

Пояснювальна записка до дипломного проєкту

на тему: Вебзастосунок для автоматичного підбору вакансій на основі
резюме та адаптації резюме за допомогою нейромереж для ІТ-галузі
(комплексна тема). Вебзастосунок та агрегація вакансій

КПІ.ПІ-1122.045440.02.81

Київ – 2025

ЗМІСТ

ВСТУП	4
1 РОЗРОБЛЕННЯ ВИМОГ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ.....	5
1.1 Розроблення функціональних вимог	5
1.2 Постановка завдання на розробку програмного забезпечення	9
Висновки до розділу	10
2 КОНСТРУЮВАННЯ ТА РОЗРОБЛЕННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ	11
2.1 Архітектура програмного забезпечення.....	11
2.2 Архітектурні рішення та обґрунтування вибору засобів розробки.....	12
2.3 Конструювання програмного забезпечення.....	14
2.3.1 Розробка методу збору вакансій	19
Висновки до розділу	19
3 АНАЛІЗ ЯКОСТІ ТА ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ	21
3.1 Аналіз якості ПЗ.....	21
3.2 Опис процесів тестування.....	22
Висновки до розділу	29
ВИСНОВКИ.....	30
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	31
ДОДАТОК А ЗВІТ ПОДІБНОСТІ.....	32

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

IDE	– Integrated Development Environment – інтегроване середовище розробки.
API	– Application programming interface, прикладний програмний Інтерфейс.
IT	– Інформаційні технології.
БД	– База даних.

ВСТУП

В умовах сучасного світу пошук роботи на ІТ ринку України є дуже трудомістким і може займати місяці, а деколи і роки. В рамках такого стану дуже важливим є пошук якнайбільш релевантних вакансій з усіх можливих сайтів роботи, та постійний їх моніторинг на предмет появи нових вакансій.

Дипломний проєкт присвячено розробці вебзастосунку для пошуку вакансій пошукачами роботи та отримання рекомендацій до адаптації резюме. Метою дипломного проєкту є створення вебзастосунку, котрий допомагає пошукачам роботи підбирати найбільш релевантні вакансії в ІТ-галузі та адаптувати зміст резюме користувача від вимоги обраних вакансій, що підвищує точність процесу працевлаштування для обох сторін на ринку праці.

Для досягнення поставленої мети дипломного проєкту в рамках індивідуальної частини роботи буде розроблено вебзастосунок для пошуку роботи та адаптації резюме, розроблено методи збору вакансій з різних сайтів пошуку роботи, а також методи автоматичного надсилання повідомлень про нові вакансії користувачам.

1 РОЗРОБЛЕННЯ ВИМОГ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

1.1 Розроблення функціональних вимог

В таблиці 1.1 наведено загальну модель вимог до програмного забезпечення, а в таблиці 1.2 наведений опис функціональних вимог до програмного забезпечення. Матрицю трасування вимог можна побачити в таблиці 1.3.

Таблиця 1.1 – Загальна модель вимог

№	Назва	ID Вимоги	Пріоритети	Ризики
1	Система авторизації користувача.	FR-1	Високий	Високий
1.1	Реєстрація користувача.	FR-2	Високий	Високий
1.2	Авторизація користувача.	FR-3	Високий	Високий
1.3	Відновлення паролю	FR-4	Середній	Середній
1.4	Вихід із акаунту.	FR-5	Середній	Низький
2	Завантаження резюме	FR-6	Високий	Високий
3	Пошук вакансій	FR-7	Високий	Середній
3.1	Відображення характеристик вакансій	FR-8	Середній	Низький
3.2	Фільтрація вакансій	FR-9	Низький	Низький
4	Адаптація резюме	FR-10	Високий	Середній
4.1	Зчитування тексту вакансії за посиланням	FR-11	Середній	Низький
4.2	Відображення оцінки релевантності та рекомендацій	FR-12	Високий	Середній
5	Надсилання нотифікацій про нові вакансії	FR-13	Середній	Середній

Таблиця 1.2 – Перелік функціональних вимог

Назва	Опис
FR-1	<p>Система авторизації користувача.</p> <p>При переході неавторизованим користувачем на вебсайт він переадресовується на сторінку входу в профіль. З неї він може перейти на сторінку реєстрації або відновлення паролю, а після входу користувач може вийти з профілю.</p>
FR-2	<p>Реєстрація користувача.</p> <p>Для реєстрації користувач має ввести імейл, ім'я та пароль та натиснути кнопку реєстрації. Якщо імейл не є валідним чи вже наявний в системі або одне з полів є порожнім, відображається відповідне повідомлення і кнопка не є активною. Після натиснення кнопки користувачу надсилається імейл про підтвердження імейл адреси, та на екрані відображається відповідне повідомлення. Після підтвердження імейлу користувач може увійти в профіль.</p>
FR-3	<p>Авторизація користувача.</p> <p>Після реєстрації і підтвердження імейлу користувач може увійти в профіль. Для цього він має ввести імейл та пароль та натиснути кнопку входу. Якщо імейл не є валідним або пароль є порожнім чи введено неправильно користувач отримує відповідне повідомлення і кнопка входу стає неактивною. Після успішного входу користувач переадресується на головну сторінку.</p>
FR-4	<p>Відновлення паролю.</p> <p>З сторінки входу в профіль можна перемкнутись на форму відновлення паролю. Там користувач має ввести свою імейл адресу, і якщо вона наявна в системі кнопка відновлення стає активною, інакше користувач отримує відповідне повідомлення про помилку. Після натискання кнопки користувачу надсилається імейл з посиланням, перейшовши по якому він може встановити новий пароль.</p>

Продовження таблиці 1.2

FR-5	<p>Вихід з акаунту.</p> <p>На усіх сторінках крім сторінки авторизації присутня кнопка виходу з профілю, після натиснення якої юзер виходить з акаунту та переадресується на сторінку авторизації.</p>
FR-6	<p>Завантаження резюме.</p> <p>На головній сторінці застосунку присутня кнопка завантаження резюме, після натиснення якої відкривається вікно вибору файлу, в якому можна вибрати pdf, docx або doc файл. Після вибору файлу і його збереження, користувачу відображається відповідне повідомлення.</p>
FR-7	<p>Пошук вакансій.</p> <p>На головній сторінці застосунку присутня кнопка пошуку вакансій, яка є активною якщо користувач має завантажене резюме в застосунку. Після натиснення кнопки користувач переадресується на сторінку результатів пошуку.</p>
FR-8	<p>Відображення характеристик вакансій.</p> <p>На сторінці результатів пошуку вакансії відображаються у вигляді списку, в якому присутня інформація про назву вакансії, оцінку релевантності до резюме користувача, її локацію та зарплатню якщо такі дані доступні. По кліку на назву вакансії користувач переадресовується на сайт вакансії.</p>
FR-9	<p>Фільтрація вакансій.</p> <p>Після отримання усіх результатів пошуку вакансій користувач може відфільтрувати їх за локацією, або наявністю інформації про зарплатню.</p>

Продовження таблиці 2.2

FR-10	<p>Адаптація резюме.</p> <p>На головній сторінці застосунку присутня кнопка пошуку вакансій, яка є активною якщо користувач має завантажене резюме в застосунку. Після натиснення кнопки відкривається вікно, в якому користувачу пропонується ввести посилання на вакансію або її текст. Після введення тексту вакансії кнопка отримання рекомендацій до резюме стає активною, і після її натиснення відображається вікно з результатом.</p>
FR-11	<p>Зчитування тексту вакансії за посиланням.</p> <p>У вікні завантаження вакансії для адаптації резюме присутнє поле вводу посилання на вакансії та кнопка “Зчитати”. Після натиснення кнопки текст вакансії зчитується з сайту та заповнюється в поле вводу тексту вакансії, після чого його можна відредагувати. Якщо текст не вдалось зчитати, відображається відповідне повідомлення, і користувачу пропонується ввести текст вручну або перевірити посилання.</p>
FR-12	<p>Відображення оцінки релевантності та рекомендацій.</p> <p>Після натиснення кнопки отримання рекомендацій до резюме відкривається вікно, в якому показується оцінка релевантності резюме користувача до введеної вакансії, та список ключових слів які варто додати в резюме.</p>
FR-13	<p>Надсилення нотифікацій про нові вакансії.</p> <p>Під час завантаження нових вакансій в базу даних застосунку відбувається перевірка їх релевантності до резюме користувача, і при співпадинні надсилається імейл з новими вакансіями.</p>

Таблиця 1.3 – Матриця трасування функціональних вимог

	FR- 1	FR- 2	FR- 3	FR- 4	FR- 5	FR- 6	FR- 7	FR- 8	FR- 9	FR- 10	FR- 11	FR- 12	FR- 13
UC- 01	+	+											
UC- 02	+		+										
UC- 03	+			+									
UC- 04						+							
UC- 05							+	+	+				
UC- 06										+	+	+	
UC- 07													+

1.2 Постановка завдання на розробку програмного забезпечення

Метою дипломного проєкту є створення вебзастосунку, котрий допомагає пошукачам роботи підбирати найбільш релевантні вакансії в ІТ-галузі та адаптувати зміст резюме користувача від вимоги обраних вакансій, що підвищує точність процесу працевлаштування для обох сторін на ринку праці.

Для досягнення поставленої мети необхідно розв'язати наступні задачі:

- проектування та розробка вебзастосунку;
- авторизація користувача;

- робота з резюме користувача (зчитування, збереження, підтримка можливості адаптації під вакансії);
- агрегація IT-вакансій зі сторонніх сайтів і їх збереження для подальшого аналізу;
- автоматизоване надсилання повідомлень користувачу з релевантними вакансіями.

Висновки до розділу

У першому розділі пояснювальної записки сформульовано функціональні вимоги індивідуальної частини дипломного проєкту та сформовано матрицю трасування функціональних вимог для демонстрації відповідності вимог до варіантів використання проєкту.

В результаті сформульовано постановку завдання на розробку індивідуальної частини програмного забезпечення.

2 КОНСТРУЮВАННЯ ТА РОЗРОБЛЕННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

2.1 Архітектура програмного забезпечення

Для клієнтської сторони вебзастосунку було обрано використання Single-Page Application архітектури.

Серверна частина застосунку використовує трьохрівневу архітектуру, тобто складається з:

- API що надає інтерфейс для роботи з застосунком;
- Business logic layer що містить логіку застосунку;
- Data access layer для роботи з базою даних.

Серверна частина застосунку містить наступні компоненти:

- компонент користувачів: відповідає за збереження профілю користувача та його резюме та верифікацію юзера за допомогою запитів до Firebase [1]. При завантаженні резюме він також нотифікує API сервісу машинного навчання для препроцесінгу резюме;
- компонент нотифікацій: відповідає за відправлення імейлів користувачам, використовуючи запити до Mailjet [2];
- компонент скрапінгу: відповідає за збір описів вакансій за посиланням;
- компонент вакансій: відповідає за обробку запитів про отримання вакансій, їх описів та адаптацію резюме, використовуючи запити до API сервісу машинного навчання, дані з бази даних та компонент скрапінгу;
- компонент агрегації вакансій: відповідає за збір даних про вакансії та підтримання їх в актуальному стані за допомогою запитів до сайтів вакансій. Після отримання вакансій цей компонент передає нові вакансії API сервісу машинного навчання для препроцесінгу та отримання оцінок релевантності для користувачів, які направляє компоненту нотифікацій для направлення користувачам.

Діаграму компонентів в нотації C4 наведено на рисунку 2.1:

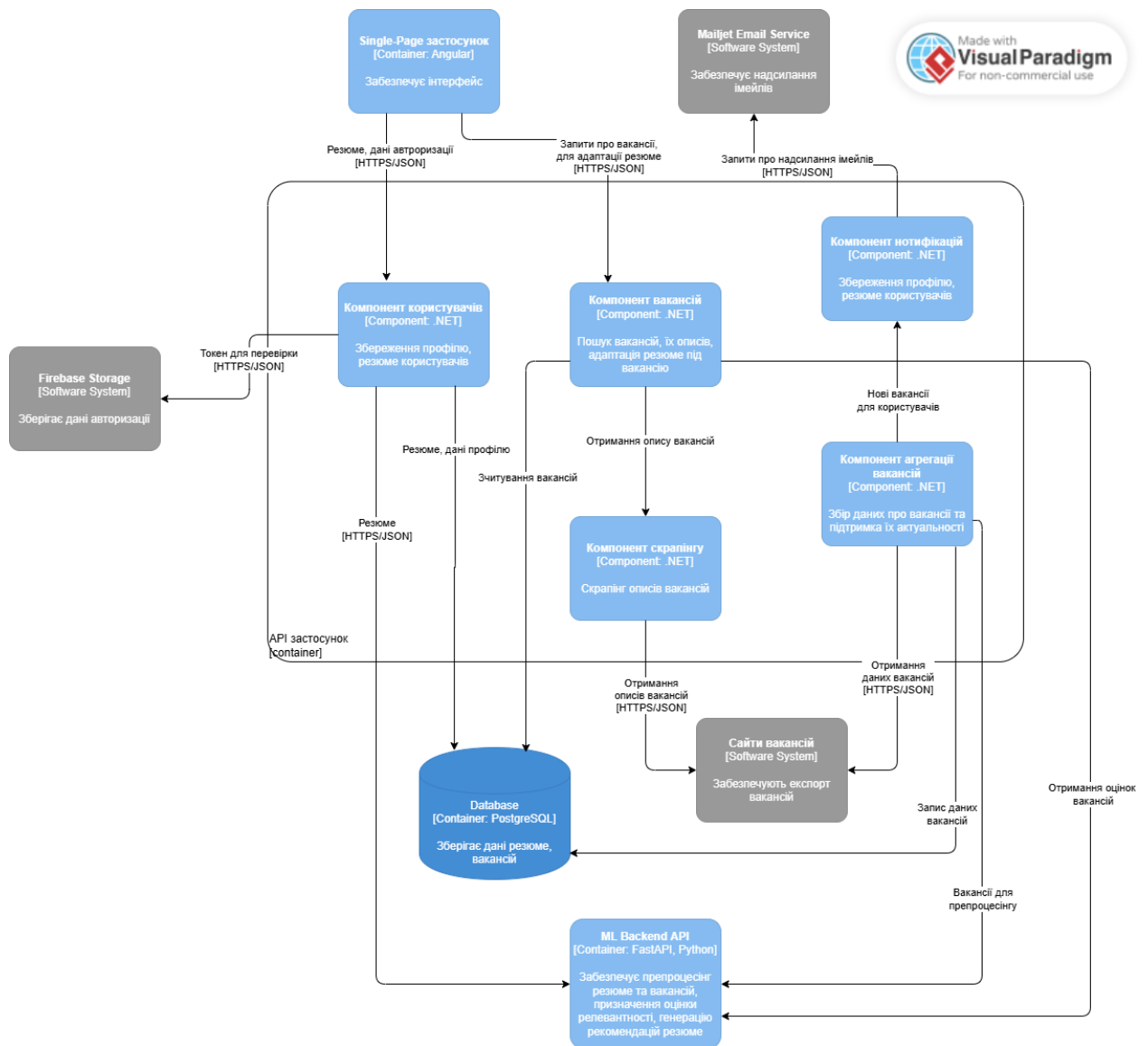


Рисунок 2.1 - Діаграма третього рівня C4 для API сервісу серверної частини вебзастосунку

Діаграму основних класів серверної частини застосунку наведено у графічному матеріалі, кресленні 2.

2.2 Архітектурні рішення та обґрунтування вибору засобів розробки

Для створення клієнтської сторони вебзастосунку було обрано використати фреймворк Angular [3], з використанням NgRx Store [4] для роботи з даними. NgRx Store ізолює логіку роботи з даними і виклики API сервісів від логіки представлення даних, що значно спрощує модифікацію коду та покращує його якість за рахунок чіткого розділення обов'язків.

Діаграма компонентів NgRx Store що використовуються для менеджменту стану клієнтської сторони застосунку наведена на рисунку 2.2:

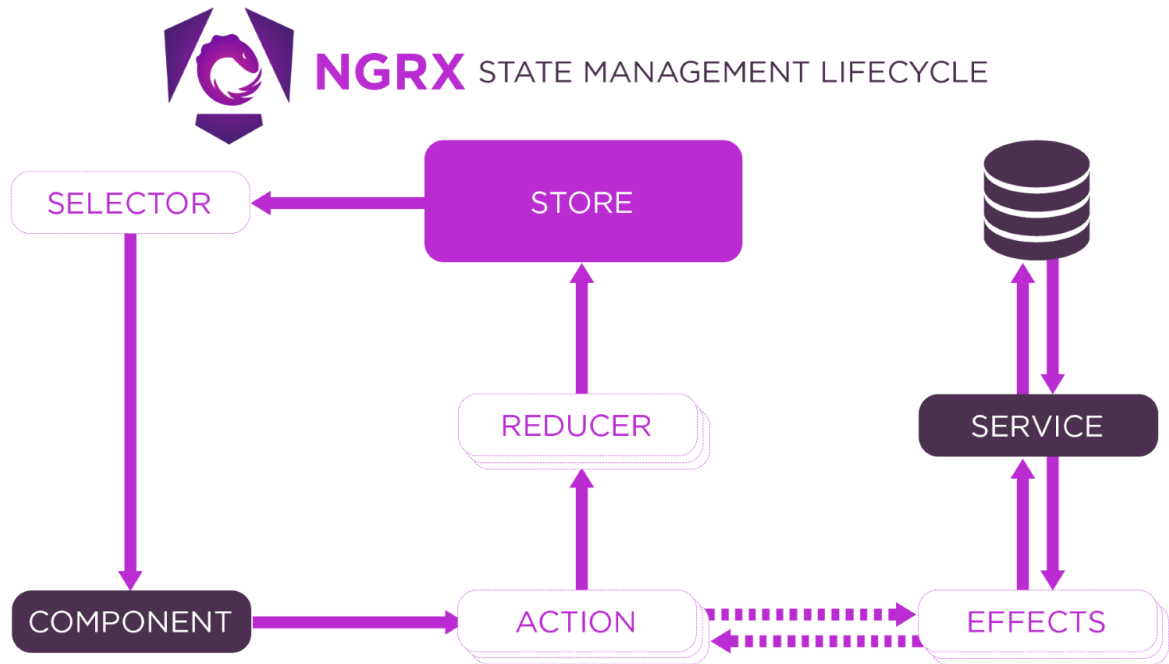


Рисунок 2.2 - Діаграма компонентів NgRx Store [4]

Для API сервісу застосунку було обрано використати веб-фреймворк ASP.NET [5] з використанням мови C# завдяки його продуктивності та простоті роботи з базою даних за допомогою Entity Framework [6].

Для авторизації користувачів було обрано використати сервіс Firebase завдяки безпеці даних користувачів, що він пропонує, його зручній інтеграції та гарній ціновій політиці, що дозволяє обмежене використання в рамках безкоштовної підписки. При реєстрації та авторизації користувача пароль та логін передаються Firebase за допомогою HTTPS запиту, після чого він надає токен користувача та його унікальний ідентифікатор, що використовуються для автентифікації запитів на API застосунку. Це забезпечує безпеку даних користувача, оскільки його пароль не зберігається ніде в самому застосунку, а повністю керується Firebase, який також надає функціонал підтвердження імейлу та відновлення паролю для додаткової безпеки.

Для надсилання імейлів користувачу було розглянуто використання одного з 2 сервісів: Mailjet та SendGrid [7]. Переваги SendGrid включають дуже

поширене використання та гарну підтримку в рамках інфраструктури .NET, але в безкоштовній його підписці дозволено лише 100 листів на день, впродовж 60 днів. Натомість безкоштовна підписка Mailjet дозволяє відправку до 200 листів на день без обмеження в часі підписки, і також має .NET бібліотеку для роботи з ним, хоч і менш популярну. Завдяки перевагам безкоштовної підписки для надсилання імейл повідомлень було обрано використання сервісу Mailjet.

В якості інструменту для розробки клієнтської частини застосунку було розглянуто Visual Studio Code та WebStorm. Перевагою WebStorm є те, що він є повноцінним IDE з нативною підтримкою Angular, що забезпечує зручні інструменти створення компонентів та рефакторингу. Значним недоліком Webstorm є швидкість роботи і кількість ресурсів що він витрачає. Натомість, Visual Studio Code є дуже легковаговим редактором коду з дуже великою можливістю конфігурації за допомогою плагінів, завдяки чому було обрано саме його використання.

Для розробки серверної частини було розглянуто використання таких IDE як Rider та Visual Studio. Основною і практично єдиною суттєвою перевагою Visual Studio є наявність безкоштовної версії. Rider же у порівнянні з Visual Studio має більш зрозумілий інтерфейс, використовує менше пам'яті та має більш зручні та швидкі інструменти рефакторингу, завдяки чому було обрано саме його використання.

2.3 Конструювання програмного забезпечення

Опис основних методів серверної частини застосунку наведено в таблицях 3.1-3.13:

Таблиця 2.1 – Опис методів класу UserService

Назва методу	Призначення методу
GetCurrent	Отримання поточного користувача за токеном
Create	Створення користувача
CheckExisting	Перевірка існування користувача у базі даних з вказаним імейлом
AddResume	Створення резюме для користувача та надсилання його айді на апі машинного навчання для попередньої обробки
GetResumeId	Отримання айді резюме користувача
AddClaims	Додавання даних про айді користувача до токenu авторизації

Таблиця 2.2 – Опис методів класу VacancyService

Назва методу	Призначення методу
Get	Отримання айді релевантних вакансій з апі машинного навчання та повернення повних моделей вакансій
AdaptResume	Отримання тексту вакансії за айді або як параметр в залежності від перегрузки, передача його та резюме користувача до апі машинного навчання та повернення результату
ParseVacancy	Отримання тексту вакансії за посиланням на неї, при наявності dou.ua в посиланні текст дістається з бази даних, при наявності postjobfree.com в посиланні використовується скрапер

Таблиця 2.3 – Опис методів класу PostJobFreeVacancyScrapper

Назва методу	Призначення методу
Scrape	Статичний метод для скрапінгу тексту вакансії з сайту postjobfree.com за посиланням

Таблиця 2.4 – Опис методів класу MachineLearningApiService

Назва методу	Призначення методу
NotifyResumeCreated	Надсилання повідомлення на апі машинного навчання про необхідність попередньої обробки резюме
NotifyVacanciesUpdated	Надсилання нових вакансій на апі машинного навчання з метою попередньої обробки та отримання їх оцінок релевантності для користувачів
GetVacancyScores	Надсилання айді резюме користувача на апі машинного навчання з метою отримання релевантних для нього вакансій
GetResumeAdaptation	Надсилання айді резюме та тексту вакансії з метою отримання рекомендацій до адаптації

Таблиця 2.5 – Опис методів класу TextExtractorService

Назва методу	Призначення методу
ReadPdf	Отримання тексту .pdf файлу
ReadDocx	Отримання тексту .docx файлу
ReadDoc	Отримання тексту .doc файлу

Таблиця 2.6 – Опис методів класу AggregatorJob

Назва методу	Призначення методу
Execute	Виклик агрегаторів вакансій з метою їх збору та збереження в базу даних, надсилання отриманих вакансій на апі машинного навчання, надсилання імейлів користувачам про нові релевантні для них вакансії

Таблиця 2.7 – Опис методів класу DouVacancyAggregatorService

Назва методу	Призначення методу
Aggregate	Ітерація по посиланням на експорт компаній dou.ua з метою збору з них вакансій, збереження їх у базі даних
GetVacancies	Отримання та мапінг вакансій за посиланням на експорт

Таблиця 2.8 – Опис методів класу PostJobVacancyAggregatorService

Назва методу	Призначення методу
Aggregate	Отримання вакансій з postjobfree.com по всім необхідним категоріям за допомогою скрапера, збереження їх в базі даних

Таблиця 2.9 – Опис методів класу PostJobVacanciesScraper

Назва методу	Призначення методу
Scrape	Скрапінг вакансій з сторінки пошуку вакансій сайту postjobfree.com за текстом пошуку

Таблиця 2.10 – Опис методів класу EmailService

Назва методу	Призначення методу
SendAsync	Надсилення імейл повідомлень, використовуючи API Mailjet

Таблиця 2.11 – Опис методів класу ScraperJob

Назва методу	Призначення методу
StartAsync	Запуск збору посилань на компанії Dou при першому запуску серверу

Таблиця 2.12 – Опис методів класу CompanyService

Назва методу	Призначення методу
Create	Створення записів у таблиці компаній в базі даних
HasCompanies	Перевірка наявності компаній в базі даних

Таблиця 2.13 – Опис методів класу DouCompanyScraper

Назва методу	Призначення методу
Scrape	Скрапінг посилань на топ1500 компаній dou.ua зі сторінки компаній

Опис утиліт, бібліотек та іншого стороннього програмного забезпечення, що використовується у розробці наведено в таблиці 3.14.

Таблиця 2.14 - Опис утиліт та програмних бібліотек

№ п/п	Назва утиліти	Опис застосування
1	Jetbrains Rider	Головне середовище розробки серверної частини дипломного проекту
2	Visual Studio Code	Головне середовище розробки клієнтської частини дипломного проекту
3	Entity Framework	ORM фреймворк для роботи з базою даних
4	ASP.NET Core	Фреймворк для побудови веб-апі застосунку
5	Selenium	Бібліотека для скрапінгу даних
6	PdfPig	Бібліотека для зчитування резюме у форматі pdf
7	NPOI	Бібліотека для зчитування резюме у форматі doc
8	OpenXML	Бібліотека для зчитування резюме у форматі docx
9	Quartz	Бібліотека для автоматичного запуску коду агрегації вакансій з заданою періодичністю
10	AutoMapper	Бібліотека для мапінгу моделей застосунку

2.3.1 Розробка методу збору вакансій

В рамках розробки програмного забезпечення дипломного проєкту постала задача зчитування вакансій з сайту dou.ua. Основними проблемами, які мав вирішувати метод, котрий розробляється, була відсутність єдиного API сайту для експорту вакансій та необхідність підтримки актуального списку вакансій у програмному забезпеченні.

Для збору вакансій було вирішено використовувати експорт вакансій компаній ДООУ, доступний за посиланням формату <https://jobs.dou.ua/companies/{companyName}/vacancies/export/>. За посиланням такого типу від ДООУ можна отримати JSON-файл, що містить всю інформацію про вакансії певної компанії. Таким чином задача зводиться до збору назв компаній, присутніх на ДООУ, та збір вакансій для кожної з них.

Для отримання списку назв компаній при першому запуску програмного забезпечення запускається вебскрапер, що переходить за посиланням <https://jobs.dou.ua/companies/> та ініціює пагінацію, поки на сторінці не буде зображено не менше ніж 1500 компаній. Після цього він зчитує посилання на кожну з них у форматі <https://jobs.dou.ua/companies/{companyName}/> та записує до бази даних програмного забезпечення.

Після цього та кожного дня протягом роботи програмного забезпечення запускається агрегатор вакансій, який для кожного отриманого посилання на компанію додає /vacancies/export та таким чином виконує експорт вакансій для кожної компанії, порівнює з наявними у базі даних та проводить синхронізацію. Таким чином в базі даних буде підтримуватись актуальний список більшості вакансій ДООУ з актуальністю на момент відпрацювання агрегатора раз на день.

Висновки до розділу

В рамках другого розділу пояснювальної записки було описано архітектуру індивідуальної частини дипломного проєкту та архітектурні рішення, пов'язані з нею. Після цього було наведено опис основних методів

серверної частини застосунку, а також утиліти та програмні бібліотеки, використані в розробці програмного забезпечення і описано алгоритм збору вакансій.

3 АНАЛІЗ ЯКОСТІ ТА ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Аналіз якості ПЗ

Для статичного аналізу коду серверної частини вебзастосунку було обрано використання вбудований в JetBrains Rider статичний аналізатор Qodana [8]. Він дозволяє легко проаналізувати кодову базу на платформі .NET та виявити потенційні проблеми. Слід зауважити, що статичні аналізатори не можуть проаналізувати код, що використовує рефлексію, тому часто серед валідних зауважень надає помилки про не використання типів чи їх членів, хоча ті і використовуються неявно.

Звіт Qodana про кодову базу наведено на рисунку 3.1:

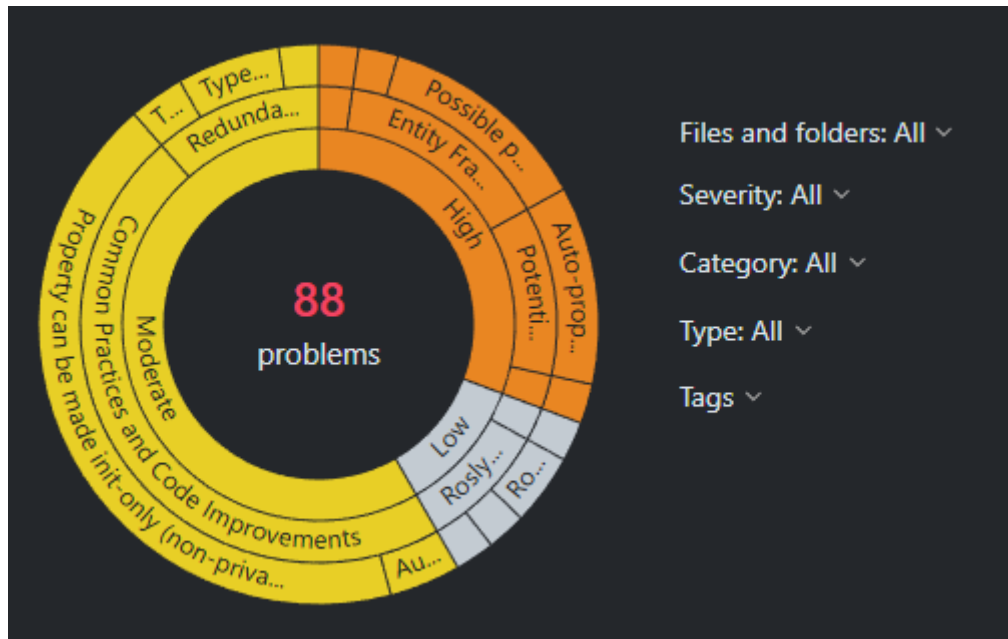


Рисунок 3.1 – Звіт Qodana

Звіт показує наявність 88 проблем із кодом, серед яких 56 середньої суворості і 26 – високої, що вже є непоганим показником. Проте як було сказано вище, серед цих проблем присутні багато таких, що пов'язані з відсутністю явного використання елементів коду, і не є валідними. Результат перевірки після видалення таких помилок зі звіту наведено на рисунку 3.2:

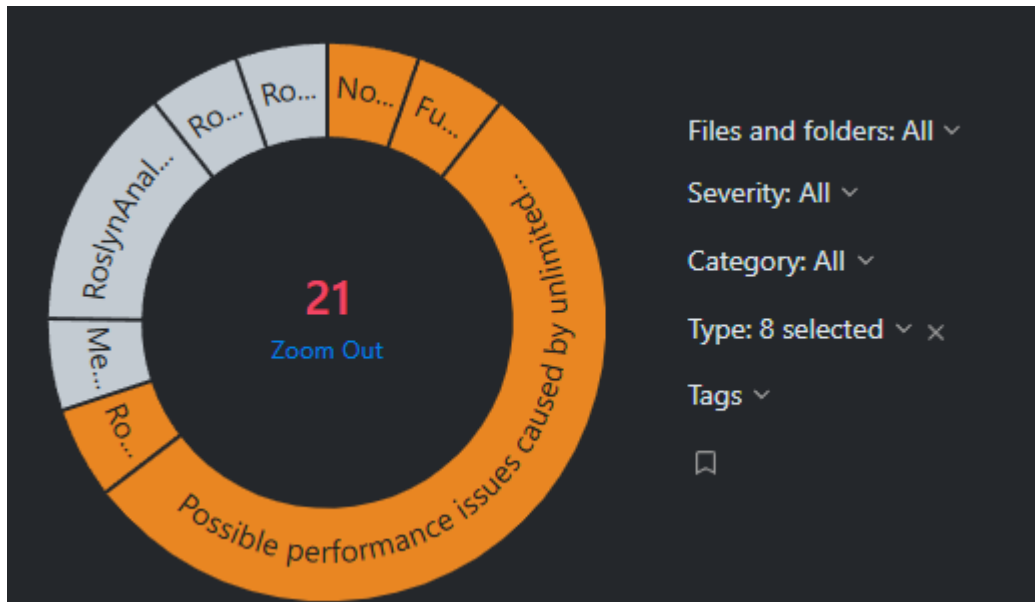


Рисунок 3.2 – Звіт Qodana після фільтрації типів проблем

Як бачимо, усі помилки середньої суворості були пов'язані з вказаною вище проблемою статичних аналізаторів, і після фільтрації типів помилок залишилось лише 15 проблем високої суворості, з яких 12 – потенційні проблеми з продуктивністю застосунку через необмежені за довжиною строки в базі даних, що безумовно є потенційною проблемою на яку варто звернути увагу, але в рамках предметної області роботи з текстами вакансій та резюме не обмежені за довжиною строки є необхідністю.

Загалом за звітом статичного аналізатора можна зробити висновок про якісне написання коду, з невеликою кількістю проблем.

3.2 Опис процесів тестування

Тестування виконується згідно документу «Програма та методика тестування».

Основним методом тестування програмного забезпечення було мануальне тестування. Опис проведених тестів наведено у таблицях 3.1-3.15

Таблиця 3.1 – Тест 1.1 Реєстрація користувача

Початковий стан системи	Користувач знаходиться на сторінці реєстрації
Вхідні дані	Електронна пошта, ім'я, пароль
Опис проведення тесту	У відповідні поля вводяться: коректна електронна пошта, яка не наявна у базі даних, ім'я користувача та пароль, користувач натискає кнопку реєстрації
Очікуваний результат	Реєстрація проходить успішно, користувач додається у систему, відображається повідомлення про надісланий
Фактичний результат	Збігається з очікуванням.

Таблиця 3.2 – Тест 1.2 Спроба входу в профіль до верифікації імейлу

Початковий стан системи	Користувач знаходиться на сторінці реєстрації, користувач зареєструвався але не підтвердив імейл
Вхідні дані	Електронна пошта, пароль
Опис проведення тесту	У відповідні поля вводяться: коректна електронна пошта, коректний пароль, користувач натискає кнопку входу
Очікуваний результат	Відображається повідомлення про те що користувач має підтвердити імейл перед входом
Фактичний результат	Збігається з очікуванням.

Таблиця 3.3 – Тест 1.3 Вхід в профіль

Початковий стан системи	Користувач знаходиться на сторінці реєстрації, користувач зареєструвався та підтвердив імейл
Вхідні дані	Електронна пошта, пароль
Опис проведення тесту	У відповідні поля вводяться: коректна електронна пошта, коректний пароль, користувач натискає кнопку входу
Очікуваний результат	Користувач переадресовується на головну сторінку застосунку і відображається повідомлення про успішну авторизацію
Фактичний результат	Збігається з очікуванням.

Таблиця 3.4 – Тест 1.4 Відновлення паролю

Початковий стан системи	Користувач знаходиться на сторінці реєстрації, користувач хоче відновити пароль
Вхідні дані	Електронна пошта
Опис проведення тесту	У відповідне поле вводиться коректна електронна пошта, що присутня у системі, користувач натискає кнопку відновлення паролю
Очікуваний результат	Відображається повідомлення про надісланий імейл відновлення паролю, користувачу надсилається імейл з посиланням, після переходу по якому він вводить новий пароль, за допомогою якого може увійти в профіль
Фактичний результат	Збігається з очікуванням.

Таблиця 3.5 – Тест 1.5 Завантаження резюме у форматі PDF

Початковий стан системи	Користувач знаходиться на головній сторінці, користувач не має завантаженого резюме
Вхідні дані	Резюме у форматі PDF
Опис проведення тесту	Користувач натискає кнопку завантаження резюме, обирає резюме у форматі PDF
Очікуваний результат	Відображається повідомлення про успішне завантаження резюме, кнопки пошуку вакансій та адаптації стають активними
Фактичний результат	Збігається з очікуванням.

Таблиця 3.6 – Тест 1.6 Завантаження резюме у форматі .doc

Початковий стан системи	Користувач знаходиться на головній сторінці, користувач не має завантаженого резюме
Вхідні дані	Резюме у форматі .doc
Опис проведення тесту	Користувач натискає кнопку завантаження резюме, обирає резюме у форматі .doc
Очікуваний результат	Відображається повідомлення про успішне завантаження резюме, кнопки пошуку вакансій та адаптації стають активними
Фактичний результат	Збігається з очікуванням.

Таблиця 3.7 – Тест 1.7 Завантаження резюме у форматі .docx

Початковий стан системи	Користувач знаходиться на головній сторінці, користувач не має завантаженого резюме
Вхідні дані	Резюме у форматі .docx
Опис проведення тесту	Користувач натискає кнопку завантаження резюме, обирає резюме у форматі .docx
Очікуваний результат	Відображається повідомлення про успішне завантаження резюме, кнопки пошуку вакансій та адаптації стають активними
Фактичний результат	Збігається з очікуванням.

Таблиця 3.8 – Тест 1.8 Пошук вакансій

Початковий стан системи	Користувач знаходиться на головній сторінці, користувач завантажив резюме
Вхідні дані	-
Опис проведення тесту	Користувач натискає кнопку пошуку вакансій
Очікуваний результат	Користувач переадресовується на сторінку пошуку вакансій, на ній присутні вакансії за резюме користувача, відсортовані за оцінкою релевантності. На кожній вакансії присутня інформація про назву позиції та локацію, оцінку релевантності, та зарплатню за наявності такої інформації
Фактичний результат	Збігається з очікуванням.

Таблиця 3.9 – Тест 1.9 Фільтрація вакансій

Початковий стан системи	Користувач знаходиться на сторінці результатів пошуку
Вхідні дані	Текст для пошуку вакансій, локація
Опис проведення тесту	Користувач вводить дані у поле пошуку вакансій за назвою, обирає локацію з переліку
Очікуваний результат	Відображаються тільки вакансії, в назві яких є введений в поле пошуку текст, а в переліку локацій є обрана користувачем
Фактичний результат	Збігається з очікуванням.

Таблиця 3.10 – Тест 1.10 Адаптація вакансій з пошуку

Початковий стан системи	Користувач знаходиться на сторінці результатів пошуку
Вхідні дані	Вакансія
Опис проведення тесту	Користувач натискає на кнопку адаптації резюме для однієї з вакансій в пошуку.
Очікуваний результат	Відображається вікно з оцінкою співпадіння ключових слів резюме користувача та обраної вакансії, ключові слова пов'язані з навичками що наявні у вакансії але відсутні у резюме
Фактичний результат	Збігається з очікуванням.

Таблиця 3.11 – Тест 1.11 Зчитування тексту вакансії за посиланням

Початковий стан системи	Користувач знаходиться на головній сторінці, має завантажене резюме
Вхідні дані	Посилання на вакансію з сайту dou або postjobfree
Опис проведення тесту	Користувач натискає на кнопку адаптації резюме. Користувач вводить у поле посилання на вакансію та натискає кнопку зчитування тексту вакансії
Очікуваний результат	Поле тексту вакансії заповнюється згідно даних вакансії за введеним посиланням
Фактичний результат	Збігається з очікуванням.

Таблиця 3.12 – Тест 1.12 Зчитування тексту вакансії за не валідним посиланням

Початковий стан системи	Користувач знаходиться на головній сторінці, має завантажене резюме
Вхідні дані	Не валідне посилання на вакансію
Опис проведення тесту	Користувач натискає на кнопку адаптації резюме. Користувач вводить у поле текст що не є посиланням на вакансію dou, postjobfree і натискає на кнопку зчитування тексту
Очікуваний результат	Відображається повідомлення про не валідність посилання
Фактичний результат	Збігається з очікуванням.

Таблиця 3.13 – Тест 1.13 Адаптація резюме за текстом вакансії

Початковий стан системи	Користувач знаходиться на головній сторінці, має завантажене резюме
Вхідні дані	Текст вакансії або посилання
Опис проведення тесту	Користувач відкриває вікно адаптації резюме, вводить посилання на вакансію і натискає зчитування або вводить текст вручну, натискає кнопку адаптації резюме
Очікуваний результат	Відображається вікно з оцінкою співпадіння ключових слів резюме користувача та обраної вакансії, ключові слова пов'язані з навичками що наявні у вакансії але відсутні у резюме
Фактичний результат	Збігається з очікуванням.

Висновки до розділу

В третьому розділі пояснювальної записки до індивідуальної частини дипломного проєкту було проаналізовано якість програмного забезпечення за допомогою статичного аналізатора Qodana та зроблено висновки про якість програмного забезпечення, та наведено тести які були використані для мануального тестування вебзастосунку.

ВИСНОВКИ

В індивідуальній частині дипломного проєкту описано архітектуру вебзастосунку для автоматичного пошуку вакансій та адаптації резюме під вакансії, розроблено алгоритми агрегації вакансій з сайтів пошуку роботи та автоматичного надсилання користувачам імейлів про нові релевантні вакансії за їх резюме.

В першому розділі сформульовано функціональні вимоги індивідуальної частини дипломного проєкту та сформовано матрицю трасування функціональних вимог для демонстрації відповідності вимог до варіантів використання проєкту, наведено постановку завдання на розробку програмного забезпечення.

В другому розділі описано архітектуру вебзастосунку, обґрунтування засобі розробки та наведено метод збору ІТ-вакансій з сайту dou.ua

В третьому розділі наведено оцінку якості розробленого вебзастосунку та описані результати функціонального тестування.

Завдяки реалізації збору вакансій з різних сайтів було досягнуто мету дипломного проєкту - пришвидшення пошуку роботи користувачами, оскільки їм не потрібно переглядати декілька сайтів для пошуку роботи, а також було усунено необхідність моніторингу таких сайтів завдяки реалізації автоматичних повідомлень про нові вакансії, що підходять до резюме користувача.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

- 1) Firebase. URL: <https://firebase.google.com/> (дата звернення: 28.05.2025)
- 2) Mailjet. URL: <https://www.mailjet.com/> (дата звернення: 28.05.2025)
- 3) Angular. URL: <https://angular.dev/> (дата звернення: 28.05.2025)
- 4) NgRx Store. URL: <https://ngrx.io/guide/store> (дата звернення: 28.05.2025)
- 5) ASP.NET. URL: <https://dotnet.microsoft.com/en-us/apps/aspnet> (дата звернення: 28.05.2025)
- 6) Entity Framework. URL: <https://learn.microsoft.com/en-us/aspnet/entity-framework> (дата звернення: 28.05.2025)
- 7) SendGrid. URL: <https://sendgrid.com/en-us> (дата звернення: 28.05.2025)
- 8) JetBrains Qodana. URL: <https://www.jetbrains.com/qodana/> (дата звернення: 28.05.2025)

ДОДАТОК А ЗВІТ ПОДІБНОСТІ



Дата звіту 6/1/2025
Дата редагування 6/1/2025

Документ прийнятий

Звіт подібності

метадані

Назва організації
National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute
Заголовок
ІП-11_Рябов_ПЗ
Автор Науковий керівник / Експерт
РябовЛіщук К.І.
підрозділ
ФІОТ, К-а інформатики та програмної інженерії

Обсяг знайдених подібностей

Коефіцієнт подібності визначає, який відсоток тексту по відношенню до загального обсягу тексту було знайдено в різних джерелах. Зверніть увагу, що високі значення коефіцієнта не автоматично означають плагіат. Звіт має аналізувати компетентна / уповноважена особа.



10
Довжина фрази для коефіцієнта подібності 2

4071
Кількість слів

30656
Кількість символів