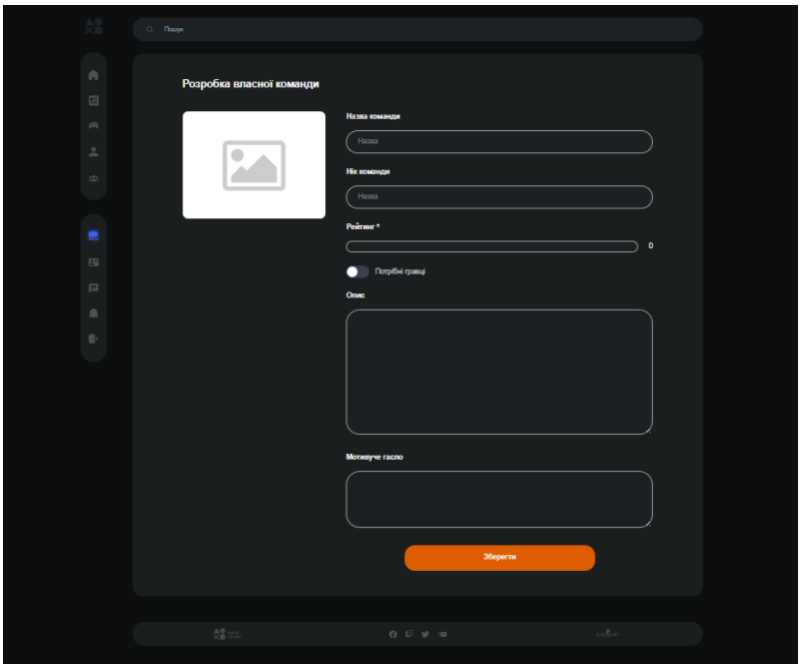

Рисунок 4.53 – Шаблон створення кіберспортивної команди

На наступному рисунку, а саме рисунку 4.54 представлено приклад заповнення інформації що до власної команди.

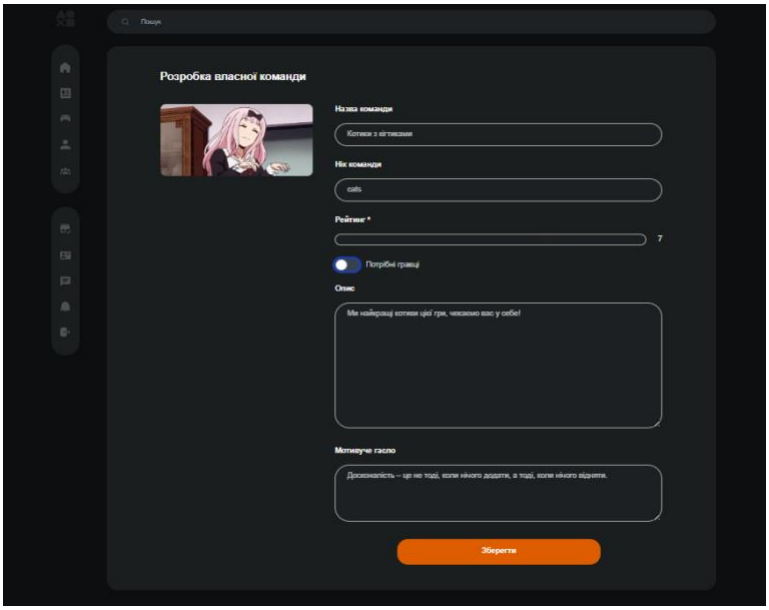

Рисунок 4.54 – Приклад створення кіберспортивної команди

Після створення можна редагувати основу інформацію про кіберспортивну команду (рис.4.55). Також можна запрошувати інших гравців до себе в команду.

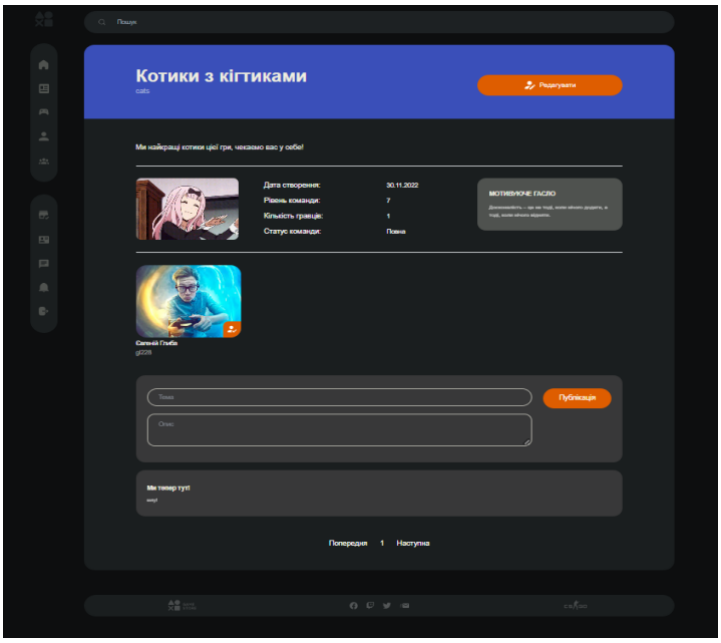

Рисунок 4.55 – Приклад створення кіберспортивної команди

Приклад отримання такого запиту представлена на рисунку 4.56. Так інший користувач отримує запит на окремій сторінці із нотифікаціями.

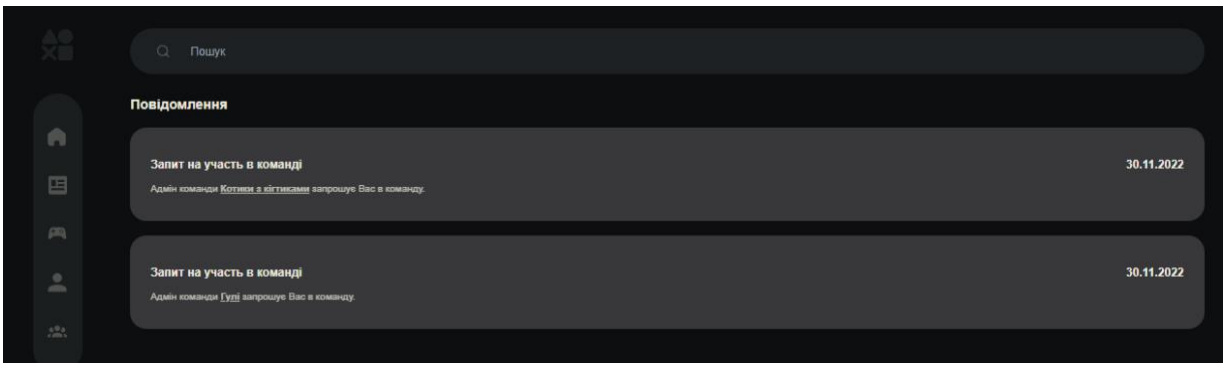

Рисунок 4.56 – Приклад отримання нотифікації від кіберспортивної команди

Перейшовши по запиту від певної команди, користувач може прийняти або відхилити запит (рис.4.57).

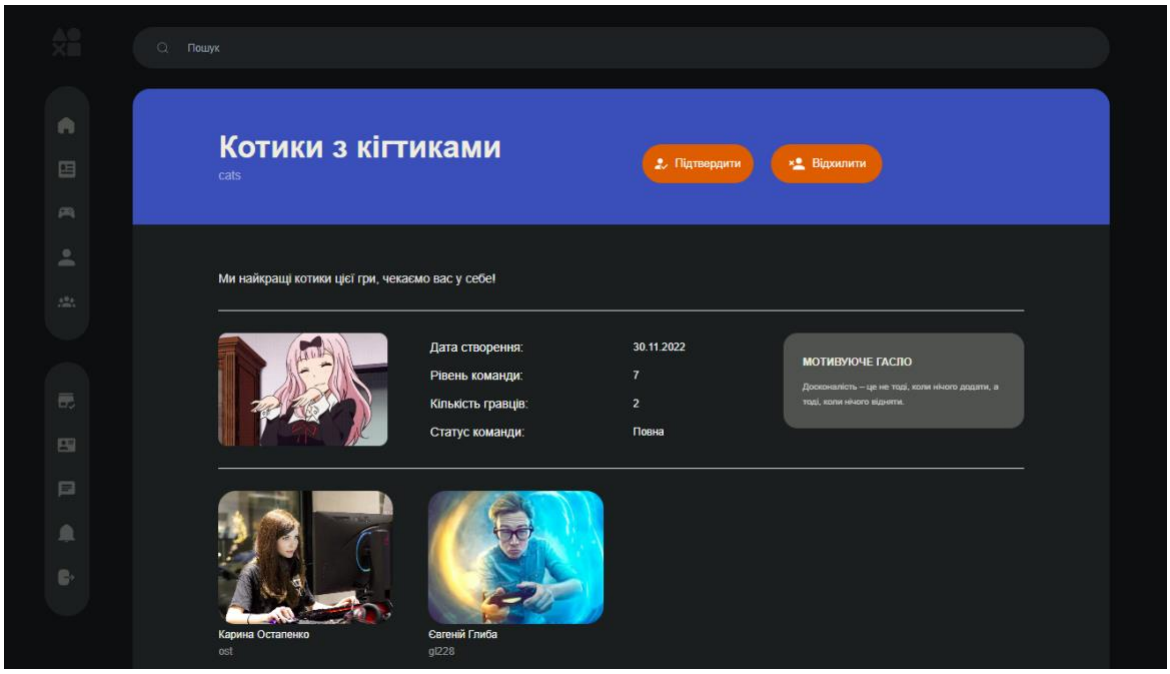

Рисунок 4.57 – Приклад реакції на отриманий запит

4.3 Адміністрування інформаційної системи

Вхід на панель адміністратора проходить на спеціальну сторінку, де адміністратор повинен заповнити поля електронної пошти та паролю (рис.4.58).

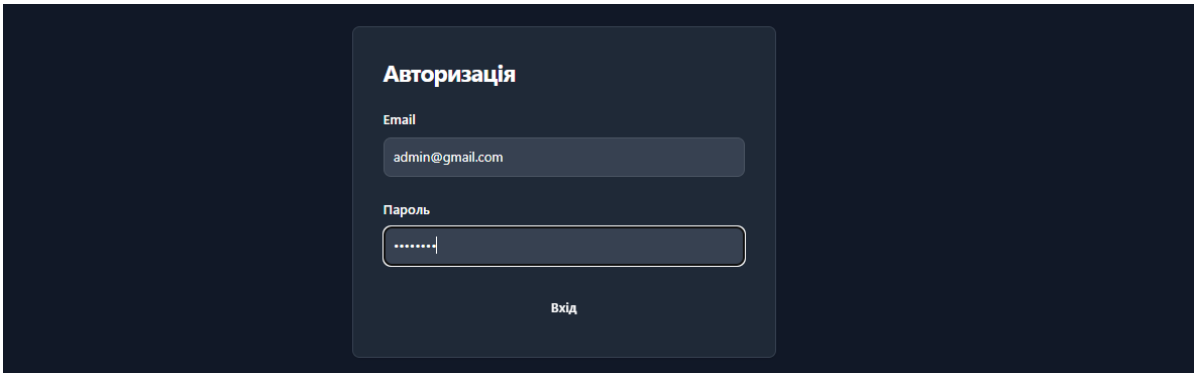

Рисунок 4.58 – Вхід в панель адміністратора

Після авторизації адміністратора, відкривається головна сторінка з відображенням навігаційного меню, статистичної інформації та актуальних турнірів (рис.4.59).

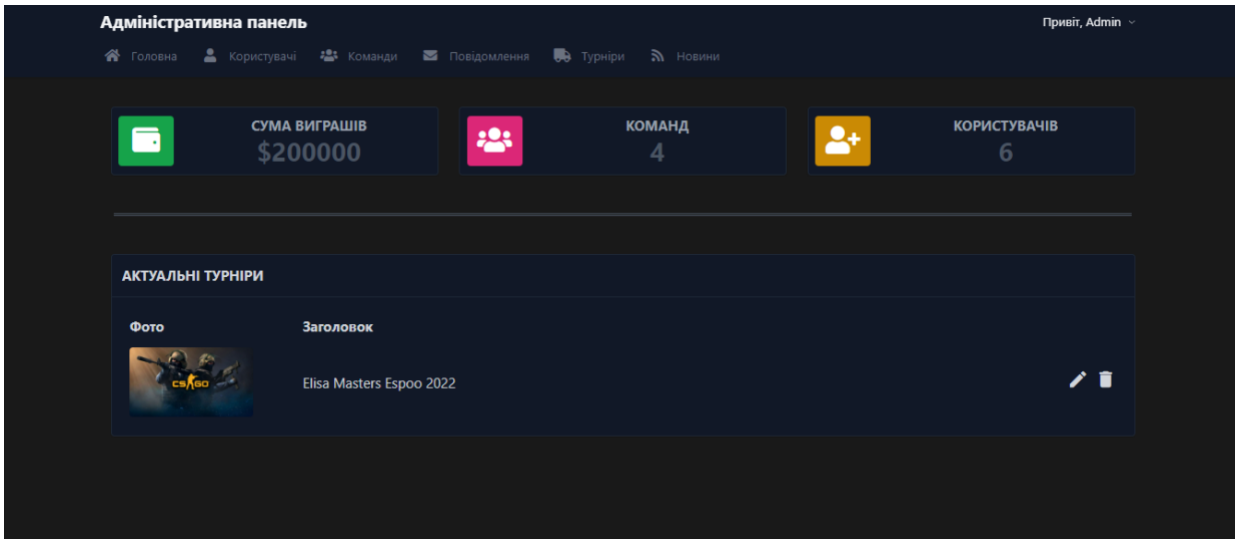

Рисунок 4.59 – Приклад реакції на отриманий запит

З головної сторінки є можливість редагувати та видаляти турніри, для цього потрібно натиснути на бажаний турнір та після цього можна редагувати всю інформацію про турнір.

При редагуванні обраного турніру можна редагувати загальну інформацію та додати результати турніру. Цей функціонал потрібен для публікації результатів турнір в інформаційній системі.

Для додавання та зручного форматування опису та самих результатів було додано зручний функціонал, в якому можна повноцінно змінювати текст. Прикладом є рисунок 4.60.

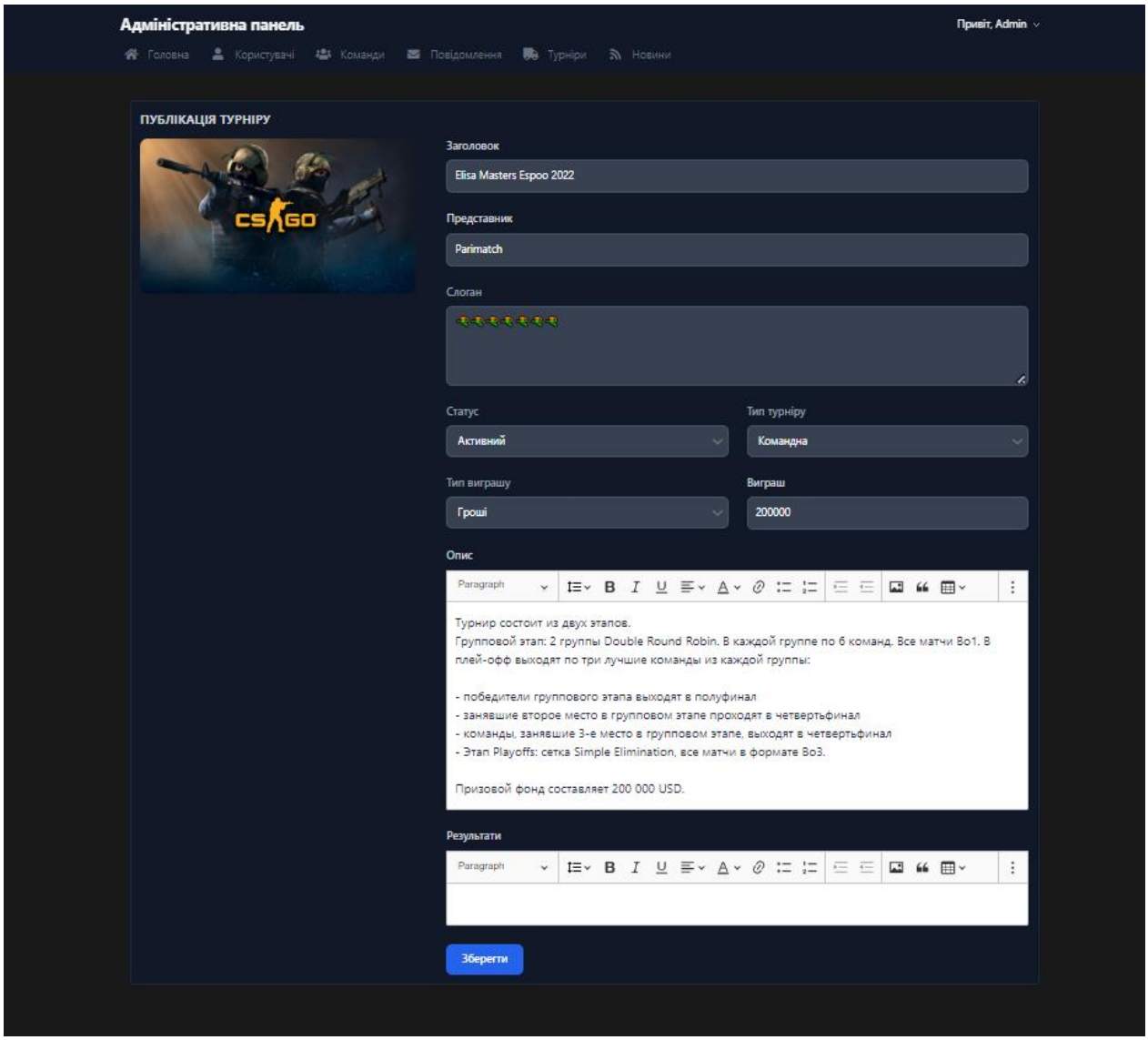

Рисунок 4.60 – Редагування турніру

Також, створювати редагувати та видаляти турніри можна через «Турніри». Щоб створити турнір, потрібно заповнити всі ті ж поля, що надаються для редагування (рис.4.61).

Фото додаються у будь-якому форматі, залежить від потреб організаторів турніру. Якісне відображення було реалізовано при розробці.

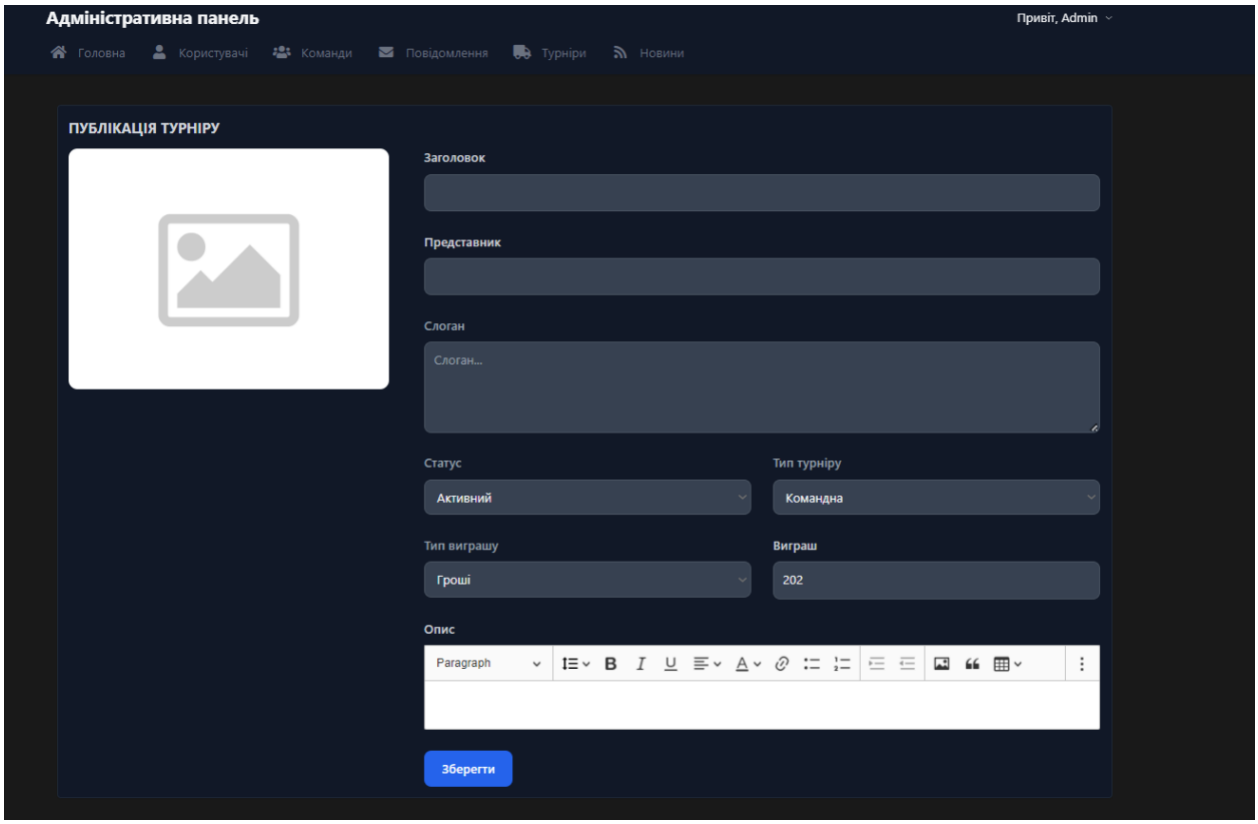

Рисунок 4.61 – Створення нового турніру

На панелі «Користувачі» можна переглядати інформацію зареєстрованих користувачів, блокувати або розблоковувати їх. Після блокування, або розблокування користувача на екрані з’являється сповіщення (рис.4.62).

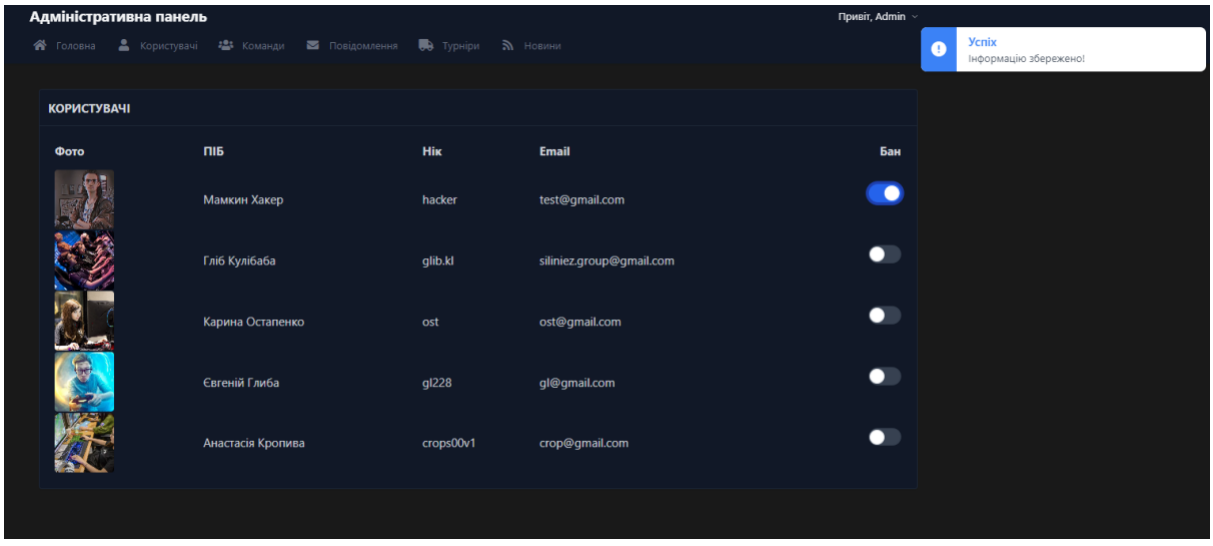

Рисунок 4.62 – Панель «Користувачі»

На панелі «Команди» надається можливість переглядати інформацію створених команд та їх блокування. Блокування команди остаточне, та розблокуванню не підлягає (рис.4.63).

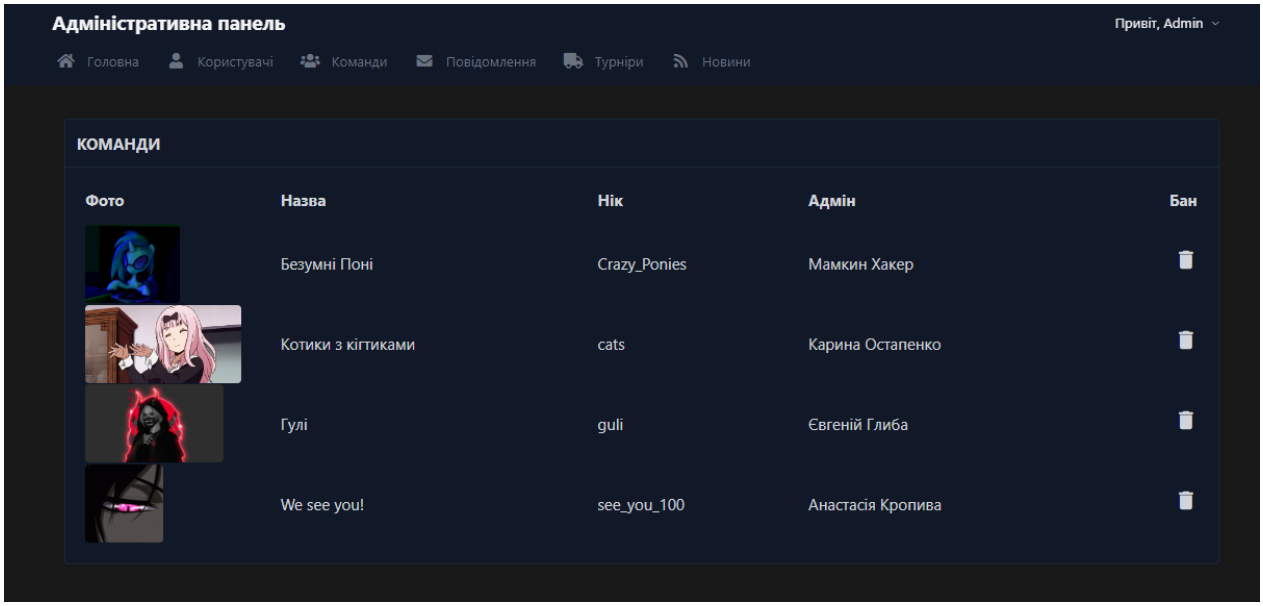

Рисунок 4.63 – Панель «Команди»

На панелі «Повідомлення» відкривається можливість проводити публікацію турнірів, або робити розсилку для всіх користувачів в полі інформаційної системи. Поля для заповнення, а саме «Заголовок» та «Повідомлення» – обов’язкові для заповнення адміністратором (рис.4.64).

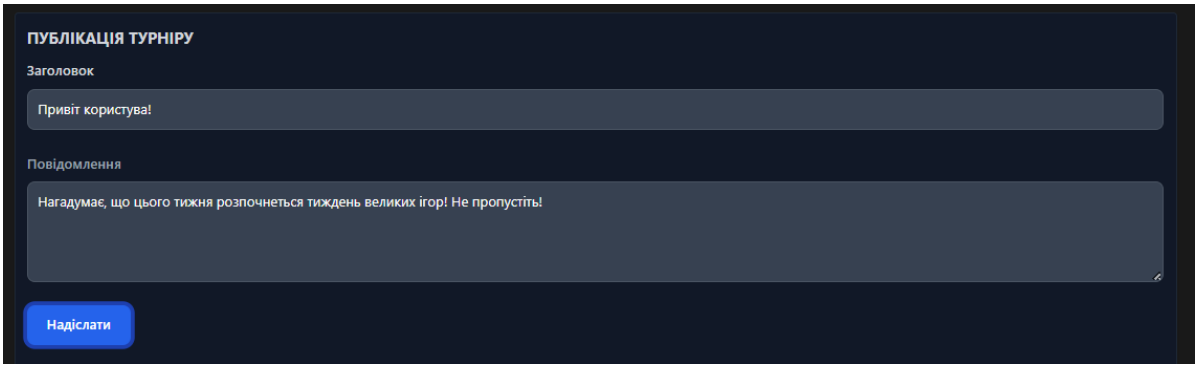

Рисунок 4.64 – Приклад реакції на отриманий запит

На панелі «Команди» надається можливість переглядати інформацію створених команд та їх блокування. Блокування команди остаточне, та розблокуванню не підлягає (рис.4.65).

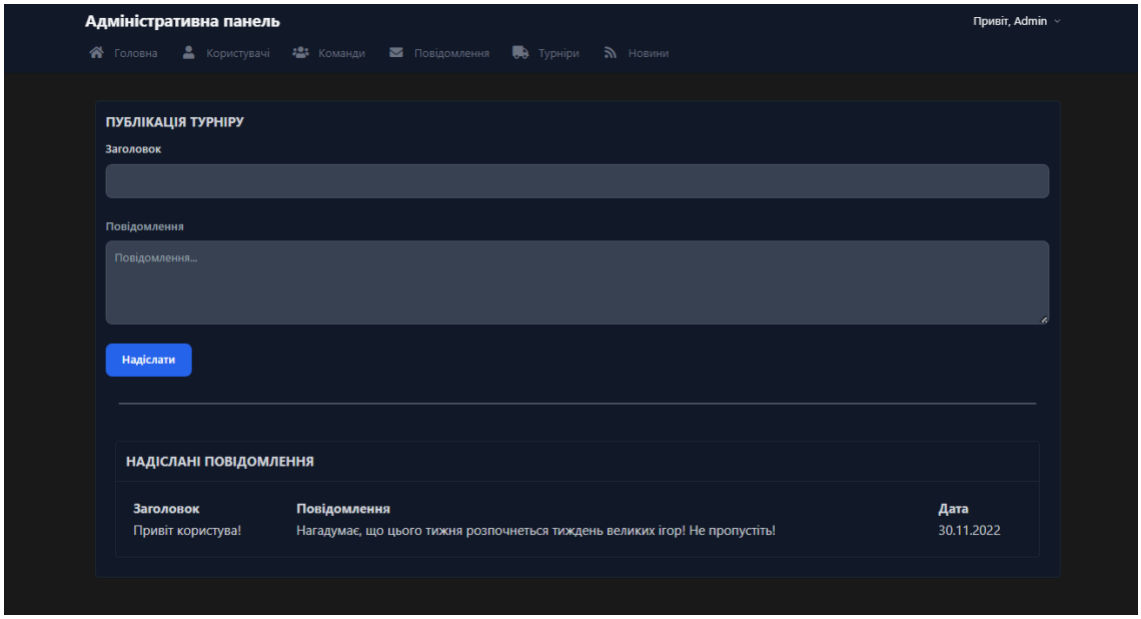

Рисунок 4.65 – Приклад реакції на отриманий запит

Публікація новини, яка буде відображатися на відповідній веб-сторінці, створюється через панель адміністратора. Адміністратор може не лише створити новину, а й редагувати та видаляти новини (рис.4.66-67).

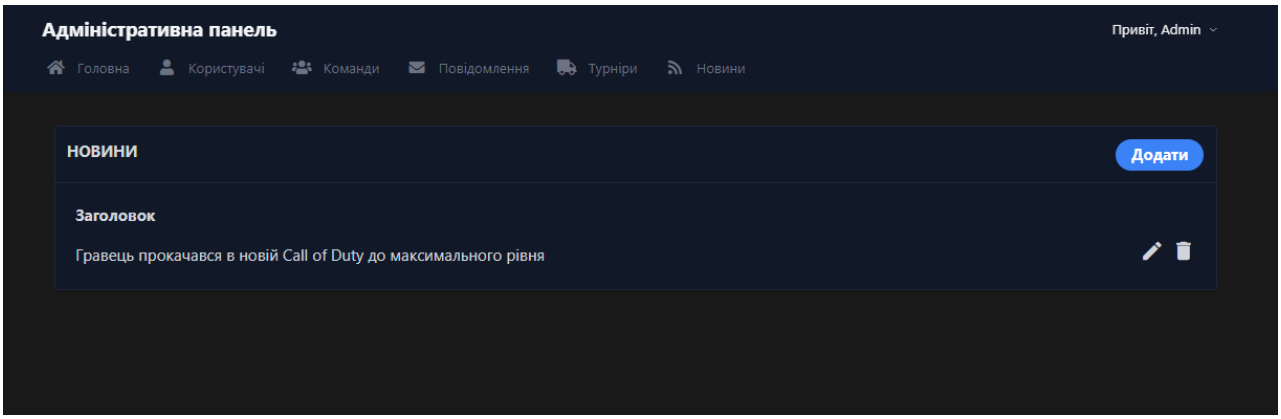

Рисунок 4.66 – Панель «Новини»

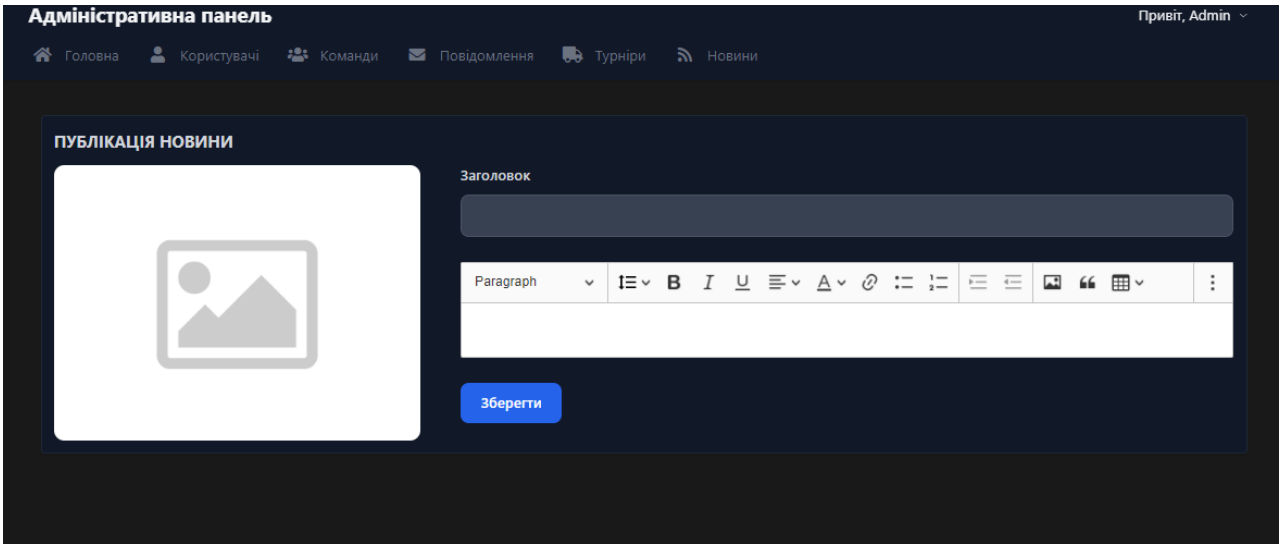

Рисунок 4.67 – Створення нової новини

Адміністратор може змінити власний пароль за допомогою налаштування акаунту адміністратора (рис.4.68).

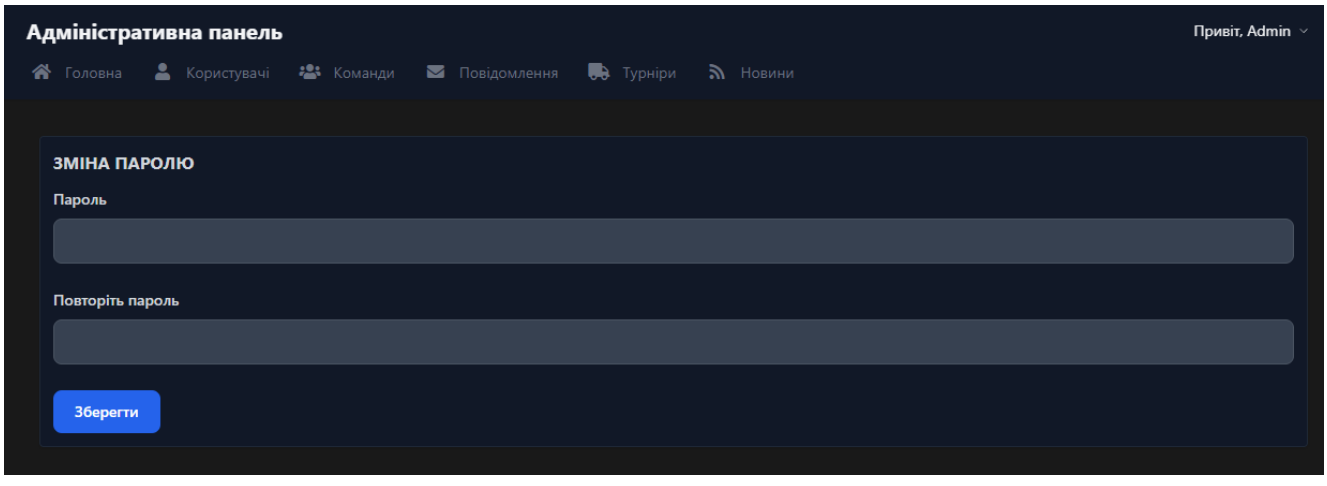

Рисунок 4.68 – Зміна пароля адміністратора

ВИСНОВКИ

У результаті роботи над кваліфікаційною роботою магістра було досліджено та проаналізовано актуальні сайти конкурентів у кіберспортивному напрямі, а саме дисципліні CS GO. На основі цього аналізу були виявлені як недоліки, так й переваги.

Було виконано реалізацію всього необхідного переліку функціонала, що забезпечить полегшення пошуку членів команди для участі в турнірах, розповсюдження та популяризацію даної кіберспортивної дисципліни.

При виконанні кваліфікаційної роботи було поставлено головну мету розробки інформаційної системи по кіберспортивних турнірах. Після аналізу було обрано програмне забезпечення та інструментарій.

Крім того, було продемонстровано приклади використання інформаційної системи для кіберспортивних турнірів з дисципліни CS GO зі сторони звичайного користувача та адміністратора. Весь функціонал був продемонстрований й детальній варіації.

Результатом кваліфікаційної роботи магістра є розроблена інформаційна система для кіберспортивних турнірів з дисципліни CS GO. Адже практичне значення інформаційної системи для кіберспортивних турнірів є спрощення організації та проведення турнірі. Це дозволить користувачам створювати власні команди та брати участь у турнірах.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. How technology determines the growth of the esports industry [Електронний ресурс] – Режим доступу до ресурсу: URL: https://techweez.com/2021/11/23/technology-growth-esports-industry/ (дата звернення: 02.10.2022);

2. Waste Paper Managment. International Journal of Engineering and Advanced Technology. 8, 96–100 (2020));

3. F. LUNENBURG, LEADERSHIP VERSUS MANAGMENT: A KEY DISTINCTION—At LEAST IN THEORY. Main Issues Of Pedagogy And Psychology. 3, 15–18 (2020));

4. K. Rudolf et al., Demographics and health behavior of video game and esports players in germany: The esports study 2019. International Journal of Environmental Research and Public Health. 17 (2020);

5. B. H. K. S. Brenda, in eSports Yearbook 2015/16, J. Hiltscher, T. M. Scholz, Eds. (Books on Demand GmbH, 2017), pp. 9–35;

6. M. A. Pluss et al., Esports: The chess of the 21st century. Frontiers in Psychology. 10 (2019);

7. A. Bascón-Seda, A. R. Rodríguez-Sánchez, Esports and science: Tuning into the phenomenon of electronic sports. Cultura, Ciencia y Deporte. 15, 341–352 (2020).

8. T. Iwatsuki, G. Hagiwara, M. E. Dugan, Effectively optimizing esports performance through movement science principles. International Journal of Sports Science and Coaching. 17 (2022), pp. 202–207.

9. How advanced technology drives the esports industry forward [Електронний ресурс] – Режим доступу до ресурсу: URL: https://www.msn.com/en-us/news/technology/how-advanced-technology-drives-the-esports-industry-forward/ar-AAXG3og (дата звернення: 05.10.2022);

10. D. Illy, J. Florack, in Ratgeber Videospiel- und Internetabhängigkeit (Elsevier, 2018), pp. 25–29;

11. The Rise of Esports, How Data Drives It, and What the Future Looks Like [Електронний ресурс] – Режим доступу до ресурсу: URL: https://www.gameopedia.com/the-rise-and-future-of-esports/ (дата звернення: 06.10.2022);

12. E. Hayday, H. Collison-Randall, S. Kelly, in Esports Insights (Routledge, 2022), pp. 1–20, pp. 76–90;

13. H. Collison-Randall, Esports (Routledge, 2022);

14. C. A. West, K. Hinerman, thesis, Lamar University - Beaumont (2020);

15. «Gameinside.ua» [Електронний ресурс] – Режим доступу до ресурсу: URL: https://gameside.ua/news/ (дата звернення: 18.10.2022);

16. «Upea.com.ua» [Електронний ресурс] – Режим доступу до ресурсу: URL: https://upea.com.ua (дата звернення: 18.10.2022);

17. «Uesf.org.ua» [Електронний ресурс] – Режим доступу до ресурсу: URL: https://uesf.org.ua (дата звернення: 18.10.2022);

18. M. A. Hakam, A. Triayudi, N. Hayati, Implementasi Metode Agile pada Sistem Manajemen Zakat Berbasis Website dengan Framework Laravel. Jurnal JTIK (Jurnal Teknologi Informasi dan Komunikasi). 6, 111–116 (2022);

19. A. R. Saulina Panjaitan, U. Rahardja, Q. Aini, N. P. Lestari Santoso, D. Apriliasari, Management Innovation of Kuliah Kerja Praktek (KKP). APTISI Transactions on Management (ATM). 6, 62–73 (2022);

20. A. Alip, S. Kosasi, I. D. A. E. Yuliani, G. Syarifudin, D. David, Implementasi Arsitektur Model View Controller Pada Website Toko Online. Jurnal Bumigora Information Technology (BITe). 3, 135–150 (2022);

21. H. Samosir, T. A. Prasetyo, S. Lumbantobing, D. O. Naibaho, C. R. T. Situmorang, in 17th International Conference on Quality in Research, QIR 2021: International Symposium on Electrical and Computer Engineering (Institute of Electrical and Electronics Engineers Inc., 2021), pp. 60–65;

22. F. Alfiah, R. P. Mufti Al Rasyid, U. Umriyah, Sistem Advertensi Properti Berbasis Laravel Framework Pada PT. Mekar Baru Properti Indonesia. Journal Sensi. 8, 97–107 (2022) ;

23. M. Y. Effendy, E. Nurninawati, A. Ari Setiyawan, Design And Build A Web-Based Asset Management Information System at Pt Thamrin Telekomunikasi Network. Aptisi Transactions on Technopreneurship (ATT). 4, 48–58 (2022).

ДОДАТОК А

ПЛАНУВАННЯ РОБІТ
для розробки кваліфікаційної роботи магістра «Інформаційна система для кіберспортивних турнірів з дисципліни CS GO»

А.1 ІДЕНТИФІКАЦІЯ МЕТИ ІТ-ПРОЕКТУ

Метою проекту є розробка інформаційної системи для кіберспортивних турнірів з дисципліни CS GO.

1.1.1 Деталізація мети методом SMART:

S – Метою проекту є розробка інформаційної системи для кіберспортивних турнірів з дисципліни CS GO за допомогою якого користувач прийняти участь у турнірах.

M – Система зможе виступити як платформа, за допомогою якого можна буде популяризувати і просувати кіберспортині турніри.

A – Головною ціллю є популізувати кіберспортині турніри.

R – Крім популяризації проект можна монетизувати, за рахунок збору коштів на проведення подій або ж просто залучення спонсорів.

T – на розробку такого проекту потрібно – 6 тижнів. В розрахунку, що в середньому буде використовуватись 6 годин в день.

А.2 ПЛАНУВАННЯ ЗМІСТУ СТРУКТУРИ РОБІТ ІНФОРМАЦІЙНОЇ СИСТЕМИ

WBS (рис. А.2.1) є ієрархічною структурою, що показує подальший розподіл необхідних для виконання мети зусиль; наприклад, програма, проект чи договір. Інструмент спрямований на детальне планування, оцінку вартості, визначення та розподіл персональної відповідальності виконавців та інші - тобто, на основні роботи і результати, що визначають зміст проекту.

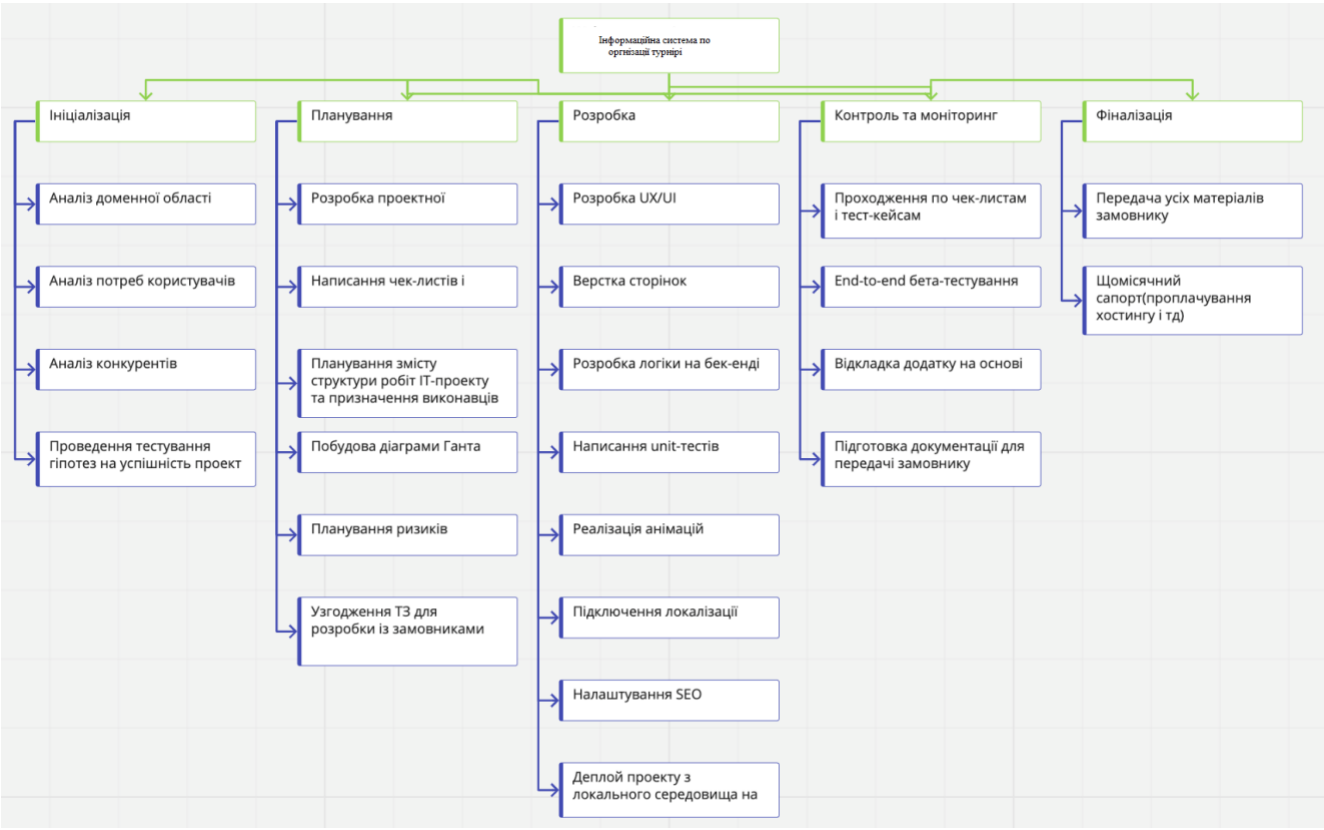

Рисунок А.2.1 – WBS діаграма

Після того, як була створена WBS структура проекту, наступним необхідним етапом є розробка OBS (рис.А.3.2). Організаційна структура представляє собою графічне відображення учасників проекту та їх відповідальних осіб, які задіяні в реалізації проекту.

Задля того, щоб максимально зменшити бюджет проекту було прийнято рішення взяти на роботу middle Project Manager із уміннями бізнес аналітика, який буде слідкувати за бізнес частиною процесу і відповідати за розробку. Також було прийнято рішення не брати devops'а тому, що зараз ця позиція занадто дорого коштує на ринку. Ми не будемо використовувавти клаудні рішення, тому хоститися сайт буде на звичайному сервері, і перенести і налаштувати його зможе той же самий PM або розробник.

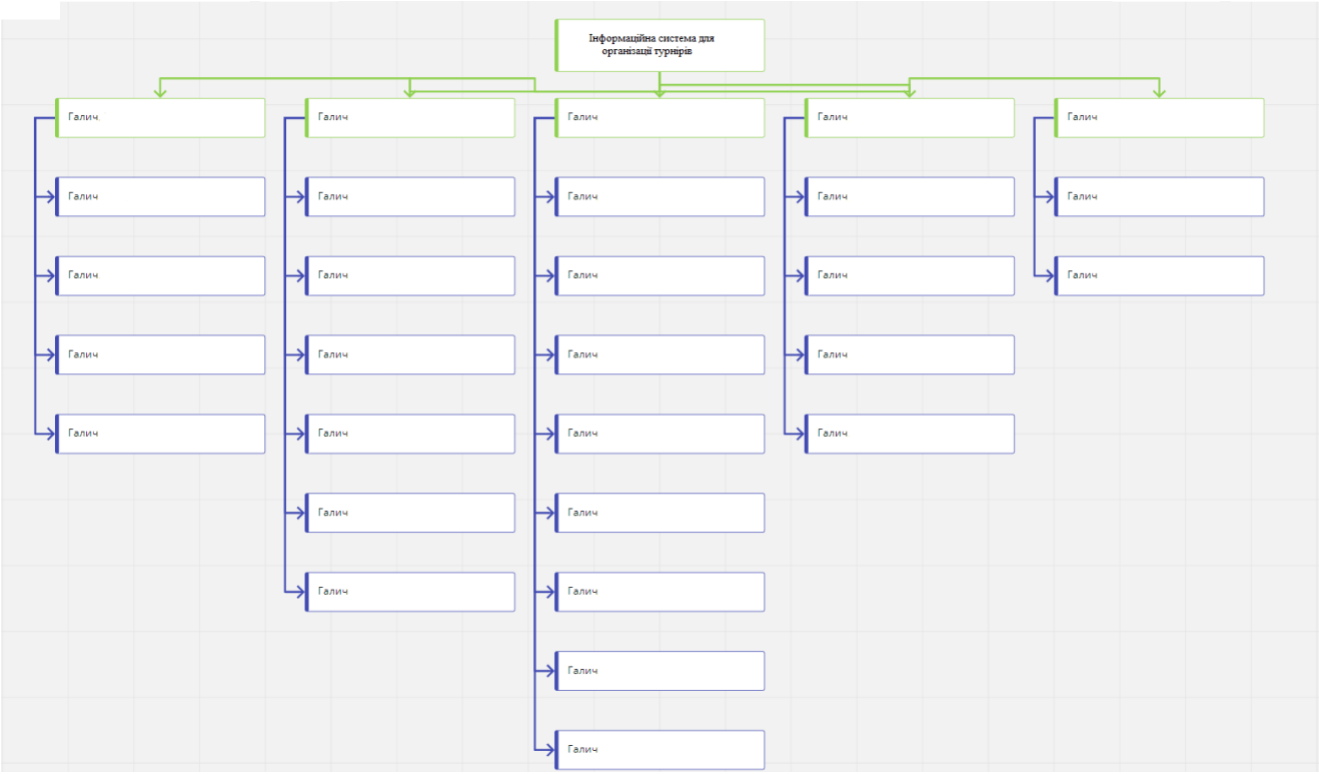

Рисунок А.2.2 – OBS діаграма

А.3 ПОБУДОВА КАЛЕНДАРНОГО ГРАФІКУ ВИКОНАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

Діаграма Ганта – це горизонтальна діаграма з тимчасової шкалою, яка використовується для ілюстрації плану робіт за проектом з прив'язкою до часу.

За допомогою діаграм Ганта керівники проектів і менеджери по продукту розбивають проекти на робочі завдання для зручності управління, підтримують порядок в роботі і роблять залежності між завданнями наочними.

Діаграми Ганта дозволяють спростити складові проекти. За допомогою цього засобу можна досить наочно і зручно для узагальнення представити велику кількість даних. Завдяки цій гістограмі велика кількість зацікавлених осіб, команд або їх учасників не стане проблемою для запису завдань, як і часті зміни обсягу роботи. Ще одна перевага використання діаграми Ганта полягає в тому, що вона дає загальне уявлення про проект в цілому, в тому числі про всі контрольних точках і терміни виконання. Діаграму Ганта можна уявити як ефективний засіб раннього попередження.

Розглянемо створену діаграму Ганта до заданої інформаційної системи (рис.А.2.3-4).

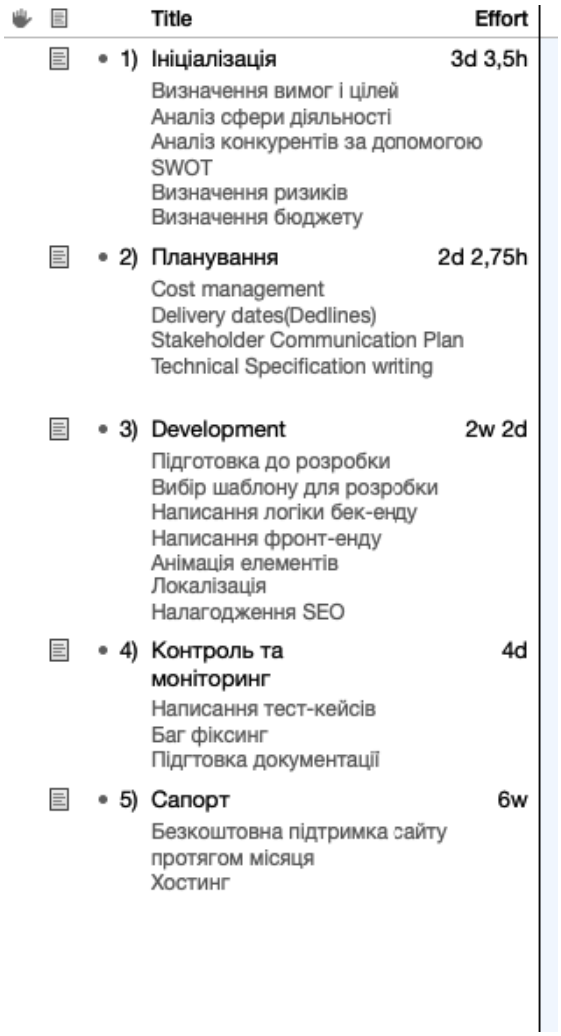

Рисунок А.2.3 – Діаграма Ганта(1)

1.2

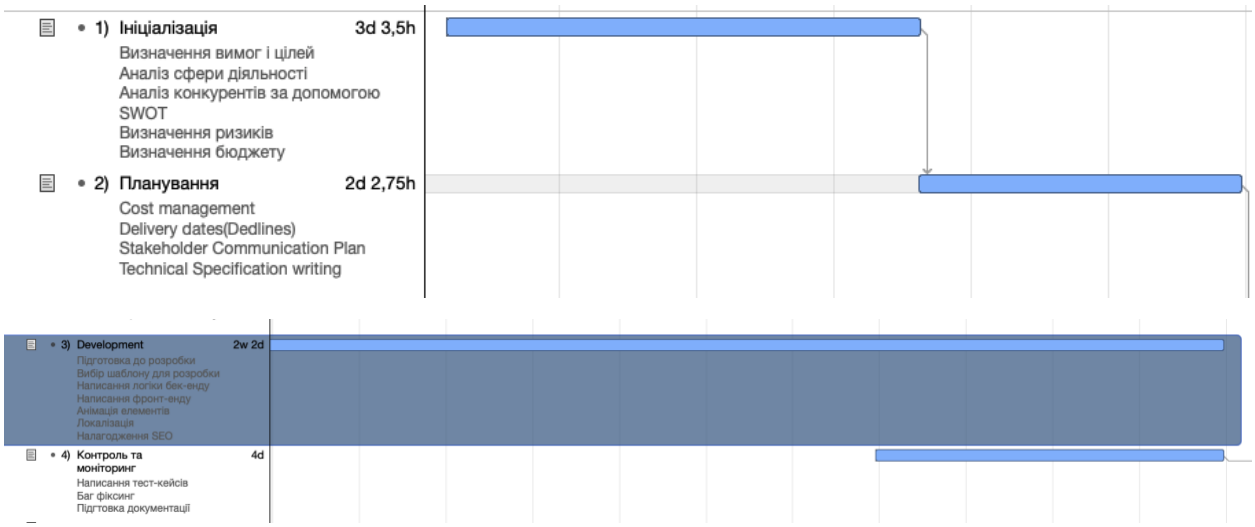

Рисунок А.2.4 – Діаграма Ганта(2)

Дата початку проекту – 01.09.2022. На виконання проекту відводиться по 6 годин на добу. Робочими днями вважаються також вихідні та святкові дні.

Як видно з графіку виконання проекту загальний час, який відводиться на його виконання – 57 днів робочих днів. Оцінка давалась методом експертної оцінки і аналогової, на основах старих проектів. До кожної оцінки було додано 30% ризику, щоб закласти час та виконати все вчасно.

А.4 ПЛАНУВАННЯ РИЗИКІВ ПРОЕКТУ

Ризиком є ймовірна подія, яка у випадку її виникнення може як негативно, так і позитивно вплинути на конкретний проект. Управління ризиком – це процес зміни ризиків та реагування на події під час виконання проекту. При реалізації проекту важливою частиною є моніторинг ризиків. Отже, в даному випадку можна виділити деякі ризики.

Під час процесу аналізу для визначення числових значень ймовірності появи ступеня впливу, застосовувалася методика експертних оцінок. Виходячи цих оцінок можливо знайти ранг ризиків: R = P * L, де

- R – ранг ризику;
- P – ймовірність виникнення;
- L – ступінь впливу.

Шкала оцінки ризику може відповідати емпіричній шкалі оцінки ризику:

- 5 балів - критичний ризик (0,81 - 1);
- 4 бали - максимальний ризик (0,61 - 0,8);
- 3 бали - високий ризик (0,41 - 0,6);
- 2 бали - нормальний ризик (0,31 - 0,4); 1 бал - малий ризик (0 - 0,3).

Таблиця Б.4.1 – Розрахунок ризиків

N	Назва ризику	Опис	P	L	R	План реагування
1	Розширення обсяг робіт	Висунутий додатковий функціонал, який впливає на скоуп	0,7	0,8	0,56	У разі настання ризику бюджет буде розширено, тому потрібно застрахуватись і відкласти гроші на додаткові 20 годин

N	Назва ризику	Опис	P	L	R	План реагування
2	Відставання від графіку	По ряду причин відбувається відставання від дедлайну	0,6	0,6	0,36	В такому разі прийдеться або скороити скоуп робіт, або витратити додаткові кошти на додаткові робочі години розробника. До 20 годин закласти кошти ще на 10 годин
3	Проблеми з тенікою	По ряду причин, можуть виникнути проблеми з технікою, на якій виконується розробкак	0,3	0,9	0,27	Якщо таке станеться, проект може бути провалений. Тому краще, мати додаткову можливість використовувати іншу техніку

ДОДАТОК Б

КОД РЕАЛІЗАЗІЇ ІНФОРМАЦІЙНОЇ СИСТЕМИ
для розробки кваліфікаційної роботи магістра
«Інформаційна система для кіберспортивних турнірів
з дисципліни CS GO»

UserController.php

Контролер для роботи з користувачами.

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use Illuminate\Support\Facades\Validator;
use Illuminate\Support\Facades\Auth;
use Illuminate\Support\Facades\Hash;

use App\Models\User;
use App\Models\UserPublications;
use App\Models\UserHasTeams;
use App\Models\Notifications;
use App\Models\Teams;

class UserController extends Controller
{
   protected $publicStorage = "/uploads/users/";

   function profile()
   {
      $model = User::with('publications', 'userTeam.team.users', 'userTeam.team.acceptUsers', 'myTeam.users')-
>find(Auth::id());
      return response()->json($model);
   }

   function profileUpdate(Request $request)
   {
      $model = User::find(Auth::id());
      $data = $request->all();
      $data['need_team'] = $request['need_team'] == 'true' ? 1 : 0;
      if (isset($request['new_photo']) && $request['new_photo'] != "null" && $request['new_photo'] != "undefined") {
         $name =  $this->publicStorage . uniqid() . '.' . $request['new_photo']->getClientOriginalExtension();
         $request['new_photo']->move(public_path() . $this->publicStorage, $name);
         $data['photo'] = $name;
      }
      $model->update($data);

      return response('ok', 200);
   }

   function update(Request $request, $id)
   {
      $model = User::find($id);
      $data = $request->all();
      $data['is_ban'] = $request['is_ban'] == 'true' ? 1 : 0;
      $model->update($data);

      return response('ok', 200);
   }

   function postPublication(Request $request) {
      $data = $request->all();
      $data['user_id'] = Auth::id();
      $model = new UserPublications();
      $response = $model->create($data);
```

```
        return response()->json($response);
    }

    function delPublication($id) {
        UserPublications::find($id)->delete();
    }

    function get(Request $request)
    {
        $model = User::where('role', 'user', 'userTeam');
        if($request->title) {
            $model->where('name', 'like', '%' . $request->name . '%');
        }
        if($request->rate_from) {
            $model->where('level', '<=', $request->rate_from);
        }
        if($request->rate_to) {
            $model->where('level', '>=', $request->rate_to);
        }
        if($request->in_team == 1) {
            $model->whereHas('userTeam', function($query) {
                $query->where('users_has_teams_status_id', 2);
            });
        }
        if($request->in_team == 2) {
            $model->whereDoesntHave('userTeam', function($query) {
                $query->where('users_has_teams_status_id', 2);
            });
        }
        if($request->need_team == 1) {
            $model->where('need_team', 1);
        }
        if($request->need_team == 2) {
            $model->where('need_team', 0);
        }
        $data = $model->get();
        return response()->json($data);
    }

    function getId(Request $request, $id)
    {
        $model = User::with('publications', 'userTeam.team.users', 'userTeam.team.acceptUsers', 'myTeam.users',
'myTeam.acceptUsers')->find($id);
        if($request->invite) {
            $model['users_has_teams_status_id'] = UserHasTeams::find($request->invite)->users_has_teams_status_id;
        }
        return response()->json($model);
    }

    function getNotification()
    {
        $model = Notifications::where('users_id', Auth::id())->get();
        return response()->json($model);
    }

    function inviteToTeam($id) {
        $model = new UserHasTeams();
        $responseInvite = $model->create([
            'users_id' => Auth::id(),
            'teams_id' => $id,
```

```
            'users_has_teams_status_id' => 1
        ]);

        $modelTeam = Teams::find($id);

        $modelNotification = new Notifications();

        $user = Auth::user();

        $modelNotification->create([
            'title' => 'Запит на участь в команді',
            'text' => 'Гравець <a href="/user/'.$user->id.'/'.$responseInvite->id.'">' . $user->name . " " . $user->surname . "</a>
хоче вступити в вашу команду.",
            'users_id' => $modelTeam->creator_id
        ]);
    }

    function inviteToTeamUser(Request $request) {
        $model = new UserHasTeams();
        $responseInvite = $model->create([
            'users_id' => $request->users_id,
            'teams_id' => $request->teams_id,
            'users_has_teams_status_id' => 1
        ]);

        $modelTeam = Teams::find($request->teams_id);

        $modelNotification = new Notifications();

        $modelNotification->create([
            'title' => 'Запит на участь в команді',
            'text' => 'Адмін команди <a href="/team/'.$request->teams_id.'/'.$responseInvite->id.'">' . $modelTeam->title .
"</a> запрошує Вас в команду.",
            'users_id' => $request->users_id
        ]);
    }

    function updateInviteToTeam(Request $request, $id) {
        UserHasTeams::find($id)->update([
            'users_has_teams_status_id' => $request->status
        ]);
    }
}
```

TeamController.php

Контролер для роботи з командами.

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;

use App\Models\Teams;
use App\Models\TeamPublications;
use App\Models\UserHasTeams;

class TeamController extends Controller
{
```

```
protected $publicStorage = "/uploads/teams/";

public function get(Request $request) {
    $model = Teams::with('users', 'acceptUsers', 'creator');
    if($request->limit) {
        $model->limit($request->limit);
    }
    if($request->title) {
        $model->where('title', 'like', '%' . $request->title . '%');
    }
    if($request->rate_from) {
        $model->where('level', '<=', $request->rate_from);
    }
    if($request->rate_to) {
        $model->where('level', '>=', $request->rate_to);
    }
    if($request->need_users == 1) {
        $model->where('players_needed', 1);
    }
    if($request->need_users == 2) {
        $model->where('players_needed', 0);
    }
    $data = $model->get();
    return response()->json($data);
}

public function getId(Request $request, $id) {
    $model = Teams::with('publications', 'acceptUsers.user')->find($id);
    if($request->invite) {
        $model['users_has_teams_status_id'] = UserHasTeams::find($request->invite)->users_has_teams_status_id;
    }
    return response()->json($model);
}

public function myTeam() {
    $model = Teams::where('creator_id', Auth::id())->with('publications', 'acceptUsers.user')->first();
    return response()->json($model);
}

public function createTeam(Request $request) {
    $model = new Teams();
    $data = $request->all();
    $data['creator_id'] = Auth::id();
    $data['players_needed'] = $request['players_needed'] == 'true' ? 1 : 0;
    if (isset($request['new_photo']) && $request['new_photo'] != "null" && $request['new_photo'] != "undefined") {
        $name =  $this->publicStorage . uniqid() . '.' . $request['new_photo']->getClientOriginalExtension();
        $request['new_photo']->move(public_path() . $this->publicStorage, $name);
        $data['photo'] = $name;
    }
    $response = $model->create($data);

    return response()->json([
        "id" => $response->id
    ]);
}

public function updateTeam(Request $request, $id) {
    $model = Teams::find($id);
    $data = $request->all();
    $data['team_status_id'] = 1;
```

```
        $data['creator_id'] = Auth::id();
        $data['players_needed'] = $request['players_needed'] == 'true' ? 1 : 0;
        if (isset($request['new_photo']) && $request['new_photo'] != "null" && $request['new_photo'] != "undefined") {
            $name =  $this->publicStorage . uniqid() . '.' . $request['new_photo']->getClientOriginalExtension();
            $request['new_photo']->move(public_path() . $this->publicStorage, $name);
            $data['photo'] = $name;
        }
        $model->update($data);

        return response('ok', 200);
    }

    function postPublication(Request $request) {
        $data = $request->all();
        $model = new TeamPublications();
        $response = $model->create($data);
        return response()->json($response);
    }

    function delPublication($id) {
        TeamPublications::find($id)->delete();
    }

    function removeUserFromTeam($id, $team_id) {
        UserHasTeams::where('users_id', $id)->where('teams_id', $team_id)->delete();
    }

    function deleteTeam($id) {
        Teams::find($id)->delete();
    }
}
```

TournamentsController.php

Контролер для роботи з турнірами.

```
<?php

namespace App\Http\Controllers;

use App\Models\Tournaments;
use App\Models\TournamentStatus;
use App\Models\TournamentTypes;
use App\Models\WinningTypes;
use App\Models\ApplicationStatus;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;

class TournamentsController extends Controller
{
    protected $publicStorage = '/uploads/tournaments/';

    public function index(Request $request)
    {
        $model = Tournaments::with('teamApplications.team', 'userApplications.user');
        if($request->limit) {
            $model->limit($request->limit);
        }
        if($request->title) {
            $model->where('title', 'like', '%' . $request->title . '%');
        }
```

```
    if($request->tournament_status_id) {
        $model->where('tournament_status_id', $request->tournament_status_id);
    }
    if($request->winning_types_id) {
        $model->where('winning_types_id', $request->winning_types_id);
    }
    if($request->tournament_types_id) {
        $model->where('tournament_types_id', $request->tournament_types_id);
    }
    if($request->epresentative) {
        $model->where('epresentative', $request->epresentative);
    }
    $data = $model->get();
    return response()->json($data);
}

public function store(Request $request)
{
    $data = $request->all();
    $model = new Tournaments();
    if (isset($request['new_photo']) && $request['new_photo'] != "null" && $request['new_photo'] != "undefined") {
        $name =  $this->publicStorage . uniqid() . '.' . $request['new_photo']->getClientOriginalExtension();
        $request['new_photo']->move(public_path() . $this->publicStorage, $name);
        $data['photo'] = $name;
    }
    $data['users_id'] = Auth::id();
    $model->create($data);
}

public function show($id)
{
    $model = Tournaments::with('status', 'type', 'teamApplications.team', 'userApplications.user')->find($id);
    return response()->json($model);
}

public function update(Request $request, $id)
{
    $data = $request->all();
    $model = Tournaments::find($id);
    if (isset($request['new_photo']) && $request['new_photo'] != "null" && $request['new_photo'] != "undefined") {
        $name =  $this->publicStorage . uniqid() . '.' . $request['new_photo']->getClientOriginalExtension();
        $request['new_photo']->move(public_path() . $this->publicStorage, $name);
        $data['photo'] = $name;
    }
    $model->update($data);
    return response('ok', 200);
}

public function destroy($id)
{
    Tournaments::find($id)->delete();
    return response('ok', 200);
}

function tournamentStatus() {
    $model = TournamentStatus::get();
    return response()->json($model);
}

function tournamentTypes() {
```

```
        $model = TournamentTypes::get();
        return response()->json($model);
    }

    function tournamentWinners() {
        $model = WinningTypes::get();
        return response()->json($model);
    }

    function applicationsStatus() {
        $model = ApplicationStatus::get();
        return response()->json($model);
    }

    function getEpresentatives() {
        $model = Tournaments::select('epresentative')->distinct('epresentative')->get();
        return response()->json($model);
    }
}
```

ApplicationController.php

Контролер для роботи з заявками.

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;

use App\Models\TournamentTeamApplications;
use App\Models\TournamentUserApplications;

class ApplicationController extends Controller
{
    function teamApplication(Request $request, $id) {
        $model = new TournamentTeamApplications();
        $model->create([
            'tournaments_id' => $id,
            'teams_id' => $request->teams_id,
            'application_status_id' => 1
        ]);
    }

    function updateTeamApplication(Request $request, $id) {
        $model = TournamentTeamApplications::find($id);
        $data = $request->all();
        $model->update($data);
    }

    function userApplication(Request $request, $id) {
        $model = new TournamentUserApplications();
        $model->create([
            'tournaments_id' => $id,
            'users_id' => Auth::id(),
            'application_status_id' => 1
        ]);
    }

    function updateUserApplication(Request $request, $id) {
```

```
        $model = TournamentUserApplications::find($id);
        $data = $request->all();
        $model->update($data);
    }
}
```

AuthController.php

Контроллер авторизації користувачів сайту. Містить методи авторизації, реєстрації, завершення сесії та перевірки статусу авторизованого користувача. У випадку якщо сесія закінчилась відбувається переадресація на сторінку авторизації.

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use Illuminate\Support\Facades\Validator;
use Illuminate\Support\Facades\Auth;
use Illuminate\Support\Facades\Hash;

use App\Models\User;

class AuthController extends Controller
{
    public function login(Request $request)
    {
        try {
            $validateUser = Validator::make($request->all(),
            [
                'email' => 'required|email',
                'password' => 'required'
            ]);

            if($validateUser->fails()){
                return response()->json([
                    'status' => false,
                    'message' => 'validation error',
                    'errors' => $validateUser->errors()
                ], 401);
            }

            if(!Auth::attempt(['email' => $request->email, 'password' => $request->password, 'role' => 'user', 'is_ban' => 0])){
                return response()->json([
                    'status' => false,
                    'message' => 'Email & Password does not match with our record.',
                ], 401);
            }

            $user = User::where('email', $request->email)->with('publications', 'myTeam.users', 'userTeam.team.users',
'userTeam.team.acceptUsers')->first();

            return response()->json([
                'status' => true,
                'message' => 'User Logged In Successfully',
```

```
            'user' => $user,
            'token' => $user->createToken("API TOKEN")->plainTextToken
        ], 200);

    } catch (\Throwable $th) {
        return response()->json([
            'status' => false,
            'message' => $th->getMessage()
        ], 500);
    }
}

public function loginAdmin(Request $request)
{
    try {
        $validateUser = Validator::make($request->all(),
        [
            'email' => 'required|email',
            'password' => 'required'
        ]);

        if($validateUser->fails()){
            return response()->json([
                'status' => false,
                'message' => 'validation error',
                'errors' => $validateUser->errors()
            ], 401);
        }

        if(!Auth::attempt(['email' => $request->email, 'password' => $request->password, 'role' => 'admin'])){
            return response()->json([
                'status' => false,
                'message' => 'Email & Password does not match with our record.',
            ], 401);
        }

        $user = User::where('email', $request->email)->first();

        return response()->json([
            'status' => true,
            'message' => 'User Logged In Successfully',
            'user' => $user,
            'token' => $user->createToken("API TOKEN")->plainTextToken
        ], 200);

    } catch (\Throwable $th) {
        return response()->json([
            'status' => false,
            'message' => $th->getMessage()
        ], 500);
    }
}

function register(Request $request) {
    $request->validate([
        'email' => 'required|string|email|unique:users',
        'password' => 'required|string',
        'name' => 'required|string',
        'surname' => 'required|string'
    ]);
```

```
        $user = new User();
        $data = $request->all();
        $data["password"] = Hash::make($request->password);
        $user->create($data);
        return response('ok', 200);
    }

    public function logout()
    {
        Auth::logout();
    }
}
```

app.vue

Головний компонент який підключає всі сторінки сайту.

```
<template>
    <div>
        <router-view v-slot="{ Component, route }">
            <transition name="fade" mode="out-in">
                <component :is="Component" :key="route.path" />
            </transition>
        </router-view>
        <Toast />
    </div>
</template>
<script>
import Toast from '../components/Toast.vue'
export default {
    components: {
        Toast
    }
}
</script>
<style lang="scss" scoped>
.fade-enter-active,
.fade-leave-active {
    transition: opacity 0.3s ease-in-out;
}

.fade-enter-from,
.fade-leave-to {
    opacity: 0;
}
</style>
```

MyTeam.vue

Компонент команди.

```
<template>
    <div class="page">
        <div class="header">
            <div class="grid grid-cols-12 md:gap-12 h-full items-center">
                <div class="md:col-span-8 col-span-12">
                    <div class="title">
                        {{ team.title }}
                    </div>
                    <div class="subtitle">{{ team.nick }}</div>
```

```
            </div>
            <div class="md:col-span-4 col-span-12" v-if="team.creator_id == authUser().id">
              <router-link :to="'/team/'+team.id+'/edit'" class="btn yellow"><img src="/img/person-icon-edit.svg" alt="">
Редагувати</router-link>
            </div>
            <div class="md:col-span-4 col-span-12" v-if="team.users_has_teams_status_id == 1">
              <div class="grid grid-cols-2 gap-5">
                <button @click="acceptInvite(2)" class="btn yellow"><img src="/img/person-check-icon.svg" alt="">
Підтвердити</button>
                <button @click="acceptInvite(3)" class="btn yellow"><img src="/img/account-remove.svg" alt="">
Відхилити</button>
              </div>
            </div>
          </div>
        </div>
        <div class="page-content">
          <p>{{ team.description }}</p>
          <hr>
          <div class="grid grid-cols-12 md:gap-12">
            <div class="md:col-span-3 col-span-12">
              <img class="avatar" v-if="team.photo" :src="team.photo" alt="">
            </div>
            <div class="md:col-span-5 col-span-12">
              <table class="mt-4 md:mt-0">
                <tr>
                  <td>Дата створення:</td>
                  <td class="desc">{{ team.created_at }}</td>
                </tr>
                <tr>
                  <td>Рівень команди:</td>
                  <td class="desc">{{ team.level }}</td>
                </tr>
                <tr>
                  <td>Кількість гравців:</td>
                  <td class="desc">{{ team.accept_users.length }}</td>
                </tr>
                <tr>
                  <td>Статус команди:</td>
                  <td class="desc">{{ team.players_needed == 1 ? 'Повна' : 'Не повна' }}</td>
                </tr>
              </table>
            </div>
            <div class="md:col-span-4 col-span-12">
              <div class="gassle mt-4 md:mt-0" v-if="team.slogan">
                <div class="title">МОТИВУЮЧЕ ГАСЛО</div>
                <p>{{ team.slogan }}</p>
              </div>
            </div>
          </div>
          <hr>
          <div class="grid grid-cols-12 md:gap-10 teams mb-10">
            <div class="md:col-span-3 col-span-6 team-item" v-for="(item, index) in team.accept_users" :key="index">
              <UserItem @removeTeamUser="removeTeamUser" :item="item.user" />
            </div>
          </div>
          <div class="grid grid-cols-12 md:gap-5 publication" v-if="team.creator_id == authUser().id">
            <div class="md:col-span-10 col-span-12">
              <input type="text" v-model="publication.title" placeholder="Тема">
              <textarea placeholder="Опис" v-model="publication.description"></textarea>
            </div>
```

```
            <div class="md:col-span-2 col-span-12">
                <button @click="addPost()">Публікація</button>
            </div>
        </div>
        <PublicationItem
            v-for="(publication, index) in team.publications"
            :publication="publication"
            :key="index"
            @delPost="delPost"
        />
    </div>
    <Pagination />
  </div>
</template>

<script>

import { mapGetters } from 'vuex';
import Pagination from '../components/Pagination.vue'
import UserItem from '../components/UserItem.vue'
import PublicationItem from '../components/PublicationItem.vue'
export default {
  components: {
    Pagination,
    UserItem,
    PublicationItem
  },
  data() {
    return {
      team: {
        title: "",
        nick: "",
        description: "",
        photo: "",
        slogan: "",
        level: "",
        created_at: "",
        accept_users: [],
        publications: [],
        creator_id: null
      },
      publication: {
        title: "",
        description: ""
      }
    }
  },
  mounted() {
    this.getData();
  },
  methods: {
    ...mapGetters(["authUser"]),
    getData() {
      let params = {};
      if(this.$route.params.invite) {
        params.invite = this.$route.params.invite;
      }
      if(this.$route.name == 'my-team') {
        axios.get('/api/my-team', {
            params
```

```
            })
            .then((response) => {
                this.team = Object.assign(this.team, response.data);
            })
        }
        if(this.$route.name == 'team') {
            axios.get('/api/team/'+this.$route.params.id, {
                params
            })
            .then((response) => {
                this.team = Object.assign(this.team, response.data);
            })
        }
    },
    addPost() {
        this.publication.teams_id = this.authUser().my_team.id;
        axios.post('/api/team/publication', this.publication)
        .then((response) => {
            this.team.publications.push(response.data)
            this.$notify({
                title: "Успіх",
                text: "Інформацію збережено!",
                type: "info",
                group: "foo",
            }, 4000)
            this.publication = {
                title: "",
                description: ""
            }
        })
    },
    delPost(data) {
        let index = this.team.publications.indexOf(data);
        axios.delete('/api/team/publication/'+data.id)
        .then(() => {
            this.team.publications.splice(index, 1);
            this.$notify({
                title: "Успіх",
                text: "Публікацію видалено!",
                type: "info",
                group: "foo",
            }, 4000)
        })
    },
    removeTeamUser(data) {
        let index = this.team.accept_users.indexOf(data);
        axios.delete('/api/team/remove-user/'+data.id+'/'+this.team.id)
        .then(() => {
            this.team.accept_users.splice(index, 1);
            this.$notify({
                title: "Успіх",
                text: "Учасника видалено!",
                type: "info",
                group: "foo",
            }, 4000)
        })
    },
    acceptInvite(status) {
        axios.post('/api/user/update-invite-to-team/'+this.$route.params.invite, {
            status
```

```
            })
            .then(() => {
               this.getData();
            })
         }
      }
   }
</script>


<style lang="scss" scoped>
   .header {
      background: #3A4FB9;
      padding: 45px 97px;
      position: relative;
      margin-bottom: 50px;
      .title {
         color: #EFEEE0;
         font-weight: 700;
         font-size: 35px;
         line-height: 120%;
      }
      .subtitle {
         color: rgba(255, 255, 255, 0.5);
         font-weight: 400;
         font-size: 14px;
      }
      .btn {
         padding: 7px 20px;
         border-radius: 25px;
         margin-top: 15px;
         font-weight: 400;
         min-height: 44px;
         width: 100%;
         font-size: 13px;
      }
   }
   table {
      width: 100%;
      td {
         padding: 5px 0;
         font-weight: 400;
         font-size: 14px;
      }.desc {
         font-weight: 300;
         font-size: 12px;
      }
   }
   .avatar {
      height: 100%;
      object-fit: cover;
      border-radius: 10px;
   }
   @media screen and (max-width: 600px) {
      .header {
         padding: 25px 30px;
      }
   }
</style>
```

Profile.vue

Компонент профіля користувача.

```
<template>
  <div class="page">
    <div v-if="chatModal"
      class="overflow-x-hidden overflow-y-auto fixed inset-0 z-50 outline-none focus:outline-none justify-center items-center flex">
      <div class="relative p-4 w-full max-w-2xl h-full md:h-auto modal-dialog">
        <!--content-->
        <div class="relative bg-white rounded-lg shadow dark:bg-gray-700">
          <!--header-->
          <div class="flex justify-between items-start p-4 rounded-t border-b dark:border-gray-600">
            <button type="button" @click="toggleModal()"
              class="text-gray-400 bg-transparent hover:bg-gray-200 hover:text-gray-900 rounded-lg text-sm p-1.5 ml-auto inline-flex items-center dark:hover:bg-gray-600 dark:hover:text-white"
              data-modal-toggle="defaultModal">
              <svg aria-hidden="true" class="w-5 h-5" fill="currentColor" viewBox="0 0 20 20"
                xmlns="http://www.w3.org/2000/svg">
                <path fill-rule="evenodd"
                  d="M4.293 4.293a1 1 0 011.414 0L10 8.586l4.293-4.293a1 1 0 111.414 1.414L11.414 10l4.293 4.293a1 1 0 01-1.414 1.414L10 11.414l-4.293 4.293a1 1 0 01-1.414-1.414L8.586 10 4.293 5.707a1 1 0 010-1.414z"
                  clip-rule="evenodd"></path>
              </svg>
              <span class="sr-only">Close modal</span>
            </button>
          </div>
          <!--body-->
          <div class="p-6 space-y-6">
            <textarea id="message" rows="4"
              class="block p-2.5 w-full text-sm text-gray-900 bg-gray-50 rounded-lg border border-gray-300 focus:ring-blue-500 focus:border-blue-500 dark:bg-gray-700 dark:border-gray-600 dark:placeholder-gray-400 dark:text-white dark:focus:ring-blue-500 dark:focus:border-blue-500"
              placeholder="Введіть повідомленні..." v-model="message"></textarea>

          </div>
          <!--footer-->
          <div
            class="flex items-center p-6 space-x-2 rounded-b border-t border-gray-200 dark:border-gray-600">
            <button @click="sendMessage()"
              class="text-white bg-blue-700 hover:bg-blue-800 focus:ring-4 focus:outline-none focus:ring-blue-300 font-medium rounded-lg text-sm px-5 py-2.5 text-center dark:bg-blue-600 dark:hover:bg-blue-700 dark:focus:ring-blue-800">
              Надіслати
            </button>
          </div>
        </div>
      </div>
    </div>
    <div v-if="chatModal" class="opacity-25 fixed inset-0 z-40 bg-black"></div>

    <div class="header">
      <div class="grid grid-cols-12 gap-12 h-full items-center">
        <div class="col-span-8">
          <div class="grid grid-cols-12 md:gap-12">
            <div class="md:col-span-7 col-span-12">
              <div class="name">
                {{ user.name }} {{ user.surname }}
              </div>
```

```
                <table>
                  <tr>
                    <td>Дата реєстрації:</td>
                    <td class="desc">{{ user.created_at }}</td>
                  </tr>
                  <tr>
                    <td>Рейтинг гравця:</td>
                    <td class="desc">ESL DH ATLANTA 2023</td>
                  </tr>
                  <tr>
                    <td>Нік в грі:</td>
                    <td class="desc">{{ user.nick }}</td>
                  </tr>
                </table>
              </div>
              <div class="md:col-span-5 col-span-12">
                <div class="gassle" v-if="user.slogan">
                  <div class="title">МОТИВУЮЧЕ ГАСЛО</div>
                  <p>{{ user.slogan }}</p>
                </div>
                <router-link class="btn yellow" to="/profile/edit" v-if="$route.name == 'profile'">
                  <img src="/img/person-icon-edit.svg">Редагувати
                </router-link>
                <button v-on:click="toggleModal()" type="button" class="btn yellow"
                  v-if="authUser().id != user.id">
                  <img src="/img/message-icon.svg">Написати
                </button>
                <div class="grid grid-cols-2 gap-5" v-if="user.users_has_teams_status_id == 1">
                  <button class="btn yellow" @click="acceptInvite(2)">
                    <img src="/img/person-icon-plus.svg">Прийняти
                  </button>
                  <button class="btn yellow" @click="acceptInvite(3)">
                    <img src="/img/account-remove.svg">Відхилити
                  </button>
                </div>
              </div>
            </div>
          </div>
          <div class="col-span-4">
            <img v-if="user.photo" class="avatar" :src="user.photo" alt="">
          </div>
        </div>
      </div>
      <div class="page-content">
        <hr v-if="user.user_team">
        <div class="grid grid-cols-12 md:gap-8" v-if="user.user_team">
          <div class="md:col-span-9 col-span-12">
            <p class="mb-3 md:mb-0">{{ user.user_team.team.description }}</p>
          </div>
          <div class="md:col-span-3 col-span-12 teams">
            <TeamItem :item="user.user_team.team" />
          </div>
        </div>
        <hr v-if="user.my_team">
        <div class="grid grid-cols-12 md:gap-8" v-if="user.my_team">
          <div class="md:col-span-9 col-span-12">
            <p class="mb-3 md:mb-0">{{ user.my_team.description }}</p>
          </div>
          <div class="md:col-span-3 col-span-12 teams">
            <TeamItem :item="user.my_team" />
```

```
            </div>
        </div>
        <hr>
        <div class="grid grid-cols-12 md:gap-5 publication" v-if="$route.name == 'profile'">
            <div class="md:col-span-10 col-span-12">
                <input type="text" v-model="publication.title" placeholder="Тема">
                <textarea v-model="publication.description" placeholder="Опис"></textarea>
            </div>
            <div class="md:col-span-2 col-span-12">
                <button @click="addPost()">Публікація</button>
            </div>
        </div>
        <PublicationItem v-for="(publication, index) in user.publications" :publication="publication" :key="index"
            @delPost="delPost" />
    </div>
    <Pagination />
  </div>
</template>

<script>
import { mapGetters } from 'vuex';
import Pagination from '../components/Pagination.vue'
import TeamItem from '../components/TeamItem.vue'
import PublicationItem from '../components/PublicationItem.vue'
export default {
  components: {
    Pagination,
    TeamItem,
    PublicationItem
  },
  data() {
    return {
      message: "",
      user: {
        publications: [],
        my_team: {
          accept_users: []
        }
      },
      publication: {
        title: "",
        description: ""
      },
      chatModal: false
    }
  },
  mounted() {
    this.getData();
  },
  methods: {
    ...mapGetters(["authUser"]),
    toggleModal() {
      this.chatModal = !this.chatModal;
    },
    getData() {
      if (this.$route.name == 'profile') {
        axios.get('/api/profile')
          .then((response) => {
            this.user = Object.assign(this.user, response.data);
          })
```

```
            }
            if (this.$route.name == 'user') {
                let params = {};
                if (this.$route.params.invite) {
                    params.invite = this.$route.params.invite;
                }
                axios.get('/api/user/' + this.$route.params.id, {
                    params
                })
                    .then((response) => {
                        this.user = Object.assign(this.user, response.data);
                    })
            }
        },
        addPost() {
            axios.post('/api/user/publication', this.publication)
                .then((response) => {
                    this.user.publications.push(response.data)
                    this.$notify({
                        title: "Успіх",
                        text: "Інформацію збережено!",
                        type: "info",
                        group: "foo",
                    }, 4000)
                    this.publication = {
                        title: "",
                        description: ""
                    }
                })
        },
        delPost(data) {
            let index = this.user.publications.indexOf(data);
            axios.delete('/api/user/publication/' + data.id)
                .then(() => {
                    this.user.publications.splice(index, 1);
                })
        },
        acceptInvite(status) {
            axios.post('/api/user/update-invite-to-team/' + this.$route.params.invite, {
                status
            })
                .then(() => {
                    this.getData();
                })
        },
        sendMessage() {
            axios.post('/api/chats', {
                users_id: this.user.id,
                message: this.message
            })
                .then(() => {
                    this.chatModal = false;
                    this.message = "";
                })
        },
    }
}
</script>
```

```
<style lang="scss" scoped>
.header {
   background: #3A4FB9;
   padding: 0 0 0 97px;
   position: relative;

   .name {
      color: #EFEEE0;
      font-weight: 700;
      font-size: 25px;
      line-height: 120%;
      margin-bottom: 25px;
   }

   table {
      width: 100%;

      td {
         padding: 5px 0;
         font-weight: 400;
         font-size: 14px;
         line-height: 17px;
      }

      .desc {
         font-weight: 300;
         font-size: 12px;
         line-height: 15px;
      }
   }

   .btn {
      padding: 7px 20px;
      border-radius: 25px;
      margin-top: 15px;
      font-weight: 400;
      font-size: 10px;
      min-height: 25px;
      display: inline-flex;
   }
}

@media screen and (max-width: 600px) {
   .header {
      padding: 25px 30px;
   }
   .gassle {
      margin-top: 15px;
   }
}
</style>
```

GameItem.vue

Компонент сторінки з турніром.

```
<template>
   <div class="page">
      <div class="header">
         <div class="title">
            {{tournaments.title}}
```

```
        </div>
        <div class="flex justify-between items-center">
          <button v-if="tournaments.tournament_status_id == 1 && tournaments.tournament_types_id == 1 &&
authUser().my_team" class="btn yellow">
            <span v-if="checkTeamApplication && checkTeamApplication.application_status_id == 1">
              <img src="/img/account-clock.svg">
              Заявку подано
            </span>
            <span v-else-if="checkTeamApplication && checkTeamApplication.application_status_id == 2">
              <img src="/img/person-check-icon.svg">
              Заявку прийнято
            </span>
            <span v-else-if="checkTeamApplication && checkTeamApplication.application_status_id == 3">
              <img src="/img/account-remove.svg">
              Заявку відхилено
            </span>
            <span v-else @click="sendTeamApp()">
              <img src="/img/person-icon-plus.svg">
              Подати зяавку
            </span>
          </button>
          <button v-if="tournaments.tournament_status_id == 1 && tournaments.tournament_types_id == 2" class="btn
yellow">
            <span v-if="checkUserApplication && checkUserApplication.application_status_id == 1">
              <img src="/img/account-clock.svg">
              Заявку подано
            </span>
            <span v-else-if="checkUserApplication && checkUserApplication.application_status_id == 2">
              <img src="/img/person-check-icon.svg">
              Заявку прийнято
            </span>
            <span v-else-if="checkUserApplication && checkUserApplication.application_status_id == 3">
              <img src="/img/account-remove.svg">
              Заявку відхилено
            </span>
            <span v-else @click="sendUserApp()">
              <img src="/img/person-icon-plus.svg">
              Подати зяавку
            </span>
          </button>
          <span class="flex items-center"><img src="/img/fire-icon.svg" class="mr-1"> {{
tournaments.team_applications.length + tournaments.user_applications.length }} Поданих заяв</span>
        </div>
      </div>
      <div class="page-content">
        <hr>
        <div class="grid grid-cols-12 md:gap-12">
          <div class="md:col-span-2 col-span-12">
            <img class="avatar" :src="tournaments.photo" alt="">
          </div>
          <div class="md:col-span-5 col-span-12">
            <table class="mt-4 md:mt-0">
              <tr>
                <td>Дата проведення:</td>
                <td class="desc">{{tournaments.created_at}}</td>
              </tr>
              <tr>
                <td>Представник:</td>
                <td class="desc">{{tournaments.epresentative}}</td>
              </tr>
```

```
                    <tr>
                        <td>Виграш:</td>
                        <td class="desc">{{tournaments.winning_types_id == 1 ? '$' : ''}} {{tournaments.winning}}</td>
                    </tr>
                    <tr>
                        <td>Статус:</td>
                        <td class="desc">{{tournaments.status.title}}</td>
                    </tr>
                </table>
            </div>
            <div class="md:col-span-5 col-span-12 mt-4 md:mt-0">
                <div class="gassle">
                    <div class="title">МОТИВУЮЧЕ ГАСЛО</div>
                    <p>{{tournaments.slogan}}</p>
                </div>
            </div>
        </div>
        <hr>
        <p v-html="tournaments.description"></p>
        <div v-if="tournaments.result">
            <hr>
            <h4 class="text-2xl font-bold dark:text-white mb-3">Результати</h4>
            <p v-html="tournaments.result"></p>
        </div>
    </div>
  </div>
</template>

<script>

import Pagination from '../components/Pagination.vue'
import TeamItem from '../components/TeamItem.vue'
import PublicationItem from '../components/PublicationItem.vue'

import { mapGetters } from 'vuex';

export default {
  components: {
    Pagination,
    TeamItem,
    PublicationItem
  },
  data() {
    return {
      tournaments: {
        title: "",
        status: {
          title: ""
        },
        team_applications: [],
        user_applications: []
      }
    }
  },
  mounted() {
    this.getData();
  },
  computed: {
    checkTeamApplication() {
      return this.tournaments.team_applications.find(item => item.teams_id == this.authUser().my_team.id)
```

```
        },
        checkUserApplication() {
            return this.tournaments.user_applications.find(item => item.users_id == this.authUser().id)
        }
    },
    methods: {
        ...mapGetters(["authUser"]),
        getData() {
            axios.get('/api/tournaments/'+this.$route.params.id)
                .then((response) => {
                    this.tournaments = Object.assign(this.tournaments, response.data);
                })
        },
        sendTeamApp() {
            axios.post('/api/team-application/'+this.$route.params.id, {
                teams_id: this.authUser().my_team.id
            })
                .then(() => {
                    this.getData();
                })
        },
        sendUserApp() {
            axios.post('/api/user-application/'+this.$route.params.id)
                .then(() => {
                    this.getData();
                })
        }
    }
}
</script>

<style lang="scss" scoped>
    .header {
        background: #3A4FB9;
        padding: 45px 97px;
        position: relative;
        .title {
            color: #EFEEE0;
            font-weight: 700;
            font-size: 35px;
            line-height: 120%;
            margin-bottom: 30px;
        }
        .btn {
            padding: 7px 35px;
            border-radius: 25px;
            margin-top: 15px;
            font-weight: 400;
            min-height: 44px;
            font-size: 13px;
        }
    }
    .gassle {
        background: #4F524F;
        border-radius: 16px;
        padding: 22px;
        color: #EFEEE0;
        font-weight: 400;
        font-size: 10px;
        line-height: 120%;
```

```
        .title  {
            font-weight: 700;
            font-size: 12px;
            margin-bottom: 8px;
        }
        p {
            opacity: 0.75;
        }
    }
    table {
        width: 100%;
        td {
            padding: 5px 0;
            font-weight: 400;
            font-size: 14px;
            line-height: 17px;
        }.desc {
            font-weight: 300;
            font-size: 12px;
            line-height: 15px;
        }
    }
    .avatar {
        border-radius: 12px;
        object-fit: cover;
    }
    @media screen and (max-width: 600px) {
        .header {
            padding: 25px 30px;
        }
    }
</style>
```