

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**  
**ДЕРЖАВНИЙ ЗАКЛАД**  
**„ЛУГАНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ**  
**ІМЕНІ ТАРАСА ШЕВЧЕНКА”**

Навчально-науковий інститут математики та інформаційних технологій

Кафедра математики та інформатики

**Лановенко Валерій**

**ДОСЛІДЖЕННЯ ВЕБТЕХНОЛОГІЙ ПОБУДОВИ**  
**ІНФОРМАЦІЙНОГО ВЕБЗАСТОСУНКУ З ПОШУКОВОЮ**  
**ФУНКЦІОНАЛЬНІСТЮ**

**кваліфікаційна робота**  
**здобувача вищої освіти другого (магістерського) рівня**  
**освітньої програми «Комп’ютерні науки»**  
**за спеціальністю 122 „ Комп’ютерні науки ”**

Особистий підпис \_\_\_\_\_

Науковий керівник \_\_\_\_\_

Завідувач кафедри \_\_\_\_\_

## АНОТАЦІЯ

**Лановенко В.**

**Тема:** Дослідження вебтехнологій розробки пошукових системи.

**Спеціальність:** 122 „Комп’ютерні науки”.

**Установа:** ДЗ ЛНУ імені Тараса Шевченка, 2026р.

**Кваліфікаційна робота містить:** 63 стор., 18 табл., 25 рис., 34 джерела, 2 додатки.

**Об’єкт дослідження** – вебтехнології, процес програмної реалізації пошукової інформаційної системи.

**Предмет дослідження** – технології створення пошукової інформаційної системи.

**Мета роботи** – аналіз програмних вебтехнологій та розробка пошукової інформаційної системи.

**Результати роботи.** Досліджено веб технології розробки пошукової інформаційної системи. Досліджено функціональні можливості існуючих сучасних аналогів, виявлено їх переваги та недоліки. Обрано вебтехнології та розроблено пошукову інформаційну систему.

**Ключові слова:** PHP, MySQL, HTML, CSS, JS, JQuery

## ABSTRACT

**Lanovenko V.**

**Theme:** Research into web technologies for search engine development.

**Specialty:** 122 "Computer Science"

**Institution:** Luhansk Taras Shevchenko National University, 2023

**Qualification work contains:** 63 pages, 18 tables, 25 figures, 34 sources, 2 appendices.

**Object of research** is web technologies, process of software implementation of search information system.

**Subject of research** - technologies of creation of search information system.

**Purpose of the study** is the analysis of software web technologies and development of search information system.

**Results of research.** Web technologies of development of search information system were investigated. Functional capabilities of existing modern analogues were investigated, their advantages and disadvantages were identified. Web technologies were selected and a search information system.

**Keywords:** PHP, MySQL, HTML, CSS, JS, JQuery

## ЗМІСТ

Стор.

|  |           |
|--|-----------|
| <b>ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, СКОРОЧЕНЬ</b>                                      | <b>5</b>  |
| ВСТУП .....  | 6         |
| РОЗДІЛ 1 .....   | 8         |
| АНАЛІЗ СУЧАСНОГО СТАНУ ДОСЛІДЖУВАНОЇ ПРОБЛЕМИ.....   | 8         |
| 1.1 Аналіз предметної області .....  | 9         |
| 1.2 Аналіз систем пошуку виконавців ІТ-проектів .....                                      | 17        |
| 1.3 Висновки до розділу 1 .....  | 19        |
| РОЗДІЛ 2 .....   | 21        |
| АНАЛІЗ СУЧАСНИХ ТЕХНОЛОГІЙ ДЛЯ РЕАЛІЗАЦІЇ ПОШУКОВОЇ<br>ІНФОРМАЦІЙНОЇ СИСТЕМИ .....         | 21        |
| 2.1 Аналіз програмних веб-технологій для розробки пошукової<br>інформаційної системи ..... | 21        |
| 2.2 Аналіз засобів реалізації баз даних у вебтехнологіях.....                              | 23        |
| 2.3 Мова розмітки веб-документів HTML.....   | 24        |
| 2.4 Каскадні таблиці стилів (CSS).....   | 25        |
| 2.5 Динамічна мова JavaScript .....  | 27        |
| 2.6 Висновки до розділу 2 .....  | 28        |
| <b>РОЗДІЛ 3 .....</b>  | <b>30</b> |
| <b>АРХІТЕКТУРА ТА ОПИС РОЗРОБКИ.....</b>   | <b>30</b> |
| 3.1 Вимоги до проєкту веборієнтованої ІС .....   | 30        |
| 3.2 ER-діаграма.....   | 31        |
| 3.3 Структура таблиць бази даних .....   | 33        |
| 3.4 Опис користувацької частини додатку.....   | 39        |
| 3.5 Висновки до розділу 3 .....  | 49        |
| ВИСНОВКИ.....  | 50        |
| ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ.....   | 52        |
| <b>ДОДАТКИ.....</b>  | <b>55</b> |
| Додаток А. Головна сторінка роботодавця .....  | 55        |
| Додаток Б. Код employee_home.php .....   | 55        |

**ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, СКОРОЧЕНЬ**

|      |   |   |
|------|---|---|
| CGI  | - | Common Gateway Interface;                       |
| CMS  | - | система управління вмістом;                     |
| CSS  | - | Cascade Style Sheets;                           |
| DDOS | - | Distributed Denial-of- service attack;          |
| DOM  | - | Document Object Model;                          |
| DTD  | - | Document Type Definition;                       |
| EDGE | - | Enhanced Data rates for GSM Evolution;          |
| ERM  | - | модель зв'язку сутностей;                       |
| FDA  | - | Food and Drug Administration;                   |
| FSM  | - | Finite-state machine;                           |
| GPRS | - | General Packet Radio Service;                   |
| HTML | - | Hyper Text Markup Language;                     |
| HTTP | - | HyperText Transfer Protocol;                    |
| ISO  | - | International Organization for Standardization; |
| JS   | - | JavaScript;                                     |
| PHP  | - | Hypertext Preprocessor;                         |
| SEO  | - | Search Engine Optimization;                     |
| SGML | - | Standard Generalized Markup Language;           |
| SQL  | - | Structured Query Language;                      |
| UML  | - | Unified Modeling Language;                      |
| IC   | - | інформаційна система;                           |
| IT   | - | інформаційні технології;                        |
| OC   | - | операційна система;                             |
| ПЗ   | - | програмне забезпечення;                         |
| ПК   | - | персональний комп'ютер.                         |

## ВСТУП

Воєнні події в Україні призвели до суттєвих обмежень у багатьох сферах життєдіяльності, кардинально трансформували світовий ринок праці. Значна кількість людей втратила постійну роботу, однак, прагнучи адаптуватися до нових умов, вони все активніше переходять до дистанційних форм зайнятості, де інтернет став невід'ємною складовою професійного життя. Для більшості таким виходом став фріланс.

Сучасний технологічний простір динамічно розвивається: щодня зростає кількість запитів від підприємців і власників бізнесу на виконання різноманітних завдань, зокрема створення логотипів, сайтів, дизайну чи програмного забезпечення. Водночас пошук надійного й компетентного виконавця стає дедалі складнішим, особливо з огляду на проблему довіри між замовниками та виконавцями. Хоча нині існує чимало онлайн-сервісів, що сприяють встановленню вигідних партнерських відносин, в Україні цей напрям залишається недостатньо розвиненим, а значна частина іноземних платформ є заблокованою. Тому створення вітчизняного сучасного порталу для фрілансерів є надзвичайно актуальним завданням.

Метою роботи є аналіз вебтехнологій і розробка пошукової інформаційної системи — порталу для фрілансерів.

Для досягнення поставленої мети необхідно розв'язати такі завдання:

- дослідити ринок подібних сервісів;
- проаналізувати вимоги до системи;
- спроектувати архітектуру програмного продукту;
- реалізувати програмне забезпечення.

Об'єктом дослідження є сервіси, що забезпечують самостійне управління робочим процесом фрілансера та взаємодію із замовниками.

Предметом дослідження є вебсервіси-посередники між клієнтами та фрілансерами, які працюють у науковій, технічній і творчій сферах.

Методи дослідження включають:

- теоретичні: аналіз науково-технічних джерел та інтернет-ресурсів;
- емпіричні: дослідження підходів до оптимізації сучасних вебтехнологій у процесі створення пошукових інформаційних систем.

Наукова новизна полягає у розробці пошукової інформаційної системи «FreeLance».

Практичне значення роботи визначається створенням програмного продукту «FreeLance» із використанням PHP, MySQL, HTML, CSS, JavaScript та JQuery. Розроблений вебсервіс стане в нагоді для комунікації між клієнтами та виконавцями. Результати дослідження також можуть бути застосовані під час вивчення принципів побудови багаторівневих вебсистем і серверних алгоритмів обробки даних.

Робота складається зі вступу, трьох розділів, висновків, списку використаних джерел і додатків. Загальний обсяг становить 70 сторінок, кількість використаних джерел — 34.

У першому розділі подано аналіз сучасних вебдодатків для фрілансерів та визначено основні вимоги до створення нового вебзастосування «FreeLance».

У другому розділі розглянуто сучасні технології веброзробки, охарактеризовано популярні інструменти та виявлено їх недоліки. Обґрунтовано вибір засобів для реалізації магістерського проєкту.

У третьому розділі описано процес програмної реалізації вебзастосування «FreeLance» та наведено основні етапи його створення.

Додатки містять зображення головної сторінки системи та фрагменти програмного коду файлу *employee\_home.php*.

## РОЗДІЛ 1

### АНАЛІЗ СУЧАСНОГО СТАНУ ДОСЛІДЖУВАНОЇ ПРОБЛЕМИ

У сучасному світі дедалі більшої популярності набуває позаштатна форма зайнятості, відома як фріланс. Для багатьох вона стає не лише альтернативою традиційній роботі, а й першим кроком до створення власної справи. Серед основних переваг фрілансу — можливість працювати з будь-якого зручного місця, самостійно планувати робочий час і не залежати від жорсткого контролю керівництва.

В Україні фріланс здебільшого асоціюється з ІТ-сферою та охоплює такі напрями, як проєктування й адміністрування вебсайтів, наповнення їх контентом, веб- та графічний дизайн. Цей формат діяльності добре підходить і для початківців, які лише роблять перші кроки в галузі інформаційних технологій.

Типова модель роботи фрілансера здається простою: вибір онлайн-платформи, подання заявок на проєкти, виконання завдань і отримання оплати. Проте на практиці процес значно складніший — складно знайти вигідне замовлення та вирізнитися серед великої кількості конкурентів.

Фріланс-біржі — це онлайн-сервіси, що забезпечують можливість дистанційної співпраці між замовниками та виконавцями, дозволяють розміщувати власні послуги, виконувати різнопланові завдання та знаходити потенційних клієнтів для реалізації проєктів у різних сферах.

Попри глобальне поширення фрілансу, в Україні інтерес до віддаленої роботи зазнавав коливань. Після послаблення карантинних обмежень у травні спостерігалось тимчасове зниження пошукової активності за запитами «фріланс» і «віддалена робота», яке згодом змінилося різким зростанням. Згідно зі статистикою платформи Freelancehunt, у 2006 році в Україні було зареєстровано близько 1,2 тисячі фрілансерів, тоді як у 2021 році кількість користувачів уже сягнула одного мільйона. Після початку повномасштабної



війни чисельність фрілансерів не зменшилася — навпаки, все більше людей почали обирати цей формат роботи та реєструватися на спеціалізованих платформах.

Це зумовлено тим, що на подібних платформах доволі складно отримувати високий дохід, а ключову роль у співпраці відіграє рівень довіри між замовником і виконавцем. Попри те, що портал виконує функцію посередника, він не завжди спроможний повністю убезпечити користувачів від шахрайства. В Україні також поширена думка, що стабільна робота в перевіреному колективі дозволяє швидше набути професійного досвіду, хоча саме фріланс дає змогу значно легше сформувати якісне портфоліо та напрацювати практичні навички.

### **1.1 Аналіз предметної області**

Портали є різновидом вебресурсів із розширеним функціоналом і зручною системою навігації. Такі сайти зазвичай орієнтовані на конкретну тематику та об'єднують спільноти користувачів зі схожими інтересами.

Портал для фрілансерів — це вебсайт, що забезпечує можливість дистанційної співпраці для розробників, дизайнерів та інших спеціалістів, виконуючи роль посередника між виконавцем і клієнтом. Після проходження реєстрації користувач може розміщувати власні завдання або пропонувати послуги, які зберігаються в базі даних системи. Потенційний замовник, переглядаючи детальну інформацію про проєкт, має змогу ознайомитися з пропозиціями фрілансера та відомостями про нього, після чого зв'язатися з виконавцем безпосередньо через портал і погодити умови співпраці.

Розробка подібних сайтів зазвичай здійснюється із застосуванням систем керування контентом. Великі портали використовують власні спеціалізовані рішення для управління та моніторингу процесів, тоді як для середніх і невеликих проєктів часто застосовуються безкоштовні платформи, зокрема OpenCart, Drupal або WordPress.

Система управління сайтом може бути встановлена на хостинг

самостійно, розроблена як приватне програмне забезпечення або використовуватися на основі тимчасових ліцензій.

Виділяють два основні типи порталів для пошуку віддаленої роботи:

1. сайти з вакансіями, де роботодавці публікують оголошення про набір співробітників для дистанційної зайнятості;
2. фріланс-біржі, на яких спеціалісти працюють на себе, самостійно обираючи проєкти та регулюючи власний робочий час.

Сьогодні існує чимало фріланс-платформ, які настільки міцно закріпилися на ринку, що автоматично з'являються серед перших результатів пошуку за запитом, пов'язаним з фрілансом. До українських сервісів такого типу належать, зокрема: [Freelance.ua](#), [Weblancer.net](#), [Free-lance.ua](#), [Kabanchik.ua](#).

Водночас фахівцям не варто обмежуватися лише вітчизняними ресурсами, адже ефективний пошук замовлень передбачає використання й міжнародних платформ.

### *Freelancehunt*

Щодня на біржі з'являється понад тисячу активних завдань від замовників.

Перша фріланс-біржа в Україні була створена у 2003 році — нею став сервіс [Weblancer.net](#). Станом на сьогодні в Україні функціонує низка популярних платформ для віддаленої роботи [9].

[Freelancehunt](#) є однією з найбільших і найвідоміших фріланс-бірж в Україні. Платформа пропонує проєкти у різноманітних напрямках, зокрема:

- дизайн;
- розробка програмного забезпечення;
- системне адміністрування;
- бекенд-розробка;
- управління проєктами;
- маркетинг;

- переклад;
- створення та наповнення контенту;
- юридичні послуги;
- інші галузі.

Щодня на платформі публікується понад тисячу активних завдань від замовників, що свідчить про її високу популярність і затребуваність серед користувачів.

На рисунку 1.1 представлено стартову сторінку Freelancehunt.

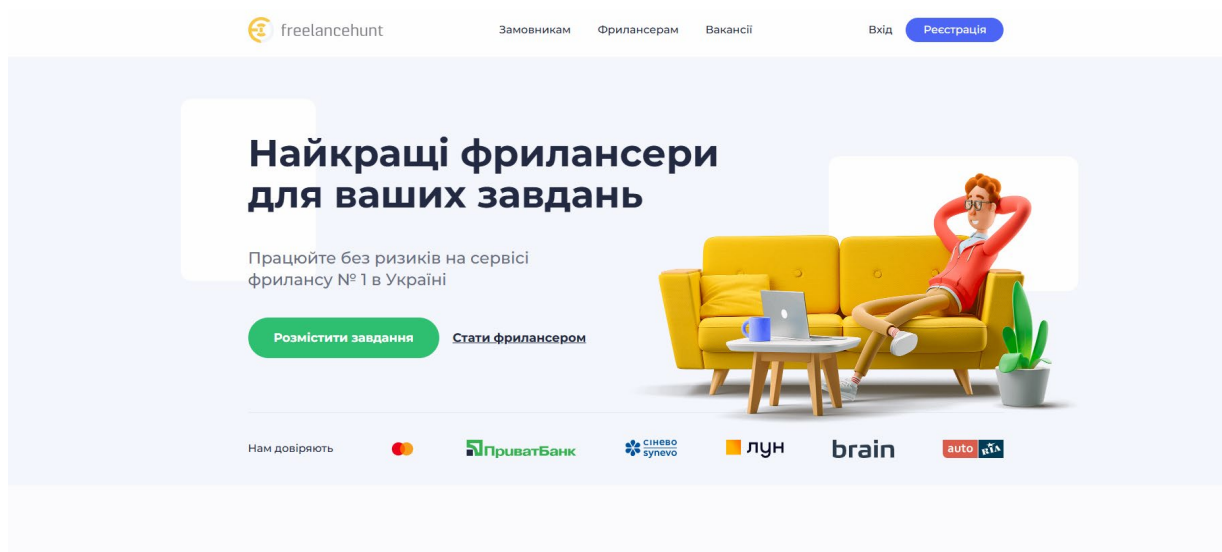


Рисунок 1.1. Стартова сторінка Freelancehunt

Механізм роботи фріланс-біржі є таким:

1. замовник публікує проєкт із детальним описом завдання;
2. фрілансери подають заявки, зазначаючи вартість, строки виконання та власні коментарі;
3. замовник обирає виконавця;
4. фрілансер виконує поставлене завдання;
5. замовник приймає результат роботи та здійснює оплату;
6. після завершення проєкту сторони залишають взаємні відгуки.

Фрілансери мають можливість подавати необмежену кількість заявок на різні проєкти.

Платформа Freelancehunt утримує комісію за проведення угод, яка становить 9 %. Оплата комісії може здійснюватися як замовником, так і виконавцем, або розподілятися між ними порівну — по 5 % з кожної сторони.

Згідно зі статистичними даними сервісу Freelancehunt [1], 83,3 % користувачів працюють удома, 7,2 % — в офісах, 5,5 % поєднують домашній і офісний формати роботи, 2,1 % виконують завдання в кафе, а 0,6 % — у коворкінгах.

Окремою функцією платформи є розділ «Конкурси», у якому замовники можуть оголосити змагання з неймінгу, створення логотипів чи ілюстрацій. Після заповнення брифу та проходження модерації фрілансери надсилають власні варіанти робіт, а замовник обирає переможця.

У 2021 році на платформі було запроваджено розділ «Вакансії», призначений для пошуку штатних працівників, а не для разових фріланс-проектів.

4 листопада 2021 року в Україні вперше відзначили День фрілансера з ініціативи сервісу Freelancehunt, який був зареєстрований саме цього дня.

### *Freelance.ua*

Фріланс-біржа Freelance.ua розпочала свою діяльність в Україні у 2006 році [14].

На рисунку 1.2 зображено головну сторінку вебсайту.

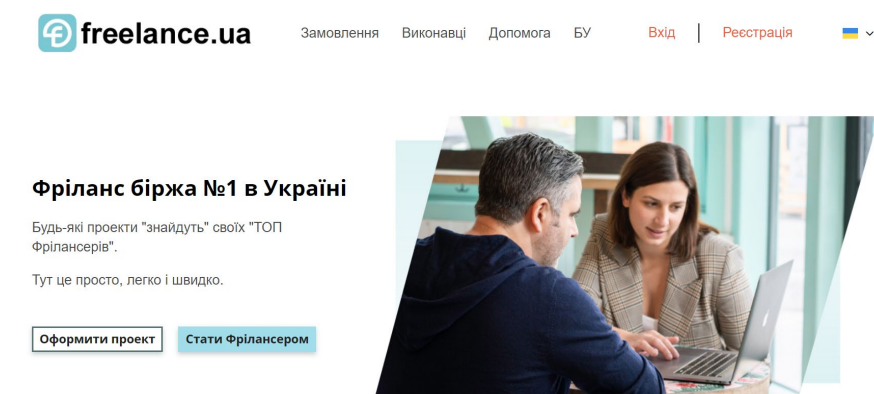


Рисунок 1.2. Стартова сторінка freelance.ua

На платформі Freelance.ua користувачі можуть знаходити замовлення в різних напрямках, зокрема:

- вебпрограмування;
- дизайн вебсайтів;
- верстка сторінок;
- розробка логотипів;
- копірайтинг;
- відеомонтаж;
- SEO-просування;
- контекстна реклама та інші види діяльності.

Для зручності пошуку вакансій сервіс пропонує декілька категорій:

- разові проєкти;
- постійна віддалена робота;
- робота в офісі.

Усі фінансові операції на Freelance.ua здійснюються через систему «безпечна угода», комісія якої становить 7,5 % і сплачується замовником. Платформа підтримує два типи акаунтів:

- базовий, що дозволяє подавати одну заявку на добу на проєкти з бюджетом до 400 грн;
- Pro — платний обліковий, який надає можливість необмежено подавати заявки та брати участь у проєктах із бюджетом понад 400 грн.

### *Weblancer.net*

Сервіс Weblancer.net функціонує понад 18 років і об'єднує більше ніж 1,5 мільйона користувачів. На біржі представлені такі напрями, як веброзробка, програмування програмного забезпечення, створення текстів і переклади, графіка та фотографія, поліграфія й айдентика, архітектура та інжиніринг, адміністрування вебсайтів та багато інших сфер [14].

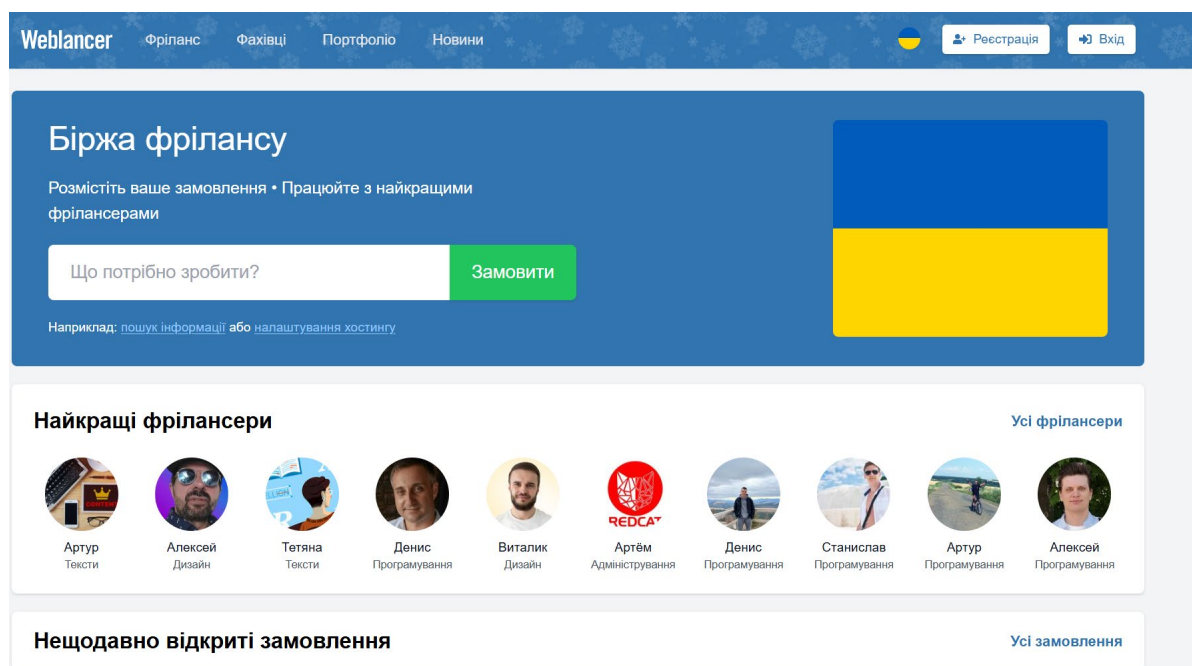


Рисунок 1.3. Стартова сторінка weblancer.net

На платформі можна безоплатно публікувати разові замовлення, конкурси, а також розміщувати резюме й портфоліо. Після реєстрації кожен фрілансер отримує п'ять безкоштовних заявок. Додатково ще 50 заявок нараховується за повне заповнення профілю, а саме:

- додавання фотографії — +10 заявок;
- заповнення резюме — +10 заявок;
- перелік наданих послуг — +10 заявок;
- завантаження робіт у портфоліо — +20 заявок.

Фрілансери також мають можливість придбати додаткові заявки двома способами: оформлення передплати терміном на три місяці, вартість якої залежить від обраної категорії; – разова купівля універсальних заявок, тарифи на які стають доступними після підтвердження номера телефону.

Комісія сервісу Weblancer.net становить 5 % від вартості виконаного замовлення. Усі спірні питання на платформі розглядаються професійними юристами.

### *Free-lance.ua*

До найпопулярніших категорій на платформі Free-lance.ua належать менеджмент, розробка вебсайтів, дизайн, арт, програмування, оптимізація, тексти, переклади та інші напрями (рис. 1.4). Біржа пропонує три основні формати співпраці між замовниками та фрілансерами:

- проєкти — разові швидкі завдання;
- конкурси — замовник встановлює бюджет і отримує від виконавців різні варіанти виконання, найчастіше у сферах неймінгу, створення логотипів чи ілюстрацій;
- вакансії — можливість працевлаштування фрілансера до штату компанії.

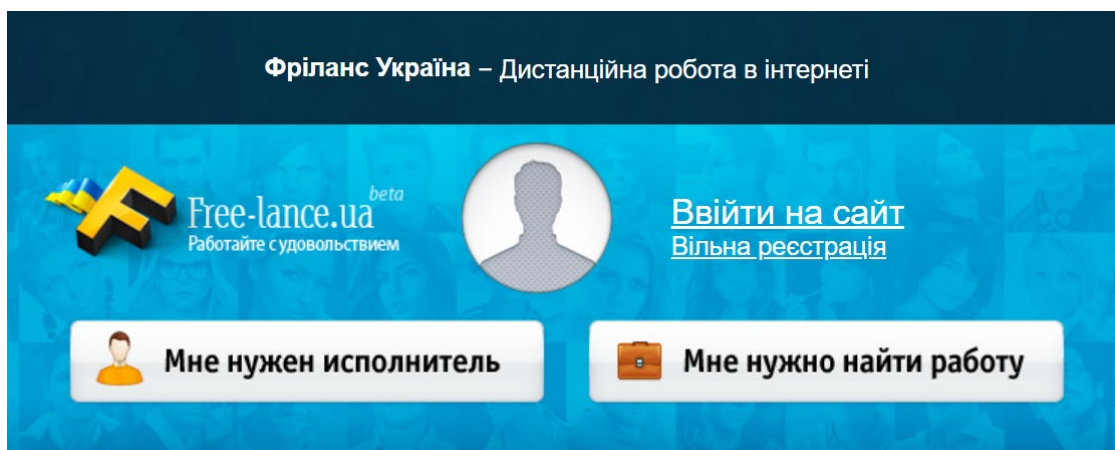


Рисунок 1.4. Стартова сторінка free-lance.ua

### *Kabanchik.ua*

Платформа Kabanchik.ua (рис. 1.6) функціонує з 2012 року та суттєво відрізняється від класичних фріланс-бірж за принципом своєї роботи.

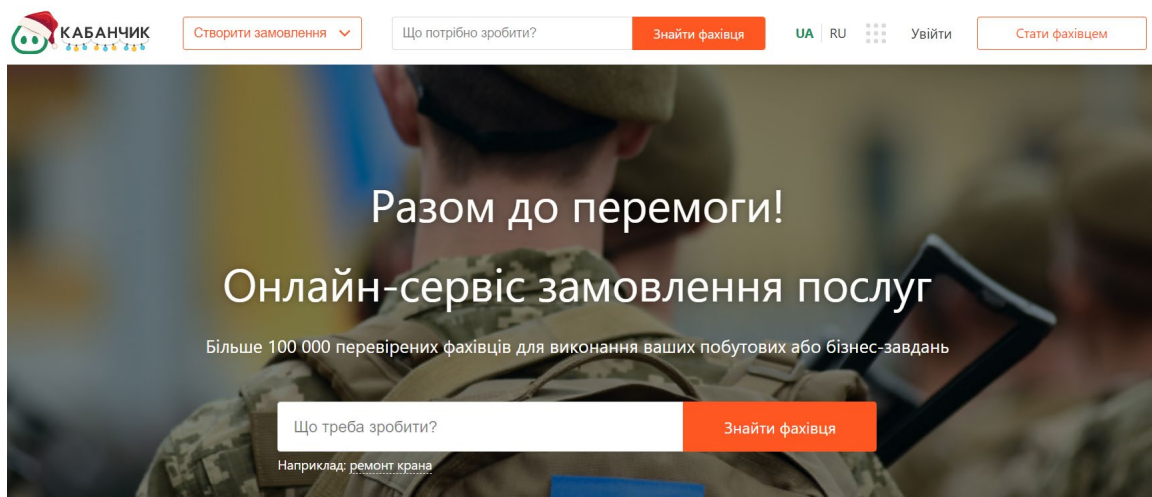


Рисунок 1.5. Стартова сторінка kabanchik.ua

На Kabanchik.ua можна замовляти різноманітні послуги через інтернет, зокрема:

- оздоблювальні та будівельні роботи;
- кур'єрські та побутові послуги;
- ремонт техніки;
- послуги для авто;
- дизайн;
- організація свят;
- репетиторство та інші види послуг.

На платформі легко знайти фахівця, який догляне за домашнім улюбленцем або відремонтує побутову техніку. Окремий розділ присвячено саме фріланс-послугам, які поділяються на такі категорії: робота в інтернеті, розробка вебсайтів та додатків, дизайн, інтернет-реклама та інші напрями.

Сервіс має власний мобільний додаток, доступний у Google Play та App Store. У 2015 році Kabanchik.ua придбав український майданчик Prom.ua.

Українські фахівці також можуть працювати на міжнародних платформах, серед яких найбільш популярні:

- Upwork.com;
- Fiverr.com;



- WorkingNomads.co/jobs;
- DesignHill.com;
- Gigster.com.

Ринок фрілансу в Україні продовжує зростати, навіть під час війни. За даними Freelancehunt, у 2022 році кількість виконаних замовлень зросла на 31 % порівняно з попереднім роком, а обсяг фінансових операцій збільшився на 29 %.

## 1.2 Аналіз систем пошуку виконавців ІТ-проектів

Процес отримання замовлення на фрілансі зазвичай виглядає так: спеціаліст обирає платформу, подає заявку на цікавий проєкт, виконує роботу та отримує оплату. Проте на практиці все значно складніше: знайти вигідний проєкт і виділитися серед інших виконавців буває непросто.

Існує безліч спеціалізованих вебсайтів, які допомагають фрілансерам знаходити нові замовлення. Кожна біржа має власні правила участі, механізми безпечної оплати та гарантії для користувачів.

Для порівняння були проаналізовані чотири провідні фріланс-платформи, які мають високі позиції на ринку (табл. 1.1).

Таблиця 1.1 – Порівняльна характеристика

| № | Назва                | Переваги   | недоліки                                    |
|---|----------------------|--|---|
| 1 | <b>Freelancehunt</b> | Різні способи оплати.<br>Прозора робота: правила сервісу та інструкції детально описані в базі знань.<br>Зворотній зв'язок: команда відповідає на всі запитання через форум, соцмережі та блог.<br>Інструменти безпеки: Сейф, Бізнес-Сейф. | Висока конкуренція на популярні види робіт. |

|   |                   |  |  |
|---|-------------------|--|--|
|   |                   | <p>Легке просування по рейтингу.</p> <p>Різноманітні можливості заробітку: проекти, конкурси та бонусна програма.</p>  |  |
| 2 | <b>Weblancer</b>  | <p>Можна отримати консультації від досвідчених колег.</p> <p>Досвідчений і перевірений механізм роботи, збалансований під потреби користувачів.</p>  | <p>Оплата через «безпечну угоду» можлива лише на гаманець WebMoney.</p> <p>Висока комісія за операції.</p> <p>Прив'язка до IP-адреси.</p> <p>Своєрідна робота адміністрації з клієнтами.</p> |
| 3 | <b>Free-lance</b> | <p>Можливість знайти виконавців з усього світу без прив'язки до часових поясів.</p> <p>Зручна служба підтримки, проблеми вирішуються швидко.</p> <p>Безпечні платежі.</p> <p>Зручний чат для спілкування з клієнтами.</p> <p>Тести на компетентність допомагають оцінити навички фрілансера.</p> | <p>Велика кількість неякісних або фейкових проєктів.</p> <p>Автоматизовані заявки фрілансерів ускладнюють оцінку реальних можливостей виконавця.</p>   |
| 4 | <b>UpWork</b>     | <p>Високі тарифи на проєкти, порівняно з бюджетними платформами.</p> <p>Платежі вбудовані в систему,</p>   | <p>Висока конкуренція.</p> <p>Комісія за проєкт від 5 % до 20 %, залежно від заробітку.</p>  |

|  |  |  |   |
|--|--|--|---|
|  |  | легко здійснювати оплату.<br>Нова структура подання пропозицій підвищує шанси на працевлаштування.<br>Віддалена робота забезпечує гнучкість. | Користувацький досвід на деяких сторінках застарілий. |
|--|--|--|---|

Кожна існуюча інформаційна фріланс-платформа виступає своєрідним містком між традиційною економікою та цифровими ринками. Проаналізувавши сучасні сервіси, можна виділити ключові розділи та функції, які доцільно врахувати при розробці власної інформаційної платформи.

По-перше, важливо приділити особливу увагу зручності та надійності використання. Візуальна складова вебсайту є критичною, оскільки саме вона впливає на перше враження користувача: привертає увагу, зацікавлює та утримує від переходу з платформи.

По-друге, необхідно враховувати політику комісій та оплату за використання платформи. На початковому етапі не планується стягування плати за реєстрацію, проте ця функція може бути впроваджена пізніше. Важливо, щоб відсоток комісії був помірним, адже він безпосередньо впливає на залученість користувачів та загальну активність платформи.

### 1.3 Висновки до розділу 1

Проаналізувавши сучасні вітчизняні пошукові сервіси, можна визначити основний функціонал, який характерний для всіх платформ, а також спільні недоліки.

Базовий функціонал таких сайтів включає:

- реєстрацію та авторизацію користувачів;
- перегляд локальної інформації та новин порталу;
- зручну навігацію по сайту;
- пошукову систему;

- створення завдань і замовлень для фрілансерів та клієнтів;
- перегляд завдань і замовлень;
- можливість залишати відгуки на виконані завдання;
- обмін особистими повідомленнями між користувачами.

Спільні недоліки існуючих платформ включають:

- складну або незручну навігацію;
- інтерфейс із «кривими» елементами та надмірною інформаційною завантаженістю;
- надмірну кількість умов для початку роботи: необхідність оформлення передплати, введення зайвих даних, велика кількість реклами.

Для розробки проєкту важливо чітко визначити вимоги до інформаційної системи (ІС):

*Функціональні вимоги* — забезпечення основних операцій користувачів;

*Вимоги до інтерфейсу* — зручність, простота та інтуїтивність;

*Вимоги до продуктивності* — швидка обробка даних та стабільність роботи платформи;

*Вимоги до безпеки* — захист персональних даних та фінансових операцій.

## **РОЗДІЛ 2**

### **АНАЛІЗ СУЧАСНИХ ТЕХНОЛОГІЙ ДЛЯ РЕАЛІЗАЦІЇ ПОШУКОВОЇ ІНФОРМАЦІЙНОЇ СИСТЕМИ**

Існує багато платформ, які дозволяють користувачам знаходити роботу або отримувати допомогу в різних сферах діяльності. Через зростання кількості таких сервісів та конкуренції на ринку доступні робочі місця часто не забезпечують дохід на рівні прожиткового мінімуму, що створює труднощі для багатьох людей. З іншого боку пошук кваліфікованого фахівця потребує від кандидатури вимагає відповідного рівня професіоналізму, особистісних якостей та інших характеристик.

В Україні вже існують сервіси, які дозволяють знайти компетентних виконавців і сприяють встановленню вигідних ділових відносин. Паралельно розробляються нові платформи з розширеними можливостями, що використовують сучасні веб-технології. Для подальшої реалізації пошукової інформаційної системи доцільно провести аналіз сучасних технологій створення веб-орієнтованих інформаційних систем.

#### **2.1 Аналіз програмних веб-технологій для розробки пошукової інформаційної системи**

Сучасні веб-технології спрямовані на ефективне керування інформаційним наповненням сайту та обробку даних, що надходять від відвідувачів. Зазвичай такі рішення базуються на серверних технологіях, наприклад ASP, ASP.NET, JSP, PHP, або на готових платформах для створення корпоративних сайтів.

ASP / ASP.NET (Active Server Pages) — технологія Microsoft, що дозволяє створювати динамічні веб-сторінки з відокремленням функціональної частини від дизайну. ASP-сторінки містять HTML-текст, а також сценарії на JavaScript або VBScript. Сервер обробляє запити та генерує HTML-код, який відображається у браузері. Ця технологія стала базою для

розвитку інших платформ, таких як PHP та JSP.

JSP (Java Server Pages) — технологія для створення серверних сторінок на Java, що є розширенням Java Servlet API. JSP забезпечує кросплатформенність та дозволяє розробляти зручні для користувача веб-додатки. Специфікація JSF (Java Server Faces) визначає правила побудови серверних компонентів інтерфейсу користувача.

CGI (Common Gateway Interface) — одна з перших технологій створення серверних веб-застосунків. CGI дозволяє виконувати серверні скрипти, які генерують HTML-код у відповідь на запит браузера. CGI-застосунки можуть бути написані на різних мовах програмування та запускатися як консольні програми під керуванням операційної системи, що обслуговує веб-сервер.

PHP (Personal Home Pages / Hypertext Preprocessor) — вільна та популярна технологія для створення серверних веб-сторінок. PHP заснована на використанні CGI-застосунків і дозволяє інтегрувати скрипти безпосередньо в HTML-код. Головною перевагою PHP є її практичність та гнучкість: мова легко інтегрується з HTML, JavaScript, WML, XML та іншими мовами програмування для вебу. PHP надає програмістам інструменти для швидкої та ефективної реалізації завдань.

Результати порівняльного аналізу сучасних технологій розробки веб-ресурсів наведено в таблиці 2.1. Розглянуті технології забезпечують сучасну функціональність веб-додатків, ефективний супровід процесів створення сайтів та управління інформаційним контентом.

*Таблиця 2.1 – Порівняння сучасних технологій розробки веб-сайтів*

|  | PHP | JSP | ASP.NET |
|--|-----|-----|---------|
| Багатоплатформеність                     | +   | +   | -       |
| Продуктивність                           | +/- | +/- | +       |
| Простота використання                    | +   | +/- | +/-     |
| Наявність доступних програмних бібліотек | +   | +   | +       |

|                            |     |     |   |
|----------------------------|-----|-----|---|
| Розподіл дизайну та логіки | +/- | +/- | + |
|----------------------------|-----|-----|---|

Результати проведеного аналізу дозволяють зробити висновок про переваги PHP-технології в експлуатації. PHP користується значною популярністю серед вебпрограмістів і на сьогодні є однією з найпоширеніших мов програмування для створення вебдодатків та серверних скриптів. Основні переваги PHP полягають у практичності, легкості застосування, ефективності, високій продуктивності та гнучкості, що робить її оптимальним вибором для розробки динамічних вебсайтів.

## 2.2 Аналіз засобів реалізації баз даних у вебтехнологіях

Робота з базами даних є ключовою складовою процесу розробки динамічних вебсайтів. Базы даних використовуються для зберігання різнопланової інформації, що необхідна для роботи ресурсу. У спрощеному вигляді база даних являє собою набір взаємозв'язаних таблиць, розміри та кількість яких можуть бути різними. На сервері бази даних акумулюють інформацію статистичного та оперативного характеру.

Серед найбільш популярних систем керування базами даних (СКБД) виділяють MySQL, SQL, Oracle Database та інші. Вибір конкретної СКБД визначається функціональними вимогами та характеристиками інформаційної системи.

MySQL — одна з найпоширеніших безкоштовних СКБД, яка активно використовується для створення динамічних веб-сторінок. Вона підтримує роботу з різними мовами програмування, є багатопоточною, проста у встановленні та налаштуванні. Серед її переваг: підтримка паралельної роботи багатьох користувачів, обробка великих обсягів даних (до 50 млн рядків у таблицях), висока швидкість виконання запитів та надійна система безпеки.

Oracle Database — об'єктно-реляційна СКБД, яка орієнтована на

операційні системи Windows, Unix, Linux та MacOS. На відміну від MySQL, Oracle має ширшу сферу застосування і використовується у великих комерційних та державних інформаційних системах. Основними обмеженнями при розробці невеликих вебсайтів є висока вартість ліцензії та складність у пошуку хостингу з підтримкою Oracle.

SQL (Structured Query Language) — декларативна мова для роботи з реляційними базами даних, що дозволяє формувати запити, оновлювати та керувати даними, створювати схеми бази даних і модифікувати їх, а також контролювати доступ до інформаційних ресурсів. SQL може виконувати інтерактивні запити або використовуватися як вбудована мова у прикладних програмах для управління даними. Крім того, стандарт SQL підтримує механізми перевірки та захисту інформації, що забезпечує надійність роботи з базою даних.

*Таблиця 2.2. – Порівняльна характеристика СКБД*

|                        |  | SQL | Oracle | MySQL |
|------------------------|--|-----|--------|-------|
| Надійність             |  | +   | +      | +     |
| Швидкодія              |  | -   | +      | +     |
| Простота               |  | -   | -      | +     |
| Зручність використання |  | +/- | +      | +     |

### **2.3 Мова розмітки веб-документів HTML**

Більшість веб-сторінок створюються із використанням мови розмітки HTML або її строгого варіанту XHTML. HTML інтерпретується браузерами та відображається у вигляді документа, зручного для користувача. До п'ятої версії HTML належить до стандарту SGML (ISO 8879), тоді як XHTML дотримується суворих правил XML, фактично розширюючи HTML для забезпечення сумісності з XML-документами.



З появою HTML5 були введені нові елементи та атрибути, що спрощують семантичну розмітку та замінюють універсальні елементи `<div>` і `<span>`. Наприклад:

`<nav>` — блок навігації сайту;

`<footer>` — нижня частина сторінки;

`<audio>` та `<video>` — мультимедійні елементи замість `<object>`.

HTML5 також включає API, доступне для взаємодії з JavaScript, і підтримує розширені можливості DOM (Document Object Model), який забезпечує програмний доступ до структури документів HTML, XHTML і XML. XHTML5 — це XML-серіалізація HTML5, яка вимагає суворого дотримання синтаксису XML і визначає MIME-тип документа для правильного відображення.

Вибір між HTML5 і XHTML5 визначається обраним типом медіа (MIME-type), який визначає формат документа.

## 2.4 Каскадні таблиці стилів (CSS)

CSS (Cascading Style Sheets) — формальна мова опису зовнішнього вигляду документів, створених за допомогою мов розмітки (HTML, XHTML). Основна мета CSS — розділити логічну структуру документа від його презентації, що дозволяє легко змінювати стиль оформлення без зміни HTML-коду.

Правила CSS можуть зберігатися:

1. У зовнішньому файлі і підключатися за допомогою `<link>` у `<head>`:  

```
<link rel="stylesheet" href="style.css">
```
2. У зовнішньому файлі, підключеному через директиву `@import` всередині тегу `<style>`:  

```
<style media="all">
    @import url(style.css);
</style>
```
3. У внутрішньому CSS, розташованому всередині тегів `<style>` у `<head>`:

```
<style>
  body { color: red; }
</style>
```

4. У елементі документа, застосованому локально до конкретного тега через атрибут style:

```
<p style="font-size: 20px; color: green; font-family: Arial, Helvetica, sans-serif;">
```

...

```
</p>
```

Для XML-документів CSS підключається через спеціальне посилання:

```
<?xml-stylesheet type="text/css" href="style.css"?>
```

До появи CSS оформлення здійснювалося безпосередньо у HTML-кодi.

CSS дозволяє:

- Застосовувати різні стилі для різних пристроїв (екран, друк, мобільні пристрої, Брайль);
- Зменшувати час завантаження сторінок, оскільки браузер кешує зовнішні стилі;
- Легко змінювати дизайн сайту, редагуючи лише один файл CSS;
- Використовувати додаткові ефекти, наприклад, фіксоване меню або обтікання тексту блоками.
- Недоліки CSS:
  - Різне відображення в різних браузерах;
  - Іноді необхідно редагувати HTML-код через складні взаємозв'язки між селекторами CSS;
  - Деякі розширення CSS3 замінюються зображеннями для кросбраузерності (наприклад, закруглені кути).

CSS використовує модель блоку, де властивість width визначає ширину вмісту, без урахування відступів і рамок. Селектори CSS дозволяють задавати стилі для окремих елементів або груп елементів (дочірні, нащадки,

сестринські). Це забезпечує гнучке управління структурою документа і підвищує доступність контенту для різних типів відтворення.

## 2.5 Динамічна мова JavaScript

Для реалізації інтерактивності та анімації на веб-платформі фріланс-біржі буде використана мова JavaScript (JS) та її популярні фреймворки.

JavaScript — прототипно-орієнтована скриптова мова програмування, діалект ECMAScript, широко застосовується у браузерях для додавання інтерактивності веб-сторінок. Основні характеристики мови:

- динамічна та слабка типізація;
- автоматичне керування пам'яттю;
- прототипне програмування;
- функції як об'єкти першого класу;
- С-подібний, але спрощений синтаксис.
- JS використовується для:
  - створення інтерактивних веб-сторінок;
  - односторінкових веб-застосунків (ReactJS, AngularJS, Vue.js);
  - серверного програмування (Node.js);
  - десктопних застосунків (Electron, NW.js);
  - мобільних застосунків (React Native, Cordova);
  - сценаріїв у прикладному ПЗ (наприклад, Adobe Creative Suite);
  - інтеграції в PDF-документи та інші середовища.

Поява AJAX (Asynchronous JavaScript and XML) дозволила виконувати запити до сервера в асинхронному режимі, оновлюючи окремі частини сторінки без її повного перезавантаження. Це значно підвищило ефективність і інтерактивність веб-застосунків.

JS підтримує концепції об'єктно-орієнтованого та функціонального програмування: об'єкти з прототипами, функції як об'єкти першого класу, каррінг, анонімні функції та замикання (closures). Базові вбудовані об'єкти включають: Global, Object, Array, Function, Date, Math та ін.

### *Бібліотека jQuery*

jQuery — популярна бібліотека JavaScript, яка спрощує взаємодію з HTML та DOM, обробку подій, анімацію та роботу з AJAX. Основні принципи:

- Відокремлення поведінки від структури HTML (ненав'язливий JS);
- Модульність через ядро та плагіни, що дозволяє підключати лише необхідний функціонал;
- Спрощена робота з DOM-елементами та AJAX-запитами (\$.post()).

#### Переваги jQuery:

- Малий розмір дистрибутива;
- Простота у використанні та добре документована;
- Лаконічний та розширюваний синтаксис.

#### Недоліки jQuery:

- Можливе уповільнення роботи при великих проектах;
- Проблеми сумісності з деякими браузерами.

## **2.6 Висновки до розділу 2**

У цьому розділі проведено аналіз сучасних веб-технологій для розробки інформаційних систем: PHP, MySQL, HTML, CSS, JavaScript та jQuery. Основні висновки: PHP — популярна серверна мова, гнучка та ефективна для створення динамічних веб-додатків; MySQL — надійна, швидка та доступна СКБД, оптимальна для зберігання даних динамічних сайтів; HTML5 та XHTML5 — забезпечують структуру та семантичну розмітку документа; CSS3 — відокремлює зовнішній вигляд від структури документа, дозволяє кросбраузерне та адаптивне оформлення; JavaScript — динамічна мова сценаріїв для інтерактивності, з підтримкою прототипного ООП та функціонального програмування; jQuery — бібліотека JS, що спрощує роботу з DOM, подіями та AJAX-запитами. Підхід на основі вбудовування інтерпретованого коду в HTML-шаблони дозволяє ефективно реалізовувати динамічні веб-застосунки. Кожна технологія має свої переваги та обмеження, що визначає їх область застосування.

Для створення пошукової інформаційної системи фріланс-біржі оптимально використовувати комбінацію: PHP + MySQL + HTML + CSS + JavaScript + jQuery, що забезпечує інтерактивність, динамічність, зручний інтерфейс та надійну роботу з базами даних.

## **РОЗДІЛ 3**

### **АРХІТЕКТУРА ТА ОПИС РОЗРОБКИ**

Метою даного проєкту є розроблення веб-орієнтованої інформаційної системи для пошуку виконавців іт-проєктів, яка буде корисною як для замовників, так і для фахівців-виконавців. Запропонований сервіс забезпечує зручний пошук замовлень, спрощує процес комунікації між клієнтами та фрілансерами, а також дозволяє оцінювати професійний рівень спеціалістів за допомогою портфолію, відгуків, рейтингу та статистичних показників профілю. Уся зазначена інформація формується безпосередньо під час роботи на платформі та є достовірною.

У цьому розділі наведено вимоги до системи, описано моделі даних, концептуальні схеми та принципи функціонування розробленого програмного продукту.

#### **3.1 Вимоги до проєкту веборієнтованої ІС**

Цільовими користувачами системи є програмісти, дизайнери, seo-спеціалісти, фахівці з відео- та аудіомонтажу, копірайтери і рерайтери, менеджери інтернет-проєктів, адміністратори соціальних мереж, а також розробники мобільних додатків. Система передбачається як відкрита платформа для розміщення та виконання іт-проєктів.

До основних функціональних вимог віднесено: реєстрацію та авторизацію користувачів; створення і публікацію замовлень з можливістю додавання файлів; перегляд переліку робіт; реалізацію механізму пошуку за різними критеріями; формування заявок на виконання проєктів; обмін повідомленнями між замовником і виконавцем у режимі чату; ведення рейтингу та історії проєктів; систему сповіщень; можливість блокування користувачів і функціонування служби підтримки.

У системі передбачено три основні групи користувачів: адміністратори, замовники та виконавці. Адміністратори здійснюють блокування облікових

записів і надають підтримку користувачам. Замовники мають змогу створювати, редагувати та видаляти завдання, а також взаємодіяти з виконавцями та оцінювати їхні роботи. Виконавці переглядають доступні завдання, подають заявки на участь у проєктах, спілкуються із замовниками, оцінюють роботи інших фахівців, формують командні акаунти та генерують власні резюме.

Особливістю системи є можливість створення облікового запису типу «команда», що дозволяє об'єднувати декілька виконавців для спільної роботи над проєктами. Крім того, передбачено вбудований конструктор резюме, який автоматично формує документ на основі навичок і досвіду користувача, а також функцію ведення блогу, доступну як замовникам, так і виконавцям.

Реалізація системи здійснюється із застосуванням сучасних веб-технологій php, html, css, javascript, jquery та скбд mysql. Загальні вимоги до програмного забезпечення охоплюють відповідність операційній системі, зручність інтерфейсу та необхідний функціональний набір.

### 3.2 ER-діаграма

Модель «сутність-зв'язок» (ERM) є концептуальним представленням даних та використовується для створення реляційної бази даних.

Основні елементи ER-діаграми:

- *Сутність* – об'єкт, про який зберігається інформація (Рис. 3.1).
- *Зв'язок* – іменована асоціація між сутностями, яка відображає взаємодію об'єктів (Рис. 3.2).
- *Атрибути* – властивості сутностей та зв'язків, що описують їх характеристики (Рис. 3.3).

Використовуючи зазначені елементи, була побудована ER-діаграма для розробленого продукту (Рис. 3.4), що відображає:

- користувачів (адміністратори, замовники, виконавці);
- замовлення та заявки;
- взаємодію через чат;

- рейтинг, портфоліо та резюме;
- ведення блогу та історію проєктів.

ER-діаграма дозволяє чітко структурувати дані та забезпечує ефективну розробку реляційної бази даних для веб-орієнтованої ІС.



Рис. 3.1. Позначення сутності в ER-діаграмах

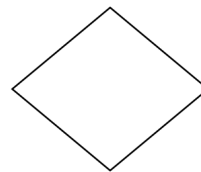


Рис. 3.2. Позначення зв'язку в ER-діаграмах

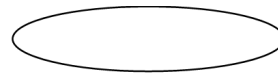


Рис. 3.3. Позначення атрибуту в ER-діаграмах

Користуючись зазначеними вище фігурами була побудована ER-діаграма для розробленого продукту (рис. 3.4)



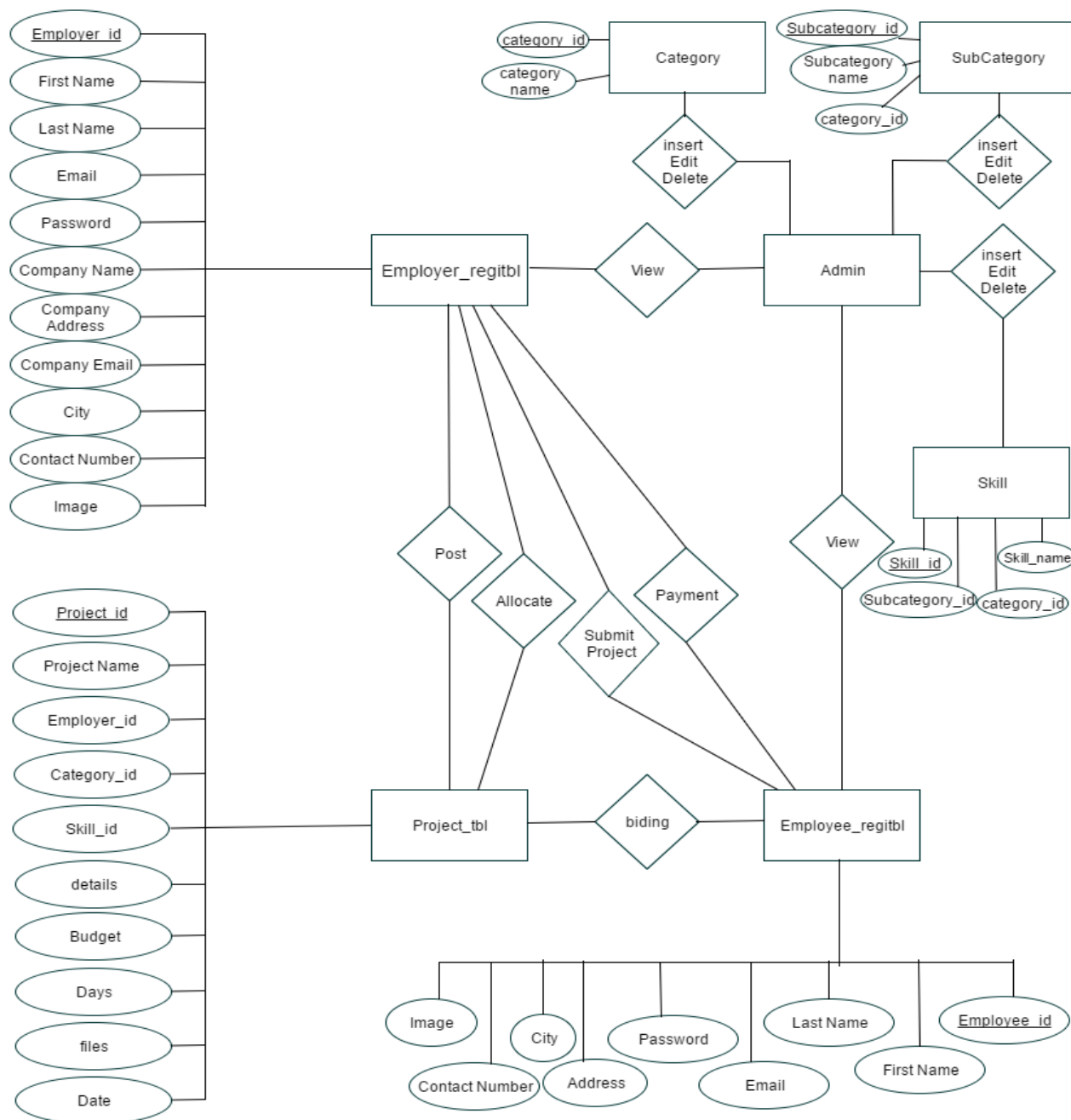


Рис. 3.4. ER-діаграма додатку

### 3.3 Структура таблиць бази даних

Веб-додаток оперує базою даних на засадах MySQL, що складається за 15 таблиць створених відповідно до їх доменних сутностей. До цього переліку входить:

- Employee\_regtbl - таблиця використовується для зберігання всієї інформації про робітника.

Таблиця 3.1. Структура таблиці Employee\_regitbl

| Поле        | Тип     | Обмеження      | Опис                            |
|-------------|---------|----------------|---------------------------------|
| employee_id | int(11) | Auto increment | Унікальний ідентифікатор        |
| Fn          | Text    | Not Null       | Зберегти ім'я                   |
| Ln          | Text    | Not Null       | Зберегти прізвище               |
| Em          | Text    | Not Null       | Електронна пошта                |
| Un          | Text    | Not Null       | Ім'я користувача.               |
| Pass        | Text    | Not Null       | Пароль.                         |
| cn1         | Text    | Not Null       | Мобільний номер.                |
| cn2         | Text    | Not Null       | Запасний мобільний номер.       |
| Address     | Text    | Not Null       | Адреса магазину.                |
| City        | Text    | Not Null       | Назва міста магазину.           |
| Que         | Text    | Not Null       | Магазин питання.                |
| Ans         | Text    | Not Null       | Відповідь.                      |
| Balance     | int(11) | Not Null       | Баланс для працівника           |
| Img_upld    | Text    | Not Null       | Зображення працівника магазину. |

– Employer\_regitbl - таблиця використовується для зберігання всієї інформації про роботодавця.

Таблиця 3.2. Структура таблиці Employer\_regitbl

| Поле        | Тип     | Обмеження      | Опис                              |
|-------------|---------|----------------|-----------------------------------|
| employer_id | int(11) | Auto increment | Унікальний ідентифікатор          |
| Fn          | Text    | Not Null       | Ім'я                              |
| Ln          | Text    | Not Null       | Прізвище                          |
| Em          | Text    | Not Null       | Адреса електронної пошти магазину |
| Un          | text    | Not Null       | Ім'я користувача                  |
| Pass        | text    | Not Null       | Пароль                            |

|          |         |          |                        |
|----------|---------|----------|------------------------|
| Cnm      | text    | Not Null | Назва компанії         |
| Cadd     | text    | Not Null | Адреса компанії        |
| Cem      | text    | Not Null | Електронна пошта       |
| cnl      | text    | Not Null | Мобільний номер        |
| City     | text    | Not Null | Місто                  |
| Que      | text    | Not Null | Питання                |
| Ans      | text    | Not Null | Відповідь              |
| Balance  | int(11) | Not Null | Баланс                 |
| Img_upld | text    | Not Null | Зображення роботодавця |

– Adminlogin - таблиця використовується для зберігання інформації про адміністратора.

Таблиця 3.3. Структура таблиці Adminlogin

| Поле     | Тип     | Обмеження      | Опис                     |
|----------|---------|----------------|--------------------------|
| Admid    | int(11) | Auto increment | Унікальний ідентифікатор |
| Admemail | text    | Not Null       | Електронна пошта         |
| Admpass  | text    | Not Null       | Пароль                   |
| Admdp    | text    | Not Null       | Нікнейм                  |
| Secque   | text    | Not Null       | Питання                  |
| Secans   | text    | Not Null       | Відповідь                |

– Catetbl - таблиця використовується для зберігання всієї інформації про категорію проекту.

Таблиця 3.4. Структура таблиці Catetbl

| Поле      | Тип     | Обмеження      | Опис                     |
|-----------|---------|----------------|--------------------------|
| cate_id   | int(11) | Auto increment | Унікальний ідентифікатор |
| cate name | text    | Not Null       | Назва категорії          |

– Subcatetbl - таблиця використовується для зберігання інформації про підкатегорію.

Таблиця 3.5. Структура таблиці Subcatetbl

| Поле | Тип | Обмеження | Опис |
|------|-----|-----------|------|
|------|-----|-----------|------|

|          |         |                |                          |
|----------|---------|----------------|--------------------------|
| sub_id   | int(11) | Auto increment | Унікальний ідентифікатор |
| sub_name | text    | Not Null       | Назва підкатегорії       |
| cate_id  | int(11) | Not Null       | Ідентифікатор категорії  |

– Skilltbl - таблиця використовується для зберігання інформації про навички робітника.

*Таблиця 3.6. Структура таблиці Skilltbl*

| Поле       | Тип     | Обмеження      | Опис                       |
|------------|---------|----------------|----------------------------|
| skill_id   | int(11) | Auto increment | Унікальний ідентифікатор   |
| skill_name | text    | Not Null       | Назва навички              |
| cate_id    | int(11) | Not Null       | Ідентифікатор категорії    |
| sub_id     | int(11) | Not Null       | Ідентифікатор підкатегорії |

– Contactustbl - таблиця використовується для зберігання всіх повідомлень адміністратору, які можуть надсилати будь-які зареєстровані або незареєстровані користувачі.

*Таблиця 3.7. Структура таблиці Contactustbl*

| Поле    | Тип     | Обмеження      | Опис                     |
|---------|---------|----------------|--------------------------|
| comp_id | int(11) | Auto increment | Унікальний ідентифікатор |
| Email   | text    | Not Null       | Електронна пошта         |
| Subject | text    | Not Null       | Тема                     |
| Msg     | text    | Not Null       | Повідомлення             |

– Prjtbl - Ця таблиця використовується для зберігання всієї інформації про проєкт, який може опублікувати роботодавець.

*Таблиця 3.8. Структура таблиці Prjtbl*

| Поле        | Тип     | Обмеження      | Опис                      |
|-------------|---------|----------------|---------------------------|
| p_id        | int(11) | Auto increment | Унікальний ідентифікатор  |
| employer_id | int(11) | Not Null       | Ідентифікатор роботодавця |
| p_name      | text    | Not Null       | Назва проєкту             |
| cate_id     | int(11) | Not Null       | Ідентифікатор категорії   |
| skill_id    | int(11) | Not Null       | Ідентифікатор навичок     |
| Details     | text    | Not Null       | Деталі проєкту            |
| Budget      | text    | Not Null       | Бюджет проєкту            |
| Days        | int(11) | Not Null       | День закінчення           |
| Files       | text    | Not Null       | Вкладений файл            |
| Date        | date    | Not Null       | Дата розміщення           |

– Prj\_allocated - таблиця використовується для зберігання всієї інформації про проєкт з виділеним бюджетом, що виділяється замовником.

Таблиця 3.9. Структура таблиці Prj\_allocated

| Поле         | Тип     | Обмеження      | Опис                      |
|--------------|---------|----------------|---------------------------|
| allocated_id | int(11) | Auto increment | Унікальний ідентифікатор  |
| employer_id  | int(11) | Not Null       | Ідентифікатор роботодавця |
| employee_id  | int(11) | Not Null       | ID робітника              |
| p_id         | int(11) | Not Null       | Ідентифікатор проєкту     |
| Budget       | int(11) | Not Null       | Бюджет проєкту            |
| Date         | date    | Not Null       | Дату розміщення           |

– Message - таблиця використовується для зберігання всіх повідомлень, які можуть надсилати працівник і роботодавець.

Таблиця 3.10. Структура таблиці Message

| Поле | Тип     | Обмеження      | Опис                     |
|------|---------|----------------|--------------------------|
| Id   | int(11) | Auto increment | Унікальний ідентифікатор |

|          |         |          |                       |
|----------|---------|----------|-----------------------|
| p_id     | int(11) | Not Null | Ідентифікатор проєкту |
| Username | text    | Not Null | Ім'я користувача      |
| Message  | text    | Not Null | Повідомлення          |

– Bidtbl - таблиця використовується для зберігання всієї інформації про тендери на проєкти, які може робити робітник.

*Таблиця 3.11. Структура таблиці Bidtbl*

| Поле        | Тип     | Обмеження      | Опис                      |
|-------------|---------|----------------|---------------------------|
| bid_id      | int(11) | Auto increment | Унікальний ідентифікатор  |
| employer_id | int(11) | Not Null       | Ідентифікатор роботодавця |
| employee_id | int(11) | Not Null       | ID робітника              |
| Budget      | text    | Not Null       | Бюджет проєкту            |
| Days        | int(11) | Auto increment | Днів до завершення        |
| project_id  | int(11) | Not Null       | Ідентифікатор проєкту     |
| Status      | text    | Not Null       | Статус заказу             |
| Date        | date    | Not Null       | Дату розміщення           |

– Choosed\_skill - таблиця використовується для зберігання всієї інформації про навички, які може вибрати працівник.

*Таблиця 3.12. Структура таблиці Choosed\_skill*

| Поле        | Тип     | Обмеження      | Опис                     |
|-------------|---------|----------------|--------------------------|
| cs_id       | int(11) | Auto increment | Унікальний ідентифікатор |
| skill_id    | int(11) | Not Null       | Ідентифікатор навичок    |
| employee_id | int(11) | Not Null       | ID робітника             |

– Completeprj - таблиця використовується для зберігання інформації про завершені проєкти.

*Таблиця 3.13. Структура таблиці Completeprj*

| Поле           | Тип     | Обмеження      | Опис                      |
|----------------|---------|----------------|---------------------------|
| completeprj_id | int(11) | Auto increment | Унікальний ідентифікатор  |
| employer_id    | int(11) | Not Null       | Ідентифікатор роботодавця |
| employee_id    | int(11) | Not Null       | ID робітника              |
| Prjfile        | text    | Not Null       | Комп. файл проекту        |
| complete_date  | date    | Not Null       | Дата завершення           |

– Employee\_acc - таблиця використовується для зберігання інформації про облікові записи працівника.

*Таблиця 3.14. Структура таблиці Employee\_acc*

| Поле        | Тип     | Обмеження      | Опис                     |
|-------------|---------|----------------|--------------------------|
| acc_id      | int(11) | Auto increment | Унікальний ідентифікатор |
| employee_id | int(11) | Not Null       | ID робітника             |
| Name        | text    | Not Null       | Назва облікового запису  |
| ac_no       | text    | Not Null       | Номер рахунку            |

– Employerpaytbl - таблиця використовується для зберігання інформації про оплачені проекти, якими може керувати роботодавець.

*Таблиця 3.15. Структура таблиці Employerpaytbl*

| Поле           | Тип     | Обмеження      | Опис                      |
|----------------|---------|----------------|---------------------------|
| employerpay_id | int(11) | Auto increment | Унікальний ідентифікатор  |
| p_id           | int(11) | Not Null       | Ідентифікатор проекту     |
| Amount         | int(11) | Not Null       | Кількість робітників      |
| employer_id    | int(11) | Not Null       | Ідентифікатор роботодавця |
| employee_id    | int(11) | Not Null       | ID робітника              |

### 3.4 Опис користувальницької частини додатку

Додаток має 4 типи користувачів:

- Гість

- Адміністратор
- Робітник
- Роботодавець

### Гість

Гостьовий тип створено для незареєстрованих користувачів, які або планують реєстрацію, або бажають ознайомитись з додатком та представленими на ньому пропозиціями. Сторінку для гостьового типу авторизації наведено на рис. 3.5.

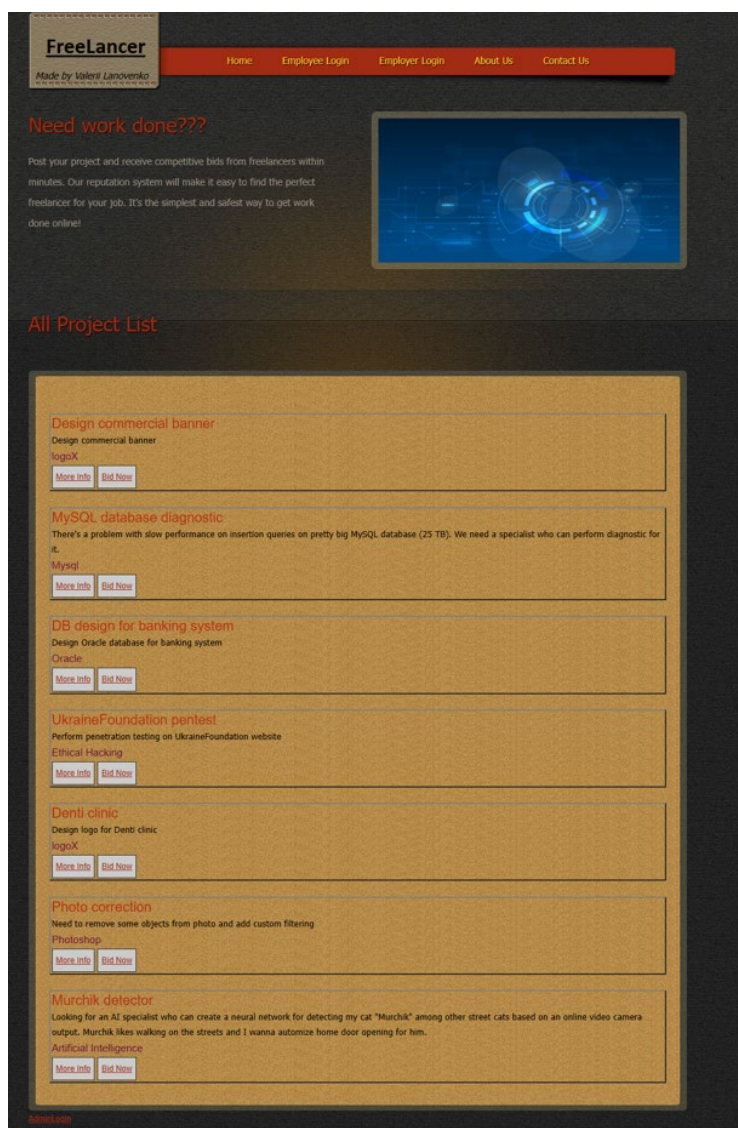


Рис. 3.5 Головний екран додатку для незареєстрованих користувачів

Навігаційне меню містить усі можливі варіанти роботи з додатком для гостю. У змісті головної сторінки для гостьового користувача наведено усі відкриті проєкти.



Якщо користувач не гість, можливо відгукнутись на проєкт при натисненні на «Bid now». В іншому випадку пропонується пройти авторизацію як робітник. Форма авторизації (рис. 3.6) використовується однакова для усіх типів користувачів.

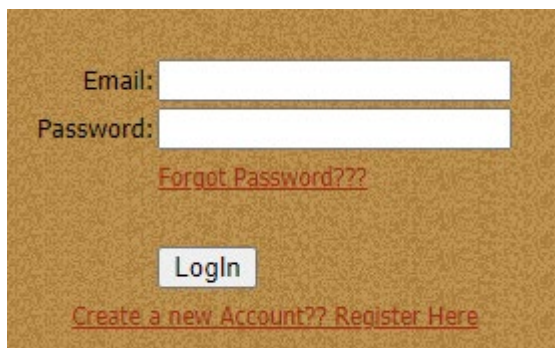
The image shows a login form on a textured brown background. It includes two white input fields for 'Email:' and 'Password:'. Below the password field is a red link 'Forgot Password???'. A grey 'Login' button is centered below the links. At the bottom is another red link 'Create a new Account?? Register Here'.

Рис. 3.6 Форма авторизації

У випадку відсутності користувача в базі він може зареєструватися використовуючи форму реєстрації (рис. 3.7).

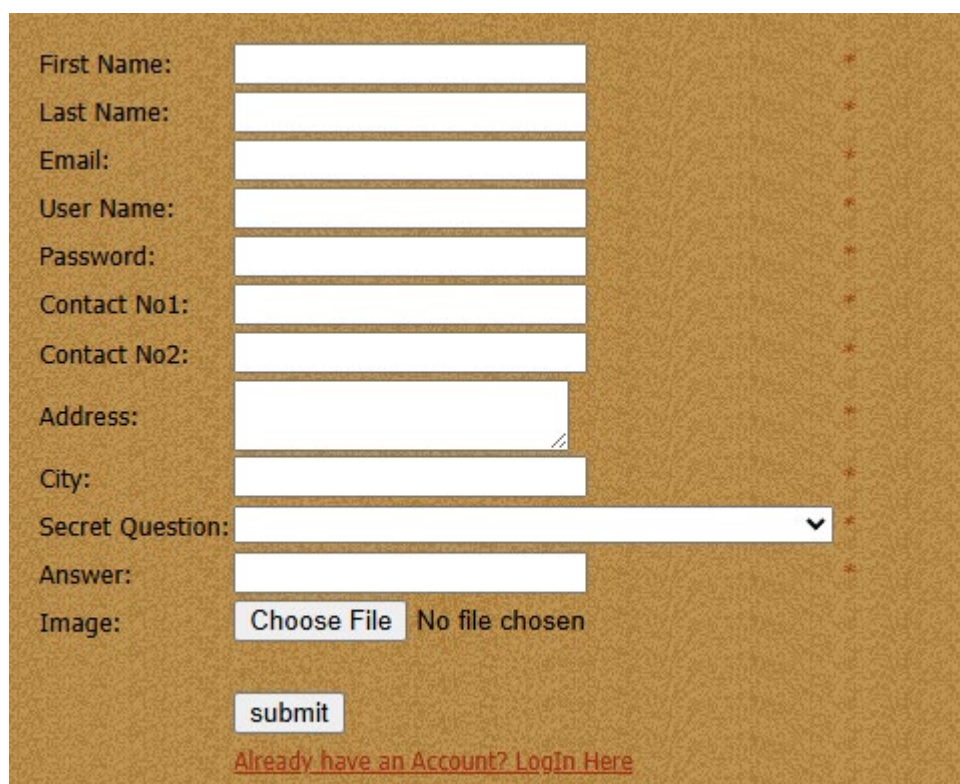
The image shows a registration form on a textured brown background. It contains multiple white input fields for 'First Name:', 'Last Name:', 'Email:', 'User Name:', 'Password:', 'Contact No1:', 'Contact No2:', 'Address:', 'City:', 'Secret Question:', and 'Answer:'. The 'Secret Question:' field is a dropdown menu. To the right of each field is a small red asterisk. Below the 'Answer:' field is a file upload section with a 'Choose File' button and the text 'No file chosen'. A grey 'submit' button is at the bottom. At the very bottom is a red link 'Already have an Account? LogIn Here'.

Рис. 3.7 Форма реєстрації

При натисненні на «More info» біля проєкту здійснюється перехід на інформаційну сторінку проєкту (рис. 3.8). Вона доступна без авторизації.



Рис. 3.8 Інформаційна сторінка проекту

Для гостьового користувача була створена форма зв'язку з адміністратором для донесення зауважень щодо роботи додатку.

Рис. 3.9 Форма зв'язку з адміністратором

А також сторінку «About us» де стисло описано основні напрямки проєктів які присутні в пропозиціях.





Рис. 3.10. Сторінка «About us»

### Адміністратор

Адміністратор має таку саму сторінку перегляду проєктів як гість, але в свою чергу має можливість редагувати системні дані додатку. Під системними даними рахується дані про категорії проєкту, підкатегорії, навички, користувачі та адміністративні повідомлення.

Можливі категорії проєктів редагуються зі спеціальної сторінки, що наведена на рис. 3.11.

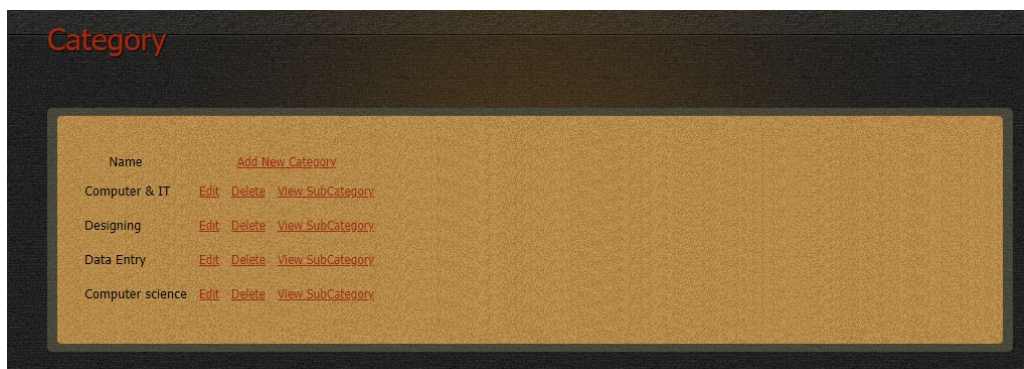


Рис. 3.11 Сторінка редагування категорій проєктів

Категорії залежні за зв'язком one-to-many до підкатегорій. Сторінка редагування підкатегорій проілюстровано на рис. 3.12.

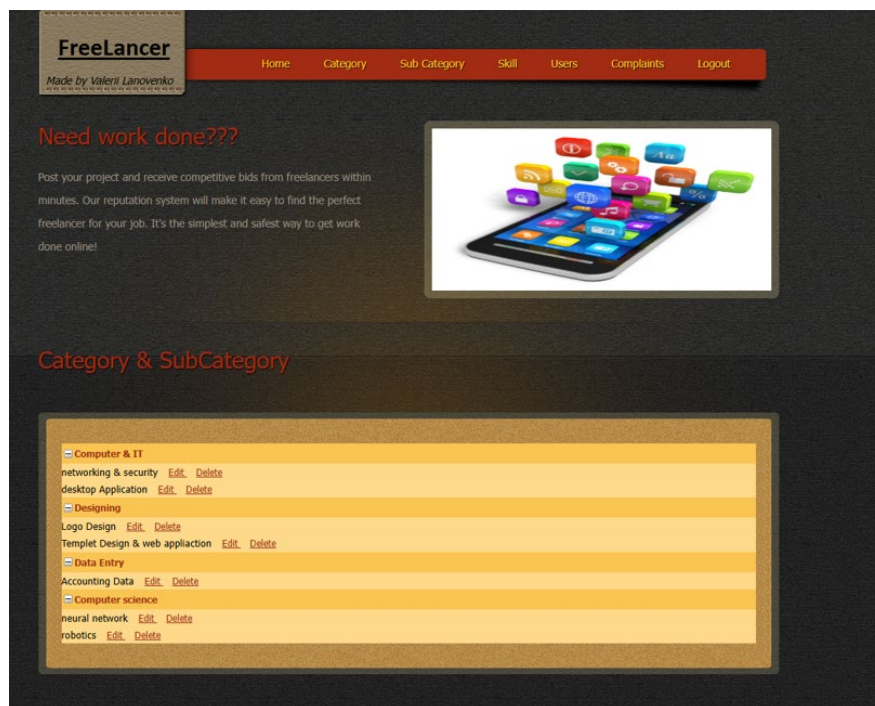


Рис. 3.12. Сторінка редагування підкатегорій

Так само як категорії залежні від підкатегорій, підкатегорії зв'язані з навичками, які необхідні для виконання проєкту. Для їх редагування було створено окрему сторінку (рис. 3.13).



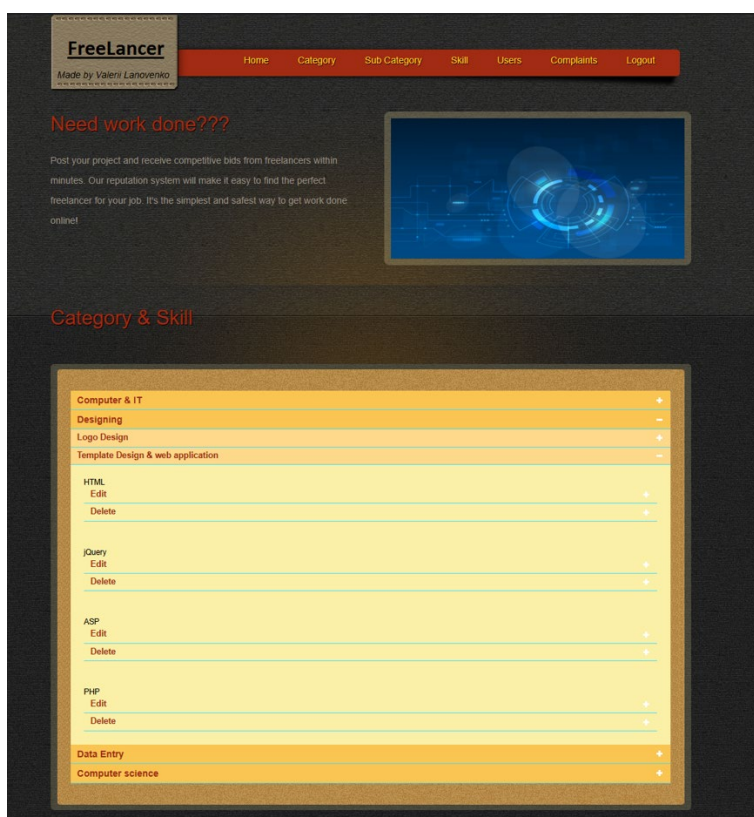


Рис. 3.13. Сторінка редагування навичок

Для перегляду та видалення повідомлень користувачів була створена спеціальна сторінка.

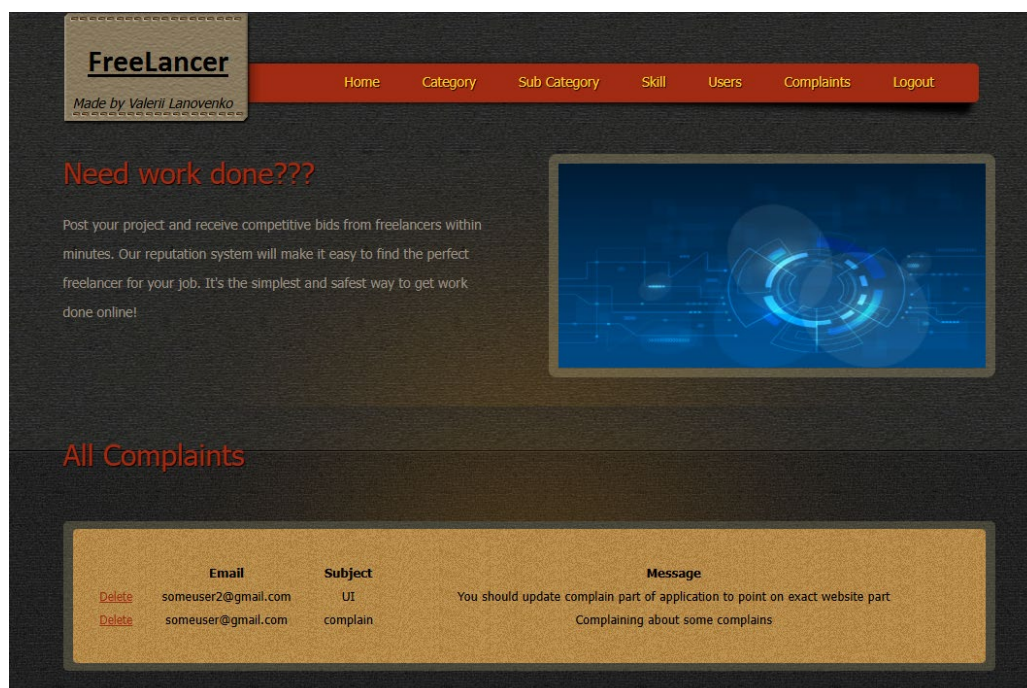
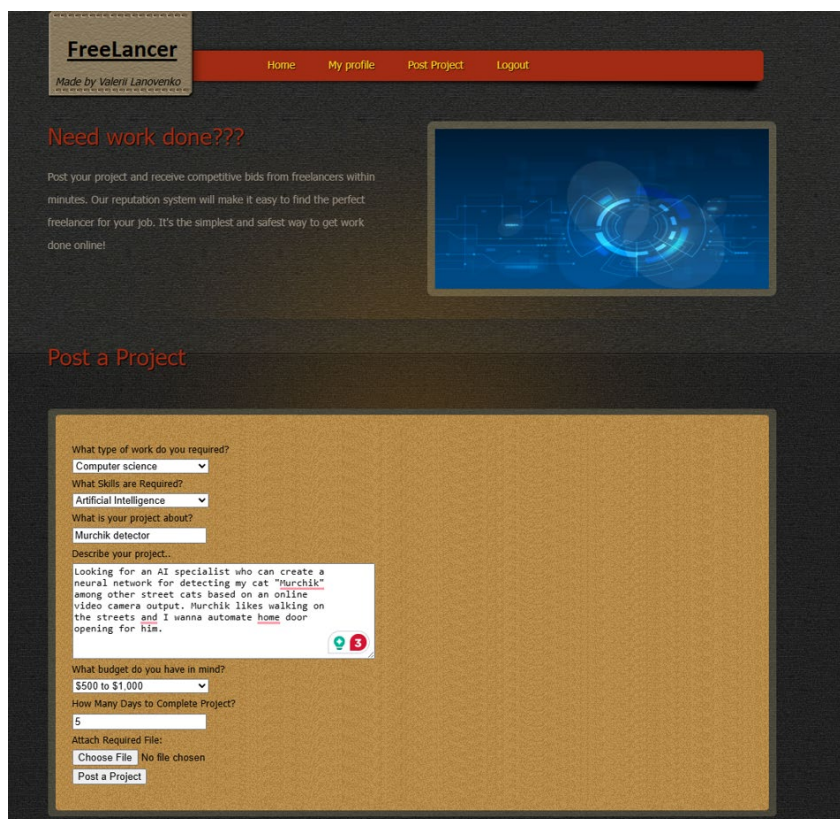


Рис. 3.14 Сторінка з повідомленнями для адміністратора

## Роботодавець

Головна сторінка роботодавця ( див. Додаток А) суттєво відрізняється від попередніх. На верхній частині присутнє фото роботодавця та функція поповнення рахунку. Список проєктів відрізняється групуванням проєктів за їх статусами – закінчені проєкти, проєкти на які виділено бюджет, проєкти за які проведено повну або часткову оплату. В останній групі показуються абсолютно всі проєкти від поточного замовника (роботодавця). Зі списку проєктів можливо проводити оплату роботи виконавця зі списку. Виконавці проєкту обираються на інформаційній сторінці проєкту з поміж робітників, що запропонували свої послуги.

У арсеналі роботодавця знаходиться основний функціонал наповнення додатку – можливість створювати проєкти. Функціонал створення проєктів показано на рис. 3.15.



The screenshot displays the Freelancer website interface. At the top, there is a navigation bar with links: Home, My profile, Post Project, and Logout. Below the navigation bar, a section titled 'Need work done???' encourages users to post projects and receive bids. To the right of this text is a blue graphic with a circular pattern. Below this section is the 'Post a Project' form, which is a large yellow box. The form contains several fields: 'What type of work do you required?' (dropdown menu with 'Computer science' selected), 'What Skills are Required?' (dropdown menu with 'Artificial Intelligence' selected), 'What is your project about?' (text input field with 'Murchik detector' entered), 'Describe your project...' (text area with a sample description about an AI specialist for cat detection), 'What budget do you have in mind?' (dropdown menu with '\$500 to \$1,000' selected), 'How Many Days to Complete Project?' (text input field with '5' entered), and 'Attach Required File:' (with buttons for 'Choose File' and 'No file chosen'). At the bottom of the form is a 'Post a Project' button.

Рис. 3.15 Сторінка створення проєкту

Якщо у роботодавця виникає необхідність оновити дані про себе, він може зробити це на сторінці редагування профілю (рис. 3.16).

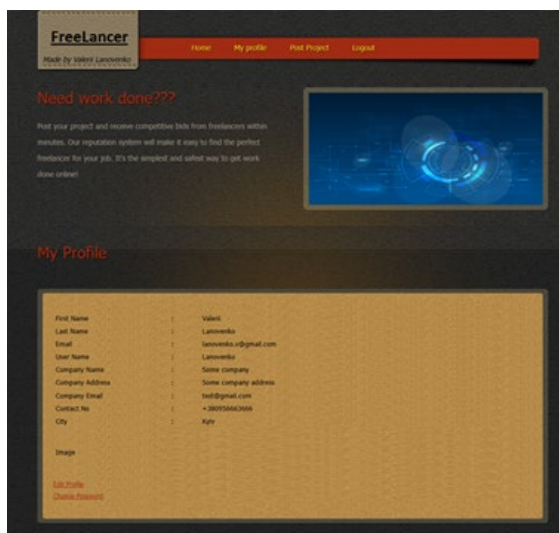


Рис. 3.16 Сторінка редагування профілю роботодавця

### Робітник

Головна сторінка ідентична зі сторінкою роботодавця. Відмінності є лише в відсутності групи проєктів на які виділено бюджет, а також наявності форми заповнення платіжних даних.

Основним функціоналом є відгуки на проєкти замовників. На інформаційній сторінці проєкту (рис. 3.17), на відміну від роботодавця, робітник може залишити заявку на виконання роботи з вказанням термінів та вартості робіт. Усі заявки подані робітником відображаються на спеціальній сторінці заявок (рис. 3.18).

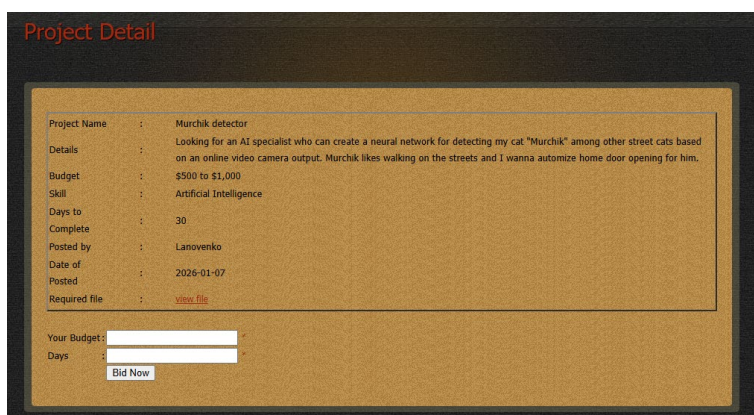


Рис. 3.17 Інформаційна сторінка проєкту для робітника



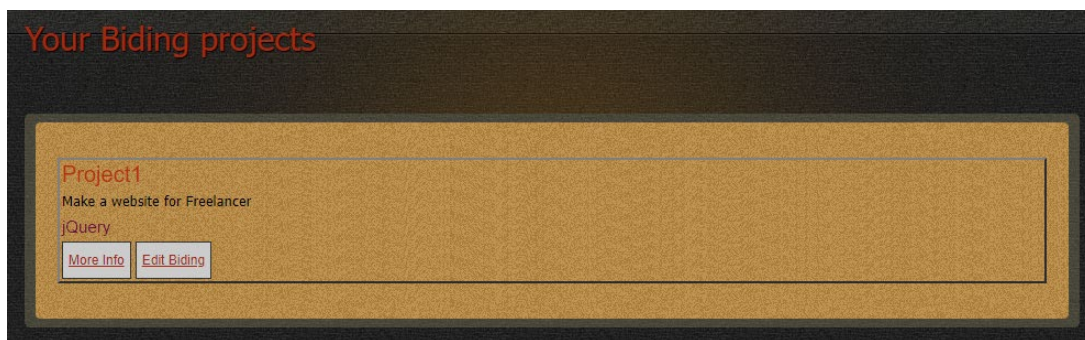


Рис. 3.18. Сторінка заявок робітника

Проте не кожен робітник, може подавати заявку на проведення робіт. Необхідна наявність співпадіння навичок вказаних в проєкті та профілі робітника. Навички робітника можливо редагувати на спеціальній сторінці редагування навичок (рис. 3.19). Контактні дані та фото редагуються на сторінці аналогічній сторінці редагування профілю роботодавця.

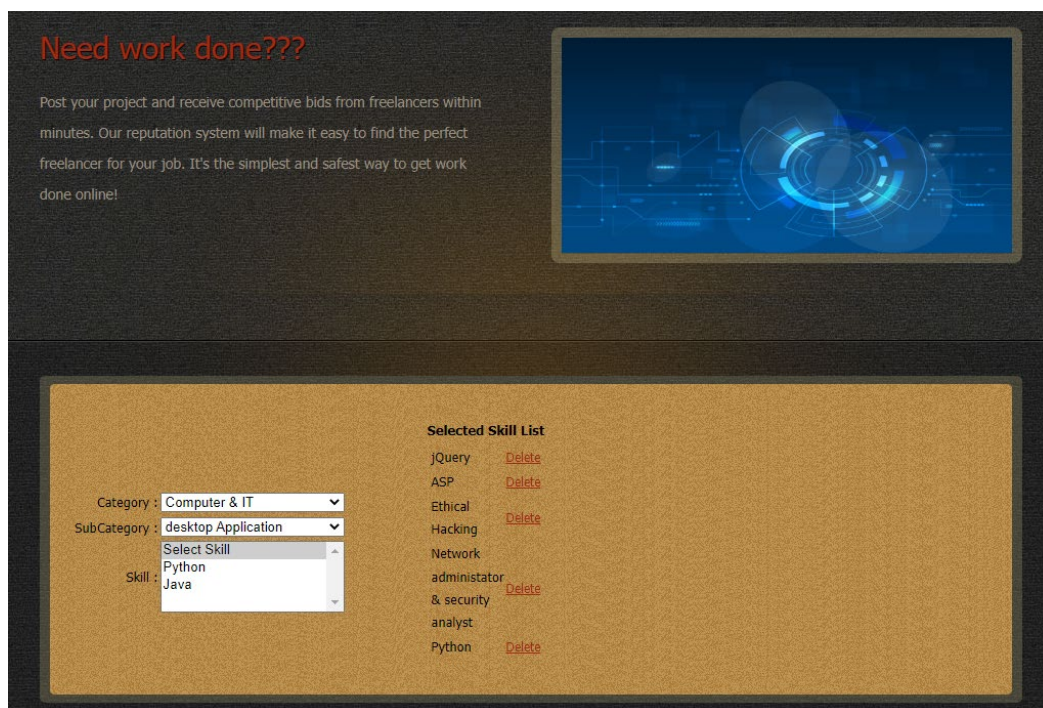


Рис. 3.19. Сторінка редагування навичок робітника

Задля полегшення пошуку пропозицій створено сторінку пошуку проєктів за вказаними в профілі навичками (рис. 3.20). На ній перераховані абсолютно усі проєкти, що підходять робітнику.



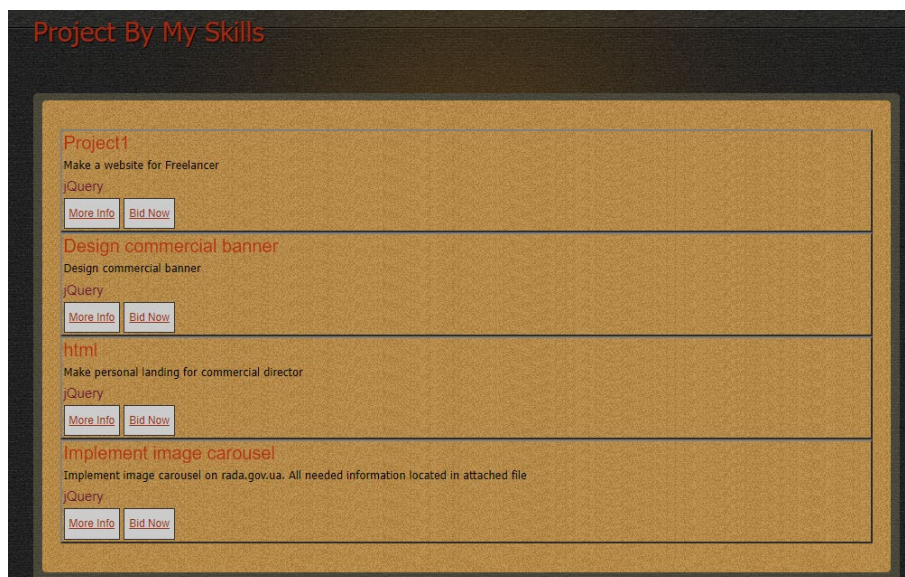


Рис. 3.20 Сторінка відібраних проєктів

### 3.5 Висновки до розділу 3

В ході проведеної роботи розроблено веб-додаток, який було спроектовано та реалізовано фріланс біржі з пошуковою системою.

Структура проєкту проста і добре організована, щоб користувач мав змогу легко користуватися додатком, а розробник з легкістю і мінімальними затратами часу зміг модернізувати під потреби.

Для подальшого розвитку до розробленої системи можна внести наступні поліпшення:

- розробити мобільні версії додатку, щоб надати користувачам можливість бути більш мобільним;
- додати систему трекінгу проєкту для робітників та роботодавців

Розроблена система може бути впроваджена для будь-яких організацій та закладів, які користуються фріланс послугами.

## ВИСНОВКИ

Сьогодні фріланс активно розвивається, зокрема на ринку ІТ-послуг. Існує значна кількість спеціалізованих вебресурсів, що допомагають виконавцям знаходити нові замовлення. Проаналізувавши діяльність таких платформ, їхні функціональні можливості та недоліки, було обґрунтовано доцільність подальших досліджень у цьому напрямі та прийнято рішення про створення власної веб-орієнтованої інформаційної системи для пошуку виконавців ІТ-проектів.

У межах випускної кваліфікаційної роботи проведено огляд і аналіз сучасного стану проблеми, зокрема розглянуто особливості розробки веборієнтованих інформаційних систем та сформовано вимоги до їх функціональних можливостей. Виконано дослідження предметної області, визначено вимоги до проєктованої системи, обрано методи й інструменти її реалізації, а також створено модель інформаційної системи, спроектовано інтерфейс і структуру бази даних.

До основних функцій розробленого програмного продукту належать: реєстрація й авторизація користувачів, розміщення замовлень із можливістю додавання файлів, перегляд завдань зареєстрованими користувачами, пошук робіт за різними критеріями, створення заявок на виконання проєктів, обмін повідомленнями між замовником і виконавцем, формування сповіщень, рейтингу та історії проєктів, а також блокування користувачів.

Система надає виконавцям можливість виконувати замовлення, створювати портфоліо та взаємодіяти з клієнтами. Замовники, у свою чергу, можуть публікувати завдання, обирати виконавців і спілкуватися з ними за допомогою чату. Важливою особливістю сервісу є підтримка різних типів облікових записів — як для індивідуального користування, так і для команд розробників або компаній, що значно розширює функціональні можливості платформи та дозволяє отримувати комплексні послуги з мінімальними затратами часу на пошук фахівців.

Проект має добре структуровану архітектуру, що забезпечує зручність роботи для користувачів та дозволяє розробникам оперативно модернізувати систему відповідно до нових вимог, зокрема створювати мобільні версії або впроваджувати модулі трекінгу проєктів.

Розроблена пошукова інформаційна система може бути використана як відкрита фріланс-платформа, яка спрощує процес пошуку замовлень і комунікацію між клієнтами та виконавцями. Інтерфейс веб-системи є інтуїтивно зрозумілим і надійним, а реалізовані механізми захисту забезпечують доступ до функціоналу лише для авторизованих користувачів.

## ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Adriaan de Jonge, Phil Dutson. jQuery, jQuery UI, and jQuery Mobile: Recipes and Examples – Addison-Wesley Professional, 2012. 400 pages.
2. ASP.NET Core7. URL: <https://learn.microsoft.com/aspnet/core/tutorials/razor-pages/razor-pages-start?view=aspnetcore-7.0&tabs=visual-studio>. (Date of access: 24.12.2022).
3. Browser Support | jQuery. URL: <https://jquery.com/browser-support/>. (Date of access: 24.12.2022).
4. Chris Dawson, Ben Straub. Building Tools with GitHub: Customize Your Workflow – O'Reilly Media, 2016. 302 pages.
5. David Geary, Cay S. Horstmann. Core JavaServer Faces, 3rd Edition. 2011. 656 p.
6. Deepu K Sasidharan, Sendil Kumar N. Full Stack Development with JHipster – Packt Publishing, 2018. 380 pages.
7. Eric Elliott. Programming JavaScript Applications: Robust Web Architecture with Node, HTML5, and Modern JS Libraries – O'Reilly Media, 2014. 254 pages
8. Free-lance.ua. URL: [free-lance.ua](https://free-lance.ua). (Date of access: 24.12.2022).
9. Freelancehunt. URL: [freelancehunt.com](https://freelancehunt.com). (Date of access: 24.12.2022).
10. Jason Price, Oracle Database 11g SQL. McGraw-Hill Osborne Media, 2007. 656с.
11. jQuery Overview. URL: <https://www.tutorialspoint.com/jquery/jquery-overview.htm>. (Date of access: 24.12.2022).
12. Kabanchik.ua. URL: [kabanchik.ua](https://kabanchik.ua). (Date of access: 24.12.2022).
13. Matt Frisbie. Professional JavaScript for Web Developers. 4th Edition. Wrox. 2019. 1200p.
14. MC.today. С. Кіт. Найкращі українські та зарубіжні біржі фрілансу (07 Dec 2022). URL: <https://mc.today/birzhi-frilansa/>. (Date of access: 24.12.2022). <https://mc.today/birzhi-frilansa/>
15. MySQL Tutorial. URL: <https://www.w3schools.com/MySQL/default.asp>.

(Date of access: 24.12.2022).

16. Philip A. Koryaka. CGI - Simple Common Gateway Interface Class URL: <http://www.xserver.ru/computer/langprogr/cgi/>. (Date of access: 24.12.2022).

17. UML моделювання. URL: <https://habr.com>. (Date of access: 24.12.2022).

18. UpWork. URL: <https://www.upwork.com/>. (Date of access: 24.12.2022).

19. Weblancer.net. URL: [weblancer.net](https://www.weblancer.net). (Date of access: 24.12.2022).

20. Why Quill - Quill Rich Text Editor. URL: <https://quilljs.com/guides/why-quill>. (Date of access: 24.12.2022).

21. Адміністрування. URL: <https://workspace.google.com/intl/uk/products/admin/>. (Date of access: 24.12.2022).

22. Введение в поисковую оптимизацию. URL: <https://developers.google.com/search/docs/fundamentals/seo-starter-guide?hl>. (Date of access: 24.12.2022).

23. Джон Дакетт - HTML и CSS. Разработка и дизайн веб-сайтов [пер. С англ. М.А. Райтмана]. М. : Эскиммо, 2013. 480 с.

24. Доценко С. І. Людино-машинний інтерфейс: навч.посібник. Харків: УкрДУЗТ, 2022. 135 с.

25. Дронов В. А. HTML5, CSS3 и Web 2.0. Разработка современных Web-сайтов. СПб. : БХВ, 2011. 416 с.

26. Кевин Мельтцер Разработка CGI-приложений на Perl. Вильямс. 2011. 400с.

27. Кешування та продуктивність веб-додатків. URL: <https://habr.com/company/ruvds/blog/350310/>. (Date of access: 24.12.2022).

28. Котеров Д.В., Симдянов И.В. PHP 8. СПб.: БХВ, 2021. – 850с.

29. Основы CSS3. URL: <https://metanit.com/web/html5/>. (Date of access: 24.12.2022).

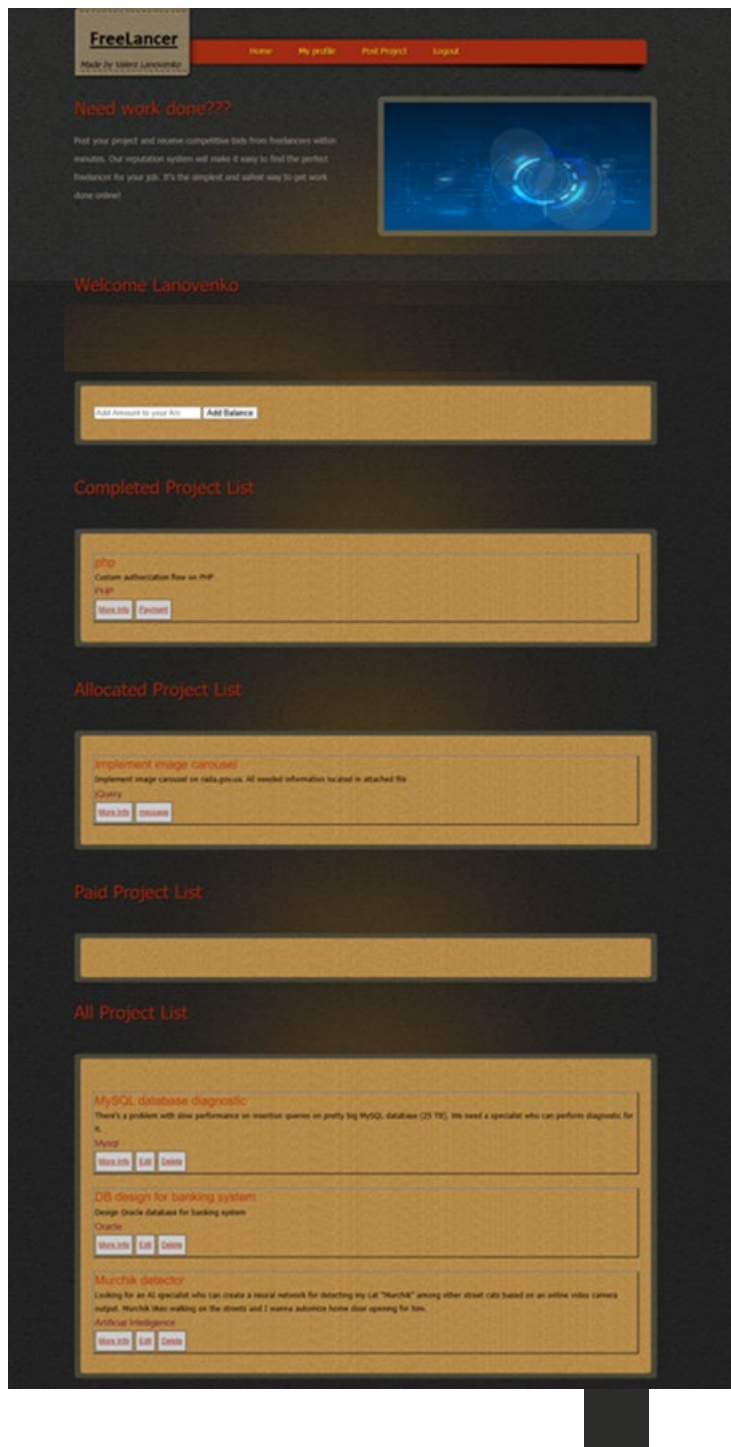
30. Пол МакФедрис. Web Design Playground (2019). URL: [https://vk.com/wall-54530371\\_296845](https://vk.com/wall-54530371_296845). (Date of access: 24.12.2022).

31. Посібник з HTML та CSS3. URL: <https://metanit.com/web/html5/>. (Date of access: 24.12.2022).

32. Учебник HTML с CSS. URL:[https://www.schoolsw3.com/html/html\\_css.php](https://www.schoolsw3.com/html/html_css.php). (Date of access: 24.12.2022).
33. Хеник Б. HTML и CSS: путь к совершенству. СПб. : Питер, 2011. 336 с.
34. Язык программирования Python. URL:<https://metanit.com/python/>. (Date of access: 24.12.2022).

## ДОДАТКИ

### Додаток А. Головна сторінка роботодавця



### Додаток Б. Код employee\_home.php

```
<?php
session_start();
ob_start();
```

```

include("header.php");
include("Config.php");
$eid = $_SESSION['employer_id'];
$em = $_SESSION['em'];
$sql = "select * from employer_regitbl where em='$em'";
$res = mysql_query($sql);
$row = mysql_fetch_array($res);
if (isset($_POST['submit'])) {
    mysql_query("update employer_regitbl set balance='" .
$_POST['addbal'] . "'where employer_id=" . $eid) or
die("Error");
}
?>
<style type="text/css">
    .link {
        font-family: Arial, Helvetica, Sans-serif;
        font-size: 10px;
        display: inline-block;
        border: solid 1px #333;
        background: #CCC;
        color: #333;
        text-decoration: none;
        padding: 5px;
    }
    .link:hover {
        border: solid 1px #CCC;
        background: #333;
        color: #CCC;
    }
</style>
<div id="tooplate_middle">
    <div id="mid_title">Welcome <?php echo $_SESSION['unm'];
?></div>
    
    <div class="cleaner"></div>
</div> <!-- end of middle -->
<div id="tooplate_main_top"></div>
<div id="tooplate_main">
    <div id="gallery">
        <form method="POST">
            <table>
                <tr>

```



```

                <td><input type="text" name="addbal"
placeholder="Add Amount to your A/c"></td>
                <td><input type="submit"
name="submit" value="Add Balance"></td>
            </tr>
        </table>
    </form>
    <div class="cleaner"></div>
</div>
<div class="cleaner"></div>
</div> <!-- end of main -->
<div id="tooplate_main_bot"></div>
<div id="tooplate_middle">
    <div id="mid_title"></div>
    <div id="mid_title">Completed Project List</div>
</div> <!-- end of middle -->
<div id="tooplate_main_top"></div>
<div id="tooplate_main">
    <div id="gallery">
        <?php
            $result1 = mysql_query("select * from completeprj
where employer_id=" . $eid, $con) or die(mysql_error());
            if (mysql_num_rows($result1) > 0) {
                while ($r1 = mysql_fetch_array($result1)) {
                    $result2 = mysql_query("select * from
prjtbl where p_id=" . $r1[1], $con) or die("Error in Select
Query2");
                    if (mysql_num_rows($result2) > 0) {
                        while ($r2 =
mysql_fetch_array($result2)) {
                            ?>
                                <table width="100%" border="2">
                                    <tr>
                                        <td align="left"
style="font-family:Arial;font-size:20px;color:#B6300E">
                                            <?php echo $r2[2];
                                        ?></td>
                                    <tr>
                                        <td align="left"><?php
echo $r2[5]; ?></td>
                                    </tr>
                                </table>
                            <?php

```

```

                                $result10 =
mysql_query("select * from skilltbl where skill_id=" .
$r2[4]);
                                if
(mysql_num_rows($result10) > 0) {
                                while ($r10 =
mysql_fetch_array($result10)) {
                                ?>
                                <tr>
                                <td
align="left" style="font-family:Arial;font-
size:15px;color:#6E103C"><?php echo $r10[1]; ?></td>
                                </tr>
                                <?php }
                                } ?>
                                <tr>
                                <td><a
href="employercompleteprjview.php?pid=<?php echo $r2[0];
?>& cid=<?php echo $r1[0]; ?>" class='link'>More
Info</a>
                                <a
href="employerpayment.php?pid=<?php echo $r2[0]; ?>&
cid=<?php echo $r1[0]; ?>" class='link'>Payment</a>
                                </td>
                                </tr>
                                </table>
                                <?php
                                }
                                }
                                }
                                ?>
                                <div class="cleaner"></div>
                                </div>
                                <div class="cleaner"></div>
</div> <!-- end of main -->
<div id="tooplate_main_bot"></div>
<div id="tooplate_middle">
    <div id="mid_title"></div>
    <div id="mid_title">Allocated Project List</div>
</div> <!-- end of middle -->
<div id="tooplate_main_top"></div>
<div id="tooplate_main">
    <div id="gallery">

```

```

        <?php
            $result11 = mysql_query("select * from
prj_allocated where employer_id=" . $eid, $con) or
die(mysql_error());
            if (mysql_num_rows($result11) > 0) {
                while ($r11 = mysql_fetch_array($result11)) {
                    $result12 = mysql_query("select * from
prjtbl where p_id=" . $r11[1], $con) or die("Error in Select
Query2");
                    if (mysql_num_rows($result12) > 0) {
                        while ($r12 =
mysql_fetch_array($result12)) {
                            ?>
                                <table width="100%" border="2">
                                    <tr>
                                        <td align="left"
style="font-family:Arial;font-size:20px;color:#B6300E">
                                            <?php echo
$r12[2]; ?></td>
                                                <tr>
                                                    <td align="left"><?php
echo $r12[5]; ?></td>
                                                        </tr>
                                                            <?php
$result1 =
mysql_query("select * from skilltbl where skill_id=" .
$r12[4]);
                                                                if (mysql_num_rows($result1)
> 0) {
                                                                    while ($r1 =
mysql_fetch_array($result1)) {
                                                                        ?>
                                                                            <tr>
                                                                                <td
align="left" style="font-family:Arial;font-
size:15px;color:#6E103C"><?php echo $r1[1]; ?></td>
                                                                                    </tr>
                                                                                        <?php }
                                                                                          } ?>
                                                                                              <tr>
                                                                                                  <td><a
href="prj_allocate.php?pid=<?php echo $r12[0]; ?>"
class='link'>More Info</a>

```

```

                                <a
href="message.php?pid=<?php echo $r12[0]; ?>"
class='link'>message</a>
                                </td>
                                </tr>
                                </table>

        <?php
                                }
                                }
                                }
        }
        ?>
        <div class="cleaner"></div>
    </div>
    <div class="cleaner"></div>
</div> <!-- end of main -->
<div id="tooplate_main_bot"></div>
<div id="tooplate_middle">
    <div id="mid_title"></div>
    <div id="mid_title">Paid Project List</div>
</div> <!-- end of middle -->
<div id="tooplate_main_top"></div>
<div id="tooplate_main">
    <div id="gallery">
        <?php
            $result10 = mysql_query("select employerpay_id,p_id
from employerpaytbl where employer_id=" . $eid, $con) or
die("Error in Select Query1");
            if (mysql_num_rows($result10) > 0) {
                while ($r10 = mysql_fetch_array($result10)) {
                    $result20 = mysql_query("select * from
prjtbl where p_id=" . $r10[1], $con) or die("Error in Select
Query2");
                    if (mysql_num_rows($result20) > 0) {
                        while ($r20 =
mysql_fetch_array($result20)) {
                            ?>
                                <table width="100%" border="2">
                                    <tr>
                                        <td align="left"
style="font-family:Arial;font-size:20px;color:#B6300E">
                                            <?php echo
$r20[2]; ?></td>
                                        <tr>

```

```

                                <td align="left"><?php
echo $r20[5]; ?></td>
                                </tr>
                                <?php
                                $result1 =
mysql_query("select * from skilltbl where skill_id=" .
$r20[4]);
                                if (mysql_num_rows($result1)
> 0) {
                                while ($r1 =
mysql_fetch_array($result1)) {
                                ?>
                                <tr>
                                <td
align="left" style="font-family:Arial;font-
size:15px;color:#6E103C"><?php echo $r1[1]; ?></td>
                                </tr>
                                <?php }
                                } ?>
                                <tr>
                                <td><a
href="employercompleteprjview.php?pid=<?php echo $r20[0];
?>&amp; cid=<?php echo $r10[0]; ?>" class='link'>More
Info</a>
                                </td>
                                </tr>
                                </table>
                                <?php
                                }
                                }
                                }
                                }
                                ?>
                                <div class="cleaner"></div>
                                </div>
                                <div class="cleaner"></div>
</div> <!-- end of main -->
<div id="tooplate_main_bot"></div>
<div id="tooplate_middle">
    <div id="mid_title">All Project List</div>
    <div class="cleaner"></div>
</div>
<!-- end of middle -->
<div id="tooplate_main_top"></div>

```

```

<div id="tooplate_main">
    <div id="gallery">
        <?php
            $result = mysql_query("select * from prjtbl WHERE
p_id NOT IN (select p_id from completeprj) AND p_id NOT IN
(select p_id from prj_allocated) AND p_id NOT IN (select
p_id from employerpaytbl) AND employer_id=" . $eid, $con) or
die("Error in Select Query");
            if (mysql_num_rows($result) > 0) {
                while ($r = mysql_fetch_array($result)) {
                    ?> <br />
                        <table width="100%" border="2">
                            <tr>
                                <td align="left" style="font-
family:Arial;font-size:20px;color:#B6300E"><?php echo $r[2];
?></td>
                                </tr>
                                <tr>
                                    <td align="left"><?php echo
$r[5]; ?></td>
                                    </tr>
                                <?php
                                    $result1 = mysql_query("select * from
skilltbl where skill_id=" . $r[4]);
                                    if (mysql_num_rows($result1) > 0) {
                                        while ($r1 =
mysql_fetch_array($result1)) {
                                            ?>
                                                <tr>
                                                    <td align="left"
style="font-family:Arial;font-size:15px;color:#6E103C"><?php
echo $r1[1]; ?></td>
                                                    </tr>
                                                <?php }
                                                } ?>
                                                <tr>
                                                    <td><a
href="employerviewbid.php?pid=<?php echo $r[0]; ?>"
class='link'>More Info</a>
                                                    <a
href="employereditprj.php?pid=<?php echo $r[0]; ?>"
class='link'>Edit</a>

```

```
                                <a  
href="employerdelprj.php?pid=<?php echo $r[0]; ?>"  
class='link'>Delete</a>  
                                </td>  
                            </tr>  
                        </table>  
        <?php    }  
    }  
    ?>  
    <?php  
include("footer.php");  
    ?>
```