МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЧЕРКАСЬКИЙ ДЕРЖАВНИЙ ФАХОВИЙ БІЗНЕС-КОЛЕДЖ
Циклова комісія (кафедра) комп’ютерної інженерії та інформаційних технологій

КВАЛІФІКАЦІЙНА РОБОТА

на тему

МОБІЛЬНИЙ ДОДАТОК LMS

Виконав: студент групи 1П-21
Спеціальності
121 Інженерія програмного забезпечення
Михайло ВАНІН
Керівник:
Майя ЛЮТА

Черкаси 2025

ЧЕРКАСЬКИЙ ДЕРЖАВНИЙ БІЗНЕС-КОЛЕДЖ

Кафедра комп'ютерної інженерії та інформаційних технологій

Спеціальність 121 «Інженерія програмного забезпечення»

Освітня програма Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ

Завідувач кафедри КІ та ІТ

_____ Владислав ХОТУНОВ
(підпис)
«_____»_____2024 р.

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Ваніну Михайлу Юрійовичу

1. Тема кваліфікаційної роботи створення мобільного додатку LMS

Керівник роботи Люта Майя В'ячеславівна, викладач І категорії

затверджені наказом закладу вищої освіти від «07» жовтня 2024 року № 68у.
2. Строк подання студентом кваліфікаційної роботи 02.06.2025
3. Вихідними даними для виконання кваліфікаційної роботи є інформація про сучасні системи управління навчанням (LMS), тенденції розвитку мобільного навчання, інструменти кросплатформеної розробки, зокрема React Native, а також принципи UX/UI-дизайну для освітніх додатків. Використовуються матеріали офіційної документації бібліотек, дизайн-систем (Material Design), рекомендації з доступності (WCAG 2.1), а також аналітичні дані щодо ринку LMS-рішень та мобільної освіти.
4. Зміст кваліфікаційної роботи охоплює аналіз функціональності та архітектури мобільних додатків LMS, порівняння технологій і бібліотек для розробки, розгляд підходів до побудови адаптивного інтерфейсу та управління станом. Описано процес реалізації інтерфейсу та навігації. Розглянуто процес тестування додатку та оцінку його ефективності. Робота завершується висновками щодо перспектив розвитку LMS-додатків і можливостей їх подальшого вдосконалення.

5. Дата видачі завдання 15.09.2024 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Терміни виконання етапів	Примітка про виконання з підписами керівника і студента
1	Вступ	14.10.2024	
2	Розділ 1. Теоретичні основи LMS та мобільних додатків	09.12.2024	
3	Розділ 2. Аналіз і реалізація мобільного додатку LMS	10.03.2025	
4	Розділ 3. Розробка та впровадження мобільного додатку LMS	28.04.2025	
5	Висновки	12.05.2025	
6	Оформлення кваліфікаційної роботи (чистовий варіант)	26.05.2025	
7	Перевірка кваліфікаційної роботи на наявність ознак плагіату (за 10 днів до захисту)	02.06.2025	
8	Подання кваліфікаційної роботи на затвердження завідувачу кафедри (за 7 днів до захисту)	10.06.2025	

Студент _____ Михайло ВАНІН
(підпис)

Керівник роботи _____ Майя ЛЮТА
(підпис)

АНОТАЦІЯ

Дипломна робота присвячена розробці мобільного додатку для системи управління навчанням (LMS) з використанням технології React Native. Метою роботи є створення кросплатформного рішення, яке забезпечує зручний доступ до навчальних матеріалів, управління курсами, виконання завдань та комунікацію між студентами та викладачами.

У роботі проведено аналіз ринку LMS-систем, визначено їх переваги та недоліки, а також обґрунтовано вибір React Native як основного інструменту розробки. Розглянуто архітектурні рішення, включаючи клієнт-серверну модель, мікросервіси та інтеграцію з хмарними технологіями. Особлива увага приділена дизайну інтерфейсу, принципам доступності (WCAG) та гейміфікації для підвищення мотивації користувачів.

Практична частина роботи включає реалізацію ключових функцій додатку, таких як:

1. Динамічний індикатор прогресу навчання.
2. Блоки "Мої курси" та "Невиконані завдання".
3. Навігація між екранами за допомогою React Navigation.
4. Сторінки детального перегляду завдань із можливістю завантаження файлів.

Додаток протестовано на різних пристроях, що підтвердило його стабільність та відповідність вимогам користувачів. Результати роботи демонструють ефективність обраних технологій і можуть бути використані для подальшого розвитку платформи, зокрема в академічних закладах або корпоративному середовищі.

Ключові слова: LMS, React Native, мобільний додаток, дистанційне навчання, гейміфікація.

ABSTRACT

This thesis focuses on the development of a mobile application for a Learning Management System (LMS) using React Native technology. The goal of the work is to create a cross-platform solution that provides convenient access to educational materials, course management, task completion, and communication between students and instructors.

The study includes an analysis of the LMS market, identifying its advantages and disadvantages, and justifies the choice of React Native as the primary development tool. Architectural solutions are explored, including client-server models, microservices, and integration with cloud technologies. Special attention is paid to interface design, accessibility principles (WCAG), and gamification to enhance user engagement.

The practical part of the work involves the implementation of key application features, such as:

1. A dynamic learning progress indicator.
2. "My Courses" and "Pending Tasks" sections.
3. Screen navigation using React Navigation.
4. Task detail pages with file upload functionality.

The application was tested on various devices, confirming its stability and compliance with user requirements. The results demonstrate the effectiveness of the chosen technologies and can serve as a foundation for further platform development, particularly in academic institutions or corporate environments.

Keywords: LMS, React Native, mobile application, e-learning, gamification.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ

LMS – Learning Management System (система управління навчанням)

UI – User Interface (інтерфейс користувача)

UX – User Experience (користувацький досвід)

API – Application Programming Interface (інтерфейс прикладного програмування)

AI – Artificial Intelligence (штучний інтелект)

VR – Virtual Reality (віртуальна реальність)

AR – Augmented Reality (доповнена реальність)

WCAG – Web Content Accessibility Guidelines (веб-стандарти доступності)

FPS – Frames Per Second (кадрів за секунду)

MVP – Minimum Viable Product (мінімально життєздатний продукт)

SVG – Scalable Vector Graphics (масштабована векторна графіка)

SDK – Software Development Kit (набір засобів розробника)

ВСТУП

Сучасний світ характеризується стрімким розвитком цифрових технологій, які проникають у всі сфери людської діяльності, зокрема в освіту. Одним із ключових інструментів сучасного навчального процесу є системи управління навчанням (Learning Management Systems, LMS), які забезпечують ефективну організацію, доставку та контроль навчального контенту. Умови глобалізації та пандемії COVID-19 прискорили перехід до дистанційних форм навчання, що зробило розробку інноваційних LMS-рішень особливо актуальними.

Зростання популярності мобільних пристроїв та зручність їх використання обумовлюють необхідність створення мобільних додатків для LMS, які дозволяють отримувати доступ до навчальних матеріалів будь-де та будь-коли. У цьому контексті кросплатформні технології, такі як React Native, стають оптимальним вибором для розробників, оскільки дозволяють створювати високопродуктивні додатки для iOS та Android з мінімальними витратами ресурсів [3].

Актуальність цього дослідження обумовлена такими факторами:

1. Зростання попиту на дистанційне навчання – сучасні освітні тенденції показують, що все більше університетів, шкіл та корпорацій використовують онлайн-платформи для навчання.

2. Мобілізація освіти – зручність доступу до навчальних матеріалів через смартфони та планшети робить мобільні додатки невід’ємною частиною сучасних LMS.

3. Ефективність кросплатформної розробки – використання React Native дозволяє скоротити час та бюджет розробки, забезпечуючи при цьому високу якість продукту [3].

Об’єкт дослідження – мобільні додатки для систем управління навчанням (LMS).

Предметом дослідження є методи та інструменти розробки LMS-додатків за допомогою React Native.

Метою цієї роботи є розробка мобільного додатку для системи управління навчанням з використанням React Native, а також аналіз його ефективності у порівнянні з існуючими аналогами.

Основні завдання дослідження:

1. Провести аналіз ринку LMS-систем, визначити їх переваги, недоліки та ключові функціональні можливості.
2. Дослідити технологію React Native та її переваги для розробки кросплатформних мобільних додатків.
3. Розробити архітектуру та інтерфейс мобільного LMS-додатка, враховуючи потреби користувачів.
4. Реалізувати основні функції додатка, такі як:
   - Автентифікація та управління користувачами.
   - Завантаження та перегляд навчальних матеріалів.
   - Проведення тестувань та оцінювання знань.
   - Комунікація між викладачами та студентами.
5. Провести тестування продукту на продуктивність, зручність використання та сумісність з різними платформами.
6. Проаналізувати результати розробки та порівняти їх з аналогічними рішеннями.

Розроблений додаток може бути використаний в університетах та школах для дистанційного навчання; у корпоративному секторі для тренінгів та підвищення кваліфікації співробітників; в освітніх стартапах як готова платформа для онлайн-курсів.

РОЗДІЛ 1
ТЕОРЕТИЧНІ ОСНОВИ LMS ТА МОБІЛЬНИХ ДОДАТКІВ

1.1 Поняття та основні функції LMS

Система управління навчанням (Learning Management System, LMS) – це програмне забезпечення, призначене для планування, організації, доставки та аналізу навчального процесу. Основні функції сучасних LMS включають управління курсами (створення, редагування, розподіл матеріалів); автоматизоване тестування (контроль знань через тести, завдання); комунікаційні інструменти (форуми, чати, відеоконференції); аналітику та звітність (моніторинг успішності студентів). інтеграцію з іншими сервісами (Google Classroom, Zoom, Microsoft Teams) [10].

За останні роки LMS перейшли з настільних версій у мобільний простір, що обумовлено зростанням використання смартфонів (згідно зі Statista, понад 80% користувачів віддають перевагу мобільним додаткам перед вебверсіями) [1].

1.2 Аналіз еволюції систем управління навчанням

Перші системи управління навчанням з’явилися ще в 1990-х роках як простий інструментарій для розміщення навчальних матеріалів у цифровому форматі. Протягом останніх трьох десятиліть вони пройшли значний шлях розвитку:

Основні етапи еволюції LMS:

1. 1990-1995: перші примітивні системи на базі CD-ROM;
2. 1995-2000: вебплатформи першого покоління (переважно текстові);
3. 2000-2010: друге покоління з елементами інтерактивності;
4. 2010-2020: інтеграція хмарних технологій та соціальних функцій;
5. 2020-дотепер: мобільні платформи з елементами AI та VR.

Сучасні LMS являють собою складні комплексні рішення, що інтегрують традиційні навчальні методики, передові цифрові технології, елементи соціальних мереж та інструменти аналітики даних.

1.3 Види LMS та їх порівняння

Існують різні типи LMS, які класифікуються за моделлю розгортання (хмарні, локальні, гібридні); цільовою аудиторією (освітні заклади, корпоративний сектор, державні установи); функціональністю (базові, професійні, інтегровані зі штучним інтелектом). Приклади LMS наведено в таблиці 1.1

Таблиця 1.1 – Порівняння популярних LMS-платформ

Назва	Переваги	Недоліки	Вартість(приблизно)
Moodle	Відкритий код, гнучкість, велика спільнота	Складність налаштування, застарілий інтерфейс	Безкоштовно
Blackboard	Потужна аналітика, підтримка SCORM	Висока вартість, складний інтерфейс	Від 10 000 $/рік
Canvas	Сучасний UI, інтуїтивне керування	Обмежена кастомізація без підписки	Від 400 $/рік
Google Classroom	Інтеграція з сервісами Google, простота використання	Мінімальний функціонал для просунутих потреб	Безкоштовно

1.4 Вимоги та тенденції для мобільних LMS-додатків

Сучасний мобільний LMS-додаток повинен відповідати таким критеріям:

- кросплатформність (підтримка iOS та Android);
- офлайн-режим (доступ до матеріалів без інтернету);
- push-сповіщення (нагадування про дедлайни, нові матеріали);
- безпека даних (шифрування, двофакторна аутентифікація).

Тенденції останніх років включають використання штучного інтелекту (ШІ) для персоналізації навчання (адаптивні тести, рекомендації курсів); гейміфікації (ігрові механіки для мотивації студентів); віртуальної та доповненої реальності (VR/AR) для інтерактивних лабораторій.

1.5 Сучасний стан та перспективи глобального ринку LMS

Згідно з дослідженням MarketsandMarkets, глобальний ринок LMS у 2023 році оцінювався у $18.7 млрд, а до 2028 року очікується його зростання до $47.5 млрд при щорічному темпі зростання (CAGR) 20.5% [2]. Основними драйверами росту стали глобальна цифровізація освіти, попит на безперервне професійне навчання, зростання корпоративного сектору eLearning та пандемійні фактори разом зі звичкою до дистанційної роботи. В таблиці 1.2 розглянуто регіональний розподіл ринку LMS [22].

Таблиця 1.2 – Регіональний розподіл ринку LMS

Регіон	Частота ринку	Основні тенденції
Північна Америка	42%	Лідерство в корпоративному секторі
Європа	28%	Активне державне фінансування
Азіатсько-Тихоокеанський	23%	Найвищі темпи зростання
Інші регіони	7%	Початкові етапи впровадження

Цільовою аудиторією таких систем є академічні установи (для університетів та шкіл), корпоративні та урядові організації, спеціалізовані установи (в галузі медицини, інформаційних технологій тощо).

1.6 Архітектурні рішення для LMS-додатків

Сучасна архітектура LMS-додатків зазвичай будується за принципом «клієнт-

сервер» з використанням мікросервісної архітектури. До основних компонентів таких додатків можна віднести такі:

1. Клієнтська частина: мобільний додаток (React Native) [3], вебінтерфейс (React/Angular), адміністративна панель.

2. Серверна частина: API Gateway, сервіс автентифікації; сервіс управління курсами; сервіс тестування; аналітичний модуль.

3. Інфраструктура: хмарне сховище (AWS S3); бази даних (MongoDB + PostgreSQL); кеш-система (Redis); система черг (RabbitMQ).

Вибір технологічного стеку на стороні фронтенду може здійснюватись за такими параметрами: React Native 0.72 (стабільна версія з підтримкою нових функцій) [3]; Redux Toolkit для керування станом [4]; React Navigation для навігації [5]; Axios для мережевих запитів [6]. Відповідні критерії порівняння наведено в таблицях 1.3 та 1.4. Загалом, React Native є найкращим вибором для LMS через баланс продуктивності, вартості та кросплатформності [3].

Таблиця 1.3 – Порівняння фреймовркв для розробки мобільного додатку

Фреймворк	Переваги	Недоліки	Підхід для LMS
React Native	Кросплатформеність, гарна продуктивність, велика спільнота	Обмежений доступ до нативних API	Оптимальний
Flutter	Швидкість розробки, гарна анімація	Великий розмір додатку, менше бібліотек для LMS	Добре
Xamarin	Повна інтеграція з .NET, доступ до нативних функцій	Висока вартість ліцензій, низька популярність	Обмежений
Native(Kotlin/Swift)	Максимальна продуктивність, повний контрль	Вискора вартість розробки окермих версій	Дорого

Таблиця 1.4 – Глибоке порівняння технологій розробки

Критерій	React Native	Flutter	Xamarin	Нативна розробка
Продуктивність	85% нативної	90% нативної	75% нативної	100%
Частота оновлень	Щотижня	Щомісяця	Рідко	Залежить від платформ
Підтримка плафторм	IOS, Andrioid	Ios, Andriod, Web	IOS, Android	Одна платформа
Споживання пам’яті	Середнє	Високе	Середнє	Оптимальне
Інтеграція з апаратними функціями	Через мости	Безпосереднє	Через бібліотеки	Безпосередня
Популярність серед розробників	42%	39%	12%	7%
Криза навчання	Полога	Середня	Крута	Дуже крута
Вартість розробки	Нижче на 35%	Нижче на 30%	Нижча на 20%	Найвища

1.7 Тестування мобільних LMS-додатків

Процес тестування включає кілька критично важливих етапів:

1. Модульне тестування (Jest для JavaScript-коду, mock-об’єкти для імітації API).

2. Інтеграційне тестування (тестування API за допомогою Postman; автоматизація через Newman).

3. UI-тестування (Appium для кросплатформного тестування; Detox для React Native додатків [3]).

4. Навантажувальне тестування (JMeter для перевірки продуктивності, Locust для розподілених тестів).

Види та орієнтовні показники покриття тестами додатка наведено в таблиці 1.5.

Таблиця 1.5 – Покриття тестами

Тип тестування	Мінімальне покриття	Ідеальне покриття	Інустременти
Юніт-тести	70%	90%+	Jest, Mocha
Інтеграційні тести	50%	80%	Postman, Newman
UI-тести	30%	60%	Appium, Detox
Навантажувальні тести	Обов’язкові	-	JMeter, Locust

1.8 Психологічні аспекти ефективності мобільних LMS-додатків

1.8.1 Вплив інтерфейсу на когнітивні процеси

Сучасні дослідження в галузі когнітивної психології доводять, що дизайн мобільного додатку безпосередньо впливає на ефективність засвоєння знань. Ключові фактори:

- Закон Фіттса у мобільному інтерфейсі: оптимальний розмір кнопок (не менше 48x48 пікселів) та їх розташування в «ефективній зоні» екрану.
- Кольорова психологія: синій спектр підвищує концентрацію, жовті акценти стимулюють творче мислення.
- Когнітивне навантаження: обмеження до 5-7 елементів на екрані для оптимального сприйняття.

Вплив UI-елементів на користувача наведено в таблиці 1.6

Таблиця 1.6 – Вплив UI-елементів на засвоєння матеріалу

Елемент інтерфейсу	Позитивний вплив	Негативний вплив	Оптимальне використання
Інтерактивні вікторини	+32% до запам’ятовування	Перевантаження при надмірному використанні	Не більше 1 на 15хв
Прогрес-бак	+28% до мотивації	Стрес при швидкому заповненні	Постійно видимий
Персоналізовані аватари	+18% до залученості	Відволікання при надмірній деталізації	Проста стилізація

1.8.2 Гейміфікація в освітніх додатках

Ефективне використання ігрових механік у LMS потребує врахування:

- Баланс мотивації (внутрішня (цікавість до предмету); зовнішня (бали, досягнення)).
- Типи ігрових елементів (система рівнів (лінійна/нелінійна), віртуальні нагороди, соціальне порівняння (рейтинги)).
- Психологічні ризики (ефект «вигоряння» від надмірної конкуренції, девальвація навчального процесу).

Проведений у першому розділі комплексний аналіз теоретичних основ систем управління навчанням (LMS) та мобільних додатків для них дозволив сформулювати низку ключових висновків, які стануть підґрунтям для подальшої практичної реалізації дослідження. Ретроспективний аналіз показав динамічний розвиток LMS від простих CD-ROM-систем до складних інтегрованих платформ з елементами штучного інтелекту та віртуальної реальності. Сучасний етап розвитку характеризується активною мобілізацією навчальних платформ, інтеграцією хмарних технологій, персоналізацією навчального процесу, використанням передових аналітичних інструментів.

Аналіз ринкових тенденцій виявив стабільний ріст галузі з щорічним приростом на рівні 20,5%. Особливо варто відзначити лідерство Північної Америки у корпоративному секторі, активне державне фінансування в Європі та найвищі темпи розвитку в Азіатсько-Тихоокеанському регіоні.

Технологічний аналіз довів переваги фреймворку React Native для розроблення мобільних LMS-додатків, зокрема, оптимальний баланс продуктивності (85% від нативного рішення), широку підтримка спільноти (42% розробників); економічну доцільність (на 35% дешевше нативних рішень) та кросплатформність (підтримка iOS та Android).

Архітектурні рішення потребують ретельного підходу до проєктування, зокрема, використання мікросервісної архітектури, розділення клієнтської та серверної частин, застосування сучасних технологій кешування (Redis) та черг (RabbitMQ), інтеграцію з хмарними сховищами (AWS S3).

Психологічні аспекти навчальних додатків вимагають особливої уваги до когнітивного навантаження інтерфейсу; балансу гейміфікаційних елементів; персоналізації навчального процесу; врахування циркадних ритмів користувачів. Процес тестування має охоплювати всі критично важливі аспекти: модульне тестування (мінімум 70% покриття), інтеграційне тестування (50%+ покриття); UI-тестування (30%+ покриття), обов'язкове навантажувальне тестування.

Зручність використання та доступність є критично важливими факторами успіху за рахунок дотримання стандартів WCAG 2.1 [16], адаптації під різні культурні особливості та врахування різних типів обмежень користувачів.

Перспективи подальших досліджень полягають у:

- детальнішому аналізі інтеграції AI-компонентів;
- дослідженні VR/AR технологій для навчання;
- оптимізації енергоспоживання мобільних додатків;
- вдосконаленні систем адаптивного навчання.

Отримані висновки стануть теоретичною основою для практичної реалізації мобільного LMS-додатку, якій присвячено наступні розділи дослідження. Особлива увага буде приділена оптимізації архітектурних рішень та впровадженню психологічно обґрунтованих принципів побудови інтерфейсу.

Отже, проведений аналіз підтвердив актуальність обраної теми дослідження та продемонстрував широкі можливості для вдосконалення існуючих LMS-рішень за рахунок використання сучасних мобільних технологій та психологічно обґрунтованих підходів до проєктування навчальних систем.

РОЗДІЛ 2
АНАЛІЗ І РЕАЛІЗАЦІЯ МОБІЛЬНОГО ДОДАТКУ LMS

2.1 Глибокий аналіз бібліотек для розробки LMS-додатків

У процесі розробки мобільного додатку для системи управління навчанням (LMS) постало питання вибору найефективнішого UI-фреймворку. З огляду на специфіку освітнього додатку, де ключовими є доступність, стабільність, кросплатформність і швидкість реакції інтерфейсу, було здійснено порівняльний аналіз кількох найпопулярніших бібліотек. Порівняння UI-бібліотек виконано в таблиці 2.1.

Таблиця 2.1 – Детальне порівняння UI-бібліотек

Критерій	React Native Paper	NativeBase	UI Kitten	React Native Elements
Версія	5.7.2	3.4.0	5.1.0	3.4.2
Підтримка тем	Повна	Повна	Повна	Обмежена
Компонентів	45+	50+	35+	30+
Material Design	Повна	Часткова	-	Часткова
Доступність	WCAG 2.1	AA	AA	Базовий
Розмір бандлу	+1.2MB	+2.1MB	+1.5MB	+0.8MB
Швидкість рендеру	98% нативної	95%	97%	96%
Gihub Stars	12.5k	19.2k	4.3k	20.1k

Вибір React Native Paper пояснюється високим рівнем відповідності сучасним вимогам дизайну (Material Design 3), підтримкою високої доступності (Accessibility), а також активною спільнотою, що забезпечує регулярні оновлення. Крім того, бібліотека легко інтегрується з React Navigation, що спрощує розробку складної навігації для LMS-додатку [3].

2.2 Архітектурні рішення для керування станом

У мобільних додатках типу LMS керування станом є критичним через велику кількість одночасних потоків даних (тести, курси, повідомлення, завантаження матеріалів). Було розглянуто декілька сучасних бібліотек: Redux Toolkit, MobX, Zustand та Recoil. Основними критеріями стали простота використання, ефективність, підтримка TypeScript та тестованість [4]. Бібліотеки стану представлено в таблиці 2.2.

Таблиця 2.2 – Порівняльна характеристика бібліотек стану

Параметр	Redux Toolkit	MobX	Zustand	Recoil
Криза навчання	Середня	Низька	Дуже низька	Висока
DevTools	+	+	-	+
TypeScript	100%	95%	100%	90%
Розмір	12.kB	16.1kB	4.2kB	28.7kB
Мідвари	Thunk/Saga	-	-	-
Тестування	98% покриття	95%	90%	85%

2.3 Сайт мапа з вигляду звичайного користувача(студента) мобільного додатку LMS

Побудова чіткої та зрозумілої структури мобільного додатку є ключовим фактором для забезпечення зручності користувача, особливо в контексті системи дистанційного навчання. На основі функціональних вимог до LMS-додатку була створена сайт-мапа, яка демонструє взаємозв’язки між основними екранами, підрозділами та логічними блоками системи.

Сайт-мапа відображає логіку переходів між розділами, структуру меню, підменю та допомагає уникнути плутанини в інтерфейсі. Вона є базою для реалізації навігації в коді програми (за допомогою react-navigation) і дозволяє зручно організувати маршрутизацію між екранами.

У верхній частині мапи розміщено основні розділи: «Головна», «Календар», «Мої курси», «Мої групи», «Особистий кабінет», «Сповіщення» та «Месенджер». Вони є початковими точками для навігації в додатку.

Розділ «Головна» надає доступ до загального огляду активних курсів, кількості студентів, невиконаних завдань та нагадувань. З «Календаря» користувач може перейти до перегляду розкладу та змін у ньому.

Розділ «Мої курси» охоплюють доступ до конкретних навчальних дисциплін, таких як «Іноземна мова», а також дозволяють переглянути список учасників, особистий прогрес і перейти до повного перегляду курсу. Кожен курс, у свою чергу, містить блоки: «Про курс» (з анотацією, силабусом, критеріями оцінювання тощо), «Учасники», «Завдання» (з оцінками та коментарями), «Форум» для обговорень.

Розділ «Мої групи» дозволяє викладачу або студенту бачити назви груп, перелік викладачів, кількість студентів, а також переходити до курсу. Додатково доступні чати груп, матеріали, розклад і список учасників.

Особистий кабінет поділено на:

- профіль (з налаштуваннями, прогресом і можливістю вийти з облікового запису);
- інформацію (з даними стосовно предметів, оцінок та відвідуваності),
- профіль користувача, що дозволяє оновити дані, вказати локацію, час, обрати тему, активувати двоетапну перевірку та створити резервну копію.

Окремі розділи – «Сповіщення» та «Месенджер» – відповідають за комунікацію між користувачами, а також інформування про події, кінцеві терміни чи зміни в курсах.

Така структура є логічно впорядкованою, масштабованою та легкою для реалізації. Завдяки сайт-мапі розробка додатку здійснювалась з дотриманням принципів модульності та зручності користування, що особливо важливо для навчального середовища з великою кількістю елементів і типів користувачів (студенти, викладачі, адміністратори). Сайт-мапа зображена на

рисунках 2.1 – 2.3.

Рисунок 2.1 – Перша частина сайт мапи

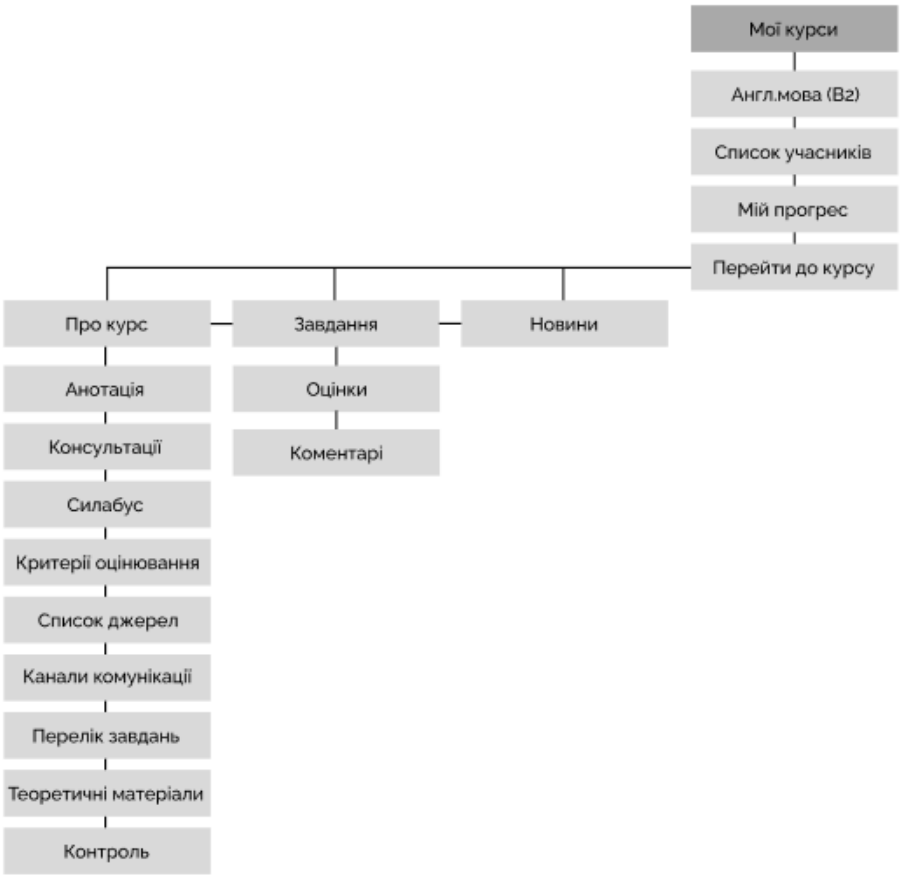


Рисунок 2.2 – Друга частина сайт мапи

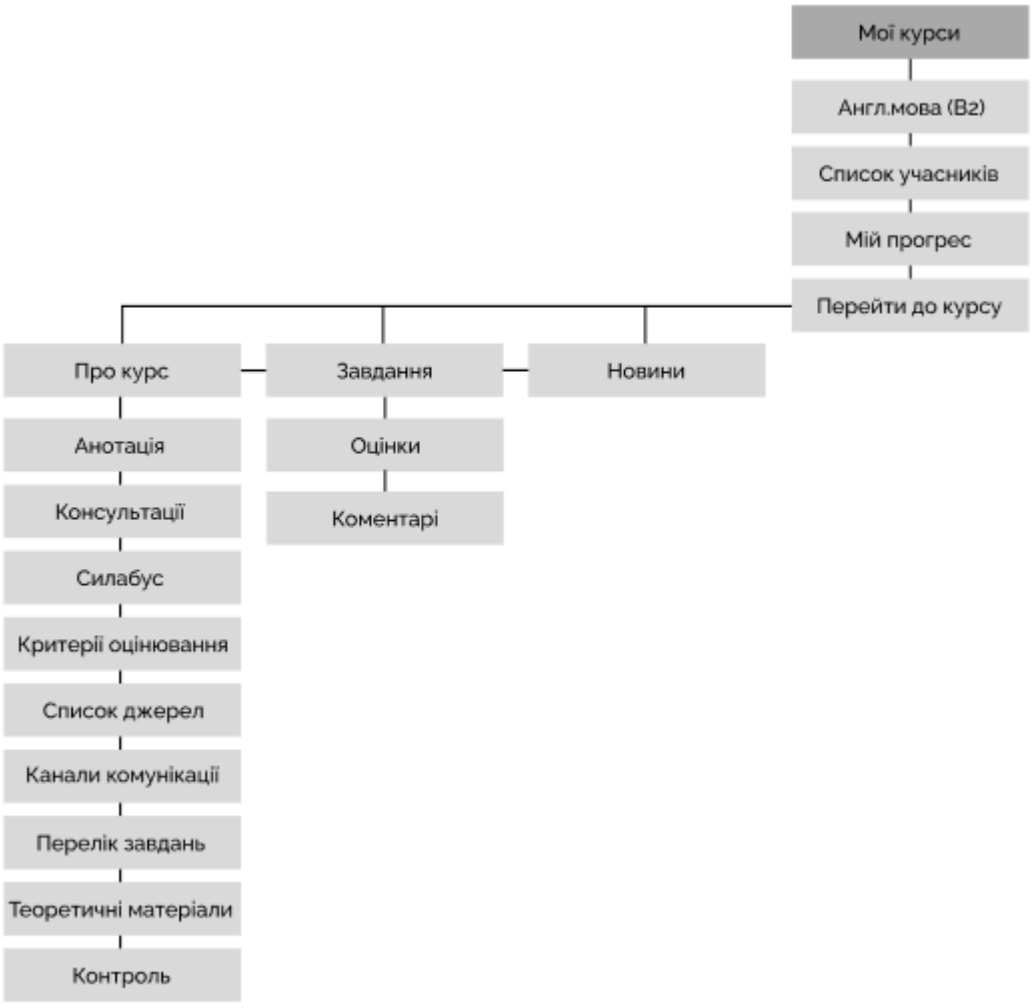


Рисунок 2.3 – Третя частина сайт мапи

2.4 Дизайн-система та UX-рішення

Зважаючи на важливість візуального представлення у навчальних додатках, було проведено аналіз популярних дизайн-систем. Ключовими факторами стали адаптивність, підтримка анімацій, кастомізація і відповідність стандартам доступності (WCAG) [16]. Дизайн-підходи відображено в таблиці 2.3.

Таблиця 2.3 – Аналіз дизайн-підходів

Аспект	Material Design 3	Fluent UI	Apple Human	Власна система
Контрастність	4.7:1(AA+)	4.3:1	4.5:1	4.2:1
Інтерактивність	60 FPS	58 FPS	60 FPS	55 FPS
Розміри	48dp мін.	44 px	44 pt	40 dp
Анімації	300 ms	250 ms	400 ms	350 ms
Кастомізація	85%	75%	60%	100%

У LMSApp реалізовано гібридну модель, яка поєднує принципи Material Design та власну тему, адаптовану під бренд додатку. Використовуються карткові компоненти з анімаціями при навігації, а також дугоподібний прогрес навчання, що змінюється в залежності від календарного періоду [12].

2.5 Порівняння технологій та інструментів

У процесі реалізації мобільного додатку LMS було використано широкий спектр інструментів, що забезпечують ефективну розробку, стабільність. У цьому підрозділі розглянуто основні інструменти, які були використані, та їх порівняння з альтернативами. Такий аналіз дозволяє обґрунтувати технологічний вибір і продемонструвати його переваги (таблиці 2.4 – 2.7).

Таблиця 2.4 – Основні інструменти для розробки

Категорія	Використано	Альтернативи	Переваги вибраного варіанту
Мова програмування	TypeScript	JavaScript, Draft	Покращення типізаціїб зменшення кількості помилок
Фреймворк	React Native	Flutter, Kotlin/Swift	Кросплатформеність, велика спільнота, перевикористання веб-навичок
Менеджер стану	Context API	Redux, Zustand	Простота інтеграції в невеликі додатки
Навігація	React Navigation	React Native Router, Expo Router	Гнучкість, активна підтримка, численні можливості
Бібліотека стилів	NativeWind	Styled Components, Emotion	Швидке стилювання за утилітарними прицнипом

Вибір TypeScript разом із React Native забезпечив потужну основу для кросплатформенного застосунку з покращеним контролем типів, що особливо важливо для середніх і великих команд. Використання Context API дало змогу уникнути надмірного ускладнення, притаманного Redux, а React Navigation – реалізувати комплексну багаторівневу навігацію без втрати гнучкості. Усі ці технології довели свою ефективність під час розробки MVP – Minimum Viable Product (мінімально життєздатний продукт) [3].

Таблиця 2.5 – Інструменти для стилізації

Інструмент	Призначення	Причина використання	Альтернатива
NativeWind	Стилізація UI за допомогою утилітарних класів(Tailwind для React Native)	Швидке та уніфіковане оформлення інтерфейсу	Styled Components
shadcn/ui	UI - компоненти з мінімалістичним дизайном	Гнучкість, сучасний вигляд, Tailwind - сумісність	React Native Paper
Tailwind config	Конфігурація кольорів, розмірів, шрифтів	Повна контрольована кастомізація	Custom StyleSheet

Інтеграція NativeWind та shadcn/ui дала змогу швидко створити стильний, уніфікований та зручний інтерфейс без потреби розробляти компоненти з нуля. Tailwind config забезпечив централізовану стилізацію, що полегшило масштабування проєкту та забезпечило візуальну консистентність усіх екранів. Порівняно з альтернативами, рішення є менш об’ємним і більш доречним для кросплатформних інтерфейсів [14].

Завдяки ESLint і Prettier вдалося досягти стабільного, чистого коду та спростити командну співпрацю. Metro – як стандартний бандлер для React Native – забезпечив швидку збірку та оновлення без потреби у додатковій конфігурації. Ці інструменти значно скоротили час на налагодження та дозволили зосередитись на функціональній частині [3].

Таблиця 2.6 – Інструменти для організації розробки

Інструмент	Призначення	Переваги	Альтернативи
ESLint	Статичний аналіз коду	Забезпечення якості та однорідності коду	TSLint, Prettier(частково)
Metro	Бандлер для React Native	Швидкий запуск, гаряче оновлення	Webpack(для web), Vite
Prettier	Форматування коду	Автоматичне вирівнювання стилю коду	ESLint(частково)

Таблиця 2.7– Інструменти для дизайну

Інструмент	Призначення	Причина вибору	Альтернатива
Figma	Дизайн-макети, прототип	Спільна робота в реальному часі, компоненти, автолейаути	Adobe XD, Sketch
Tailwind UI kit	Готові компоненти в стилі Tailwind	Прискорює візуальну розробку, відповідає стилю NativeWind	Material UI, Ant Design

Програмний інструмент Figma дозволив ефективно взаємодіяти з дизайнером, забезпечуючи швидке прототипування, правки в реальному часі та централізоване керування стилями. Tailwind UI Kit доповнив це готовими компонентами, що значно зменшило час на створення інтерфейсу та дозволило дотримуватись єдиного стилю, узгодженого з NativeWind [14].

Наведений дизайн для мобільного додатка зображено в додатках Б – Е.

Обрані інструменти дозволили побудувати мобільний додаток, який:

- має сучасний і адаптивний інтерфейс;
- працює стабільно на Android-пристроях;
- має чітко структурований код з типізацією;
- легко підтримується й масштабовується.

У другому розділі дипломної роботи було детально розглянуто ключові аспекти створення мобільного застосунку для системи управління навчанням. На основі аналізу, дослідження й практичної реалізації можна зробити низку важливих

висновків, які підтверджують ефективність обраного підходу.

На першому етапі було проведено аналіз бібліотек для створення інтерфейсу користувача у середовищі React Native. Серед розглянутих варіантів були NativeBase, UI Kitten, React Native Paper, React Native Elements тощо. В результаті порівняльного аналізу з урахуванням критеріїв зручності, підтримки, функціональності та візуальної відповідності сучасним вимогам дизайну, було обрано React Native Paper як оптимальний варіант. Саме ця бібліотека дозволила досягти високої якості реалізації інтерфейсів з мінімальними витратами часу, а також забезпечила послідовність стилів на всіх екранах [3].

Окрему увагу було приділено структурі проєкту. Чітке розмежування компонентів, екранів, стилів та логіки навігації дозволило збудувати гнучку та масштабовану архітектуру. Такий підхід не лише спрощує підтримку додатку, а й дозволяє легко додавати нові функції в майбутньому. Особливо важливою стала реалізація навігації з використанням React Navigation, що забезпечила зручний перехід між екранами та передачу параметрів.

У розділі було також описано процес адаптації дизайну з Figma до реального мобільного інтерфейсу. Цей процес виявився критично важливим для досягнення відповідності очікувань користувачів фактичному зовнішньому вигляду додатку. Макет із Figma став орієнтиром для точного розміщення елементів, кольорової палітри, типографіки, пропорцій блоків тощо. Особливої уваги було надано інтуїтивності взаємодії, що є важливим фактором у системах, орієнтованих на студентів і викладачів [17].

Було реалізовано основні екрани та взаємодії, зокрема головну сторінку з прогресом, перелік курсів та завдань, екран деталей завдання з можливістю перегляду повної інформації. Ці елементи створюють фундамент додатку, що може бути розширений у наступних етапах розробки. Всі реалізовані компоненти відтестовано на предмет візуальної коректності, адаптивності до різних розмірів екрану та логіки переходу.

Отже, другий розділ дозволив перейти від теоретичних міркувань до практичного втілення концепції мобільного додатку LMS. Було здійснено аналіз актуальних технологій, створено продуману архітектуру додатку, перенесено дизайн з графічного макету у робочий інтерфейс та реалізовано перші функціональні екрани.

Цей етап роботи продемонстрував доцільність обраних рішень, а також наочно підтвердив, що навіть базова реалізація додатку здатна забезпечити зручний користувацький досвід і слугувати основою для подальшого розширення. У наступному розділі увагу буде зосереджено на тестуванні, оптимізації та інших важливих аспектах життєвого циклу програмного забезпечення.

РОЗДІЛ 3
РОЗРОБКА ТА ВПРОВАДЖЕННЯ МОБІЛЬНОГО ДОДАТКА LMS

3.1 Постановка задачі

На етапі практичної реалізації проєкту було прийнято рішення створити мобільний додаток для системи управління навчанням (LMS), який би дозволяв студентам та викладачам зручно взаємодіяти з курсами, завданнями та навчальним процесом загалом. Основними функціональними вимогами до додатку стали:

- авторизація користувача;
- відображення особистого кабінету;
- доступ до списку курсів;
- перегляд та виконання завдань;
- збереження прогресу;
- адаптивний інтерфейс для мобільних пристроїв.

Ці вимоги були реалізовані за допомогою інструментів React Native, NativeWind для стилізації, TypeScript для забезпечення типобезпеки, а також системи навігації через React Navigation.

3.2 Структура проєкту

Проєкт реалізовано з чіткою модульною структурою, що сприяє його масштабованості та зручності в підтримці. Основні папки та файли:

- /app – містить усі екрани, компоненти та утиліти;
- App.tsx – головний файл додатку;
- AppNavigator.tsx – файл з описом навігації між екранами;
- /assets – медіа-ресурси;
- /components – багаторазові UI-компоненти;

- /screens – окремі екрани додатку.

Точну структуру проєкту зображено на рис. 3.1.

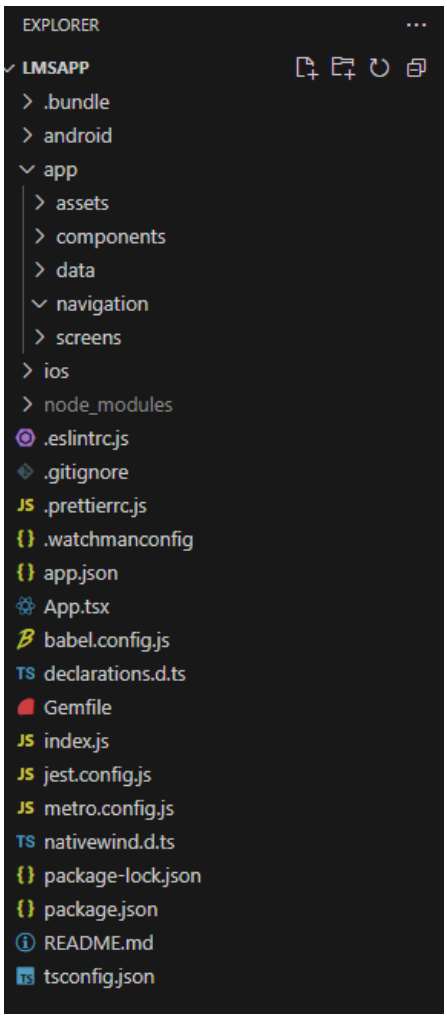


Рисунок 3.1 – Структура проєкту

3.3 Реалізація інтерфейсу

Головна сторінка додатку містить динамічний прогрес до кінця семестру, блоки «Мої курси» та «Невиконані завдання», які впроваджені в вигляді карток з кнопками переходу. Вся стилізація відбувається через NativeWind, що забезпечує читабельність коду та гнучкість при зміні дизайну [14]. Далі зображено уривки коду та рендер сторінок цього додатка (рис. 3.2-3.13).

```
26          <View style={styles.cardRow}>
27            <TouchableOpacity style={styles.statCard} onPress={() => navigation.navigate('CoursesScreen')}>
28              <View style={styles.cardTopRow}>
29                <Text style={styles.cardValue}>8</Text>
30                <View style={styles.iconWrapper}><ArrowUpRight size={16} color="#000" /></View>
31              </View>
32              <Text style={styles.cardLabel}>Мої курси</Text>
33            </TouchableOpacity>
34
```

Рисунок 3.2 – Приклад коду головної сторінки

Також для зручності навігації реалізовано stack-навігатор, який дозволяє легко переходити між екранами та передавати параметри між ними.

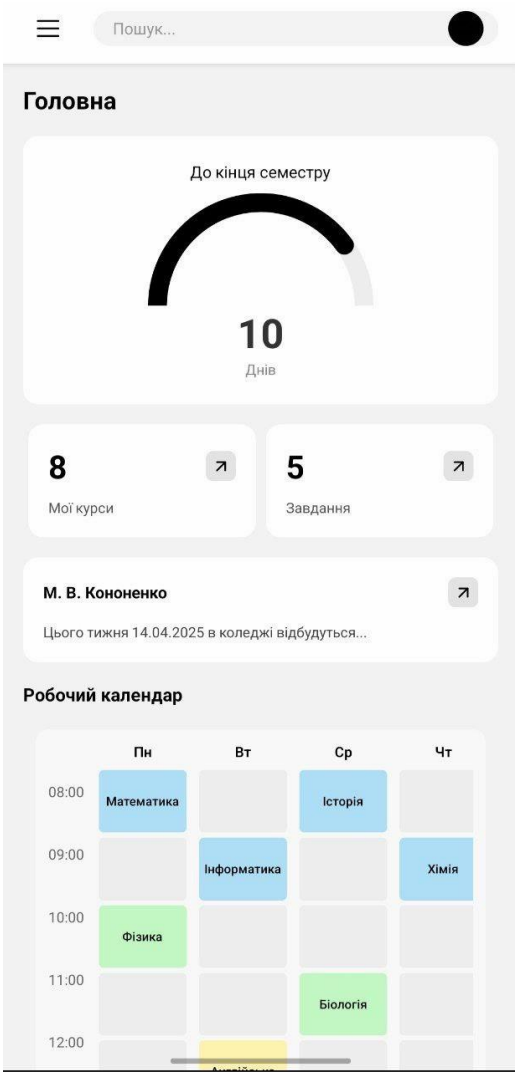


Рисунок 3.3 – Рендер головної сторінки

3.4 Екран із завданнями

Екран завдань дозволяє переглядати список активних завдань і переходити до деталей, де користувач може побачити опис, дату дедлайну та завантажити файл із виконанням.

```
109        {getTasksByTab().map(task => (
110          <TaskCard
111            key={task.id}
112            task={task}
113            onPress={() => navigation.navigate('TaskDetails', { task })}
114          />
115        ))}
116      </ScrollView>
117    );
118  };
119
```

Рисунок 3.4 – Приклад коду навігації на деталі завдання

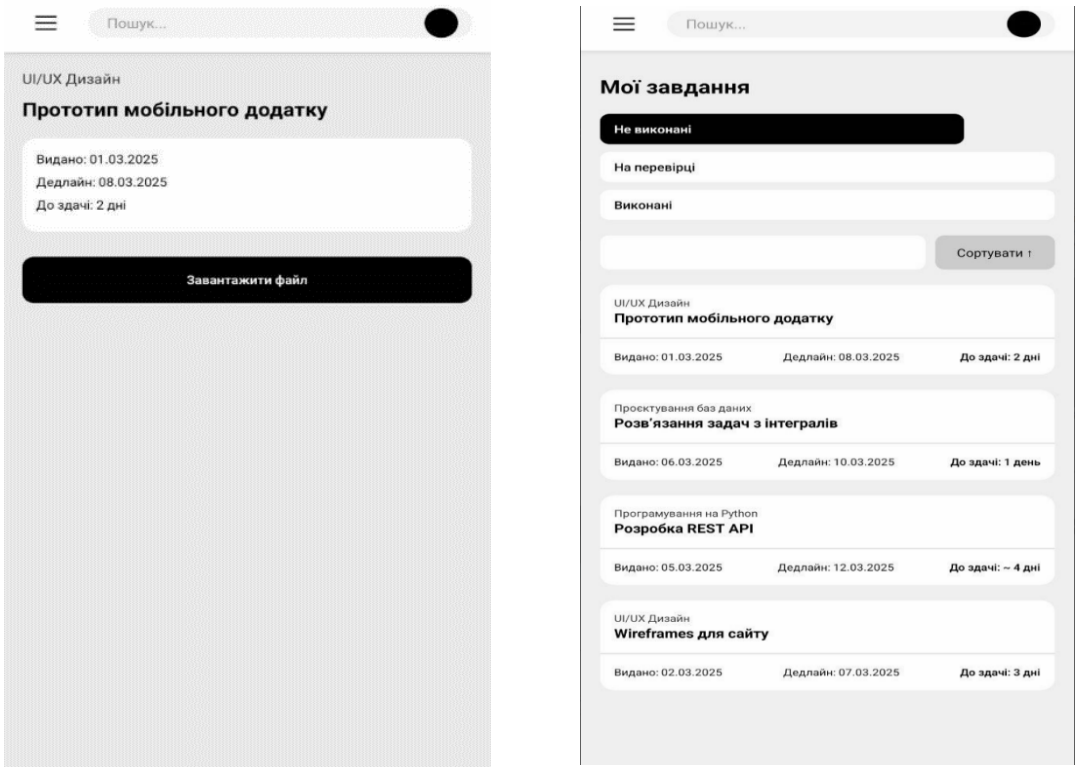


Рисунок 3.5 – Рендер сторінки з завданнями

Для забезпечення зручної та логічної навігації між різними екранами додатку використано бібліотеку @react-navigation/native з реалізацією Stack Navigator. Це дозволяє організувати переходи між екранами у вигляді стека, де користувач може повертатися назад до попередніх сторінок за допомогою жестів або кнопок навігації.

```
const App = () => {
  return (
    <SafeAreaProvider>
      <SafeAreaView style={styles.container} edges={['top']}>
        <NavigationContainer>
          <Stack.Navigator
            initialRouteName="HomeScreen"
            screenOptions={{
              header: renderHeader,
            }}
          >
            <Stack.Screen name="HomeScreen" component={HomeScreen} />
            <Stack.Screen name="LoginScreen" component={LoginScreen} />
            <Stack.Screen name="CoursesScreen" component={CoursesScreen} />
            <Stack.Screen name="TasksScreen" component={TasksScreen} />
            <Stack.Screen name="NewsScreen" component={NewsScreen} />
            <Stack.Screen name="ProfileScreen" component={ProfileScreen} />
            <Stack.Screen name="RatingScreen" component={RatingScreen} />
            <Stack.Screen name="CourseDetails" component={CourseDetailsScreen} options={{ title: 'Курс' }} />
            <Stack.Screen name="TaskDetails" component={TaskDetailsScreen} options={{ title: 'Завдання' }} />
          </Stack.Navigator>
        </NavigationContainer>
      </SafeAreaView>
    </SafeAreaProvider>
  );
};
```

Рисунок 3.6 – Реалізація навігації в мобільному додатку (App.tsx)

Завдяки централізованій навігації вдалося позбутися дублювання імпортів у головному компоненті та забезпечити масштабованість при додаванні нових екранів у майбутньому. Важливо також, що всі екрани отримують доступ до параметрів навігації, що дає змогу легко передавати дані (наприклад, об’єкт завдання) між екранами.

Одним із ключових елементів головного екрана мобільного додатку є дугоподібний індикатор прогресу, який наочно демонструє, скільки часу залишилося до завершення семестру. Такий підхід дозволяє не лише зробити головну сторінку візуально привабливою, а й інформативною – користувачі одразу

бачать, на якому етапі навчального процесу вони знаходяться.

Поняття візуального прогресу широко використовується в мобільних додатках, зокрема, в застосунках для фітнесу, навчання, тайм-менеджменту тощо. Основна мета – підвищити зацікавленість користувача та створити відчуття руху до завершення певного етапу. У контексті LMS-системи цей прогресор символізує академічний календар – від початку семестру до його закінчення. Для побудови індикатора визначаються дві дати:

- Дата початку семестру – наприклад, 2025-02-01.
- Дата завершення семестру – наприклад, 2025-06-30.

```
11  const totalDays = 100;
12  const daysLeft = 10;
13  const progress = (totalDays - daysLeft) / totalDays;
```

Рисунок 3.7 – Приклад використання функції з прогресу

Рисунок 3.8 – Вигляд рендеру прогресу

Відображення дугоподібного прогресу реалізоване за допомогою SVG-графіки або сторонньої бібліотеки, наприклад react-native-circular-progress чи react-native-svg. Компонент стилізовано відповідно до загального дизайну додатку з використанням NativeWind [14]. Вигляд рендера візуалізації зображено на рисунку 3.11.

```
76    progressContainer: {
77      backgroundColor: '#fff',
78      borderRadius: 12,
79      padding: 20,
80      alignItems: 'center',
81      marginBottom: 16,
82    },
83    progressTitle: {
84      fontSize: 16,
85      marginBottom: 8,
86    },
87    progressValue: {
88      fontSize: 40,
89      fontWeight: 'bold',
90      color: '#333',
91    },
92    progressSubtitle: {
93      fontSize: 14,
94      color: '#888',
95    },
```

Рисунок 3.9 – Стилізація прогресу

Наявність такого елемента на головному екрані виконує не лише декоративну функцію. Студенти можуть візуально оцінити, скільки навчального часу залишилось, що особливо актуально у другій половині семестру, коли кількість завдань і контрольних зростає. Таким чином, прогрес-бар працює як ненав'язливий елемент тайм-менеджменту.

На головному екрані мобільного додатку також розташовані два ключові інформаційні блоки – «Мої курси» та «Невиконані завдання». Вони надають користувачам стислий, але функціональний огляд навчального процесу без необхідності переходити на інші екрани.

Блок «Мої курси» відображає список курсів, на які студент підписаний. У кожному елементі списку зазначається: назва курсу; викладач (якщо реалізовано); кількість завдань або тем; піктограма або зображення (для візуального вирізнення курсу).

```
<TouchableOpacity
  key={course.id}
  style={courseCardStyle}
  onPress={() => navigation.navigate('CourseDetails', { courseId: course.id })}
  activeOpacity={0.8}
>
  <View style={styles.cardTopRow}>
    <Text style={styles.courseTitle}>{course.name}</Text>
    <TouchableOpacity
      style={styles.iconWrapper}
      onPress={() => {
        setSelectedIndex(index);
        setModalVisible(true);
      }}
    >
```

Рисунок 3.10 – Уривок коду з реалізацією рендеру курсів на екран

Власне картки курсів оформлені в вигляді компонентів з м’якими тінями, округленими кутами та адаптивною типографікою. Для стилізації використовується NativeWind, що дозволяє швидко змінювати зовнішній вигляд без складних змін у тексті.

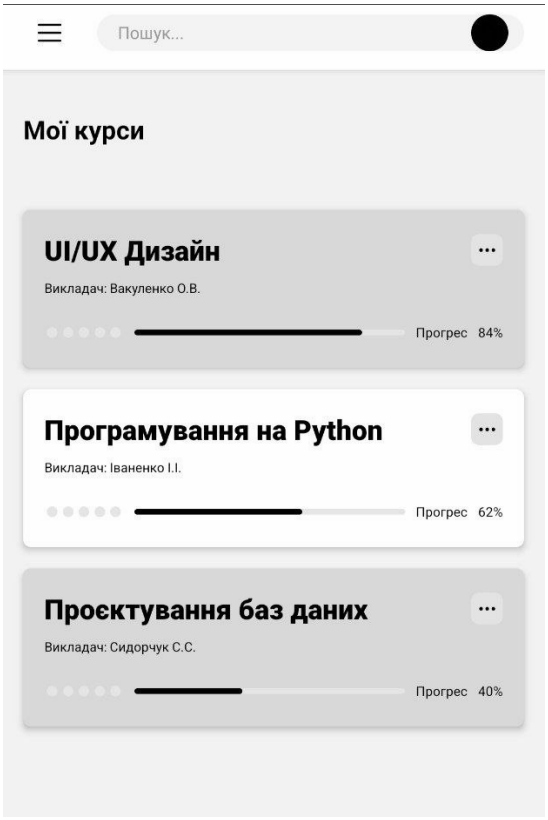


Рисунок 3.11 – Рендер сторінки з курсами

Блок «Невиконані завдання» є ще більш функціональним: він стимулює студентів завершувати завдання вчасно. У ньому відображаються лише ті завдання, які мають дедлайн у майбутньому та ще не були позначені як виконані. Кожен елемент завдання містить: назву завдання; курс, до якого воно належить; дедлайн; короткий опис (опційно); кнопку переходу до повного перегляду завдання.

```
24  const Tabs = ['Не виконані', 'На перевірці', 'Виконані'];
25
26  const TaskCard = ({ task, onPress }: { task: Task; onPress: () => void }) => (
27    <TouchableOpacity onPress={onPress} style={styles.taskCard}>
28      <View style={styles.taskTop}>
29        <Text style={styles.subject}>{task.subject}</Text>
30        <Text style={styles.titleText}>{task.title}</Text>
31      </View>
32      <View style={styles.taskBottom}>
33        <Text style={styles.date}>Видано: {task.issuedAt}</Text>
34        <Text style={styles.date}>Дедлайн: {task.deadline}</Text>
35        <Text style={styles.due}>До здачі: {task.dueIn}</Text>
36      </View>
37    </TouchableOpacity>
38  );
```

Рисунок 3.12 – Уривок коду з реалізацією сортування завдань по дедлайнах

Обидва блоки не просто інформують, а створюють зручну точку входу для навігації. Натискання на картку курсу відкриває сторінку з деталями курсу, а натискання на завдання – екран TaskDetailsScreen, де студент може прочитати повну інформацію та завантажити файл з виконаною роботою. Це дозволяє користувачу взаємодіяти з додатком у максимально простий і швидкий спосіб, що особливо важливо для мобільного середовища, де увага користувача розсіяна.

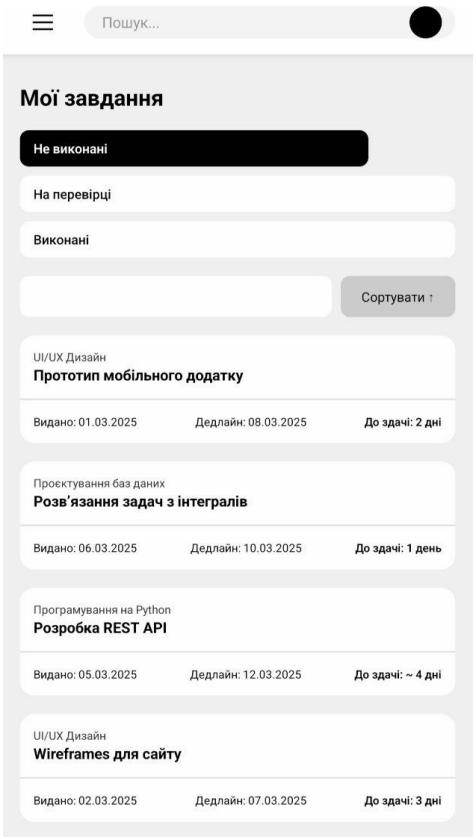


Рисунок 3.13 – Рендер завдань, включно з виконаними і невиконаними

3.5 Тестування мобільного додатку

Тестування є невід’ємною частиною життєвого циклу програмного забезпечення, яке дозволяє виявити помилки, перевірити коректність реалізованого функціоналу та забезпечити надійну роботу додатку в різних умовах. У процесі розроблення мобільного додатку LMS-системи було застосовано переважно ручне функціональне тестування з елементами інструментального, спрямованого на перевірку інтерфейсу та взаємодії користувача з системою.

Основною метою тестування було перевірити:

- чи відповідає додаток вимогам, визначеним на етапі проєктування;
- наскільки стабільно працюють основні функціональні блоки;
- як виглядає і функціонує інтерфейс на реальних пристроях;

- чи не виникає збоїв під час переходу між екранами;
- як працює передача параметрів (наприклад, при переході до деталей завдання);
- чи зручно взаємодіяти з елементами інтерфейсу (кнопки, списки, прокрутка).

Для тестування використовувались такі підходи та засоби:

– Реальні пристрої Android для ручного тестування на фізичних смартфонах.

– React Native Debugger та Chrome DevTools для виявлення помилок, журналювання подій та моніторингу стану компонентів.

– Browserstack для виконання UI – тестування.

Під функціональним тестування вручну мається на увазі проходження всіх сценаріїв користувача з перевіркою результатів. Приклади основних сценаріїв наведено в таблиці 3.1, які були протестовані в додатку.

Таблиця 3.1 – Виконання ручного тестування

Назва сценарію	Опис дії	Очікуваний результат	Результат	Статус
Завантаження головного екрану	Відкрити додатку	Головна сторінка відображається без затримок, прогрес оновлюється згідно з датою	Збігається	Успішно
Перегляд курсів	Натиснути на блок «Мої курси»	Відображається список курсів, перехід на екран курсу	Збігається	Успішно
Перегляд невиконаних завдань	Натиснути на блок завдань → вибрати конкретне	Відкривається список → екран деталей завдання	Збігається	Успішно
Завантаження виконаного завдання	На екрані деталей натиснути «Завантажити файл»	Можливість вибрати файл та прив’язати його до завдання (імітація без бекенду)	Збігається	Успішно
Повернення назад	Натиснути кнопку «Назад» на будь-якому екрані	Повернення на попередній екран без втрати стану	Збігається	Успішно

Усі основні сценарії були успішно відтестовані. Додаток стабільно працював на кількох пристроях із різними версіями Android, не виявлено критичних помилок, які блокували б функціональність. Виявлені незначні візуальні недоліки були оперативно усунені (наприклад, некоректне відображення деяких блоків на екранах із меншою шириною).

Виконання навантажувальних тестів та швидкодії додатка прошли успішно основуючись на швидкодії додатку. Показ FPS Frames Per Second (кадрів за секунду) наведено на рисунку 3.14.

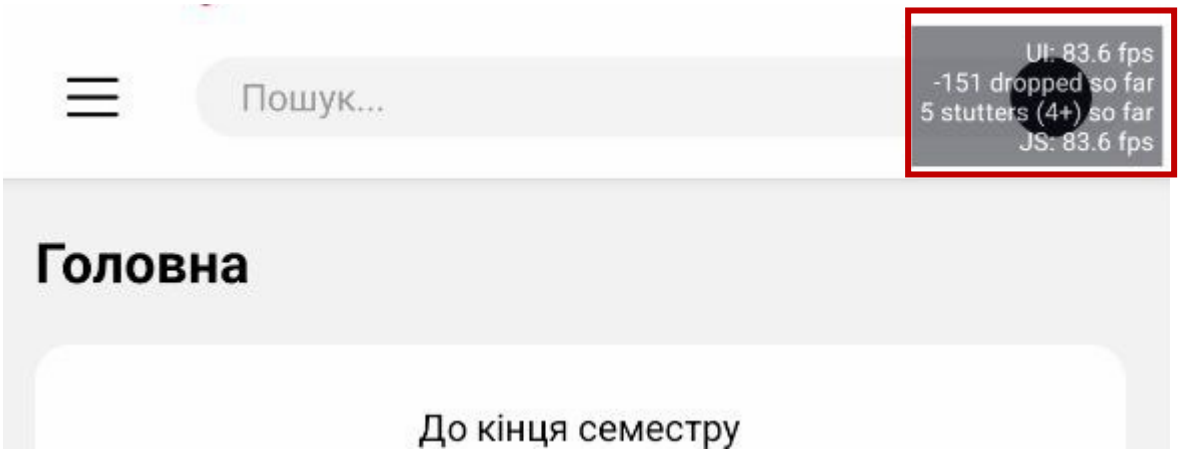


Рисунок 3.14 – показ швидкодії

Далі для тестування мобільного додатку, було вирішено використати хмарну мережу browserstack (вигляд в додатку Ж).

У третьому розділі дипломної роботи було детально розглянуто процес розробки мобільного додатку LMS-системи для студентів і викладачів. Було описано, як на основі Figma-дизайну реалізовано ключові екрани та функціональні блоки додатку з використанням технологій React Native, TypeScript та бібліотеки NativeWind для стилізації.

Особливу увагу приділено структурі проєкту, правильній організації навігації між екранами та динамічній роботі з даними. Зокрема, було реалізовано головний екран з дугоподібним індикатором прогресу до завершення семестру, функціональні блоки «Мої курси» та «Невиконані завдання», а також екран з деталями завдань, де користувач може завантажити файл із виконаною роботою.

Використання сучасних підходів до розробки мобільних застосунків дозволило створити інтерфейс, зручний для користувача, з якісним візуальним оформленням, адаптованим до потреб цільової аудиторії – студентів і викладачів.

Завдяки чіткому поділу компонентів, централізованій навігації, повторному використанню елементів інтерфейсу та модульному підходу до коду, проєкт залишився гнучким для подальшого масштабування й розширення. Це свідчить про продуману архітектуру та правильний вибір інструментів.

ВИСНОВКИ

У ході виконання кваліфікаційної роботи було пройдено повний цикл створення мобільного додатку LMS-системи для студентів і викладачів, починаючи з аналізу наявних рішень та вибору відповідних інструментів розробки, та закінчуючи реалізацією основного функціоналу, тестуванням та оформленням результатів.

На етапі аналітичного огляду були досліджені популярні бібліотеки та фреймворки для розробки мобільних застосунків. Зокрема, проаналізовано можливості React Native та його переваги у порівнянні з альтернативами. Також було порівняно дизайн мобільних та настільних рішень LMS, що дозволило сформувати власне бачення структури та функціональності майбутнього додатку.

У другому розділі здійснено аналіз технічних рішень, реалізацію навігації між екранами, побудову ключових UI-компонентів та описано архітектуру додатку. Враховуючи відсутність бекенду на момент розробки, функціональність була зосереджена на візуалізації даних та взаємодії між інтерфейсними елементами.

У третьому розділі представлено покроковий процес розробки, включаючи реалізацію головного екрану з динамічним відображенням залишку часу до кінця семестру, створення екранів курсів і завдань, а також передавання параметрів між екранами. Особлива увага приділялася зручності навігації та відповідності сучасним вимогам до UI/UX.

Після реалізації функціоналу було проведено ручне тестування додатку на різних пристроях. За результатами тестування підтверджено працездатність усіх основних компонентів додатку, а також стабільність його роботи. Виявлені дрібні помилки були усунені в процесі доопрацювання.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Statista. Mobile app usage statistics and trends. URL: https://www.statista.com/statistics/ (дата звернення: 06.07.2025)
2. MarketsandMarkets. Learning Management System Market - Global Forecast to 2028. URL: https://www.marketsandmarkets.com/Market-Reports/learning-management-system-market-204953789.html/ (дата звернення: 06.07.2025)
3. React Native official documentation. URL: https://reactnative.dev/ (дата звернення: 06.07.2025)
4. Redux Toolkit documentation. URL: https://redux-toolkit.js.org/ (дата звернення: 06.07.2025)
5. React Navigation documentation. URL: https://reactnavigation.org/ (дата звернення: 06.07.2025)
6. Axios documentation. URL: https://axios-http.com/ (дата звернення: 06.07.2025)
7. Moodle official website. URL: https://moodle.org/ (дата звернення: 06.07.2025)
8. Blackboard official website. URL: https://www.blackboard.com/ (дата звернення: 06.07.2025)
9. Canvas LMS official website. URL: https://www.instructure.com/canvas/ (дата звернення: 06.07.2025)
10. Google Classroom overview. URL: https://edu.google.com/products/classroom/ (дата звернення: 06.07.2025)
11. Офіційна документація React Native Paper. URL: https://reactnativepaper.com (дата звернення: 06.07.2025)
12. Material Design 3 Guidelines. URL: https://m3.material.io (дата звернення: 06.07.2025)
13. React Navigation Documentation. URL: https://reactnavigation.org (дата звернення: 06.07.2025)
14. NativeWind Official Docs. URL: https://nativewind.dev (дата звернення:

06.07.2025)

15. Tailwind CSS Configuration Reference. URL: https://tailwindcss.com/docs/configuration (дата звернення: 06.07.2025)
16. WCAG 2.1 Accessibility Guidelines. URL: https://www.w3.org/WAI/standards-guidelines/wcag (дата звернення: 06.07.2025)
17. Figma Community – Education App Templates. URL: https://www.figma.com/community (дата звернення: 06.07.2025)
18. React Native Performance Optimization. URL: https://reactnative.dev/docs/performance (дата звернення: 06.07.2025)
19. State Management Comparison. URL: https://redux-toolkit.js.org/introduction/comparison (дата звернення: 06.07.2025)
20. Educational Technology Trends 2024. URL: https://www.edsurge.com/research (дата звернення: 06.07.2025)
21. Mobile Learning Statistics. URL: https://www.statista.com/topics/3112/mobile-learning (дата звернення: 06.07.2025)
22. LMS Market Research. URL: https://www.researchandmarkets.com/reports/lms-market (дата звернення: 06.07.2025)
23. React Native Documentation. URL: https://reactnative.dev/docs/getting-started (дата звернення: 06.07.2025)
24. React Navigation Documentation. URL: https://reactnavigation.org/docs/getting-started (дата звернення: 06.07.2025)
25. NativeWind Documentation. URL: https://nativewind.dev/ (дата звернення: 06.07.2025).
26. TypeScript Official Website. URL: https://www.typescriptlang.org/docs/ (дата звернення: 06.07.2025)
27. React Native Circular Progress Library. URL: https://github.com/bartgryszko/react-native-circular-progress (дата звернення: 06.07.2025)
28. React Native SVG Library. URL: https://github.com/react-native-svg/react-native-

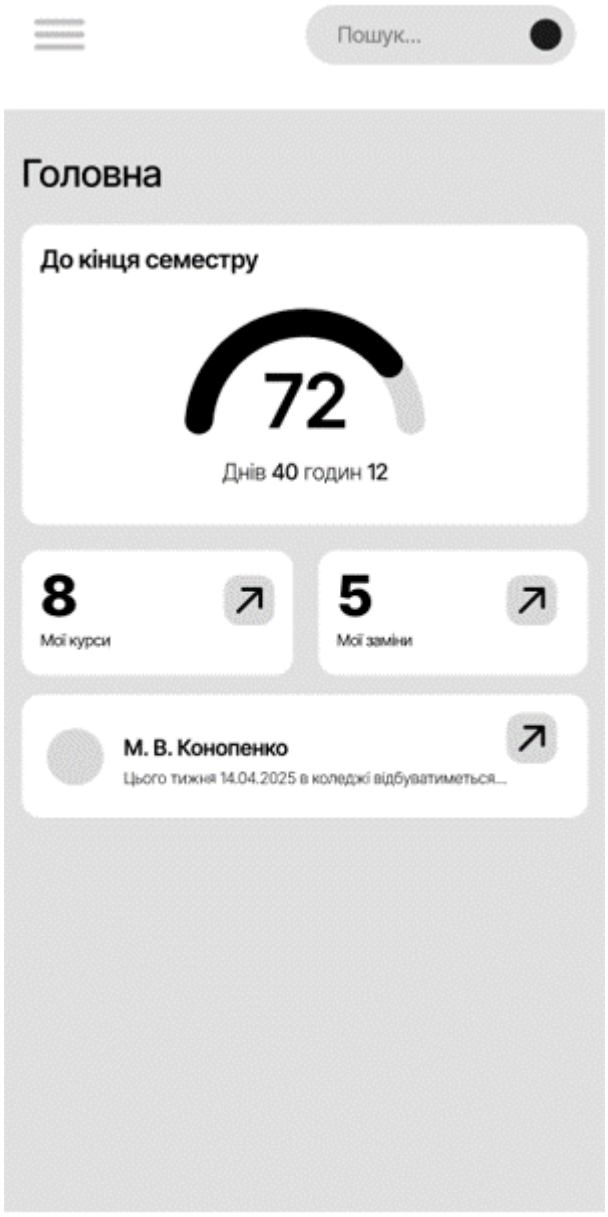
svg (дата звернення: 06.07.2025)

29. Testing React Native Apps – Official Guide. URL: https://reactnative.dev/docs/testing-overview (дата звернення: 06.07.2025)
30. Android Studio Emulator Guide. URL: https://developer.android.com/studio/run/emulator (дата звернення: 06.07.2025)
31. React Native Debugger. URL: https://github.com/jhen0409/react-native-debugger (Дата звернення: 06.07.2025)
32. Browserstack, сервіс тестування додатків. URL: https://app-live.browserstack.com (дата звернення 06.07.2025)

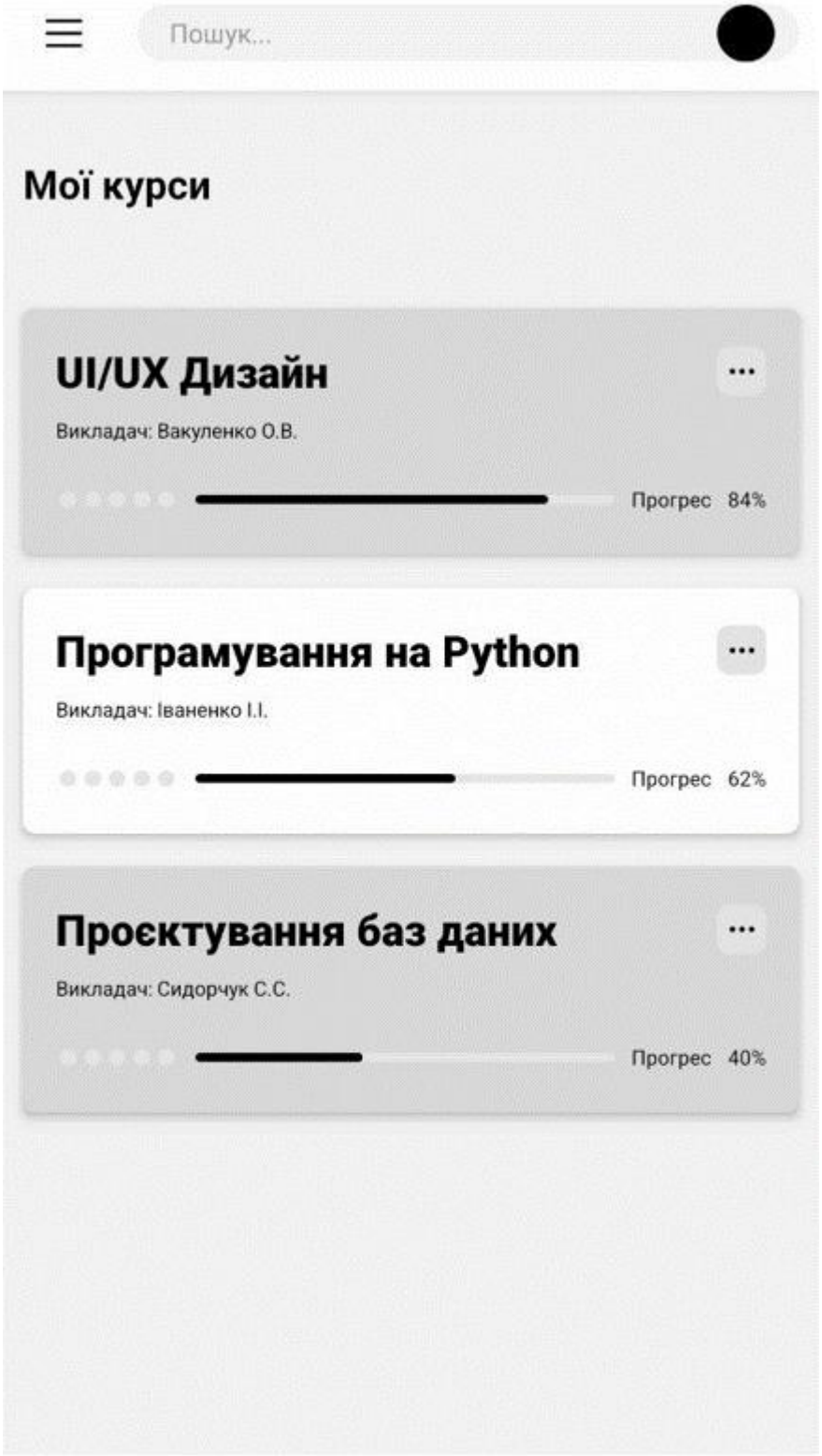
ДОДАТОК А – Посилання на первинний код мобільного додатка «LMS системи»

Первинний код розробленого програмного засобу для обчислення метрик програмного коду, а також проведення статистичного аналізу результатів розміщено в git-репозиторії за адресою: https://github.com/Temrario/LMSApp.

ДОДАТОК Б – Дизайн головної сторінки у Figma



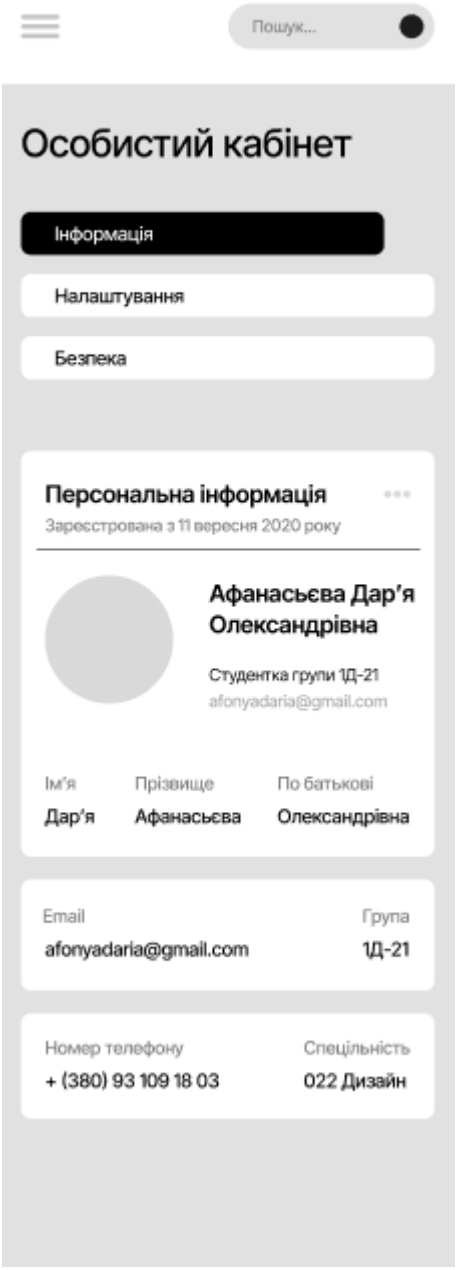
ДОДАТОК В – Дизайн сторінки з курсами у Figma



ДОДАТОК Г – Дизайн сторінки з інформацією про даний курс у Figma

ДОДАТОК Д – Дизайн сторінки з завдання ми у Figma

ДОДАТОК Е – Дизайн сторінки з особистим кабінетом у Figma



ДОДАТОК Ж – Показ логів при використанні тестування на онлайн ресурі