Київський національний університет імені Тараса Шевченка

Факультет інформаційних технологій

Кафедра програмних систем і технологій

УДК 004.41

На правах рукопису

ВИПУСКНА КВАЛІФІКАЦІЙНА БАКАЛАВРСЬКА РОБОТА

Тема: “HR-Portal”

Спеціальність – 121 “Інженерія програмного забезпечення”

ПОЯСНЮВАЛЬНА ЗАПИСКА

ВКБР.ІПЗ - 22.00.00.000 ІПЗ

Студент

ІПЗ-42 /Павло ПШЕНИШНЮК/

Науковий керівник

к.ф.-м.н., доц. /Оксана КОВТУН/

Консультант з питань нормоконтролю

/Тамара ЧАПОВСЬКА/

Допускається до захисту Зав. каф.

д.т.н.,проф. /Олексій БИЧКОВ/

Київ – 2021

Рішенням Екзаменаційної комісії

випускна кваліфікаційна робота студента

_____

захищена з оцінкою

_____

Голова Екзаменаційної комісії

професор, доктор техн. наук Бондарчук А.П.

Київський національний університет імені Тараса Шевченка

Факультет інформаційних технологій

Кафедра програмних систем і технологій

Освітньо-кваліфікаційний рівень бакалавр

Спеціальність 121 “Інженерія програмного забезпечення”

ЗАТВЕРДЖЕНО

Зав. кафедри програмних систем і технологій

_____ (Бичков О.С.)

ЗАВДАННЯ

НА ВИПУСКНУ КВАЛІФІКАЦІЙНУ БАКАЛАВРСЬКУ РОБОТУ СТУДЕНТУ

Пшенишнюку Павлу Олександровичу

1. Тема випускної кваліфікаційної бакалаврської роботи “HR-Portal”

керівник проекту (роботи) Ковтун Оксана Іванівна, к.ф.-м.н., доцент

затверджені наказом вищого навчального закладу від „11” листопада 2020р. №6

2. Строк здачі студентом закінченої роботи „__” _____ 2021 р.

3. Вхідні дані до проекту (роботи): підручники, навчальні посібники, офіційна документація, статті зарубіжних авторів, інтернет-ресурси.

4. Зміст розрахунково - пояснювальної записки (перелік питань, які потрібно розробити)

1. Аналіз предметної області

2. Огляд технологій

3. Реалізація програмного комплексу

5. Перелік графічного матеріалу (з точним забезпеченням обов’язкових креслень)

1. Загальний вигляд веб-додатку на ранніх етапах розробки (рис. 1.1., ст. 17)

2. Дизайн головної сторінки (ч. 1) (рис. 1.2., ст. 18)

3. Дизайн головної сторінки (ч. 2) (рис. 1.3., ст. 18)

4. Порівняння пакетів популярних фреймворків/бібліотек JS (рис. 2.1., ст. 27)

6. Консультанти розділів проекту (роботи)

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
Розділ 1. Аналіз предметної області	Ковтун О.І.	10.11.2020	20.01.2021
Розділ 2 Огляд технологій	Ковтун О.І.	25.01.2021	02.03.2021
Розділ 3. Реалізація програмного комплексу	Ковтун О.І.	02.03.2021	20.05.2021

7. Дата видачі завдання _____

Керівник _____ (Оксана КОВТУН)

Завдання прийняв до виконання _____ (Павло ПШЕНИШНЮК)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів бакалаврської роботи	Термін виконання етапів роботи	Відмітка про виконання
1.	Уточнення постановки задачі	10.11.2020-02.12.2020	Виконано
2.	Аналіз літератури	02.12.2020 - 11.01.2021	Виконано
3.	Огляд та аналіз існуючих методів, концепцій та алгоритмів вирішення завдання	11.01.2021 - 25.01.2021	Виконано
4.	Побудова алгоритмічної моделі основних процесів	25.01.2021 - 02.03.2021	Виконано
5.	Розробка програмного забезпечення	02.03.2021 - 05.04.2021	Виконано
6.	Тестування розробленого програмного забезпечення	05.04.2021 - 20.05.2021	Виконано
7.	Оформлення і друк пояснювальної записки	20.05.2021 - 01.06.2021	Виконано
8.	Отримання рецензії	02.06.2021 - 05.06.2021	Виконано
9.	Оформлення презентації	05.06.2021-09.06.2021	Виконано
10.	Затвердження пояснювальної записки роботи виконувачем обов'язки завідувача кафедри	_____	
11.	Захист дипломної роботи	22.06.2020	

Студент-бакалавр _____ (Павло ПШЕНИШНЮК)

Керівник роботи _____ (Оксана КОВТУН)

АНОТАЦІЯ

Випускна кваліфікаційна бакалаврська робота: 65 с., 17 рис., 2 додат., 20 джерел.

Тема: HR-Portal.

Об'єкт дослідження: програмний комплекс, що надає можливість оцінки роботи HR/Рекрутерів з боку користувачів та шукачів роботи.

Мета роботи: розробка веб-додатку HR-Portal. Надати можливість шукачам роботи попередньо проаналізувати оцінку HR, перед відгуком на вакансію. Надати HR/Рекрутерам можливість підвищити свій авторитет та авторитет компанії у сфері IT.

Предмет дослідження: веб-технології, дослідження впливу HR/Рекрутерів на вибір місця роботи шукачів.

Результати дослідження: Розглянуто методики розробки програмного забезпечення для HR-Portal. Запропоновано алгоритмічно-оптимізований підхід створення веб-додатку. Здійснено докладний аналіз впливу HR/Рекрутерів на вибір місця роботи шукачів. Програмно реалізовано веб-додаток HR-Portal.

Висновок:

В результаті досліджень було розроблено програмний комплекс, що надає можливість оцінки роботи HR/Рекрутерів з боку користувачів та шукачів роботи.

АНОТАЦИЯ

Выпускная квалификационная бакалаврская работа: 65 с., 17 рис., 2 прилож., 20 источников.

Тема: HR-Portal.

Объект исследования: программный комплекс, предоставляющий возможность оценки работы HR/Рекрутеров со стороны пользователей и соискателей.

Цель работы: разработка веб-приложения HR-Portal. Предоставить возможность соискателям предварительно проанализировать оценку HR/Рекрутера, перед откликом на вакансию. Предоставить HR/Рекрутерам возможность повысить свой авторитет и авторитет компании в сфере IT.

Предмет исследования: веб-технологии, исследования влияния HR/Рекрутеров на выбор места работы соискателей.

Результаты исследования: Рассмотрены методики разработки программного обеспечения для HR-Portal. Предложено алгоритмически-оптимизированный подход создания веб-приложения. Осуществлен подробный анализ влияния HR/Рекрутеров на выбор места работы соискателей. Программно-реализовано веб-приложение HR-Portal.

Вывод:

В результате исследований был разработан программный комплекс, предоставляющий возможность оценки работы HR / Рекрутеров со стороны пользователей и соискателей.

ANNOTATION

Graduation qualification bachelor's thesis: 65 p., 17 figures, 2 additions, 20 sources.

Topic: HR-Portal.

Object of research: Software package that provides an opportunity to evaluate the work of HR / Recruiters by users and applicants.

Purpose: Development of the web application HR-Portal. Provide applicants the opportunity to pre-analyze the HR evaluation, before responding to the vacancy. Give HR / Recruiters the opportunity to increase their credibility and the company's credibility in the field of IT.

Subject of research: Web technologies, research of the HR / Recruiters' influence on the choice of applicants.

Work results: Reviewed methods of software development for HR-Portal. Proposed an algorithmically optimized approach of creating a web application. Made a detailed analysis of the influence of HR/Recruiters on the choice of applicants. Programmatically developed the web application HR-Portal.

Conclusion:

As a result of the research, a software package that provides an opportunity to evaluate the work of HR / Recruiters by users and applicants, was developed.

ЗМІСТ

ВСТУП

Актуальність роботи

HR-портал – це веб-додаток для тих, хто шукає роботу і для тих, хто цю роботу пропонує. Але перш за все, головна ціль створення цього проекту – допомога шукачам переглянути правдиві відгуки про першу людину, яку вона побачить перед собою у компанії при співбесіді – HR-а/Рекрутера.

Змоделюємо ситуацію: ви знаходитесь у пошуку роботи, та переглядаєте значну кількість вакансій, куди надсилаєте своє резюме, якщо вакансія задовольняє ваші вимоги. Потім ви чекаєте відповіді від компанії, і в 90% випадків вам відповідає HR (якщо вам повезе) і назначає онлайн або живу співбесіду. Ви збираєтесь з силами, повторюєте матеріал та вирушаєте на співбесіду. Там вас зустрічає обличчя компанії – HR. Це та людина, яка повинна зв’язувати компанію та претендентів на роботу, на її плечах лежить велика задача – надати претенденту гарне перше враження про компанію, про її цінності і атмосферу всередині колективу. Рішення шукача щодо роботи в цій компаній насамперед залежить від враження, яке створить для нього HR. Але в наші дні, не кожен рекрутер це розуміє, і часто буває, що поведінка HR на співбесіді або після її завершення залишає бажати кращого. І після цього перше враження претендента може зіпсуватись, що вплине на його рішення чи хоче він працювати в компанії, якості якої не задовольняє його особистим потребам. І звичайно, після проходження співбесіди кандидат очікує отримати фідбек, незважаючи на позитивну або негативну відповідь. Але й на таке здатен не кожен рекрутер в наш час. Таким чином компанія, сама не знаючи про це, може втратити потенційного професіонала у своїй команді.

HR-портал – це унікальний спосіб уникнути таких ситуацій та бути в курсі всього про компанію, рекрутерів, атмосферу, яка на нього чекає та

заздалегідь бути готовим, або взагалі не розглядати цю компанію в якості майбутнього місця працевлаштування.

Задачі, які повинен вирішувати проект

- Вільний доступ до всіх HR:

Кожен користувач/кандидат повинен мати змогу написати свій особистий відгук про виконану роботу конкретного рекрутера та надати йому свою оцінку. Відгуки будуть переглядатися на наявність некоректних висловлювань, та за потреби видалятись/редагуватись. Оцінки від усіх користувачів будуть додаватися, та з них буде формуватися середній рейтинг рекрутера.

- Рейтинг «Топ-50 HR України»

Кожен користувач веб-додатку матиме можливість переглянути рейтинг «Топ-50 HR України», який буде формуватися виходячи з особистого рейтингу кожного конкретного HR, та буде оновлюватись кожен місяць.

- Новини

На головній сторінці веб-додатку будуть відображатись головні новини світу IT.

- Статистика заробітних плат

Окрема сторінка, на якій користувачі зможуть дізнатись статистику заробітних плат у світі IT в Україні у кожному кварталі року.

Потенційні користувачі

- Люди, перебуваючі у пошуку роботи

Кожна людина, яка перебувала у пошуку роботи хоч раз замислювалась, що її чекає на майбутній співбесіді. HR-Portal дозволяє відкрити завісу цієї таємниці і прийти на співбесіду підготовленим.

Кожен користувач зможе дізнатись усі відгуки про HR в компанії, в якій він бажає отримати роботу та зробити для себе висновки.

- Компанії

Компанія зможе переглядати відгуки і рейтинг своїх працівників, і робити для себе висновки. Кожна компанія хоче знати, чи втрачає вона потенційно дуже цінні кадри через некомпетентність та непрофесійність рекрутерів.

- HR/Рекрутери

HR/Рекрутерам надається можливість підняти свою репутацію та відповідно репутацію своєї компанії в очах майбутніх кандидатів.

Порівняння роботи з відомими схожими проектами

Проаналізувавши всі схожі за типом сайти ми вирішили вдосконалити деяку їх функціональність.

На сьогоднішній день не існує порталу для спільноти IT-рекрутерів України. А як відомо, HR-ів в нинішньому IT-просторі України велика кількість, яка тільки зростає.

Унікальність проекту полягає в тому, що кожна людина, яка перебуває в пошуку роботи, може переглянути рейтинг і відгуки про людину, яка стане першою для неї в співбесіді в новій компанії.

IT-рекрутери зможуть заробляти собі авторитет серед претендентів, працівників та компаній.

Компанії зможуть дізнаватися оцінку роботи своїх робітників, та бути в курсі, чи втрачають вони потенційно цінні кадри через недостатньо професійну роботу IT-рекрутерів.

Мета і задачі дослідження

До початкових умов в контексті дипломної роботи ми можемо віднести розуміння як працюють сучасні веб-технології.

До очікуваних результатів проекту можна віднести наступні пункти:

- Проаналізувати сучасні тренди у світі веб-дизайну та формуванні структури та макетів веб-додатків
- Обрати та обґрунтувати обрані технології та інструмнти при створенні веб-додатку.
- Винести переваги та недоліки обраного архітектурного шаблону.
- Розробити архітектуру модулів (компонентів) до веб-додатку HR-Portal, та обміну даних між цими модулями.
- Програмно реалізувати веб-додаток за обраними технологіями.
- Проаналізувати виконану роботу, знайти його переваги та недоліки та розписати як можна покращити проект.

Об'єкт дослідження – програмний комплекс, що надає можливість оцінки роботи HR/Рекрутерів з боку користувачів та шукачів роботи.

Предмет дослідження – веб-технології, дослідження впливу HR/Рекрутерів на вибір місця роботи шукачів.

Методи дослідження - В якості методу дослідження був використаний класичний метод юзабіліті-тестування «Думки вголос».

Новизна одержаних результатів

В ході дослідження схожих за типом мереж в інтернеті були отримані наступні результати: запропоновано новий у своєму роді веб-додаток для оцінки роботи HR/Рекрутерів.

Практичне значення одержаних результатів

Розроблений веб-додаток може бути використаний при пошуку користувачами нової роботи, щоб мати змогу оцінити роботу HR/Рекрутера, на співбесіду до якого потрапить.

Особистий внесок студента

Основним результатом є програмний комплекс веб-додатку, алгоритми роботи та весь функціонал програми. Основні частини вихідного коду програмної реалізації можна знайти у **Додатках**.

Структура та обсяг роботи

Робота викладена на 65 сторінках друкованого тексту, який складається із вступу, трьох розділів, висновків, списку використаних джерел (20 найменувань). Робота містить 17 рисунків та 2 додатки.

РОЗДІЛ 1

АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Аналіз створення дизайну веб-додатків

Як замовник, ми розробляємо дизайн самостійно, виходячи з наших вподобань, та враховуючи тренди у світі веб-дизайну 2021 року.

Дизайн – це один з найголовніших компонентів веб-додатку, тому що перше, на що звертає увагу користувач – це зовнішній вигляд, і вже потім на контент.

При розробці дизайну необхідно керуватись основними правилами веб-дизайну у наші часи, а саме [11]:

- Дизайн повинен бути єдиним, незалежно від платформи

Наш веб-додаток будуть відвідувати з різних пристроїв: стаціонарного комп'ютера або ноутбука, планшета, телефону. Важлива частина роботи над UX полягає в наданні кожному відвідувачу однаковою версії веб-додатку, незалежно від пристрою.

- Проста і зрозуміла навігація

Навігація - це ключове поняття в зручності призначеного для користувача інтерфейсу. Треба пам’ятати: «неважливо, наскільки хороший ваш веб-додаток, якщо користувачі не можуть в ньому розібратися». Тому навігація у веб-додатку повинна бути: простою, зрозумілою, однаковою.

- Колір посилань

Посилання - ключовий елемент в процесі навігації. Якщо не змінювати колір відвіданого посилання, користувач може випадково перейти по ній знову. Розуміння, де користувач вже був і де він зараз, допомагає вирішити, куди йти тепер.

- Простий пошук по сторінкам

Вперше переглядаючи веб-додаток, користувач лише пробігає сторінки очима, не вчитуючись в текст. Йому потрібно знайти рішення своєї проблеми, а дизайнер повинен йому в цьому допомогти через правильну візуальну ієрархію. Візуальна ієрархія - розстановка або подання елементів по їх важливості; наприклад, куди погляд падає спочатку, куди потім і т. д.

- Перевірка посилань

Користувач втрачає довіру до веб-додатку кожен раз, коли по посиланню вилітає помилка 404 (неіснуюча сторінка). Клікнувши по посиланню, користувач чекає вирішення проблеми, а не повідомлення про помилку.

- Клікабельні елементи

Користувач визначає властивості об'єкта по його виду. Підкреслені слова без посилань або елементи без гіперпосилання запросто збивають відвідувачів з пантелику. Користувачі повинні розуміти, де на сторінці статичний контент, а де динамічний, який можна розглянути при натисканні або натискання на екран. Активні елементи повинні виділятися.

Враховуючи всі ці правила, можна розробити унікальний та правильний з точки зору user-friendly (дружній до користувача) інтерфейс веб-додатку.

На перших етапах розробки дизайн мав такий вигляд:

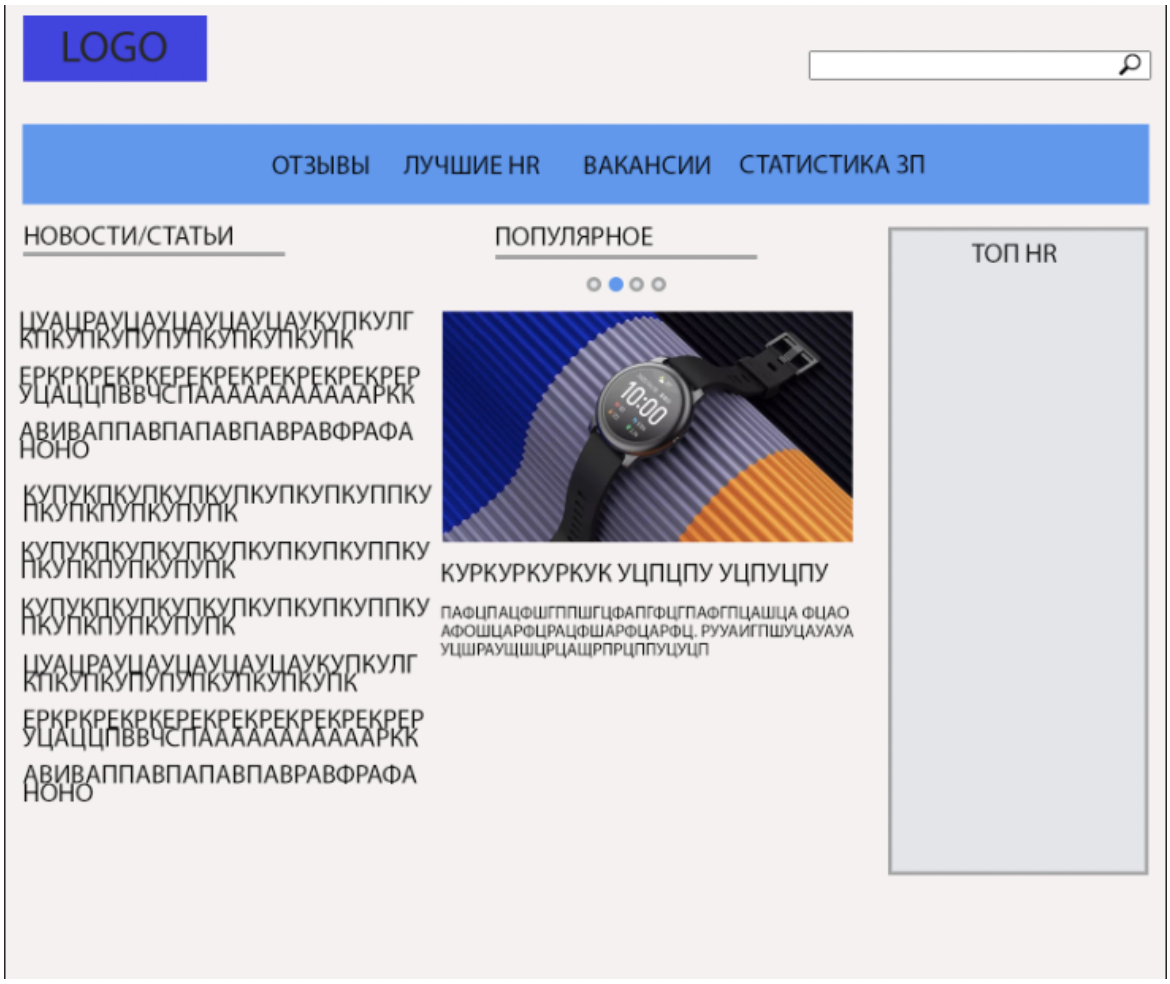

Рис. 1.1. Загальний вигляд веб-додатку на ранніх етапах розробки.

Після повного аналізу трендів світового веб-дизайну 2021 року, та загальних особистих вражень ми вирішили зробити дизайн біль сучасним.

На кінцевих етапах розробки дизайну був прийнятий та затверджений остаточний варіант дизайну: основні кольори, розташування блоків та анімація. Остаточний варіант головної сторінки матиме наступний вигляд:

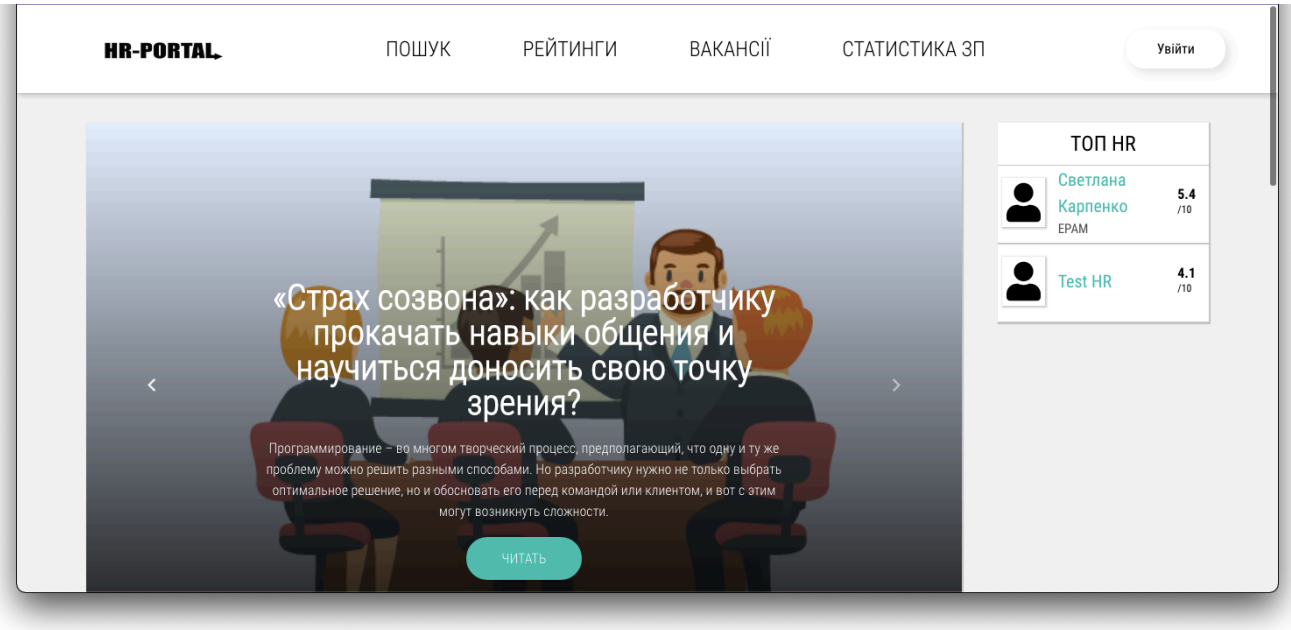

Рис. 1.2. Дизайн головної сторінки (ч. 1).

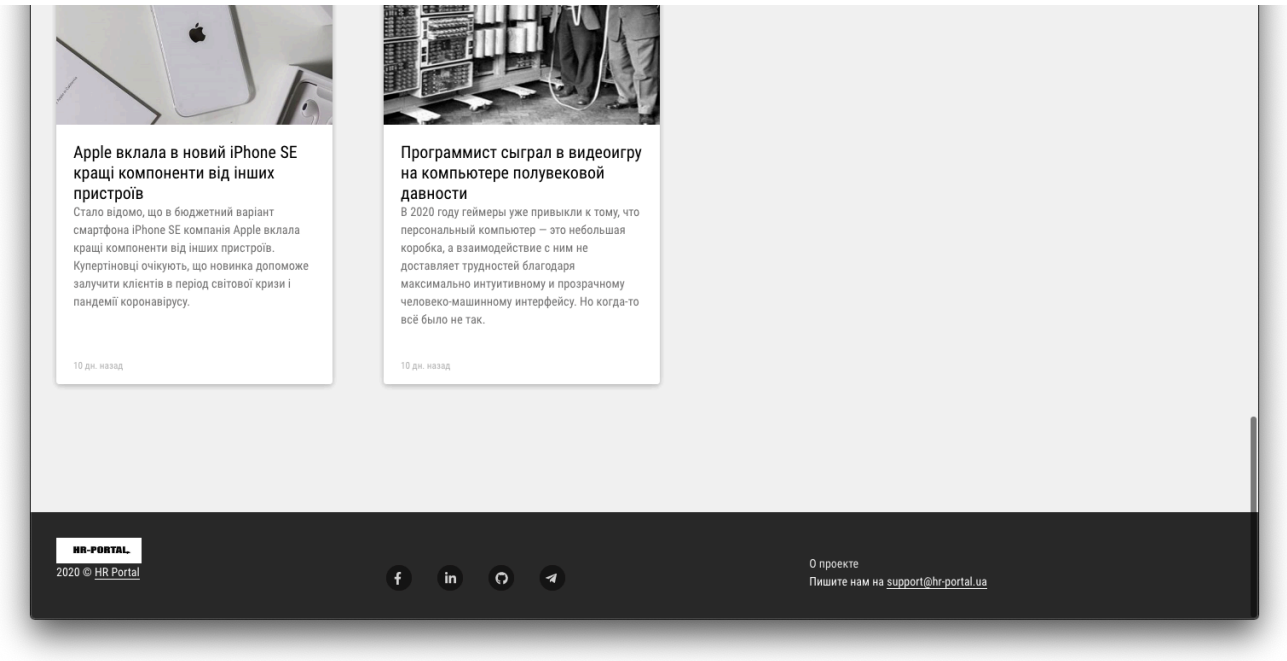

Рис. 1.3. Дизайн головної сторінки (ч. 2).

1.2 Аналіз юридичних обов’язків при створенні веб-порталу

Користувацька угода виступає як юридично обов’язковий контракт між веб-додатком та його користувачами.

Це угода, яка встановлює правила та вказівки, на які користувачі повинні погодитись і дотримуватися їх для використання та доступу до веб-сайту або мобільного додатка.

У цій угоді ми можемо включити необхідні розділи, щоб інформувати користувачів про вказівки щодо використання нашого веб-додатку, а також що трапляється, якщо користувачі зловживатимуть нашим веб-додатком.

Користувацька угода [12]:

1. Вступ

1.1 Договір

Ви погоджуєтеся з тим, що натискаючи «Створити акаунт», «Зареєструватися» і т.д., реєструючись, входячи в систему, ви погоджуєтеся укласти юридичний договір з HR-Portal (навіть якщо ви використовуєте наш веб-додаток від імені компанії). Якщо ви не приймаєте умови цього договору (далі - «Договір» або «Угода»), не натискайте «Створити акаунт» і т. д., не отримуйте доступ до жодної з наших Послуг і не використовуйте їх іншим чином. Якщо ви побажаєте розірвати цей договір, ви можете зробити це в будь-який час, закривши свій обліковий запис і утримавшись від доступу до Послуг HR-Portal і їх подальшого використання.

1.2 Користувачі та відвідувачі

Реєструючись та приєднуючись Послуг HR-Portal, ви отримуєте статус Користувача. Якщо ви вирішуєте не реєструватися для використання Послуг

HR-Portal, ви можете отримувати доступ до окремих функцій в якості «Відвідувача».

2. Обов’язки

2.1 Умови надання Послуг

Надання послуг передумовлюється набуттям користувача повних 18 років.

Ви погоджуєтеся та засвідчуєте те, що для використання Послуг: (1) ви повинні досягти «мінімального віку» (див. Визначення нижче); 2) у вас буде тільки один обліковий запис HR-Portal, зареєстрований на ваші справжні ім'я та прізвище; 3) на вас не накладено ніяких обмежень на використання Послуг з боку HR-Portal. Створення облікового запису з використанням завідомо неправдивої інформації (включаючи створення облікових записів, зареєстрованих на інших осіб і осіб молодше 18 років) є порушенням умов HR-Portal.

«Мінімальний вік» становить 18 років. Однак якщо застосовне законодавство передбачає, що для правомірного надання компанією HR-Portal Послуг без згоди батьків (включаючи використання персональних даних) Учасник повинен бути старше, тоді в якості мінімального вік повинен використовуватися вік, передбачений чинним законодавством.

2.2 Ваш обліковий запис

Учасники є власниками облікового запису. Як Учасник ви берете на себе наступні зобов'язання: 1) використовувати надійний пароль і зберігати його в таємниці; 2) не передавати іншим особам ніякі дані зі свого облікового запису (наприклад, контакти); 3) виконувати вимоги закону і керуватися списком «Дозволені і заборонені дії». Ви несете відповідальність за будь-які дії, що здійснюються через ваш обліковий запис, за винятком випадків, коли

така обліковий запис була вами закрита або ви повідомили про неправомірне використання вашого профілю іншими особами.

2.3 Обмін і доступ до інформації

Приймаючи цю угоду, Ви погоджуєтесь на те, що Послуги HR-Portal допускають обмін інформацією різними способами, наприклад, за допомогою вашого профілю, статей, посилань на новинні статті, оголошень про вакансії. Інші Учасники зможуть залишати свої власні відгуки про вас, та вашу роботу, лише з професійної точки зору. Всі коментарі, які будуть містити персональні образи, торкатися приватного життя, будуть в негайному порядку видалятися. Інші Учасники, Відвідувачі і інші особи (в тому числі за межами Послуг) можуть переглядати інформацію і контент, якими ви ділитеся або публікуєте.

3. Загальні умови користування Веб-додатком

3.1 Для вільного перегляду Контенту, що розміщений на HR-Portal, мати створений акаунт, або бути залогіненим у систему не є обов’язковим.

3.2 Будь-яке інше застосування функціональних можливостей Веб-Додатку, можливе тільки після створення акаунту та процесу логіну до Веб-додатку відповідно до встановлених HR-Portal правилами.

3.3 Перелік функціональних можливостей Веб-Додатку, використання яких вимагає попередньої реєстрації та / або авторизації, а також прийняття в необхідних випадках додаткових документів на використання Сервісів, визначається за одноосібним розсудом HR-Portal і може час від часу змінюватися.

3.4 Після створення унікального акаунту, що пов’язується з Користувачем в Веб-додатку. Унікальний акаунт є ключовим фактором для використання більшості функціональних можливостей Веб-додатку.

3.5 При створенні акаунту Користувач повинен надати валідні дані про себе і підтримувати ці дані актуалізованими. Якщо Користувач надає невірну

інформацію або у HR-Portal є підстави вважати, що надана Користувачем інформація неповна або недостовірна, HR-Portal має право на свій розсуд заблокувати або видалити акаунт Користувача, а також відмовити Користувачеві в використанні Веб-додатку повністю або в певній частині .

3.6 На вимогу команди HR-Portal кожен Користувач зобов’язаний надати дані, які були верифіковані при створенні акаунту, та разом з цим, відповідні документи, підтверджуючі ці дані.

3.7 Всі дані, що були вказані при створенні акаунту Користувача не переходять в руки третіх осіб, оброблюються лише системою HR-Portal відповідно Політики конфіденційності.

3.8 При реєстрації Користувач самостійно вибирає собі логін (унікальне символьне ім'я облікового запису Користувача) і пароль для доступу до Особистого кабінету. **3.9** Зареєстрований Користувач самостійно визначає порядок використання Особистого кабінету і інших функціональних можливостей Веб-додатку, включаючи умови використання відповідного Сервісу, які однак ні за яких умов не можуть суперечити цій Угоді за винятками, встановленими в додаткових документах.

4. Дозволені й заборонені дії в HR-Portal

4.1 Дозволені дії

Ви погоджуєтеся з тим, що ви зобов'язані:

a) дотримуватися чинного законодавства, включаючи в числі іншого закони про недоторканність приватного життя, закони про інтелектуальну власність, закони про боротьбу зі спамом, закони про експортний контроль, податкове законодавство та нормативні вимоги;
b) надавати HR-Portal достовірну інформацію і підтримувати її в актуальному стані;
c) використовувати в своєму профілі ваші справжні ім'я та прізвище;

d) використовувати Веб-додаток професійно і для ділових цілей.

4.2 Заборонені дії

Ви надаєте згоду на те, що ви забов'язуєтесь не:

a) створювати в HR-Portal фальшиві облікові записи, вводити в оману щодо своєї особи, створювати профіль Учасника для кого-небудь, крім себе (реальної особи), а також використовувати або намагатися використати чужий обліковий запис;
b) розробляти, підтримувати або використовувати програмні продукти, пристрої, скрипти, програми-роботи або інші засоби і процеси (включаючи програми-обхідники, Plug-in і надбудови браузерів, а також будь-які інші технології) для веб-скрейпінга наданих Послуг займатись копіюванням наданих веб-додатком Послуг.
c) блокувати будь-які функції системи безпеки або обходити будь-які елементи управління доступом або обмеження на використання послуг (наприклад, використовувати великі букви при пошуку за ключовими словами або перегляді профілів);
d) розкривати інформацію, не заручившись згодою на її розкриття (наприклад, конфіденційну інформацію інших осіб, в тому числі свого роботодавця);
e) порушувати права на інтелектуальну власність інших осіб, в тому числі авторські права, права на патенти і товарні знаки, комерційні таємниці, і інші майнові права.
f) порушувати право інтелектуальної власності та інші права HR-Portal, в тому числі копіювати або поширювати навчальні відео- або інші матеріали HR-Portal; копіювати або поширювати технології HR-Portal, крім випущених за ліцензією з відкритим вихідним кодом; використовувати слово «HR-Portal» або логотипи HR-Portal в будь-

якому фірмовому найменуванні, повідомленні електронної пошти або URL-адресу, крім випадків, коли Правилами використання бренду передбачено інше;

g) розміщувати контент, що містить програмні віруси, «черв'яки» або будь-який інший шкідливий код;
h) здійснювати розтин технології, декомпілювання, розбір, розшифровку або інші способи отримання вихідного коду Послуг або будь-яких пов'язаних з ними технологій, крім технологій з відкритим вихідним кодом;
i) давати підстави припускати або заявляти про наявність у вас партнерських відносин з HR-Portal або про схвалення та підтримку вашої діяльності компанією HR-Portal без явно вираженої згоди HR-Portal;
j) надавати в тимчасове користування, здавати в оренду, обмінювати, продавати / перепродувати або іншим чином монетизувати Послуги, або пов'язані з ними дані, або доступ до них без згоди HR-Portal;
k) використовувати програми-роботи або інші автоматизовані методи для доступу до Послуг, додавання або завантаження контактів, відправки або переадресації повідомлень;
l) відслідковувати доступність, працездатність або функціональність Послуг для цілей конкуренції;
m) відтворювати, створювати дзеркала або іншим способом імітувати зовнішній вигляд або функціонування Послуг;
n) виконувати накладення або іншим чином змінювати Послуги HR-Portal або їх зовнішній вигляд (наприклад, за допомогою вставки елементів в Послуги або видалення, приховування або затемнення рекламного оголошення, яке відображається в рамках Послуг);

о) перешкоджати нормальному функціонуванню Послуг або створювати надмірне навантаження на них (наприклад, використовуючи спам, атаки типу «відмова в обслуговуванні» (DoS), віруси, ігрові алгоритми);

1.3 Висновки до розділу

В даному розділі був проведений аналіз сучасних трендів веб-технологій, і, як наслідок, був розроблений макет дизайну різноманітних сторінок веб-додатку. Також, було проаналізована юридична сторона питання відносин між сайтом та його користувачами, і, як наслідок, створено повноцінну користувацьку угоду для веб-додатку HR-Portal.

РОЗДІЛ 2

ОГЛЯД ТЕХНОЛОГІЙ

2.1 Архітектура фронтенд частини веб-додатку

Як нам відомо, веб-додаток складається з 3 основних компоненти – HTML, CSS, JavaScript (далі – JS). HTML призначений для створення умовного каркасу веб-додатку, CSS оформлює цей каркас у приємний для користувача вигляд, а JS оживляє все це. Можна сказати, що JS – це серце веб-додатку. І на цьому етапі необхідно обрати механізм функціонування цього серця, а саме – фреймворк або бібліотеку.

В наш час писати веб-додатки, сайти, і все, що пов’язано з інтернетом лише на чистому JS не є доцільним рішенням, тому що на даний момент для розробки клієнтської частини веб-додатку існує велика кількість різноманітних фреймворків та бібліотек, які покращують і прискорюють цей процес. Всі вони покликані для оптимізації програмного забезпечення (далі – ПЗ) та підвищення ефективності роботи веб-додатку. Але найголовніша, фундаментальна причина існування фреймворків полягає в тому, що вони допомагають вирішувати завдання синхронізації призначеного для користувацького інтерфейсу і внутрішнього стану програми. Це - надзвичайно складна і важлива задача.

2.1.1 Чому розробники використовують фреймворки/бібліотеки [1]

- **Час-гроші**. Створені фреймворки дозволяють створити миттєве відчуття прогресу з самого початку, чого клієнт прагне з першого дня. Крім того, чим швидше ви розвиваєтесь, тим більше грошей ви заробляєте, оскільки час, що звільняється фреймворком, може бути перенаправлений на отримання більшої продуктивності.

- **Підтримка ком'юніті**. Розробка, особливо фріланс, може бути важкою, оскільки ви занурені у віртуальний світ, і якщо ви єдиний веб-розробник інтерфейсу в команді, це означає, що ви єдиний, хто має досвід, щоб знайти рішення. Але якщо фронтенд-фреймворк, який ви використовуєте, має надійну підтримку, то з іншого боку світу знайдеться хтось, хто зіткнувся з тією ж проблемою і, можливо, зможе вам допомогти.
- **Стандарти прекрасні**. Фреймворки забезпечують стандарт з моменту їх встановлення, спрямовуючи вас на роздуми та кодування певним чином. Ви можете просто стежити за тим, як все робиться в рамках. Це полегшує спільну роботу.

2.1.2 Вибір фреймворка/бібліотеки JS для графічного інтерфейсу

На даний момент у світі фронтенд розробки існує дуже велика кількість фреймворків та бібліотек. Топ-5 найпопулярніших – це React, Vue, Angular, Ember та Backbone.

Переглянемо порівняння пакетів цих інструментів [2]:

	stars	forks	issues	updated	created	size
angular	59604	28878	464	Oct 22, 2019	Jan 6, 2010	minzipped size 61.9 KB
ember-source	21253	4196	326	Nov 7, 2019	May 26, 2011	minzipped size 325.2 KB
react	139001	26379	902	Nov 6, 2019	May 24, 2013	minzipped size 2.6 KB
vue	151663	22554	395	Nov 5, 2019	Jul 29, 2013	minzipped size 22.8 KB
backbone	27552	5675	88	Nov 6, 2019	Oct 1, 2010	minzipped size 13.5 KB

Рис. 2.1. Порівняння пакетів популярних фреймворків/бібліотек JS.

Як ми можемо бачити, на даний момент у світі першим по кількості завантажень є React, але за кількістю вподобань користувачів лідерство займає Vue.

Також переглянемо обсяги завантажень пакетів цих інструментів:

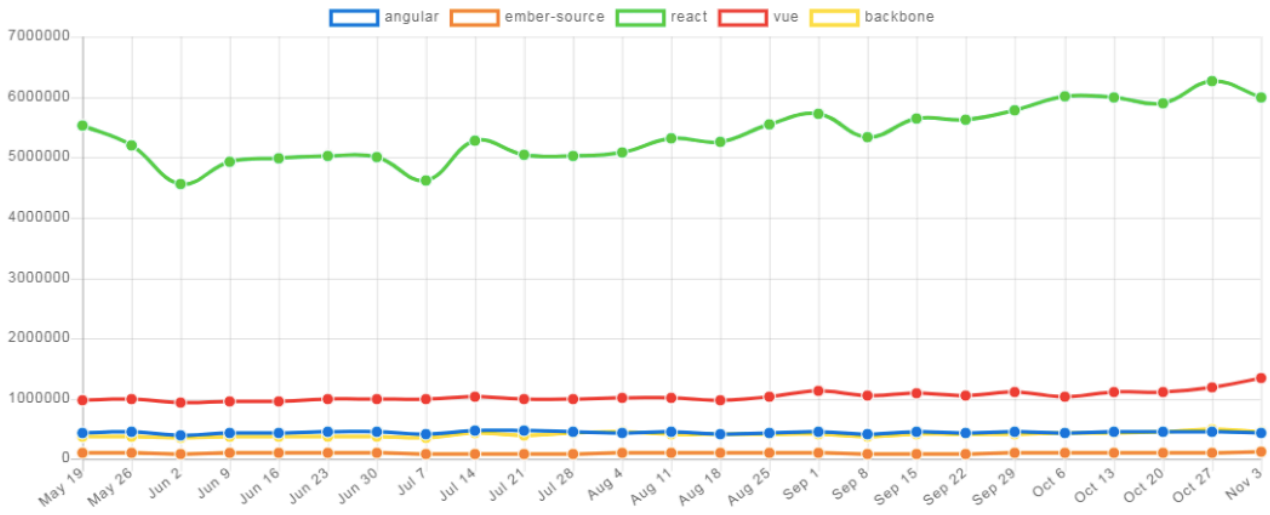

Рис. 2.2. Обсяги завантажень пакетів фреймворків/бібліотек JS.

Розберемо трохи детальніше ці фреймворки/бібліотеки [5] :

- **React**

На даний момент у сфері веб-розробки на React припадає більше 60% вибору програмістів. React ґрунтується на ідеях реактивного програмування, також привносячи велику кількість власних концептуальних рішень.

На React написані відомі додатки як Facebook та Instagram, де бібліотека вражає своєю ефективністю всередині високотрафікованих динамічних додатків.

При написанні SPA (Single Page Application) буде найефективнішим вибором.

- **Angular**

Раніше AngularJS була «золотою дочірнею» серед фреймворків JavaScript, оскільки вона була представлена корпорацією Google у 2012 році. Фундаментом Angular є концепція Model-View-Controller, що є одним з золотих стандартів концепцій розробки програмного забезпечення.

Angular заснований на TypeScript, тобто на типізованому JavaScript. Головна відмінність Angular від React є те, що перший – фреймворк, другий - біблотека. Головна різниця - інверсія контролю. Зрештою, це означає, що ми маємо каркас, в який ми додаємо функціональність. [8].

- **Vue [7]**

Vue.js - це прогресивний фреймворк JavaScript, яка використовується для побудови інтерфейсів користувача та SPA (односторінкові програми). Цей фреймворк відомий своєю швидкою кривою навчання. Це така проста у вивченні та доступна бібліотека, що, знаючи HTML, CSS та JavaScript, ми можемо почати створювати веб-програми у Vue.js. Крива швидкого навчання є своєрідною фішкою цього фреймворку. Це універсальний продукт для скромних потреб як бібліотека, або повноцінний фреймворк для створення величезних веб-програм.

- **Ember**

Ember володіє досить-таки хитромудрої архітектурою, яка дозволяє швидко створювати величезні клієнтські програми. У ньому реалізовані типові MVC-ідеї. Ember-додатки будуються з адаптерів, компонентів, контролерів, допоміжних об'єктів, моделей, маршрутів, сервісів, шаблонів, утиліт, доповнень.

Якщо ви розроблюєте великий багатотрафікований і функціональний додаток, то Ember – це ваш вибір.

- **Backbone**

Це полегшена бібліотека для структурування коду JavaScript. Він має структуру MVC / MV *, яка складається з трьох компонентів. Загальна картина в Backbone схожа з іншими фреймворками, він побудований навколо основного набору речей. У вас є події, моделі, колекції, подання та маршрутизатори

Якщо вам потрібно розробити додаток, з яким будуть працювати користувачі, що належать до різних груп, то для поділу моделей можна скористатися колекціями (масивами) Backbone. У моделях, колекціях, маршрутах і уявленнях Backbone можна користуватися подіями.

Проаналізувавши всі відомості і особливості кожного фреймворку наш вибір пав на React.

2.1.3 Чому саме React і які його переваги

React поставляється з JSX, необов'язковим розширенням синтаксису, яке дозволяє писати власні компоненти. Ці компоненти в основному приймають цитування у форматі HTML, а також роблять усі підкомпоненти чудово зрозумілими для розробників.

Хоча було багато суперечок щодо JSX, але це вже стосується написання користувацьких компонентів, створення великих обсягів програм та перетворення HTML-макетів у дерева ReactElement.

Найкращі якості бібліотеки React[9]:

- **Швидкість**. React в основному дозволяє розробникам використовувати окремі частини своїх додатків як на стороні клієнта, так і на стороні сервера, що в кінцевому рахунку підвищує швидкість процесу розробки.

- **Гнучкість**. Порівняно з іншими фронтенд-фреймворками, код React простіший в обслуговуванні та гнучкий завдяки своїй модульній структурі. Ця гнучкість, в свою чергу, економить величезну кількість часу та витрат для бізнесу.
- **Продуктивність**. React JS був розроблений для забезпечення високої продуктивності. Ядро фреймворку пропонує віртуальну програму DOM та рендеринг на стороні сервера, що робить складні програми надзвичайно швидкими.
- **Юзабіліті**. Розгортання React досить легко здійснити, якщо ви володієте якимись базовими знаннями JavaScript.

Всі ці особоливості React ідеально підходять для нашого проекту з технічної точки зору, з цим фреймворком розробка нашого веб-додатку буде найбільш ефективною та продуктивною. Саме тому наш вибір – React.

Але також React має свої недоліки, тому що React - це тільки представлення.

З цим фреймворком ми не отримаємо:

- Власну систему подій;
- Роботу з ajax-запитами;
- Будь-якої шар даних;
- Promises;
- Фреймворк на всі випадки життя;
- Будь-які думки про те, як реалізувати все вищезазначене.

У реальному світі React сам по собі не потрібен. Гірше того, це призводить до того, що кожен винаходить свій велосипед.

Тому з цим фреймворком нам необхідно використовувати інші бібліотеки, як наприклад бібліотеки для контролю стану нашого додатку (Redux, MobX).

На цьому етапі, проаналізувавши всі особливості розробки клієнтської частини веб-додатку ми прийшли до рішення включити до нашого проекту ще одну бібліотеку – Redux.

2.1.4 Чому саме Redux [10]

Redux - це спосіб управління «станом», або ми можемо сказати, що це кеш чи пам’ять, до якого всі компоненти можуть отримати доступ структурованим способом. Саме ця опція найкраще підходить у зв’язку з React, тому що там управління «станом» реалізоване на найкраим чином. До «стану» необхідно отриматувати доступ за допомогою "Редьюсера" та "Екшенів". Слід розуміти, інтерфейс Redux перехоплює контроль над усім додатком, а виступає у ролі мозку, який передає йому сигнали у вигляді повідомлень.

Які переваги дає такий підхід? [4]

- Великі програми мають великі «стани» додатків, і управління ними стає все більш незручним у міру зростання вашого додатка. Крім того, ви можете мати компоненти, які використовують ті самі дані, але розміщуються випадково в дереві DOM. Ось чому нам потрібні бібліотеки управління, такі як Redux.
- Принцип роботи Redux захоплюючий і в той же час такий простий. Його шаблон дуже інтуїтивно зрозумілий, а назви його функцій говорять самі за себе.
- Ще одна закономірність, якої дотримується Redux, називається «незмінність». І цей термін ви досить часто знайдете і в інших фреймворках та бібліотеках. Незмінність означає, що ми не змінюємо

об'єкт стану та його властивості безпосередньо. Натомість ми робимо новий об’єкт, перераховуємо новий стан програми та оновлюємо його за допомогою новоствореного об’єкта. Нам необхідно залишити свій старий об’єкт стану цілим.

Саме завдяки цим якостям бібліотеки Redux ми дійшли висновку, що це – ідеальний варіант для нашого майбутнього веб-додатку.

2.2 Архітектура бекенд частини веб-додатку

Бекенд (або серверна часина) - це програмне забезпечення, яке працює на сервері, з яким може взаємодіяти ваша програма та веб-сайт. Ми можемо використовувати цей сервер для багатьох речей: зберігання даних, обмін даними, керувати своїм додатком, вміст хосту для показу у нашому додатку, керувати своєю бізнес-логікою центрально, щоб не довелося повторно впроваджувати цю бізнес-логіку у свої програми.

2.2.1 Вибір мови програмування для бекенду

Для розробки бекенду кожен рік створюється щось нове та унікальне, тому необхідно зробити правильний та ефективний вибір:

- **JavaScript**

JavaScript - одна з найпопулярніших мов за останнє десятиліття. Вона дозволяє розробникам створювати інтерфейс і бекенд з однаковим синтаксисом, що значно зменшує навантаження.

Express.JS та Node.js дозволяють керувати та запускати обидва кінці програми, тоді як API допомагають у швидшій та легшій розробці додатків.

- **Go (Golang)**

Система одночасності Go, заснована на Goroutines, є міцною базою для розробки паралельних додатків. Тому він вже активно використовується для створення мережевих інструментів (зокрема, Kubernetes), але також отримує популярність як серверна мова для веб-додатків. У стандартній бібліотеці вже є все, що потрібно для реалізації Rest-Backend.

- **Python**

Python простий у використанні, потужний та універсальний. Його читабельність робить його чудовою першою мовою програмування - вона дозволяє мислити як програміст і не витрачати час на заплутаний синтаксис.

Завдяки інтерфейсу шлюзу веб-серверів (WSGI) було визначено стандартний API для розробки веб-додатків, і навколо нього розвинулось декілька фреймворків, таких як Django та Falcon.

- **Ruby**

Ця мова програмування була розроблена в 1990 році японським експертом з програмування. Чудовим у цій мові є те, що вона має синтаксис, подібний до Python та Java. Це забезпечує великі можливості автоматизації. Саме тому такі платформи, як Airbnb та Esty, використовують саме цю мову програмування.

- **C#**

C# - одна з найпопулярніших мов для створення серверної системи. Саме завдяки таким дивовижним особливостям, як автоматизація серверів Windows. Окрім цього, вона чудовий, оскільки виконує код дуже швидко. C# також може бути використаний для розробки ігор та створення додатків CLI.

2.2.2 Причини обрання JavaScript (NodeJS)?

Для бекенд частини веб-додатку HR-портал наш вибір пав на NodeJS.

Перш за все, використання Node.js як серверної технології надає значний приріст завдяки використанню однієї мови як на фронтенді, так і на

бекенді. Це означає, що розробка стає більш ефективною та багатофункціональною, що, в свою чергу, призводить до зниження витрат на розробку.

Окрім цього, варто зазначити, що JavaScript є найпопулярнішою мовою програмування, тому кодова база додатку буде простішою для розуміння для більшості інженерів.

Ми також зможемо повторно використовувати код і ділитися ним між інтерфейсом та серверними частинами програми, що пришвидшує процес розробки.

Крім того, спільнота Node.js постійно зростає - кількість запитань на StackOverflow постійно збільшується, тому база знань про технологію широко доступна.

Той факт, що весь стек технологій Node.js є відкритим та безкоштовним, також є чудовою новиною.

NodeJS пропонує чудові рішення для управління пакетами, npm або yarn, а кількість доступних інструментів з відкритим кодом в реєстрі npm величезна і швидко зростає.

Це лише деякі з багатьох переваг Node.js, які слід врахувати, вибираючи технологію для наступного проекту [14].

2.2.3 Що таке ExpressJS і чому ми його обрали [6]

ExpressJS - це вбудований фреймворк NodeJS, який може допомогти створювати веб-додатки на стороні сервера швидше та розумніше. Простота, мінімалізм, гнучкість, масштабованість - деякі з його характеристик, і оскільки він виготовлений у самому NodeJS, він успадкував і його продуктивність.

ExpressJS зробив для NodeJS те, що Bootstrap зробив для HTML / CSS та адаптивного веб-дизайну. Це зробило кодування в NodeJS чудовим і надало

програмістам деякі додаткові функції для розширення кодування на стороні сервера.

ExpressJS - це найвідоміший фреймворк NodeJS - настільки, що коли більшість людей говорять про NodeJS, вони, безумовно, мають на увазі NodeJS + ExpressJS.

Найважливіші функції ExpressJS, на які ми опиралися при виборі:

- **Швидша розробка на стороні сервера**. Express.js надає багато часто використовуваних функцій Node.js у формі функцій, які можна легко використовувати в будь-якому місці програми. Це позбавляє від необхідності кодувати протягом декількох годин і тим самим економить час.
- **Middleware**. Це частина програми, яка має доступ до бази даних, запиту клієнта та інших проміжних програм. Він головним чином відповідає за систематичну організацію різних функцій Express.js. Саме Middleware ми використовуємо у фронтент частині з допомогою Redux.
- **Роутинг**. ExpressJS забезпечує надзвичайно вдосконалений механізм маршрутизації, який допомагає зберегти стан веб-сторінки за допомогою їх URL-адрес.

2.2.4 Вибір бази даних для веб-додатку [16]

Те, як ви керуєте даними у своєму додатку, відіграє вирішальну роль у забезпеченні позитивного досвіду для користувачів. Зрештою, не має значення, наскільки добре розроблений інтерфейс вашої програми та наскільки чистий ваш код, якщо ваша програма не може швидко отримувати, обробляти та доставляти інформацію. Більше того, усі ці дані слід захищати,

щоб зловмисники не могли їх взяти. На щастя, цього можна досягти за допомогою грамотно обраної системи управління базами даних.

База даних - це місце, де ви зберігаєте та впорядковуєте всі дані, які збираєте через свій додаток, тоді як система управління базами даних (СУБД) - це програмне забезпечення для зручного управління цією базою даних.

На ринку існує більше 300 систем управління базами даних. Вибір між такою кількістю інструментів справді надзвичайний.

База даних SQL проти NoSQL:

Що стосується вибору бази даних, однією з найбільших проблем є вибір між структурою даних SQL (реляційна) та NoSQL (нереляційна). Хоча обидва мають хороші показники, є деякі ключові відмінності, про які повинні пам’ятати.

Бази даних SQL

Реляційна база даних - це сукупність таблиць, які мають заздалегідь визначені зв’язки між собою. Для підтримки та запиту реляційної бази даних система управління базами даних використовує структуровану мову запитів (SQL), загальну програму користувача, яка забезпечує простий інтерфейс програмування для взаємодії з базою даних.

Реляційні бази даних складаються з рядків, які називаються кортежами, і стовпців, званих атрибутами. Кортежі в таблиці мають однакові атрибути.

Бази даних NoSQL

Нереляційні або розподілені бази даних, слугують альтернативою реляційним базам даних. Вони можуть зберігати та обробляти неструктуровані дані (дані із соціальних мереж, фотографії, файли MP3

тощо), пропонуючи розробникам більшу гнучкість та більшу масштабованість.

Дані в нереляційних базах даних можна змінювати на льоту, не впливаючи на наявні дані. Крім того, бази даних NoSQL можна запускати на декількох серверах, тому їх масштабування дешевше і простіше, ніж масштабування баз даних SQL.

Найпопулярніші системи управління базами даних на сьогодні:

- OracleDB
- MySQL
- PostgreSQL
- MongoDB
- Redis
- Elascticsearch

Проаналізувавши список представлених систем управління базами даних, наш вибір пав на MongoDB, так як вона чудово підходить під наш додаток, тому що для нашого додатку важко визначити схему для БД, так як ми маємо динамічно-змінючийся додаток, який працює у реальному часі.

MongoDB має такі переваги:

- **Проста установка.** За допомогою хостингу A2 ви можете встановити MongoDB в один клік через Webuzo. Це програма встановлення безкоштовного програмного забезпечення, сумісна з багатьма шаблонами операційних систем (ОС).
- **Висока швидкість.** MongoDB - це орієнтована на документи база даних. Доступ до документів простий шляхом індексації. Отже, це забезпечує швидку відповідь на запит.

- **База даних без схем.** На відміну від баз даних, які покладаються на SQL, MongoDB - це об'єктно-орієнтована СУБД. Вона використовує документи, організовані у колекції, замість таблиць.

2.2.5 Вибір ORM для СУБД та бекенд частини

ORM або Об’єктно-реляційне відображення - це ідея написання запитів до бази даних, використовуючи об’єктно-орієнтовану парадигму улюбленої мови програмування. ORM допомагає взаємодіяти з базою даних, використовуючи вибрану нами мову замість SQL.

Існує багато різних бібліотек, які дозволяють здійснювати запити та маніпулювати даними з вашого додатка JavaScript, і кожна з них відрізняється своїм дизайном та рівнем абстракції:

- **Prisma.** Prisma відрізняється від більшості ORM тим, що моделі не визначаються у класах, а в схемі Prisma, основній конфігурації та файлі визначення моделі даних, що використовується інструментарієм Prisma. У схемі Prisma ви визначаєте своє джерело даних, як-от база даних PostgreSQL, та моделі, такі як користувачі та повідомлення та відносини між ними.
- **Sequelize.** Це ORM, який підтримує Postgres, MySQL, MariaDB, SQLite та Microsoft SQL Server. Він слідує традиційному шаблону ORM визначення моделей шляхом розширення класу Model.
- **Mongoose.** Це популярний та добре підтримуваний інструмент моделювання об'єктів Node.js для MongoDB. Mongoose - це маппер об'єктного документа, оскільки Mongoose - це база даних, що базується на документах. Це дозволяє моделювати дані за допомогою схем, і також вона включає вбудовану перевірку, побудову запитів та бізнес-логіку.

Аналізуючи наведену вище інформацію, наш вибір ORM пав на Mongoose, і ось ключові особливості нашого вибору:

- **Схеми**. MongoDB - це денормалізована база даних NoSQL. Це робить її по суті без схем, оскільки документи мають різні набори полів з різними типами даних. Mongoose визначає схему для моделей даних, щоб документи відповідали певній структурі із заздалегідь визначеними типами даних.
- **Перевірка.** Mongoose має вбудовану перевірку для визначень схем. Це позбавляє від написання купи перевірочного коду. Просто включивши у визначення схем такі речі, як required: true, Mongoose надає нестандартні перевірки для ваших колекцій (включаючи типи даних).
- **Повернення результатів.** Mongoose полегшує повернення оновлених документів або результатів запитів. Поки власний драйвер повертає об’єкт із позначкою успіху та кількістю модифікованих документів, Mongoose повертає сам оновлений об’єкт, щоб ви могли легко працювати з результатами.

2.3 Висновки до розділу

В даному розділі буд проведений огляд технологій, задля того, щоб обрати найкращі з них для розробки нашого веб-додатку. Ми проаналізували такі види фреймворків, як React, Angular, Vue, Ember та Backbone, і дійшли до висновку, що саме React, з допомогою Redux є найкращим вибором для нашого програмного комплексу. Також, були проаналізовані мови програмування для серверної частини веб-додатку, а саме: Go, Python, C# та NodeJS. В процесі огляду цих технологій ми дійшли висновку, що в поєднанні з React та Redux, найкращим вибором для серверної частини буде NodeJS, яку ми під’єднаємо до бази даних MongoDB.

РОЗДІЛ 3

РЕАЛІЗАЦІЯ ПРОГРАМНОГО КОМПЛЕКСУ

3.1 Структура фронтенд частини веб-додатка

Умовно процес створення веб-додатку можна розділити на 3 етапи:

- Планування

Даний етап можна розділити на кілька підетапів: створення ідеї, розробка структури проекту, опрацювання макета проекту

- Дизайн

Створивши дизайн-модель проекту можна переходити до створення візуального макету дизайну веб-сторінок.

- Розробка

Отже, процес дизайну макета сторінки плавно перетікає в процес «оживлення» зробленого на попередніх етапах. Перш ніж відразу починати писати HTML, CSS і JS варто трохи поговорити про структуру.

3.1.1 Структура папок і файлів

Структура проекту, тобто розміщення та ієрархія файлів і папок є дійсно важливою частиною дизайну програми. Додаток має бути структурованим з урахуванням наступного:

- **Компонованість**: Модулі/компоненти повинні бути добре компоновані та розділені. Знаходження файлів та папок повинно бути дуже швидким.
- **Послідовність**: Незалежно від того, за якою структурою каталогів ми вирішили слідувати, вона повинна бути узгодженою протягом усієї програми.

- **Папки за функцією**: Папки / каталоги слід називати та впорядковувати відповідно до об’єкта. Це дозволяє розробникам дуже легко знаходити компоненти, над якими їм потрібно працювати.

В нашому проекті структура папок і файлів виглядає наступним чином:

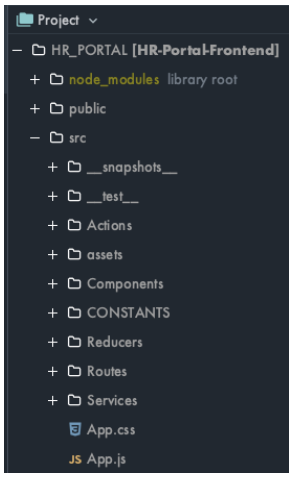

Рис. 3.1. Структура папок і файлів в нашому проекті.

Як ми бачимо, дерево папок і файлів нашого проекту дуже чітка та структурована, лише по назві папки можна зрозуміти, за що відповідають файли або папки у ній.

3.1.2 Структура веб-додатка

Перш ніж перейти до опису компонентів, не зайвим буде розібратися в термінології.

Сайтмеп - це структура сторінок веб-додатка представлена в ієрархічній моделі. Така схема допомагає оцінити обсяг сторінок, які формують веб-додаток, а також зрозуміти логіку їх взаємозв'язку. Іноді це примітивні по організації і структурі схеми. А часом дуже складні з багаторівневою вкладеністю карти. Сайтмепи допомагають планувати розподіл змісту і механіку навігації майбутнього веб-додатку.

Сайтмеп нашого веб-додатка має наступний вигляд [17]:

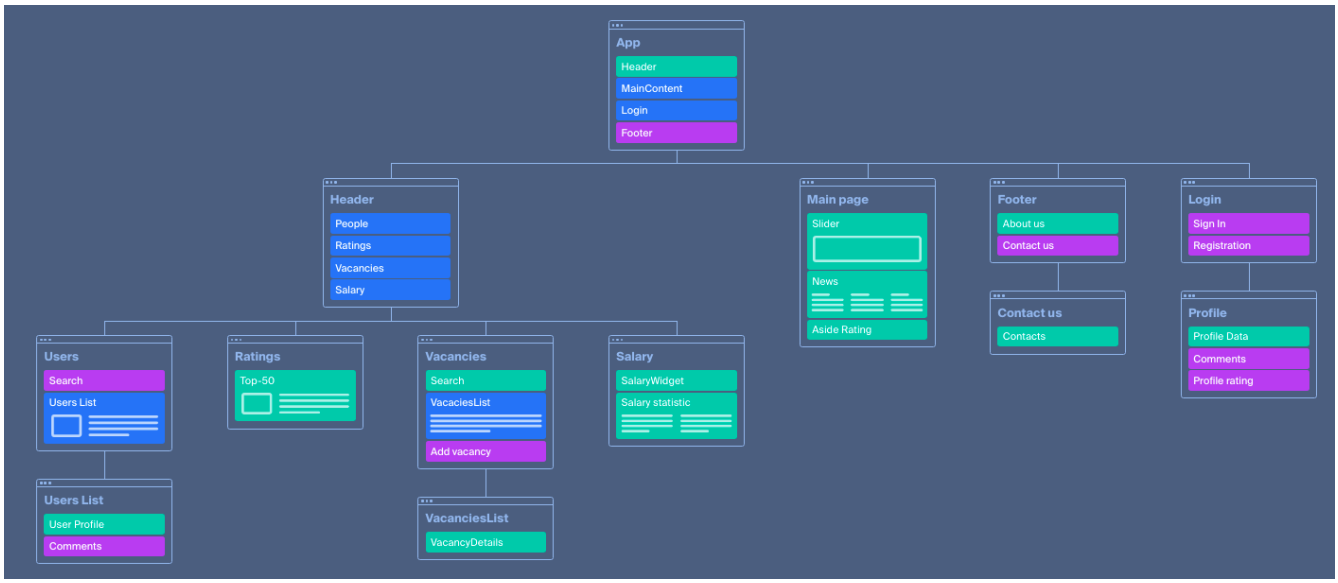

Рис. 3.2. Сайтмеп нашого веб-додатка.

На даній діаграмі зображені усі компоненти, які існують в нашому веб-додату, та зв’язки між ними.

Синій колір – компоненти, які є свого роду контейнерами, в них прописана логіка та отримуються дані, які потім передаються в нижчі по ієрархії компоненти.

Зелений колір – компоненти, які не містять в собі ніякої логіки та займаються лише відображенням контенту.

Фіолетовий колір – компоненти, які виконують якісь дії.

3.1.3 Опис структурних компонентів

Виходячи с сайтмепу нашого веб-додатку (див. **Рис. 3.2**) ми маємо наступні структурні компоненти нашого веб-додатку:

- **Компонент «App»**

Найголовніший компонент нашого веб-додатку. Він вбирає в собі усі компоненти веб-додатку та відповідає за відмальовку та рендер усіх існуючих компонентів програми.

- **Компонент «Login»**

Компонент, який відповідає за авторизацію та реєстрацію користувачів. Він містить в собі запит на сервер при реєстрації/авторизації, після якого він отримує та передає дані у внутрішній стан програми, які потім відображаються на сторінці профіля.

- **Компонент «Header»**

Компонент, який відмальовується на кожній сторінці веб-додатку. Так звана «шапка» сайту. Містить в собі навігаційну панель, за допомогою якої користувач може потрапити на бажану сторінку.

- **Компонент «Main Page»**

Головна сторінка веб-додатку, яку бачить користувач при першому візиті. Містить в собі такі компоненти: слайдер з новинами, список найголовніших новин тижня, та рейтинг HR-ів на боковій панелі. Кожен з цих компонентів веде на конкретну новину, або профіль конкретно обраного HR-а відповідно.

- **Компонент «Footer»**

Компонент, який відмальовується на кожній сторінці веб-додатку. Так званий «підвал» сайту. Містить в собі логотип, роки існування веб-додатку, посилання на соціальні мережі, інформацію про проект, та електронну адресу веб-додатку.

- **Компонент «Users»**

Компонент, який містить в собі список людей, які отримуються через запит на сервер. Також компонент містить в собі інструмент пошуку по заданим критеріям користувача.

- **Компонент «Profile»**

Компонент, який являє собою сторінку профіля, попередньо обраного користувачем/відвідувачем. Містить в собі персональну інформацію людини, також розділ з відгуками, де користувач може залишити свій особистий коментар, при умові, що він авторизований/зареєстрований.

- **Компонент «Ratings»**

Компонент, який містить в собі рейтинг «ТОП 50 HR України», який оновлюється кожен місяць, в якому користувач зможе переглянути інформацію про найкращих IT-Рекрутерів України.

- **Компонент «Vacancies»**

Компонент, який являє собою список вакансій у світі IT. Також компонент містить інструмент пошуку по вакансіям, з заданими конкретними критеріями. Посилання на вакансію веде на сторінку вакансії, де користувач може переглянути всю інформацію про вакансію, та надіслати своє резюме.

- **Компонент «Salary»**

Компонент, який містить в собі інформаційний зарплатний віджет, який відображаю заробітні плати у IT-світі по обраній посаді за квартал.

3.2 Структура серверної частини

Будь-який динамічний веб-додаток базується на фреймворку - програмному забезпеченні веб-додатків, яке контролює побудову веб-сторінок та полегшує обслуговування. Спосіб відображення таких веб-програм на екрані користувача не визначається заздалегідь, а динамічно формується логікою програми, яка реалізована на стороні сервера чи клієнта програми.

Динамічний веб-додаток генерує сторінки/дані в режимі реального часу, відштовхуючись від запиту, відповідна відповідь ініціюватиметься з сервера і досягне клієнта. Залежно від відповіді, код на стороні клієнта буде змінюватися, як передбачається.

Динамічні веб-програми відрізняються між собою тим, як вони працюють, і випадки їх використання визначають їх підхід до розробки та архітектуру.

На зображеній нижче схемі зображено просту архітектуру динамічного сайту [18].

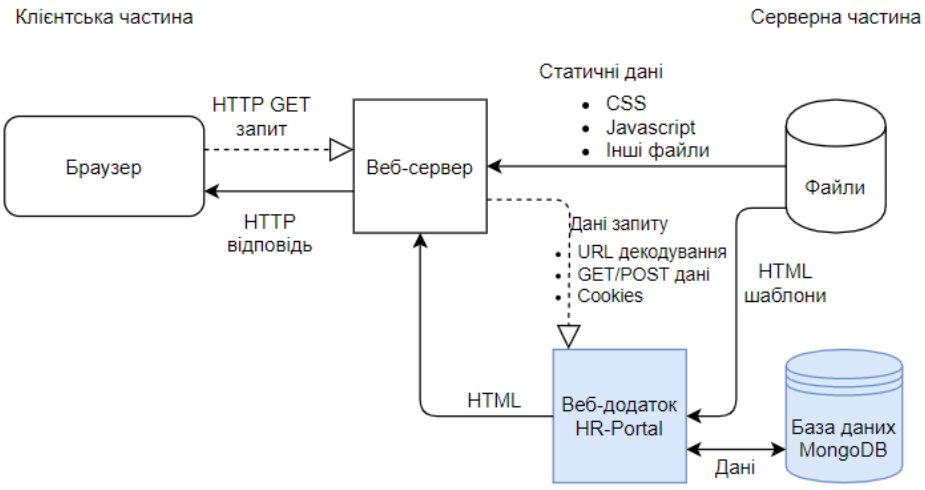

Рис. 3.3. Схема архітектури динамічного сайту.

Програмування серверної частини дозволяє зберігати інформацію в базі даних і динамічно створювати, повертати HTML і інші типи файлів.

Рис. 3.4. Збережені дані користувачів у базі даних у вигляді документів.

3.2.1 Структура бекенду веб-додатка HR-портал

На наступному зображені ми можемо бачити потокову структуру даних при запитах/відповідях на або з серверу [19].

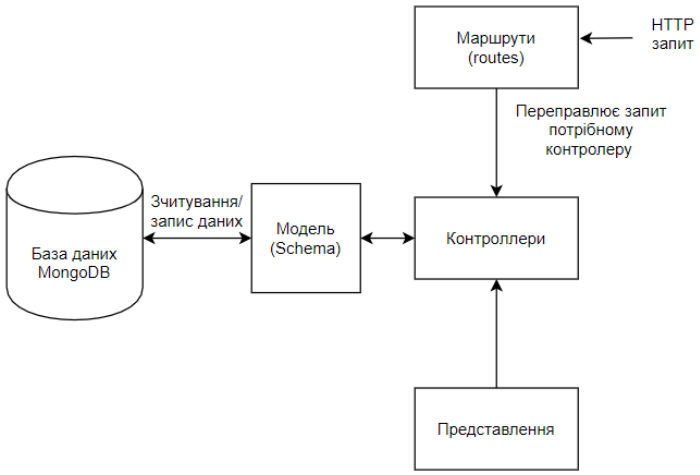

Рис. 3.5. Діаграма основного потоку даних у серверній частині додатку.

Бекенд веб-додатку «HR-портал» складається з наступних компонентів:

- **ORM моделей**

Моделі виступають в ролі функціональної обгортки, яка виступає об’єктом для кожної окремої частини даних, які надходять с клієнтської частини. Проходячи валідацію, влаштовану у Mongoose, формується кінцевий об’єкт типу «ключ» - «значення», після чого він відправляється в базу даних.

- **Роутів**

Роути, тобто маршрути, є одним з найважливіших частин функціоналу бекенду. За допомогою адреси роуту, сервер розуміє які дані до нього надходять, що треба модифікувати, або які дані необхідно повернути. Роутів може існувати необмежена кількість, залежно від розміру програмного комплексу, що розроблюється. Кожна кінцева адреса роуту може мати декілька реалізованих запитів (get, post, update, тощо).

Моделі, створені для додатку:

- **User**

Головна модель, яка формується при першому створенні акаунту, та не може змінюватися в майбутньому. Має такі обов'язкові поля як ім'я (name), прізвище (lastName), ел. адреса (email), та ключове поле – userId, яку слугує унікальним ідентифікатором користувача та є необхідним у процесі логіну та наступних дій на веб-додатку.

- **Profile**

Модель слугує доповненням до моделі User, яка включає в себе всю додаткову інформацію про користувача, таку як дата народження, досвід роботи, місце роботи, телефон, соц. мережі, тощо. Ця модель при створенні підв'язується до моделі User по ключовому полю userId і є її невід'ємною частиною.

- **Vacancy**

Містить поля з інформацією для складання вакансії у відповідному розділі: heading (заголовок, назва вакансії), company (компанія), description (опис вакансії), salary (заробітня плата), category (категорія), employment (зайнятість), date (дата).

- **News**

Містить поля для складання новини: imgURL (посилання на зображення новини), header (заголовок новини), shortDescription (короткий опис новини), text (повний текст новини), date (дата).

Крім того, для всіх полів створені вимоги валідації – тип даних, обов'язковість заповнення, максимальна і мінімальна кількість символів.

- **Feedbacks**

Модель, яка створюється про відправленні рейтингового відгуку на HR/Рекрутера. Вона є більш складною, так як повинна мати в собі підв'язку

то того, хто відправив відгук, та до того, кому цей відгук був відправлений. Зв’язок реалізований за допомогою ключового поля кожного користувача – userId.

Роути, які функціонують у додатку [3]:

- **Authentication.** Роут, що використовується при реєстрації та створенні нового користувача. Також має метод повернення даних про усіх створних користувачів.
- **Verify.** Функціональний роут, що спрацьовує тільки при логіні користувача до веб-додатку, який перевірає валідність userId та токену, що виступає у ролі ключа авторизації.
- **Profile.** Роут, який містить в собі усі функціонально важливі методи дій користувача у веб-додатку, а саме: оновлення інформації, видалення акаунту, та повернення інформації про акаунт.
- **Resumes.** Роут, що реалізовує метод відправки файлу резюме, збереженні його на сервері або подальшому поверненні його на клієнтську частину.
- **Vacancy.** Роут, що містить у собі методи вакансій, а саме: створення, оновлення, отримання конкретної вакансії по її ідентифікатору, отримання списку усіх вакансій.
- **News.** Роут, що містить у собі методі новин, а саме: створення новин, оновлення, та отримання унікальної, або списку усіх існуючих новин.

3.3 Опис алгоритму веб-додатку

Алгоритм - це процедура або формула для вирішення проблеми, заснована на проведенні послідовності зазначених дій. Комп’ютерну програму можна розглядати як складний алгоритм. В математиці та

інформатиці під алгоритмом зазвичай розуміють невелику процедуру, яка вирішує періодичну задачу [20].

3.3.1 Алгоритм роботи фронтенд частини

В нашому веб-додатку алгоритм виглядає наступним чином:

1. Перший візит користувача на сайт:
   1) Відображається головна сторінка сайту
   2) Йде запит на сервер, для отримання даних для рейтингу «ТОП HR» та секції «Головні новини»
   3) Отримання даних з сервера та відображення їх на головній сторінці.
   4) Зупинка програми в очікуванні наступних дій користувача.
2. Натискання кнопки «Вхід»:
   1) Відображення сторінки логіну, з можливістю зареєструватися, або увійти, якщо користувач вже зареєстрований.
      1.1) Реєстрація користувача:
      1.1.1) Користувач вводить вводить дані для реєстрації (Ім’я, прізвище, електронну пошту та пароль) та натискає «Створити акаунт»
      1.1.2) Йде пост-запит на сервер, дані користувачі відправляються на сервер та записуються у таблицю в базі даних.
      1.1.3) Приходить відповідь сервера, і у випадки коректних даних реєстрація проходить успішно, користувач отримує відповідне повідомлення та переноситься на сторінку «Входу»
      1.2) Вхід користувача:
      1.2.1) Користувач вводить свої особисті дані для входу
      1.2.2) Йде запит на сервер
      - якщо дані коректні, користувач переходить на головну сторінку, і його статус змінюється на «авторизований»

- якщо дані некоректні, виводиться відповідне повідомлення.

2) Зупинка програми в очікуванні наступних дій користувача.

3. Натискання секції «Рейтинг»:
   1) Перехід на сторінку «Рейтинг»
   2) Запит на сервер, отримання даних рейтингу
   3) Відображення даних сторінці
   4) Натискання на блок рейтингу:
      4.1) Перехід на профіль користувача, на якого було натиснуто.
      4.2) Йде запит на сервер, отримання даних профілю користувача
      4.3) Відображення даних профілю на сторінці
   5) Зупинка програми для очікування наступних дій
4. Натискання секції «Вакансії»:
   1) Перехід на сторінку «Вакансії»
   2) Запит на сервер, отримання даних вакансій
   3) Відображення даних сторінці у вигляді списку вакансій
   4) Натискання на вакансію:
      4.4) Перехід на сторінку вакансії, на яку було натиснуто
      4.5) Йде запит на сервер, отримання даних вакансії
      4.6) Відображення даних вакансії на сторінці
5. Натискання секції «Статистика ЗП»:
   1) Перехід на сторінку «Статистика ЗП»
   2) Запит на сервер, отримання даних заробітних плат
   3) Відображення даних сторінці у вигляді віджету

3.3.2 Алгоритм роботи серверної частини

В нашому веб-додатку алгоритм виглядає наступним чином:

1. Перший візит користувача на сайт:
   1) Відображається головна сторінка сайту
   2) Відправляється GET-запит на сервер за адресою api/user/profile, для отримання даних для рейтингу «ТОП HR» та секції «Головні новини»
   3) Отримання даних з сервера та відображення їх на головній сторінці.
   4) Зупинка програми в очікуванні наступних дій користувача.
2. Натискання кнопки «Вхід»:
   1) Відображення сторінки логіну, з можливістю зареєструватися, або увійти, якщо користувач вже зареєстрований.

   1.1) Реєстрація користувача:

   1.1.1) Користувач вводить вводить дані для реєстрації (Ім’я, прізвище, електронну пошту та пароль) та натискає «Створити акаунт»

   1.1.2) Йде POST-запит на сервер за адресою api/user/register, дані користувачі відправляються на сервер та записуються у колекцію в базі даних.

   1.1.3) Приходить відповідь сервера, і у випадку коректних даних реєстрація проходить успішно, користувач отримує відповідне повідомлення та переноситься на сторінку «Входу»

   1.2) Вхід користувача:

   1.2.1) Користувач вводить свої особисті дані для входу

   1.2.2) Надсилається POST-запит на сервер за адресою api/user/login

   - якщо дані коректні, користувач переходить на головну сторінку, і його статус змінюється на «авторизований»

- якщо дані некоректні, виводиться відповідне повідомлення.

2) Зупинка програми в очікуванні наступних дій користувача.

3. Натискання секції «Рейтинг»:
   1) Перехід на сторінку «Рейтинг»
   2) Надсилання GET-запит на сервер, отримання даних рейтингу
   3) Відображення даних сторінці
   4) Натискання на блок рейтингу:

   4.1) Перехід на профіль користувача.

   4.2) Надсилання GET-запиту на сервер за адресою api/user/profile, отримання даних профілю користувача

   4.3) Відображення даних профілю на сторінці
   5) Зупинка програми для очікування наступних дій
4. Натискання секції «Вакансії»:
   1) Перехід на сторінку «Вакансії»
   2) Надсилання GET-запиту на сервер за адресою api/vacancy, отримання даних вакансій
   3) Відображення даних сторінці у вигляді списку вакансій
   4) Натискання на вакансію:

   4.4) Перехід на сторінку вакансії, на яку було натиснуто

   4.5) Надсилання GET-запиту на сервер за посиланням api/user/vacancy, отримання даних вакансії

   4.6) Відображення даних вакансії на сторінці
5. Натискання секції «Статистика ЗП»:
   1) Перехід на сторінку «Статистика ЗП»
   2) Надсилання GET-запиту на сервер, отримання даних заробітних плат
   3) Відображення даних сторінці.

3.4 Висновки до розділу

В даному розділі ми описали процес розробки програмного комплексу, продемонструвавши і описавши функціональність та компоненти фронтенд та серверної частини нашого веб-додатку.

Також, ми створили та описали структуру кожної із частин веб-додатку, та продемонстрували алгоритм роботи кожної із частин програмного комплексу.

ВИСНОВКИ

На основі проведених досліджень, завдяки великій кількості інформації стосовно теми створення веб-додатків та веб-сайтів нам вдалося створити унікальний в своєму роді портал, який представляє собою спільноту IT-рекрутерів України, з можливістю звичайних користувачів, претендентів на бажану роботу, директорів компаній, переглядати рейтинги HR та залишати свої відгуки та побажання стосовно роботи та професійного рівня конкретного рекрутера на вільній основі.

Нам вдалося вирішити проблеми незнання претендента, що його чекає на співбесіді, до чого бути готовим, та які навики спілкування з людьми йому треба підтягнути.

На створення такого великого веб-додатку нам знадобилося велика кількість часу та зусиль, ми вивчили та застосовували передові технології у сфері веб-розробки, такі як: технології фронтенд частини React та Redux; технології серверної частини NodeJS та MongoDB. Ми покращили свої знання у розробці веб-додатків, та здобули для себе багато досвіду, вирішуючи велику кількість проблем, які з’являлися по мірі розробки.

Проблеми, з якими зіштовхувались на різних етапах були абсолютно різні, а саме: зв’язок з сервером, отримування та зберігання даних, авторизація/реєстрація користувача, динамічні запити та динамічна зміна контенту веб-додатку.

Найбільша проблема, яка більше за все гальмувала процес дослідження та розробки веб-додатку – це авторизація/реєстрація користувача. Ця проблема вимагала дуже широких знань у сфері розробки веб-додатків, а також тонкого розуміння зв’язку клієнтської частини з серверною.

Проблеми, які нам не вдалося дослідити і запровадити в наш проект також існують – це адмін-панель, в якій динамічно можна додавати новини та

вакансії. На даному етапі це дуже складна та функціональна проблема, яка потребує великої кількості часу та зусиль, для її вирішення.

Також, в нас є плани, як в майбутньому цей проект можна модернізувати та покращити, а також вивести його на комерційну основу.

А саме:

- Додати адмін-панель в якій динамічно можна додавати і редагувати новини та вакансії, та загалом корегувати весь контент веб-додатку.
- Запровадити користувацькі ролі, а саме: претендент (людина, що перебуває в пошуку роботи), HR/Рекрутер та Компанія. Всі ці ролі будуть відрізнятися один від одного за послугами та можливостями, які вони отримають.
- Ввести режим унікальних можливостей за платну підписку, а саме: необмежена кількість переглядів особистих контактів IT-рекрутерів, можливість переглянути кількість людей/компаній, які переглядали твій профіль, профіль користувача з платною підпискою буде з’влятися у топі списка пошуку.

Отже, HR-Portal є свого роду платформою, де IT-рекрутери України мають унікальну можливість стати більш знайомими простим користувачам, та людям, які шукають роботу, та зможуть заробляти собі авторитет у IT-світі роботодавців України, а також заробляти авторитет, привертати увагу та інтерес до компанії, в якій вони працюють.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Портал програмістів Хабр [Електронний ресурс] / Режим доступу: https://habr.com/ru/ (СДВИНУТЬ КАК АБЗАЦ)

2. Портал порівняння пакетів npm [Електронний ресурс] / Режим доступу: https://www.npmtrends.com/

3. Сайт питань та відповідей програмістів Stack Overflow [Електронний ресурс] / Режим доступу: https://ru.stackoverflow.com

4. Алекс Бэнкс React и Redux. Функциональная веб-разработка / Алекс Бэнкс, Ева Порселло; [пер з англ.]. – :Питер, 2018. – 360с.

5. IT-блог Medium [Електронний ресурс] / Режим доступу: https://medium.com

6. Сайт фреймворка Express [Електронний ресурс] / Режим доступу: https://expressjs.com/.

7. Спільнота програмістів MDN [Електронний ресурс] / Режим доступу: https://developer.mozilla.org/.

8. Брэд Дейли, Брендан Дейли, Калеб Дейли. Разработка веб-приложений с помощью Node.js, MongoDB и Angular: исчерпывающее руководство по использованию стека MEAN: навч. пос. Диалектика-Вильямс. –: Росія, 2020. 656 с.

9. Портал курсів програмування Hexlet [Електронний ресурс] / Режим доступу: https://ru.hexlet.io/

10. Офіційний сайт Redux [Електронний ресурс] / Режим доступу: https://redux.js.org/

11. Jennifer Robbins, Learning Web Design: A Beginner's Guide to HTML, CSS, JavaScript, and Web Graphics. –: Канада, 2012. – 621с.

12. Соціальна мережа LinkedIn [Електронний ресурс] / Режим доступу: linkedin.com/legal/user-agreement

13. Ethan Brown, Web Development with Node and Express: Leveraging the JavaScript Stack. –: США, 2014. – 332с.

14. Офіційний сайт NodeJS [Електронний ресурс] / Режим доступу: https://nodejs.org/uk/

15. Портал курсів програмування Data Flair [Електронний ресурс] / Режим доступу: https://data-flair.training/

16. IT-блог Xplenty [Електронний ресурс] / Режим доступу: https://www.xplenty.com/blog/

17. Сайт створення сайтмеп структури [Електронний ресурс] / Режим доступу: https://octopus.do/

18. Сайт створення UML-діаграм [Електронний ресурс] / Режим доступу: https://app.diagrams.net/

19. Jason Krol, Web Development with MongoDB and NodeJS. –: Британія, 2015. – 300с.

20. Thomas H. Cormen, Introduction to Algorithms. –: США, 2009. – 1292с.

ДОДАТКИ

Додаток А

Фронтент частина веб-додатку

Як ми можемо бачити з сайтмепу нашого веб-додатка (див. Рис. 3.2) ми маємо головний компонент App, який відповідає за рендер сторінки, та містить в собі всі інші компоненти. Він має такий вигляд:

```
render() {
    const {alert} = this.props;

    // Скривання хедеру та футеру на сторінках реєстрації та логіну
    const shouldShowHeaderAndFooter = history.location.pathname !== ROUTES.LOGIN
        && history.location.pathname !== ROUTES.REGISTRATION && history.location.pathname !== ROUTES.PROFILE_REGISTER;

    return (
        <div className="app">
            <Router history={history}>
                {shouldShowHeaderAndFooter && <Header/>}
                {
                    alert.message &&
                    <div style={{marginBottom: 0}} className={`alert ${alert.type}`}>
                        {alert.message}
                    </div>
                }
                { /*Роутинг сайту (сторінки і навігація по ним)*/}
                <Switch>
                    <Route exact path={ROUTES.MAIN} component={MainPage}/>
                    <Route path={ROUTES.SEARCH} component={SearchPage}/>
                    <Route path={ROUTES.RATINGS} component={RatingPage}/>
                    <Route path={ROUTES.VACANCIES} component={VacanciesPage}/>
                    <Route path={`${ROUTES.VACANCY_DETAILS}/:vacancyID`} component={VacancyDetailsContainer}/>
                    <Route path={`${ROUTES.NEWS}/:post`} component={NewsPageContainer}/>
                    <Route path={ROUTES.STATS} component={StatsPage}/>
                    <Route path={ROUTES.ABOUT} component={AboutPage}/>
                    <Route path={ROUTES.LOGIN} component={SignInPage}/>
                    <Route path={`${ROUTES.PROFILE}/:userId?`} component={ProfilePageContainer}/>
                    <Route path={`${ROUTES.MYRESUMES}/:userId?`} component={ResumesPageContainer} />
                    <Route path={ROUTES.CREATEVACANCY} component={CreateVacancy}/>
                    <Route path={ROUTES.REGISTRATION} component={RegistrationPage}/>
                    <Route path={ROUTES.PROFILE_REGISTER} component={ProfileHelper}/>
```

Рис. А.1. Компонент App.

Цей компонент відповідає за роутинг нашого веб-додатку (переходи с одної сторінку на іншу) та за відображення компонентів, які є всередині нього.

React працює так, що кореневий компонент нашого додатку поєднується з єдиним блочним тегом HTML, в якому і відображається наш

веб-додаток. Цей кореневий компонент за прийнятою нормою називається index.js. В нашому проекті він виглядає наступним чином:

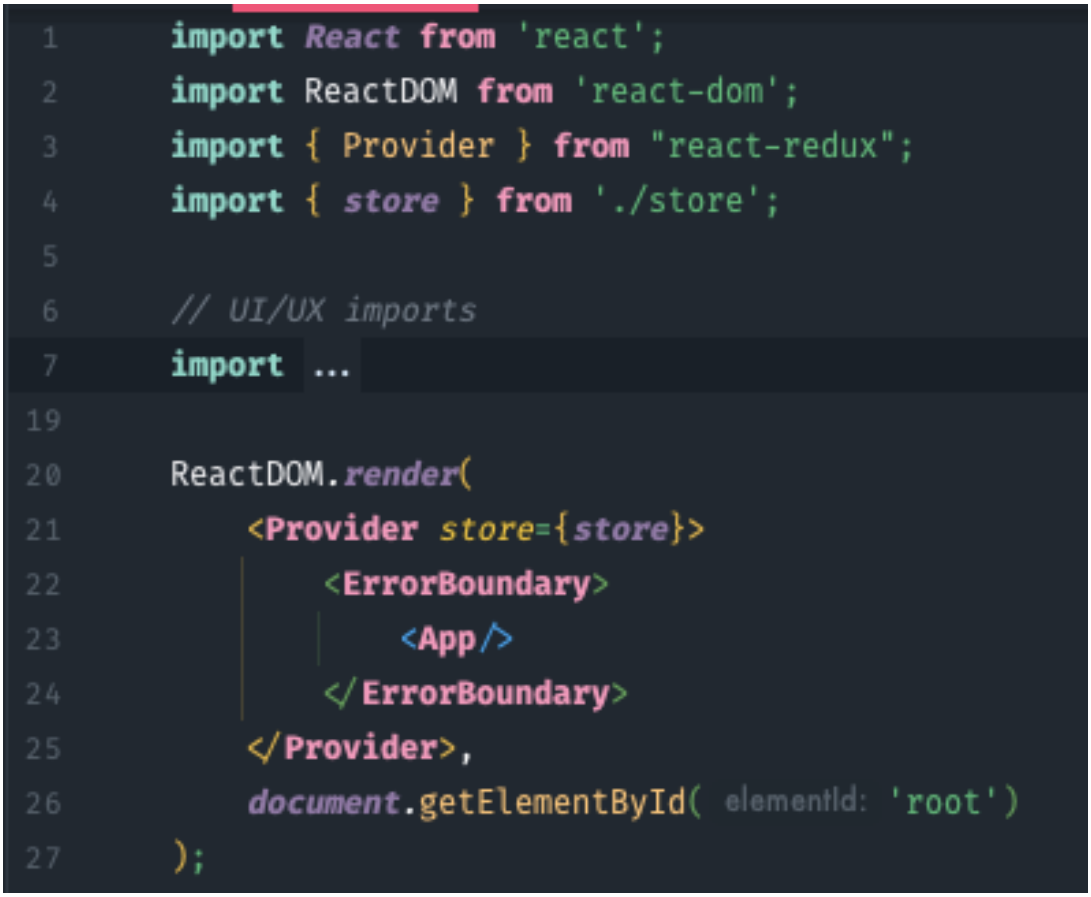

Рис. А.2. Кореневий компонент index.js.

Як ми бачимо, ми наш компонент App знаходиться у функції render, яка в свою чергу займається поєднанням React та HTML, та відображанням нашого веб-додатку.

Тепер ми підходимо до одного з головних компонентів нашого веб-додатку – компонент store (далі – стор). Стор зберігає весь стан нашого додатку, та всі дії, які відбуваються на нашому веб-додатку, проходять через стор.

Сам компонент стор має наступний вигляд :

```
export const store = createStore(
    rootReducer,
    composeEnhancers(
    applyMiddleware(
        thunkMiddleware,
        loggerMiddleware
    ))
);
```

Рис. А.3. Компонент store.

Додаток Б

Серверна частина веб-додатку

Програмна реалізація головного компоненту index.js:

```
const express = require('express');
const multer = require("multer");
const app = express();
const dotenv = require('dotenv');
const mongoose = require('mongoose');


const cors = require('cors');
const PORT = process.env.PORT || 3000;


app.use(cors());
app.options('*', cors());
app.use(express.static(__dirname));

const storageConfig = multer.diskStorage( opts: {
    destination: (req, file, cb) =>{
        cb(null, "resumes");
    },
    filename: (req, file, cb) =>{
        cb(null, file.originalname);
    }
});

app.use(multer( options: {storage: storageConfig}).single( name: "filedata"));

//Import Routes
const authRoute = require('./routes/auth');
const postRoute = require('./routes/post');
const profRoute = require('./routes/profile');
const vacanRoute = require('./routes/vacancy');
const newsRoute = require('./routes/news');
const feedbackRoute = require('./routes/feedbacks');
const resumeRoute = require('./routes/resumes');


dotenv.config();
```

Рис. Б.1. Головний компонент index.js.

Програмна реалізація запиту реєстрації нового користувача:

```
15  router.post('/register', async (req, res) => {
16      //lets validate a data before we make a user
17      const {error} = registerValidation(req.body);
18
19      if (error) return res.status(400).send(error.details[0].message);
20
21      //checking if the user is already in database
22      const emailExist = await User.findOne({email: req.body.email});
23      if (emailExist) return res.status(400).send( data: 'Такой адрес эл. почты уже зарегестрирован');
24
25      //Hash passwords
26      const salt = await bcrypt.genSalt(10);
27      const hashedPassword = await bcrypt.hash(req.body.password, salt);
28
29      //create a new user
30      const user = new User({
31          name: req.body.name,
32          lastName: req.body.lastName,
33          email: req.body.email,
34          password: hashedPassword,
35          role: req.body.role
36      });
37      try {
38          const savedUser = await user.save();
39          res.send(user._id);
40      } catch (err) {
41          res.status(400).send(err);
42      }
43  });
44
```

Рис. Б.2. Реалізація запиту реєстрації нового користувача.

Об’єктна модель користувача:

```
const mongoose = require('mongoose');

const userSchema = new mongoose.Schema({
    name: {
        type: String,
        required: true,
        min: 2,
        max: 255
    },
    lastName: {
        type: String,
        required: true,
        max: 255
    },
    email: {
        type: String,
        required: true,
        max: 255,
        min: 6
    },
    password: {
        type: String,
        required: true,
        max: 1024,
        min: 6
    },
    role: {
        type: String,
        required: true,
        max: 124,
        min: 2
    },
    date: {
        type: Date,
        default: Date.now
    },

});
```

Рис. Б.3. Об’єктна модель користувача.