НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Факультет прикладної математики

Кафедра програмного забезпечення комп'ютерних систем

«До захисту допущено»
Завідувач кафедри
_____ Євгенія СУЛЕМА
«___»_____ 2023 р.

Дипломний проєкт

на здобуття ступеня бакалавра

за освітньо-професійною програмою «Інженерія програмного забезпечення мультимедійних та інформаційно-пошукових систем» спеціальності 121 Інженерія програмного забезпечення
на тему: «Програмний застосунок «Inner Growth» для управління розвитком компетенцій співробітників»

Виконала:
студентка IV курсу, групи КП-91
Костікова Катерина Вадимівна _____

Керівник:
Доцент кафедри ПЗКС, к.т.н., доцент,
Люшенко Леся Анатоліївна _____

Консультант з нормоконтролю:
Доцент кафедри ПЗКС, к.т.н., доцент,
Онай Микола Володимирович _____

Рецензент:
Доцент кафедри СПСКС, к.т.н, доцент,
Тарасенко-Клятченко Оксана Володимирівна _____

Засвідчую, що у цьому дипломному проєкті немає запозичень з праць інших авторів без відповідних посилань.
Студентка _____

Київ – 2023 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет прикладної математики
Кафедра програмного забезпечення комп'ютерних систем

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 121 «Інженерія програмного забезпечення»

Освітньо-професійна програма «Інженерія програмного забезпечення мультимедійних та інформаційно-пошукових систем»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Євгенія СУЛЕМА
«__»_____ 2022 р.

ЗАВДАННЯ

на дипломний проєкт студенту

Костіковій Катерині Вадимівні

1. Тема проєкту «Програмний застосунок «Inner Growth» для управління розвитком компетенцій співробітників», керівник проєкту Люшенко Леся Анатоліївна, доцент кафедри ПЗКС, к.т.н., доцент, затверджені наказом по університету від «31» травня 2023 р. № 2107-с

2. Термін подання студентом проєкту «16» червня 2023 р.

3. Вихідні дані до проєкту: див. Технічне завдання.

4. Зміст пояснювальної записки:
   - аналіз існуючих рішень поставленої задачі;
   - обґрунтування вибору засобів реалізації;
   - розроблення застосунку «Inner Growth»;
   - аналіз розробленого застосунку.

5. Перелік обов'язкового графічного матеріалу:
   - діаграма діяльності основного сценарію використання (креслення);
   - схема бази даних (креслення);
   - діаграма варіантів використання програмного застосунку (плакат);
   - архітектура програмного застосунку (плакат).

6. Консультанти розділів проєкту

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Нормоконтроль	Онай М.В., доцент		

7. Дата видачі завдання «30» жовтня 2022 р.

Календарний план

№ з/п	Назва етапів виконання дипломного проєкту	Термін виконання етапів проєкту	Примітка
1.	Вивчення літератури за тематикою проєкту	15.11.2022	
2.	Розроблення та узгодження технічного завдання	27.11.2022	
3.	Розроблення архітектури програмного застосунку	16.12.2022	
4.	Розроблення дизайну сторінок та графічних елементів	03.02.2023	
5.	Підготовка першого розділу дипломного проєкту	20.03.2023	
6.	Підготовка другого розділу дипломного проєкту	14.04.2023	
7.	Програмна реалізація програмного застосунку	29.04.2023	
8.	Підготовка третього розділу дипломного проєкту	17.03.2021	
9.	Підготовка матеріалів третього розділудипломного проєкту	05.05.2023	
10.	Тестування програмного застосунку	07.05.2023	
11.	Підготовка четвертого розділу дипломного проєкту	13.05.2023	
12.	Підготовка матеріалів графічної частини проєкту	27.05.2023	
13.	Оформлення технічної документації проєкту	29.05.2023	

Студент Катерина КОСТІКОВА

Керівник проєкту Леся ЛЮШЕНКО

АНОТАЦІЯ

Даний дипломний проєкт присвячений розробці програмного застосунку для управління розвитком компетенцій співробітників.

У роботі виконано обґрунтування та аналіз тематики проєкту, вивчено наявні рішення для розв'язання поставленої задачі, обрано засоби для реалізації проєкту, проведено аналіз архітектури та структури програмного забезпечення, а також оцінено розроблене програмне забезпечення.

Програмний застосунок реалізовано у вигляді трьох модулів: мобільний, браузерний та серверний. Надає можливість компаніям відслідковувати розвиток співробітників. В якості методології цілепокладання було обрано Objective Key Result. В особистому профілі користувача міститься вся інформація про вміння та досвід співробітника. Користувач може подавати заявки на участь у персональних можливостях для розвитку. Співробітник може створювати, відслідковувати по отримувати результат кожного циклу OKR. Адміністратор реєструє користувачів в системі та надає інформацію про компанію.

Програмний застосунок забезпечує безпеку інформації шляхом обмеження доступу лише для користувачів, які зареєстровані і авторизовані. Тільки після реєстрації та входу в систему користувачам надається можливість переглядати сторінки та використовувати функціонал мобільного застосунку. Також додано перевірку введених даних для всіх полів вводу користувача.

У даному дипломному проєкті було розроблено архітектуру та структуру програмного застосунку, алгоритм взаємодії графічних елементів, логіку різних ролей користувачів і відслідковування розвитку компетенцій працівників, а також інтерфейс та дизайн програмного застосунку.

ABSTRACT

This diploma project is dedicated to the development of a software application for managing the development of employee competencies.

The work includes justification and analysis of the project topic, study of existing solutions to solve the problem, selection of tools for project implementation, analysis of the software architecture and structure, and evaluation of the developed software.

The software application is implemented in three modules: mobile, browser, and server. It allows companies to track employee development. Objective Key Result was chosen as the goal-setting methodology. The user's personal profile contains all the information about the employee's skills and experience. The user can apply for participation in personal development opportunities. The employee can create, track and receive the result of each OKR cycle. The administrator registers users in the system and provides information about the company.

The software application ensures the security of information by restricting access only to users who are registered and authorised. Only after registration and logging in users can view pages and use the functionality of the mobile application. Validation of the entered data for all user input fields has also been added.

In this diploma project, I have developed the architecture and structure of the software application, the algorithm for the interaction of graphic elements, the logic of different user roles and tracking the development of employee competencies, as well as the interface and design of the software application.

ДП.045440-01-90 Програмний застосунок «Inner Growth» для управління розвитком компетенцій співробітників.
Відомість проєкту

Позначення	Найменування	Кіл-ть	Примітка
	Документація проєкту		
ДП.045440-02-91	Програмний застосунок	5	
	«Inner Growth» для		
	управління розвитком		
	компетенцій співробітників.		
	Технічне завдання		
ДП.045440-03-81	Програмний застосунок	66	
	«Inner Growth» для		
	управління розвитком		
	компетенцій співробітників.		
	Пояснювальна записка		
ДП.045440-04-51	Програмний застосунок	4	
	«Inner Growth» для		
	управління розвитком		
	компетенцій співробітників.		
	Програма та методика		
	тестування		
ДП.045440-05-34	Програмний застосунок	35	
	«Inner Growth» для		
	управління розвитком		
	компетенцій співробітників.		
	Керівництво користувача		

Позначення	Найменування	Кіл-ть	Примітка
ДП.045440-06-99	Програмний застосунок	1	
	«Inner Growth» для		
	управління розвитком		
	компетенцій співробітників.		
	Діаграма діяльності		
	основного сценарію		
	використання. Схема		
	діаграми		
ДП.045440-07-99	Програмний застосунок	1	
	«Inner Growth» для		
	управління розвитком		
	компетенцій співробітників.		
	Схема бази даних. Схема		
	даних		
ДП.045440-08-98	Програмний застосунок	1	
	«Inner Growth» для		
	управління розвитком		
	компетенцій співробітників.		
	Компакт-диск		

Факультет прикладної математики

Кафедра програмного забезпечення комп'ютерних систем

"ЗАТВЕРДЖЕНО"

Завідувач кафедри

_____ Євгенія СУЛЕМА

"___"_____ 2022 р.

ПРОГРАМНИЙ ЗАСТОСУНОК «INNER GROWTH» ДЛЯ УПРАВЛІННЯ РОЗВИТКОМ КОМПЕТЕНЦІЙ СПІВРОБІТНИКІВ

Технічне завдання

ДП.045440-02-91

"ПОГОДЖЕНО"

Керівник проєкту:

_____ Леся ЛЮШЕНКО

Нормоконтроль:

_____ Микола ОНАЙ

Виконавець:

_____ Катерина КОСТІКОВА

2022

ЗМІСТ

1. НАЙМЕНУВАННЯ ТА ГАЛУЗЬ ЗАСТОСУВАННЯ

Назва розробки: Програмний застосунок «Inner Growth» для управління розвитком компетенцій співробітників.

Галузь застосування: інформаційні технології.

2. ПІДСТАВА ДЛЯ РОЗРОБЛЕННЯ

Підставою для розроблення є завдання на дипломне проектування, затверджене кафедрою програмного забезпечення комп'ютерних систем Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського» (КПІ ім. Ігоря Сікорського).

3. ПРИЗНАЧЕННЯ РОЗРОБКИ

Розробка призначена для використання компаніями в якості програмного забезпечення для відслідковування та контролю за розвитком фахових та поведінкових компетенцій співробітників.

4. ВИМОГИ ДО ПРОГРАМНОГО ПРОДУКТУ

Програмний застосунок повинен забезпечувати такі основні функції:

1) Створення акаунту компанії адміністратором та додавання акаунтів співробітників та менторів.
2) Налаштування особистого профілю співробітника чи ментора.
3) Можливість для співробітника подавати заявку на участь у можливостях для розвитку.
4) Створення та відслідковування прогресу OKR.

Додаткові вимоги:

1) Надання користувачам безперебійного доступу до додатку.
2) Зручний та нативно зрозумілий користувацький інтерфейс.

3) Забезпечення безпеки і конфіденційності даних про компанію та співробітників.

5. ВИМОГИ ДО ПРОЄКТНОЇ ДОКУМЕНТАЦІЇ

У процесі виконання проекту повинна бути розроблена наступна документація:

1) пояснювальна записка;

2) програма та методика тестування;

3) керівництво користувача;

4) креслення:

- «Діаграма діяльності основного сценарію використання»;
- «Схема бази даних».

6. ЕТАПИ ПРОЄКТУВАННЯ

Вивчення літератури за тематикою роботи 15.11.2022
Розроблення та узгодження технічного завдання 27.11.2022
Розроблення архітектури програмного застосунку..... 16.12.2022
Розроблення дизайну сторінок та графічних елементів 03.02.2023
Підготовка першого розділу дипломного проєкту..... 20.03.2023
Підготовка другого розділу дипломного проєкту 14.04.2023
Програмна реалізація програмного застосунку..... 29.04.2023
Підготовка третього розділу дипломного проєкту..... 05.05.2023
Тестування програмного застосунку 07.05.2023
Підготовка четвертого розділу дипломного проєкту..... .13.05.2023
Підготовка матеріалів графічної частини проєкту..... 27.05.2023
Оформлення технічної документації проєкту..... 29.05.2023

7. ПОРЯДОК ТЕСТУВАННЯ РОЗРОБКИ

Тестування розробленого програмного застосунку виконується відповідно до “Програми та методики тестування”.

Факультет прикладної математики

Кафедра програмного забезпечення комп'ютерних систем

«ЗАТВЕРДЖЕНО»

Завідувач кафедри

_____ Євгенія СУЛЕМА

«___»_____ 2023 р.

ПРОГРАМНИЙ ЗАСТОСУНОК «INNER GROWTH» ДЛЯ УПРАВЛІННЯ РОЗВИТКОМ КОМПЕТЕНЦІЙ СПІВРОБІТНИКІВ

Пояснювальна записка

ДП.045440-03-81

"ПОГОДЖЕНО"

Керівник проєкту:

_____ Леся ЛЮШЕНКО

Нормоконтроль:

_____ Микола ОНАЙ

Виконавець:

_____ Катерина КОСТІКОВА

2023

ЗМІСТ

СПИСОК ТЕРМІНІВ, СКОРОЧЕНЬ ТА ПОЗНАЧЕНЬ

Програмний застосунок (ПЗ) – користувацька комп'ютерна програма створена для вирішення конкретних прикладних завдань користувача.

База даних (БД) – представляє собою колекцію інформації, яка впорядкована та організована за певними принципами, що описують її характеристики і взаємозв'язки між її компонентами.

Інтерфейс користувача (UI) – це засіб, який дозволяє користувачу зручно взаємодіяти з інформаційною системою. Він включає в себе набір інструментів, які обробляють і відображають інформацію, що максимально відповідають потребам та зручності користувача.

Операційна система (ОС) – набір програм, які виконують управління апаратною частиною комп'ютера або віртуальної машини. Вона забезпечує керування обчислювальним процесом та організовує взаємодію з користувачем.

Технологічна інтенсивність – використання наявних технологій для створення нових продуктів або рішень.

Продуктивність персоналу – ефективність та результативність, з якою працівники виконують свої обов'язки та досягають поставлених цілей. Вона вимірюється шляхом порівняння виробничих результатів або виконання завдань з ресурсами, витраченими на їх досягнення, такими як час, праця, гроші та інші ресурси.

Objective Key Result (OKR) – це методологія спільного цілепокладання, яка допомагає командам і окремим особам встановлювати складні й амбітні цілі з вимірюваними результатами. Цей підхід дозволяє відстежувати прогрес, координувати дії та стимулювати співпрацю для досягнення вимірюваних цілей.

Key Performance Iindicators (KPI) – це набір кількісних вимірювань, що використовуються для оцінки ефективності роботи компанії.

REST API – (Representational State Transfer, «передача репрезентативного стану») – підхід до архітектури мережевих протоколів, які надають доступ до інформаційних ресурсів.

HTTP – протокол для передачі даних, який використовується в комп'ютерних мережах. Назва скорочена від Hyper Text Transfer Protocol, протокол передачі гіпертекстових документів.

MVP (англ. Minimum viable product) – продукт з мінімальним функціоналом, який можна дати користувачам для використання. Використовується для тестування ідей у розробці програм з мінімальними затратами ресурсів.

Model-View-ViewModel (MVVM) – це патерн проєктування програмного забезпечення, який структурований таким чином, щоб розділити логіку програми та елементи керування користувацьким інтерфейсом.

JSON (JavaScript Object Notation) – це формат даних, який використовується для легкого обміну інформацією між системами. Він використовує зрозумілий для людини синтаксис, що складається з пар ключ-значення або масивів, завдяки чому широко використовується у веб-програмах та API.

ВСТУП

У високотехнологічних галузях, таких як інформаційні технології (IT), успіх і конкурентоспроможність компаній значною мірою залежать від фахових компетенцій, досвіду та творчого потенціалу співробітників. Постійне зростання та розвиток компаній вимагають актуальних компетенцій співробітників, а це в свою чергу вимагає наявність сучасних інструментів управління їх розвитком. За результатами опитування, проведеного серед осіб, відповідальних за прийняття рішень у бізнесі, 75% респондентів вбачають розвиток технологічної інтенсивності як найефективніший спосіб створення конкурентних переваг. Технологічна інтенсивність визначається здатністю фахівців до постійного навчання і наявністю культури навчання персоналу у компанії [1].

Цей дипломний проект присвячений створенню інструменту для управління розвитком компетенцій персоналу компанії з можливістю моніторингу досягнення цілей та менторської підтримки. Таким інструментом є програмний застосунок “Inner Growth” для управління розвитком фахових та поведінкових компетенцій співробітників. За допомогою цього програмного забезпечення компанії зможуть створювати персоналізовані плани розвитку для кожного співробітника, враховуючи його поточні компетенції, кар'єрні прагнення та вимоги організації. Програмне забезпечення надаватиме можливості для розвитку щодо відповідних курсів, навчальних програм та навчальних ресурсів для ефективного подолання виявлених прогалин.

Основна мета дипломного проєкту створити програмний застосунок “Inner Growth”, який дозволить визначати, планувати та відстежувати, а також оцінювати розвиток компетенцій співробітників компанії. Таким чином, це дає можливість компаніям узгоджувати індивідуальні цілі співробітників з бізнес-цілями компанії, сприяти професійному зростанню та забезпечувати висококваліфіковану робочу силу, здатну відповідати

потребам галузі. Окрім того «Inner Growth» забезпечить моніторинг прогресу у розвитку компетенцій співробітників в режимі реального часу. Це дасть змогу керівникам та спеціалістам з управління персоналом переглядати індивідуальну ефективність у досягнені поставлених цілей, надавати своєчасний зворотній зв'язок та приймати обґрунтовані рішення щодо просування по службі, переходу на іншу посаду тощо.

1. АНАЛІЗ ІСНУЮЧИХ РІШЕНЬ ПОСТАВЛЕНОЇ ЗАДАЧІ

1.1. Загальні положення та аналіз предметної області

Сучасні тенденції розвитку IT полягають у впроваджуються новітніх технологій, що в свою чергу вимагає постійного фахового розвитку персоналу IT компаній. Згідно з дослідженням від компанії Microsoft, результати показують пряму залежність успіху компанії на IT ринку від використання сучасних технологій.

Відповідно до дослідження, проведеного міжнародною мережею компаній Deloitte, організації, які мають високу культуру навчання, частіше виходять на ринок першими з інноваційними продуктами або рішеннями на 56% більше, порівняно з іншими компаніями [1]. Тому для IT компаній потрібно постійно зважати на важливість навчання співробітників та формуванні актуальних компетенцій персоналу.

Отже, IT компаніям необхідно мати змогу оперативно відслідковувати та аналізувати прогрес розвитку фахових чи поведінкових компетенцій співробітників. Тому виникає потреба в створенні відповідних програмних рішень для управління розвитком компетенцій персоналу високотехнологічних компаній, насамперед в IT.

Для фіксування та аналізу результатів прогресу розвитку компетенцій персоналу обрано методику управління цілями «Objectives and Key Results» (OKR), яка дозволяє ставити та погоджувати цілі на всіх рівнях управління організацією: від стратегічного рівня організації до індивідуального рівня співробітника. Ця методика дозволяє здійснювати моніторинг шляхів досягнення цілей та контролювати ключові результати розвитку персоналу, а також мотивувати співробітників розвиватися та тримати фокус на визначених цілях. Переваги OKR полягають у встановленні конкретних, вимірюваних, досяжних, релевантних та обмежених у часі цілей, а також у визначенні ключових результатів, які допомагають відстежувати прогрес і вимірювати успіх. Для інтенсивного впровадження та використання

методики OKR актуальним є програмне рішення, яке дозволить компаніям підвищити продуктивність в розвитку компетенцій співробітників та тримати високий рівень управління процесом підвищення кваліфікації персоналу високотехнологічних компаній.

Ринок програмних застосунків, які створені для моніторингу розвитку за методикою OKR має достатню кількість програмних рішень. Далі розглянемо застосунки, які найбільш доступні та орієновані саме на індивідуальний підхід до розвитку співробітників в компанії.

1.2. Аналіз аналогічних застосунків

Після проведення аналізу обраної предметної області було досліджено доступні вебзастосунки та мобільні застосунки з торгівельних площадок “Google Play Market” та “AppStore”. Розглянемо декілька з них, а саме “*Weekdone*”, “*Lattice*”, “*Perdoo*” та “*Plai*”.

Weekdone (https://weekdone.com/)

Weekdone – це програмне забезпечення основане на методиці OKR, яке допомагає користувачам встановлювати структуровані цілі. Weekdone дозволяє відстежувати прогрес досягнення в інтервалі одного тижня та контролювати хід виконання поставлених завдань. Також передбачено звітування про виконання завдань та коментування прогресу.

Програмне забезпечення “*Weekdone*” є вебдзастосунком та має мобільний застосунок для IOS та Android. Доступний у браузерах Chrome, Microsoft Edge та інших найбільш відомих браузерах. Вимоги для IOS це IPhone чи IPad з встановленою IOS версією не менше 9.0.

Системні вимоги для Android:

- Версія Android не менше за 4.4.
- Мінімально 512 мегабайт оперативної пам’яті.
- Мінімум 500 мегабайт вільної постійної пам’яті.

Переваги “Weekdone”:

1. Оптимізовані процеси обміну повідомленнями та відстеження цілей.
2. Можливість коментування прогресу іншими користувачами.
3. Можливість залучення декількох людей до спільного OKR.

Недоліки “Weekdone”:

1. Повільність ПЗ через перенасиченість інтерфейсу.
2. Застосунок більше спрямований для планування проєкту, а не розвитку персоналу.
3. Планувати цілі лише в рамках одного тижня.

Lattice (https://lattice.com/)

Lattice – це платформа для аналізу продуктивності персоналу, яка пропонує різні інструменти для моніторингу цілей, інтеграції та консолідації OKR. Серед інструментів наявні такі можливості: чат для відповідей на питання та пропозиції покращення робочих процесів від співробітників; відслідковування розвитку успіху за методологією OKR; матриця фахових компетенцій; перегляд компенсацій згідно розвитку. Програмне забезпечення допомагає інтегрувати OKR компанії в робоче навантаження та завдання своїх співробітників.

Вебзастосунок “*Lattice*” доступний у браузерах Chrome, Microsoft Edge та інших найбільш відомих браузерах.

Переваги “Lattice”:

1. Інтеграція з програмами Salesforce, Slack та Microsoft Teams.
2. Аналітика, яка дозволяє у подальшому генерувати ідеї для покращення продуктивності.

Недоліки “Lattice”:

1. Відсутня можливість відстеження прогресу та коментування іншими співробітниками. Тобто ORK доступний лише для одного працівника.
2. Не працює на мобільних версіях браузерів.

Perdoo (https://www.perdoo.com/)

Perdoo – це програмне забезпечення, яке складається з мобільного і браузерного застосунків. Надає можливість організаціям впроваджувати та стимулювати професійне зростання за допомогою OKR, Key Performance Iindicators (KPI) тощо. Тобто воно дозволяє користувачам пов'язувати методики OKR та KPI. ПЗ допомагає організаціям аналізувати та планувати поставлені цілі, такі як мапа кар’єрного розвитку.

Мобільний застосунок доступний для IOS та Android. Вимоги для IOS це IPhone чи IPad з встановленою IOS версією не менше 12.4.

Системні вимоги для Android:

- Версія Android не менше за 5.0.
- Мінімально 512 мегабайт оперативної пам’яті.
- Мінімум 350 мегабайт вільної постійної пам’яті.

Переваги “Perdoo”:

1. Наявне створення та планування мапи кар’єрного розвитку для фахових компетенцій співробітників.
2. Аналітика щодо досягнення поставлених цілей та просуванню по кар’єрній мапі розвитку.
3. Інтеграція зі Slack або Microsoft Teams.

Недоліки “Perdoo”:

1. Відсутня можливість відстеження прогресу та коментування іншими співробітниками. Тобто ORK доступний лише для одного працівника.
2. Мобільна версія пропонує обмежені можливості та функції.

Plai (https://www.plai.team/)

Plai – це мобільний застосунок для управління персоналом і продуктивністю. Plai надає користувачам переглядати та відстежувати активні OKR усіх співробітників компанії. Також застосунок надає можливість планувати зустрічі та надавати фідбек співробітникам. У застосунку наявні інтеграції зі Slack та Microsoft Teams.

Переваги “Perdoo”:

1. Можливість надання коментарів стосовно прогресу виконання чи результату OKR.

Недоліки “Perdoo”:

1. Відсутня приватність та розподілення користувачів по ролям у застосунку.
2. Застосунок має прив’язку до ОС мобільного пристрою користувача.

Мобільний застосунок “Plai” доступний лише для ОС Android.

Системні вимоги:

- Версія Android не менше за 5.0.
- Мінімально 512 мегабайт оперативної пам’яті.
- Мінімум 400 мегабайт вільної постійної пам’яті.

1.3. Актуальність розробки програмного застосунку “Inner Growth”

Проведемо порівняльний аналіз застосунків-аналогів, які створені для впровадженйня системи управління розвитком співробітників (табл. 1).

Таблиця 1

Порівняння функціональності оглянутих застосунків і запропонованого рішення

Застосунок	Weekdone	Lattice	Perdoo	Plai
Наявність внутрішньої особистої інформації (резюме) співробітників	–	–	+	–
Кросплатформеність	+	–	+	–
Наявність мапи карє’рного розвитку	–	–	+	–

Продовження табл. 1

Наявність виду користувача Ментор, який покращує процес розвитку	+	–	–	–
Наявність можливості обирати рекомендовані можливості для розвитку	+	+	–	–

Розроблюваний програмний застосунок “Inner Growth”, на відміну від проаналізованих аналогів, буде кросплатформним, тобто працюватиме і на Android, і на IOS. Програмний застосунок буде мати наступні конкурентні переваги:

- Застосунок буде орієнтованим на індивідуальний підхід до розвитку компетенцій співробітників.
- Програмний застосунок буде мати особистий профіль співробітника з елементами резюме, що допоможе, за необхідністю, обрати нову сферу діяльності для співробітника в компанії.
- Наявність мапи кар’єрного розвитку з визначенням опанованих компетенцій. Співробітнику надається можливість підвищувати рівень поточної кваліфікації, або взагалі почати розвиватися у новій сфері діяльності. У мапі кар’єрного розвитку буде представлено список навичок, які необхідні для покращення компетентності у певній сфері.
- Кожному співробітнику буде призначений особистий ментор компанії, який буде мотивувати та наставляти у подальшому просуванні у певній сфері діяльності.
- Надано вибір персональних можливостей для розвитку компетенцій, базуючись на уміннях та навичках співробітника.

Виходячи з вище зазначеного, актуальним є створення кросплатформного програмного застосунку “Inner Growth” для управлінням розвитком компетенцій співробітників компаній за методологією OKR на основі індивідуального підходу. Окрім того, “Inner Growth” буде мати інструменти: розробки та актуалізації мапи кар’єрного розвитку, комунікації з особистим ментором та контролю прогрес розвитку компетенцій шляхом створення цілей та завдань у вигляді OKR карток.

Застосунок “Inner Growth” матиме такі функціональні можливості:

1) Створення аккаунту компанії адміністратором та додавання акаунтів співробітників та менторів.
2) Налаштування особистого профілю співробітника чи ментора.
3) Визначення ментором поточного рівня у певній області співробітників.
4) Вибір співробітником шляху розвитку за допомогою мапи кар’єрного розвитку.
5) Створення та відслідковування прогресу OKR.

1.4. Загальні вимоги до застосунку

Провівши аналіз конкурентних рішень у програмних застосунках та проаналізувавши їх переваги і недоліки, було визначено перелік функціональних та нефункціональних вимог до застосунку.

Застосунок має складатися з таких компонент:

- Користувацька, якою будуть користуватися співробітники.
- Менторська, якою будуть користуватися співробітники, які є менторами для інших працівників компанії.
- Адміністраторська, на якій адміністратор може додавати компанію, співробітники та мапи кар'єрного розвитку.
- Серверна частина, яка буде об'єднувати всі вищезазначені компоненти.

Користувацька компонента має бути реалізована у вигляді мобільного застосунку. Функціональні вимоги користувача до застосунку:

- авторизація за допомогою електронної пошти після реєстрації акаунту адміністратором;
- вибір рекомендованих можливостей для розвитку;
- можливість налаштовувати свій персональний профіль (додавати інформацію про себе, свої вміння);
- можливість переглядати індивідуальні напрями розвитку у вигляді мапи кар'єрного розвитку;
- можливість створювати, переглядати та завершувати етап плану розвитку на певний період за методологією OKR;
- перегляд прогресу свого розвитку.

Адміністраторська компонента, яка буде реалізована у вигляді браузерної сторінки, має забезпечувати можливості для адміністратора:

- створити сторінку компанії;
- створити/видалити акаунти для працівників та назначати менторів;
- створити, редагувати чи видалити мапу кар'єрного розвитку.

Менторська компонента має бути реалізована у вигляді мобільного застосунку (бути частиною користувацької компоненти) і має відповідати наступним функціональним вимогам:

- можливість переглядати профіль користувача (співробітника);
- можливість підтверджувати рівень професійності співробітника та його навичок.

Розроблені компоненти мають виконувати наступні нефункціональні вимоги:

1) надавати користувачам безперебійний доступ до застосунку;
2) зручний та нативно зрозумілий користувацький інтерфейс;
3) забезпечити безпеку і конфіденційність даних про компанію та співробітників.

Вимоги до якості розроблюваного ПЗ будуть реалізовані завдяки:

- проведенню функціонального тестування програмного застосунку;
- проведенню тестування запитів до сервера;
- шифрованому доступ до даних користувачів та хешування паролів при зберіганні їх у базі даних.

1.5. Вимоги до безпеки застосунку

Було проведено аналіз ризиків та визначено вимоги до безпеки даних застосунку, принципи доступності застосунку та оцінку потенційних ризиків, які можуть виникнути під час використання ПЗ.

Сформулюємо можливі негативні варіанти, з якими можна зустрітись при перебоях роботи серверу, непередбачуваних сценаріїв використання застосунку чи спробах зламу ПЗ.

Потенційно небезпечними та найбільш ризикованими вразливостями для застосунку є:

- відсутність перевірки введених даних чи неправильна перевірка;
- відсутність обробки запиту та переданих параметрів до бази даних;
- збереження даних без попереднього шифрування;
- неправильна логіка розподілення ролей у застосунку.

Докладніший опис вразливостей та потенційних загроз для розроблюваного застосунку, з наслідками для безпеки компанії та користувачів детально описані у табл. 2.

Таблиця 2

Ідентифіковані вразливості програмного забезпечення “Inner Growth”

№	Вразливість	Загроза	Наслідки
1.1	Відсутність перевірки введених даних чи неправильна перевірка	Відправка непередбачуваних даних на сервер	Часткова або повна нестабільність у роботі застосунку

Продовження табл. 2

1.2	Відсутність перевірки введених даних чи неправильна перевірка	Заборонена відправка очікуваних даних на сервер	Часткова або повна нестабільність у роботі застосунку
2.1	Відсутність обробки запиту та переданих параметрів до бази даних	SQL-ін'єкція у базу даних чи загрозливий код	Пошкодження, видалення чи викрадення даних із бази, взлом застосунку
3.1	Збереження даних без попереднього шифрування	Доступ до конфіденційних даних компанії чи користувачів	Викрадення конфіденційних даних компанії чи користувачів

Для збереження довіри користувачів до розроблюваного ПЗ було проаналізовано ризики та створено таблицю вимог до безпеки застосунку. Буде забезпечено безпеку бази даних та безпеку доступу до сторінок по ролям.

Створені вимоги до безпеки програмного забезпечення зазначені у табл. 3.

Таблиця 3

Оброблені безпекові ризики програмного забезпечення “Inner Growth”

№	Вразливість	Загроза	Безпекова вимога
1.1	Відсутність перевірки введених даних чи неправильна перевірка	Відправка непередбачуваних даних на сервер	Забезпечення перевірки введених даних для усіх користувацьких ролей при користуванні застосунком
1.2		Заборонена відправка очікуваних даних на сервер	

Продовження табл. 3

2.1	Відсутність обробки запиту та переданих параметрів до бази даних	SQL-ін'єкція у базу даних чи загрозливий код	Створення функцій для перевірки переданих параметрів та обробка всіх даних перед потраплянням у БД
3.1	Збереження даних без попереднього шифрування	Доступ до конфіденційних даних компанії чи користувачів	Шифрування даних в БД
4.1	Неправильна логіка розподілення ролей у застосунку	Доступ користувачів до заборонених даних чи налаштувань	Створити додаткові перевірки на авторизовність та роль користувача застосунку

При дотриманні зазначених безпекових вимог можна забезпечити стабільне використання застосунку користувачем.

1.6. Висновки до розділу

Після проведення аналізу предметної області було визначено, що зростання успіху компаній в сфері IT безпосередньо пов'язане з використанням сучасних технологій та постійним фаховим розвитком персоналу. Дослідження підтверджують, що організації з високою культурою навчання виходять на ринок з інноваціями швидше.

Для створення програмного застосунку було обрано методологію OKR тому, що використання таких застосунків дозволяє підвищити продуктивність розвитку компетенцій співробітників та забезпечити ефективне управління процесом підвищення кваліфікації в IT-компаніях.

Існуючі аналоги є більш направленими на управління проєктами, а не на фаховий розвиток співробітників та не є кросплатфомними. Тому такі

застосунки більше використовуються для облаштування планів розвитку самої компанії, та не є зручними для впровадження системи менторської підтримки.

Було визначено конкурентні переваги програмного застосунку. Розроблюваний програмний застосунок буде складатися з чотирьох компонент, таких як: користувацька, менторська, адміністраторська та серверна частина.

Також у розділі було визначено основні функціональні та нефункціональні вимоги до розроблюваного ПЗ та визначено вимоги до якості та безпеки програмного застосунку.

2. ОБҐРУНТУВАННЯ ВИБОРУ ЗАСОБІВ РЕАЛІЗАЦІЇ

2.1. Обґрунтування вибору мови програмування для мобільного та браузерного модулів

Існує велика кількість технологій для створення мобільного, браузерного та серверного модулів застосунку “Inner Growth”. Розглянемо можливі технології для кожного з цих модулів та виберемо найбільш вигідний стек для реалізації програмного застосунку “Inner Growth”.

Розглянемо мови програмування для створення мобільних застосунків. Згідно до зазначених вище функціональних та нефункціональних вимог, визначимо характеристики для вибору мови програмування. Мова програмування має підтримувати кросплатформність застосунку та мати розширюваність (доступні бібліотеки чи фреймворки).

Розглянемо мови програмування: *Kotlin*, *JavaScript*, *Dart*, *TypeScript*.

Kotlin

Kotlin – це сучасна статично типізована мова програмування, яка поєднує в собі об'єктно-орієнтовані та функціональні конструкції. Вона спрямована на різні платформи, включаючи JVM, і має повну сумісність з Java [2].

Мова Kotlin повністю сумісна з мовою Java, тому мова Kotlin може використовувати всі можливі розширення мови Java. Мова Kotlin повністю сумісна з ОС Android та може виконуватись у веб браузері.

Розглянемо переваги і недоліки Kotlin.

Переваги Kotlin:

- Лаконічний синтаксис та можливості мови Kotlin роблять розробку мобільних застосунків швидкою та ефективною. Це допомагає скоротити час на розробку та покращити якість коду.
- Функції безпеки нульового значення в Kotlin допомагають запобігти помилок за нульовим вказівником, які є поширеними при

розробці мобільних застосунків. Через цю функцію застосунок є більш стабільним та надійним.

Недоліки Kotlin:

- Спільнота Kotlin менша, ніж у більш поширених мов. Це може ускладнити пошук ресурсів та підтримки для вирішення конкретних проблем при розробці мобільних застосунків.
- Час компіляції Kotlin повільніший за час компіляції Java, особливо у великих проектах. Це сповільнює цикли розробки та тестування, що є недоліком для проектів, чутливих до часу.

JavaScript

JavaScript – це програмна мова, яка є динамічною, об'єктно-орієнтованою та прототипною. Вона є реалізацією стандарту ECMAScript і часто використовується для розробки скриптів на веб-сторінках. JavaScript надає змогу клієнтам (користувачам) взаємодіяти з веб-сторінками, управляти браузером, асинхронно обмінюватися даними з сервером, змінювати структуру та зовнішній вигляд сторінок [3].

JavaScript існує є універсальною мово, яка використовується для створення вебзастосунків, серверних застосунків і мобільних застосунків.

Переваги JavaScript:

- JavaScript – це універсальна мова, яку можна використовувати для розробки застосунків на різних платформах, таких як iOS, Android, Windows та веб браузерів.
- Існує велика спільнота розробників, які використовують JavaScript. Отже існує багато бібліотек і фреймворків з відкритим вихідним кодом, які можна використовувати для розробки застосунків.
- JavaScript має одні з найкращих фронтенд-фреймворків, таких як React Native та Ionic, які полегшують розробку сучасних мобільних застосунків зі складним користувацьким інтерфейсом.

Недоліки JavaScript:

- JavaScript може мати проблеми з продуктивністю на мобільних пристроях, особливо на старих або менш потужних. Це пов'язано з інтерпретованою природою мови та витратами на виконання коду у веб-перегляді.
- Мобільні застосунки, створені за допомогою JavaScript, за відсутності відповідних бібліотек, потребують підключення до Інтернету для належного функціонування.

Dart

Мова Dart має синтаксис, схожий до Java, і не вимагає явного визначення типів. Вона може бути використана для створення як серверних, так і клієнтських застосунків. Код, написаний мовою Dart, може бути запущений всередині браузера, перетворивши його на JavaScript, або виконаний безпосередньо за допомогою спеціального JavaScript-інтерпретатора. Мова Dart підходить для розробки як невеликих скриптів без жорсткої структури, так і великих модульних проектів, які підтримуються великим колективом. Для більшої явності типів, яка допомагає уникнути плутанини і помилок, можна використовувати більш жорстку типізацію у Dart [5].

Переваги Dart:

- Функція швидкого перезавантаження змін дозволяє розробникам бачити зміни в режимі реального часу, що робить процес розробки швидшим та ефективнішим.
- Dart можна використовувати для розробки кросплатформних мобільних застосунків.
- Система типізації Dart допомагає виявляти помилки на ранніх стадіях розробки, що призводить до створення більш надійних мобільних застосунків.

Недоліки Dart:

- Dart не така популярна, як інші мови, що використовуються для мобільної розробки, такі як Kotlin чи JavaScript, що призводить до браку доступних ресурсів та інструментів.
- Dart займає багато пам'яті, що може бути недоліком при розробці мобільних застосунків для пристроїв з обмеженими ресурсами.
- Dart все ще вважається відносно новою мовою програмування, тому вона не має настільки високого рівня стабільності, як більш усталені мови.

TypeScript

TypeScript є мовою програмування, яка зберігає зворотню сумісність з JavaScript і компілюється в неї. Вона відрізняється від JavaScript наявністю статичного визначення типів, підтримки повноцінних класів (аналогічних до тих, що зустрічаються у традиційних об'єктно-орієнтованих мовах) та підтримки модулів. Ці особливості сприяють покращенню швидкості розробки, полегшенню читанню, рефакторингу та повторному використанню коду. Вони також допомагають виявляти помилки на етапі розроблення та компіляції і можуть покращити продуктивність виконання програм [6].

Переваги TypeScript:

- Допомагає виявляти помилки під час компіляції, а не під час виконання, що призводить до кращої якості коду та меншої кількості помилок.
- Забезпечує кращу документацію коду завдяки чіткій системі типів. Це полегшує розуміння і підтримку коду, навіть при роботі зі складною логікою застосунку.
- Ідеально підходить для великомасштабних проектів, де підтримка та організація коду є критично важливими.
- Усі бібліотеки та фреймворки JavaScript підтримуються мовою TypeScript.

Недоліки TypeScript:

- Оскільки код TypeScript потрібно скомпілювати, перш ніж його можна буде запустити, це може додати накладні витрати на процес розробки.
- Хоча сувора система типізації TypeScript допомагає запобігти помилкам, вона також може бути занадто обмеженою для деяких випадків, які потребують динамічності коду.

Для розробки браузерного модулю існує лише дві найбільш поширені мови: JavaScript та TypeScript. Характеристики обох мов наведено вище.

Оскільки буде розроблено програмний застосунок з окремо браузерним та мобільним модулями, пріоритетом є використання однакової мови для обох модулів. Оскільки для цих модулів єдиним варіантом є або JavaScript, або TypeScript, було обрано мову TypeScript. JavaScript програє динамічністю типізації, що призводить до великої кількості помилок в процесі розробки та не робить код легким в масштабуванні. Мобільний модуль буде написано за допомогою фреймворку React Native, який дозволяє кроссплатформне програмування для ОС Android і IOS. Браузерний модуль буде реалізовано за допомогою фреймворку React, через його схожість коду з React Native, динамічність та можливість розширення застосунку.

2.2. Обґрунтування вибору мови програмування для сервера

Розглянемо мови програмування: *C#, Python, TypeScript.*

C#

C# є об'єктно-орієнтованою мовою програмування з безпечною системою типізації, призначеною для платформи .NET. Синтаксис C# схожий на C++ і Java. Мова C# використовує строгу статичну типізацію і має підтримку поліморфізму, перевантаження операторів, вказівників на функції-члени класів, атрибутів, подій, властивостей, винятків та коментарів у форматі XML [7].

C# багато запозичує від своїх попередників і заснована на кращих практиках їх використання. Однак, вона також відхиляє деякі моделі, які виявилися проблематичними при розробці програмних систем. Наприклад, на відміну від C++, C# не підтримує множинне успадкування класів [7].

Переваги C#:

- C# – це скомпільована мова, яка працює на фреймворку .NET, що забезпечує чудову продуктивність для серверних застосунків.
- Ця мова менш схильна до помилок, ніж інші динамічно типізовані мови.
- Має велику та активну спільноту розробників, які постійно діляться кодом та надають підтримку новим розробникам.

Недоліки C#:

- Хоча C# можна використовувати на багатьох платформах, вона в першу чергу призначена для розробки серверного програмного забезпечення на базі ОС Windows.
- C# вимагає встановлення на сервері фреймворку .NET, що призводить до додаткових накладних витрат при налаштуванні сервера.
- Є пропрієтарною мовою, а це означає, що є обмеження в гнучкості в плані кастомізації та інструментів розробки.

Python

Python є високорівневою мовою програмування зі строгою динамічною типізацією та інтерпретованим характером. Її об'єктно-орієнтована природа разом із використанням структур даних високого рівня, динамічної семантики та динамічного зв'язування роблять її привабливою для швидкої розробки програм та комбінування компонентів. Python також сприяє модульності та повторному використанню коду за допомогою підтримки модулів та пакетів модулів. Інтерпретатор Python та стандартні бібліотеки доступні на всіх основних платформах у вихідній та скомпільованій формі. Мова підтримує кілька парадигм програмування,

таких як об'єктно-орієнтована, процедурна, функціональна та аспектно-орієнтована [8].

Переваги Python:

- Python має величезну колекцію бібліотек та фреймворків, які можна використовувати для розробки на стороні сервера, а саме: Flask, Django та Pyramid. Ці бібліотеки допомагають швидко створювати та розгортати вебзастосунки.
- Python підходить для написання сценаріїв, таких як автоматизація повторюваних завдань, керування серверами та обробка даних.
- Існує багато бібліотек і фреймворків для Python з відкритим вихідним кодом, які можна використовувати для розробки.

Недоліки Python:

- Система керування пам'яттю в Python може призвести до витоків пам'яті, що може вплинути на продуктивність програми.
- Глобальне блокування інтерпретатора (GIL) у Python може обмежувати продуктивність багатопотокових застосунків.
- Python схильна до вразливостей у безпеці, якщо її не розробляти належним чином. При написанні серверного коду на Python важливо дотримуватися найкращих практик та рекомендацій з безпеки.

TypeScript (з використанням технології NodeJs)

Node.js – це платформа, призначена для виконання швидких та ефективних мережевих застосунків, які можна написати мовою програмування JavaScript. Вона забезпечує можливість розробки як серверних скриптів для обробки веб-запитів, так і клієнтських та серверних програм [9].

Розглянемо переваги і недоліки мови *TypeScript* з використанням технології NodeJs з точки зору розроблення серверу.

Переваги TypeScript з NodeJS:

- NodeJS відомий своєю масштабованістю, а TypeScript розвиває її, пропонуючи кращу організацію коду та модульність. Завдяки системі модулів TypeScript та можливості визначати явні інтерфейси та типи, можна створювати більш масштабовані та підтримувані серверні застосунки.
- NodeJS має широку екосистему модулів і бібліотек, що дозволяє легко знаходити готові рішення для різних завдань, пов'язаних із серверами. TypeScript повністю сумісний з JavaScript, а це означає, що можна використовувати всю екосистему NodeJS, користуючись перевагами статичної типізації, яку надає TypeScript.

Недоліки TypeScript з NodeJS:

- TypeScript потребує компіляції в JavaScript, перш ніж його можна буде запустити на NodeJS. Це додає крок збірки, і процес компіляції збільшує час розробки, особливо у великих проектах. Крім того, TypeScript може вимагати додаткового налаштування та встановлення інструментів.

Проаналізувавши мови програмування для створення серверу, було прийнято рішення розробляти серверну частину на мові програмування TypeScript з використанням технології NodeJS. Вибір пав на цю мову через можливість створення єдиної кодової бази клієнта та сервера, написаних на одній мові. Також технологія NodeJS надає можливості для розширення та швидкого масштабування через велику кількість бібліотек. Недоліком Python є вразливість мови до безпеки, що є дуже пріоритетною вимогою до надійності серверу БД. А недоліком C# є потреба в додаткових налаштуваннях для сервера та обмеженість у виборі ОС для розробки та сервера.

2.3. Обґрунтування вибору бази даних

Дані програмного застосунку треба зберігати у базі даних в серверному модулі. Розглянемо декілька варіантів СКБД для використання, а саме: *PostgreSQL*, *MongoDB* та *SQLite*.

PostgreSQL

PostgreSQL є системою управління базами даних, яка поєднує об'єктно-орієнтовані та реляційні концепції. Вона надає широку підтримку стандарту SQL та пропонує велику кількість сучасних можливостей. PostgreSQL дозволяє виконувати складні запити, використовувати зовнішні ключі, використовувати тригери, створювати оновлювані представлення та забезпечувати транзакційну цілісність. Крім того, вона має вбудовану підтримку багатоверсійного паралелізму, що дозволяє працювати з даними в одночасно виконуваних транзакціях [10].

Переваги PostgreSQL:

- СКБД відома своєю стабільністю та надійністю, навіть при високих навантаженнях на сервер та інтенсивному трафіку даних. Вона має перевірений досвід роботи з великими та складними базами даних без шкоди для продуктивності.
- Пропонує широкий спектр розширених можливостей, а саме: підтримка складних запитів, повнотекстовий пошук, типи даних JSON і XML, геопросторові дані та цілісність транзакцій. Також підтримує різні методи індексування.
- PostgreSQL може обробляти великі обсяги даних і горизонтально масштабуватися, розподіляючи дані між декількома серверами за допомогою вбудованих функцій реплікації та кластеризації.

Недоліки PostgreSQL:

- PostgreSQL може споживати значний обсяг пам'яті при роботі з великими наборами даних або складними запитами. Це може вимагати додаткового налаштування для оптимізації продуктивності і забезпечення ефективного використання пам'яті.

MongoDB

MongoDB є відкритою документо-орієнтованою системою керування базами даних (СКБД), яка не вимагає строго опису схеми таблиць. Вона займає проміжне положення між швидкими і масштабованими системами, які працюють з даними у форматі ключ-значення, і реляційними СКБД. MongoDB пропонує функціональність, яка спрощує формування запитів і забезпечує зручну роботу з даними [11].

Переваги MongoDB:

- MongoDB використовує гнучку модель даних документів, що дозволяє зберігати дані в JSON-подібних документах без попередньо визначеної схеми. Така гнучкість полегшує роботу зі структурами даних.
- Пропонує вбудовані механізми реплікації та автоматичний обхід збоїв, забезпечуючи високу доступність.
- Має підтримку складних запитів, агрегацій та індексування. Вона також підтримує геопросторові запити та повнотекстовий пошук, що дозволяє створювати розширені пошукові функції.

Недоліки MongoDB:

- MongoDB може споживати значні ресурси пам'яті, особливо при роботі з великими наборами даних або складними запитами.
- Продуктивність і масштабованість MongoDB залежать від апаратних ресурсів, тому великі розгортання часто вимагають ретельної підготовки та конфігурації апаратного забезпечення.

SQLite

SQLite – це реляційна система керування базами даних, яка має свої особливості. Однією з них є те, що SQLite не використовує клієнт-серверну модель. У випадку SQLite, рушій бази даних не є окремим процесом, з яким взаємодіє програма. Замість цього, SQLite надає бібліотеку, яка компілюється разом з програмою, і сам рушій стає частиною програми. Цей підхід має кілька переваг. Він зменшує накладні витрати, оскільки немає

потреби у встановленні та налаштуванні окремого сервера. Також час відгуку програми може бути скорочений, оскільки взаємодія з базою даних відбувається безпосередньо через виклики функцій (API) бібліотеки SQLite. Крім того, цей підхід спрощує саму програму, оскільки всі дані, включаючи структуру бази даних, таблиці, індекси та самі дані, зберігаються в одному стандартному файлі на комп'ютері, на якому виконується програма [12].

Переваги SQLite:

- Потребує мінімального налаштування та адміністрування.
- Бази даних SQLite зберігаються в одному файлі, що робить їх дуже портативними.
- SQLite не потребує окремого серверного процесу для запуску. Він працює безпосередньо на стороні клієнта, що усуває необхідність у складних конфігураціях клієнт-сервер.

Недоліки SQLite:

- Вона не підтримує мережевий доступ і не має вбудованих механізмів для управління користувачами, автентифікації або контролю доступу.
- Оскільки SQLite є файловою базою даних, вона не має архітектури клієнт-сервер. Це означає, що вона не може обробляти одночасні з'єднання від декількох клієнтів, що обмежує її використання.
- SQLite має обмеження на розмір окремих баз даних і таблиць.

Проаналізувавши варіанти наведених вище баз даних, було прийнято рішення обрати СКБД PostgreSQL, тому що вона є стабільною та надійною, витримує великі навантаження на сервер та має широкий спектр можливостей.

2.4. Висновки до розділу

Для реалізації браузерного та мобільного модулів було проаналізовано мови Kotlin, JavaScript, Dart, TypeScript. Недоліком Kotlin виявилась сумісність лише з ОС Android, що не дозволяє створити

кросплатформний застосунок. Основним недоліком Dart є те, що вона не має високого рівня стабільності. Тому серед проаналізованих мов було обрано мову TypeScript, оскільки вона має строгу типізацію, на відміну від JavaScript, та легка в масштабуванні. В якості фреймворку для реалізації мобільного модулю було обрано React Native, а для браузерного – React.

Для реалізації серверного модулю було проаналізовано мови: C#, Python, TypeScript. Недоліком C# є необхідність встановлення на сервері фреймворку .NET та база ОС Windows. Python обмежує продуктивність багатопотокових застосунків. Отже, серед проаналізованих мов було обрано TypeScript з використанням технології NodeJS, з метою сворення єдиної кодової бази між усіма модулями. Також NodeJS надає можливості для розширення та швидкого масштабування.

Серед СУБД було проаналізовано PostgreSQL, MongoDB та SQLite. MongoDB споживає значні ресурси пам'яті при роботі зі складними запитами, що є основним недоліком. SQLite не може обробляти одночасні з'єднання від декількох клієнтів, що категорично недопустимо в розроблюваному ПЗ. Тому, проаналізувавши бази даних, було прийнято рішення обрати СКБД PostgreSQL. Вона є стабільною, надійною та може витримувати великі навантаження на сервер.

3. РОЗРОБЛЕННЯ ЗАСТОСУНКУ «INNER GROWTH»

3.1. Загальна структура застосунку

Перед початком розробки було сформовано та проаналізовано функціональні та нефункціональні вимоги до застосунку «Inner Growth» та визначені пріоритети їх реалізації (табл. 4).

Таблиця 4

Функціональні вимоги до програмного застосунку

Номер	Назва	Пріоритет
1	Реєстрація в застосунку	1
2	Авторизація в застосунку	1
3	Створення користувача адміністратором	1
4	Налаштування особистого профілю	3
5	Перегляд можливостей для розвитку	4
6	Подання заявки на участь в події можливості для розвитку	4
7	Налаштування застосунку	5
8	Перегляд поточного рівня на мапі кар’єрного розвитку	3
9	Вибір іншого рівня на мапі кар’єрного розвитку	3
10	Створення циклу OKR	2
11	Завершення циклу OKR	2
12	Звітування про виконані завдання у циклі OKR	2
13	Створення сторінки компанії адміністратором	1
14	Видалення акаунтів користувачів адміністратором	4

Продовження табл. 4

15	Зміна ролі користувача адміністратором	3
16	Створення рівня компетенції адміністратором	2
17	Поєднання рівнів компетенції у мапу кар’єрного розвитку адміністратором	2
18	Перегляд списку користувачів ментором	3
19	Підтвердження набутих компетенцій користувача ментором	3

Відповідно до вимог в застосунку передбачається три ролі:

- співробітник;
- ментор;
- адміністратор.

Для кожної ролі доступні конкретні функціональні можливості застосунку, які наведемо на рис. 1.

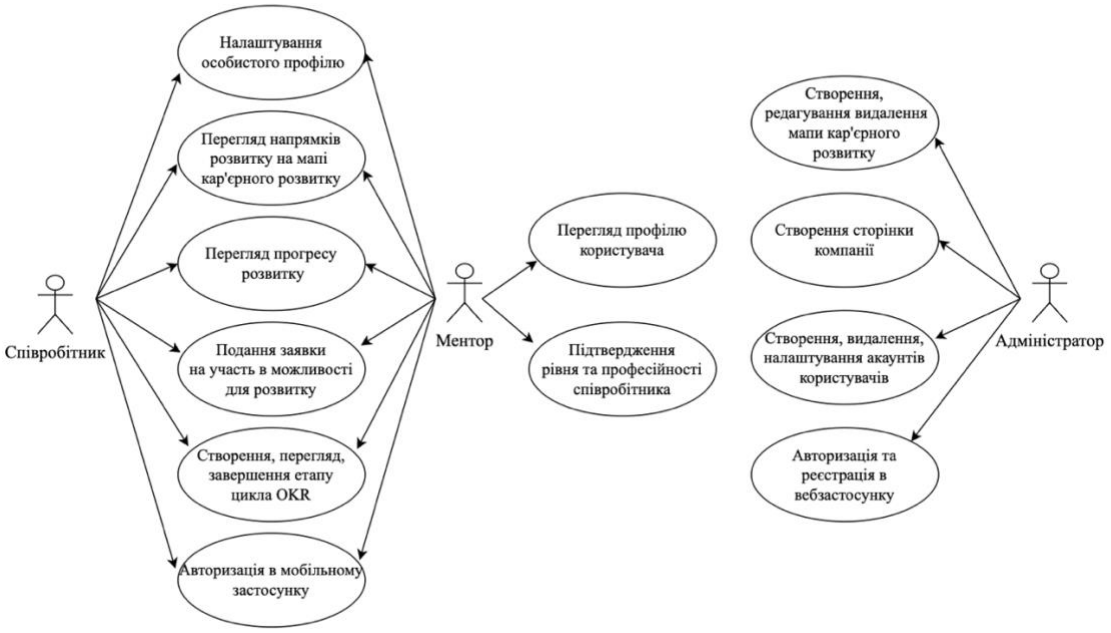


Рис. 1. Діаграма варіантів використання програмного застосунку

Неавторизований користувач має доступ лише до екрану авторизації, на якому йому необхідно зайти вже в наявний акаунт, створений адміністратором. Неавторизований адміністратор має доступ до сторінки реєстрації / авторизації в браузері. Адміністратор може зареєструвати акаунт за допомогою електронної пошти і паролю та авторизуватись в браузерному модулі застосунку. Після авторизації співробітник переходить на головний екран застосунку (рис. 2).

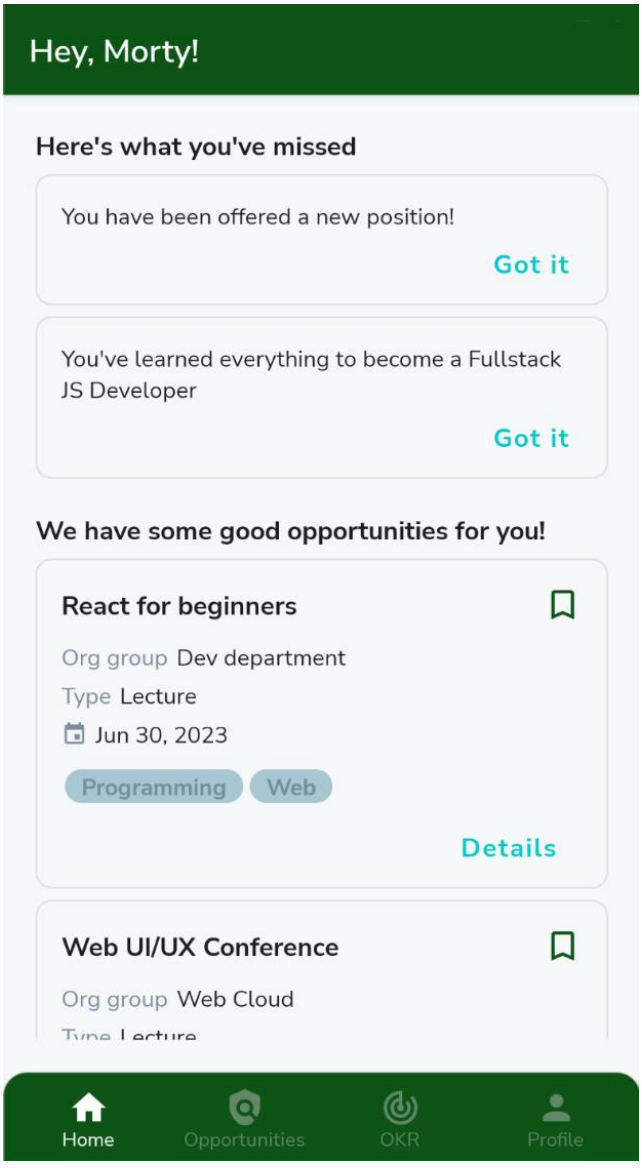


Рис. 2. Екран Home для ролі Користувач

Загалом у навігаційному меню мобільного застосунку користувачу доступні 5 сторінок:

- Home – екран на якому відображаються ключові найближчі події (завершення циклу OKR, повідомлення тощо) та рекомендовані можливості розвитку.
- Explore – екран на якому відображаються усі можливі шляхи розвитку для співробітника (мапа кар'єрного розвитку) та прогрес на поточному етапі.
- Opportunities – екран зі списком всіх можливостей для розвитку доступних в компанії. Користувач може переглядати детальну інформацію про подію та подати заявку на участь у ній.
- OKR – екран з усіма створеними OKR користувача. Користувач може відслідковувати прогрес поточного циклу OKR, відмічати виконані завдання та прогрес досягнення цілі. Також користувач може переглядати завершені цикли та створювати нові.
- Profile – екран особистого кабінету користувача. Користувач може налаштовувати застосунок, змінювати фото профіля та інформацію про себе, а саме: досвід роботи та навчання, вміння по фаховим компетенціям, поточний рівень на мапі кар'єрного розвитку тощо.

На рис. 3 зображено алгоритм роботи співробітника з циклами OKR.

Описано всі ймовірні варіанти роботи з циклом, такі як: створення циклу, цілі, ключових результатів, або перегляд та зміна показників ключових результатів.

Коли користувач переходить на екран всіх доступних циклів OKR, то він чи вона може переглянути всі поточні чи завершені цикли. Якщо поточного циклу не існує, то користувач потрапляє на екран створення цілі для циклу. Якщо поточний цикл існує, користувач може перейти на сторінку детальної інформації про нього. Якщо всі ключові результати виконано, то користувач може завершити цикл, або ввести інформацію про зміни в показниках.

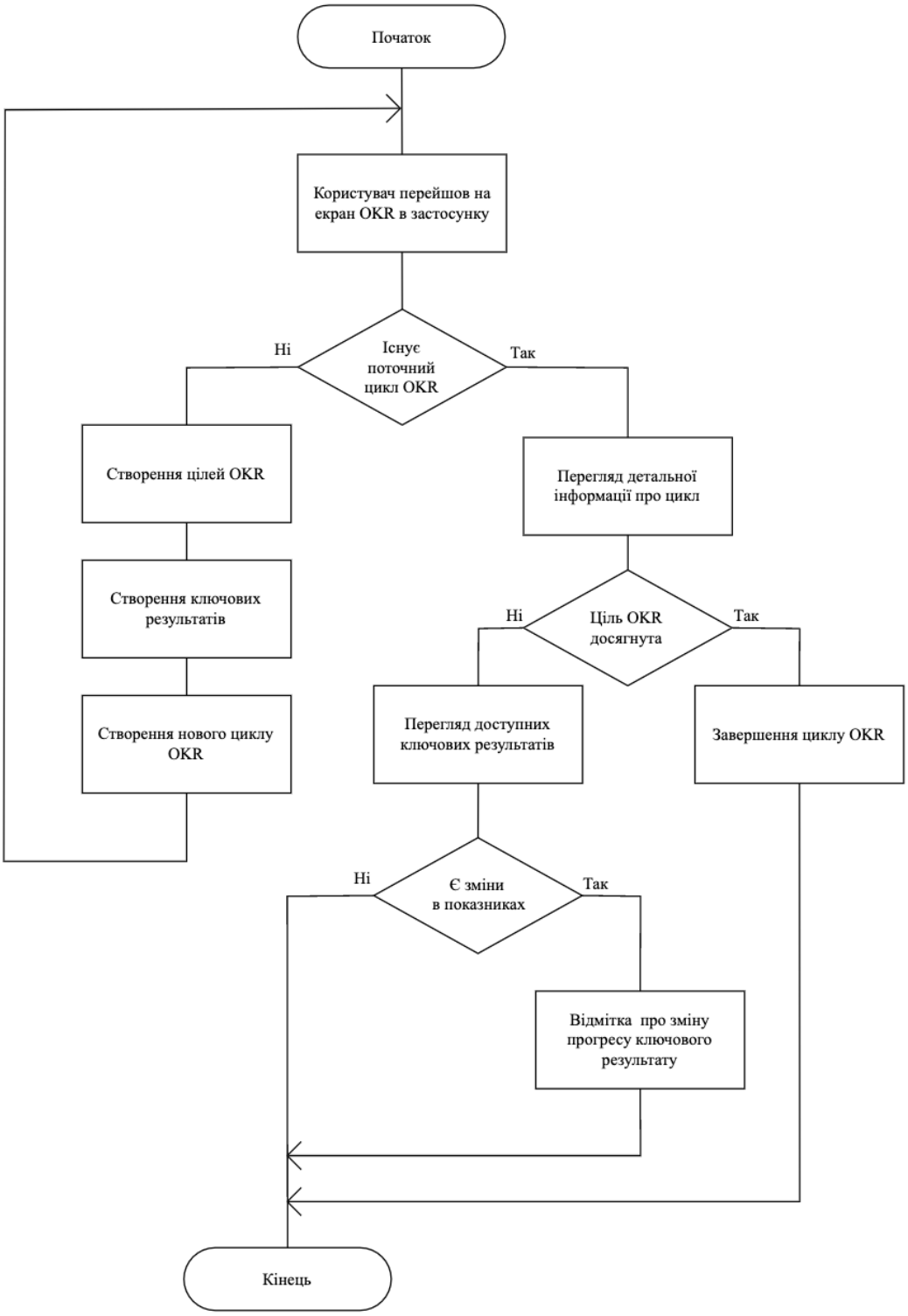


Рис. 3. Алгоритм роботи співробітника з OKR

Також новий користувач має доступ до екрану Onboarding. На цьому користувач має завершити налаштування свого акаунту, перед тим як отримати повний доступ до застосунку, тобто створити пароль, заповнити інформацію для особистого профіля.

У адміністратора в браузерному модулі застосунку доступна 1 сторінка з 4 секціями (рис. 4):

- Your company – на цій сторінці адміністратор може налаштовувати інформацію про компанію, змінювати логотип, назву та детальну інформацію про неї.
- Users – на цій сторінці адміністратор додає, редагує чи видаляє користувачів з системи. Також він може визначати та змінювати ролі користувачів.
- Career Path – ця сторінка необхідна для створення рівнів фахових компетенцій. Адміністратор створює, редагує чи видаляє рівні. Для створення рівня треба обрати, яким компетенціям має відповідати співробітник та визначити його назву.
- Connect Career Path – ця сторінка необхідна для створення мапи кар'єрного розвитку, тобто визначення розгалужень для всіх створених рівнів.

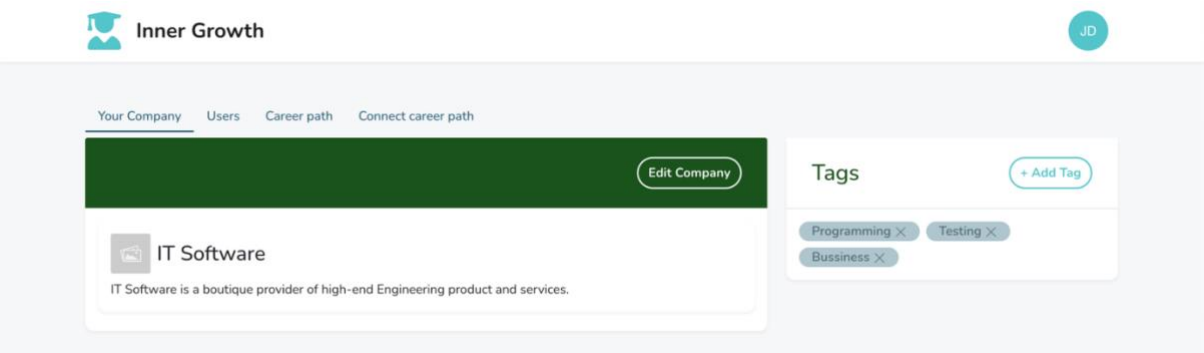


Рис. 4. Сторінка Адміністратора

Для менторів доступні всі екрані, які доступні співробітнику, а також ще 2 додаткових екрани:

- Users – екран з переліком усіх користувачів яких наставляє ментор, на ньому відображається найголовніша інформація про користувача (ПІБ, поточний рівень тощо).

- Екран детальної інформації про користувача. На ньому ментор може переглядати всю інформацію, яка доступна в профілі користувача, а також надавати підтвердження набутим знанням чи підвищення просування по мапі кар’єрного розвитку.

3.2. Архітектура застосунку «Inner Growth»

Програмний застосунок складається з двох клієнтських модулів, а саме: мобільний та браузерний. Браузерний модуль є вебзастосуноком та доступен для мобільних версій браузерів. Мобільний модуль є окремим застосунком доступен для ОС Android та IOS.

Мобільний та браузерний модулі взаємодіють зі сервером за допомогою REST API, який забезпечує стандартизований спосіб обміну даними між клієнтськими модулями і серверним модулем [13].

Загальний огляд того, як клієнтські модулі взаємодіють з сервером через REST (рис. 5):

- Клієнтський модуль надсилає запит на сервер за допомогою певного методу HTTP. Цей запит містить URL-адресу, яка ідентифікує ресурс, до якого застосунок намагається отримати доступ, і будь-які додаткові дані, необхідні для виконання запиту [14].
- Серверний модуль отримує запит і обробляє його відповідно до конкретного REST API. Сервер виконує необхідні запити до бази даних, обчислення тощо для формування відповіді.
- Сервер надсилає відповідь клієнтському модулю у вигляді HTTP-коду статусу (200 OK, 404 Not Found тощо) разом з відповідними даними в форматі JSON.

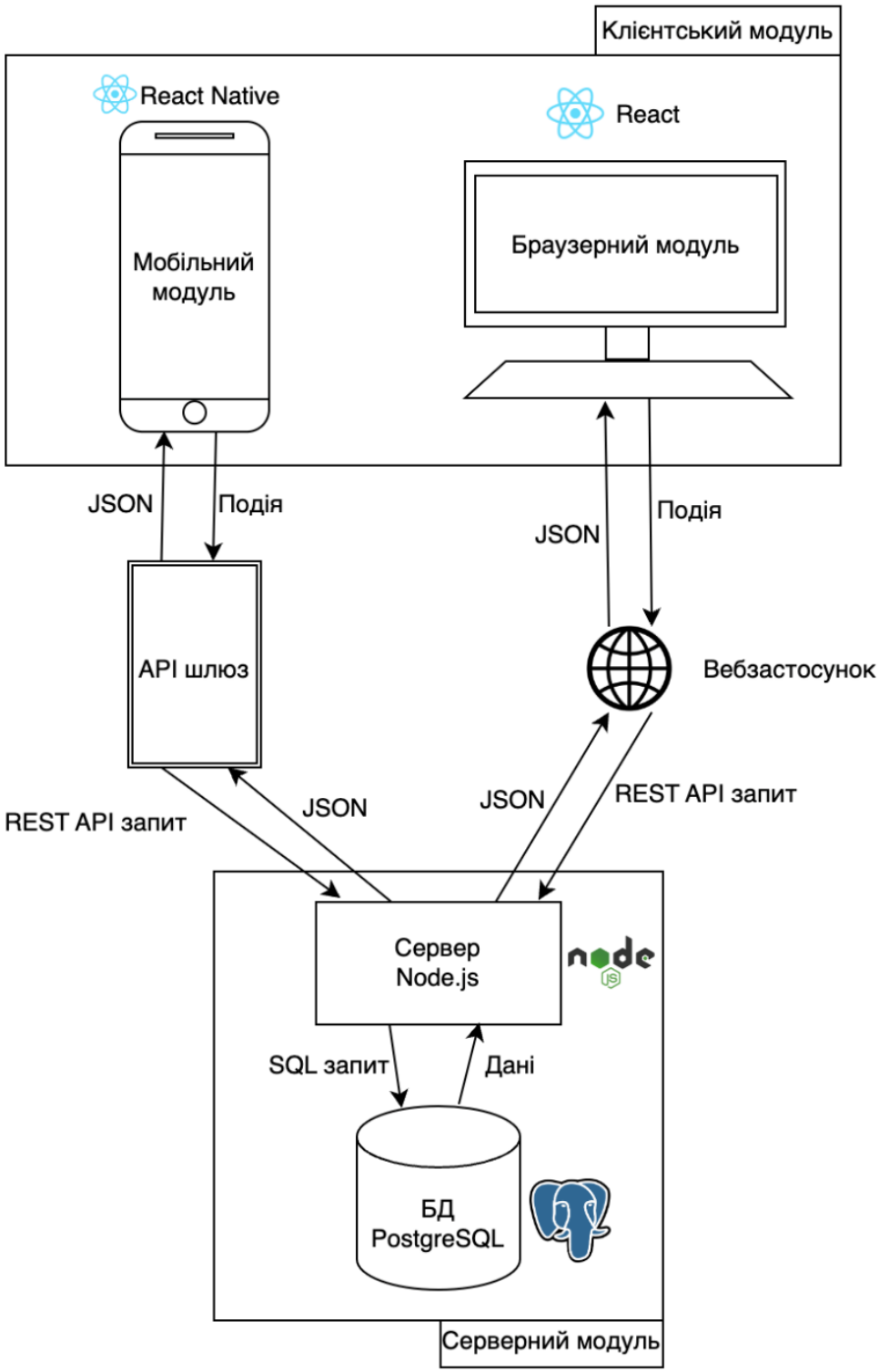


Рис. 5. Архітектура програмного застосунку

Клієнт (мобільний та браузерний модулі) містять в собі:

- Інтерфейс користувача.
- Клієнтську логіку застосунку.
- Валідаційні схеми для полів вводу даних.

Серверний модуль включає в себе:

- Бізнес логіку.
- Ідентифікацію користувача.
- Зберігання та надання доступу до даних.

Модель бази даних.

База даних для збереження та отримання інформації налічує 17 таблиць (рис. 6), а саме:

1. Company – таблиця інформації про компанію.
2. Tags – таблиця характеристик компанії.
3. Skill – таблиця навичок фахових компетенцій.
4. Domain – таблиця фахових компетенцій.
5. CareerPath – таблиця зв’язку фахових компетенцій.
6. DomainLevel – таблиця рівнів фахових компетенцій.
7. RefreshToken – таблиця токенів авторизації.
8. UserRole – таблиця ролі користувача.
9. UserSkill – таблиця навичок користувача.
10. OKR – таблиця циклу OKR.
11. Objective – таблиця цілей циклу OKR.
12. KeyResult – таблиця завдань циклу OKR.
13. User – таблиця користувачів.
14. UserSkillCategory – таблиця відношення навичок та користувачів.
15. CareerJourney – таблиця інформації про досвід користувачів.
16. SkillCategory – таблиця навичок відносно фахових компетенцій користувачів.
17. SkillObjective – таблиця відношення цілей та навичок фахових компетенцій.

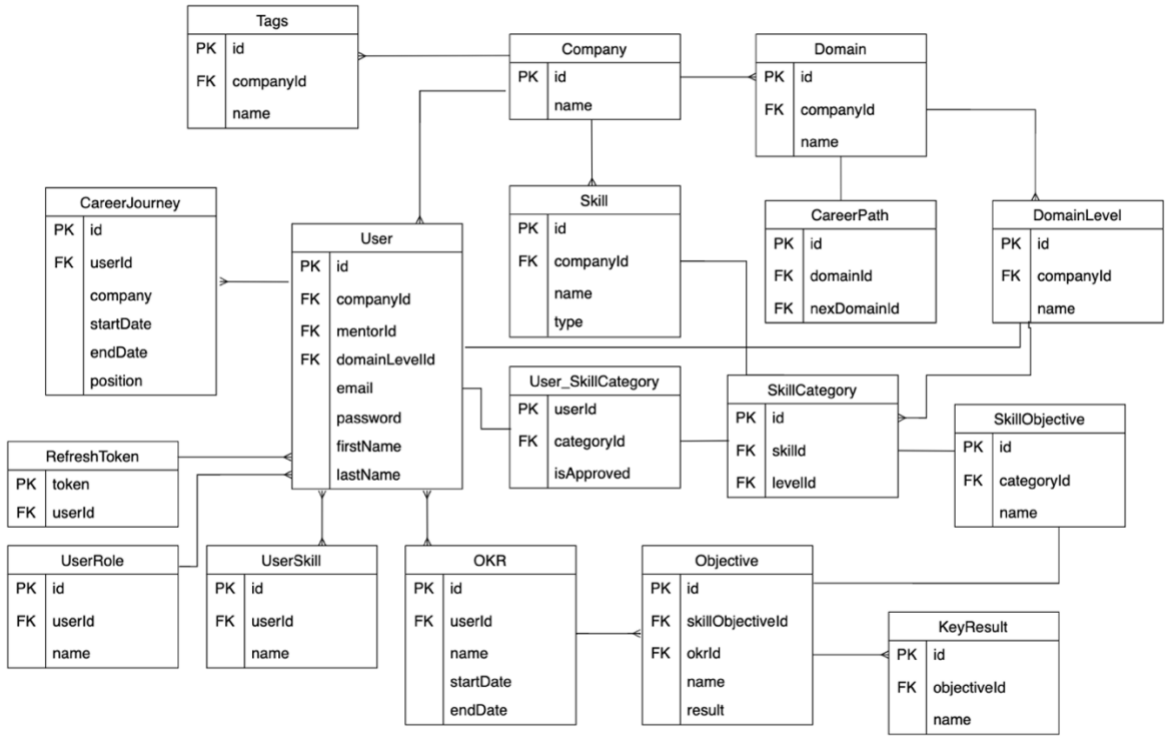


Рис. 6. Схема бази даних

Нижче наведено детальний опис всіх полів таблиць бази даних (табл. 5 – 21).

Таблиця 5

Поля таблиці Company

Назва поля	*Опис*	*Тип даних*
id	Унікальний ідентифікатор	Uuid, PK
name	Назва компанії	String

Таблиця 6

Поля таблиці Tags

Назва поля	*Опис*	*Тип даних*
id	Унікальний ідентифікатор	Uuid, PK

Продовження табл. 6

name	Назва тегу	String
companyId	Ідентифікатор компанії	Uuid, FK

Таблиця 7

Поля таблиці Skill

Назва поля	*Опис*	*Тип даних*
id	Унікальний ідентифікатор	Uuid, PK
name	Назва вміння	String
type	Вид вміння (Hard skill, soft skill, мова)	Enum
companyId	Ідентифікатор компанії	Uuid, FK

Таблиця 8

Поля таблиці Domain

Назва поля	*Опис*	*Тип даних*
id	Унікальний ідентифікатор	Uuid, PK
name	Назва фахової компетенції	String
companyId	Ідентифікатор компанії	Uuid, FK

Таблиця 9

Поля таблиці CareerPath

Назва поля	*Опис*	*Тип даних*
id	Унікальний ідентифікатор	Uuid, PK
domainId	Ідентифікатор фахової компетенції	Uuid, FK
nextDomainId	Ідентифікатор фахової компетенції доступної для просування по мапі кар’єрного розвитку	Uuid, FK

Таблиця 10

Поля таблиці DomainLevel

Назва поля	*Опис*	*Тип даних*
id	Унікальний ідентифікатор	Uuid, PK
name	Назва рівня фахової компетенції	String
nextLevel	Ідентифікатор наступного рівня фахової компетенції	Uuid, FK
prevLevel	Ідентифікатор минулого рівня фахової компетенції	Uuid, FK
domainId	Ідентифікатор фахової компетенції	Uuid, FK

Таблиця 11

Поля таблиці RefreshToken

Назва поля	*Опис*	*Тип даних*
token	Токен авторизації користувача	String
userId	Ідентифікатор користувача	Uuid, FK

Таблиця 12

Поля таблиці UserRole

Назва поля	*Опис*	*Тип даних*
id	Унікальний ідентифікатор	Uuid, PK
role	Роль користувача (Admin, Mentor, Worker)	Enum
userId	Ідентифікатор користувача	Uuid, FK

Таблиця 13

Поля таблиці UserSkill

Назва поля	*Опис*	*Тип даних*
id	Унікальний ідентифікатор	Uuid, PK
name	Назва вміння користувача	String
userId	Ідентифікатор користувача	Uuid, FK

Таблиця 14

Поля таблиці UserSkill

Назва поля	*Опис*	*Тип даних*
id	Унікальний ідентифікатор	Uuid, PK
name	Назва циклу OKR	String
userId	Ідентифікатор користувача	Uuid, FK
endDate	Дата кінця циклу OKR	Timestamp
startDate	Дата початку циклу OKR	Timestamp

Таблиця 15

Поля таблиці Objective

Назва поля	*Опис*	*Тип даних*
id	Унікальний ідентифікатор	Uuid, PK
name	Назва цілі OKR	String
orkId	Ідентифікатор циклу OKR	Uuid, FK
skillObjectiveId	Ідентифікатор відношення цілі та навички фахової компетенції	Uuid, FK
result	Результат циклу OKR	Integer

Таблиця 16

Поля таблиці KeyResult

Назва поля	*Опис*	*Тип даних*
id	Унікальний ідентифікатор	Uuid, PK
name	Назва завдання OKR	String
objectiveId	Ідентифікатор цілі OKR	Uuid, FK

Таблиця 17

Поля таблиці User

Назва поля	*Опис*	*Тип даних*
id	Унікальний ідентифікатор	Uuid, PK
email	Електронна пошта користувача	String
password	Зашифрований пароль користувача	String
firstname	Ім’я користувача	String
lastname	Прізвище користувача	String
companyId	Ідентифікатор компанії	Uuid, FK
mentorId	Ідентифікатор ментора користувача	Uuid, FK
domainLevelId	Ідентифікатор фахової компетенції користувача	Uuid, FK

Таблиця 18

Поля таблиці UserSkillCategory

Назва поля	*Опис*	*Тип даних*
userId	Ідентифікатор користувача	Uuid, PK
skillLevelId	Ідентифікатор навички	Uuid, FK
isApproved	Підтвердження навички ментором	Boolean

Таблиця 19

Поля таблиці CareerJourney

Назва поля	*Опис*	*Тип даних*
id	Унікальний ідентифікатор	Uuid, PK
userId	Ідентифікатор користувача	Uuid, FK
endDate	Дата кінця роботи	Timestamp
startDate	Дата початку роботи	Timestamp
company	Назва компанії	String
position	Назва посади	String

Таблиця 20

Поля таблиці SkillCategory

Назва поля	*Опис*	*Тип даних*
id	Унікальний ідентифікатор	Uuid, PK
levelId	Ідентифікатор рівня фахової компетенції	Uuid, FK
skillId	Ідентифікатор навички	Uuid, FK

Таблиця 21

Поля таблиці SkillObjective

Назва поля	*Опис*	*Тип даних*
id	Унікальний ідентифікатор	Uuid, PK
name	Назва цілі	String
categoryId	Ідентифікатор категорії навички	Uuid, FK

3.3. Особливості реалізації

Мобільний модуль програмного застосунку був реалізований за допомогою React Native. React Native – фреймворк інтерфейсу

користувача з відкритим кодом. Компоненти React виконуються над існуючим нативним кодом і взаємодіють із нативними API [15]. На рис. 7 наведено мобільний модуль, який реалізовано за патерном проєктування Model-View-ViewModel (MVVM).

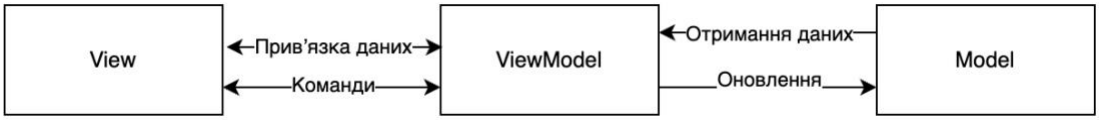


Рис. 7. Патерн проєктування MVVM

Фреймворк React Native виконує роль VVM для репрезентації даних, опис моделей наведено у TypeScript класах у лістингах 1.1 – 1.9.

Лістинг 1.1. Клас користувач

```
interface IUser {
  id: string;
  firstName: string;
  lastName: string;
  position: string;
  email: string;
  avatar?: string;
  isCompleteTest: boolean;
  role: UserRoleType;
  level: IUserLevel;
}
```

Лістинг 1.2. Клас OKR

```
interface IOkr {
  id: string;
  name: string;
  type: string;
  startDate: string;
  endDate: string;
  objectives: IObjective[];
}
```

Лістинг 1.3. Клас рівня фахової компетенції користувача

```
interface IUserLevel {
  domain: IUserDomain;
  name: string;
}
```

Лістинг 1.4. Клас ціль OKR

```
interface IObjective {
  id: string;
  name: string;
  result: number;
  keyResults: IKeyResult[];
}
```

Лістинг 1.5. Клас ключові результати OKR

```
interface IKeyResult {
  id: string;
  name: string;
}
```

Лістинг 1.6. Клас можливості для розвитку

```
interface IOpportunity {
  id: string;
  name: string;
  tags: ITag[];
  type: OpportunityType;
  organization: string;
  startDate: string;
}
```

Лістинг 1.7. Клас тег

```
interface ITag {
  id: string;
  name: string;
}
```

Лістинг 1.8. Клас освіта

```
interface IEducation {
  id?: string;
  specialization: string;
  university: string;
  degree: string;
  startDate: string;
  endDate?: string;
}
```

Лістинг 1.9. Клас досвід роботи

```
interface ICareer {
  id: string;
  position: string;
  company: string;
  startDate: string;
  endDate?: string;
}
```

Для зберігання даних в застосунку мобільного модулю та доступності цих даних під час виконання було використано бібліотеку Redux. Redux – це бібліотека, яка використовується для зберігання даних та забезпечує доступ до цих даних у будь-якій частині застосунку. Вона дозволяє зберігати необхідні дані під час роботи застосунку. Може містити дані з внутрішніх або зовнішніх джерел, дані про навігацію, інформацію про користувача тощо [16].

Код програми мобільного модулю розбитий в таких підмодулях:

- Common – підмодуль, що зберігає всі моделі які пов’язані з даними чи моделі, які використовуються для роботи застосунку.
- Components – підмодуль, що зберігає усі візуальні компоненти, які перевикористовуються в застосунку (кнопки, текстові поля, поля вводу тощо).
- Helpers – підмодуль який містить спеціальні функції які відповідають за логіку застосунку.
- Navigation – підмодуль з логікою переходів між різними екранами, передавання даних між ними та перевіряє доступність до екранів по ролям.
- Screens – підмодуль, що зберігає візуальні компоненти всіх екранів в застосунку.
- Services – підмодуль, що містить логіку запитів і отримання відповідей з серверу.
- Store – підмодуль, що містить логіку зберігання данних всереднині застосунку.
- Validation Schemas – підмодуль, який містить логіку перевірки введених користувачем даних.

Для розробки браузерного модуля програмного застосунку було використано фреймворк React. Для зберігання даних було використано бібліотеку Redux, зазначену вище.

Код програми браузерного модуля розбитий в таких підмодулях (рис. 8):

- Common – підмодуль, що зберігає всі моделі які пов’язані з даними чи моделі, які використовуються для роботи застосунку.
- Components – підмодуль, який зберігає компоненти для різних сторінок.
- Pages – підмодуль, що зберігає розмітку сторінок.
- Services – підмодуль, що містить логіку запитів і отримання відповідей з серверу.
- Navigation – підмодуль, що відповідає за перехід між сторінками та перевіряє доступність до сторінок по ролям.
- Store – підмодуль, що містить логіку зберігання данних всереднині застосунку.
- Validation Schemas – підмодуль, який містить логіку перевірки введених користувачем даних.

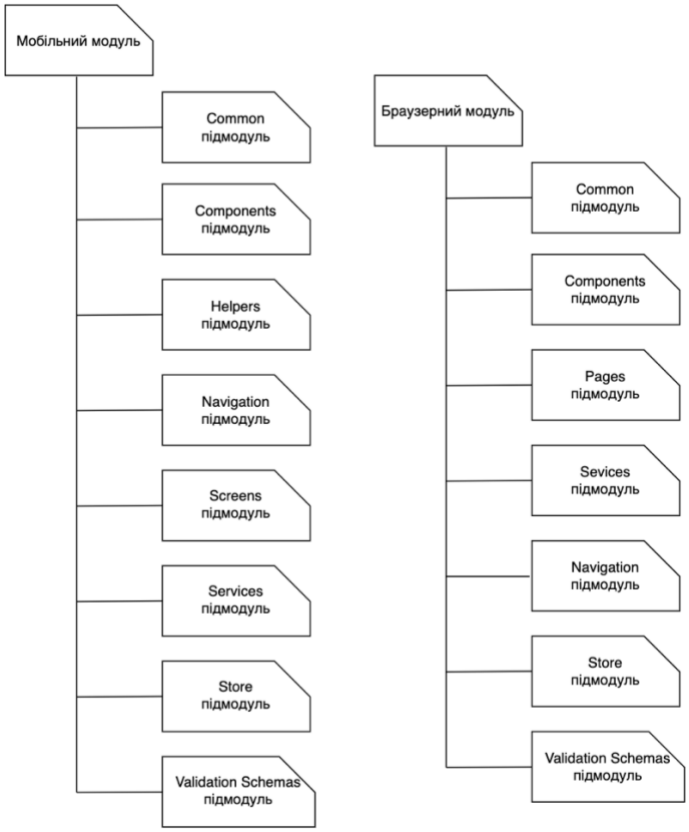


Рис. 8. Структура клієнтського модуля

3.4. Висновки до розділу

У цьому розділі наведено загальну структуру програмного застосунку. Було визначено пріоритети реалізації функціональних вимог від необхідних до непріоритетних. Відповідно до вимог було визначено три ролі користувачів в застосунку, а саме: співробітник, ментор і адміністратор.

Описано доступ до функціональності програмного застосунку в залежності від ролей користувача. Для мобільного та браузерного модулей було описано функціональність кожної сторінки чи екрану, їх призначення. Наведено діаграму варіантів використання для кожної ролі в застосунку та діаграму діяльності основного сценарію використання для користувача з роллю співробітник.

Описано архітектуру програмного застосунку. Клієнтський модуль програмного застосунку складається з браузерного і мобільного модулів та є окремими застосунками. Клієнтський модуль взаємодіє з серверним модулем за допомогою REST API. Представлено всі таблиці баз даних доступні в програмному застосунку та описано їх поля та типи.

Для реалізації браузерного модулю було використано фреймворк React, а для мобільного – React Native. Для реалізації тимчасового сховища даних під час роботи програмного застосунку було використано бібліотеку Redux. Також було описано підмодулі коду програми для клієнтського модуля та наведено їх схему.

4. АНАЛІЗ РОЗРОБЛЕНОГО ЗАСТОСУНКУ

4.1. Тестування програмного застосунку

Тестування є одним з етапів розробки програмного застосунку, який передбачає перевірку відповідності розробленого ПЗ вимогам, коректність та стабільність роботи застосунку, пошук помилок під час розробки та перевірка сумісності.

Тестування програмного застосунку відбувалось протягом усього часу розробки. Це значно знизило кількість потенційних помилок в бізнес логіці та стабільності застосунку.

В рамках дипломного проектування було проведено функціональне тестування програмного застосунку. У цьому процесі тестувальник виступає в ролі кінцевого користувача і перевіряє всі функції програми, щоб переконатися, що програма працює належним чином [19].

Було перевірено функціональні вимоги відповідно до сценаріїв використання наведених нижче (табл. 22 – 26).

Таблиця 22

Сценарій використання: Реєстрація адміністратора

Виконавець	Адміністратор
Попередні умови	–
Послідовність дій	1. Ввести email. 2. Ввести ім'я. 3. Ввести прізвище. 4. Ввести пароль. 5. Натиснути на кнопку Register.
Очікуваний результат	1. Email відображається в полі вводу. 2. Ім'я відображається в полі вводу. 3. Прізвище відображається в полі вводу.

Продовження табл. 22

	4. Пароль відображається в полі вводу. 5. Повідомлення про успішну реєстрацію та перенаправлення на головну сторінку адміністратора.

При невідповідності послідовності дій користувача має з’явитись повідомлення по помилку від відповідним полем вводу (рис. 9).

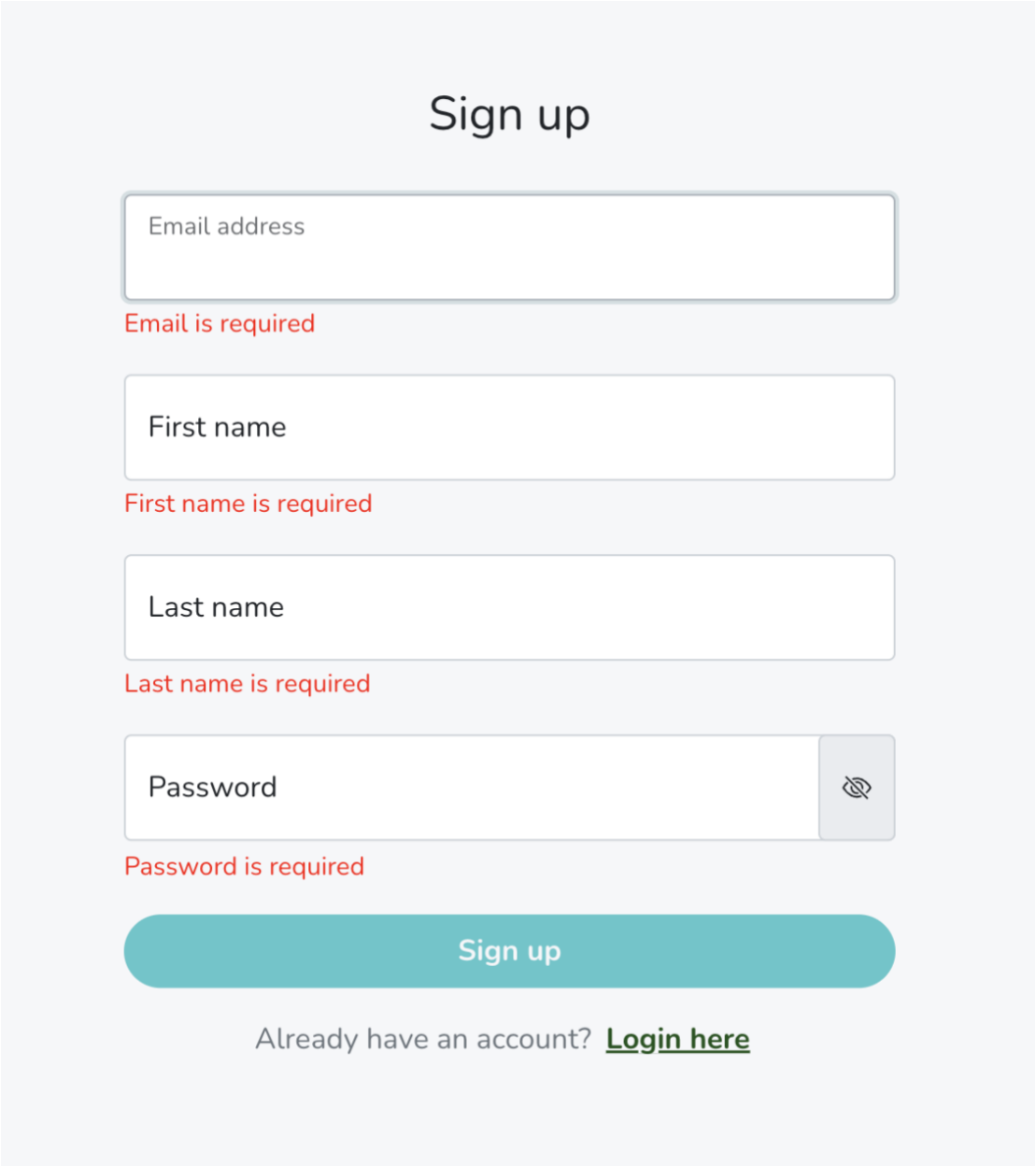


Рис. 9. Результат тестування реєстрації адміністратора

Таблиця 23

Сценарій використання: Додавання сторінки компанії адміністратором

Виконавець	Адміністратор
Попередні умови	1. Адміністратор авторизований в системі.
Послідовність дій	1. Натиснути на кнопку Add company. 2. Ввести назву компанії. 3. Ввести опис компанії. 4. Натиснути на кнопку Save у модальному вікні.
Очікуваний результат	1. Відкривається модальне вікно Add company info. 2. Компанія відображається в полі Company. 3. У полі Description відображається опис. 4. Компанія створена і з'являється в полі компанії.

При невідповідності послідовності дій користувача має з’явитись повідомлення по помилку від відповідним полем вводу. Нижче наведено приклад доданої компанії (рис. 10).

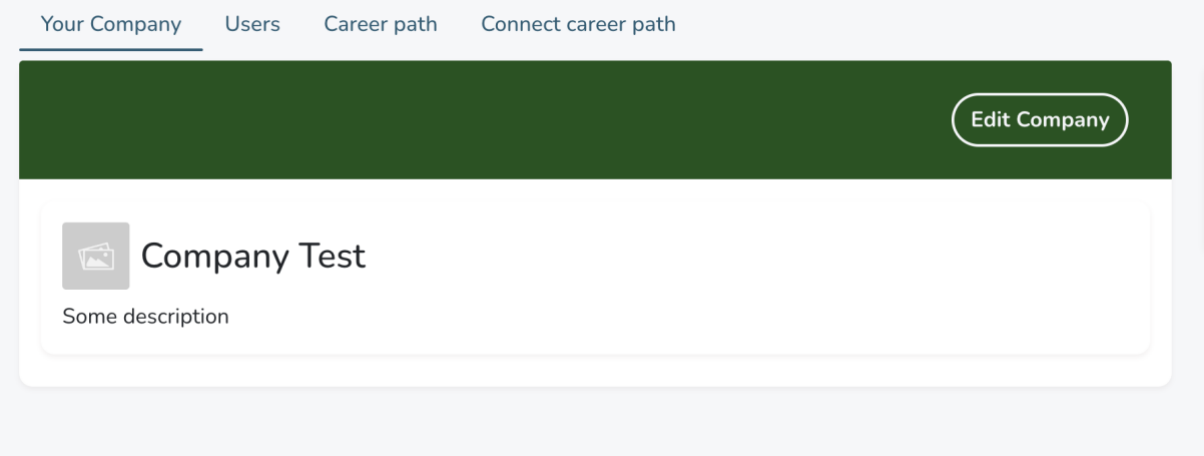


Рис. 10. Результат тестування створення компанії

Таблиця 24

Сценарій використання: Створення користувача адміністратором

Виконавець	Адміністратор
Попередні умови	1. Адміністратор авторизований в системі. 2. Адміністратор перейшов на вкладку Users.
Послідовність дій	1. Натиснути на кнопку "Add user". 2. Ввести email. 3. Виберати роль користувача. 4. Натиснути кнопку Save.
Очікуваний результат	1. Відкривається модальне вікно "Add user". 2. У полі " User email" відображається електронна пошта. 3. Роль користувача обрана. 4. Створюється користувач і з'являється в таблиці користувачів.

При невідповідності послідовності дій користувача має з’явитись повідомлення по помилку від відповідним полем вводу. Нижче наведено приклад доданого адміністратором користувача (рис. 11).

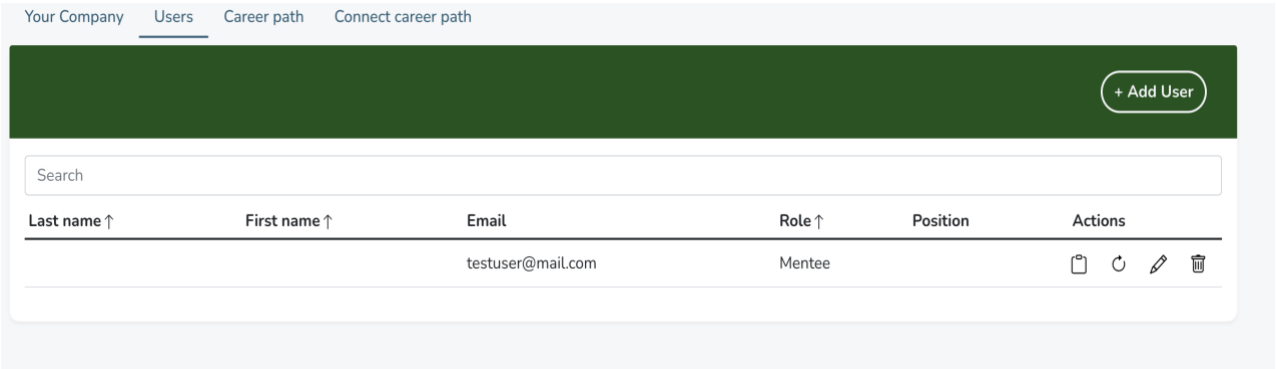


Рис. 11. Результат тестування додавання користувача

Таблиця 25

Сценарій використання: Авторизація користувча

Виконавець	Користувач
Попередні умови	–
Послідовність дій	1. Ввести email. 2. Ввести пароль. 3. Натиснути на кнопку Login.
Очікуваний результат	1. Email відображається в полі вводу. 2. Пароль відображається в полі вводу. 3. Переправлення на наступну сторінку.

При невідповідності послідовності дій користувача кнопка Login має бути заблокована (рис. 12).

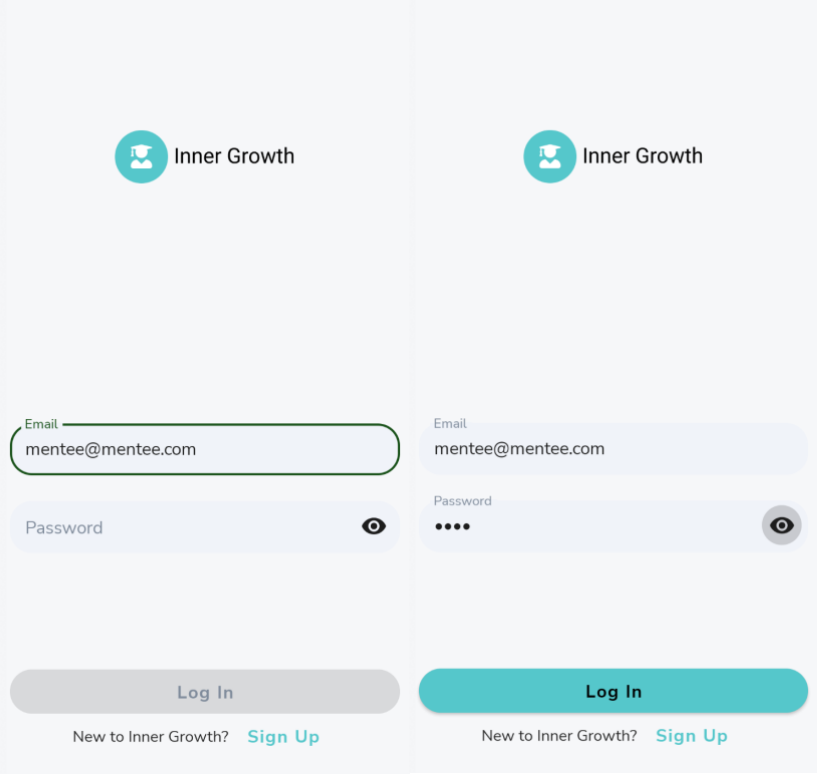


Рис. 12. Результат тестування авторизації користувача

Таблиця 26

Сценарій використання: Авторизація співробітника вперше

Виконавець	Співробітник
Попередні умови	1. Адміністратор додав співробіотника. 2. Співробітник авторизувався в системі.
Послідовність дій	1. Заповнити поля First Name, Last Name. 2. Натиснути кнопку Next. 3. Натиснути кнопку Add Experience. 4. Додати інформацію про досвід. 5. Натиснути кнопку Next. 6. Натиснути кнопку Add Education. 7. Додати інформацію про освіту. 8. Натиснути кнопку Next. 9. Натиснути кнопку Add Language. 10. Додати інформацію про мову. 11. Натиснути кнопку Complete.
Очікуваний результат	1. Відкривається сторінка Onboarding (рис. 13). 2. Показано вкладку Tell Us About Yourself. 3. Після натискання кнопки Next, користувача перенаправляє на вкладку What Is Your Experience? 4. Після натискання кнопки Next, користувача перенаправляє на вкладку What Is Your Education? 5. Після натискання кнопки Next, користувача перенаправляє на вкладку What Is Your Education? 6. Після натискання кнопки Next, користувача перенаправляє на вкладку Addition Info. 7. Після натискання кнопки Complete, користувача перенаправляє екран Home.

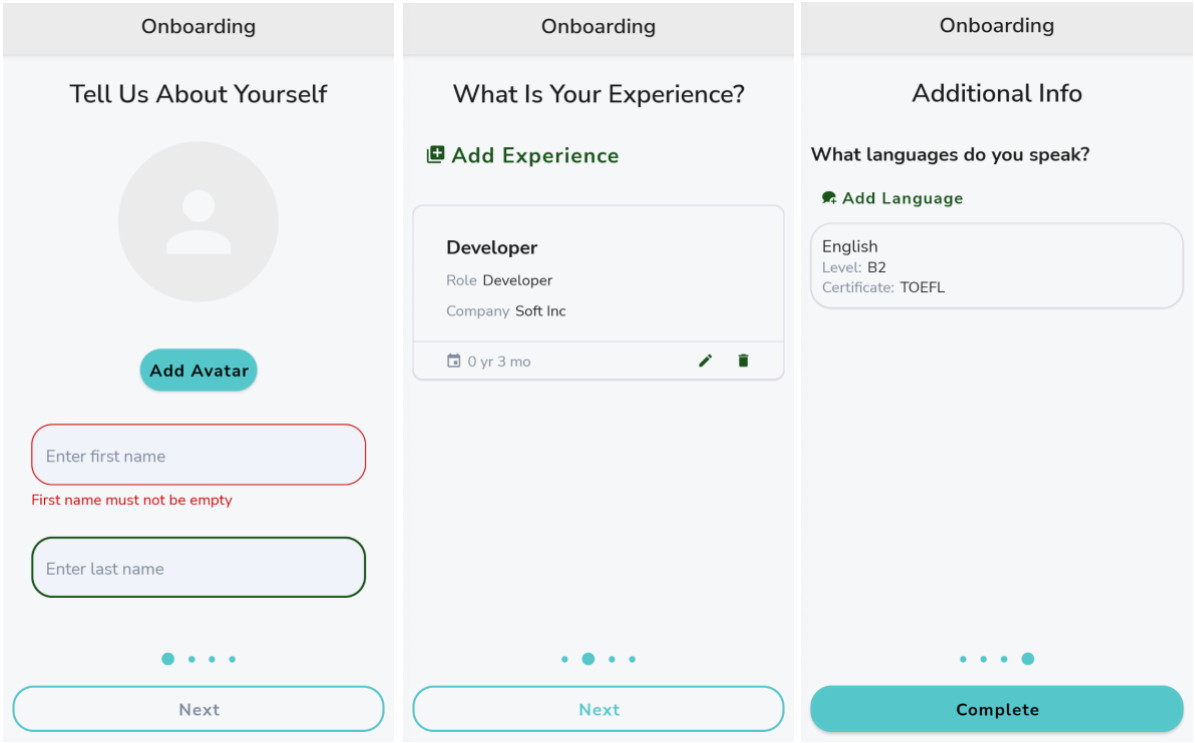


Рис. 13. Результат тестування первинної авторизації

Під час тестування було виявлено проблеми з інтерфейсом на екранах малого розміру, існувала проблема сумісності з старими версіями операційних систем та невеликі помилки в роботі функціональних вимог. Усі знайдені помилки було виправлено.

Також було проведено тестування запитів до серверу за допомогою сервісу Postman. Цей інструмент призначений для роботи з API і дає змогу тестувальнику надсилати запити до сервісів і обробляти отримані відповіді. З його допомогою можна перевірити функціональність бекенда і переконатися в його правильній роботі [20]. Було перевірено запити на відповідність HTTP заданих їм методам та правильність відповіді для кожного запиту.

Нижче наведено один із проведених тестів роботи серверу (рис. 14).

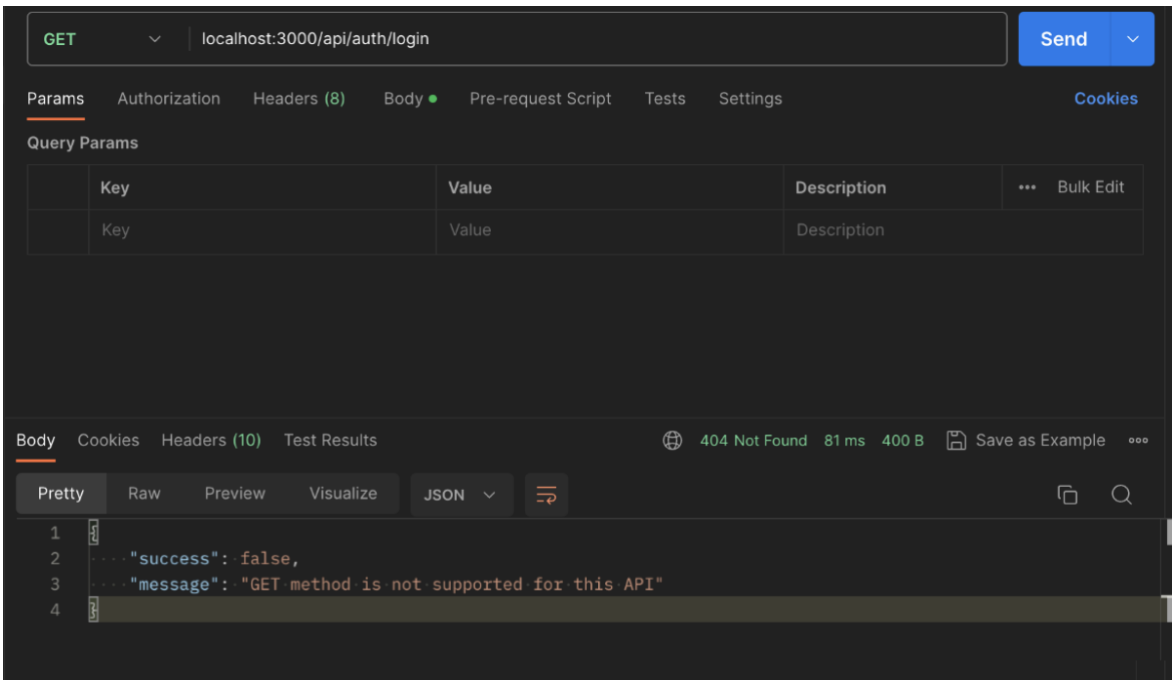


Рис. 14. Результат тестування запитів через сервіс Postman

4.2. Порівняння мобільного застосунку з наявними аналогами

Після проведеного аналізу існуючих аналогів конкурентів у першому розділі, було створено порівняльну таблицю їх переваг і недоліків. Розробка програмного застосунку відбувалася згідно вимог описаних у підрозділі 1.4.

В кінцевому результаті розроблено програмний застосунок який:

- Має мобільний модуль для користувачів ролі ментор та співробітник.
- Має браузерний модуль для ролі адміністратор.
- Зберігає вразливі данні (інформація про користувача, його пароль) в шифрованому вигляді тим самим забезпечуючи безпеку застосунку.
- Має сучасний та зрозумілий інтерфейс.

Було враховано недоліки аналогів з підрозділ 1.3 та реалізовано в програмному застосунку, а саме:

- Програмний застосунок є кросплатформним.
- Додано перелік можливостей для розвитку в компанії.
- Наявність ролі Ментор.

- Є особистий профіль користувача з резюме.

Розроблений програмний застосунок має основні необхідні функціональні можливості, що необхідні для відстеження розвитку фахових та поведінкових компетенцій співробітників. Має всі необхідні вимоги для адміністраторської сторінки. Отже, розроблений програмний застосунок може конкурувати з конкурентами аналогами.

4.3. Пропозиції для майбутнього покращення програмного застосунку

Розроблений програмний застосунок є мінімально життєздатним продуктом (MVP) [21]. Для належного конкурентоспроможного ісування на ринку програмний застосунок потребує постійної підтримки та розширення функціоналу.

У наступній версії застосунку буде реалізовано інші вимоги до програмного застосунку, а саме:

1. Функціональні вимоги ментора.
2. Доступ користувачів та менторів до мапи кар’єрного розвитку.
3. Можливість відслідковування розвитку для співробітників.

Також, було сформовано такі рекомендації щодо подальшого розширення та покращення застосунку:

- Розширення функціоналу для ролі Адмін.
- Розширення мобільного модуля поточним функціоналом Адміну.
- Додати мобільні сповіщення (Push Notifications) про початок та кінець OKR циклу.
- Додати чат для спілкування Ментора і Співробітника.
- Додати інтеграцію з іншими програмними додатками.
- Додати різні мови для інтерфейсу застосунку.

Зазначений вище додатковий функціонал буде враховано та реалізовано в наступних версіях програмного застосунку.

4.4. Висновки до розділу

У цьому розділі було описано процес тестування програмного застосунку. Функціональне тестування відбувалось протягом усього часу розробки ПЗ методом ручного тестування. Для функціональних вимог ролей адміністратор та співробітник було сформовано сценарії використання. Функціональне тестування відбувалось згідно цих сценаріїв. В результаті тестування було виявлено графічні та логічні помилки програмного застосунку. В подальшому процесі розробки всі помилки було виправлено.

Було проведено тестування запитів до серверу за допомогою сервісу Postman. Було перевірено відповідність запитів заданим для них HTTP методам та правильність відповідей запитів.

Також було проведено аналіз розробленого програмного застосунку відносно існуючих конкурентів аналогів. Було виявлено та заначено переваги програмного застосунку над конкурентами та описано результат розробки.

Оскільки програмний застосунок є MVP, то було сформовано рекомендації для подальшого розвитку та підтримки програмного застосунку. Найбільш пріоритетними завданнями є створення чата для співробітників та менторів і наявність мобільних повідомлень.

ВИСНОВКИ

Метою даного дипломного проєкту було розроблення програмного застосунку для управління розвитком фахових та поведінкових компетенцій співробітників компаній. Цей програмний застосунок допоможе компаніям підвищити рівень кваліфікації співробітників та покращить систему менторства.

Було проведено аналіз предметної області та проаналізовано контуренти аналоги. Враховуючи зібрані під час аналізу дані, було сформовано конкурентні переваги та проаналізовано функціональні і нефункціональні вимоги до програмного застосунку. Також було визначено вимоги до безпеки та якості ПЗ.

Для кожного модулю програмного застосунку, а саме: мобільного, барузерного та серверного, було проаналізовано декілька мов програмування. Після проведеного аналізу було обрано мову TypeScript, для створення єдиної кодової бази всього програмного застосунку та перевагою є строга типізація. Для мобільного модулю було обрано фреймворк React Native, а для браузерного React. В якості бази даних було обрано СКБД PostgreSQL.

Клієнтський модуль програмного застосунку складається з браузерного та мобільного модулів. Клієнтський модуль взаємодіє із серверним за допомогою REST API. Клієнтський модуль реалізовано за допомогою патерну проєктування MVVM.

Реалізований програмний застосунок має:

- Має мобільний модуль для користувачів ролі ментор та співробітник.
- Має браузерний модуль для ролі адміністратор.
- Зберігає вразливі данні в шифрованому вигляді тим самим забезпечуючи безпеку застосунку.
- Має сучасний та зрозумілий інтерфейс.

- Програмний застосунок є кросплатформним.
- Має перелік можливостей для розвитку в компанії.
- Має особистий профіль користувача з резюме.
- Співробітник має можливість відслідковувати свій розвиток за циклами OKR.

Розроблений програмний застосунок виконаний в повному обсязі, відповідає всім зазначеним вище функціональним та нефункціональним вимогам. Було проведено функціональне тестування ручним методом за наведеними сценаріями використання програмного застосунку. Також було проведено тестування запитів до серверу за допомогою сервісу Postman. Всі помилки та недоліки було виправлено.

Після проведеного аналізу розробленого ПЗ з конкурентними аналогами, було визначено переваги над конкурентами аналогами, що робить програмний застосунок конкурентоспроможним.

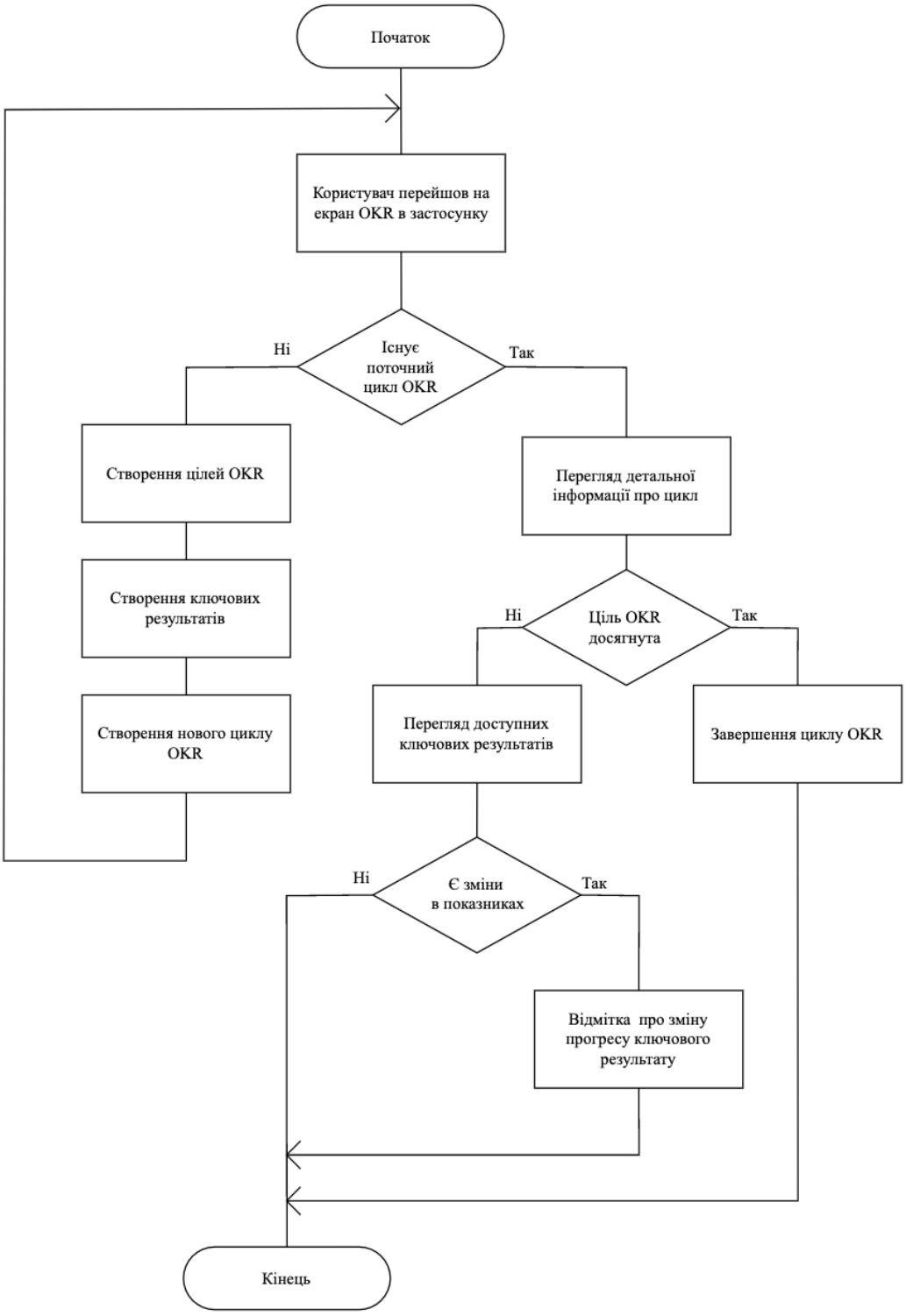
СПИСОК ВИКОРИСТАНИХ ЛІТЕРАТУРНИХ ДЖЕРЕЛ

1. 66% опитаних IT-спеціалістів вважають, що IT-сертифікація суттєво впливає на якість роботи: [Електронний ресурс] — Режим доступу: https://www.epravda.com.ua/publications/2022/02/1. — (дата звернення: 15.04.2023).
2. Atomic Kotlin [Текст] / Eckel B., Isakova S.. — Mindview LLC, 2021. — 636 с.
3. Head First. Програмування на JavaScript [Текст] / Фірмен Е.. — Фабула, 2022. — 672 с.
4. OKRs vs KPIs: [Електронний ресурс] — Режим доступу: https://business.adobe.com/blog/basics/okr-vs-kpi. — (дата звернення: 15.04.2023).
5. Dart programming language: [Електронний ресурс] — Режим доступу: https://dart.dev/. — (дата звернення: 20.04.2023).
6. Essential TypeScript 4: From Beginner to Pro [Текст] / Freeman A.. — Apress, 2021. — 581 с.
7. C#. Modern, open-source programming language for .NET. Microsoft: [Електронний ресурс] — Режим доступу: https://dotnet.microsoft.com/en-us. — (дата звернення: 21.04.2023).
8. Пришвидшений курс Python. Практичний, проєктно-орієнтований вступ до програмування [Текст] / Маттес Е.. — Видавництво Старого Лева, 2021. — 600 с.
9. Node.js v20.2.0 Documentation. Node.js: [Електронний ресурс] — Режим доступу: https://nodejs.org/docs. — (дата звернення: 25.04.2023).
10. What Is PostgreSQL?. PostgreSQL Documentation: [Електронний ресурс] — Режим доступу: https://www.postgresql.org/docs. — (дата звернення: 25.04.2023).

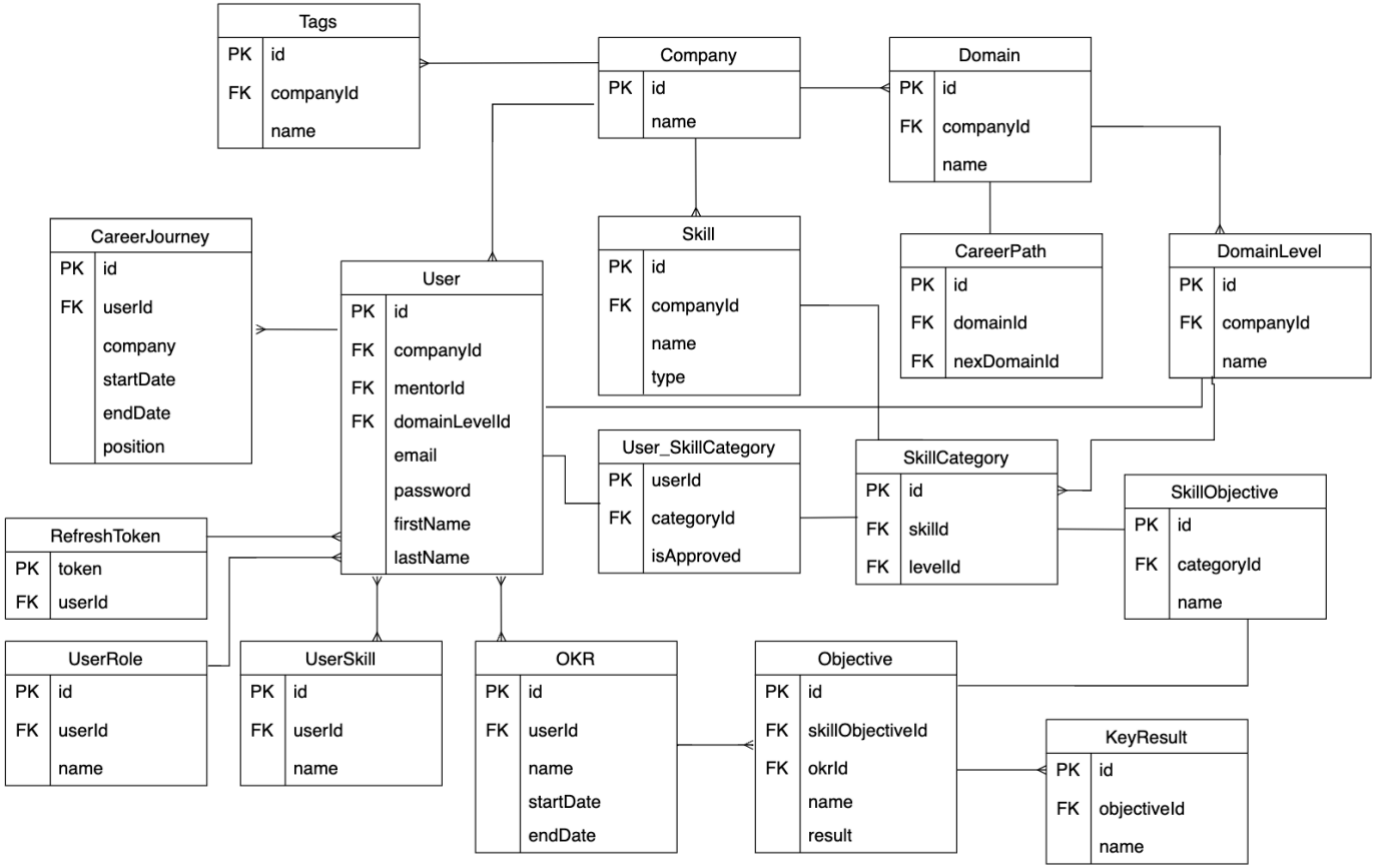
11. MongoDB for Web Development [Текст] / Pirtle M.. — Pearson Education, Limited, 2018. — 360 с.
12. SQLite: [Електронний ресурс] — Режим доступу: https://www.sqlite.org/index.html. — (дата звернення: 25.04.2023).
13. RESTful Web APIs: Services for a Changing World [Текст] / Richardson L., Ruby S., Amundsen M.. — O'Reilly Media, 2013. — 406 с.
14. HTTP: The Definitive Guide [Текст] / Gourley D., Totty B.. — O'Reilly Media, Inc., 2002. — 656 с.
15. Learning React Native. O’Reilly Online Learning: [Електронний ресурс] — Режим доступу: https://www.oreilly.com/library/view. — (дата звернення: 05.05.2023).
16. Santiago A. How Does Redux Work?. Medium: [Електронний ресурс] — Режим доступу: https://medium.com/@the.asantiagojr/how-does-redux-work-b1cce46d4fa6. — (дата звернення: 05.05.2023).
17. Чому програмне забезпечення потрібно тестувати? Голос українською: [Електронний ресурс] — Режим доступу: https://uagolos.com/chomu-prohramne. — (дата звернення: 10.05.2023).
18. Testing Techniques in Software Engineering [Текст] / P. Borba. — Springer Berlin Heidelberg, 2010. — 675 с.
19. What is Manual Testing. Software Testing Class: [Електронний ресурс] — Режим доступу: https://www.softwaretestingclass.com/what-is-manual-testing/. — (дата звернення: 15.05.2023).
20. What is Postman?. Postman API Platform: [Електронний ресурс] — Режим доступу: https://www.postman.com/product/. — (дата звернення: 15.05.2023).
21. Мінімально життєздатний продукт: [Електронний ресурс] — Режим доступу: https://uk.wikipedia.org/wiki/MVP. — (дата звернення: 15.05.2023).

ДОДАТКИ

Додаток 1
Копії графічних матеріалів

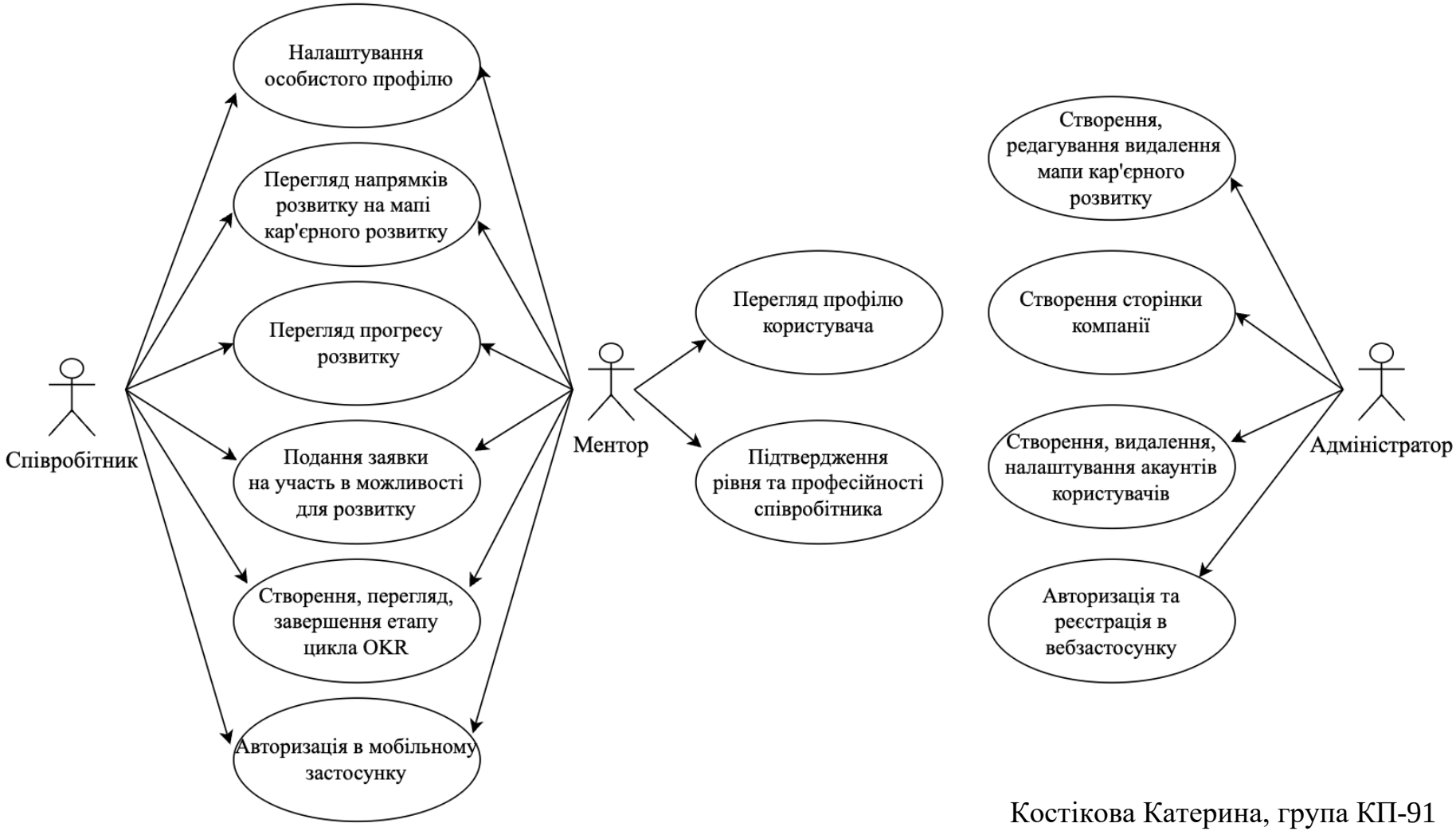


ДП.045440-06-99
Програмний застосунок «Inner Growth» для управління розвитком компетенцій співробітників. Діаграма діяльності основного сценарію використання. Схема діаграми



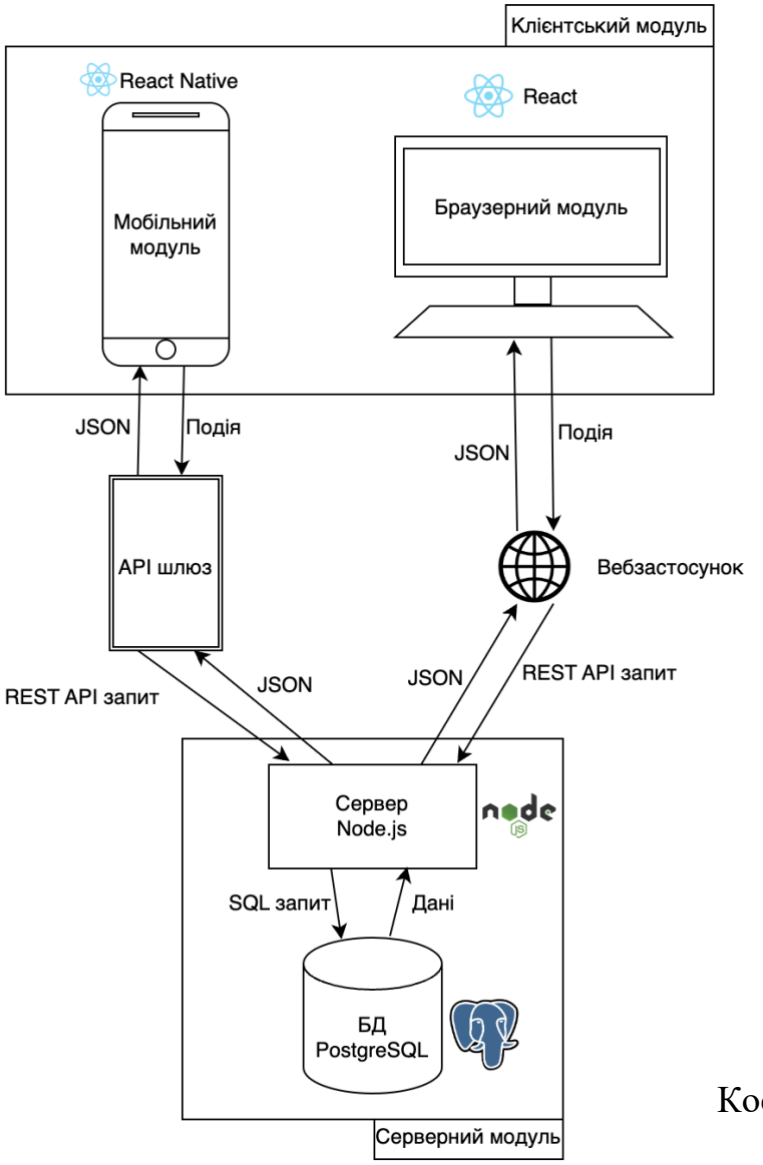
ДП.045440-07-99
Програмний застосунок «Inner Growth» для управління розвитком компетенцій співробітників. Схема бази даних. Схема даних

Діаграма використання програмного застосунку



Костікова Катерина, група КП-91

Архітектура програмного застосунку



Костікова Катерина, група КП-91

Додаток 2
Лістинг програми

2.1. Компонент навігації Root.tsx

```
import React, { useEffect } from 'react';
import { createNativeStackNavigator } from '@react-navigation/native-
stack';

import { AppRoute, AuthRoute } from 'src/common/enums/navigation';
import { useAppDispatch, useAppSelector } from 'src/hooks';
import {
  LoginScreen,
  RegisterScreen,
  CompleteRegistrationScreen,
} from 'src/screens';
import { RootStackParamList } from 'src/common/types';
import { authActions } from 'src/store/actions';
import AppNavigation from './app/app-navigation';

const RootStack = createNativeStackNavigator<RootStackParamList>();

const defaultScreenOptions = {
  headerShown: false,
};

const RootNavigation = () => {
  const dispatch = useAppDispatch();
  const { user } = useAppSelector(state => state.auth);

  useEffect(() => {
    dispatch(authActions.loadCurrentUser());
  }, [dispatch]);

  return (
    <RootStack.Navigator screenOptions={defaultScreenOptions}>
      {user ? (
        <RootStack.Screen name={AppRoute.APP} component={AppNavigation} />
      ) : (
        <>
          <RootStack.Screen name={AuthRoute.SIGN_IN}
component={LoginScreen} />
          <RootStack.Screen
            name={AuthRoute.SIGN_UP}
            component={RegisterScreen}
          />
          <RootStack.Screen
            name={AuthRoute.COMPLETE_REGISTRATION}
            component={CompleteRegistrationScreen}
          />
        </>
      )}
    </RootStack.Navigator>
  );
};

export default RootNavigation;
```

2.2. Компонент навігації AppNavigation.tsx

```
import React, { useMemo } from 'react';
import { createNativeStackNavigator } from '@react-navigation/native-
stack';
```

```
import {
  AddKeyResultScreen,
  AddOKRScreen,
  QuizScreen,
  OpportunityDetailsScreen,
} from 'src/screens';

import { AppRoute, UserRoleType } from 'src/common/enums';
import { AppStackParamList } from 'src/common/types';
import { useAppSelector, useStackScreenOptions } from 'src/hooks';
import { AppTabsNavigation } from '../app-tabs';
import { OnboardingNavigation } from '../onboarding';

const Stack = createNativeStackNavigator<AppStackParamList>();

const AppNavigation: React.FC = () => {
  const { user } = useAppSelector(state => state.auth);

  const isCompletedOnboarding = user?.firstName;

  const initialRouteName = useMemo(() => {
    if (isCompletedOnboarding) {
      return AppRoute.APP_TABS;
    }
    return AppRoute.ONBOARDING_SETUP;
  }, [isUserAdmin, isCompletedOnboarding]);

  return (
    <Stack.Navigator
      initialRouteName={initialRouteName}
      screenOptions={useStackScreenOptions()}
    >

        <>
          {isCompletedOnboarding ? (
            <>
              <Stack.Screen
                name={AppRoute.APP_TABS}
                component={AppTabsNavigation}
                options={{ headerShown: false }}
              />
              <Stack.Screen
                name={AppRoute.ADD_OKR}
                component={AddOKRScreen}
                options={{
                  title: 'Add New Objective',
                }}
              />
              <Stack.Screen
                name={AppRoute.ADD_KEY_RESULT}
                component={AddKeyResultScreen}
                options={{
                  title: 'Add Key Result',
                }}
              />
              <Stack.Screen
                name={AppRoute.OPPORTUNITY_DETAILS}
                component={OpportunityDetailsScreen}
                options={{ title: 'Opportunity Details' }}
              />
              <Stack.Screen
                name={AppRoute.QUIZ}
                component={QuizScreen}
```

```
              options={{ title: 'Quiz' }}
            />
          </>
        ) : (
          <Stack.Screen
            name={AppRoute.ONBOARDING_SETUP}
            component={OnboardingNavigation}
            options={{ headerShown: false }}
          />
        )}
      </>
    )}
  </Stack.Navigator>
 );
};

export default AppNavigation;
```

2.3. Компонент навігації AppTabsNavigation.tsx

```
import React from 'react';
import { createMaterialBottomTabNavigator } from '@react-
navigation/material-bottom-tabs';
import { EventArg } from '@react-navigation/native';

import {
  AccountIcon,
  HomeIcon,
  RadarIcon,
  ShieldSearchIcon,
} from 'src/components';

import { MenteeHomeScreen, OKRScreen, OpportunitiesScreen } from
'src/screens';

import { AppRoute } from 'src/common/enums';
import { AppTabsParamList } from 'src/common/types';
import { useAppSelector } from 'src/hooks';
import { showInfoToast } from 'src/helpers/notifications';
import { COMPLETE_QUIZ } from 'src/common/constants';
import ProfileNavigation from '../profile/profile-navigation';
import { useColor } from 'src/hooks';
import useStyles from './styles';

const Tab = createMaterialBottomTabNavigator<AppTabsParamList>();

const AppTabsNavigation = () => {
  const { user } = useAppSelector(state => state.auth);
  const styles = useStyles();
  const colorNavSecondary = useColor('NAVIGATION_SECONDARY');

  const navigationEvent = {
    tabPress: (e: EventArg<'tabPress', true, undefined>) => {
      if (!user?.isCompleteTest) {
        e.preventDefault();
        showInfoToast(COMPLETE_QUIZ);
      }
    },
  };

  return (
```

```
    <Tab.Navigator
      shifting={false}
      initialRouteName={user?.isCompleteTest ? AppRoute.HOME :
AppRoute.PROFILE}
      style={styles.container}
      barStyle={styles.barStyle}
      inactiveColor={colorNavSecondary}
    >
      <Tab.Screen
        name={AppRoute.HOME}
        component={MenteeHomeScreen}
        options={{
          tabBarIcon: ({ color }) => <HomeIcon color={color} size={25} />,
        }}
        listeners={navigationEvent}
      />
      <Tab.Screen
        name={AppRoute.OPPORTUNITIES}
        component={OpportunitiesScreen}
        options={{
          tabBarIcon: ({ color }) => (
            <ShieldSearchIcon color={color} size={25} />
          ),
        }}
        listeners={navigationEvent}
      />
      <Tab.Screen
        name={AppRoute.OKR}
        component={OKRScreen}
        options={{
          tabBarIcon: ({ color }) => <RadarIcon color={color} size={25} />,
        }}
        listeners={navigationEvent}
      />
      <Tab.Screen
        name={AppRoute.PROFILE}
        component={ProfileNavigation}
        options={{
          tabBarIcon: ({ color }) => <AccountIcon color={color} size={25}
/>,
        }}
      />
    </Tab.Navigator>
  );
};

export default AppTabsNavigation;
```

2.4. Компонент навігації OnboardingNavigation.tsx

```
import React from 'react';

import {
  createNativeStackNavigator,
  NativeStackNavigationOptions,
} from '@react-navigation/native-stack';

import { OnboardingRoute } from 'src/common/enums';

import {
```

```
  AddEducationScreen,
  AddExperienceScreen,
  AddLanguageScreen,
  AddLocationScreen,
  OnboardingScreen,
} from 'src/screens';

import { OnboardingStackParamList } from 'src/common/types';
import { useStackScreenOptions } from 'src/hooks';

const Stack = createNativeStackNavigator<OnboardingStackParamList>();

const addScreenGroupOptions: NativeStackNavigationOptions = {
  presentation: 'modal',
};

const OnboardingNavigation = () => {
  return (
    <Stack.Navigator screenOptions={useStackScreenOptions()}>
      <Stack.Screen
        name={OnboardingRoute.ONBOARDING}
        component={OnboardingScreen}
      />
      <Stack.Group screenOptions={addScreenGroupOptions}>
        <Stack.Screen
          name={OnboardingRoute.ADD_EXPERIENCE}
          component={AddExperienceScreen}
          options={{ title: 'Add Experience' }}
        />
        <Stack.Screen
          name={OnboardingRoute.ADD_EDUCATION}
          component={AddEducationScreen}
          options={{ title: 'Add Education' }}
        />
        <Stack.Screen
          name={OnboardingRoute.ADD_LANGUAGE}
          component={AddLanguageScreen}
          options={{ title: 'Add Language' }}
        />
        <Stack.Screen
          name={OnboardingRoute.ADD_LOCATION}
          component={AddLocationScreen}
          options={{ title: 'Add Location' }}
        />
      </Stack.Group>
    </Stack.Navigator>
  );
};

export default OnboardingNavigation;
```

Додаток 3

Копія презентації

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ
ІГОРЯ СІКОРСЬКОГО”

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

КАФЕДРА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ КОМП’ЮТЕРНИХ СИСТЕМ

ПРОГРАМНИЙ ЗАСТОСУНОК «INNER GROWTH» ДЛЯ УПРАВЛІННЯ РОЗВИТКОМ КОМПЕТЕНЦІЙ СПІВРОБІТНИКІВ

Виконала: студентка групи КП-91 Костікова Катерина Вадимівна

Керівник: доцент кафедри ПЗКС, к.т.н., Люшенко Леся Анатоліївна

Київ — 2023

ПОСТАНОВКА ЗАДАЧІ

Мета проєкту: розробити програмний застосунок “Inner Growth”, який дозволить визначати, планувати та відстежувати, а також оцінювати розвиток компетенцій співробітників компанії. Таким чином, це дає можливість компаніям узгоджувати індивідуальні цілі співробітників з бізнес-цілями компанії, сприяти професійному зростанню та забезпечувати висококваліфіковану робочу силу, здатну відповідати потребам галузі.

Аналіз:

- аналіз предметної області
- аналіз існуючих рішень

Розроблення:

- обрання засобів реалізації програмного застосунку
- розроблення програмного застосунку

Результати:

- порівняти з розглянутими конкурентами аналогами
- навести рекомендації для покращення та розвитку програмного застосунку

АКТУАЛЬНІСТЬ

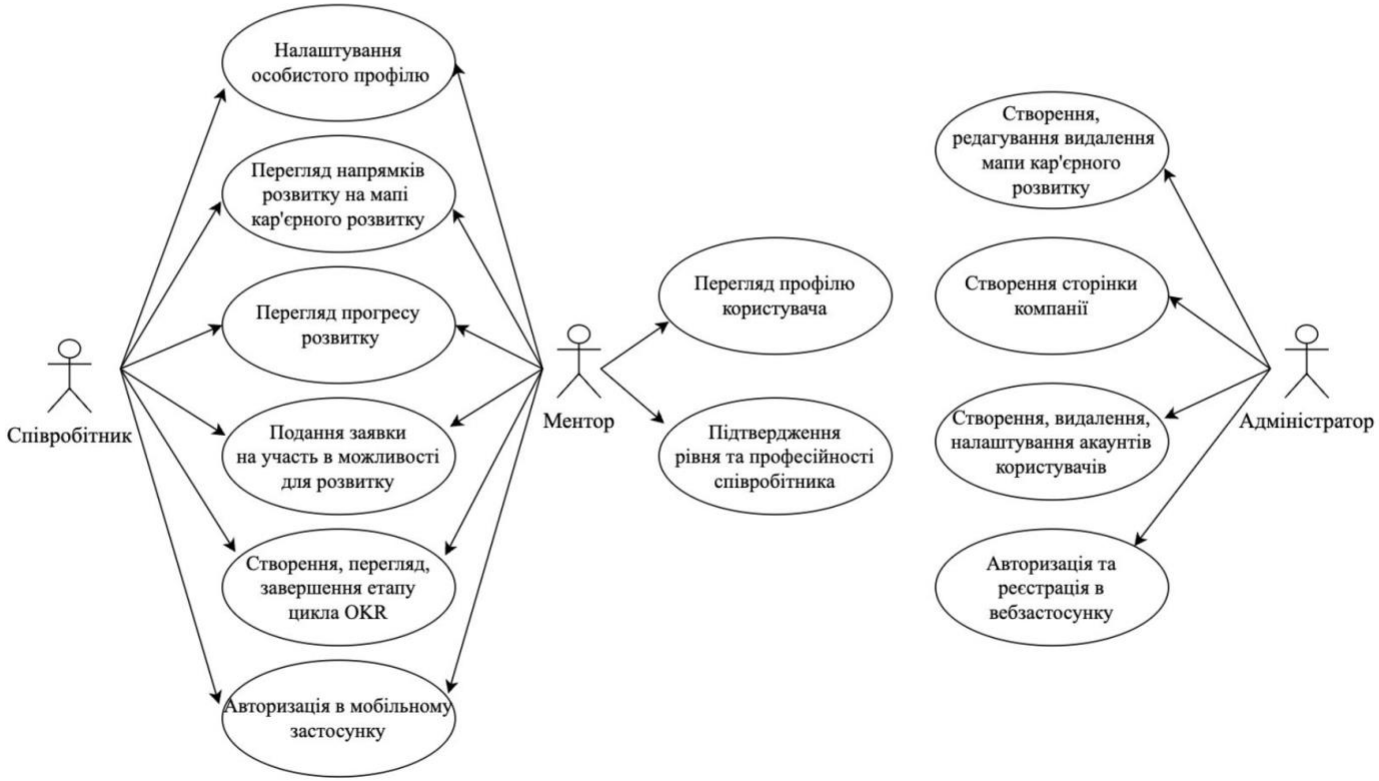
1. Успіх і конкурентоспроможність компаній значною мірою залежать від здібностей і досвіду їхньої робочої сили

2. Постійне зростання та розвиток співробітників відіграють ключову роль у випередженні конкурентів та адаптації до швидкого розвитку IT галузі.

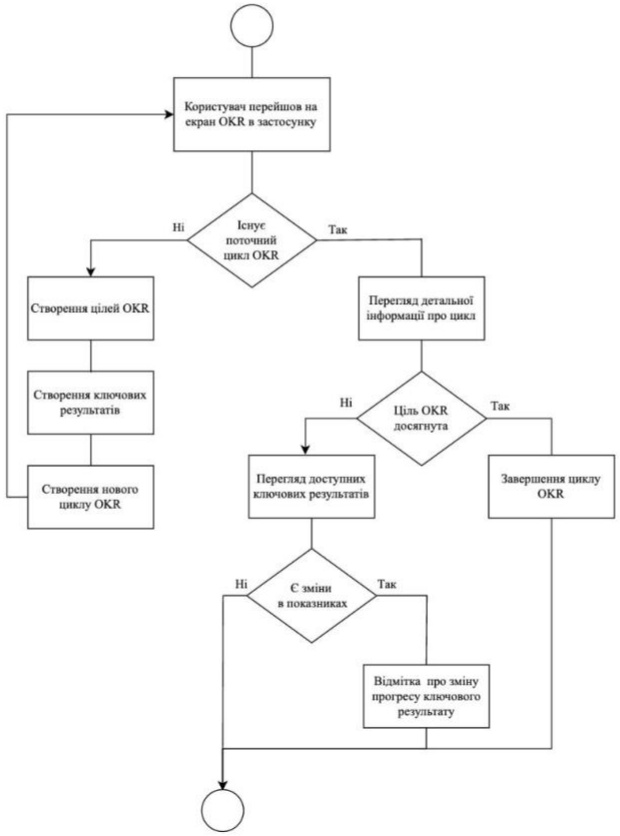
3. Недосконалість наявних рішень:
   a. Відсутність індивідуального підходу до кожного співробітника
   b. Відсутність системи менторства

ІСНУЮЧІ АНАЛОГИ

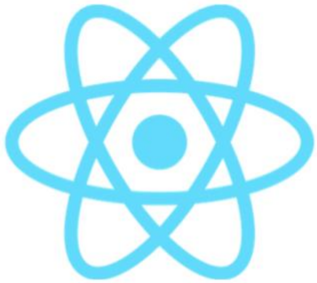
ДІАГРАМА ВАРІАНТІВ ВИКОРИСТАННЯ

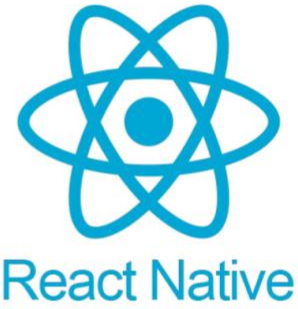


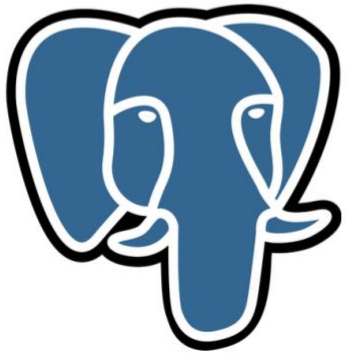
ДІАГРАМА ДІЯЛЬНОСТІ ОСНОВНОГО СЦЕНАРІЮ ВИКОРИСТАННЯ



ЗАСОБИ РЕАЛІЗАЦІЇ







АРХІТЕКТУРА ПРОГРАМНОГО ЗАСТОСУНКУ

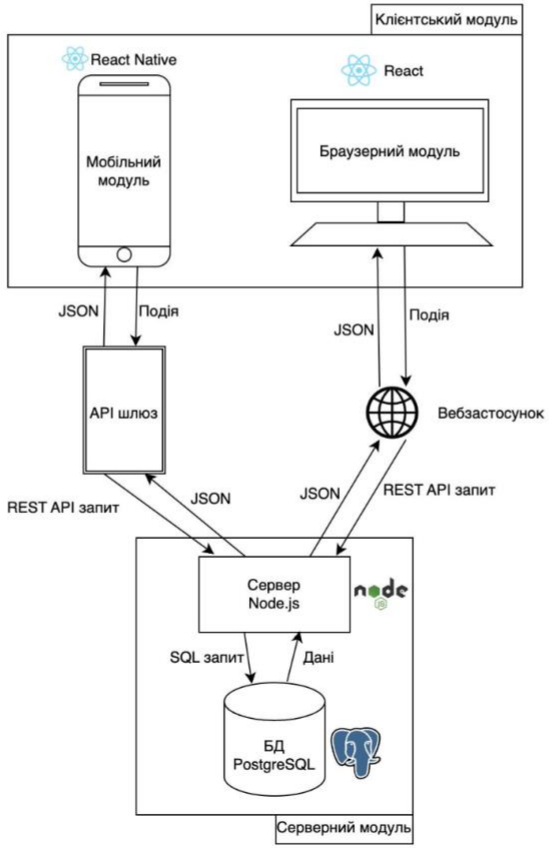
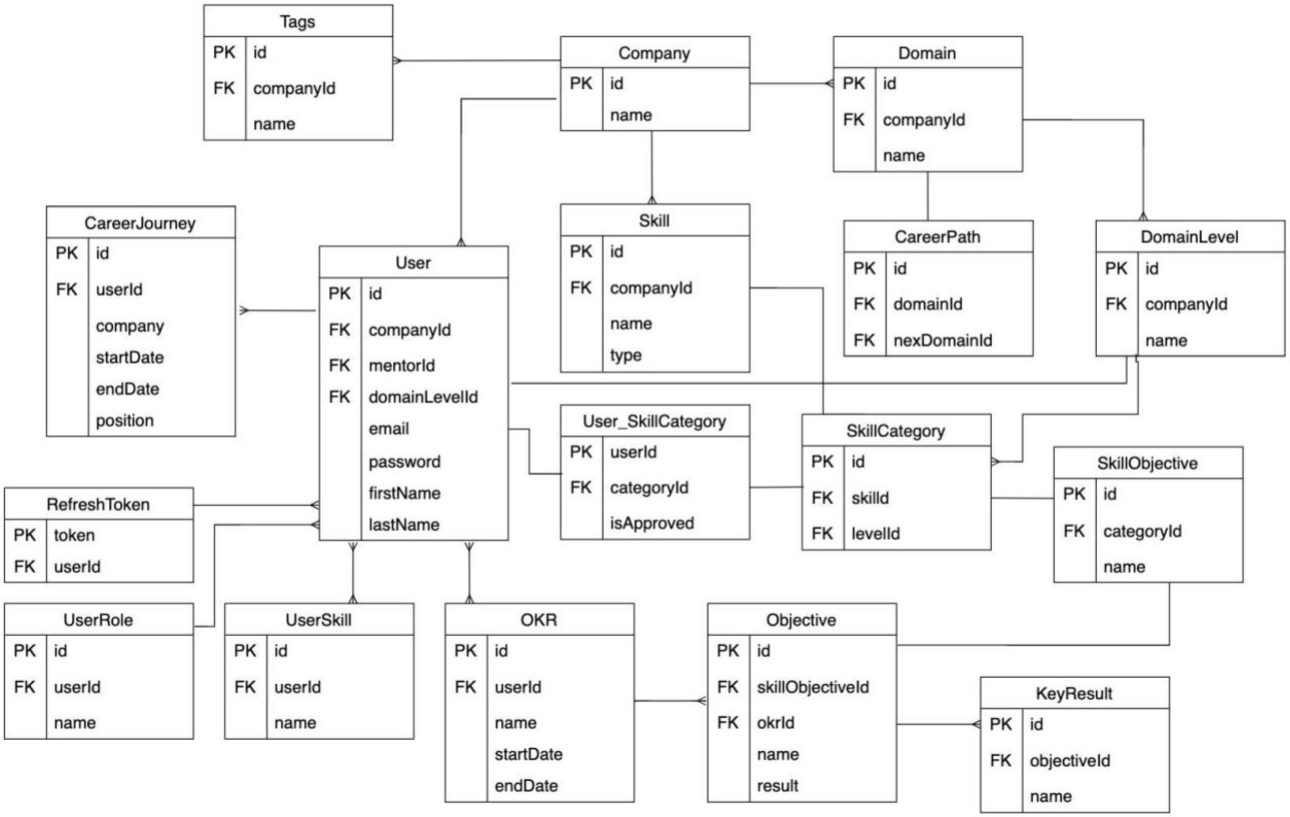
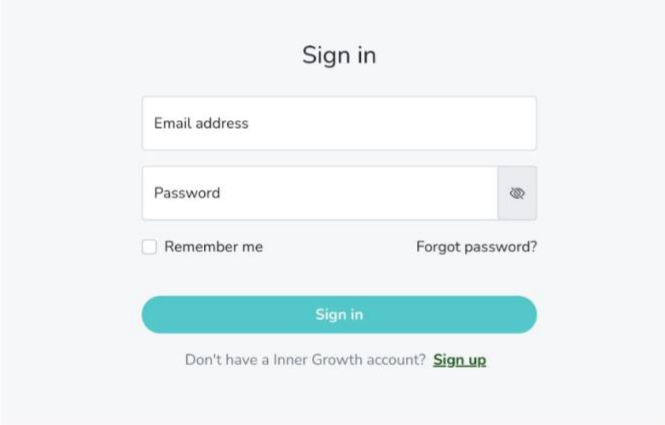
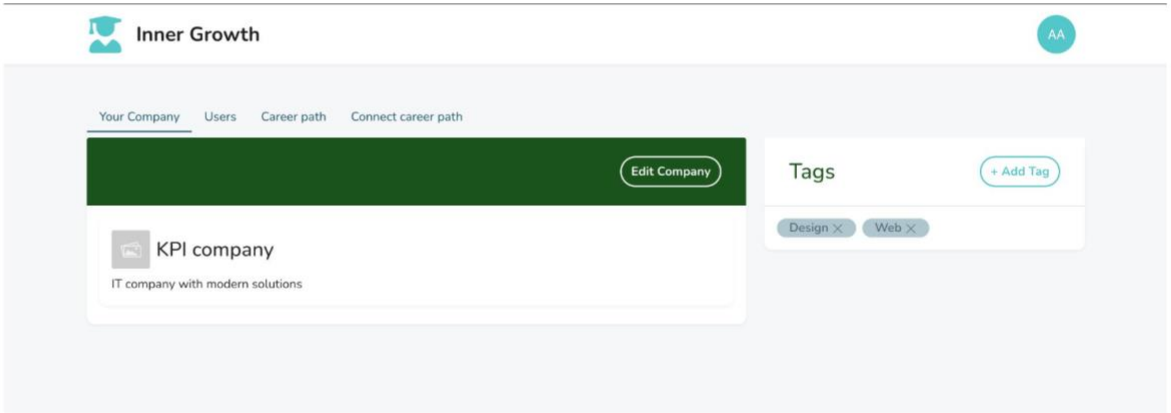


СХЕМА БАЗИ ДАНИХ

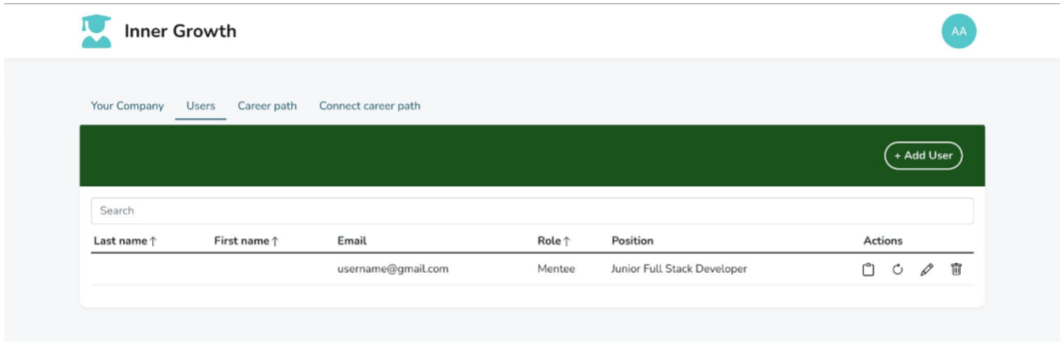


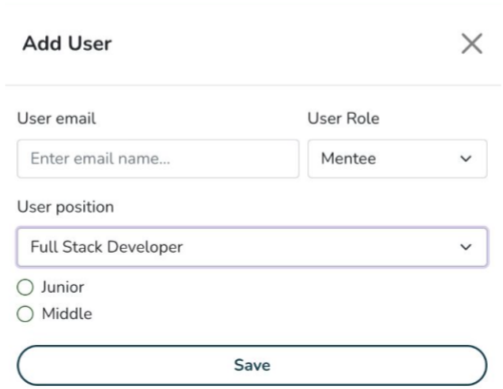
ПРИКЛАД РОБОТИ ЗАСТОСУНКУ. БРАУЗЕРНИЙ МОДУЛЬ



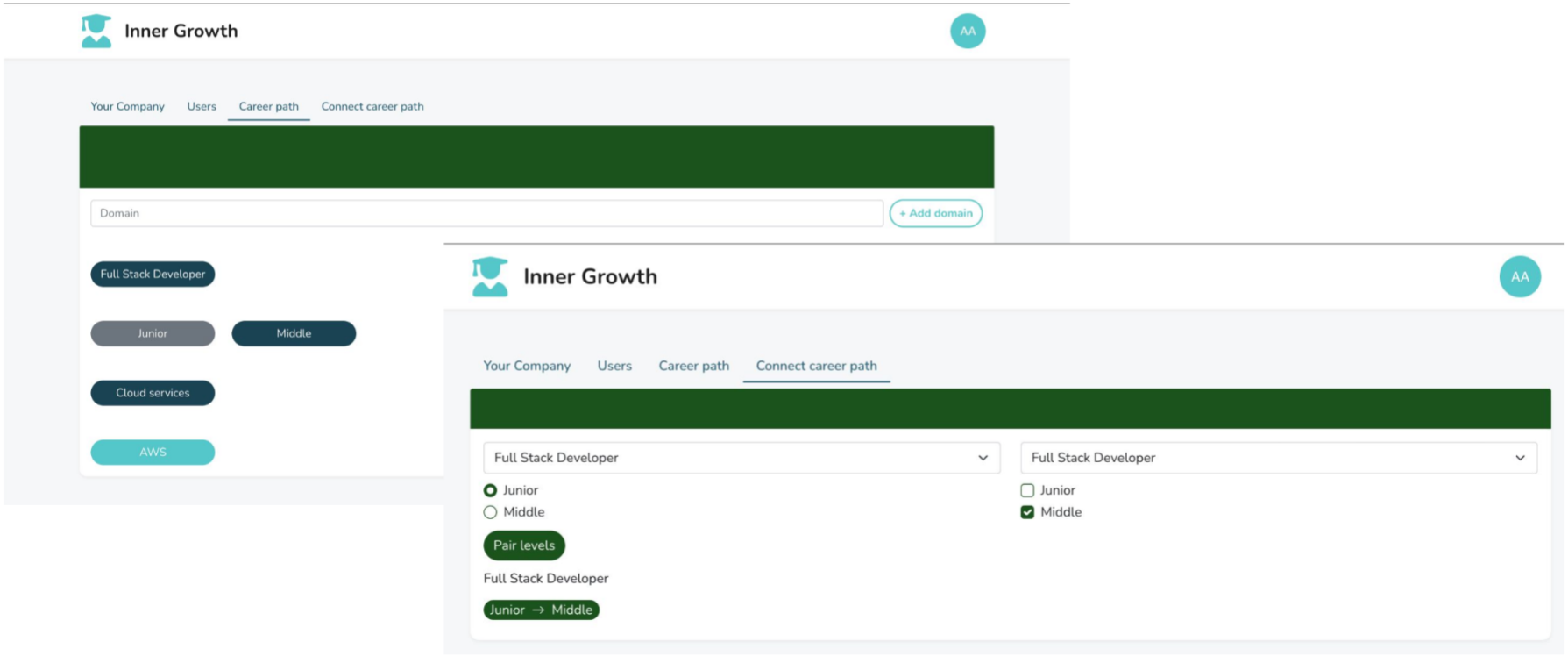


ПРИКЛАД РОБОТИ ЗАСТОСУНКУ. БРАУЗЕРНИЙ МОДУЛЬ

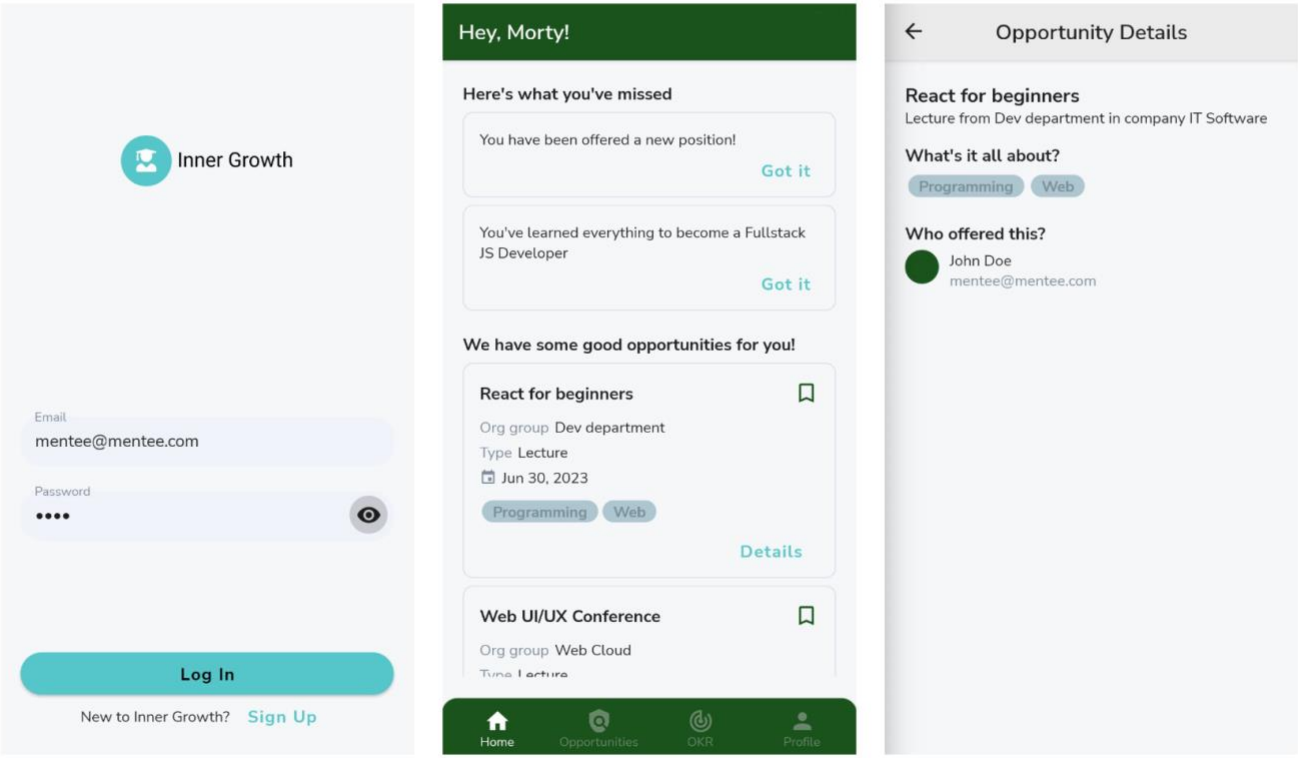




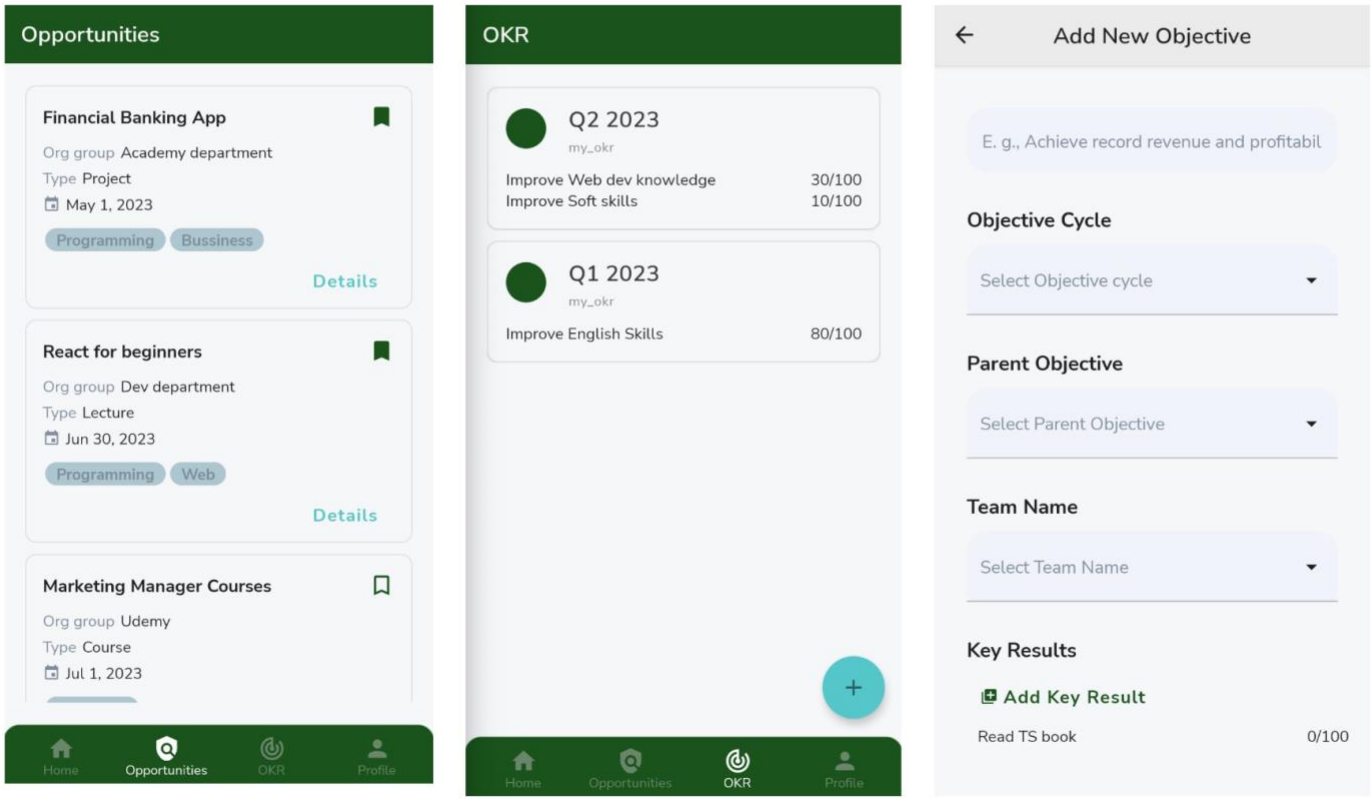
ПРИКЛАД РОБОТИ ЗАСТОСУНКУ. БРАУЗЕРНИЙ МОДУЛЬ



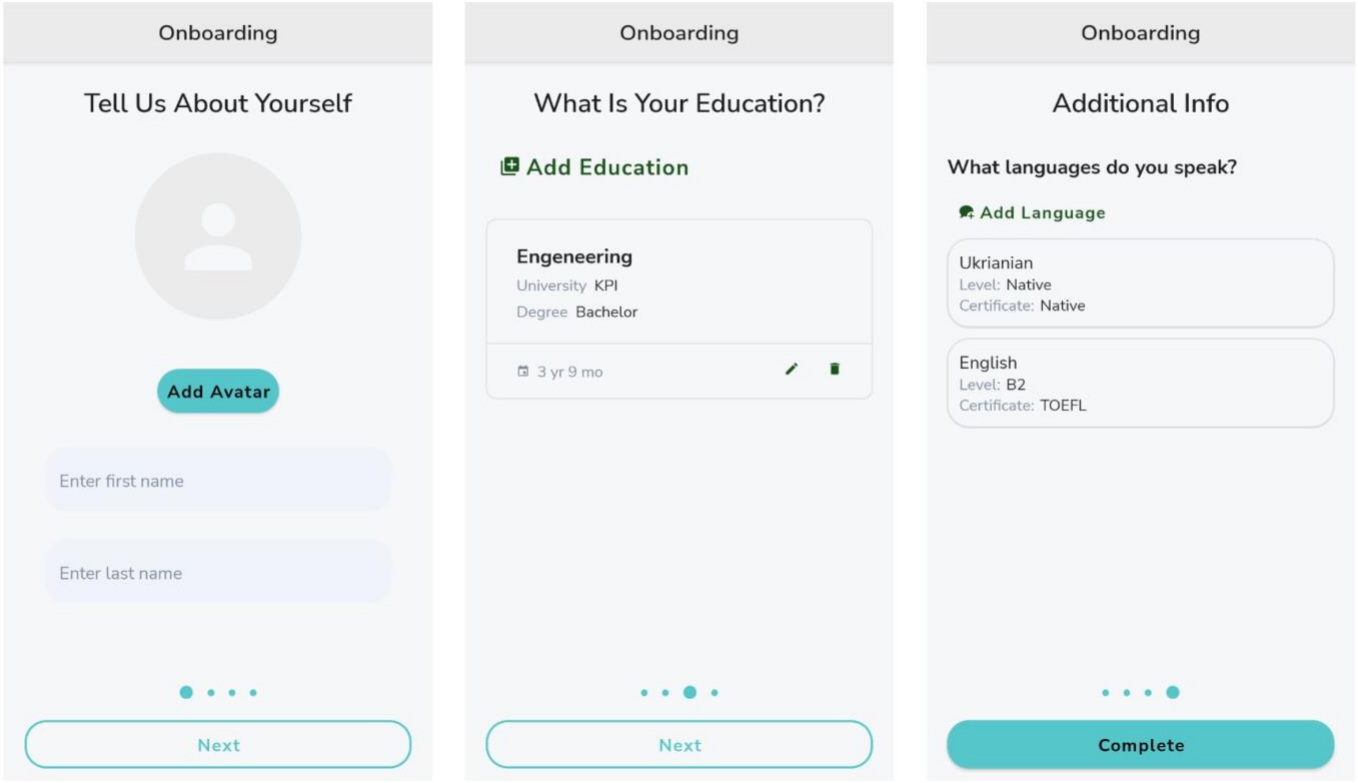
ПРИКЛАД РОБОТИ ЗАСТОСУНКУ. МОБІЛЬНИЙ МОДУЛЬ



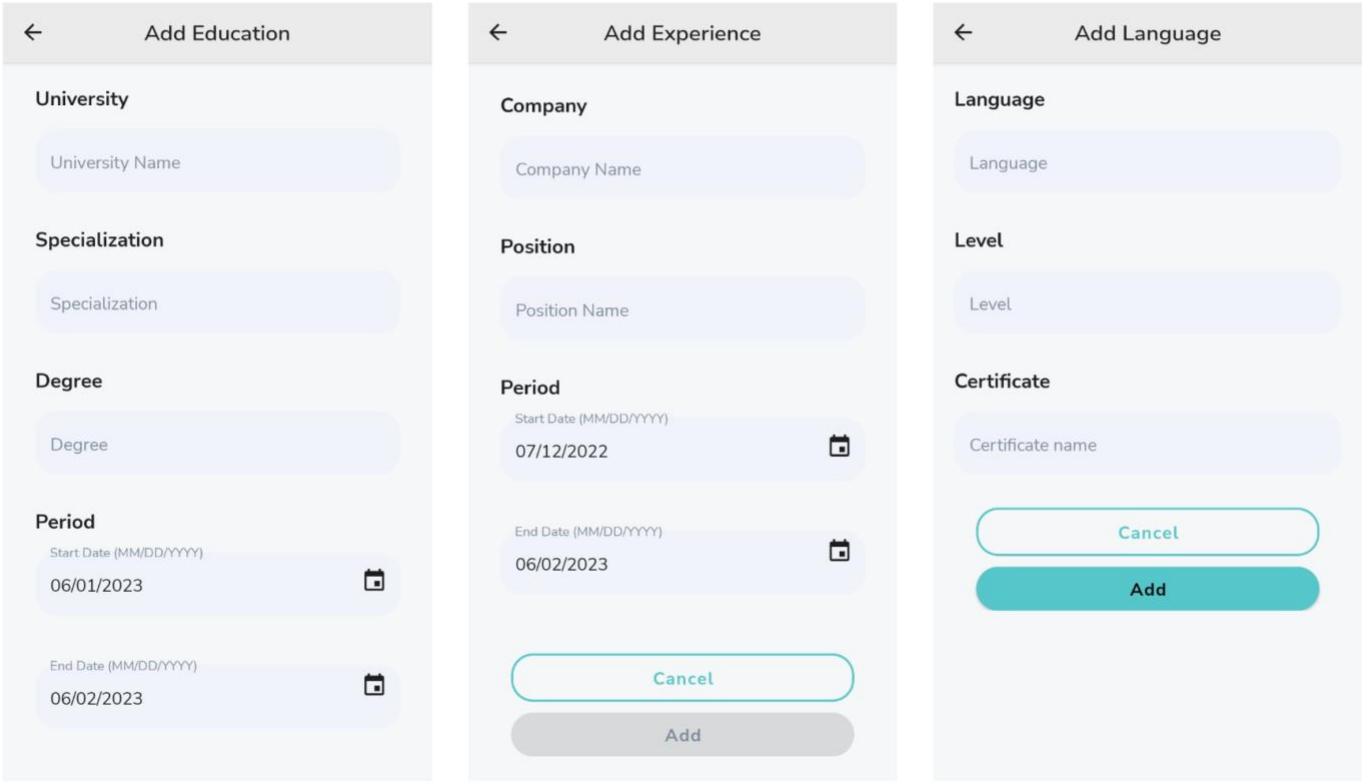
ПРИКЛАД РОБОТИ ЗАСТОСУНКУ. МОБІЛЬНИЙ МОДУЛЬ



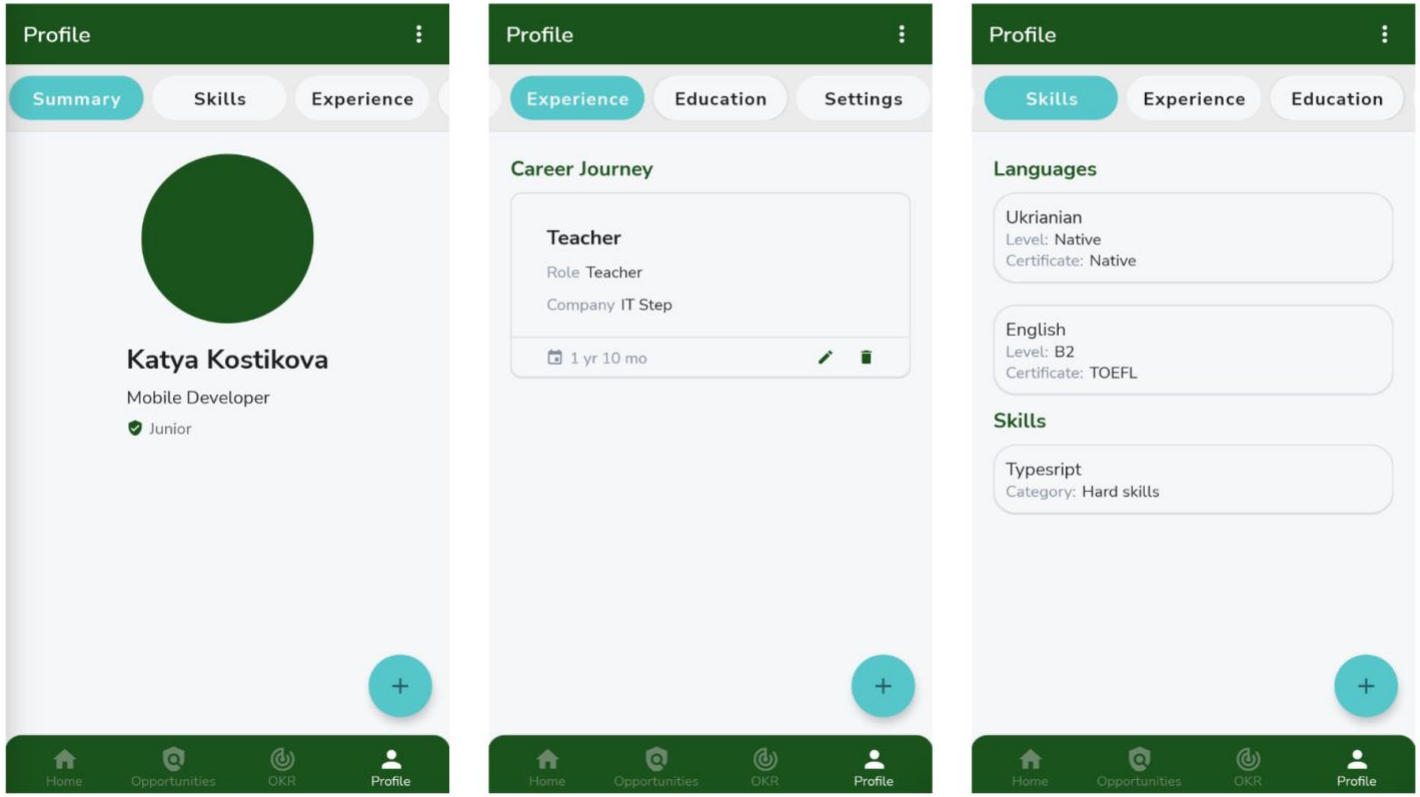
ПРИКЛАД РОБОТИ ЗАСТОСУНКУ. МОБІЛЬНИЙ МОДУЛЬ



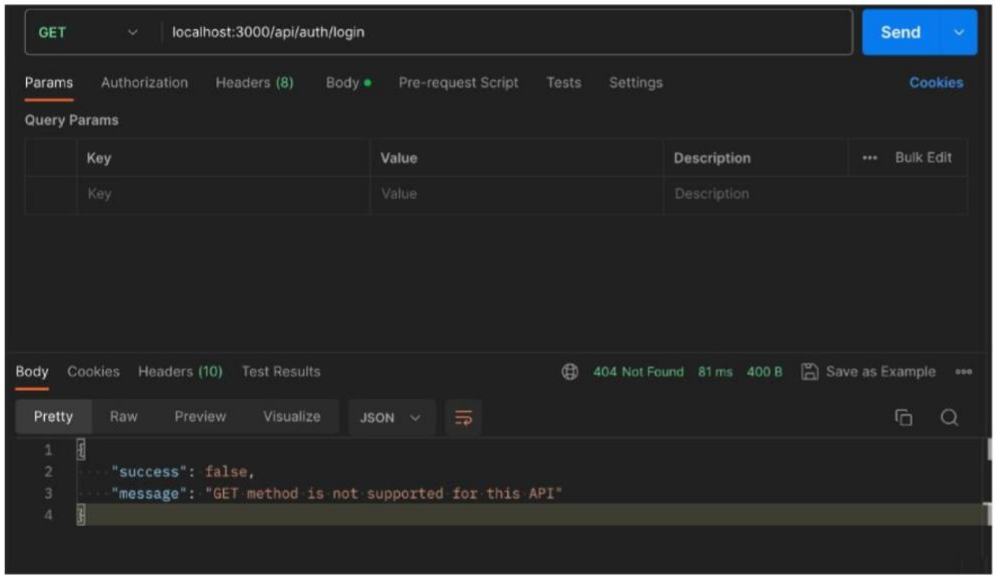
ПРИКЛАД РОБОТИ ЗАСТОСУНКУ. МОБІЛЬНИЙ МОДУЛЬ

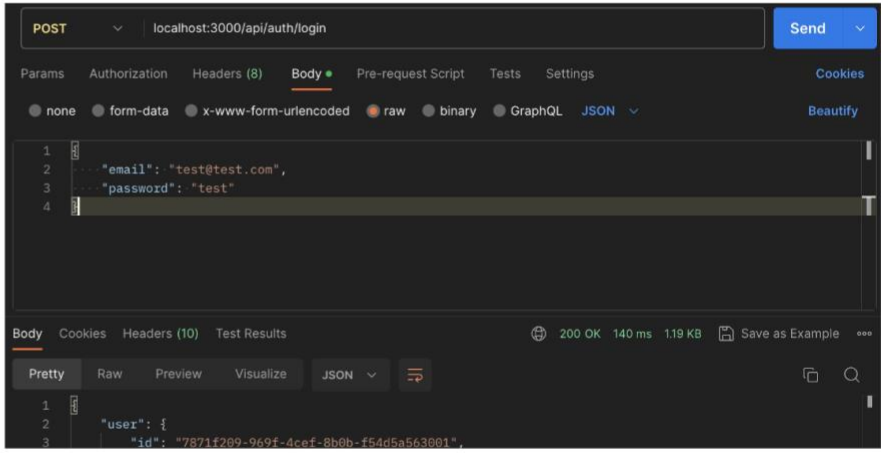


ПРИКЛАД РОБОТИ ЗАСТОСУНКУ. МОБІЛЬНИЙ МОДУЛЬ



ТЕСТУВАННЯ ЗАСТОСУНКУ. СЕРВІС POSTMAN





ПОРІВНЯННЯ З АНАЛОГАМИ

В кінцевому результаті розроблено програмний застосунок який:

- Має особистий профіль співробітника з елементами резюме, що допоможе, за необхідністю, обрати нову сферу діяльності для співробітника в компанії.
- Застосунок орієнтований на індивідуальний підхід до розвитку компетенцій співробітників.
- Надано вибір персональних можливостей для розвитку компетенцій, базуючись на уміннях та навичках співробітника.

НАПРЯМИ РОЗВИТКУ

У наступній версії застосунку буде реалізовано такі вимоги:

1. Функціональні вимоги ментора.
2. Доступ користувачів та менторів до мапи кар'єрного розвитку.
3. Можливість відслідковування розвитку для співробітників.

Рекомендації щодо подальшого розширення та покращення застосунку:

1. Розширення функціоналу для ролі Адмін.
2. Розширення мобільного модуля поточним функціоналом Адміну.
3. Додати мобільні сповіщення.
4. Додати чат для спілкування Ментора і Співробітника.
5. Додати інтеграцію з іншими програмними додатками.

ВИСНОВКИ

1. Було проведено аналіз існуючих конкурентів аналогів.
2. Застосунок розподілено на 3 модуля: мобільний, брузерний, серверний.
3. Описано архітектуру програмного застосунку та схему бази даних.
4. Розроблено ПЗ відповідно вимог безпеки та стабільності.
5. Протестовано запити до серверу та проведено функціональне тестування програмного застосунку.
6. Надано рекомендації подальшого розвитку програмного застосунку.

Розроблений програмний застосунок виконаний в повному обсязі, відповідає всім зазначеним вище функціональним та нефункціональним вимогам.

ПЕРЕВІРКА НА ПЛАГІАТ

UNICHECK

Ім'я користувача:
Люшенко Леся Анатоліївна gmail

ID перевірки:
1015547628

Дата перевірки:
11.06.2023 15:10:51 EEST

Тип перевірки:
Doc vs Internet + Library

Дата звіту:
11.06.2023 19:42:09 EEST

ID користувача:
91603

Назва документа: КП-91_Костікова

Кількість сторінок: 65 **Кількість слів:** 8143 **Кількість символів:** 64002 **Розмір файлу:** 1.94 MB **ID файлу:** 1015200037

Виявлено модифікації тексту (можуть впливати на відсоток схожості)

4.83%
Схожість

Найбільша схожість: 0.92% з джерелом з Бібліотеки (ID файлу: 1008269996)

2.1% Джерела з Інтернету	97	Сторінка 67
4.48% Джерела з Бібліотеки	209	Сторінка 67

ДЯКУЮ ЗА УВАГУ!

Факультет прикладної математики

Кафедра програмного забезпечення комп'ютерних систем

"ЗАТВЕРДЖЕНО"

Завідувач кафедри

_____ Євгенія СУЛЕМА

"___"_____ 2022 р.

ПРОГРАМНИЙ ЗАСТОСУНОК «INNER GROWTH» ДЛЯ УПРАВЛІННЯ РОЗВИТКОМ КОМПЕТЕНЦІЙ СПІВРОБІТНИКІВ

Програма та методика тестування

ДП.045440-04-51

"ПОГОДЖЕНО"

Керівник проєкту:

_____ Леся ЛЮШЕНКО

Нормоконтроль:

_____ Микола ОНАЙ

Виконавець:

_____ Катерина КОСТІКОВА

2022

ЗМІСТ

1. ОБ’ЄКТ ВИПРОБУВАНЬ

Програмний застосунок для управління розвитком фахових та поведінкових компетенцій співробітників. Складається з вебзастосунку, серверна частина якого розроблена з використанням фреймворку Express.js на основі платформи Node.js, а клієнтська – з використанням фреймворку React. Також складається з мобільного застосунку написаного з використанням фреймворку React Native.

2. МЕТА ТЕСТУВАННЯ

В процесі тестування має бути перевірене наступне:

1) відповідність функціональних вимог, реалізованих у програмному застосунку до заявлених;
2) відповідність застосунку безпековим вимогам;
3) зручність роботи з мобільним застосунком;
4) зручність роботи з вебзастосунком.

3. МЕТОДИ ТЕСТУВАННЯ

Метод White Box Testing використовується для проведення тестування на рівні «системного тестування». Цей метод дозволяє перевірити як поведінку програмного застосунку, так і код, забезпечуючи перевірку кожного модуля програми та зв'язків між ними.

Використовуються наступні методи:

1) Для функціонального тестування використовується метод Critical Path Test (тестування критичного шляху). Цей метод дозволяє перевірити правильність функціонування застосунку, перевіряючи критичні шляхи або послідовності операцій.
2) Для тестування продуктивності програмного застосунку використовуються методи Load Testing (навантажувальне

тестування) і Stress Testing (стрес-тестування). Load Testing дозволяє визначити, як програмний застосунок працює під навантаженням, перевіряючи його продуктивність і стабільність. Stress Testing використовується для перевірки межевих умов програми, шляхом навантаження системи до максимального обсягу або перевищення ресурсів.

3) Також проводиться тестування інтерфейсу, щоб перевірити правильність взаємодії програмного продукту з користувачем.

4. ЗАСОБИ ТА ПОРЯДОК ТЕСТУВАННЯ

Коректність роботи програмного застосунку тестується з використанням таких методів та утиліт:

1) перевірка на валідність шляхом введення недопустимих значень у редаговані поля;
2) ручна перевірка відповідності функціональних вимог до заявлених;
3) тестування з максимальним навантаженням серверу за допомогою утиліти Apache JMeter;
4) випробування стабільності роботи за різними показниками з використанням утиліти Apache JMeter;
5) тестування вебзастосунку в різних браузерах для перевірки сумісності;
6) тестування мобільного застосунку на різних ОС та девайсах;
7) тестування запитів до серверу за допомогою сервісу Postman.

Факультет прикладної математики

Кафедра програмного забезпечення комп'ютерних систем

"ЗАТВЕРДЖЕНО"

Завідувач кафедри

_____ Євгенія СУЛЕМА

"___"_____ 2023 р.

ПРОГРАМНИЙ ЗАСТОСУНОК «INNER GROWTH» ДЛЯ УПРАВЛІННЯ РОЗВИТКОМ КОМПЕТЕНЦІЙ СПІВРОБІТНИКІВ

Керівництво користувача

ДП.045440-05-34

"ПОГОДЖЕНО"

Керівник проєкту:

_____ Леся ЛЮШЕНКО

Нормоконтроль:

_____ Микола ОНАЙ

Виконавець:

_____ Катерина КОСТІКОВА

2023

ЗМІСТ

1. Опис структури програмного застосунку

Програмний застосунок для управління розвитком компетенцій співробітників складається з мобільного застосунку та вебзастосунку. Програмний застосунок виконаний англійською мовою.

Вебзастосунок містить наступні сторінки:

- «Sign in»;
- «Sign up»;
- «Your company»;
- «Users»;
- «Career path»;
- «Connect career path».

Перехід між сторінками відбувається натисканням відповідних кнопок на верхній навігаційній панелі.

Мобільний застосунок складається з таких сторінок:

- «Sign in»;
- «Sign up»;
- «Home»;
- «Opportunity Details»;
- «Opportunities»;
- «OKR»;
- «Profile»;
- «Add New Objective»;
- «Profile»;
- «Onboarding»;
- «Add Skill»;
- «Onboarding»;
- «Add Education»;
- «Add Language»;

- «Add Experience».

Перехід між сторінками відбувається натисканням відповідних кнопок на нижній навігаційній панелі.

2. Процедура авторизації адміністратора та співробітника

Сторінка вебзастосунку «Sign in» відкривається за замовчуванням для неавторизованих адміністраторів (рис. 1). Сторінка містить форму для вводу необхідних для авторизації даних: електронної пошти та пароля. Також міститься чек бокс для збереження даних про акаунт «Remember me», та кнопка відновлення паролю «Forgot password». Після введення коректних даних та натиснення кнопки «Sign in», адміністратор увійде в систему. Також, на сторінці міститься посилання на сторінку «Sign up».

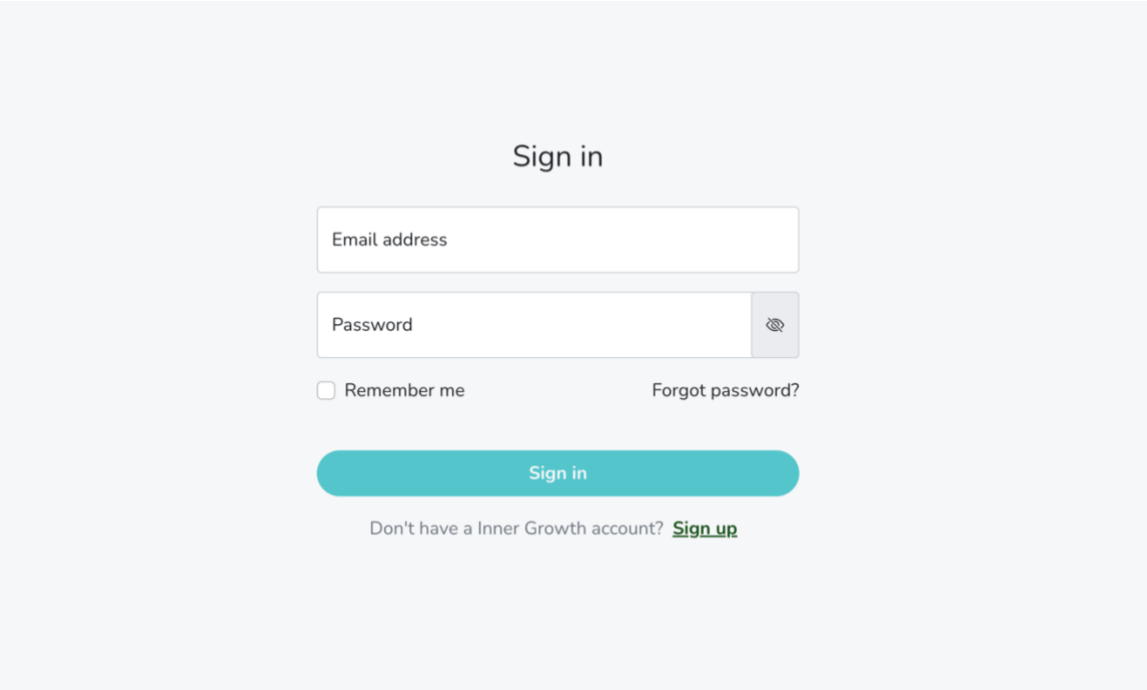


Рис. 1. Сторінка «Sign in» вебзастосунку

Сторінка мобільного застосунку «Sign in» відкривається також за замовчуванням для неавторизованих співробітників (рис. 2). Сторінка містить форму для вводу електронної пошти та пароля. Після введення коректних даних та натиснення кнопки «Log In», співробітник увійде в застосунок. Також, на сторінці міститься посилання на сторінку «Sign Up» для адміністраторів.

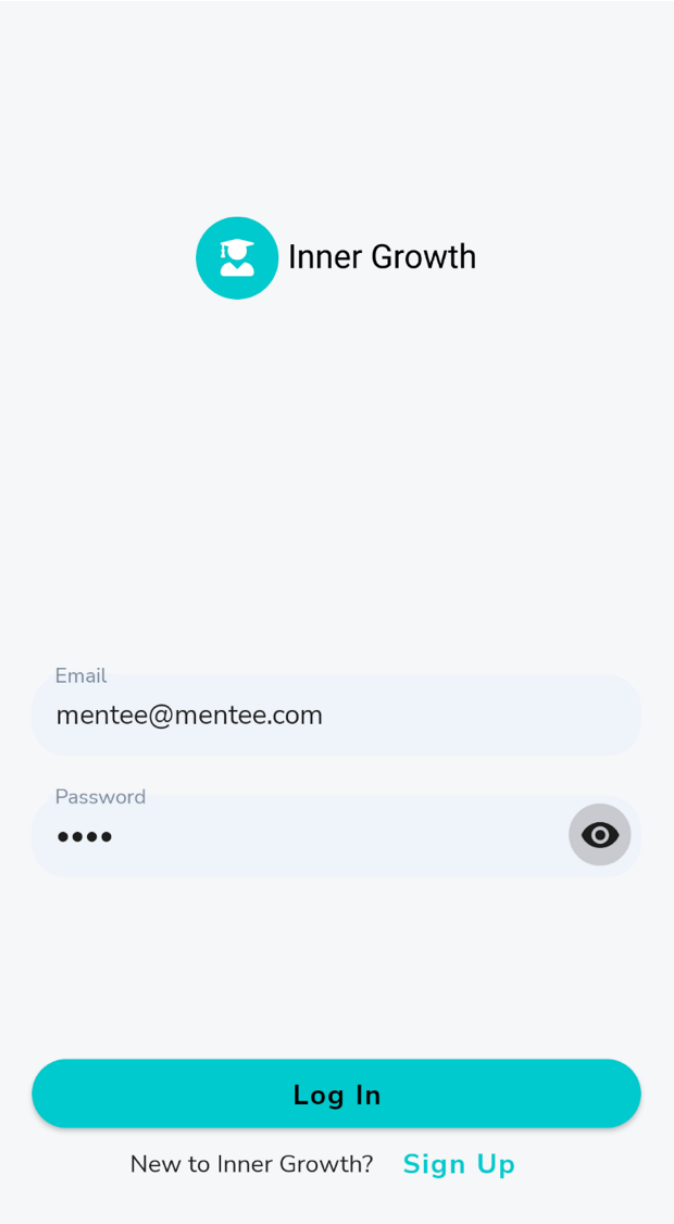


Рис. 2. Сторінка «Sign in» мобільного застосунку

3. Процедура реєстрації адміністратора

Сторінка вебзастосунку «Sign up» містить форму для вводу необхідних для реєстрації адміністратора даних: електронної пошти, ім’я, прізвища та пароля (рис. 3). Після введення коректних даних та натиснення кнопки «Sign up», адміністратор увійде в систему. Також, на сторінці міститься посилання на сторінку «Sign in».

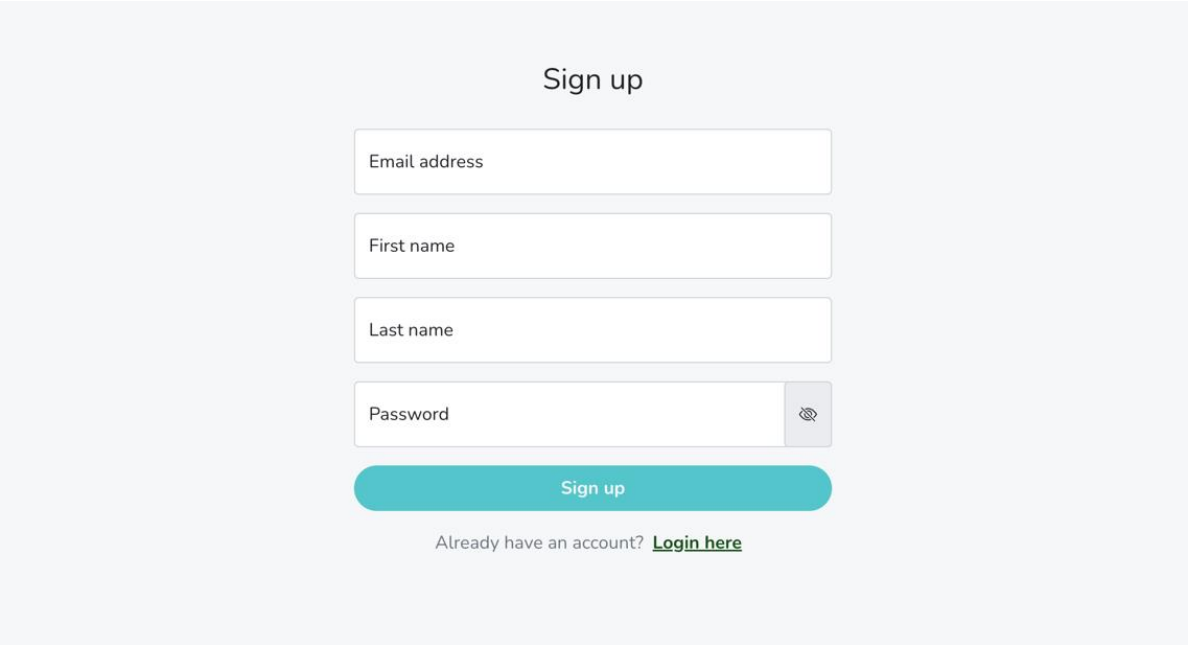


Рис. 3. Сторінка «Sign up» вебзастосунку

Сторінка мобільного застосунку «Sign Up» містить форму для вводу електронної пошти, прізвища, ім’я та пароля (рис. 4). Після введення коректних даних та натиснення кнопки «Sign Up», адміністратор зареєструється в застосунку. Також, на сторінці міститься посилання на сторінку «Sign In».

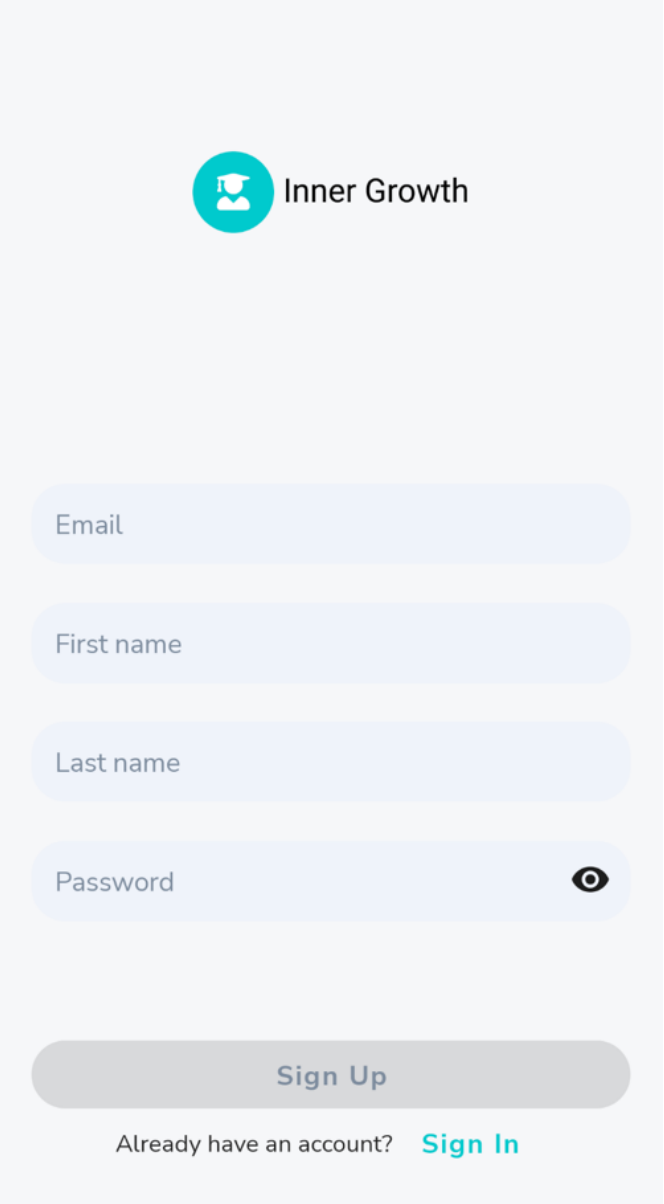


Рис. 4. Сторінка «Sign Up» мобільного застосунку

4. Сторінка «Your company» вебзастосунку для редагування даних про компанію

Сторінка вебзастосунку «Your company» містить блок з інформацією про назву, опис та фото компанії. Містить кнопку «Edit company» для редагування інформації про компанію. Також містить блок з тегами компанії, необхідними для створення подій, можливостей для розвитку. Містить кнопку «Add Tag» для додавання нових тегів (рис. 5).

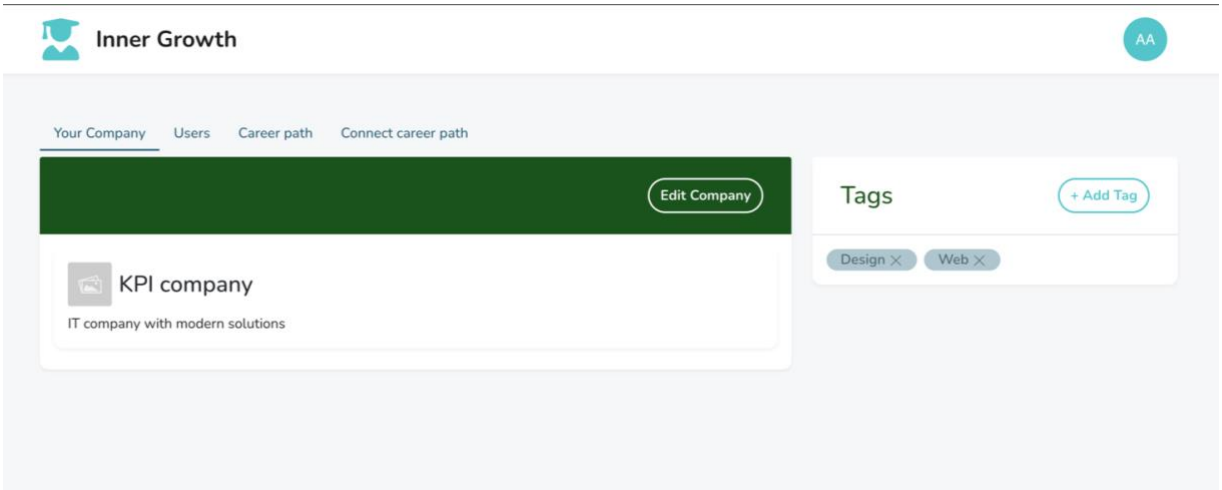


Рис. 5. Сторінка «Your company» вебзастосунку

При натисканні на кнопку «Edit company» відкривається модальне вікно з полями для додавання чи редагування інформації про компанію (рис. 6). Поля містять такі дані: файл логотипу компанії, назва та опис компанії. Поле назви компанії є обов’язковими, а поле логотипу та опису заповнюється за бажанням адміністратора. При обиранні логотипу компанії користувач має обрати файл зі свого персонального комп’ютера. Після введення коректних даних користувач натискає на кнопку «Save» та повертається назад на сторінку «Your company».

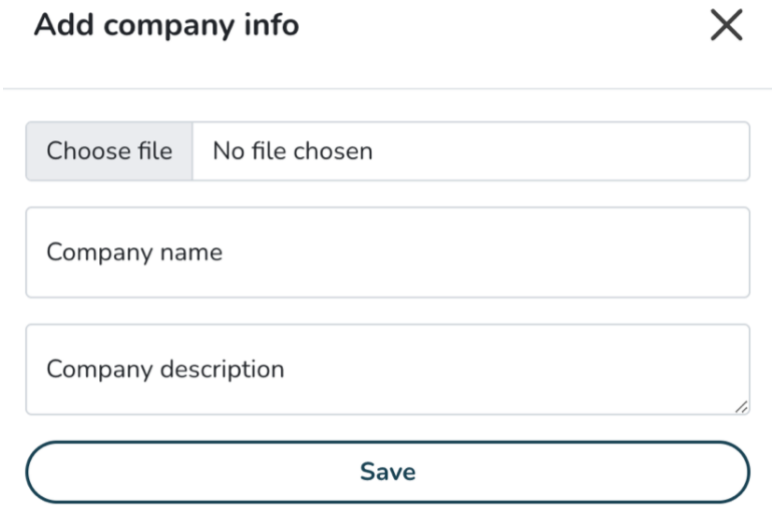


Рис. 6. Модальне вікно вебзастосунку для редагування компанії

При натисканні на кнопку «Add tags» відкривається модальне вікно з полем для додавання чи редагування тегів (рис. 7). Поле містить назву тега. Після натискання кнопки «Add» тег буде додано в перелік тегів нижче. Після додавання необхідної кількості тегів користувач натискає на кнопку «Save» та повертається назад на сторінку «Your company».

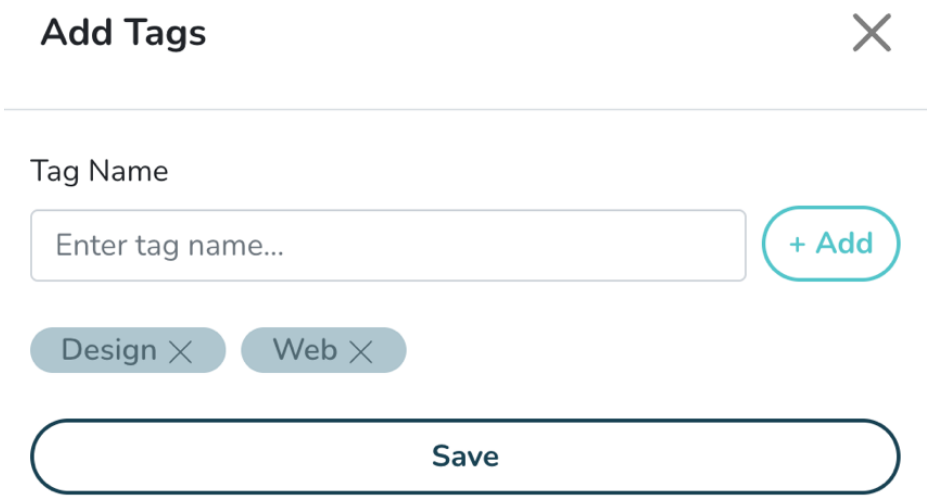


Рис. 7. Модальне вікно вебзастосунку для редагування тегів

5. Сторінка «Users» вебзастосунку для додавання користувачів

Сторінка вебзастосунку «Users» містить таблицю з інформацією про користувачів компанії (ментори, співробітники). Таблиця містить такі дані: ім’я та прізвище користувача, його електронна адреса, роль та посада (рис. 8). Користувач має завершити процедуру реєстрації в програмному застосунку для того, щоб з’явилася інформація про його чи її прізвище та ім’я. Також таблиця містить кнопки для редагування та видалення користувача. Над таблицею міститься кнопка «Add User» для додавання нового користувача. Доступне поле вводу для пошуку користувача за будь-яким полем.

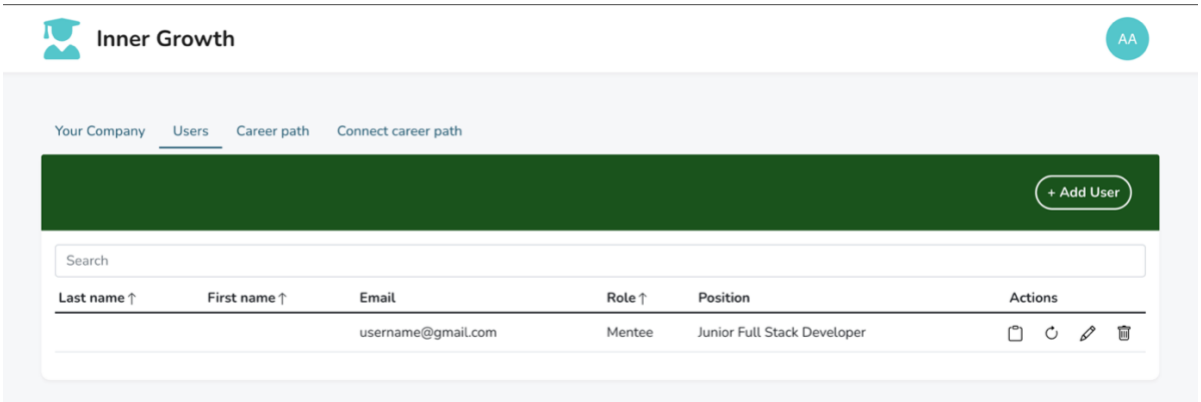


Рис. 8. Сторінка «Users» вебзастосунку

При натисканні на кнопку «Add user» відкривається модальне вікно (рис. 9). Модальне вікно містить поля: електронна адреса, роль користувача, посада та рівень фахової компетенції. Рівень фахової компетенції обирається із існуючих для компанії. Додати компетенцію можна на сторінці «Career Path». Після введення коректних даних адміністратор натискає на кнопку «Save» та повертається назад на сторінку «Users».

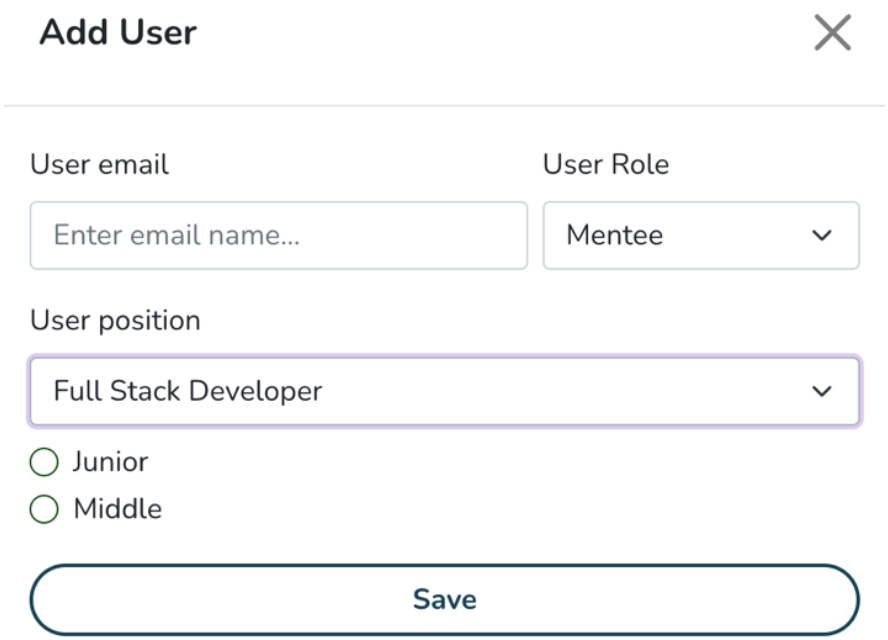


Рис. 9. Модальне вікно вебзастосунку для додавання користувача

6. Сторінка «Career path» вебзастосунку для додавання фахових компетенцій

Сторінка вебзастосунку «Career path» містить перелік фахових компетенцій. Фахові компетенції в свою чергу містять доступні для цих компетенцій рівні, та перелік навичок та технологій необхідних для засвоєння рівня (рис. 10). На початку переліку є поле для вводу назви компетенції. Після введення назви адміністратор натискає кнопку «Add domain» та компетенція додається до переліку.

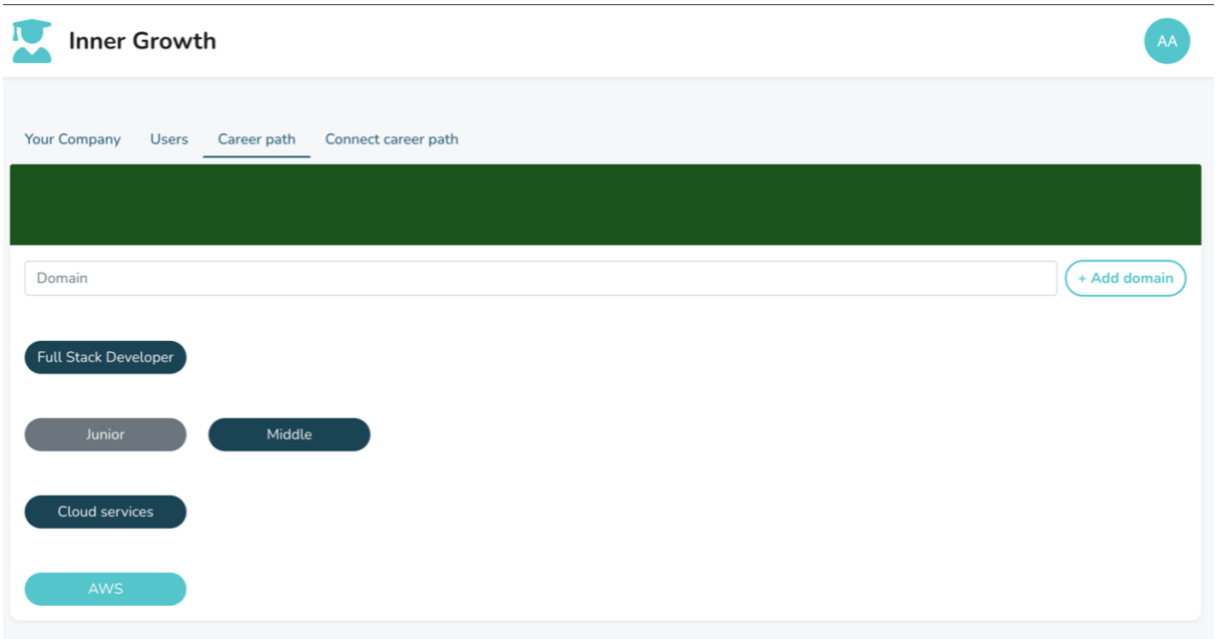


Рис. 10. Сторінка «Career Path» вебзастосунку

При наведенні мишкою на будь-яке поле у переліку з’являється випадаюче меню дій (рис. 11). При натисканні на кнопку зі стрілочкою адміністратор може створити рівень або навичку для цієї фахової компетенції. При натисканні на олівець користувач може відредагувати компетенцію, та при натисканні на корзину видалити її.

Рис. 11. Меню дій фахової компетенції

При подвійному натисканні на будь-яке поле у переліку, його можна відредагувати. Після редагування адміністратор може натиснути кнопку з галочкою для збереження змін, або хрестик для скасування дії (рис. 12).

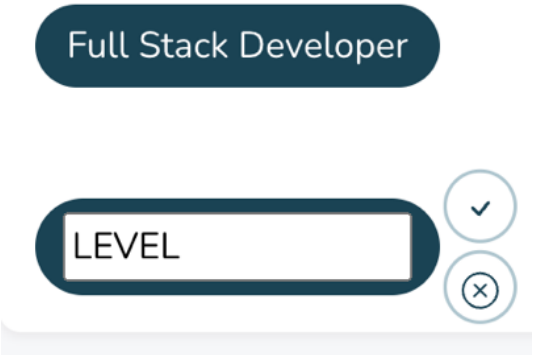


Рис. 12. Редагування полей фахової компетенції

7. Сторінка «Connect career path» вебзастосунку для створення мапи кар’єрного розвитку

Сторінка вебзастосунку «Connect career path» містить поля для створення направлень у мапі кар’єрного розвитку та напрямки для різних фахових компетенцій (рис. 13). Містить два випадаючих списки, де адміністратор має обрати з якої компетенції в яку співробітник може перекваліфікуватися чи отримати підвищення. Далі адміністратор має обрати рівні з якого на який співробітник може перейти. Далі адміністратор натискає кнопку «Pair levels» та напрямок для мапи кар’єрного розвитку додається до переліку нижче.

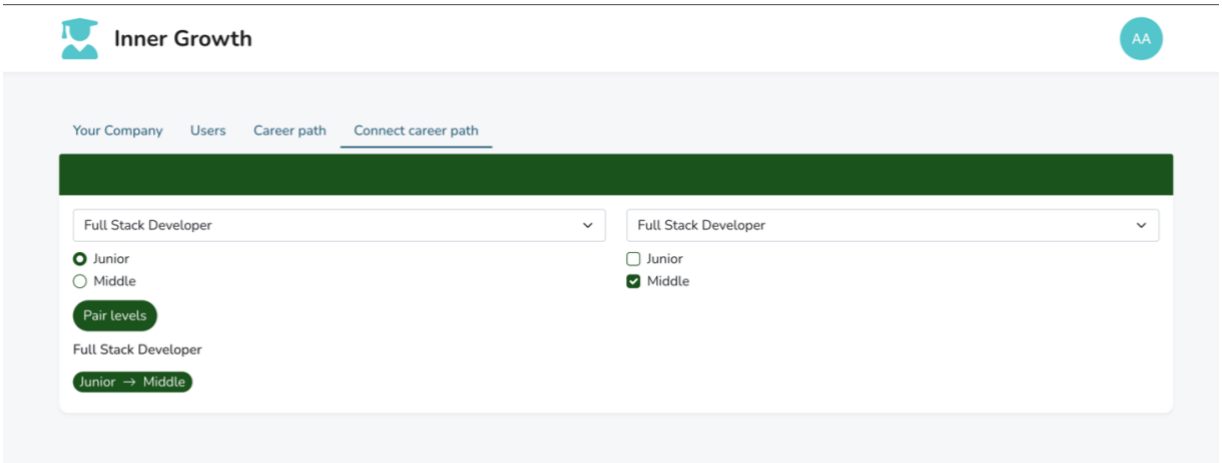


Рис. 13. Сторінка «Connect career Path» вебзастосунку

8. Сторінка «Onboarding» мобільного застосунку

Сторінка мобільного застосунку «Onboarding» доступна користувачу після реєстрації акаунту адміністратором. На ній користувач має завершити етап реєстрації. На першій секції користувачу необхідно надати такі дані: ім’я, прізвище та фото профіля за необхідності (рис. 14). Після введення коректних даних користувач натискає кнопку «Next».

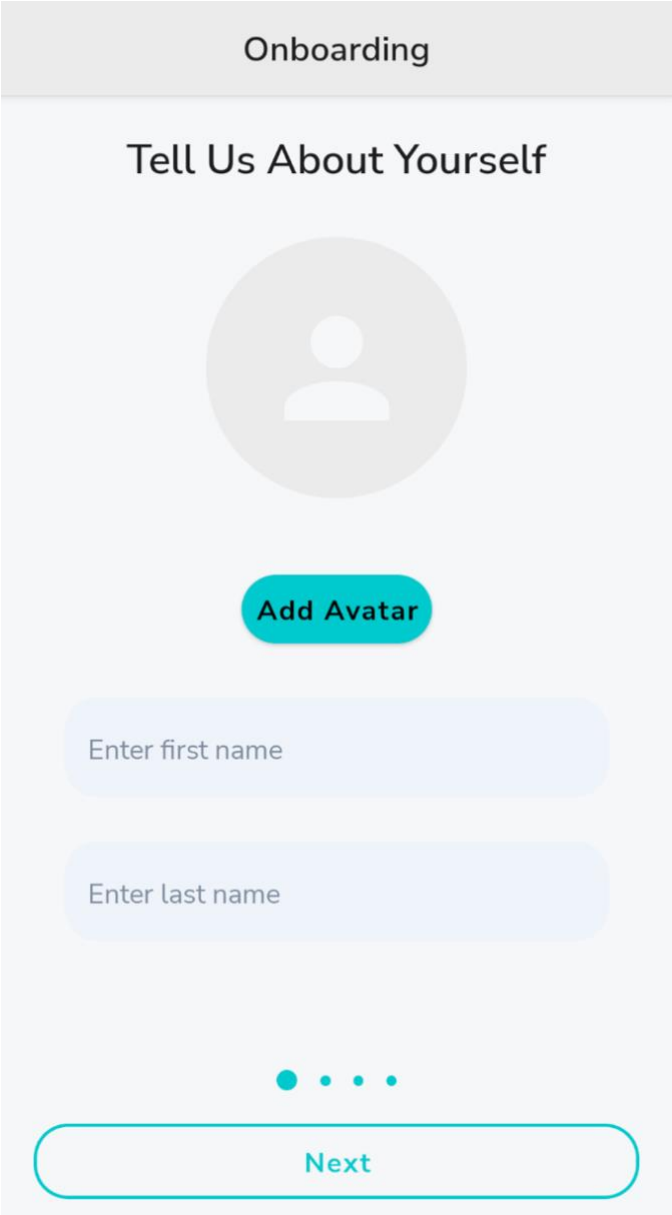


Рис. 14. Сторінка «Onboarding» мобільного застосунку. Секція особистої інформації

Далі користувач потрапляє на секцію даних про досвід роботи. Секція містить дані про досвід у вигляді карток (рис. 15). Картка містить інформацію про посаду, назву компанії та період роботи. Також при натисканні на кнопку у вигляді олівця користувач може відредагувати інформацію про досвід роботи. При натисканні на кнопку корзини користувач може видалити дані про досвід. При натисканні на кнопку «Add Experience» користувач перейде на сторінку для створення картки про досвід роботи. Після введення даних користувач натискає кнопку «Next».

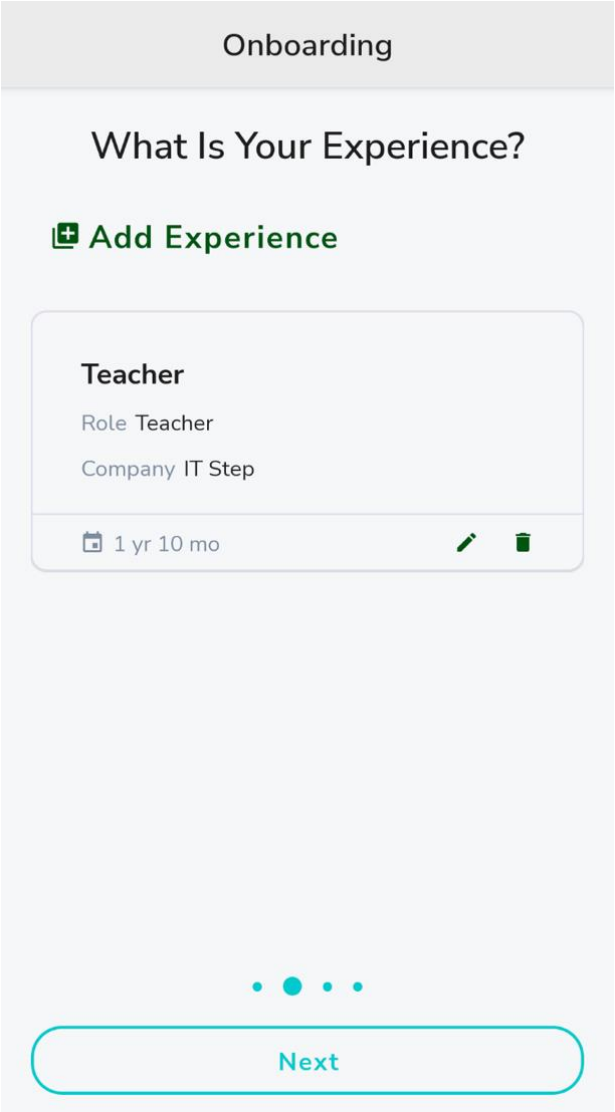


Рис. 15. Сторінка «Onboarding» мобільного застосунку. Секція даних про досвід роботи

Далі користувач потрапляє на секцію даних про освіту. Секція містить дані про освіту користувача у вигляді карток (рис. 16). Картка містить інформацію про спеціальність, назву вищого навчального закладу, освітній рівень та період навчання. Також при натисканні на кнопку у вигляді олівця користувач може відредагувати інформацію про освіту. При натисканні на кнопку корзини користувач може видалити дані про освіту. При натисканні на кнопку «Add Education» користувач перейде на сторінку для створення картки про освіту. Після введення даних користувач натискає кнопку «Next».

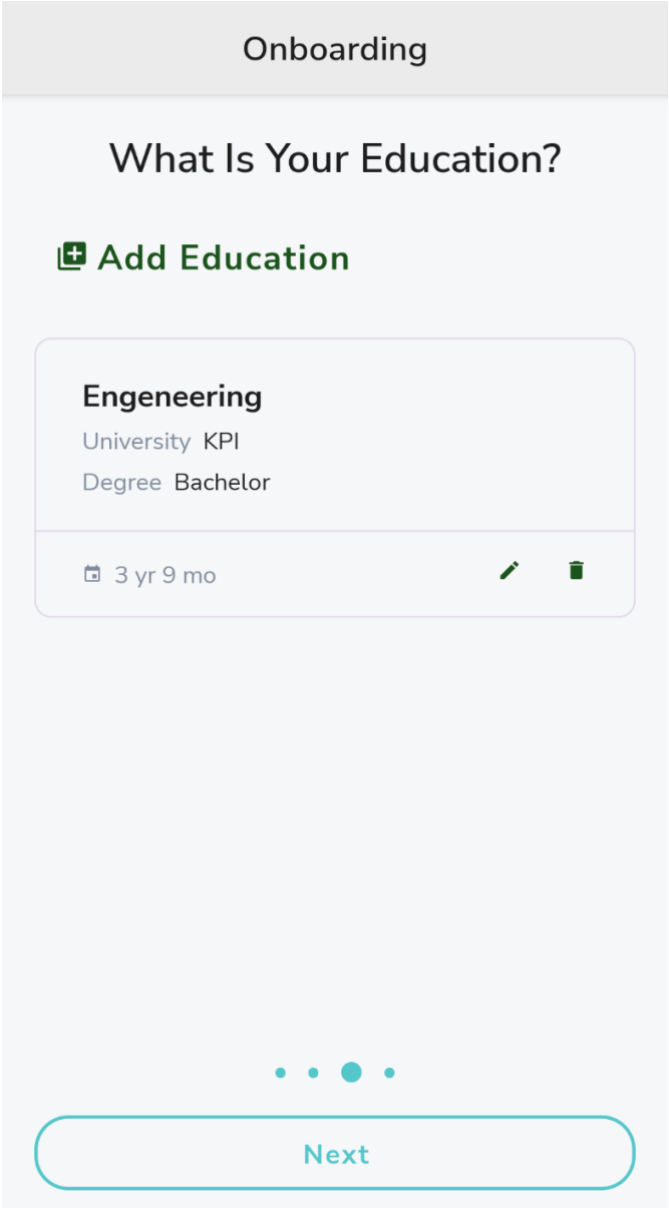


Рис. 16. Сторінка «Onboarding» мобільного застосунку. Секція даних про освіту

Далі користувач потрапляє на секцію даних про мови якими він чи вона володіє. Секція містить дані про мови у вигляді карток (рис. 17). Картка містить інформацію про назву мови, рівень володіння мовою та назву сертифікату. При натисканні на кнопку «Add Language» користувач перейде на сторінку для створення картки про володіння мовою. Після введення даних користувач натискає кнопку «Complete».

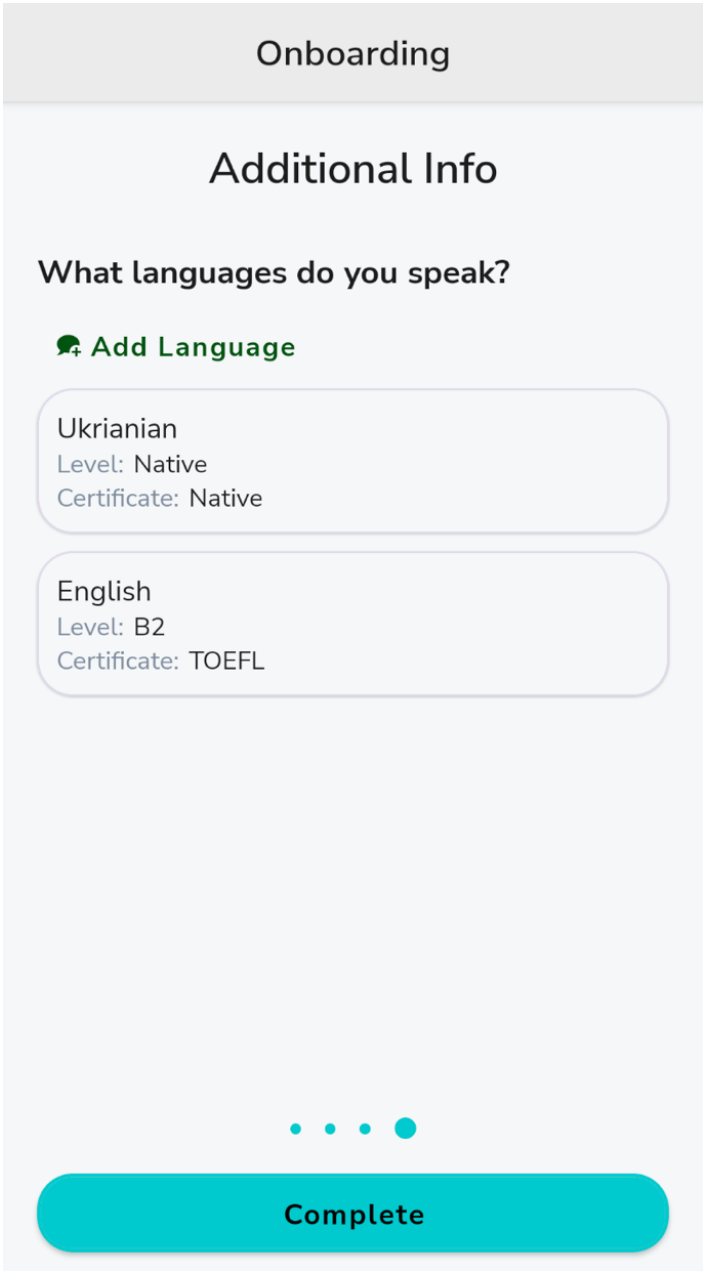


Рис. 17. Сторінка «Onboarding» мобільного застосунку. Секція даних про мови

9. Сторінки «Add Experience», «Add Education» та «Add Language» мобільного застосунку

Як було описано в розділі 8, користувач може додавати дані про освіту, мову та досвід роботи. Розглянемо кожну з сторінок для створення цих даних.

Сторінка «Add Experience» містить такі поля: назва компанії, посада, дата початку та дата кінця роботи (рис. 18). Після введення коректних даних користувач натискає кнопку «Add».

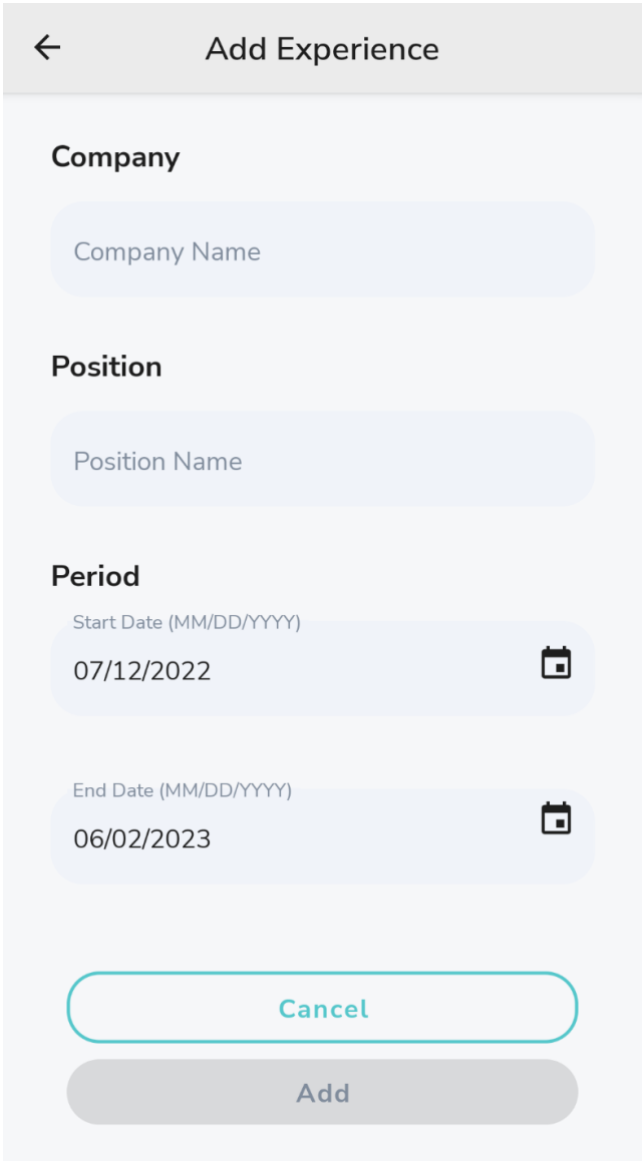


Рис. 18. Сторінка «Add Experience» мобільного застосунку.

Сторінка «Add Education» містить такі поля: назва вищого навчального закладу, спеціальність, освітній рівень, дата початку та дата кінця навчання (рис. 19). Після введення коректних даних користувач натискає кнопку «Add».

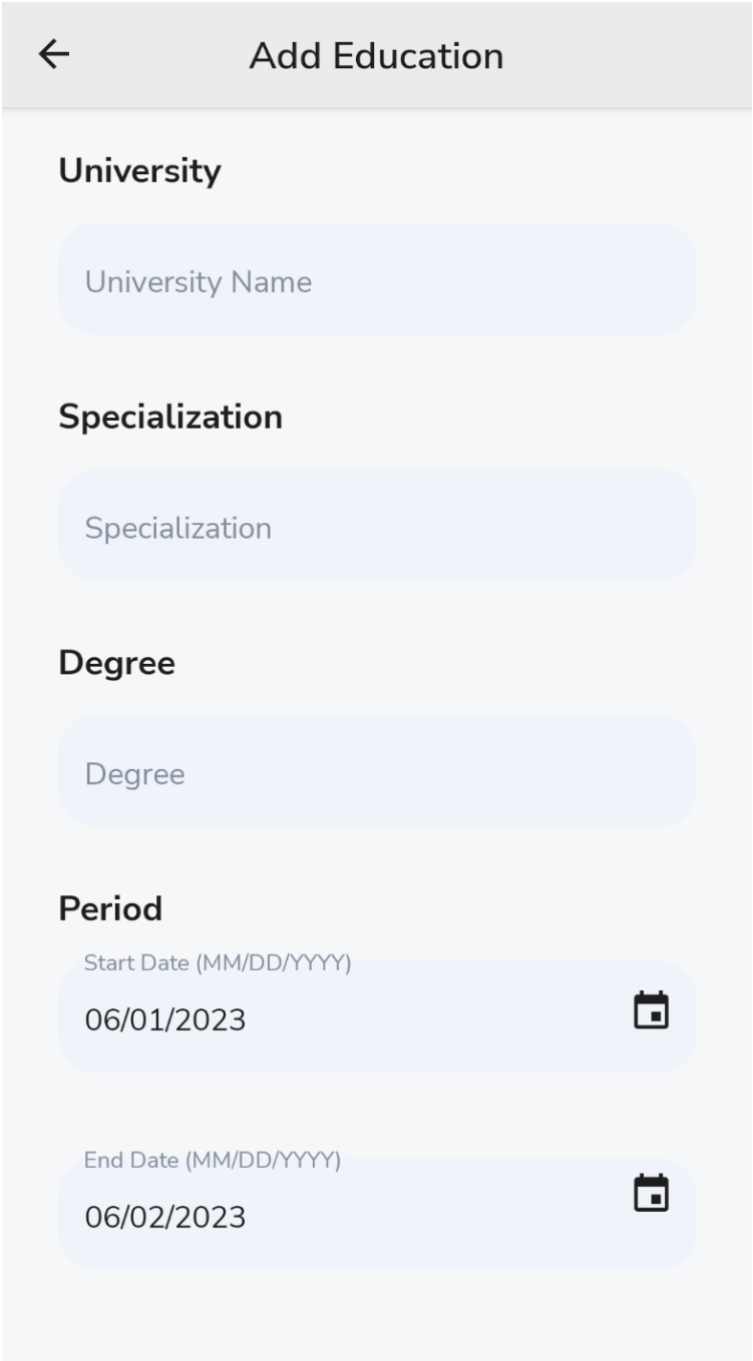


Рис. 19. Сторінка «Add Education» мобільного застосунку.

Сторінка «Add Language» містить такі поля: назва мови, рівень володіння мовою та назва сертифікату (рис. 20). Після введення коректних

даних користувач натискає кнопку «Add».

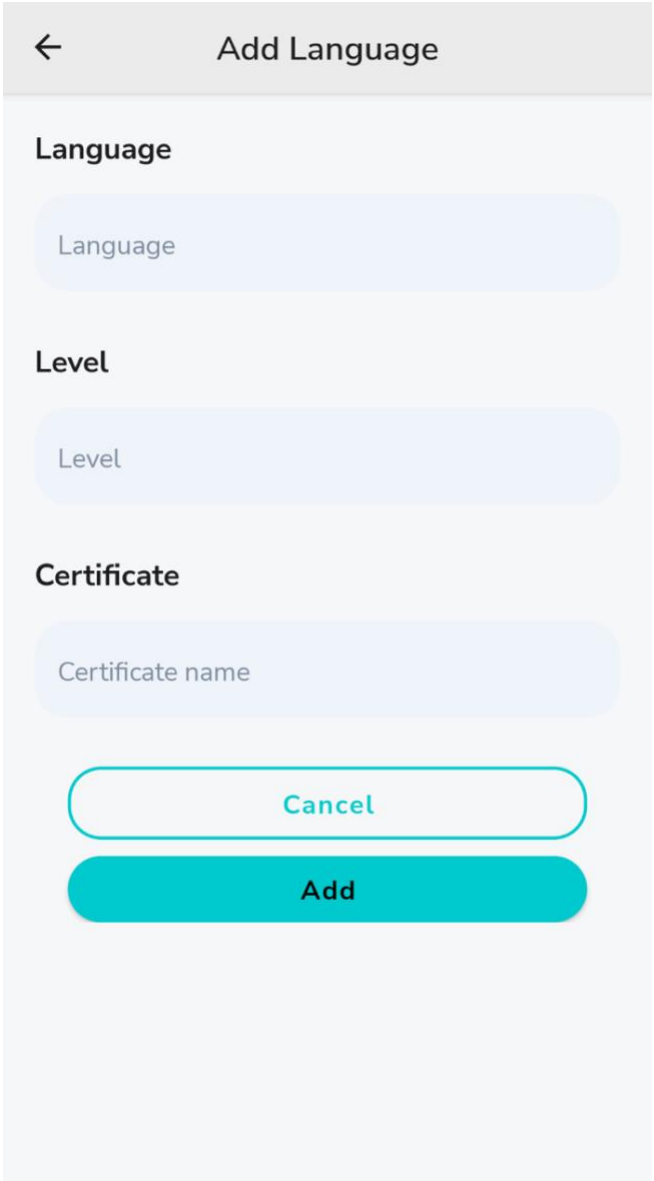


Рис. 20. Сторінка «Add Language» мобільного застосунку.

10. Сторінка «Home» мобільного застосунку

Сторінка «Home» містить дві секції: повідомлення користувача, рекомендовані події та можливості для розвитку (рис. 21). Секція повідомлень містить перелік нових повідомлень про просування по мапі кар'єрного розвитку, нові рекомендації подій тощо. При натисканні на кнопку «Got it» повідомлення помічається як прочитане та зникає зі списку. Секція рекомендованих подій містить перелік подій у вигляді карток. Подія містить таку інформацію: назва події, організаційна група, тип події, дата події та теги. При натисканні на кнопку закладки, користувач збереже подію. При натисканні на кнопку «Details» користувач перейде на сторінку детальної інформації про подію.

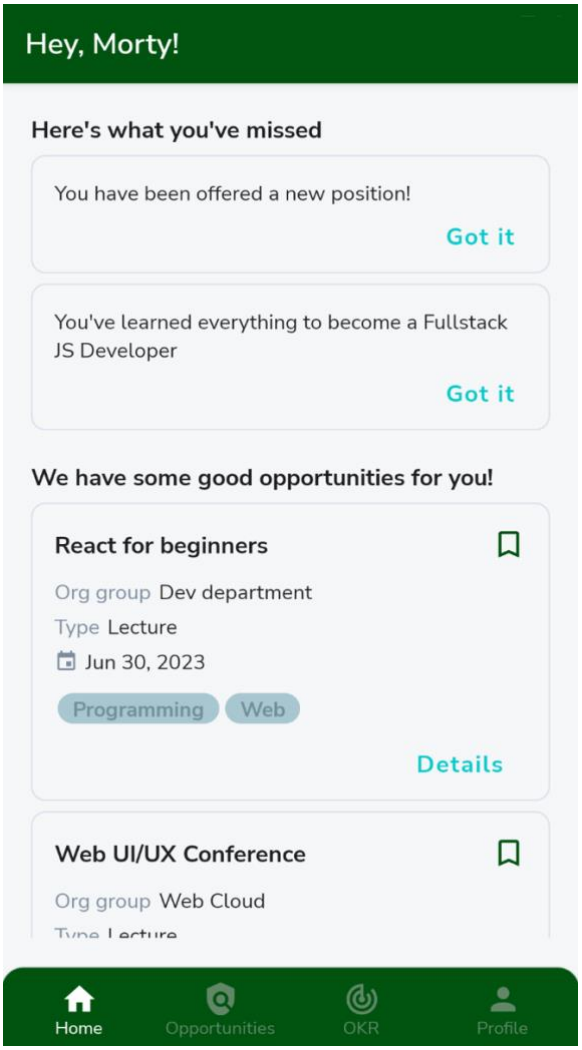


Рис. 21. Сторінка «Home» мобільного застосунку.

11. Сторінки «Opportunities» та «Opportunity Details» мобільного застосунку

Сторінка «Opportunities» містить перелік всіх подій та можливостей розвитку в компанії (рис. 22). Картки подій репрезентовано так само як і в відповідній секції описаній на сторінці «Home» у розділі 10.

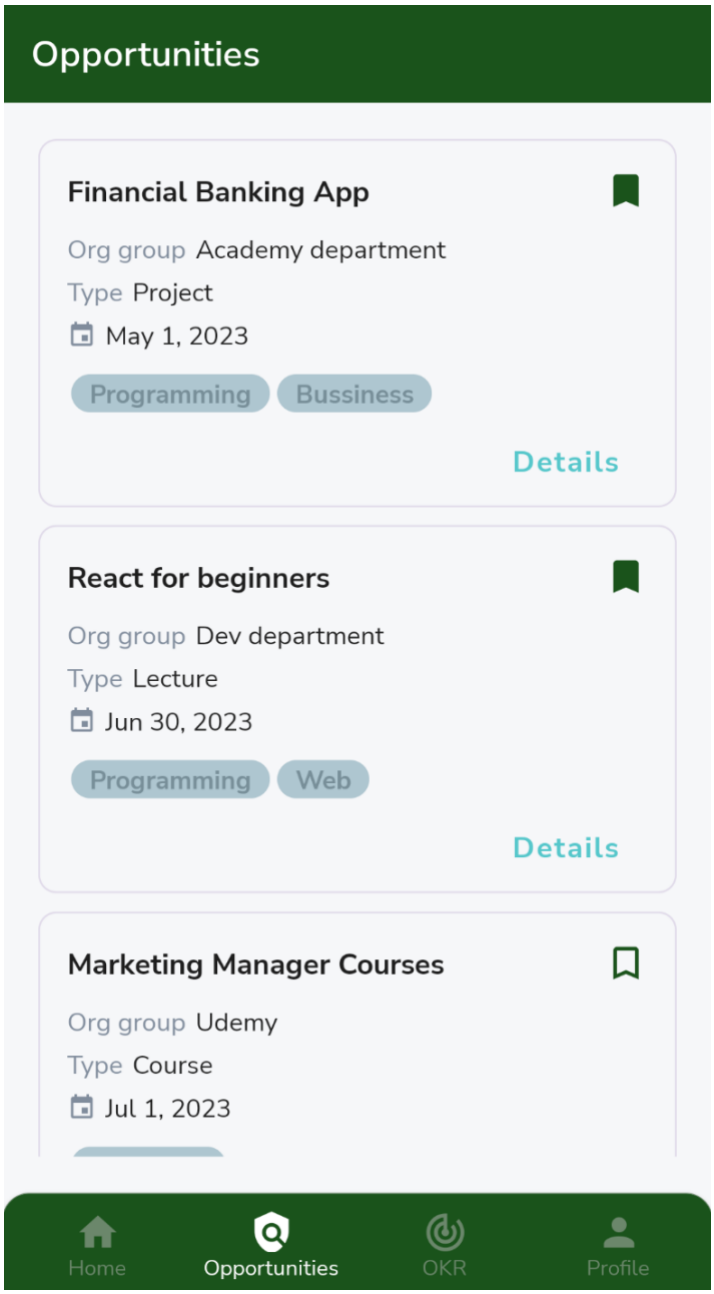


Рис. 22. Сторінка «Opportunities» мобільного застосунку.

Сторінка «Opportunity Details». Містить інформацію про подію, а

саме: назва події, опис події, теги які відповідають тематиці події, та інформація про користувача який її організовує (рис. 23).

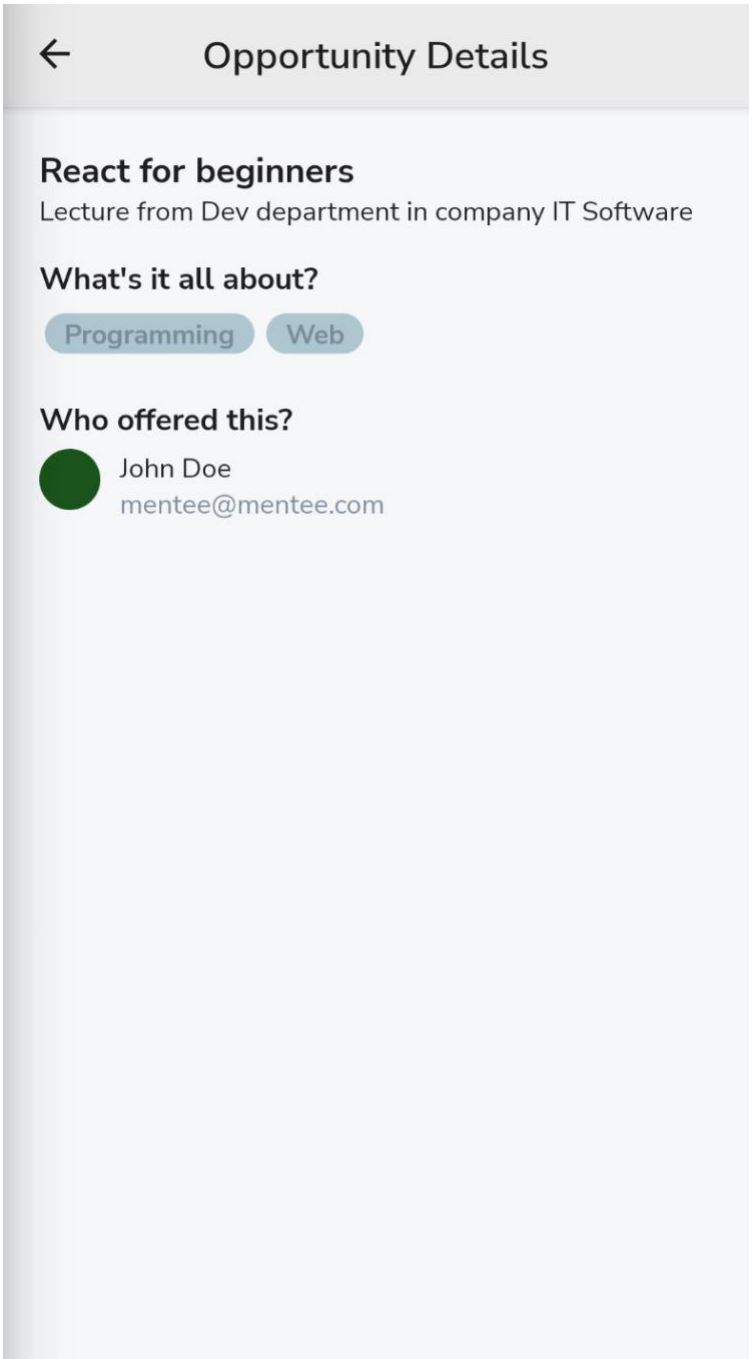


Рис. 23. Сторінка «Opportunity Details» мобільного застосунку.

12. Сторінки «OKR», «Add Objective» та «Add Key Result» мобільного застосунку

Сторінка «OKR» містить перелік всіх циклів OKR користувача. Цикл OKR містить таку інформацію: назва циклу, тип OKR та перелік ключових результатів з прогресом їх виконання (рис. 24).

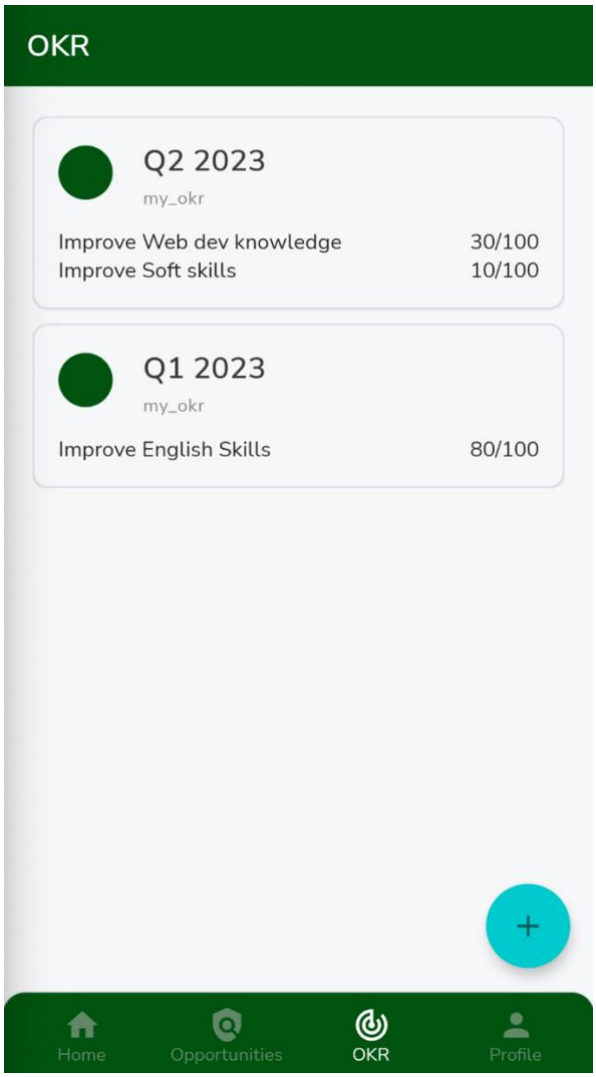


Рис. 24. Сторінка «OKR» мобільного застосунку.

При натисканні на кнопку з плюсом, у користувача буде можливість обрати який тип циклу OKR користувач хоче створити, а саме: Особистий OKR чи командний (рис. 25).

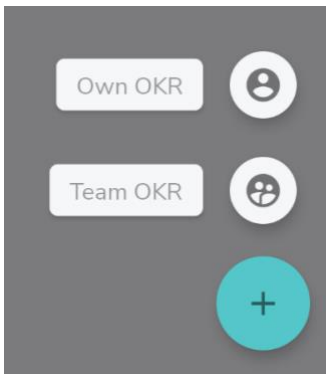


Рис. 25. Кнопка додавання нового циклу OKR.

Сторінка «Add Objective» містить такі поля: ціль, цикл OKR, батьківська ціль (ціль фахової компетенції) та перелік ключових результатів (рис. 26). Якщо це командний OKR, то користувач також має обрати назву команди, для якої буде створено цикл OKR. Користувач натискає кнопку «Add Key Result» для додавання ключового результату. Після введення коректних даних користувач натикає кнопку «Save».

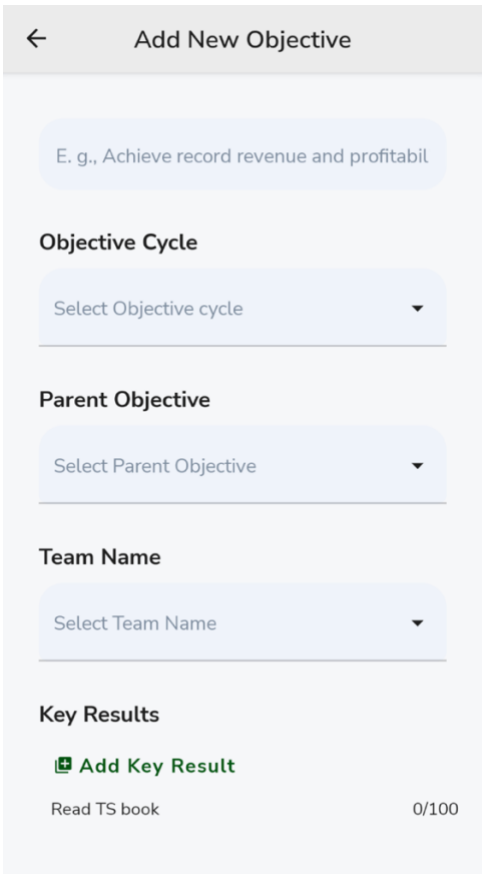


Рис. 26. Сторінка «Add Objective» мобільного застосунку.

Сторінка «Add Key Result» містить поле назви ключового результату та селектор поточного рівня досягнення вказаного результату (рис. 27). Після введення коректних даних користувач натикає кнопку «Add Key Result».

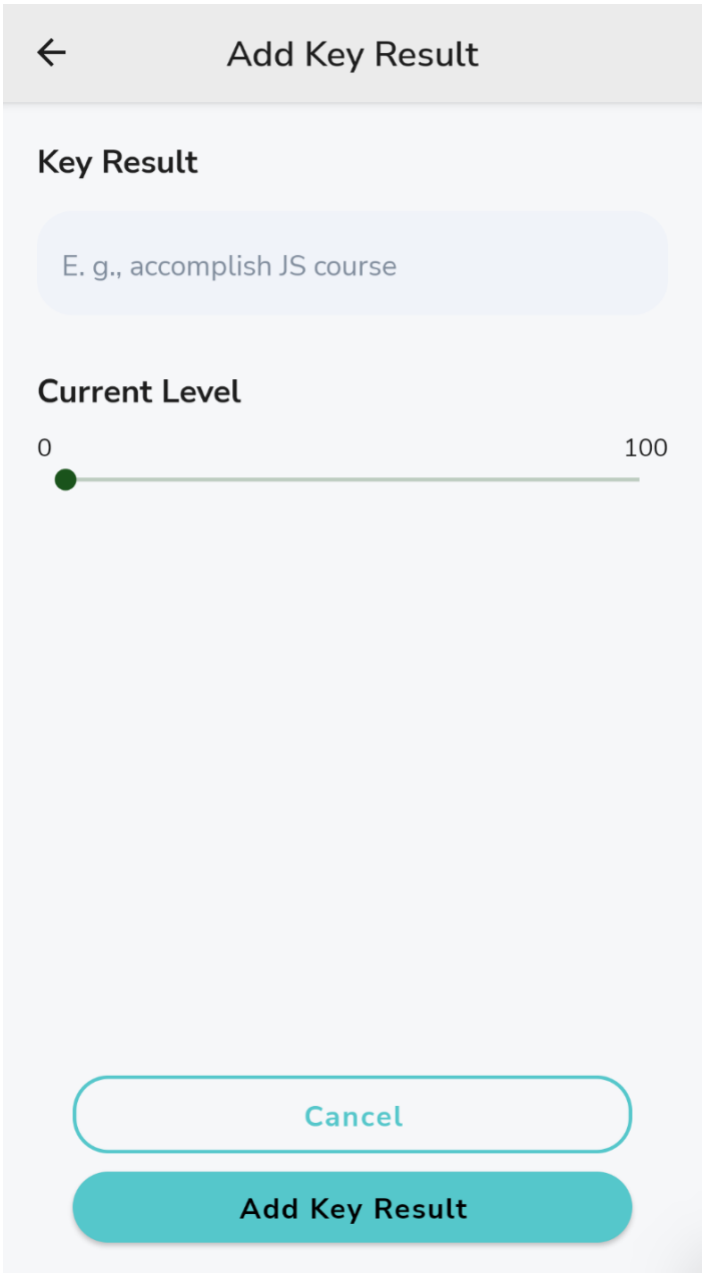


Рис. 27. Сторінка «Add Key Result» мобільного застосунку.

13. Сторінки «Profile» та «Add Skill» мобільного застосунку

Сторінка «Profile» містить такі секції: «Summary», «Skills», «Experience», «Education» та «Settings». Розглянемо детально кожну з них.

Секція «Summary», містить основну інформацію про користувача, його ім’я та прізвище, фото профіля, посаду та рівень фахової компетенції (рис. 28).

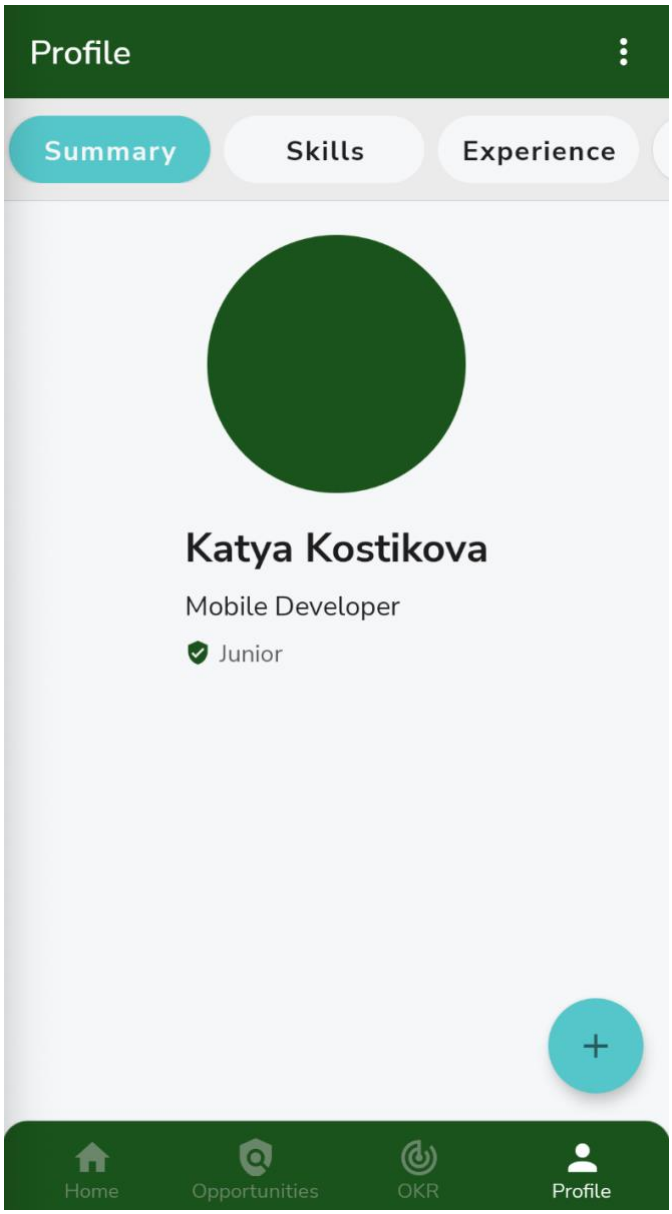


Рис. 28. Сторінка «Profile» мобільного застосунку. Секція «Summary»

Секція «Skills», містить інформацію про мови якими володіє

користувач та навички які має. Інформація про мову містить назву, рівень володіння нею та сертифікат. Інформація про навичку містить її назву та тип (рис. 29).

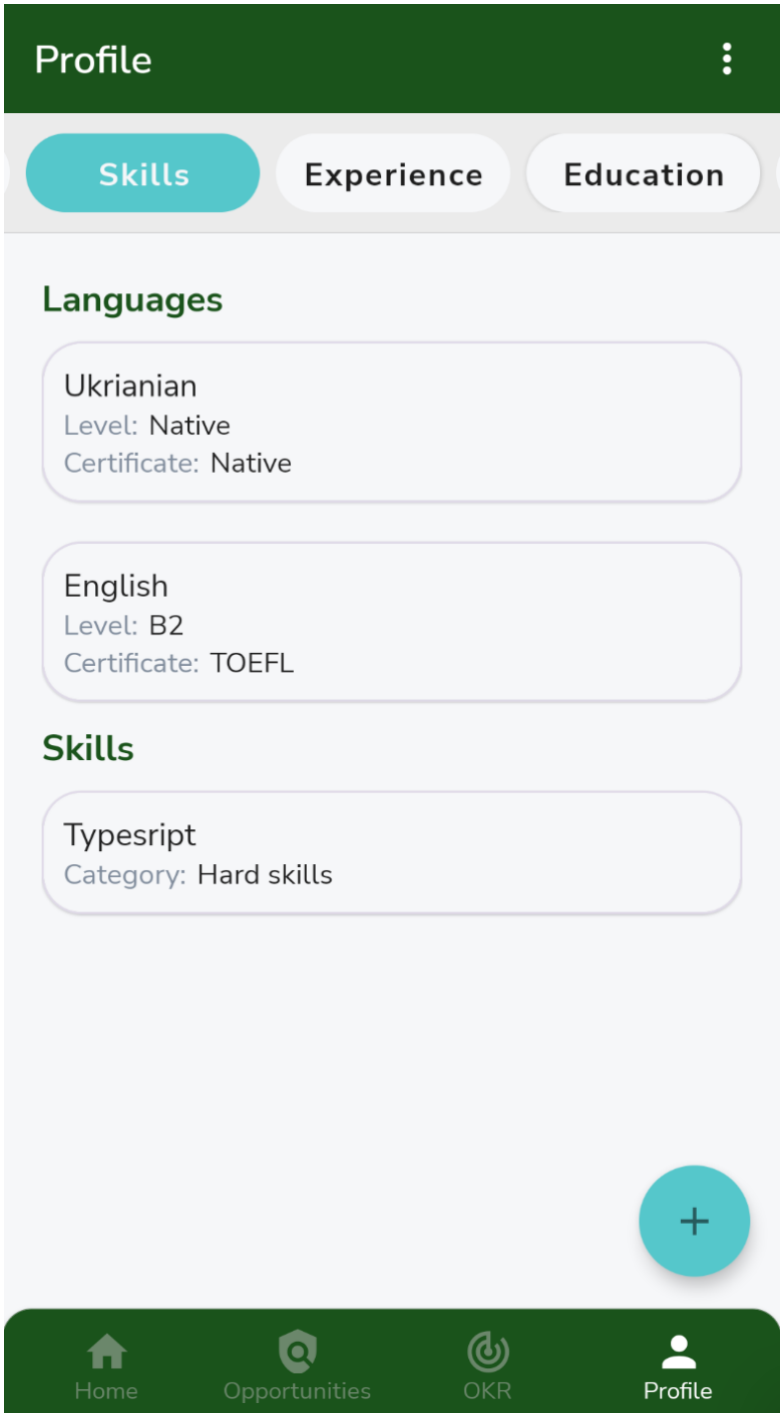


Рис. 29. Сторінка «Profile» мобільного застосунку. Секція «Skills»

Секція «Experience», містить інформацію про досвід роботи користувача (рис. 30). Ця секція дублює секцію про досвід роботи на

сторінці «Onboarding» описаної в розділі 8.

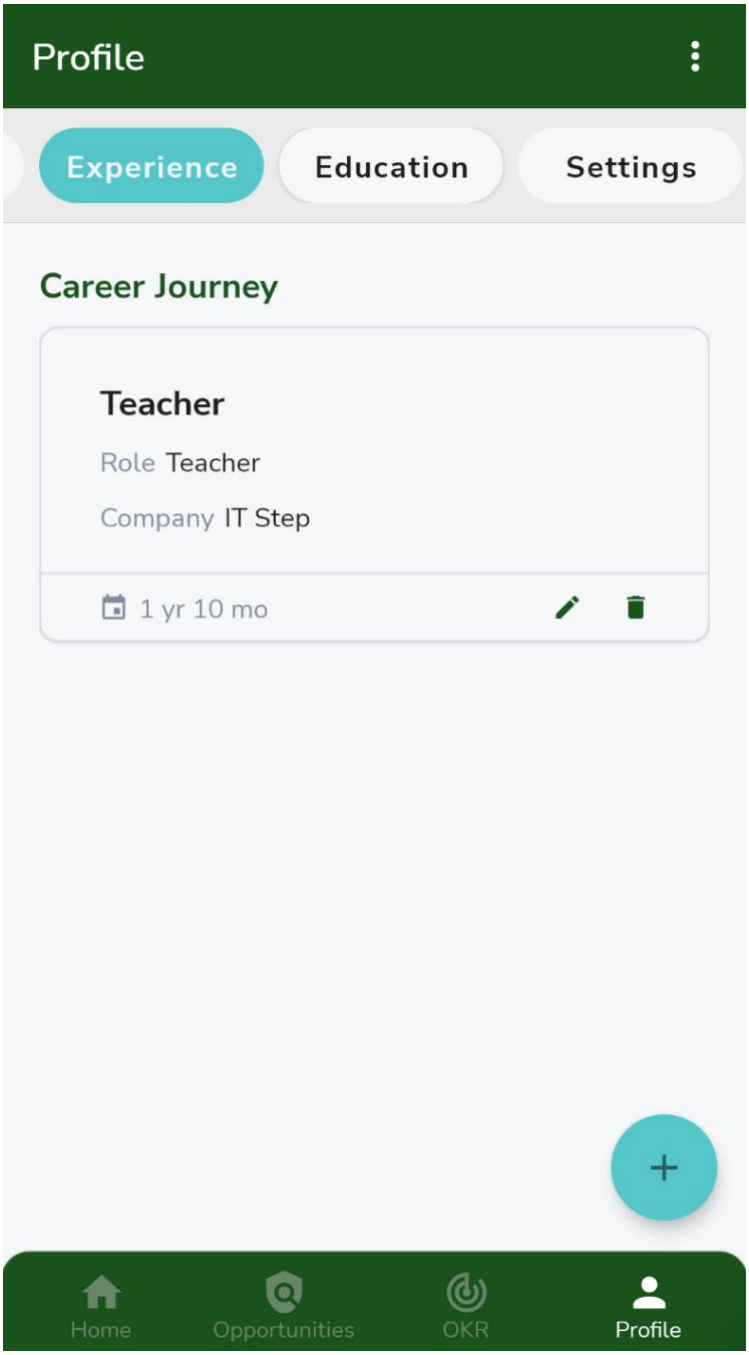


Рис. 30. Сторінка «Profile» мобільного застосунку. Секція «Experience»

Секція «Education», містить інформацію про освіту користувача (рис. 31). Ця секція дублює секцію про освіту на сторінці «Onboarding» описаної в розділі 8.

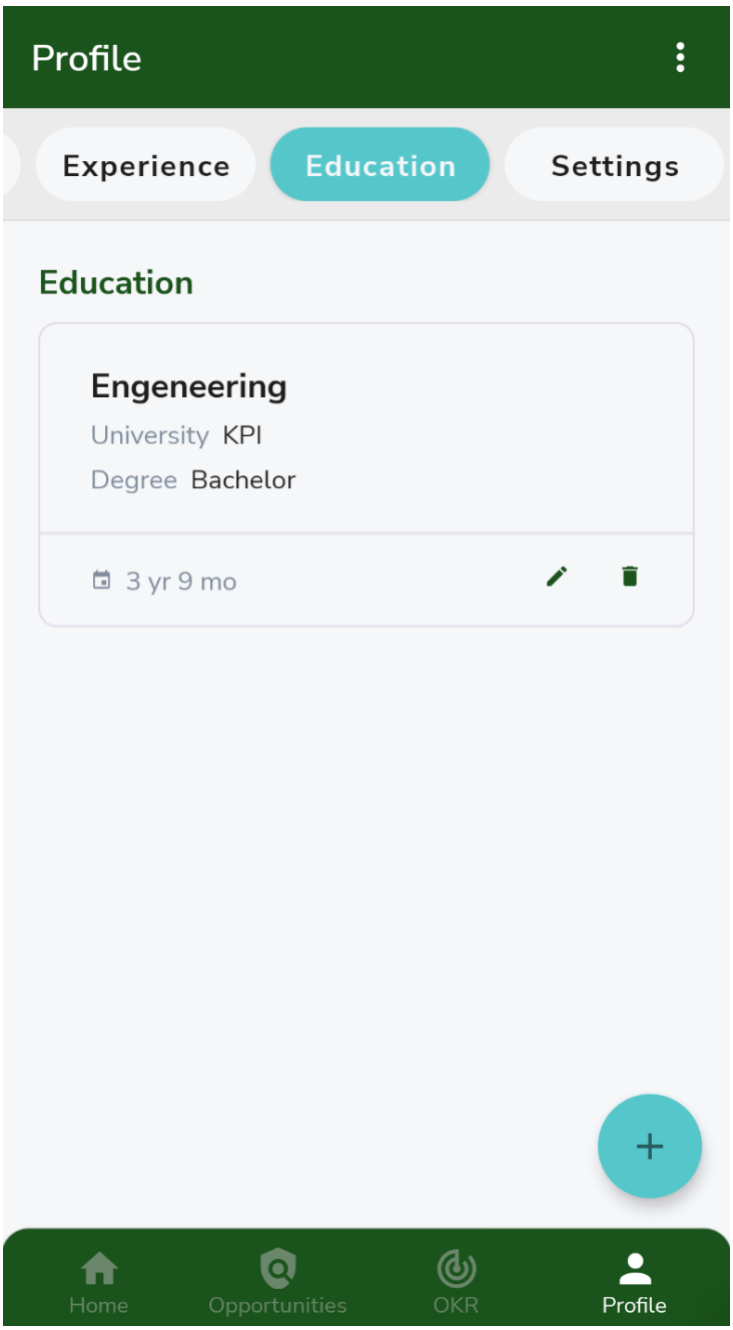


Рис. 31. Сторінка «Profile» мобільного застосунку. Секція «Education»

Секція «Settings» містить налаштування програмного застосунку (рис. 32). Користувач може змінити зовнішній вигляд програмного застосунку застосувавши темну тему. Для цього користувач має натиснути на перемикач біля «Dark theme».

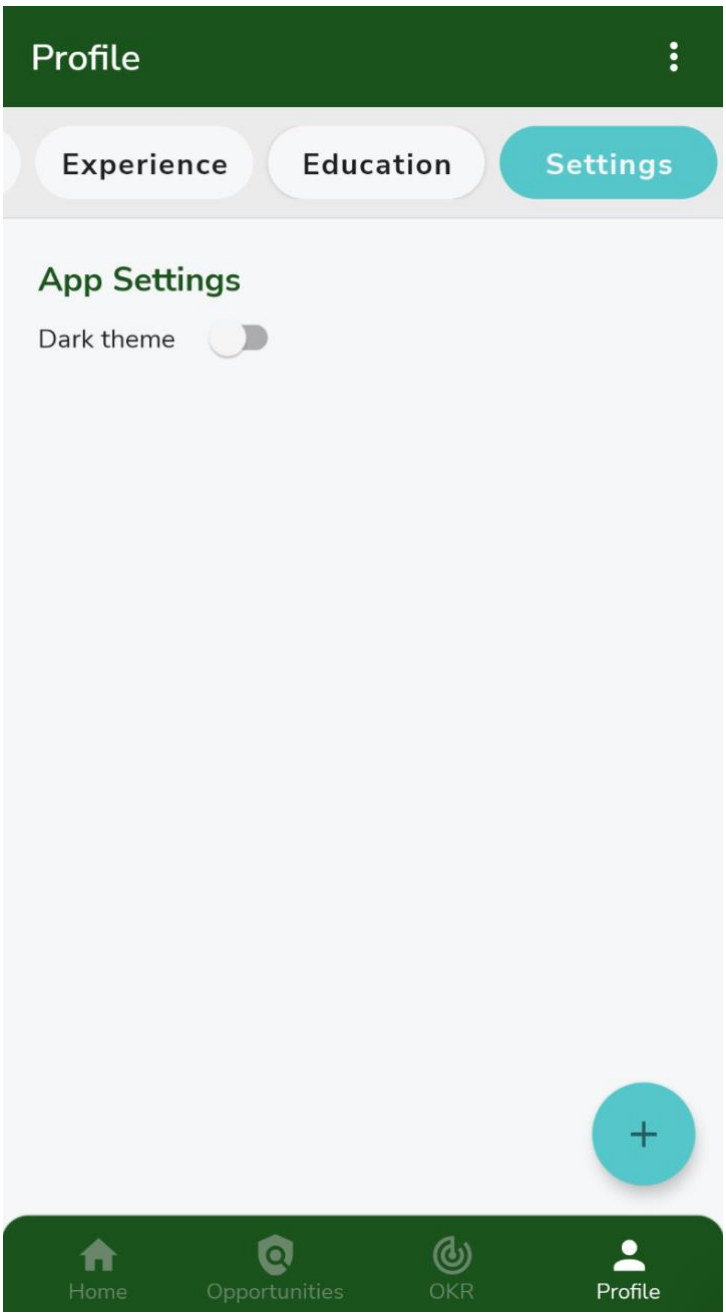


Рис. 32. Сторінка «Profile» мобільного застосунку. Секція «Settings»

При натисканні на кнопку у вигляді плюса, користувач може додати свої дані, а саме: навичку, досвід роботи, освіту та мову (рис. 33). Сторінки з формами для додавання інформації про досвід, освіту та мову описані в розділі 9.

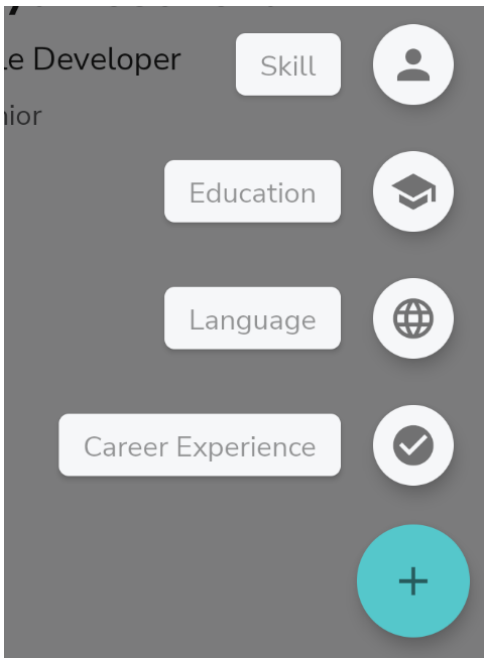


Рис. 33. Кнопка для додавання особистої інформації

Для виходу з облікового запису користувач має натиснути на три крапочки вгорі сторінки та натиснути кнопку «Logout» (рис. 34).

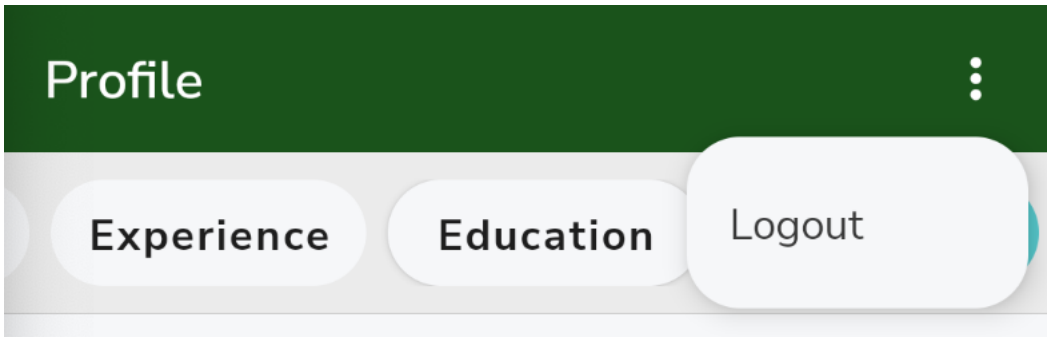


Рис. 34. Кнопка «Logout»

Сторінка «Add Skill» містить наступні поля: тип навички та назва навички (рис. 35). Після введення коректних даних користувач натискає кнопку «Save».

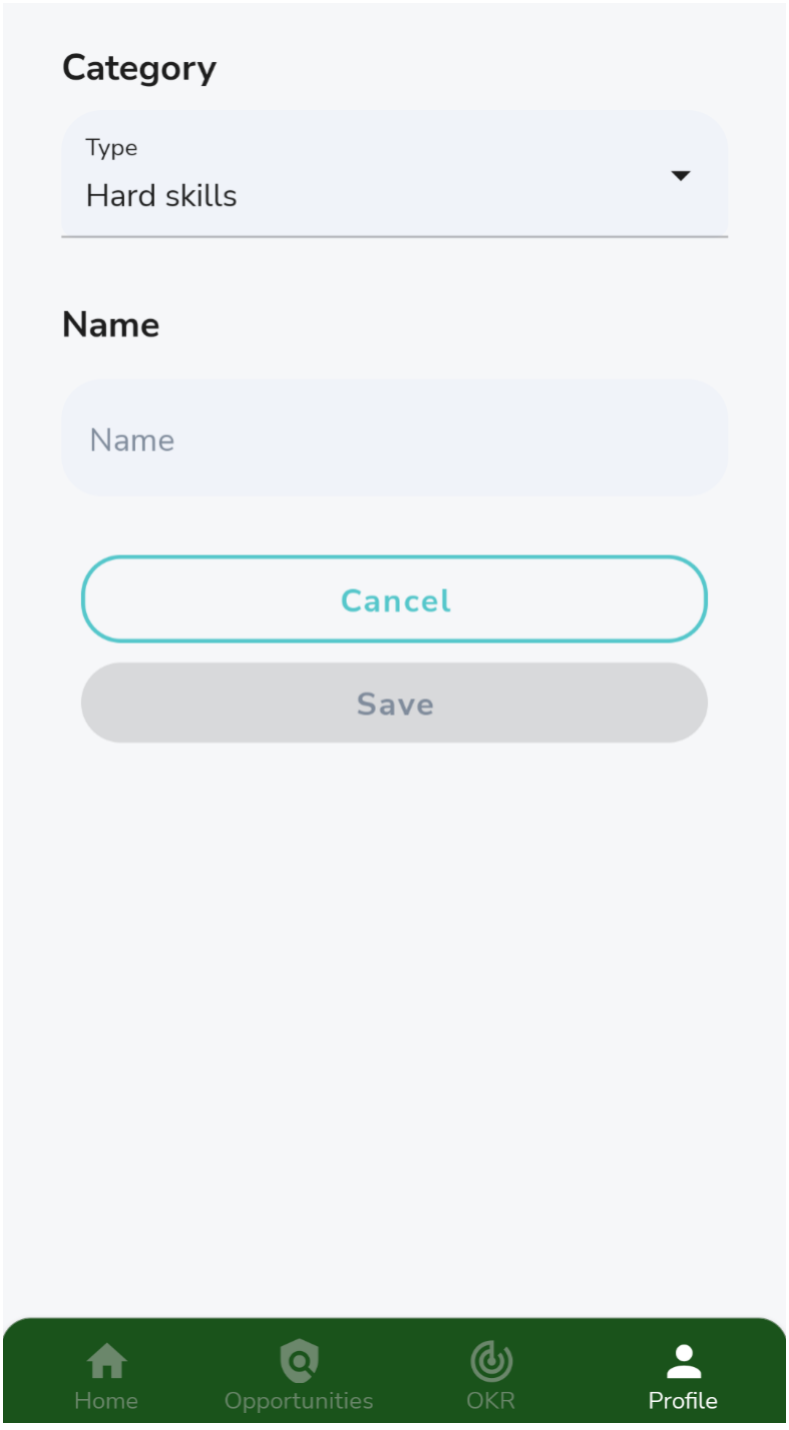


Рис. 35. Сторінка «Add skill» мобільного застосунку