УДК 004.75
DOI https://doi.org/10.32782/tnv-tech.2024.6.8

РОЗРОБКА МОБІЛЬНОГО АВТОМАТИЗОВАНОГО АГРЕГАТОРА ВАКАНСІЙ В ІТ-ГАЛУЗІ

***Ольховська О. В.** – кандидат фізико-математичних наук,*
завідувач кафедри комп'ютерних наук та інформаційних технологій
Полтавського університету економіки і торгівлі
ORCID ID: 0000-0001-5366-5995

***Кошова О. П.** – кандидат педагогічних наук,*
доцент кафедри комп'ютерних наук та інформаційних технологій
Полтавського університету економіки і торгівлі
ORCID ID: 0000-0003-0794-6774

***Черненко О. О.** – кандидат фізико-математичних наук,*
доцент кафедри комп'ютерних наук та інформаційних технологій
Полтавського університету економіки і торгівлі
ORCID ID: 0000-0002-9084-0999

***Тур В. М.** – аспірант Полтавського університету економіки і торгівлі*
ORCID ID: 0009-0003-2825-1434

У статті досліджено актуальність розробки автоматизованого агрегатора вакансій для ІТ-сфери, яка є однією з найбільш динамічних галузей сучасного ринку праці. Висвітлено ключові проблеми, пов'язані з пошуком роботи в умовах зростаючої конкуренції та великої кількості вакансій. Описано функціональність розробленого мобільного додатку для операційної системи Android, який дозволяє користувачам здійснювати пошук вакансій на таких популярних платформах, як Work.ua та Jooble. Основною особливістю додатку є використання сучасних технологій та бібліотек, таких як Kotlin, Koin, Jsoup, архітектурного шаблону MVVM і Lottie. Це забезпечує високу продуктивність, модульність та зручність використання.

Алгоритм роботи програми детально описує процес пошуку вакансій, починаючи з введення пошукового запиту, аналізу HTML-сторінок сайтів за допомогою Jsoup, збору необхідної інформації та закінчуючи відображенням результатів користувачу. Також передбачено обробку помилок, фільтрацію вакансій за заданими критеріями та управління станом підключення до Інтернету. Представлено діаграму класів програмного забезпечення, яка демонструє архітектуру основних компонентів додатку, їх взаємозв'язки та залежності. Виокремлено ключові елементи: інтерфейси, репозиторії, моделі даних, ViewModel та фрагменти.

Основні функціональні можливості додатку включають інтеграцію пошуку з різних джерел, парсинг вакансій, динамічні анімації для покращення взаємодії з користувачем, а також зручну навігацію між екранами.

Результати дослідження вказують на ефективність розробленого рішення та його потенціал для розширення. Перспективними напрямами подальших розвідок є інтеграція нових платформ, впровадження машинного навчання для персоналізації пошуку та розробка версій для інших операційних систем.

***Ключові слова:** вакансії у сфері ІТ, архітектура MVVM, UX/UI-дизайн, персоналізація сервісів, оптимізація процесів.*

Olkhovska O. V., Koshova O. P., Chernenko O. O., Tour V. M. Development of a mobile automated vacancy aggregator in the IT industry

The article explores the relevance of developing an automated job aggregator for the IT sector, which is one of the most dynamic sectors of the modern labor market. It highlights key issues related to job searching in the face of growing competition and a large number of vacancies. The

functionality of the developed mobile application for the Android operating system is described, which allows users to search for vacancies on such popular platforms as Work.ua and Jooble. The main feature of the application is the use of modern technologies and libraries, such as Kotlin, Koin, Jsoup, the MVVM architectural pattern and Lottie. This ensures high performance, modularity and ease of use. The program's algorithm describes in detail the process of searching for vacancies, starting from entering a search query, analyzing HTML pages of sites using Jsoup, collecting the necessary information, and ending with displaying the results to the user. It also provides error handling, filtering vacancies according to specified criteria, and managing the state of the Internet connection. A software class diagram is presented, which demonstrates the architecture of the main components of the application, their relationships and dependencies. Key elements are highlighted: interfaces, repositories, data models, ViewModel and fragments.

The main functionalities of the application include integration of search from various sources, job parsing, dynamic animations to improve user interaction, as well as convenient navigation between screens.

The results of the study indicate the effectiveness of the developed solution and its potential for expansion. Promising areas for further exploration include the integration of new platforms, the implementation of machine learning for search personalization, and the development of versions for other operating systems.

Key words: *vacancies in the field of IT, MVVM architecture, UX/UI design, service personalization, process optimization.*

Актуальність роботи. У сучасному світі, де ключовими факторами для багатьох користувачів стають швидкість доступу до інформації та зручність комунікації, месенджери, веб-додатки та інші інформаційні технології набувають широкого застосування у бізнес-комунікаціях, клієнтському обслуговуванні, вивченні ринку праці [1–3]. З кожним роком кількість вакансій в IT-секторі зростає, але водночас зростає й конкуренція серед претендентів на ці позиції. У такій ситуації виникає потреба в ефективних інструментах [4–11], які дозволяють швидко знаходити актуальні вакансії, відповідні навичкам і вимогам кандидатів. Автоматизовані агрегатори вакансій стають важливим елементом у процесі пошуку роботи, оскільки вони значно спрощують та прискорюють цей процес, збираючи вакансії з різних джерел і надаючи користувачам можливість фільтрувати їх за різними параметрами.

Актуальність теми зумовлена зростаючою потребою в автоматизації процесу пошуку вакансій в IT-сфері. Існуючі рішення часто не відповідають вимогам користувачів або ж мають низку обмежень, що знижує їх ефективність. Таким чином, розробка автоматизованого агрегатора вакансій, який враховуватиме сучасні вимоги ринку праці та потреби користувачів, є актуальною та важливою задачею.

Виклад основного матеріалу дослідження. Для розробки програмного забезпечення андроїд застосунку було використано наступні технології: бібліотеки AndroidX та плагінів Kotlin для управління навігацією в додатку; мова програмування Kotlin та фреймворку Koin для управління зв’язками між різними частинами додатку; бібліотеки jsoup для парсингу HTML, що дозволяє виконувати пошук вакансій на веб-сайтах; архітектурний шаблон MVVM (Model-View-ViewModel), який допомагає відокремлювати бізнес-логіку додатку від його інтерфейсу; бібліотека lottie для додавання високоякісних анімацій, які покращують взаємодію користувача з додатком; androidx.navigation: navigation-fragment-ktx, для забезпечення підтримки навігації в додатку, включаючи управління навігаційними потоками та переходами між екранами; середовище розробки Android Studio.

Розглянемо алгоритм збору та фільтрації вакансій у розробленому застосунку базується на пошуку вакансій на двох різних веб-сайтах (Work.ua та Jooble) з використанням бібліотеки Jsoup для обробки HTML-сторінок.

Крок 1. Ініціація пошуку. Пошук вакансій починається з введення користувачем пошукового запиту в інтерфейсі користувача. Після цього застосунок перевіряє, які з джерел пошуку (Work.ua чи Jooble) були вибрані для пошуку вакансій. Залежно від вибраних джерел, застосунок виконує пошук на одному або на обох сайтах.

Крок 2. Пошук на Work.ua.

1. Ініціалізація пошуку. Клас WorkUASearchUseCase викликається для виконання пошуку вакансій на Work.ua.
2. Запит до сайту. Метод getVacancyWorkUAList(query: String) використовує бібліотеку Jsoup для надсилання HTTP-запиту на Work.ua з пошуковим запитом. Адреса запиту формується шляхом додавання пошукового запиту до базового URL Work.ua (https://www.work.ua/jobs-).
3. Обробка HTML-сторінки. Після отримання HTML-сторінки з результатами пошуку Jsoup використовується для пошуку елементів з класом job-link. Це контейнер, який містить інформацію про вакансії.
4. Збір даних про вакансії. Для кожного елемента job-link застосунок витягує:
   – Посилання (link): URL вакансії.
   – Назва вакансії (title): назва посади.
   – Інформація про компанію (companyInfo): назва компанії.
   – Локація (locate): місце розташування вакансії.
   – Опис (desc): опис вакансії.
5. Додавання до списку вакансій. Зібрані дані додаються до списку `listVacancy` у вигляді об'єктів типу `VacancyModel`.
6. Обробка винятків. Якщо під час обробки виникає помилка (наприклад, HTML-елементи не можуть бути знайдені або їх формат неочікуваний), вона обробляється блоком `catch`, і збір даних продовжується для наступного елемента.

Крок 3.Пошук на Jooble.

1. Ініціалізація пошуку. Клас JoobleSearchUseCase викликається для виконання пошуку вакансій на Jooble.
2. Запит до сайту. Метод getVacancyJoobleList(query: String) надсилає HTTP-запит на Jooble з пошуковим запитом. Адреса запиту формується шляхом додавання пошукового запиту до базового URL Jooble (https://ua.jooble.org/SearchResult?p=3&ukw=).
3. Обробка HTML-сторінки. Отримана HTML-сторінка аналізується з використанням Jsoup для пошуку елементів з класом ojoFrF rHG1ci V5WdkE, які містять інформацію про вакансії.
4. Збір даних про вакансії.
5. Додавання до списку вакансій. Зібрані дані додаються до списку listVacancy у вигляді об'єктів типу `VacancyModel`.
6. Обробка винятків. Аналогічно до Work.ua, обробка помилок виконується блоком `catch`.

Крок 4.Об'єднання результатів та відображення.

1. Об'єднання списків вакансій. Після завершення пошуку на обох сайтах (якщо обидва сайти були вибрані) списки вакансій об'єднуються в єдиний список.
2. Фільтрація результатів. Фільтрація результатів може бути додана в залежності від потреб користувача. Наприклад, результати можуть бути відфільтровані за назвою посади, компанією або місцезнаходженням. Ця фільтрація може бути реалізована в методі MainViewModel.getVacancyActiveList().
3. Відображення результатів. Результати передаються до адаптера AdapterForVacancy, який відповідає за відображення списку вакансій в інтерфейсі користувача.

Крок 5. Обробка помилок та відсутності результатів. Якщо пошук на обраному сайті не дає результатів, користувачеві відображається відповідне повідомлення (наприклад, "Немає результатів"). У разі виникнення помилок під час пошуку застосунок обробляє ці помилки і може спробувати повторний пошук або повідомити користувача про проблему.

Покроковий опис роботи застосунку відповідно до блок-схеми (рис. 1):

1. Початок роботи застосунку (Start Application) – застосунок запускається, і починається виконання його логіки.

2. Ініціалізація залежностей (Initialize Dependency Injection) – на цьому етапі здійснюється ініціалізація необхідних залежностей за допомогою механізму Dependency Injection (DI), який дозволяє динамічно впроваджувати об'єкти залежностей.

3. Перевірка інтернет-з’єднання (Check Internet Connection) – застосунок перевіряє наявність інтернет-з'єднання. Це важливо для виконання подальших дій, пов'язаних із пошуком вакансій в Інтернеті. Якщо інтернет-з'єднання відсутнє (No), відображається повідомлення про відсутність інтернету (Show No Internet Message), після чого застосунок завершує роботу (End Application). Якщо інтернет-з'єднання наявне (Yes), застосунок продовжує роботу.

4. Введення пошукового запиту (Input Search Query) – користувач вводить пошуковий запит для пошуку вакансій.

5. Перевірка вибраних джерел (Check Selected Sources) – застосунок перевіряє, які джерела були вибрані для пошуку вакансій (наприклад, Work.ua чи Jooble). Якщо вибрано Work.ua, застосунок здійснює пошук вакансій на сайті Work.ua (Search Work.ua). Якщо вибрано Jooble, застосунок здійснює пошук вакансій на сайті Jooble (Search Jooble).

6. Відображення результатів (Display Results) – після завершення пошуку застосунок відображає результати пошуку вакансій, знайдених на вибраних сайтах.

7. Обробка помилок (Handle Errors) – якщо під час пошуку виникають помилки, застосунок обробляє їх належним чином.

8. Завершення роботи застосунку (End Application) – після відображення результатів або у разі відсутності інтернет-з'єднання застосунок завершує свою роботу.

Діаграма класів, побудована на основі розробленого програмного забезпечення агрегатора пошуку вакансій у IT сфері (див. рис. 2), відображає структуру основних класів, інтерфейсів та їх взаємодію в Android-застосунку для пошуку вакансій.

Основні компоненти діаграми класів:

1. Інтерфейси та їх реалізації:

– Storage (Інтерфейс):

– Визначає методи для роботи з налаштуваннями збереження стану двох параметрів: workUA і jooble. Ці параметри представляють стан (увімкнено чи вимкнено) пошуку на двох різних платформах (Work.ua та Jooble).

– DefaultStorage (Клас):

– Реалізує інтерфейс Storage. Використовує SharedPreferences для збереження та отримання стану параметрів workUA і jooble. Цей клас залежить від `Context` Android для доступу до налаштувань.

2. Репозиторії:

– Repository (Інтерфейс):

– Визначає методи для роботи з станами пошуку вакансій (getWorkUAState, getJoobleState, setWorkUAState, setJoobleState).

– DefaultRepository (Клас):

– Реалізує інтерфейс Repository. Цей клас взаємодіє з Storage для отримання і встановлення стану параметрів пошуку на різних платформах.

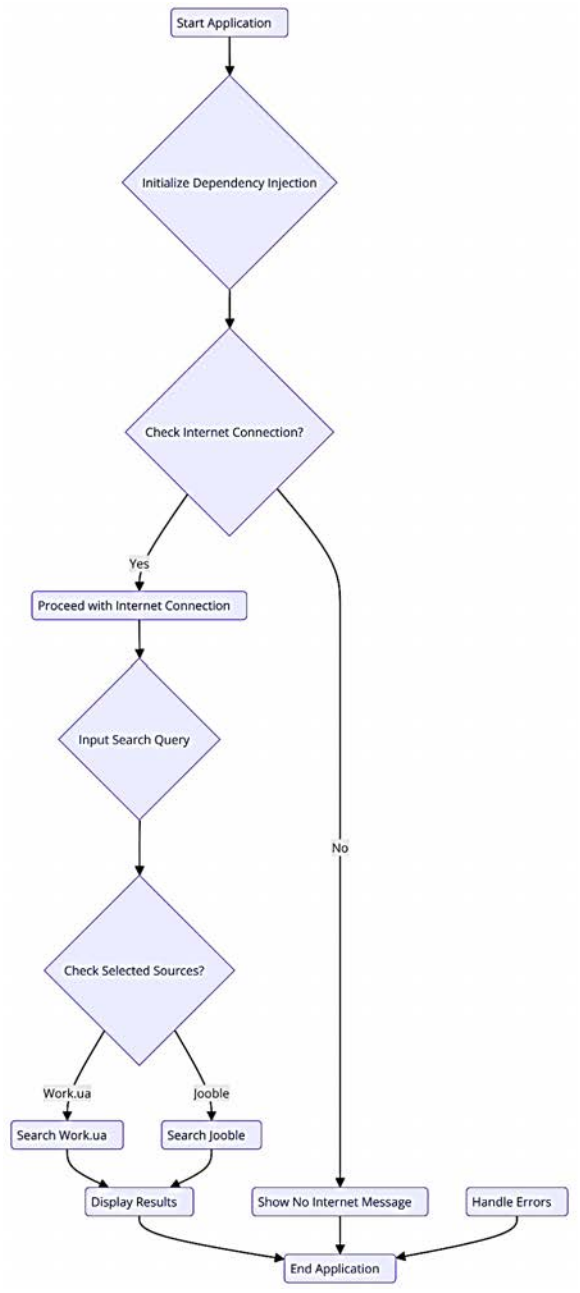

Рис. 1. Блок-схема ПЗ агрегатора вакансій

3. Репозиторії для пошуку:

– SearchRepository (Інтерфейс):

– Визначає методи для отримання списків вакансій з різних джерел (getVacancyWorkUAList, getVacancyJoobleUAList).

– DefaultSearchRepository (Клас):

– Реалізує інтерфейс SearchRepository. Використовує два класи пошуку вакансій WorkUASearchUseCase та JoobleSearchUseCase для отримання списків вакансій з Work.ua та Jooble відповідно.

4. Класи пошукових запитів:

– WorkUASearchUseCase (Клас):

– Відповідає за пошук вакансій на Work.ua. Використовує бібліотеку Jsoup для обробки HTML-сторінок і отримання вакансій.

– JoobleSearchUseCase (Клас):

– Відповідає за пошук вакансій на Jooble. Також використовує Jsoup для обробки HTML-сторінок і отримання вакансій.

5. Моделі даних:

– VacancyModel (Клас):

– Представляє модель вакансії з такими атрибутами, як назва (title), інформація про компанію (companyInfo), місцезнаходження (locate), опис (desc), і посилання (link).

6. Використання Інтернету:

– InternetListenerUseCase (Клас):

– Відповідає за перевірку стану підключення до Інтернету і створення/реєстрацію слухача мережі.

7. ViewModels:

– MainViewModel (Клас):

– Координує запити пошуку вакансій і обробляє отримані результати. Використовує SearchRepository для отримання списків вакансій.

– InternetViewModel (Клас):

– Відповідає за перевірку стану Інтернету і управління підключенням.

8. Адаптери та Фрагменти:

– AdapterForVacancy (Клас):

– Адаптер для відображення списку вакансій у RecyclerView.

– SearchFragment (Клас):

– Фрагмент для управління пошуковим інтерфейсом користувача, обробляє введення користувача та відображає результати пошуку.

Загальна структура і взаємодія.

– Класи DefaultRepository та DefaultSearchRepository реалізують відповідні інтерфейси для доступу до даних і пошуку вакансій.

– DefaultSearchRepository використовує клас WorkUASearchUseCase для пошуку вакансій на Work.ua і JoobleSearchUseCase для Jooble.

– MainViewModel і InternetViewModel керують бізнес-логікою та станом підключення до Інтернету, а також взаємодіють з SearchFragment для відображення результатів.

– Класи AdapterForVacancy і SearchFragment взаємодіють між собою для відображення списку вакансій та обробки введення користувача.

Ця діаграма класів демонструє основні компоненти застосунку, їх взаємозв'язки та залежності, що дозволяє краще зрозуміти його архітектуру та логіку роботи.

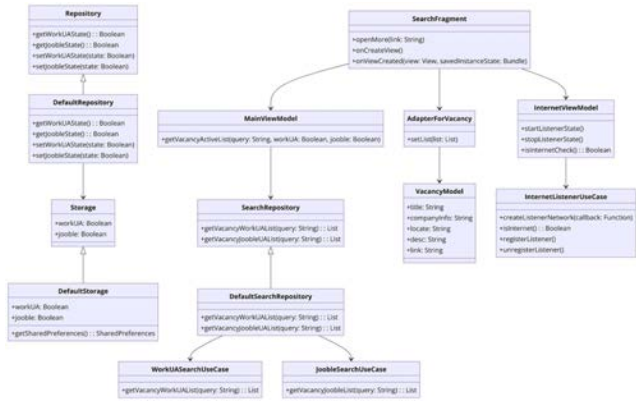

Рис. 2. Діаграма класів ПЗ агрегатора вакансій

Далі розглянемо функціональні можливості застосунку (рис. 3). У верхній частині екрану розміщено заголовок «Пошук пропозиції». Нижче є текстове поле для введення бажаної вакансії та підказка для користувача: «Введи вакансію, яку хочеш знайти». Далі необхідно обрати, на якому сайті знайти ці пропозиції. Після аналізу веб сайтів пошуку роботи, отримуємо перелік актуальних вакансій відповідно до запиту.

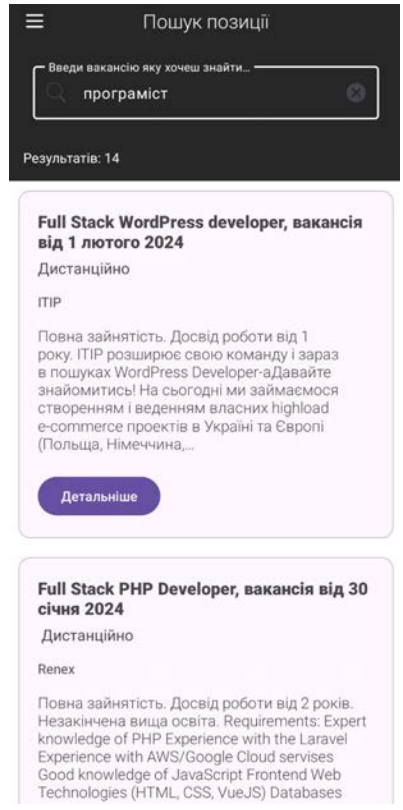

Рис. 3. Актуальні вакансії відповідно до запиту

Серед функціональних можливостей застосунку реалізовані:

1. пошук роботи – користувачі можуть вводити специфічні запити для пошуку роботи, а додаток автоматично парсить відповідні веб-сайти, надаючи актуальні вакансії;

2. парсинг веб-сайтів – з використанням бібліотеки jsoup додаток ефективно аналізує веб-сторінки, витягуючи необхідну інформацію про вакансії за заданими критеріями;

3. керування залежностями – koin спрощує управління залежностями в додатку, забезпечуючи чистоту коду та легкість розширення проекту.

Висновки. Розроблений автоматизований агрегатор вакансій для IT-сфери довів свою ефективність у вирішенні завдань пошуку роботи. Завдяки використанню сучасних технологій, таких як Kotlin, Koin, Jsoup, архітектурного шаблону MVVM та бібліотеки Lottie, вдалося створити функціональний додаток із зручним інтерфейсом і широкими можливостями.

Подальша робота над проектом сприятиме розширенню його функціональних можливостей, підвищенню точності пошуку та задоволенню потреб користувачів, які шукають роботу в умовах сучасного динамічного ринку праці.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ:

1. Слабінога, М. О., Чабан, С. В. Розробка веб-додатків в контексті оптимізації їх швидкодії. *Таврійський науковий вісник. Серія: Технічні науки*, 2022, (3), 63-69. https://doi.org/10.32851/tnv-tech.2022.3.7

2. Антіпова, К. О., Раленко, В. С. Використання штучного інтелекту в розробці Android застосунків. *Таврійський науковий вісник. Серія: Технічні науки*, 2024, (2), 100-105. https://doi.org/10.32782/tnv-tech.2024.2.9

3. Ольховська, О. В., Олексійчук, Ю. Ф., Кошова, О. П., Черненко, О. О., Бойко, О. А. Розробка telegram чат-бота для надання технічної підтримки у галузі туристичних послуг. *Таврійський науковий вісник. Серія: Технічні науки*, 2024, (6), 35-44. https://doi.org/10.32782/tnv-tech.2023.6.

4. Sadhu S. Building Android Apps with Kotlin: Start Building Amazing Applications. Apress, 2020. 350 p.

5. Schildt H. Java: The Complete Reference (11th Edition). McGraw-Hill Education, 2018. 1248 p.

6. Tsang S., Friesen J. Android Recipes: A Problem-Solution Approach. Apress, 2020. 800 p.

7. Nagy G., Borba P. Modern Android Development with Kotlin: Effective Techniques for Building Robust and Maintainable Apps. Manning Publications, 2021. 320 p.

8. Developer workflow basics [Електронний ресурс]. Режим доступу: URL: https://developer.android.com/studio/workflow – Назва з екрану.

9. Dirk Gavor. What Is Lottie and How to Use It for Animations [Електронний ресурс]. Режим доступу: URL: https://www.sliderrevolution.com/design/what-is-lottie/ – Назва з екрану.

10. How Job Aggregators Are Changing The Job Market [Електронний ресурс]. Режим доступу: URL: https://wpjobmanager.com/2023/09/25/job-board-aggregator/ – Назва з екрану.

11. How To Build A Job Aggregator [Електронний ресурс]. Режим доступу: URL: https://niceboard.co/learn/building/how-to-build-a-job-aggregator – Назва з екрану.

REFERENCES:

1. Slabinoha, M. O., Chaban, S. V. (2022) Rozrobka veb-dodatkiv v konteksti optymizatsii yikh shvydkodii [Development of web applications in the context

of optimizing their performance]. *Tavriiskyi naukovyi visnyk. Seriia: Tekhnichni nauky [Taurian Scientific Bulletin. Series: Technical sciences]*, (3), 63-69. https://doi.org/10.32851/tnv-tech.2022.3.7 [in Ukrainian].

2. Antipova, K. O., Ralenko, V. S.(2024) Vykorystannia shtuchnoho intelektu v rozrobtsi Android zastosunkiv [The use of artificial intelligence in the development of Android applications]. *Tavriiskyi naukovyi visnyk. Seriia: Tekhnichni nauky [Taurian Scientific Bulletin. Series: Technical sciences]*, (2), 100-105. https://doi.org/10.32782/tnv-tech.2024.2.9 [in Ukrainian].

3. Olkhovska, O. V., Oleksiichuk, Yu. F., Koshova, O. P., Chernenko, O. O., Boiko, O. A. (2024) Rozrobka telegram chat-bota dlia nadannia tekhnichnoi pidtrymky u haluzi turystychnykh posluh [Development of a telegram chatbot for providing technical support in the field of tourist services]. *Tavriiskyi naukovyi visnyk. Seriia: Tekhnichni nauky [Taurian Scientific Bulletin. Series: Technical sciences]*, (6), 35-44. https://doi.org/10.32782/tnv-tech.2023.6 [in Ukrainian].

4. Sadhu S. (2020) Building Android Apps with Kotlin: Start Building Amazing Applications. Apress, 350 p. [in English].

5. Schildt H. (2018) Java: The Complete Reference (11th Edition). McGraw-Hill Education, 1248 p. [in English].

6. Tsang S., Friesen J. (2020) Android Recipes: A Problem-Solution Approach. Apress, 800 p. [in English].

7. Nagy G., Borba P. (2021) Modern Android Development with Kotlin: Effective Techniques for Building Robust and Maintainable Apps. Manning Publications, 320 p. [in English].

8. Developer workflow basics [Електронний ресурс]. Режим доступу: URL: https://developer.android.com/studio/workflow – Назва з екрану [in English].

9. Dirk Gavor. What Is Lottie and How to Use It for Animations [Електронний ресурс]. Режим доступу: URL: https://www.sliderrevolution.com/design/what-is-lottie/ – Назва з екрану [in English].

10. How Job Aggregators Are Changing The Job Market [Електронний ресурс]. Режим доступу: URL: https://wpjobmanager.com/2023/09/25/job-board-aggregator/ – Назва з екрану [in English].

11. How To Build A Job Aggregator [Електронний ресурс]. Режим доступу: URL: https://niceboard.co/learn/building/how-to-build-a-job-aggregator – Назва з екрану/ [in English].