КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
імені ТАРАСА ШЕВЧЕНКА
Факультет інформаційних технологій
Кафедра прикладних інформаційних систем

122 «Комп'ютерні науки»
(шифр і назва спеціальності)

«Прикладне програмування»
(назва освітньої програми)

Кваліфікаційна робота бакалавра

на тему: «Веб-застосунок пошуку робочих місць з IT спеціальностей»

Виконала _____
(Підпис)
Ільченко Єлизавета Ігорівна
(прізвище, ім'я, по батькові)

Керівник Ващіліна Олена Валеріївна
(прізвище, ім'я, по батькові)

_____
(Резолюція «До захисту»)

Попередній захист:

_____
(Висновок: "До захисту в екзаменаційній комісії ")
Завідувач кафедри _____ Плескач В.Л.
(Підпис) (Прізвище, ініціали)
(Дата)

Київ – 2021

КАЛЕНДАРНИЙ ПЛАН

Номер	Назва етапів кваліфікаційної роботи бакалавра	Термін виконання етапів кваліфікаційної роботи бакалавра	Відмітка про виконання
1.	Вибір теми та наукового керівника кваліфікаційної роботи бакалавра	26.10.2020	Виконано
2.	Видача завдання кваліфікаційної роботи бакалавра	23.11.2020	Виконано
3.	Настановча групова співбесіда з питань кваліфікаційної роботи бакалавра	01.12.2020	Виконано
4.	Затвердження плану кваліфікаційної роботи бакалавра	18.02.2021	Виконано
5.	Підбір та вивчення літературних та інших джерел з теми дослідження	25.02.2021	Виконано
6.	Підготовка і подання науковому керівнику першого варіанту I розділу роботи	05.03.2021	Виконано
7.	Підготовка і подання науковому керівнику першого варіанту II розділу роботи	09.04.2021	Виконано
8.	Підготовка і подання науковому керівнику першого варіанту III розділу роботи	07.05.2021	Виконано
9.	Подання роботи у першому варіанті	11.05.2021	Виконано
10.	Оформлення пояснювальної записки кваліфікаційної роботи бакалавра	12.05.2021	Виконано
11.	Подання кваліфікаційної роботи бакалавра на попередній захист	24.05.2021	Виконано
12.	Врахування зауважень керівника і подання роботи в остаточному варіанті (з відповідним висновком про допуск) на кафедру	28.05.2021	Виконано
13.	Затвердження роботи в цілому (підготовка письмового відгуку керівника, письмова рецензія на бакалаврської роботу)	11.06.2021	
14.	Захист кваліфікаційної роботи бакалавра	22.06.2021	

Здобувач вищої освіти _____

(підпис)

Керівник _____

(підпис)

ВІДОМІСТЬ ДИПЛОМНОЇ РОБОТИ

Зміст пояснювальної записки (перелік питань під час дослідження)

Складові частини дипломної роботи	Обсяг, арк.
Титульний аркуш	1 ст.
Завдання до дипломної роботи (календарний план проекту)	1 ст.
Відомість дипломної роботи	1 ст.
Пояснювальна записка до дипломної роботи	1 ст.
Реферат (анотація)	1 ст.
Abstract (анотація іноземною мовою-англійською)	1 ст.
Зміст	2 ст.
Вступ	1 ст.
Розділ 1	13 ст.
Розділ 2	12 ст.
Розділ 3	15 ст.
Висновки	1 ст.
Список використаних джерел	5 ст.
Додатки	4 ст.

<table>
<tr><td></td><td></td><td></td><td></td><td colspan="3" rowspan="2">ДП ХХХХ 00.000.00</td></tr>
<tr><td></td><td>ПІБ</td><td>Підп.</td><td>Дата</td></tr>
<tr><td>Розробн.</td><td>Ільченко Є.І.</td><td></td><td></td><td rowspan="4">Відомість дипломної роботи</td><td>Лист</td><td>Листів</td></tr>
<tr><td>Керівн.</td><td>Ващіліна О. В.</td><td></td><td></td><td></td><td></td></tr>
<tr><td>Н/контр.</td><td>Макаренко С. А.</td><td></td><td></td><td colspan="2" rowspan="2"></td></tr>
<tr><td>Зав.каф.</td><td>Плескач В. Л.</td><td></td><td></td></tr>
</table>

АНОТАЦІЯ

Дипломна робота: 56 с., 31 рис., 66 джерел, 4 дод.

Тема роботи: розробка веб-застосунку для пошуку робочих місць з IT спеціальностей.

Метою роботи є створення веб-застосунку для пошуку роботи з IT спеціальності, який надасть можливість ефективної комунікації між роботодавцями та шукачами роботи.

Для досягнення цієї мети необхідно було виконати наступні **завдання**:

1. Дослідити можливі підходи до розроблення сучасних веб-застосунків;
2. Провести аналіз програмно-технологічних рішень розроблення веб-застосунків;
3. Реалізувати та впровадити веб-застосунок пошуку робочих місць з IT спеціальностей (JavaScript).

Об'єкт дослідження: процеси організації працевлаштування IT-фахівців за допомогою інформаційних технологій.

Предмет дослідження: програмно-технічні, організаційні засади, принципи, підходи до побудови веб-застосунку, призначеного для організації ефективної комунікації між роботодавцями та шукачами роботи з IT спеціальності.

Методи дослідження: дослідження теоретичних аспектів створення пошукових систем, ретельний аналіз систем, що застосовувався при вивченні прикладів сучасних методів побудови веб-застосунків, розробка та побудова власного веб-застосунку для пошуку робочих місць, метод порівняння, що застосовано для аналізу наявних ресурсів та програмних систем з пошуку роботи.

Ключові слова: JavaScript, веб-застосунок, бази даних, сайт, пошук, HTML, робота.

ABSTRACT

Thesis: 56 pages, 31 figures, 66 sources, 4 appendices.

This thesis: development of a web application for job search in IT specialties.

The aim of the work is to create a web application for job search in IT, which will provide effective communication between employers and job seekers.

To achieve this goal it was necessary to perform the following **tasks**:

1. Investigate possible approaches to the development of modern web applications;

2. To analyze software and technology solutions for web application development;

3. Implement and implement a web application for job search in IT specialties (JavaScript).

Object of research: processes of organization of employment of IT-specialists with the help of information technologies.

Subject of research: software and hardware, organizational principles, principles, approaches to building a web application designed to organize effective communication between employers and job seekers in IT specialties.

Research methods: research of theoretical aspects of search engine creation, careful analysis of systems used in studying examples of modern methods of building web applications, development and construction of own web application for job search, comparison method used for analysis of available resources and software systems job search.

Keywords: JavaScript, web application, databases, site, search, HTML, work.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМ. ПОЗНАЧЕНЬ, ТЕРМІНІВ

Інтерфейс (користувача) - це простір, де відбувається взаємодія між людьми та інформаційними системами.

Фреймворк (в програмуванні) - програмна платформа, яка визначає структуру програмної системи; програмне забезпечення, що полегшує розробку і об'єднання різних компонентів великого програмного проекту.

Бібліотека - збірник підпрограм або об'єктів, які використовуються для розробки програмного забезпечення.

Макет сайту - графічна схема із зазначенням кольорів, відступів і інших параметрів сторінки, що найбільш точно відображає роботу майбутнього сайту в інтерактивній формі.

ECMAScript - це мова сценаріїв, яка становить основу JavaScript.

Node.js - це середовище виконання з відкритим вихідним кодом, міжплатформне та внутрішнє середовище виконання JavaScript, що призначене для створення масштабованих мережевих додатків.

Ajax (Asynchronous Javascript And Xml) - це набір технологій веб-розробки, що використовують безліч веб-технологій на стороні клієнта для створення асинхронних веб-додатків та дозволяють звернення до сервера без перезавантаження сторінки.

Comet - будь-яка модель роботи веб-додатку, при якій постійне HTTP-з'єднання дозволяє веб-серверу відправляти дані браузеру без додаткового запиту з боку браузера.

JavaScript (JS) - високорівнева мова програмування, що відповідає специфікації ECMAScript та найбільш відома як мова сценаріїв веб-сторінок.

ВСТУП

Застосування різноманітних сайтів та веб-застосунків в мережі Інтернет нині майже не має меж. Такі електронні ресурси користуються популярністю у більшості сферах діяльності людини: робота, навчання, розваги, спілкування, торгівля, творчість тощо. Кількість існуючих сайтів зараз впевнено зростає до позначки 2 мільярдів. Саме тому, вміння працювати з веб-ресурсами та створювати нові веб-застосунки важко переоцінити.

Актуальність обраної теми зумовлена тим, що кожна людина рано чи пізно стикається з проблемою власного працевлаштування. Створення програмної системи пошуку роботи є значним внеском у цей процес, завдяки автоматизації якого значно покращується та спрощується процедура знаходження роботи для користувачів та пошук робітників для роботодавців з різноманітних підприємств.

У цій роботі було виконано наступні **завдання**:

- ретельно досліджено теоретичні основи побудови веб-застосунків;
- були проаналізовані існуючі системи з пошуку роботи;
- реалізовано веб-застосунок пошуку робочих місць з IT спеціальностей.

Об’єкт дослідження: процеси організації працевлаштування IT-фахівців за допомогою інформаційних технологій.

Предмет дослідження: програмно-технічні, організаційні засади, принципи, підходи до побудови веб-застосунку, призначеного для організації ефективної комунікації між роботодавцями та шукачами роботи з IT спеціальності.

До практичного значення одержаних результатів програмний продукт, що дозволяє користувачам швидко та зручно знаходити роботу або працівників у додатку в режимі онлайн. Розроблений веб-застосунок враховує кращі практики, виявлені під час аналізу функціоналу та зовнішнього вигляду подібних систем.

РОЗДІЛ 1. ОГЛЯД СУЧАСНИХ МЕТОДІВ РОЗРОБКИ ВЕБ-ЗАСТОСУНКІВ

1.1. Основні підходи до створення сайтів

Веб-розробкою вважається створення веб-сайтів та застосунків в мережі Інтернет. Процес роботи над такими продуктами поділяється на так звані “Front-end” та “Back-end” - візуальну та програмну частини відповідно.

Front-end - це все те, що бачить користувач при відкритті веб-сторінки і з чим він може взаємодіяти та безпосередньо контактувати. Ця частина включає в себе відображення функціональних завдань, призначених для користувача інтерфейсу, що виконуються на стороні клієнта, а також обробку запитів користувачів.

Back-end - програмно-апаратна частина сервісу. Робота цієї частини відбувається на стороні серверу, що отримує запити від елементів Front-end та працює над їх реалізацією. Реалізація частини включає в себе використовування таких мов програмування як PHP, Java, Python, JavaScript та інш.

Основними компонентами веб-розробки є HTML - мова сайтової розмітки, CSS - стилі сторінок та JavaScript, що реалізує різноманітні функції та всю динамічну складову застосунків.

Крім того, для покращення роботи над певним застосунком, розробники зазвичай використовують різноманітні допоміжні інструменти. Одними з таких інструментів є так звані фреймворки - програмне забезпечення, що полегшує розробку та об’єднання великої кількості компонентів застосунку в єдину систему. На відміну від бібліотек, фреймворки не лише надають доступ до певного зручного функціоналу, але й впливають на структуру проекту в цілому.

Для створення сучасних веб-застосунків за допомогою JavaScript розробники використовують такі відомі фреймворки та бібліотеки як Angular, Vue.js, React, JQuery та інш. Більш детально це питання буде розглянуто у розділі 1.4. “Інструменти JavaScript”.

1.2. Методи розроблення дизайну та макету сайту

Важливим етапом створення застосунку є його проектування - планування від ідеї до макету. Загалом після того, як ми визначилися з основною метою створення сайту, подальшу розборку можна умовно поділити на такі етапи:

- визначення всіх основних вимог до сайту: функціонал, кількість та вміст сторінок;
- розробка макетів макетів окремих сторінок у вигляді схем (рис. 1.1);
- планування дизайну, стилю оформлення сторінок, палітри кольорів;
- створення макету сайту з HTML сторінками та CSS стилями;
- впровадження функціональної частини.

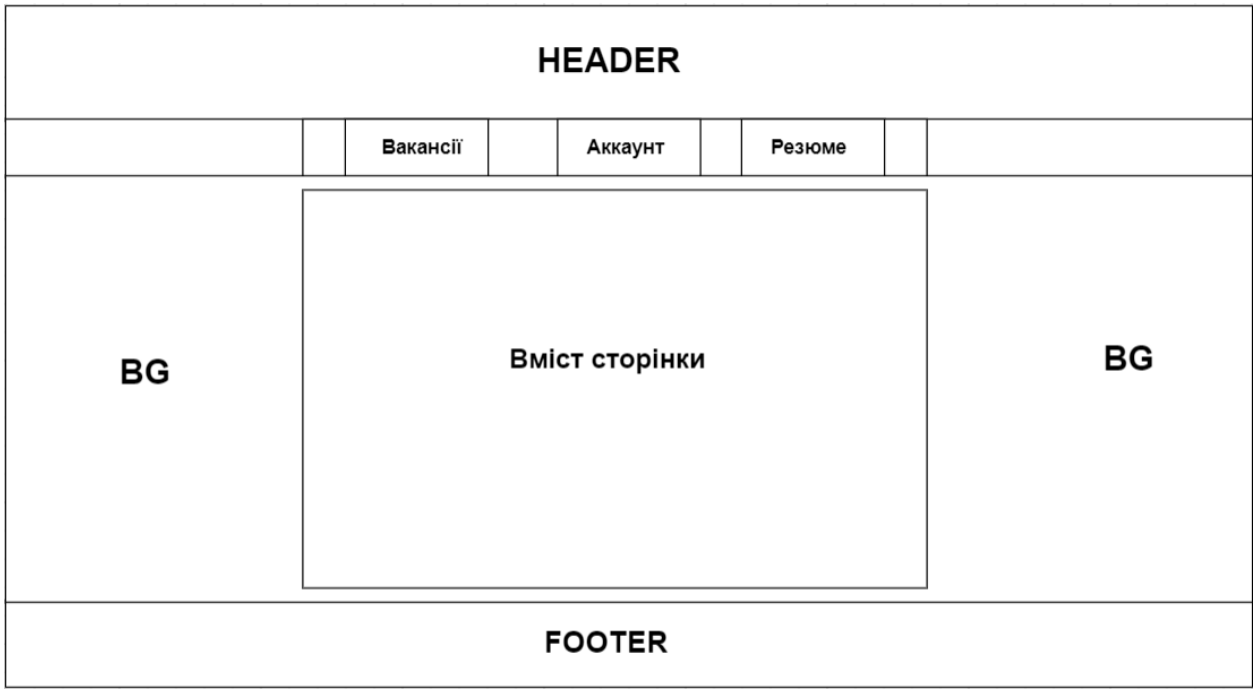


Рисунок 1.1 – Схема сторінки користувача

Існує багато інструментів, що допомагають розробляти дизайн та макет сайту. Наприклад, для створення дизайну часто використовують такі програмні засоби як Figma, Sketch, Adobe Photoshop, Adobe XD. Завдяки ним зручно створювати зовнішній вигляд сторінок, всі присутні на них елементи, такі як зображення чи функціональні частини. У засобі Figma [7] завдяки функціям анімації можна навіть розробити переходи з однієї сторінки на іншу, тим самим демонструючи принцип роботи та основний зовнішній функціонал сайту.

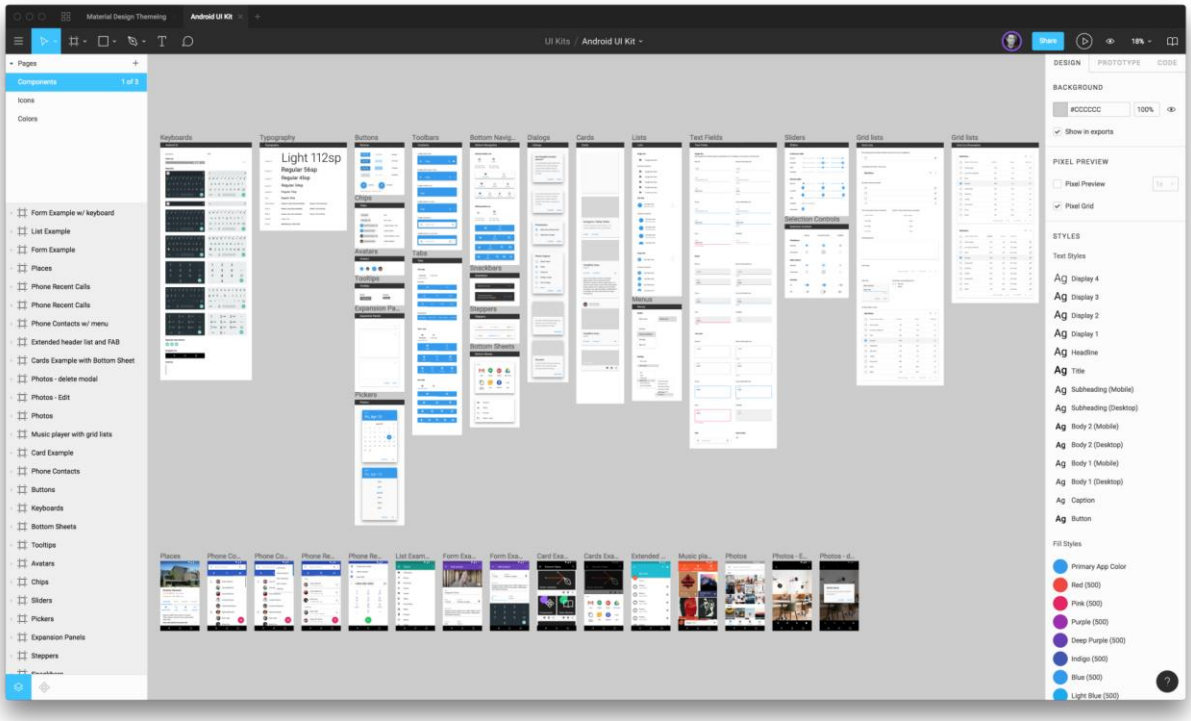


Рисунок 1.2 – Приклад роботи у Figma

Окрім макетів, ми також можемо створювати прототипи сайтів засобами програм. Наприклад, це можливо за допомогою Axure, де розробник створює сторінки з основними готовими функціональними елементами, такими як кнопки чи поля форм, та на виході отримує HTML та CSS коди, з якими потім зручно працювати далі.

Після створення макету сайту, необхідно переходити безпосередньо до роботи з кодом, впроваджуючи певні додаткові стилі та динамічність на сторінці. Робиться це за допомогою стилізації елементів засобами CSS. Але не завжди розробники пишуть код за оригінальною технологією. До стилізації, як і до написання JavaScript елементів, є різні сучасні підходи, які також було розглянуто у процесі роботи над системою.

Одними з таких підходів до створення стилів можна назвати препроцесори CSS.

1.3. Програмні можливості мови JavaScript

JavaScript є найпопулярнішою у світі мовою програмування, що відповідає специфікації ECMAScript. JavaScript працює на стороні клієнта в браузері, що може бути використано для проектування / програмування поведінки веб-сторінок. Крім того, JavaScript має ряд властивостей, властивих функціональним мовам, що додає мові додаткову гнучкість.

Ця мова програмування була створена, щоб зробити веб-сторінки динамічними. Програми на цій мові називаються скриптами. Вони вбудовуються в HTML і виконуватися автоматично при завантаженні веб-сторінки.

Разом зі швидким розвитком більшості мов програмування, JavaScript може зараз виконуватися не тільки в браузері, а й на сервері або на будь-якому іншому пристрої, який має спеціальну програму, що називається «движком» JavaScript. Але можливості JavaScript залежать від оточення, в якому він працює. Як приклад, Node.JS може підтримувати функції читання, запису довільних файлів, виконання мережевих запитів і т.д. Але у браузері для JavaScript є всі можливості, що пов'язані з маніпулюванням веб-сторінками, взаємодією з користувачем і веб-сервером.

В основному JavaScript використовується у клієнтській частині веб-додатків: клієнт-серверних програм, в якому клієнтом є браузер, а сервером - веб-сервер, що мають розподілену між сервером і клієнтом логіку. Обмін інформацією в веб-додатках відбувається по мережі. Однією з переваг такого підходу є кросплатформність: клієнти не залежать від конкретної операційної системи користувача.

Приклади програмних можливостей JavaScript:

- Додавання HTML-коду на сторінку, модифікація стилів, зміна існуючого вмісту;
- Реагування на дії користувача, клацання миші, пересування курсору, натискання клавіш;
- Надсилання мережевих запитів на віддалені сервера, завантаження і вивантаження файлів (технології Ajax і Comet);
- Отримання і встановлення кукі, зберігання даних на стороні клієнта;

1.4. Інструменти JavaScript

Для більш швидкого та ефективного створення сучасних веб-застосунків за допомогою JavaScript, розробники використовують різноманітні допоміжні інструменти. Одними з таких інструментів є JS фреймворки та бібліотеки.

Інструменти розробки, що є найкращими та користуються найбільшим попитом серед програмістів станом на 2021 рік було виявлено за допомогою вивчення загальної інформації з документацій, статистичних даних використання наведених вище засобів та актуальних вакансій для спеціалістів з певної технології.

На фоні великої кількості доступних інструментів JavaScript розробки значно вирізняються наступні фреймворки та бібліотеки: JQuery, React, Angular, Vue.js «див. рис. 1.1.». Про це свідчать дані сайту Stack Overflow [45] - найбільшої інтернет-спільноти для програмістів, яку розробники можуть вивчати, ділитися своїми знаннями та розвивати власну кар'єру. Щомісяця Stack Overflow відвідують понад 50 мільйонів професійних та майбутніх фахівців сфери, які допомагають вирішити проблеми з кодуванням та розвинути нові навички.

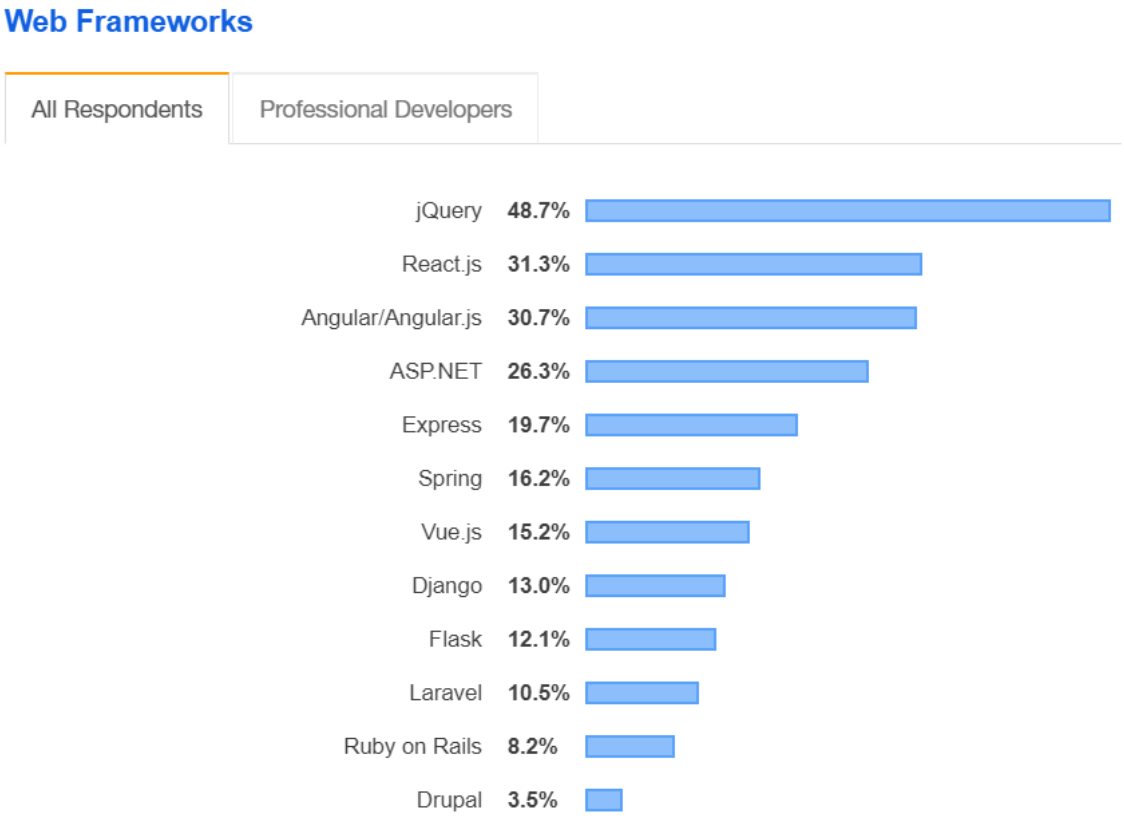


Рисунок 1.3 – Популярність веб-фреймворків згідно Stack Overflow

Схожі результати дослідження були отримані з використанням сервісу Google Trends від Google, що аналізує популярність пошукових запитів у Пошуку Google у різних регіонах, сферах діяльності з 2004 року до сьогоднішнього дня (рис.1.4). Засобами сервісу було переглянуто кількість

запитів на теми Angular, Vue.js, React та JQuery. Лідируючі позиції займають засоби JQuery та React.

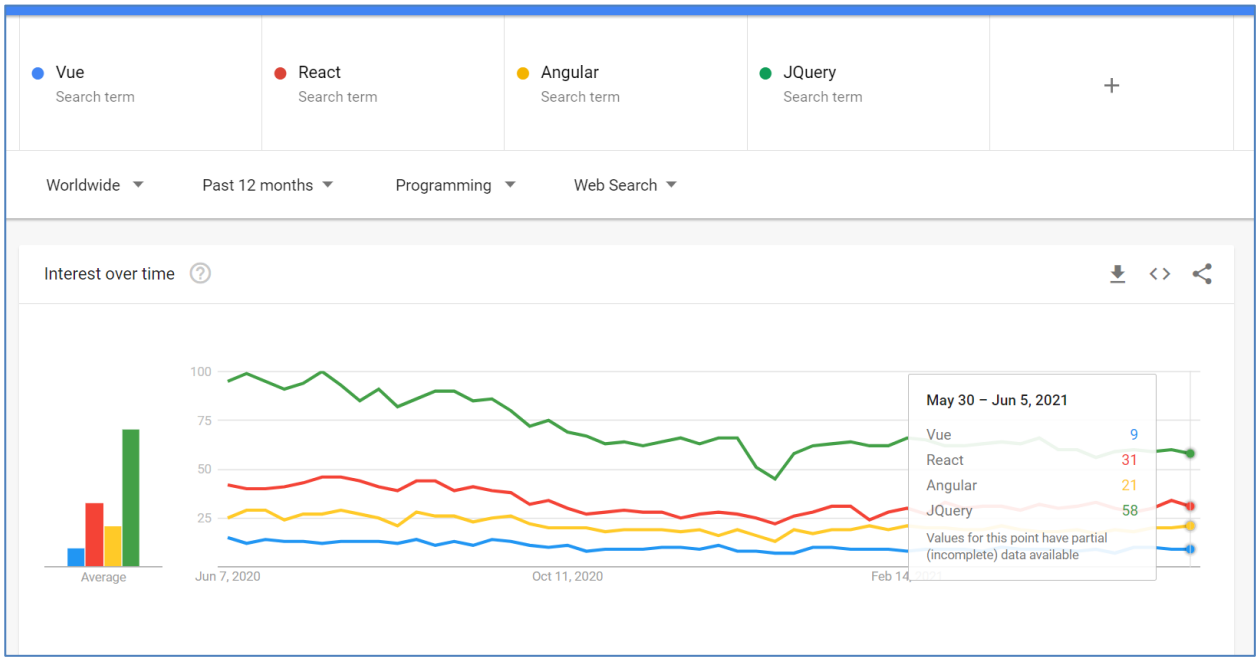


Рисунок 1.4 – статистика запитів у Google

Статистика використання також була переглянута на сайті npm - менеджеру пакетів за замовчуванням для платформи Node.js. Для порівняння було використана кількість завантажень чотирьох найвідоміших засобів за весь час існування системи «див. рис. 1.5.». На графіку видно, як, починаючи з 2017 року, React займає лідируючу позицію у цій категорії.

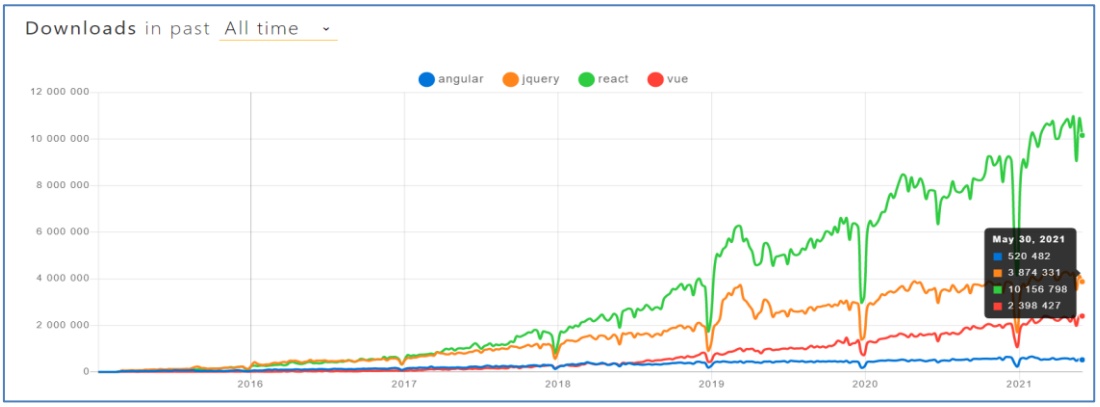


Рисунок 1.5 – статистика npm завантажень

Розглянемо всі популярні інструменти більш детально.

JQuery - легка у користуванні та багатофункціональна бібліотека JavaScript, яка зосереджена на спрощенні навігації по документу, викликів AJAX та обробки подій, створення анімації. Це бібліотека маніпуляцій з Document Object Model (DOM). DOM - деревоподібне представлення всіх елементів веб-сторінки. jQuery спрощує синтаксис пошуку, вибору та керування цими елементами.

Вона була створена у січні 2006 року Джоном Резігом. У травні 2019 року jQuery використовувалась у 73% з 10 мільйонів найпопулярніших веб-сайтів [45]. За даними сервісу BuiltWith [46], станом на 5 червня 2021 року, кількість сайтів, що використовують цю бібліотеку сягнула 71,803,422.

Серед найвідоміших платформ в Інтернеті, що використовують jQuery:

- Google.com
- Youtube.com
- Amazon.com
- Yahoo.com
- Microsoft.com
- IBM.com
- Netflix.com

Існує безліч інших бібліотек JavaScript, але jQuery, мабуть, найпопулярніша, а також найбільш розвинена. Це зумовлено тим, що цей інструмент був створений одним з перших. Але jQuery має і свої недоліки, наприклад, велика вага файлів бібліотеки, що нині, в залежності від версії становить від 27 до 274 KB. Для порівняння, вага React - 5.3 KB.

Angular - це JavaScript-фреймворк з відкритим програмним кодом, призначений для розробки SPA (Single-page application): односторінкових додатків, що складаються з однієї HTML сторінки, певних CSS стилів та великої кількості JavaScript коду.

Angular використовує MVC (Model View Controller) - шаблон дизайну програмного забезпечення, який зазвичай використовується для розробки користувацьких інтерфейсів та розділяє пов'язану логіку програми на три відповідні взаємопов'язані елементи: модель даних, інтерфейс та контроллер. Такий підхід вважається однією з характерних ознак фреймворку.

Крім того, AngularJS підтримує такий функціонал, як Ajax, управління структурою DOM, анімацію, шаблони, маршрутизацію та інш.

За даними сервісу BuiltWith [46], станом на 5 червня 2021 року, кількість сайтів, що використовують AngularJS становить 102,360. Не дивлячись на таку порівняно невелику кількість платформ, що використовують цей фреймворк, знання технології користується попитом на ринку праці. За допомогою відомого сервісу пошуку роботи та працівників LinkedIn було знайдено більше 126 тисяч вакансій на посаду AngularJS розробника «див. рис. 1.6.». Для порівняння, спеціалістів з не менш популярної технології React потребують близько 67 тисяч роботодавців. Представлені дані є результатом пошуку в Сполучених Штатах за 5 червня 2021 року. Цього ж дня вакансій на посаду AngularJS розробника в Україні налічувалося більше двох тисяч.

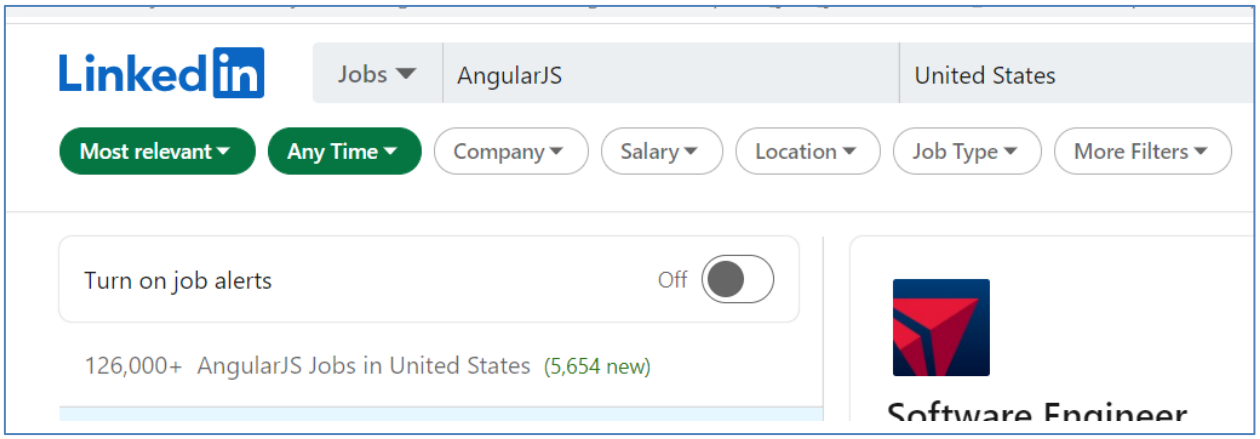


Рисунок 1.6 – доступні вакансії на LinkedIn

Наступний за популярністю JavaScript-фреймворк - Vue.js. Це сучасний та прогресивний засіб, призначений для створення веб-додатків клієнтського рівня: користувацьких інтерфейсів та вдосконалених односторінкових програм.

Vue.js також здатний доповнювати складні односторінкові програми, коли використовується у поєднанні з сучасними інструментальними засобами та підтримкою бібліотек. Цей фреймворк має досить невеликий розмір - не більше 20 кБ, і при цьому володіє гарною продуктивністю у порівнянні з такими засобами як Angular або React. Саме тому даний фреймворк останнім часом набирає обертів і стає більш популярним.

За даними сервісу BuiltWith [46], станом на 5 червня 2021 року, кількість сайтів, що використовують Vue.js становить 1,359,526. На ринку праці ситуація виявилась наступною: більше 16 тисяч вакансій в Сполучених Штатах та більше 400 вакансій в Україні, згідно даних платформи LinkedIn.

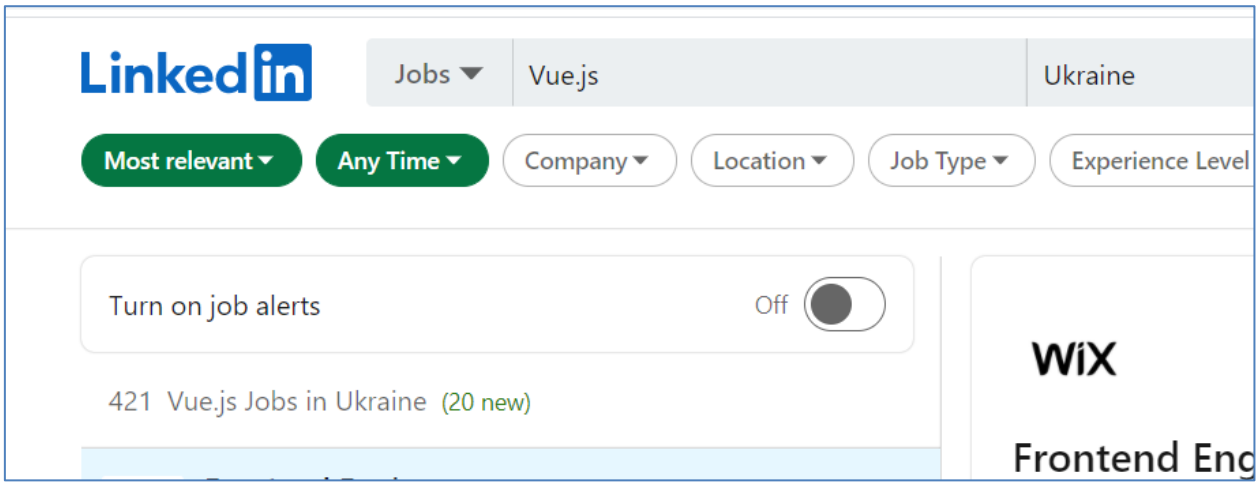


Рисунок 1.7 – доступні вакансії з Vue.js на LinkedIn

Дослідивши статистику та більшість альтернативних інструментів для розробки веб-застосунків, для роботи над проектом було обрано засіб React. Це JavaScript-бібліотека для створення користувацьких інтерфейсів, що була розроблена та підтримується Facebook.

Більш детально особливості бібліотеки розглянуто у розділі 3.1. Структура проекту. Повертаючись до статистики, за даними сервісу BuiltWith [46], станом на 5 червня 2021 року, кількість сайтів, що використовують React сягнула 5,458,419. Це в рази більше, ніж у наведених вище фреймворках. У сервісі LinkedIn знайшлось більше тисячі вакансій в Україні для фахівців з цієї технології.

Нині React використовується на таких відомих платформах:

- Github.com
- Fiverr.com
- Discord.com
- Paypal.com
- Binance.com
- Cnn.com
- Dropbox.com
- Trello.com

Висновки до розділу 1

У першому розділі було розглянуто теоретичні основи створення сайтів, а також всі сучасні методи веб-розробки.

Сайт будується за допомогою трьох головних складових: HTML розмітки, CSS стилів, JavaScript логіки. Перед розробкою веб-додатку необхідно спочатку чітко сформулювати усі вимоги та необхідний функціонал продукту, спроектувати структуру додатку. На цьому етапі розробники використовують макети та прототипи сайтів, щоб розуміти, як додаток буде виглядати у готовому вигляді.

Після вирішення проблем структури та зовнішнього вигляду, необхідно переходити до основної частини розробки, а саме програмуванню на

JavaScript. Для цього зараз використовуються безліч допоміжних засобів, таких як фреймворки та бібліотеки.

Після дослідження наявних варіантів, основним інструментом розробки було обрано засіб React та ще певний набір бібліотек, про які буде написано в пункті 3.4. Допоміжні засоби для розробки.

РОЗДІЛ 2 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ФОРМУЛЮВАННЯ СИСТЕМНИХ ВИМОГ ЗАСТОСУНКУ

2.1. Сфера застосування додатку

Створення програмної системи пошуку роботи є значним вкладом у працевлаштування населення. Завдяки автоматизації такого процесу значно покращується та спрощується процес знаходження роботи для користувачів та пошук робітників для роботодавців з різноманітних підприємств.

2.2. Огляд існуючих веб-застосунків для пошуку робочих місць

Пошук роботи є актуальною темою для багатьох людей як в Україні, так і в інших країнах світу. Типовий сайт заданого вище типу зазвичай містить в собі функціонал для створення власних резюме, вакансій та пошук необхідних за певними критеріями. Крім цього, користувачі мають змогу зареєструватися на сайті, що значно розширює його функціонал для споживача: з’являються такі можливості, як збереження обраних резюме/вакансій, редагування власного профілю, зв’язок з іншими користувачами і тд. Повний перелік функціоналу є різним для кожного сайту.

У ході аналізу існуючих систем пошуку роботи було досліджено такі відомі та загальні платформи для працевлаштування, як:

- Work.ua;
- Robota.ua;
- Jobs.ua;
- LinkedIn.com;

Особливу увагу було приділено платформам, що спеціалізуються на пошуку роботи з IT спеціальностей, а саме:

- Dice.com
- Stackoverflow.com

- Crunchboard.com
- Jobs.github.com
- Djinni.co

Функціонал цих сайтів було вивчено та проаналізовано. Кожен створений за допомогою розмітки HTML, стилів сторінок CSS, фреймворків для покращення роботи у веб середовищі та мови програмування JavaScript.

Зовні майже усі наведені вище платформи схожі між собою: відкривши сайт, користувач відразу бачить поле пошуку на головній сторінці, або потрапляє безпосередньо на сторінку для пошуку та повноцінної фільтрації результатів.

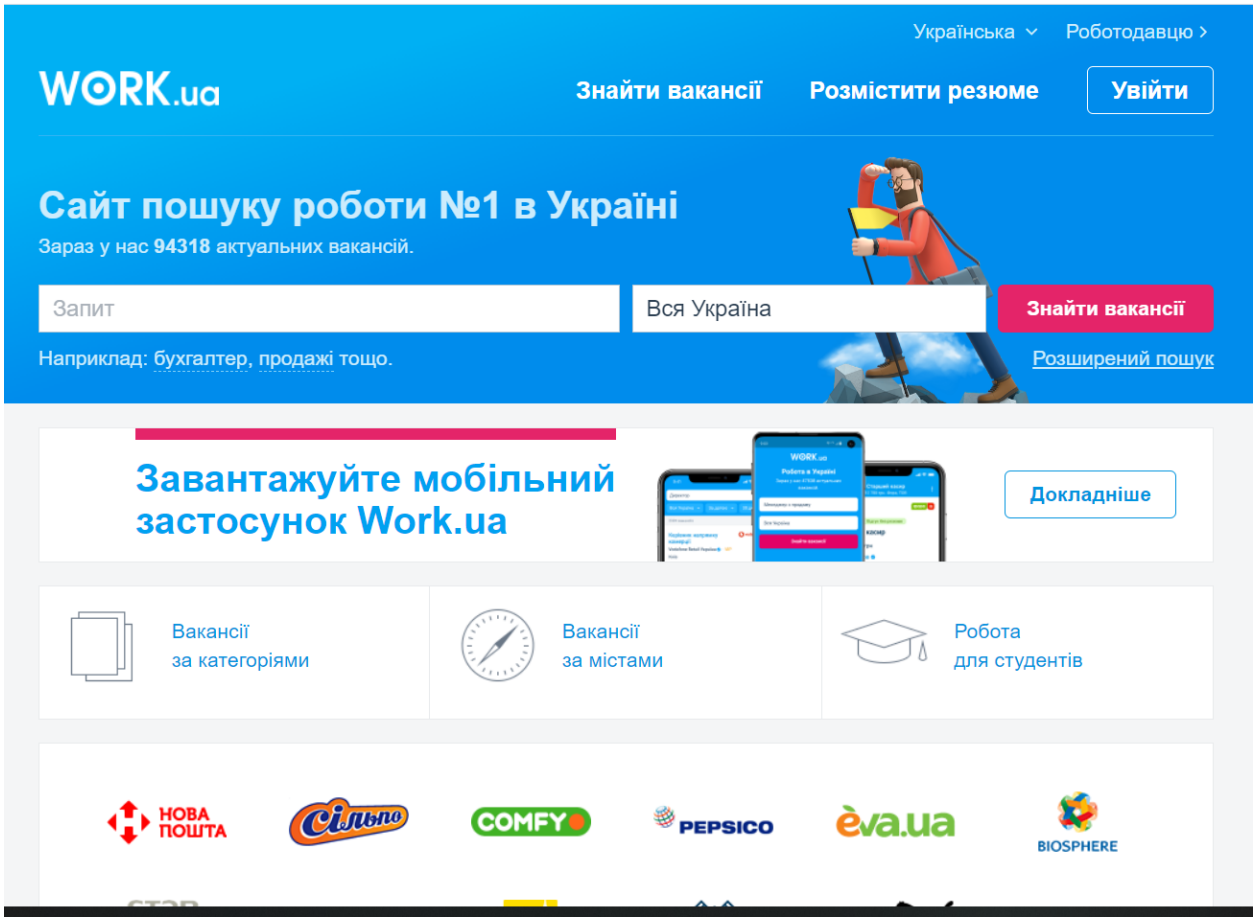


Рисунок 2.1 – головна сторінка сайту Work.ua

Серед проаналізованих систем є одне виключення: інтерфейс платформи LinkedIn. Відкривши головну сторінку сервісу, користувач має спочатку обрати, що саме він шукає. За бажанням, можливо прокрутити сторінку нижче та більш детально ознайомитись з описом платформи.

Крім цього, серед особливостей цього сайту можна виділити те, що він деяким чином схожий на соціальну мережу, ніж на звичайний пошукач: користувачі мають змогу створювати зв'язки один з одним, взаємодіяти через уподобання чи коментарі, а також схвалювати навички обраного користувача та рекомендувати його роботодавцям.

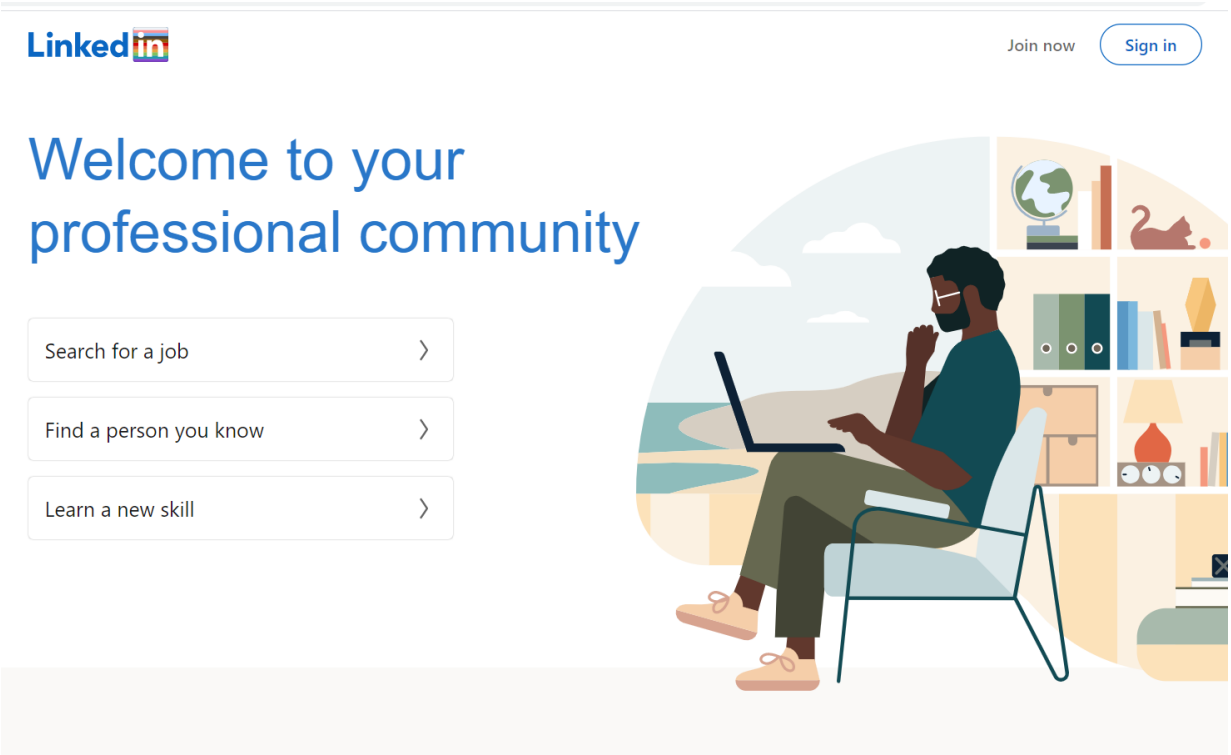


Рисунок 2.2 – Головна сторінка сайту LinkedIn

Аналізуючи українські пошукові системи, була помічена певна спільна риса: багато платформ розміщують на головній сторінці зображення та посилання на відомі компанії.

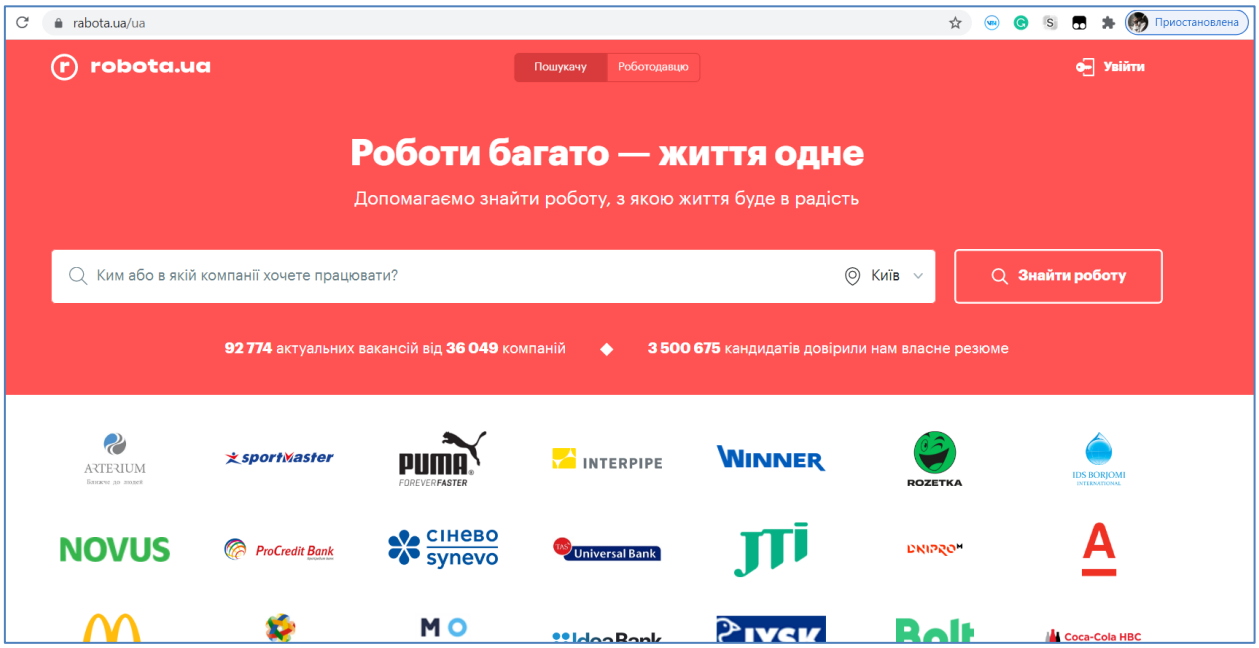


Рисунок 2.3 – Головна сторінка сайту Rabota.ua

На мою думку, як користувача та потенційного шукача роботи, це не дуже доречно, адже людина, перш за все, буде шукати роботу згідно своїм навичкам та пропонованим посадам, ніж дивитися конкретні компанії.

Хорошим прикладом організації головної сторінки можу представити сайт Indeed, де біля пошуку користувач може бачити популярні запити. Крім того, можна також відзначити простий та зручний інтерфейс додатку.

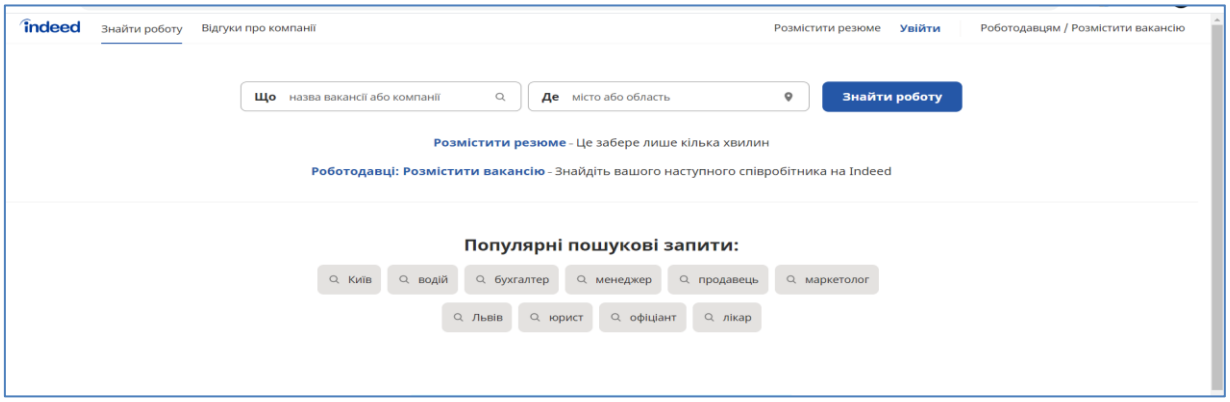


Рисунок 2.4 – Головна сторінка сайту Indeed.com

Ще одним гідним прикладом оформлення початкової сторінки вважаю сайт Work.ua, де після пошуку користувач бачить категорії, по яким може швидко та зручно перейти до підбору вакансії.

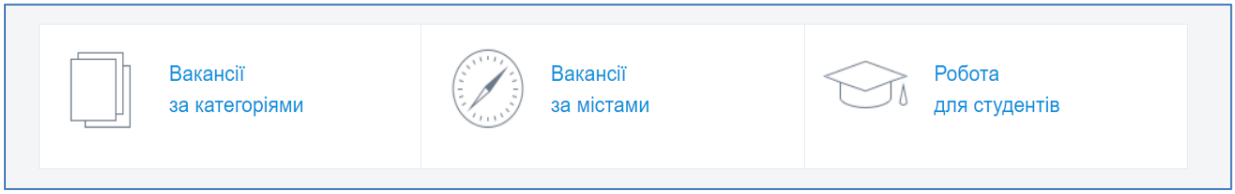


Рисунок 2.5 – Критерії пошуку сайту Work.ua

Після вибору певного блоку, наприклад, вакансій за категоріями, користувач може обрати напрям пошуку з окремого списку, при цьому ще вказавши необхідне місто.

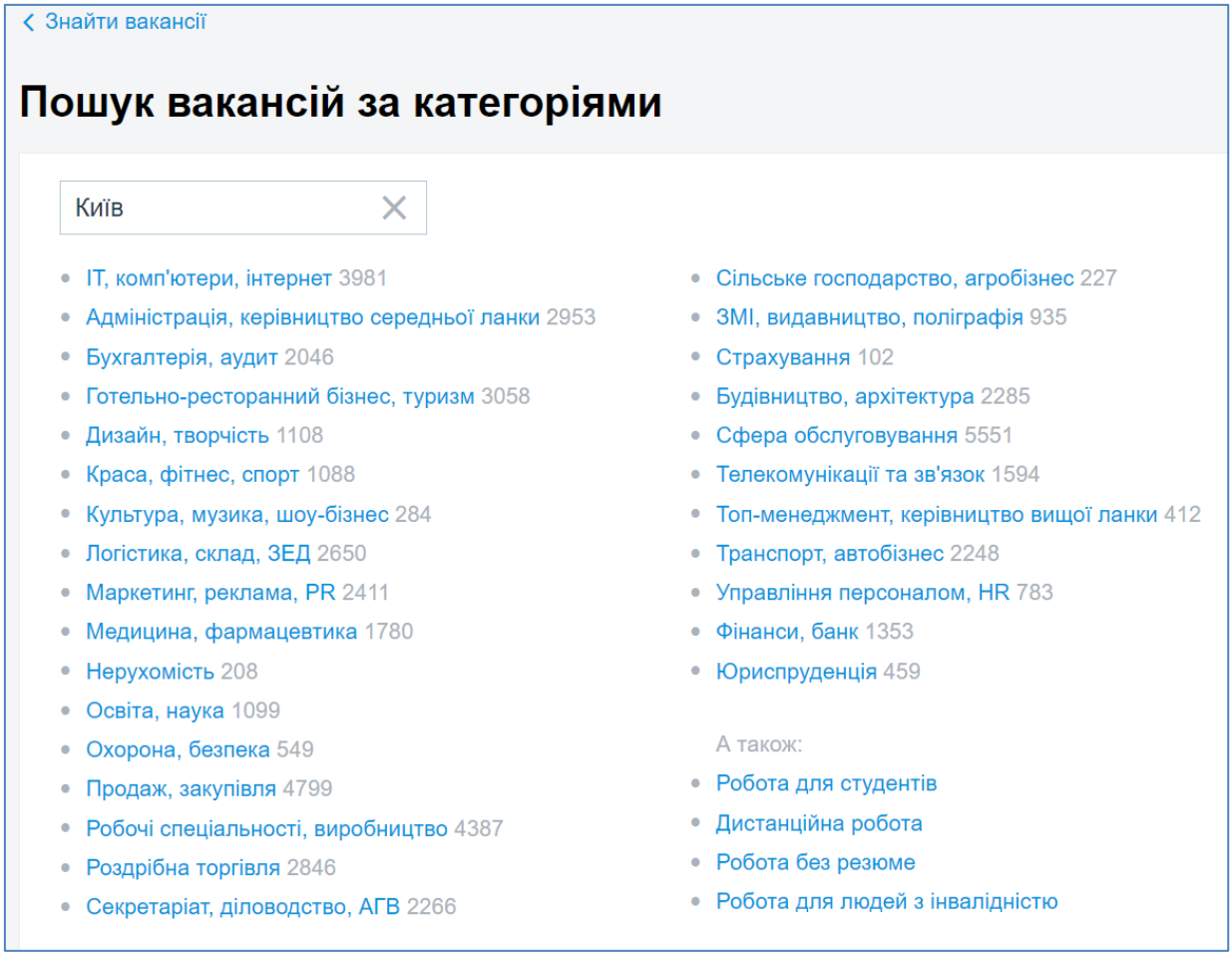


Рисунок 2.5 – Пошук вакансій сайту Work.ua

Сам процес пошуку схожий на всіх платформах: для фільтрації відкривається окрема сторінка, зверху або збоку на котрій розташовані параметри пошуку, а внизу виводяться картки з доступними вакансіями у вигляді списку.

Проаналізувавши існуючі приклади оформлення таких сторінок, було зазначено, що вони мають відповідати таким вимогам:

- Чим більше доступних критеріїв - тим краще: користувачу буде легше знайти необхідну посаду, якщо у пошуку присутній, наприклад, пункт “дистанційна робота” або “працевлаштування без досвіду”;
- В картках лише основна інформація: на деяких платформах розробники виводять у картки опис вакансії, створюючи цим досить великий об’єм тексту для кожного оголошення, що може створити труднощі для пошуку. Набагато легше знайти необхідне, коли у блоці зазначені тільки певні ключові слова.
- Оформлення також має значення. Картки повинні бути чітко відокремлені одна від одної, а дизайн оголошення треба зробити таким чином, щоб всю інформацію було легко читати, а найголовніші ключові слова виділялися серед тексту.
- Для кращого сприйняття тексту користувачем, до карток можна додавати різноманітні символи та зображення. Наприклад, для вакансії зробити блок для завантаження рисунку з логотипом фірми, що шукає працівників на певну посаду.

Прикладом якісного оформлення сторінки пошуку можна навести сервіс Dice.com.

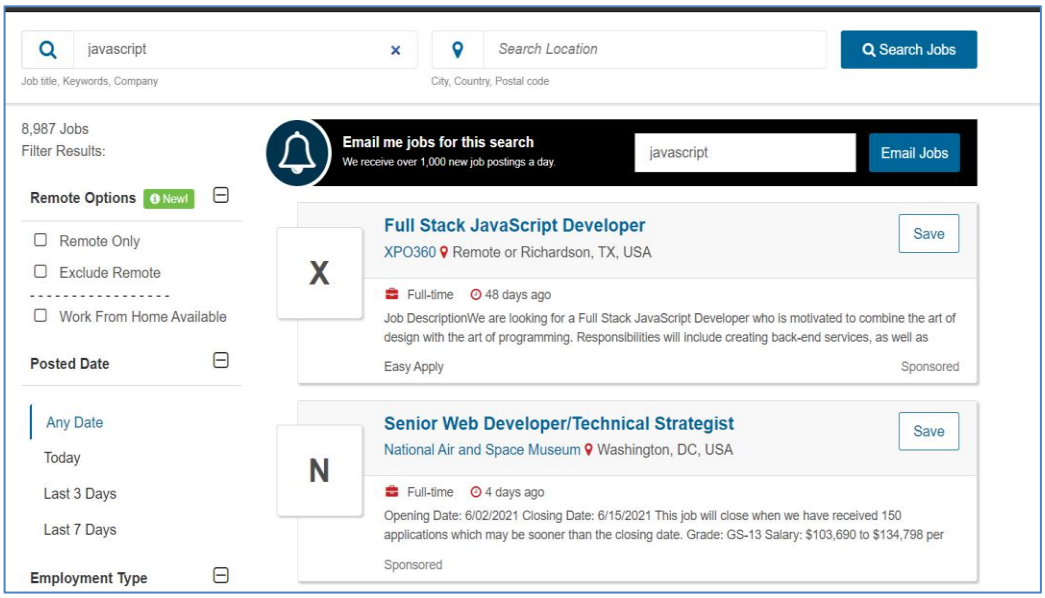


Рисунок 2.6 – Пошук вакансій сайту Dice.com

На сторінці зручно зроблений пошук, є багато критеріїв для фільтрації. Картки чітко відокремлені, є ключові слова, назви певних категорій замінені на значки, що значно полегшує сприйняття інформації.

Крім цього, також непоганим варіантом є приклад сайту, де основна увага приділена саме ключовим словам.

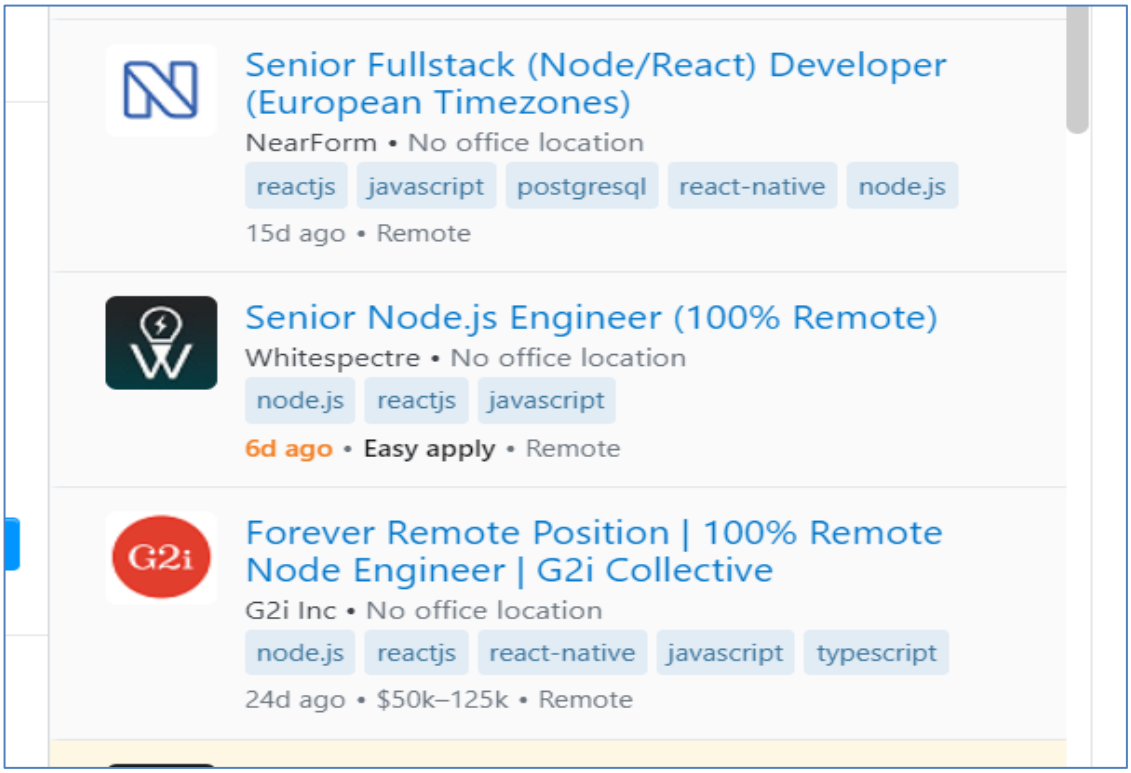


Рисунок 2.7 – список вакансій сайту Stackoverflow.com

Наступний розглянутий пункт - сторінки обраного оголошення. На кожному сайті вони оформлені по різному, хоча структура певним чином схожа: назва посади, певні ключові слова та детальний опис вакансії.

Як і у попередньому випадку, найголовніше при створенні таких сторінок - зручне розташування елементів, виокремлення найважливіших моментів та полегшення сприйняття інформації для користувача шляхом створення акцентів на різних частинах сторінки.

Одним із найкращих прикладів, згідно з цими вимогами, є оформлення сторінки на українському порталі.

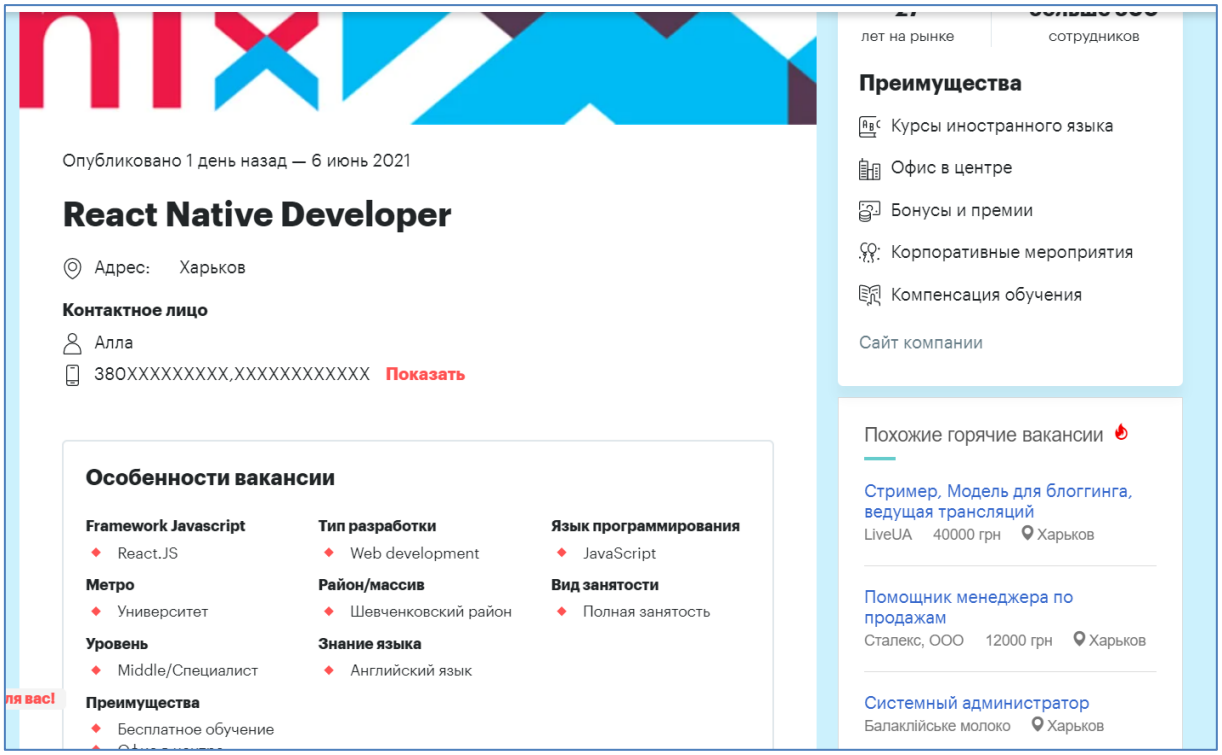
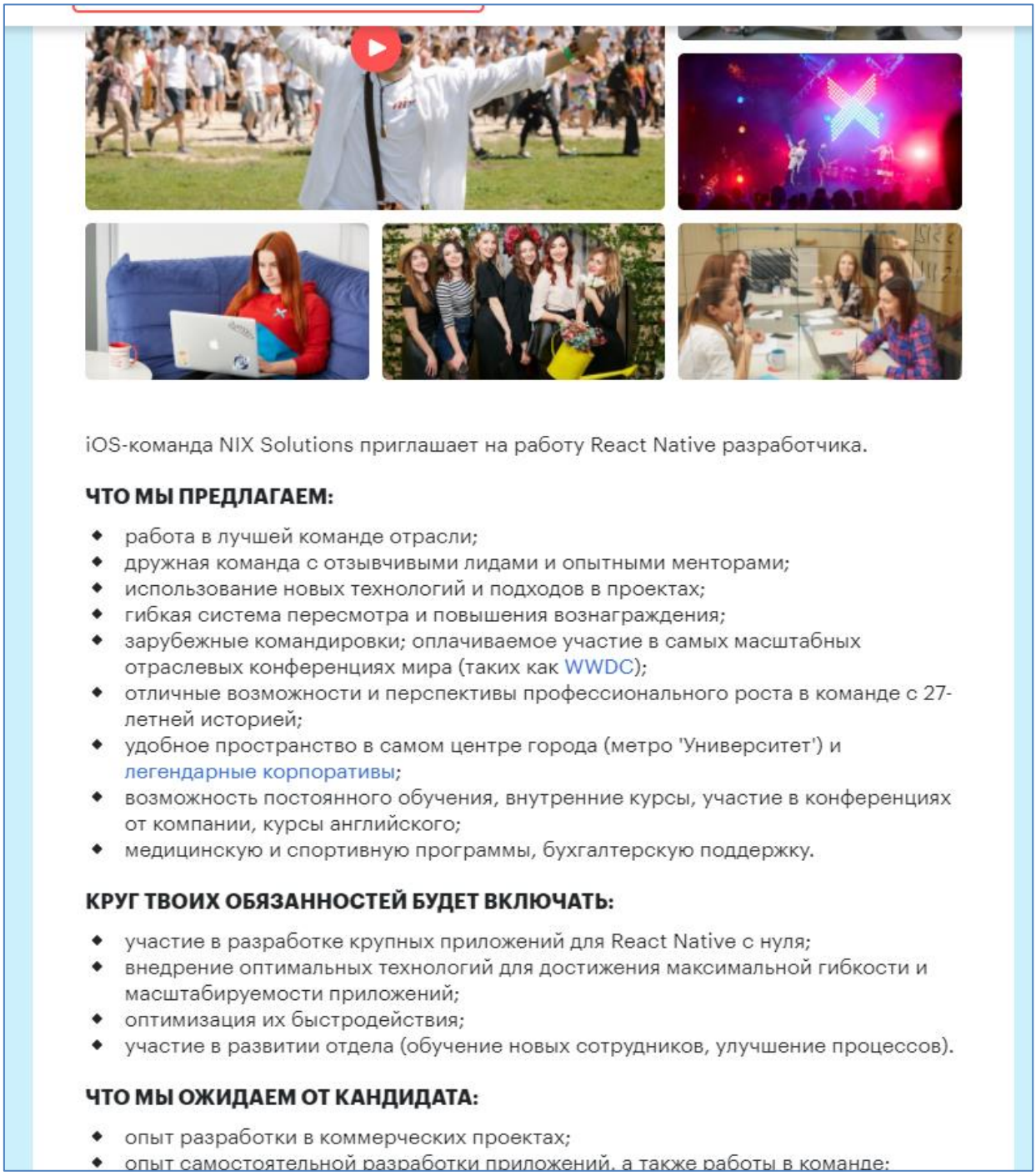


Рисунок 2.8 – Сторінка вакансії сайту Rabota.ua

Назва посади та основні моменти чітко виокремлені, особливості вакансії зроблені у вигляді блоку з найважливішою інформацією, що дуже легко та швидко сприймається, також окремо є картка компанії, де також зазначено лише найголовніші аспекти та є перехід на окрему сторінку та сайт організації.

Далі на сторінці розташований опис, який, за бажанням роботодавця, можливо доповнити за допомогою медіа файлів та зробити зручний список з усіма вимогами та перевагами вакансії. Саме оголошення оформлено окремим блоком, що не дозволяє йому візуально змішуватись з рештою інформації на сайті.



iOS-команда NIX Solutions приглашает на работу React Native разработчика.

ЧТО МЫ ПРЕДЛАГАЕМ:

- работа в лучшей команде отрасли;
- дружная команда с отзывчивыми лидами и опытными менторами;
- использование новых технологий и подходов в проектах;
- гибкая система пересмотра и повышения вознаграждения;
- зарубежные командировки; оплачиваемое участие в самых масштабных отраслевых конференциях мира (таких как WWDC);
- отличные возможности и перспективы профессионального роста в команде с 27-летней историей;
- удобное пространство в самом центре города (метро 'Университет') и легендарные корпоративы;
- возможность постоянного обучения, внутренние курсы, участие в конференциях от компании, курсы английского;
- медицинскую и спортивную программы, бухгалтерскую поддержку.

КРУГ ТВОИХ ОБЯЗАННОСТЕЙ БУДЕТ ВКЛЮЧАТЬ:

- участие в разработке крупных приложений для React Native с нуля;
- внедрение оптимальных технологий для достижения максимальной гибкости и масштабируемости приложений;
- оптимизация их быстродействия;
- участие в развитии отдела (обучение новых сотрудников, улучшение процессов).

ЧТО МЫ ОЖИДАЕМ ОТ КАНДИДАТА:

- опыт разработки в коммерческих проектах;
- опыт самостоятельной разработки приложений, а также работы в команде;

Рисунок 2.9 – Сторінка вакансії сайту Rabota.ua

Усі зазначені вище переваги та недоліки існуючих систем з пошуку роботи було ретельно проаналізовано та створено власний список вимог, яким повинен буде відповідати веб-застосунок власної розробки в межах бакалаврської роботи.

2.3. Функціональні вимоги до програми

Проаналізувавши наведені вище платформи з пошуку роботи, було знайдено певну кількість загальних властивостей, а саме:

- На сайтах обов’язково присутня можливість реєстрації та авторизації, збереження певних особистих даних, створення власних оголошень;
- Більшість вакансій та резюме розділені за певними категоріями, що значно полегшує користувачам пошук;
- Для кожного оголошення створена окрема сторінка, де можна детально ознайомитись з повним його змістом.

Кожен пункт цього списку є, безсумнівно, перевагами сайту. Завдяки сортуванню пропозицій за категоріями, значно легше знаходити потрібну інформацію. Завдяки такому функціоналу, користувач зможе використовувати таку платформу для кращого та більш простого пошуку роботи або робітника на власне підприємство.

Основним завданням роботи є створення веб−застосунку для швидкого та зручного пошуку роботи в IT сфері.

Окрім зазначеного вище функціоналу, необхідно реалізувати наступне:

- Адаптивність сайту під різноманітні пристрої та розміри екрану;
- Зручний та простий інтерфейс, без занадто великої кількості інформації, що буде відповідати всім вимогам, наведеним у попередньому пункті;

- Інтуїтивно зрозумілий функціонал платформи, яким без проблем можуть користуватись навіть користувачі без досвіду роботи зі схожими системами пошуку;
- Провести тестування функціоналу та адаптивності веб−сайту, переконатися у відсутності помилок.

Висновки до розділу 2

У ході аналізу предметної галузі застосунка було проаналізовано існуючі систем з пошуку роботи, були зазначені переваги та недоліки відомих платформ та список вимог, яким повинен буде відповідати веб-застосунок власної розробки в межах бакалаврської роботи.

Серед вимог до зовнішнього вигляду додатку необхідно виокремити зручний та максимально простий інтерфейс з достатньою кількістю інформації, але не занадто великим її обсягом; оформлення вмісту оголошень у вигляді окремих блоків, що не будуть візуально змішуватися в купу та залишаться легкими для сприйняття; створення акцентів на найважливіших ключових словах у списках оголошень, що значно спростить пошук необхідного для користувача.

Вимоги до функціоналу сайту будуть наступними:

- Можливість реєстрації та авторизації в системі;
- Можливість створювати оголошення у вигляді резюме чи вакансій;
- Можливість пошуку необхідних оголошень засобами застосунку;
- Фільтрація всіх оголошень на сайті за допомогою категорій.

РОЗДІЛ 3. СТРУКТУРА ВЕБ-ЗАСТОСУНКУ ТА ЕТАПИ РОЗРОБКИ

3.1. Структура проекту

Застосунок був написаний за допомогою React - JavaScript-бібліотеки для створення користувацьких інтерфейсів.

Вона підтримується Facebook та спільнотою окремих розробників і компаній. React можна використовувати як основу при розробці односторінкових або мобільних додатків. Однак React займається лише управлінням станом та наданням цього стану DOM, тому для створення додатків React зазвичай потрібно використовувати додаткові бібліотеки та інструменти. Основні використані допоміжні засоби зазначені у підрозділі 3.4. «Допоміжні засоби для розробки».

При розробці за допомогою React зазвичай використовується спеціальний синтаксис - JavaScript XML (JSX). Це певне розширення мови JavaScript, що створює елементи React та дозволяє використовувати HTML-подібний синтаксис для опису структури інтерфейсу.

Для початку роботи над застосунком за допомогою React було використано Create React App - середовище розробки, що є офіційно підтримуваним засобом створення односторінкових програм React. Інструмент власноруч проводить всі необхідні налаштування, конфігурацію та створює проект, з яким можна відразу працювати та розробляти власний застосунок.

Встановити та запустити роботу інструментів можна за допомогою декількох рядків у терміналі, у якому під час розробки встановлюються інші засоби для розробки, наприклад, бібліотеки, що значно полегшують процес написання коду.

3.2. Інтеграція бази даних

Для роботи над проектом була використана нереляційна база даних MongoDB. Вона є документо-орієнтованою, тому не містить у собі таблиць або схем, та зберігає дані у форматі схожому на JSON. Через те що у системі відсутня жорстка схема баз даних, при будь якій зміні концепції зберігання даних ми не повинні повністю переписувати таблиці, адже редагувати вміст бази даних досить просто.

Для роботи з базою існує велика кількість офіційних драйверів для основних мов програмування (C #, C ++, Go, Java, PHP, Node.js, Perl, Python, Rust, Ruby, Swift, Scala). Існує також достатньо неофіційних або підтримуваних спільнотою драйверів для інших мов програмування і фреймворків.

Завдяки всім цим перевагам, MongoDB широко застосовується у веб-розробці.

Для підключення до бази даних на сайті MongoDB створюється спеціальне посилання, відповідно до необхідної мови програмування. Для початку роботи з ним треба підставити у строку свій пароль та назву бази даних.

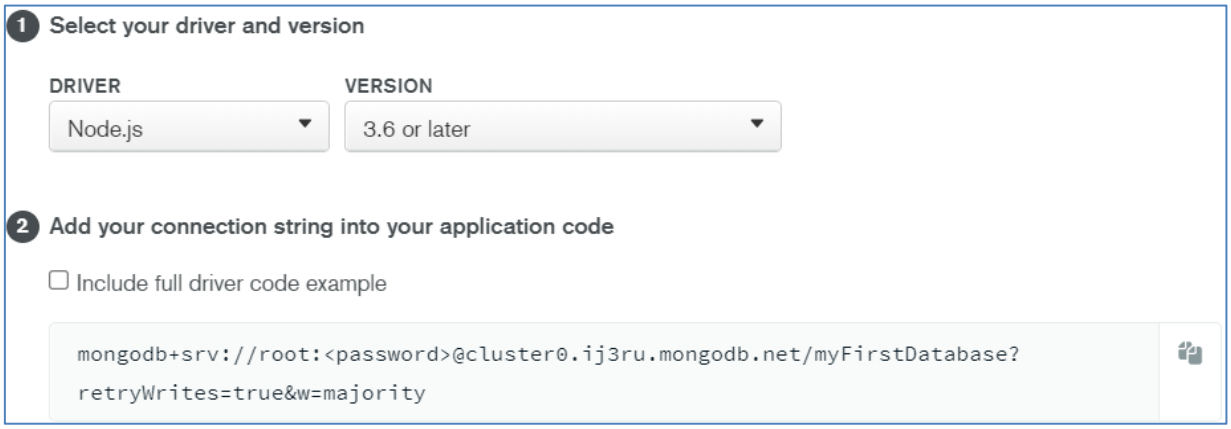


Рисунок 3.1 – Посилання для під’єднання до бази даних

Для подальшої роботи з базою я використовую JavaScript ODM (Object Data Modelling) бібліотеку Mongoose - засіб моделювання об'єктів бази даних.

Після завантаження та імпорту необхідних складових частин бібліотеки, можемо починати роботу. Нижче продемонстровано процес під’єднання до бази даних через функцію mongoose.connect(рис. 3.2).

```
async function start() {
  try {
    await mongoose.connect(config.get("db"), {
      useNewUrlParser: true,
      useUnifiedTopology: true,
      useCreateIndex: true,
    });
    app.listen(PORT, () => console.log(`Started on ${PORT}`));
  } catch (error) {
    console.log("App start error", error);
    process.exit(1);
  }
}
```

Рисунок 3.2 – Під’єднання до бази даних

В окремому файлі я прописую модель бази даних, де спочатку імпортую необхідний мені інструмент за допомогою require("mongoose") (підключення до бібліотеки Mongoose), а потім створюю схему з об’єктами, які і будуть містити необхідні мені дані. У випадку нижче продемонстрована схема для створення користувачів з їх поштовою скринькою та паролем.

```
const { Schema, model, Types } = require("mongoose");

const UserSchema = new Schema({
  email: { type: String, required: true, unique: true },
  name: { type: String, required: true, unique: true },
  password: { type: String, required: true },
```

Рисунок 3.3 – Створення моделі до бази даних

Після створення колекції та декількох користувачів, бачимо їх дані у самій базі у наступному вигляді. “_id” - унікальний номер об’єкту, присвоюється автоматично, якщо не був заданий при створенні.

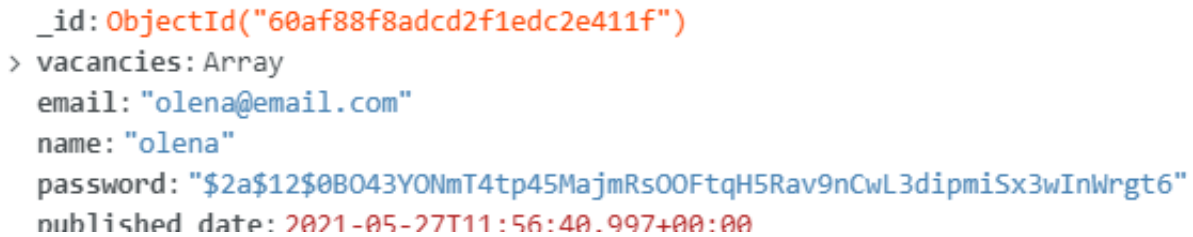


Рисунок 3.4 – Користувач у базі даних

3.3. Функціонал застосунку

На сайті передбачено наступний функціонал:

- реєстрація та авторизація користувачів;
- особиста сторінка авторизованого користувача;
- можливість створити власне оголошення - резюме;
- можливість переглядати всі існуючі на сайті резюме;
- можливість створити власне оголошення - вакансію;
- можливість переглядати всі існуючі на сайті вакансії;
- пошук серед списку оголошень;
- підбір вакансій за відповідною категорією;
- ознайомлення з повним вмістом оголошень на персональних сторінках для кожного резюме чи вакансії;
- можливість зв’язатись з кожним автором оголошення через електронну пошту.

3.4. Допоміжні засоби для розробки

Для покращення та пришвидшення розробки застосунку були застосовані певні інструменти - бібліотеки та модулі - що були завантажені у систему засобами npm.

Список допоміжних засобів:

- Bootstrap фреймворк - набір інструментів для створення сайтів і веб-застосунків. Включає в себе HTML- і CSS-шаблони оформлення компонентів веб-інтерфейсу, включаючи JavaScript-розширення.
- React Router - динамічна маршрутизація на стороні клієнта, завдяки якій застосунок працює по принципу Single Page Application.
- React Hook Form - бібліотека для зручної взаємодії з формами, що не тільки спрощує процес написання коду, але і покращує роботу застосунка в цілому, зменшуючи кількість відтворення елементів на сторінці та тим самим збільшуючи продуктивність системи.
- Styled Components - один з нових способів написання CSS в сучасному JavaScript, де стилі прописуються в межах одного компоненту. Такий метод не тільки робить стилізацію більш зручною, але і відкриває для того нові можливості.
- Express.js - фреймворк веб-додатків для Node.js, реалізований як відкрите програмне забезпечення для створення веб-додатків і API. Використовувався для роботи з MongoDB.
- JSON Web Token - відкритий стандарт (RFC 7519) для створення токенів доступу, заснований на форматі JSON. У проекті використовувався для передачі даних для аутентифікації користувачів у застосунку.
- Nodemon - інструмент, який значно пришвидшує розборку застосунку шляхом автоматичного перезапуску програми, коли виявляються зміни файлів у каталозі.

3.5. Демонстрація складових частин за стосунку

Даний підрозділ присвячений демонстрації розробленого функціоналу застосунку з пошуку робочих місць з IT спеціальностей.

3.5.1. Огляд головної сторінки та її елементів

Сторінки застосунку складаються з трьох основних елементів: меню (Header), основної частини, де розміщується вся динамічна інформація, та частини знизу (Footer). Елементи крім основного є незмінними, тобто вони відображаються на кожній сторінці застосунку.

Меню сайту або Header складається з наступних частин:

- Головна сторінка
- Вакансії
- Резюме
- Категорії
- Авторизація/Реєстрація або Власна сторінка, якщо користувач авторизувався.

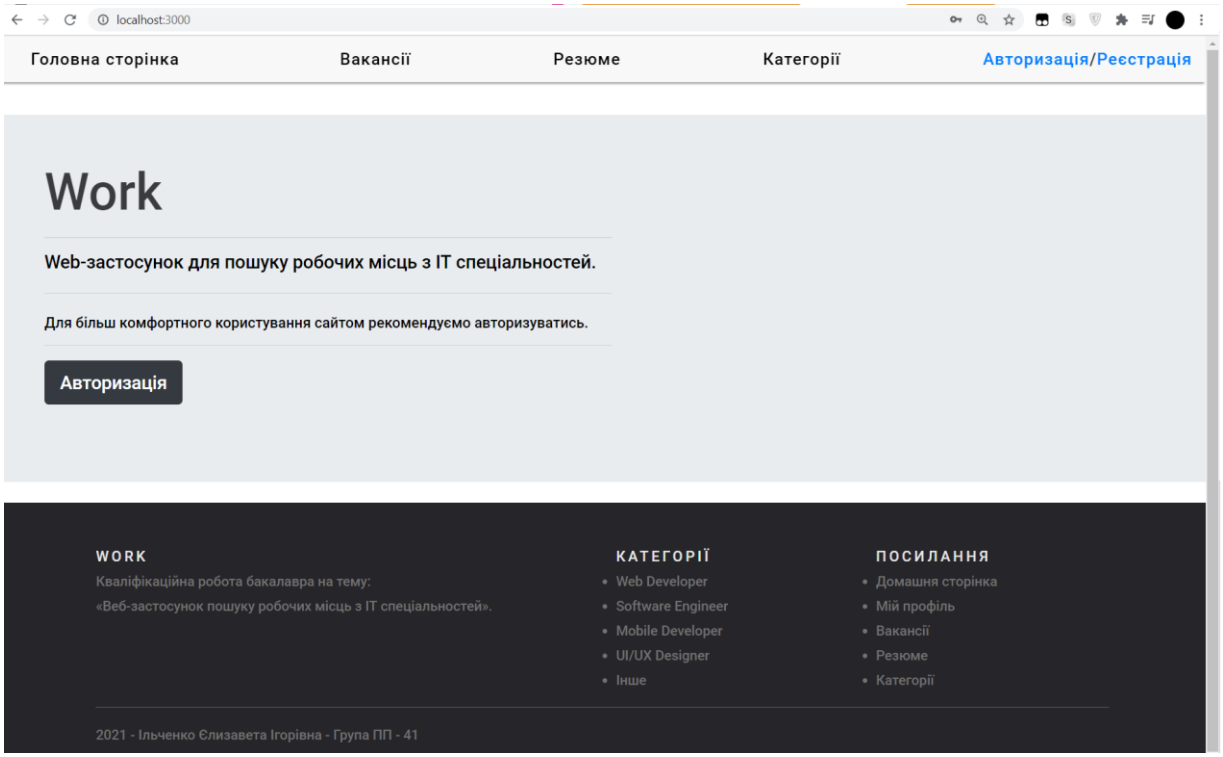


Рисунок 3.5 – Головна сторінка застосунку

3.5.2. Процес реєстрації та авторизації

Реєстрація та авторизація на сайті не є обов'язковою, але вона надає користувачам доступ до повного функціоналу застосунку. Для початку процесу на Головній сторінці потрібно натиснути кнопку "Авторизація". Якщо користувач ще не має власного акаунту, він може перейти до форми "Реєстрація" прямо у попередньому вікні.

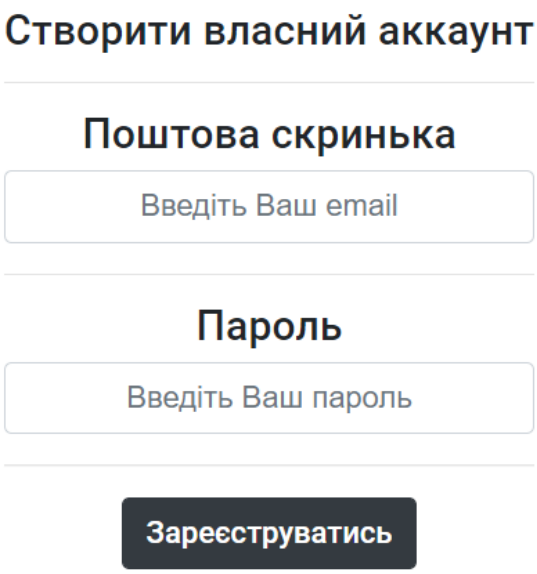


Рисунок 3.6 – Форма реєстрації

Поштова скринька в системі використовується у якості логіну та імені користувача. Крім того, завдяки ній з користувачем згодом зможуть зв'язуватись інші користувачі стосовно оголошень.

Поля форми є обов'язковими для заповнення. Крім того, поле для пошти повинне у собі містити не менше 8 знаків. Поле з паролем - не менше 6.

Для роботи з формою використовується JS бібліотека React-Hook-Form. Завдяки ній можна швидко та зручно проводити валідацію форм, керувати станом елементів та даними користувачів.

Для взаємодії прописуємо хук useForm(рис. 3.7).

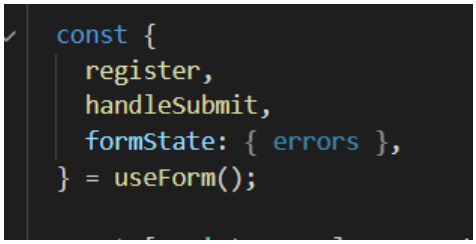


Рисунок 3.7 – Хук useForm()

Після цього додаємо коллбек у форму, що згодом спрацює при її відправці.

Рисунок 3.8 – дія handleSubmit

Крім того, вказуємо, що саме буде виконуватись при дії onSubmit. У нашому випадку, функція отримує дані про поштову скриньку та пароль користувача. Валідація форми відбувається за допомогою засобів бібліотеки React Hook Form.

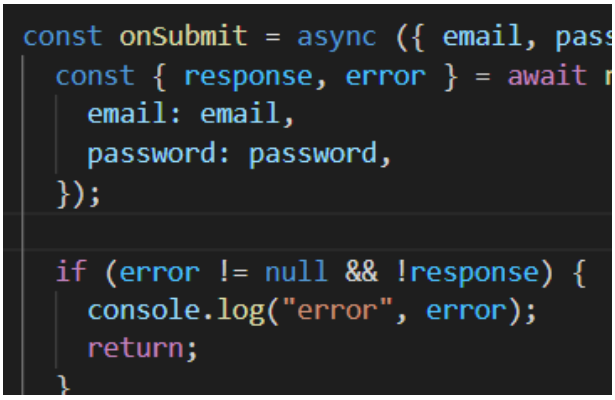


Рисунок 3.9 – помилка через пусту форму

3.5.3. Особиста сторінка користувача

Після авторизації на сайті користувач має змогу перейти до своєї сторінки. На цій сторінці знаходяться наступні елементи:

- Мої вакансії;
- Аккаунт;
- Моє резюме.

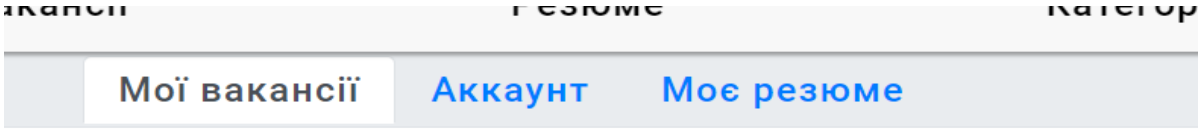


Рисунок 3.10 – Панель елементів сторінки користувача

В останній вкладці під назвою Моє резюме знаходиться форма, де користувач має можливість створити власне резюме, що після збереження відобразиться у списку всіх резюме на сайті.

У вкладці Аккаунт міститься інформація про реєстраційні дані користувача, а також є можливість вийти з аккаунту.

Вкладка Мої вакансії містить список всіх оголошень для пошуку робітника, які були створені користувачем, та кнопка для створення нової вакансії.

3.5.4. Список існуючих резюме та створення власного оголошення

Розглянемо більш детально процес створення та перегляду оголошень. Через форму “Моє резюме” можна не тільки створювати оголошення та додавати особисту інформацію, але і редагувати дані, введені раніше.

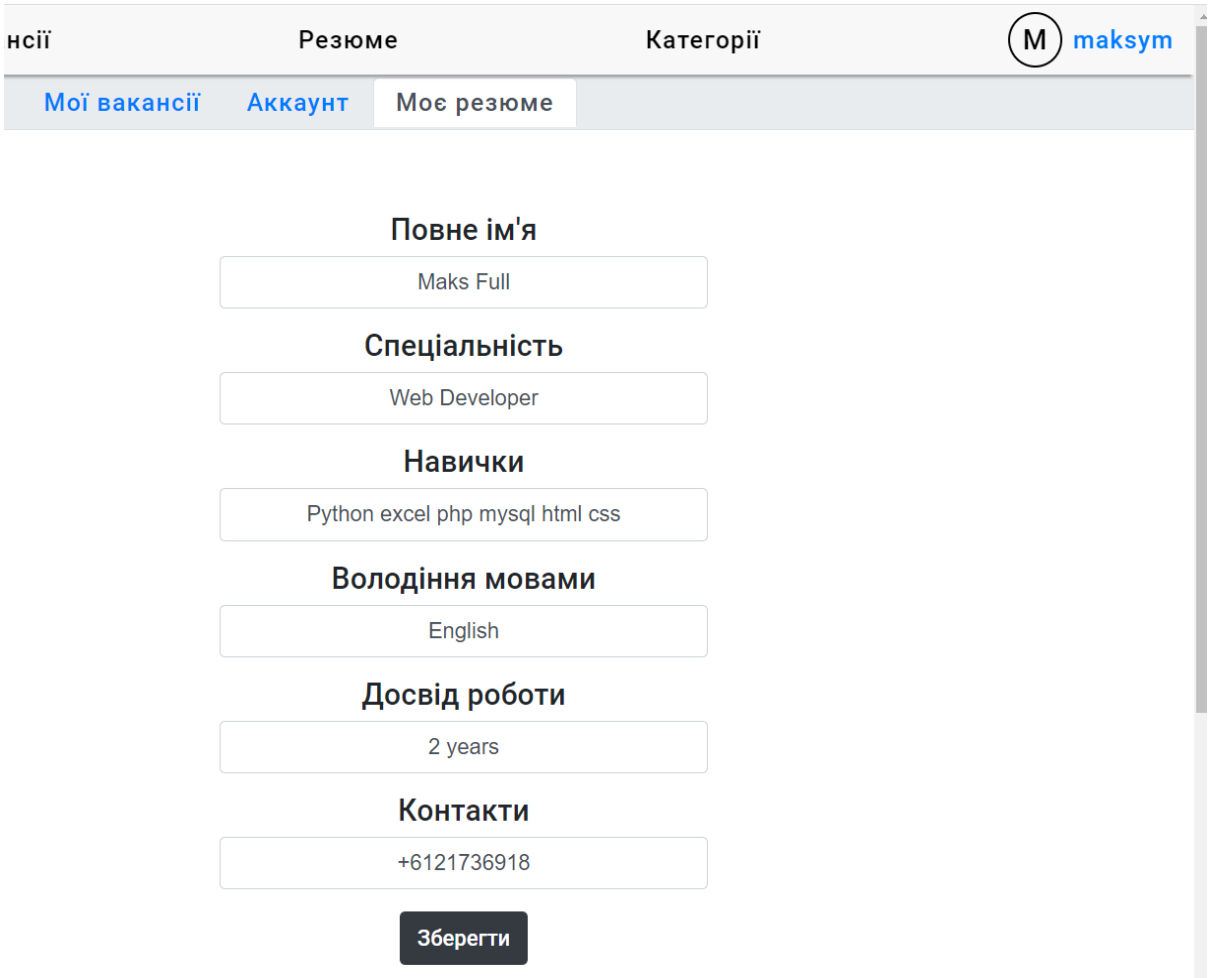


Рисунок 3.11 – Демонстрація форми створення резюме

У формі прописані наступні поля:

- Повне ім’я - від користувача очікується ПІБ;
- Спеціальність - той напрям та посада, яку користувач бажає отримати;
- Навички - через пробіл (!) прописуються всі знання користувача у IT сфері, такі як мови програмування, знайомі технології та методи розборки, програмні засоби і тд.
- Володіння мовами - через пробіл (!) прописуються всі мови, якими володіє користувач.
- Досвід роботи - очікується вся інформація стосовно минулого місця працевлаштування користувача: період роботи, організація, посада, обов’язки та інш.

- Контакти - вказується назва поштової скриньки, через яку потім буде змога зв’язатись з користувачем.

Всі поля форми є обов’язковими для заповнення. Після того як користувач збереже оголошення, він може знайти його у загальному списку оголошень, натиснувши відповідну кнопку на меню зверху або знизу.

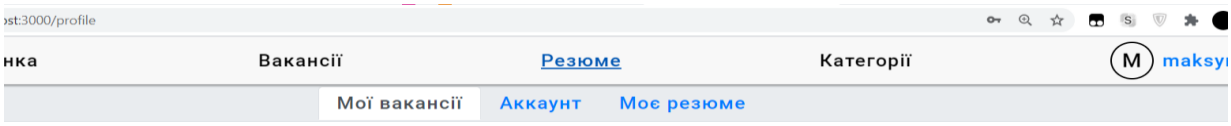


Рисунок 3.12 – демонстрація положення списку всіх резюме

Список всіх оголошень доступний навіть не авторизованим користувачам. У списку розміщені картки з основною інформацією з резюме на посилання, за допомогою якого є можливість переглянути оголошення повністю.

Елементи картки оголошення:

- Повне ім’я;
- Спеціальність;
- Досвід роботи;
- Фотографія користувача;
- Поштова скринька для зв’язку;
- Посилання на повну версію оголошення.

3.5.5. Сторінка з доступними вакансіями

Сторінка вакансії містить список всіх оголошень для пошуку робітників, що знаходяться у базі даних. Відображається список у вигляді карток з короткою та основною інформацією з вакансії.

Елементи картки оголошення:

- Категорія;
- Позиція;
- Необхідний рівень;
- Дивитись повністю - посилання на особисту сторінку оголошення.

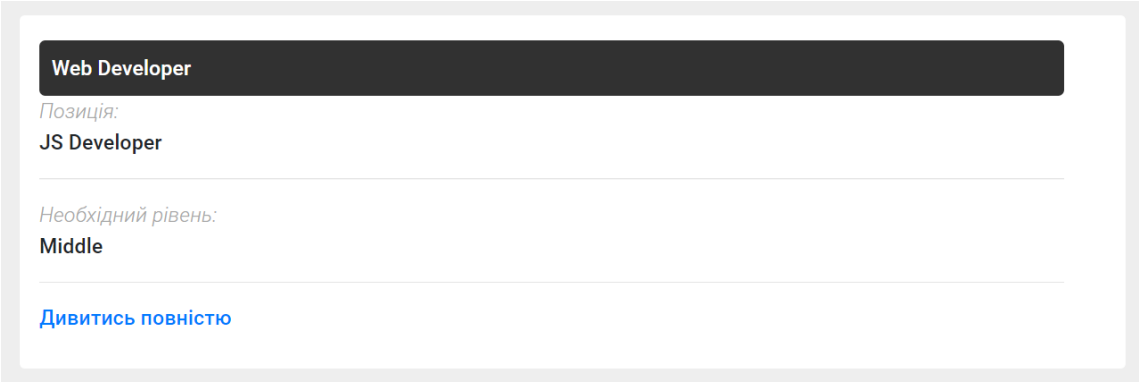


Рисунок 3.13 – Демонстрація картки певної вакансії

3.5.6. Сторінка обраного оголошення

Після переглядання списку існуючих резюме, маємо змогу відкрити сторінку обраного оголошення.

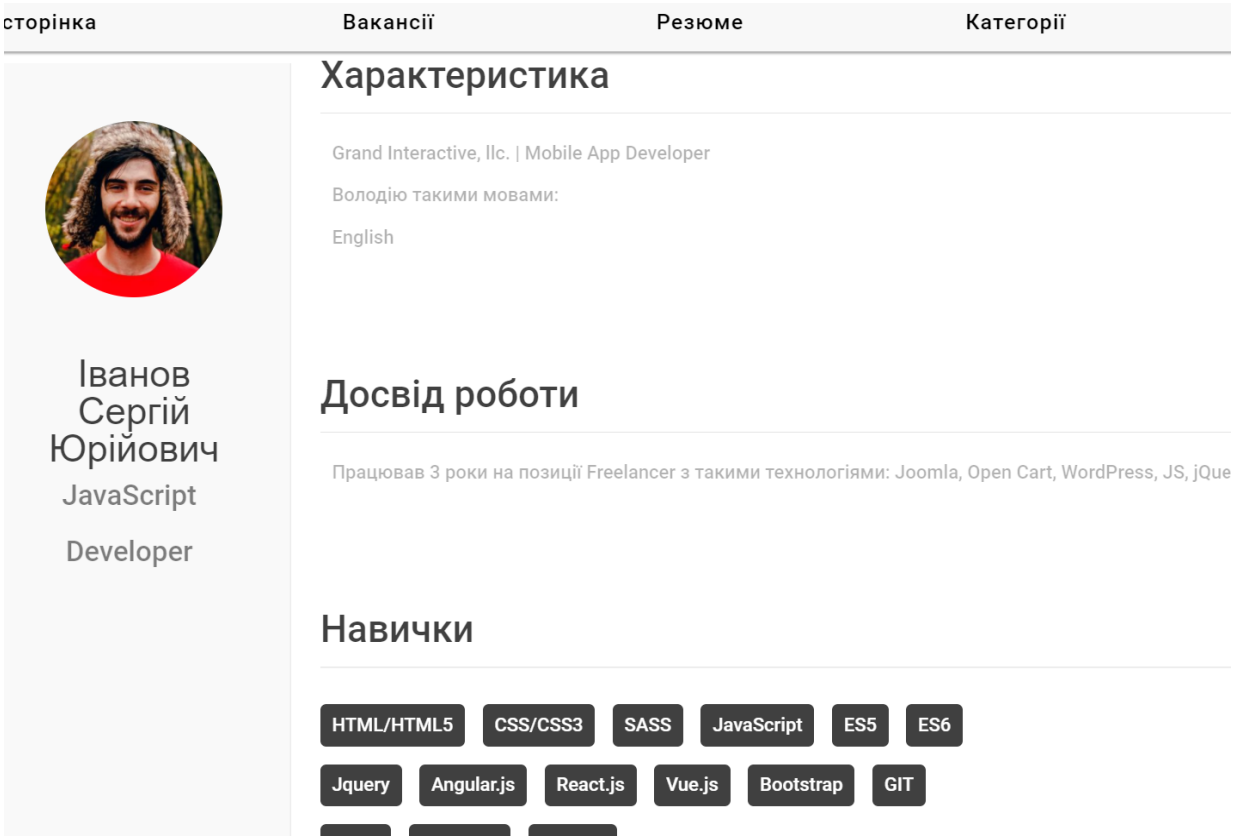


Рисунок 3.14 – демонстрація окремої сторінки з резюме

Сторінка складається з двох частин: лівої фіксованої та правої динамічної (елемент можна прокрутити, щоб побачити інформацію повністю).

Ліва частина містить у собі:

- фотографію користувача;
- повне ім’я користувача;
- спеціальність користувача.

У правій частині знаходяться розділи:

- характеристика - крім опису, містить у собі мови, якими володіє користувач;
- досвід роботи;
- навички;
- додаткова інформація - крім додаткових відомостей, є поле з контактами.

Висновки до розділу 3

У третьому розділі було розглянуто структуру веб-застосунку та етапи його розробки.

Застосунок був написаний за допомогою React - JavaScript-бібліотеки для створення користувацьких інтерфейсів. Для початку роботи використовувалась команда консолі Create React App, що значно пришвидшує процес збірки та налаштування проекту. Над збереженням даних користувачів працювала нереляційна база даних MongoDB. В окремому розділі розглянуто процес підключення до бази.

Крім того, додаток був створений за допомогою фреймворків Express.js, Bootstrap, бібліотеки React Hook Form та інструментів Nodemon, JSON Web Token, Styled Components, React Router.

На сайті реалізовані функції реєстрації та авторизації користувачів, можливість створити власні оголошення, можливість переглядати всі існуючі на сайті оголошення та пошук серед списку оголошень.

ВИСНОВОК

У результаті роботи було досліджено теоретичні основи розроблення сучасних веб-застосунків, а також всі сучасні методи веб-розробки.

Сайт будується за допомогою трьох головних складових: HTML розмітки, CSS стилів, JavaScript логіки та динаміки. Перед розробкою веб-додатку необхідно спочатку спроектувати його структуру. В межах дослідження було детально розглянуто можливості мови програмування JavaScript, включно з більшістю наявних допоміжних засобів для веб-розробки, деякі з котрих були застосовані на практиці. Після дослідження наявних варіантів, основним інструментом розробки було обрано засіб React та ще певний набір бібліотек.

У ході аналізу предметної галузі застосунка було проаналізовано існуючі системи з пошуку роботи, були зазначені переваги та недоліки відомих платформ та список вимог, яким повинен буде відповідати веб-застосунок власної розробки в межах бакалаврської роботи.

Застосунок був написаний за допомогою JavaScript-бібліотеки React, для початку роботи використовувалась команда консолі Create React App, що спростила процес створення проекту. У якості бази даних була використана нереляційна MongoDB. Крім того, додаток був створений за допомогою фреймворків Express.js, Bootstrap, бібліотеки React Hook Form та інструментів Nodemon, JSON Web Token, Styled Components, React Router.

На сайті реалізовані всі основні функції, необхідні для знаходження роботи та виконання головного завдання бакалаврської роботи: можливість реєстрації та авторизації, створення власних оголошень, перегляд існуючих оголошень та пошук серед загального списку.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Getting started with the Web *MDN Web Docs* : веб-сайт. URL: https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web (дата звернення: 13.05.2021)
2. Введение в JavaScript *Современный учебник JavaScript*: веб-сайт. URL: https://learn.javascript.ru/intro (дата звернення: 13.05.2021)
3. Getting Started *mongoose*: веб-сайт. URL: https://mongoosejs.com/docs/index.html (дата звернення: 15.05.2021)
4. The MongoDB 4.4 Manual *MongoDB*: веб-сайт. URL: https://docs.mongodb.com/manual/ (дата звернення: 15.05.2021)
5. MongoDB *Wikipedia*: веб-сайт. URL: https://en.wikipedia.org/wiki/MongoDB (дата звернення: 15.05.2021)
6. Учебник Express часть 3: Использование базы данных (с помощью Mongoose) *MDN Web Docs* : веб-сайт. URL: https://developer.mozilla.org/ru/docs/Learn/Server-side/Express_Nodejs/mongoose (дата звернення: 15.05.2021)
7. Figma *Figma*: веб-сайт. URL: https://www.figma.com/ (дата звернення: 16.05.2021)
8. Введение в Mongoose для MongoDB и Node.js *ENVATO TUTS+*: веб-сайт. URL: https://code.tutsplus.com/ru/articles/an-introduction-to-mongoose-for-mongodb-and-nodejs--cms-29527 (дата звернення: 16.05.2021)
9. Mongoose *METANIT.COM*: веб-сайт. URL: https://metanit.com/web/nodejs/6.6.php (дата звернення: 18.05.2021)
10. Bootstrap (фреймворк) *Wikipedia*: веб-сайт. URL: https://ru.wikipedia.org/wiki/Bootstrap_(%D1%84%D1%80%D0%B5%D0%B9%D0%BC%D0%B2%D0%BE%D1%80%D0%BA) (дата звернення: 18.05.2021)
11. Введение *Bootstrap*: веб-сайт. URL: https://bootstrap-4.ru/docs/5.0/getting-started/introduction/ (дата звернення: 18.05.2021)
12. A Brief Overview of React Router and Client-Side Routing *Marcella Maki*: веб-сайт. URL: https://medium.com/@marcellamaki/a-brief-overview-of-react-router-and-client-side-routing-70eb420e8cde (дата звернення: 20.05.2021)
13. Quick Start *REACT TRAINING / REACT ROUTER*: веб-сайт. URL: https://reactrouter.com/web/guides/quick-start (дата звернення: 15.05.2021)
14. Маршрутизация Определение маршрутов *METANIT.COM*: веб-сайт. URL: https://metanit.com/web/react/4.1.php (дата звернення: 15.05.2021)
15. Понимание ReactJS Router с примером на стороне клиента *betacode*: веб-сайт. URL: https://betacode.net/12137/undertanding-react-router-with-

example-on-the-client-side#:~:text=React%20Router%20%D1%8D%D1%82%D0%BE%20%D1%81%D1%82%D0%B0%D0%BD%D0%B4%D0%B0%D1%80%D1%82%D0%BD%D0%B0%D1%8F%20%D0%B1%D0%B8%D0%B1%D0%BB%D0%B8%D0%BE%D1%82%D0%B5%D0%BA%D0%B0,%D0%B2%20%D0%B2%D0%B0%D1%88%D0%B5%D0%BC%20%D0%BF%D1%80%D0%B8%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D0%B8%20%D0%BF%D0%BE%D0%BD%D1%8F%D1%82%D0%BD%D1%8B%D0%BC%20%D1%81%D0%BF%D0%BE%D1%81%D0%BE%D0%B1%D0%BE%D0%BC (дата звернення: 15.05.2021)

16. Getting started *styled components*: веб-сайт. URL: https://styled-components.com/ (дата звернення: 22.05.2021)
17. Знакомство со Styled components *Frontend Stuff*: веб-сайт. URL: https://frontend-stuff.com/blog/styled-components/ (дата звернення: 22.05.2021)
18. Express.js *Wikipedia*: веб-сайт. URL: https://uk.wikipedia.org/wiki/Express.js (дата звернення: 23.05.2021)
19. JSON Web Token *Wikipedia*: веб-сайт. URL: https://uk.wikipedia.org/wiki/JSON_Web_Token (дата звернення: 23.05.2021)
20. Express *Express*: веб-сайт. URL: https://expressjs.com/ru/ (дата звернення: 23.05.2021)
21. Introduction to JSON Web Tokens *JWT*: веб-сайт. URL: https://jwt.io/ (дата звернення: 23.05.2021)
22. MongoDB *Wikipedia*: веб-сайт. URL: https://metanit.com/web/nodejs/2.6.php (дата звернення: 25.05.2021)
23. Nodemon *METANIT.COM*: веб-сайт. URL: https://ru.reactjs.org/ (дата звернення: 25.05.2021)
24. React *Wikipedia*: веб-сайт. URL: https://uk.wikipedia.org/wiki/React (дата звернення: 25.05.2021)
25. Знакомство с JSX *React*: веб-сайт. URL: https://ru.reactjs.org/docs/introducing-jsx.html (дата звернення: 25.05.2021)
26. Getting Started *Create React App*: веб-сайт. URL: https://create-react-app.dev/docs/getting-started/ (дата звернення: 25.05.2021)
27. Основые этапы работы над веб-проектом *Tilda Education*: веб-сайт. URL: https://tilda.education/courses/web-design/basicsteps/ (дата звернення: 25.05.2021)
28. Полный гайд для веб-дизайнера: теория и 100+ инструментов *Plerdy*: веб-сайт. URL: https://www.plerdy.com/ru/blog/best-web-design-tools/ (дата звернення: 25.05.2021)

29. What is a Single Page Application? *HUSPI*: веб-сайт. URL: https://huspi.com/blog-open/definitive-guide-to-spa-why-do-we-need-single-page-applications (дата звернення: 25.05.2021)
30. 34 лучших инструмента для frontend-разработчика *Skillbox*: веб-сайт. URL: https://skillbox.ru/media/code/34_luchshikh_instrumenta_dlya_frontend_razrabotchika/ (дата звернення: 26.05.2021)
31. Single-page application *Wikipedia*: веб-сайт. URL: https://en.wikipedia.org/wiki/Single-page_application (дата звернення: 26.05.2021)
32. Список инструментов разработчика JavaScript *Tproger*: веб-сайт. URL: https://tproger.ru/translations/javascript-tool-list/ (дата звернення: 26.05.2021)
33. 9 инструментов для создания прототипа сайта *spark*: веб-сайт. URL: https://spark.ru/startup/icondesignlab/blog/31040/9-instrumentov-dlya-sozdaniya-prototipa-sajta (дата звернення: 25.05.2021)
34. Программы для веб-дизайнера: где создавать сайты? | Глава 7 *UxJournal*: веб-сайт. URL: https://ux-journal.ru/programmy-dlya-veb-dizajnera-gde-sozdavat-sajty.html (дата звернення: 25.05.2021)
35. CSS-препроцессоры *Путеводитель для новичков по CSS-препроцессору Less*: веб-сайт. URL: https://mrmlnc.gitbooks.io/less-guidebook-for-beginners/content/chapter_1/css-reprocessors.html (дата звернення: 25.05.2021)
36. CSS препроцессор *MDN Web Docs*: веб-сайт. URL: https://developer.mozilla.org/ru/docs/Glossary/CSS_preprocessor (дата звернення: 25.05.2021)
37. Препроцессоры *WebReference*: веб-сайт. URL: https://webref.ru/layout/advanced-html-css/preprocessors (дата звернення: 25.05.2021)
38. Angular vs React vs Vue 2021 *aThemes:* веб-сайт. URL: https://athemes.com/guides/angular-vs-react-vs-vue/ (дата звернення: 05.06.2021)
39. Angular vs React vs Vue - My Thoughts *Acade Mind*: веб-сайт. URL: https://academind.com/tutorials/angular-vs-react-vs-vue-my-thoughts/(дата звернення: 05.06.2021)
40. The Cost of Javascript Frameworks *Tim Kadlec*: веб-сайт. URL: https://timkadlec.com/remembers/2020-04-21-the-cost-of-javascript-frameworks/(дата звернення: 05.06.2021)
41. What Will Be The Best JavaScript Frameworks In 2021? *Codersera:* веб-сайт. URL: https://codersera.com/blog/best-javascript-frameworks/(дата звернення: 05.06.2021)

42. The 40 Best JavaScript Libraries and Frameworks for 2021 *Kinsta:* веб-сайт. URL: https://kinsta.com/blog/javascript-libraries/#the-most-popular-javascript-libraries(дата звернення: 05.06.2021)
43. jQuery *Wikipedia*: веб-сайт. URL: https://en.wikipedia.org/wiki/JQuery (дата звернення: 05.06.2021)
44. Developer Survey Results 2019 *Stack Overflow*: веб-сайт. URL: https://insights.stackoverflow.com/survey/2019#most-loved-dreaded-and-wanted(дата звернення: 05.06.2021)
45. Framework Usage Distribution in the Top 1 Million Sites *BuiltWith*: веб-сайт. URL: https://trends.builtwith.com/framework(дата звернення: 05.06.2021)
46. Find React Jobs *Indeed*: веб-сайт. URL: https://www.indeed.com/q-React-jobs.html(дата звернення: 05.06.2021)
47. jQuery – Overview *Tutorialspoint*: веб-сайт. URL: https://www.tutorialspoint.com/jquery/jquery-overview.htm (дата звернення: 05.06.2021)
48. jQuery Introduction *W3Schools:* веб-сайт. URL: https://www.w3schools.com/jquery/jquery_intro.asp (дата звернення: 05.06.2021)
49. jQuery *Wappalyzer:* веб-сайт. URL: https://www.wappalyzer.com/technologies/javascript-libraries/jquery/(дата звернення: 05.06.2021)
50. Usage statistics and market share of jQuery for websites *W3Techs:* веб-сайт. URL: https://w3techs.com/technologies/details/js-jquery(дата звернення: 05.06.2021)
51. Модель-вид-контролер *Wikipedia*: веб-сайт. URL: https://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C-%D0%B2%D0%B8%D0%B4-%D0%BA%D0%BE%D0%BD%D1%82%D1%80%D0%BE%D0%BB%D0%B5%D1%80(дата звернення: 05.06.2021)
52. AngularJS *Wikipedia*: веб-сайт. URL: https://uk.wikipedia.org/wiki/AngularJS(дата звернення: 05.06.2021)
53. AngularJS *Metanit*: веб-сайт. URL: https://metanit.com/web/angular/1.1.php (дата звернення: 05.06.2021)
54. What is Vue.js? *Vue.js:* веб-сайт. URL: https://vuejs.org/v2/guide/ (дата звернення: 05.06.2021)
55. Vue.js *Metanit*: веб-сайт. URL: https://metanit.com/web/vuejs/1.1.php(дата звернення: 05.06.2021)
56. Что такое макет сайта и для чего он нужен *Netpeak:* веб-сайт. URL: https://netpeak.net/ru/blog/chto-takoye-maket-sayta-i-dlya-chego-on-nuzhen-story/ (дата звернення: 05.06.2021)

57. JavaScript *MDN Web Docs*: веб-сайт. URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript(дата звернення: 05.06.2021)
58. Працевлаштування *Rabota:* веб-сайт. URL: https://rabota.ua/ (дата звернення: 20.05.2021)
59. Працевлаштування *Jobs*: веб-сайт. URL: https://jobs.ua/(дата звернення: 20.05.2021)
60. Працевлаштування *StackOverflow*: веб-сайт. URL: https://stackoverflow.com/jobs(дата звернення: 20.05.2021)
61. Працевлаштування *Indeed*: веб-сайт. URL: https://ua.indeed.com/(дата звернення: 20.05.2021)
62. Працевлаштування *TechCrunch*: веб-сайт. URL: https://www.crunchboard.com/jobs(дата звернення: 20.05.2021)
63. Працевлаштування *LinkedIn*: веб-сайт. URL: https://ua.linkedin.com/?trk=guest_homepage-basic_nav-header-logo(дата звернення: 20.05.2021)
64. Працевлаштування *Dice*: веб-сайт. URL: https://www.dice.com/(дата звернення: 20.05.2021)
65. Працевлаштування *GithubJobs.*: веб-сайт. URL: https://jobs.github.com/(дата звернення: 20.05.2021)
66. Працевлаштування *Work.ua*: веб-сайт. URL: https://www.work.ua/(дата звернення: 20.05.2021)

ДОДАТОК А

Демонстрація вмісту файлу підключення до бази даних

```
const express = require("express");
const config = require("config");
const mongoose = require("mongoose");
const path = require("path");
const cors = require("cors");

const app = express();

const PORT = process.env.PORT || config.get("port");

app.use(cors({ origin: true, credentials: true }));
app.use(express.json({ extended: true }));
app.use("/api/auth", require("./routes/auth"));
app.use("/api/vacancies", require("./routes/vacancies"));

async function start() {
  try {
    await mongoose.connect(config.get("db"), {
      useNewUrlParser: true,
      useUnifiedTopology: true,
      useCreateIndex: true,
    });
    app.listen(PORT, () => console.log(`Started on ${PORT}`));
  } catch (error) {
    console.log("App start error", error);
    process.exit(1);
  }
}

start();
```

Порт та посилання на базу даних в окремому файлі:

```
ATLAS_URI=mongodb+srv://root:root@cluster0.ij3ru.mongodb.net/myFirstDatabase?retryWrites=true&w=majority

PORT=5000
```

ДОДАТОК Б

Демонстрація створення моделі вакансії для зберігання даних оголошень

```
const { Schema, model, Types } = require("mongoose");

const VacancySchema = new Schema({
  name: {
    type: String,
    required: true,
  },
  description: {
    type: String,
  },
  category: {
    type: String,
  },
  level: {
    type: String,
  },
  conditions: {
    type: String,
  },
  contact: {
    type: String,
  },
  published_date: {
    type: Date,
    default: Date.now,
  },

  owner: {
    id: { type: Types.ObjectId, ref: "User" },
    name: { type: String, ref: "User" },
  },
});

module.exports = model("vacancy", VacancySchema);
```

ДОДАТОК В

Демонстрація фрагменту з обробкою даних форми створення вакансії

```
const VacancyCreationPage = () => {
  const {
    register,
    handleSubmit,
    watch,
    formState: { errors },
  } = useForm();

  const history = useHistory();

  const [createVacancy] = useFetch();

  const onSubmit = async ({ name, description, category, level,
conditions, contact }) => {
    const { response, error } = await createVacancy("post", "vacancies", {
      name,
      description,
      category,
      level,
      conditions,
      contact,
    });

    if (error != null && !response) {
      console.log("error", error);
      return;
    }
  };

return (
    <ScVacancyCreationPage.ScVacancyCreationPage>
      <div ClassName="VacCrFormGr">
        <p>Створити свою власну вакансію</p> <hr></hr>
        <Form ClassName="RegFormGr" onSubmit={handleSubmit(onSubmit)}>

          <Form.Group controlId="VacCreatePos">
            <Form.Label>Позиція</Form.Label>
            <Form.Control placeholder="Позиція"
              {...register("name", { required: true })} />
          </Form.Group>
          <hr></hr>
```

…

//таким самим чином прописуються інші поля//

…

```
          <Button variant="dark" type="submit">
            Create vacancy
          </Button>
        </Form>
      </div>
    </ScVacancyCreationPage.ScVacancyCreationPage>
```

ДОДАТОК Г

Демонстрація виводу вакансій з бази даних у картки за допомогою map()

```
const NewVacs = () => {
  const [getVacanciesRequest, { data: vacancies }] = useFetch();

  const [sortedVacancies, setSortedVacancies] = useState(vacancies);

  useEffect(() => {
    getVacanciesRequest("get", "vacancies");
  }, [getVacanciesRequest]);

  useEffect(() => {
    if (vacancies == null) return;

    setSortedVacancies(vacancies);
  }, [vacancies]);

  if (sortedVacancies == null) return null;

  return (
    <main className="newvacsbody">
      <Search
        searchData={vacancies}
        searchKeys={["name", "category"]}
        onResult={(res) => setSortedVacancies(res)}
      />
      {sortedVacancies?.map((vacancy) => (
        <div className="NewVacsCard" key={vacancy._id}>
          <div className="NewVacsCard_Category">
            <h2>Web Developer</h2>
          </div>
          <p className="grtext">Позиція:</p>
          <div className="NewVacsCard_Main">
            <b>{vacancy.name}</b>
          </div>
          <hr></hr>
          <p className="grtext">Необхідний рівень:</p>
          <b>{vacancy.level}</b> <hr></hr>
          <Link to={`vacancy/${vacancy._id}`}
className="NewVacsCard_Link">
            Дивитись повністю
          </Link>
        </div>
      ))}
    </main>
  );
};

export default NewVacs;
```