ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра Інженерії програмного забезпечення

Пояснювальна записка

до бакалаврської роботи
на ступінь вищої освіти бакалавр

на тему: «**РОЗРОБКА МОБІЛЬНОГО ДОДАТКУ ДЛЯ СИСТЕМИ ДИСТАНЦІЙНОГО НАВЧАННЯ МОВОЮ C#**»

Виконав: студент 4 курсу, групи ПД-44
спеціальності
121 Інженерія програмного забезпечення
(шифр і назва спеціальності/спеціалізації)

Медко М.М.
(прізвище та ініціали)

Керівник Поперешняк С.В.
(прізвище та ініціали)

Рецензент
(прізвище та ініціали)

Київ – 2023

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра Інженерії програмного забезпечення

Ступінь вищої освіти – «Бакалавр»

Спеціальність підготовки – 121 «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ

Завідувач кафедри

Інженерії програмного забезпечення

_____Негоденко О. В.

“___” _____2023 року

ЗАВДАННЯ

НА БАКАЛАВРСЬКУ РОБОТУ СТУДЕНТА

МЕДКА МИКИТИ МИХАЙЛОВИЧА

(прізвище, ім’я, по батькові)

1. Тема роботи: «Розробка мобільного додатку для системи дистанційного навчання мовою C#»

Керівник роботи: Поперешняк С.В. зав.кафедри, к.ф.-м.н, доцент

(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом вищого навчального закладу від «24» лютого 2023 року №26

2. Строк подання студентом роботи « 1 » червня 2023 року
3. Вхідні дані до роботи: матеріали перед-дипломної практики, принцип взаємодії клієнта з сервером, документація для взаємодії з URLS API, науково-технічна література з питань, пов’язаних з розробкою мобільного

4. додатку з використанням різних сучасних патернів, засобів та інструментів розробки.
5. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити).
   5.1. Аналіз предметної області
   5.2. Вимоги та оцінка якості системи
   5.3. Проектування та реалізація мобільного додатку
   5.4. Тестування системи та отримання результатів
6. Перелік демонстраційного матеріалу
   6.1. Мета, об’єкт предмет дослідження
   6.2. Задачі дипломної роботи
   6.3. Аналіз існуючих програмних засобів
   6.4. Вимоги до додатку
   6.5. Програмні та технічні засоби реалізації
   6.6. Діаграма варіантів використання (реєстрація)
   6.7. Діаграма варіантів використання (функціонал адміністратора)
   6.8. Діаграма послідовності (прийняття запиту на приєднання)
   6.9. Діаграма послідовності (функціонал студенту)
   6.10. Діаграма класів
   6.11. Діаграма пакетів
   6.12. Діаграма архітектури системи
   6.13. Графічний інтерфейс
   6.14. Графічний інтерфейс
   6.15. Апробація результатів дослідження
   6.16. Висновки

7. Дата видачі завдання «25» лютого 2023 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів бакалаврської роботи	Строк виконання етапів роботи	Примітка
1	Підбір науково-технічної літератури	25.02.2023 – 27.02.2023	Виконано
2	Аналіз науково-технічної літератури	28.02.2023	Виконано
3	Вступ	01.02.2023	Виконано
4	Аналіз існуючих програмних засобів	03.02.2023	Виконано
5	Проектування системи	15.02.2023	Виконано
6	Створення та тестування програмного рішення	05.03.2023	Виконано
6	Підготовка розділу 1	10.04.2023	Виконано
7	Підготовка розділу 2	15.04.2023	Виконано
8	Підготовка розділу 3	26.04.2023	Виконано
9	Підготовка розділу 4	07.05.2023	Виконано
10	Висновки, реферат	10.05.2023	Виконано
11	Розробка обов’язкових демонстраційних матеріалів	17.05.2023	Виконано
12	Попередні захист роботи	23.05.2023	Виконано
13	Здача роботи	01.06.2023	

Студент _____ _____
(підпис) (прізвище та ініціали)

Керівник роботи _____ _____
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Текстова частина бакалаврської роботи 64 с., 32 рис., 9 табл., 19 джерел.

МОБІЛЬНИЙ ДОДАТОК, C#, .NET MAUI, MVVM, REPOSITORY, ANDROID, MOODLE, GOOGLE CLASROOM, KIDDOM, ДОДАТОК ДЛЯ ДИСТАНЦІЙНОГО НАВЧАННЯ.

Об'єкт дослідження – процес дистанційного навчання за допомогою мобільного додатку.

Предмет дослідження – мобільний додаток для забезпечення доступності функцій дистанційного навчання.

Мета роботи – спрощення використання та покращення доступності окремих функцій навчального процесу для студента, за допомогою мобільного додатку на платформі .NET MAUI, мовою програмування C#.

Методи дослідження – уніфікований процес розробки додатку, методи надсилання сповіщень, збереження даних, портативність .

Наукова новизна – інтеграція з розширеними можливостями, існує система керування сесіями, автоматичне завантаження файлів, яке відсутнє у аналогів. Адаптація теми застосунку та реалізація шаблонів і функції керування особистими сесіями на різних пристроях. Використання розкладу, перегляд предметів та взаємодія з ними, інформація про університет та можливість керувати власним профілем і додатком в одному застосунку, допомагає студенту у навчальному процесі.

Мобільний додаток складається з розподіленої три-рівневої архітектури MVVM та Repository, які забезпечують ефективну розробку мобільного додатку, з можливістю розділення інтерфейсу користувача, бізнес логікою та роботу з даними.

Розроблений мобільний додаток дозволяє підвищити та спростити навчальний процес, за допомогою концентрації функцій навчального процесу.

Галузь використання – освітній процес.

Зміст

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

URLS mobile – Ukrainian remote learning system mobile (мобільний додаток української системи дистанційного навчання).

UI – user interface.

ДН – дистанційне навчання.

СДН – система дистанційного навчання.

БД – база даних.

СУБД – система управління базами даних.

ОС – операційна система.

API – application programming interface.

FCM – Firebase Cloud Messaging.

GCM – Google Cloud Messaging.

CTA – Call to action.

ООП – об’єктно орієнтоване програмування.

TDD – test-drive development.

ВСТУП

Навчання та здобуття освіти на сьогодні, як і раніше – має чітку позицію у світі. Освіта і інформаційні технології тісно переплелися між собою, оскільки у поєднанні виходить гарний результат. Але у потоці та стрімкому розвитку інформаційних технологій, кількість інформації та різноманіття технічних рішень перенасичує ринок. Гостро постає проблема зберігання, керування та обробки інформації та даних. Особливо це стосується учнів та студентів, які фактично цим живуть під час здобуття освіти різного рівня. Сучасні технології дозволяють дещо змінити підхід до навчання, одне з яких є впровадження дистанційної форми навчання.

Дистанційне навчання – це ряд технологій, які дозволяють студентам отримати доступ до основної частини їхньої навчального контенту, з використанням різних телекомунікаційних засобів: інтерактивна взаємодія між студентом та викладачем під час процесу навчання, надання студентам власних можливостей у розборі та оброці навчальних матеріалів.

Суттєвих складнощів до навчального процесу додає сукупність різних глобальних проблем і постійних невирішених питань. Як приклад, пандемія, війна, географічна віддаленість та рівень розвитку освіти в кожній країні.

Через пандемію COVID-19 усі учбові заклади перейшли на повне, часткове або змішану форму дистанційного навчання. Учні, студенти та їх викладачу були змушені навчатися з дому, а частина викладачів мала відвідувати робочі місця і продовжувати процес надання освіти. Однак освітня система не була повноцінно готова до таких кардинальних змін. Як студенти так і викладачі опинилися у скрутних умовах. Багато було проблем з рівнем оснащення кожного учасника освітнього процесу.

Найважчим випробуванням стала комунікація між учнем та викладачем, обміном інформацією та наданням матеріалів, оскільки не було уніфікованого

середовища проведення навчання в умовах пандемії. Зрозуміло, що дистанційна форма навчання має свої особливості та проблеми, які виражені у: вибір інструменту(платформи) для проведення занять, зберігання робіт, отримання матеріалів, здачу робіт і місце зустрічі групи з своїм викладачем. Важливим фактором у ДН є самоосвіта і самоорганізація.

Описані труднощі у ДН можна вирішити за допомогою комплексної системи або набору модулів, які зможуть забезпечити необхідний функціонал для підвищення якості проведення навчання у дистанційній формі. На ринку існує багато готових рішень але вони всі різні і не вирішують всі завдання в комплексі.

Для прикладу можна розглянути продукт від компанії Google, продукт під назвою Google Classroom. Ще є відома платформа з відкритим вихідним кодом – Moodle. Дані програмні засоби мають свої переваги і недоліки, через які, їх використання не може повністю задовольнити покриття проблем дистанційного навчання.

Тому призначення мобільного застосунку, що розробляється є створення зручної системи, з широким спектром функціоналу і новими технічними рішеннями, які можуть задовольнити проблеми користувачів.

1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

Процес аналізу предметної області є першим важливим етапом у розробці програмного забезпечення. Якщо предметною областю додатку, який створюється, є розробка мобільного додатку для системи дистанційного навчання, необхідно знайти аналоги та проаналізувати їх функціонал, переваги та недоліки за галуззю використання додатків.

1.1. Поняття та особливості дистанційного навчання

Впровадження дистанційного навчання через різні причини, стало новою гілкою в розвитку системи освіти у світі. Напрямок дистанційного навчання є однією із найбільш швидкозростаючих напрямів в освіті та потенційним важелем впливом на сферу освіти в цілому.

Сам термін «Дистанційне навчання» розкриває підходи, націлені на відкриття доступу до освіти та навчання, знімаючи обмеження місця і часу отримання знань.

Основна задача дистанційного навчання є створення інтерактивної і асинхронної взаємодії викладача та студента без їх фізичної зустрічі, самостійне опанування матеріалу та самодисципліну. Для досягнення максимально ефективного результату процесу навчання, необхідно створити спеціальні умови: сприятлива атмосфера, гнучкий підхід до кожного студента і всього процесу ДН в цілому.

Для задоволення вимог і покриття проблем у такий мінливий час, майбутній додаток ДН повинен бути гнучким, мати конкурентоспроможній функціонал та ціну, чіткі кроки та інструкції вирішення нагальних питань та можливість задовольнити зацікавлених осіб.

1.2. Методи дистанційного навчання

Історія започаткування форми дистанційного навчання тягнеться з 18 століття, коли існувала форма надання освіти за допомогою пошти. В той час була ідея створити форму навчання без викладача, через проблеми географічної відстані, якості і доступності освіти в різних країнах світу. Згодом цей напрям підтримували все більше і більше людей. Доходило навіть до державної підтримки та початку створення програм при університетах. З того часу багато що змінилося, з чим інформаційні технології відіграють лідируючу роль у Всесвітній павутині.

Існує «Положення про дистанційне навчання» в Україні, яке описує процес організації дистанційного навчання в асинхронному та синхронному режимі.

Асинхронний режим – це процес в якому відбується взаємодія суб'єктів дистанційного навчання, протягом якого учасники взаємодіють між собою і мають затримку у часі (один з суб'єктів не прочитав повідомлення одразу). В даному режимі використовуються такі інтерактивні освітні платформи: email, moodle (у деяких випадках), різноманітні форуми та соціальні мережі тощо. Даний формат є зручним для багатьох освітян, оскільки він не зобов'язує студента та викладача бути прив'язаним до часу проведення заняття. Але даний режим навчання потребує великої самодисципліни, самоорганізації та головне – відповідальності кожної сторони. Оскільки питання, які виникнуть, далеко не завжди можуть бути вирішені швидко, а отже можлива втрата відсотку ефективності навчання. Даний формат ще може створити відчуття ізольованості, але в той же час і самостійності.

Синхронний режим – це процес взаємодії суб'єктів дистанційного навчання, протягом якого учасники взаємодіють між собою одночасно, перебуваючи в одному електронно-освітньому середовищі або використовують засоби аудіо-, відеоконференції тощо. При використання даного режиму навчання, можна ефективно побудувати режим роботи за допомогою вище описаних платформ, де робота буде відбуватися у групах, обговорення питань та лекційних матеріалів в реальному часі. Також даний формат дає можливість викладачу контролювати

процес засвоєння інформації, робити корегування в подачі матеріалу або проводити опитування, написання тестів з моментальною перевіркою результатів.

Гібридний режим – це процес взаємодії суб'єктів дистанційного навчання, протягом якого здобувач освіти може комбінувати синхронний та асинхронний режим, підкріплюючи або повністю заміняючи – курсами. Навчальні курси, це збір лекційних-, відеоматеріалів та тестів, які можуть перевірити рівень підготовки користувача після опанування певного матеріалу. Такий тип ще називають "неформальна освіта". Змішана форма навчання має недолік, у складності створення коректного розкладу занять, постійна довготривала підготовка до заняття та велика кількість матеріалу, який підлягає вивченню та практиці.

Отже, дистанційне навчання має свої особливості, фактори які можуть як покращити, так і погіршити процес дистанційного навчання. Освоєння інформації у вільному графіку з дисциплінованим підходом, використання сучасних освітніх технологій, доступність будь-якому прошарку населення і відсутність фізичного відвідування заняття, лекцій або семінарів – це є неоціненим плюс. А недосконале володіння тією чи іншою функцією, системою, ускладнена ідентифікація студента на заняттях або відсутністю розуміння та повної віддачі процесу з будь якої сторони об'єктів СДН – може призвести до небажаних результатів.

1.3. Огляд існуючих сучасних рішень для дистанційного навчання

Для форми дистанційного навчання є характерними майже всі варіанти навчального процесу компоненти системи, а саме:

1. Цілі.
2. Зміст.
3. Засоби навчання.
4. Система контролю.
5. Оцінки;

На сьогоднішній день вже створено багато освітніх платформ та систем, призначених для дистанційного навчання, здобувачів різного рівня освіти, які включають в себе вище описані компоненти системи:

1. Дистанційні курси.
2. Соціальні мережі.
3. Електронна пошта.
4. Відео- і аудіо-конференції.
5. Різні форуми;

1.3.1 Moodle

Одна із найпоширеніших системи дистанційного в Україні, і в світі яка займає не останню позицію, є Moodle (Modular Object-Oriented Dynamic Learning Environment). Дане рішення є безкоштовним і модульним об’єктно-орієнтованим навчальним середовищем, яке використовує динамічну систему в побудові навчального процесу.

Платформа забезпечує одночасний і надійний процес роботи як викладачів, так і студентів з навчальним матеріалом. Студенти мають можливість переглядати свої курси (див. рис. 1.1 і 1.2) та оцінки за них (див. рис. 1.3) опрацьовувати матеріали та проходити тести для отримання підсумкової оцінки. Створення груп, курсів та реєстрацію студентів та викладачів у переважній більшості робить адміністратор (див. рис. 1.4 і 1.5). В деяких випадках така можливість надається користувачу особисто. Система відноситься до типу LMS, що означає – Learning management system.

Країною розробки є Австралія, а головний розробник – Мартін Дугіамас. Платформа має відкритий вихідний код, тобто будь-яка людина (розробник, тестувальник тощо) може переглядати як влаштована система з середини або безпосередньо приймати участь у розробці. Moodle розроблено на мові програмування PHP, а MySQL виступає в якості СУБД.

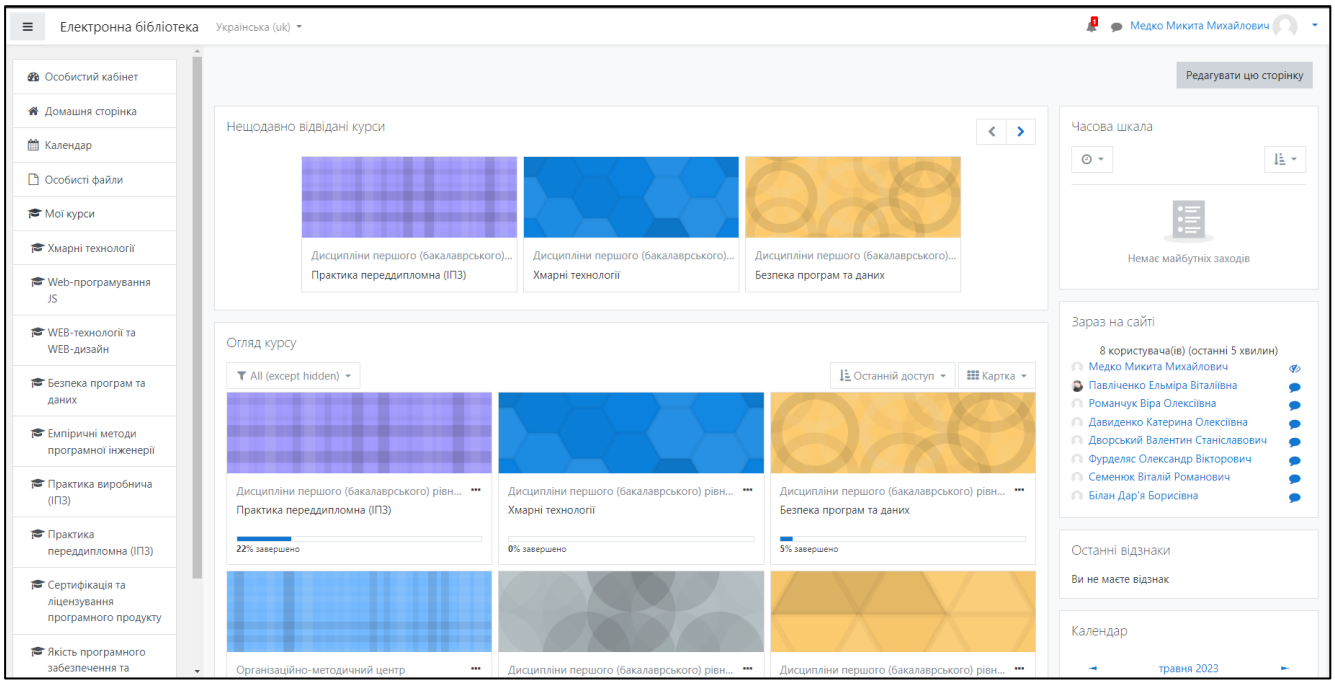


Рисунок 1.1 – Головна сторінка в системі Moodle (студент)

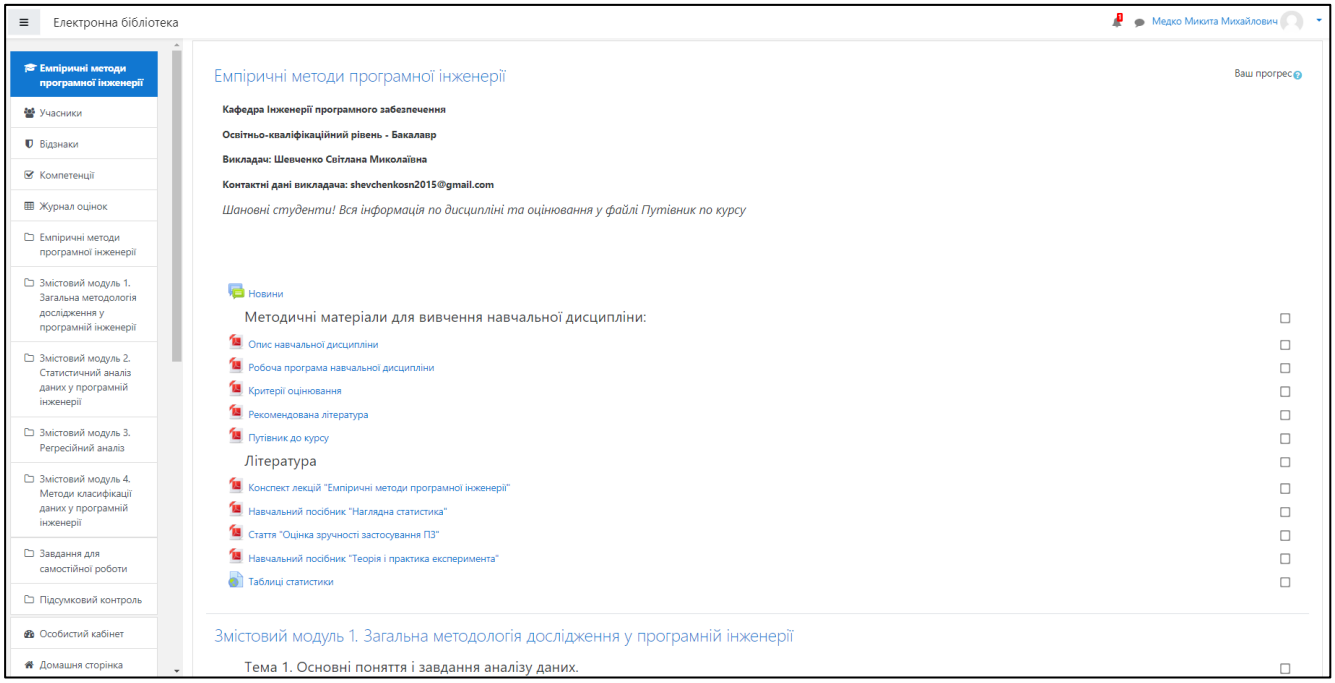


Рисунок 1.2 – Сторінка огляду курсу в системі Moodle (студент)

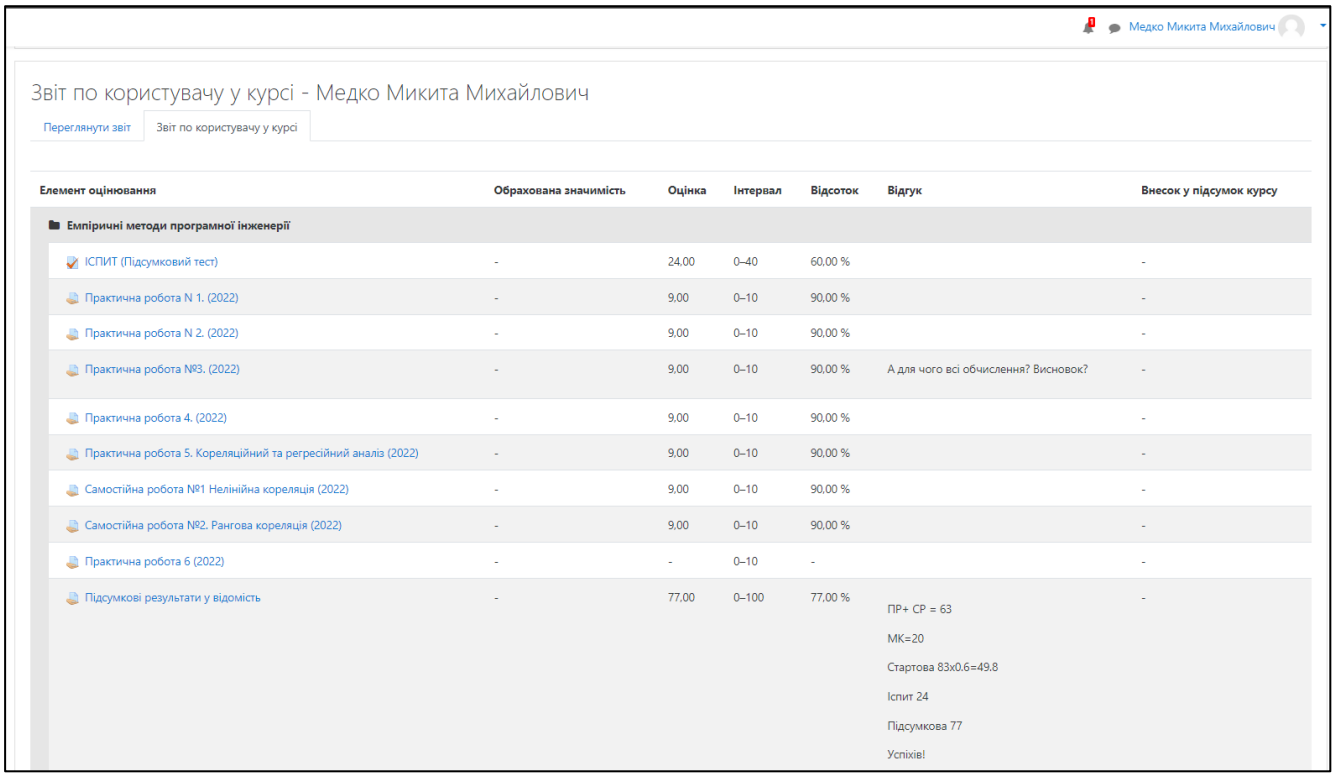


Рисунок 1.3 – Оцінки студента за курс в системі Moodle (студент)

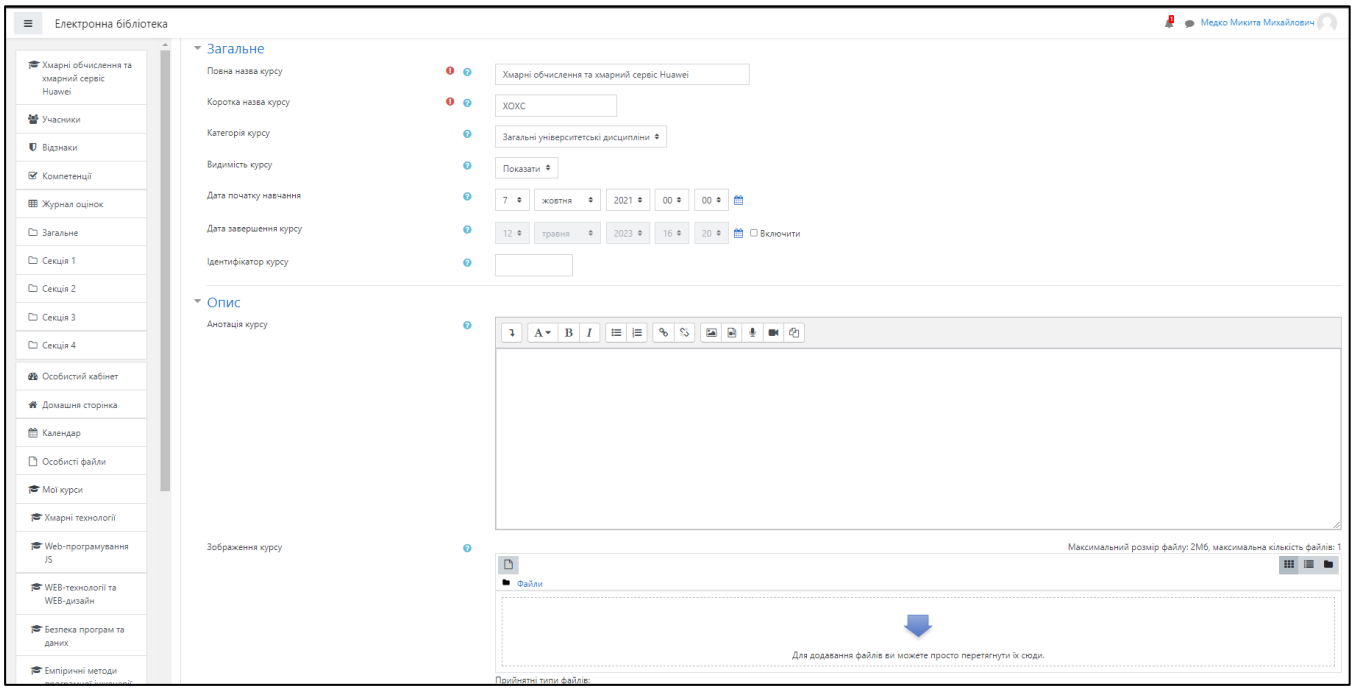


Рисунок 1.4 – Створення нового курсу в системі Moodle (адміністратор)

Рисунок 1.5 – Створення нового курсу в системі Moodle (адміністратор)

Переваги системи:

1. Присутня панель керування для адміністратора.
2. Присутня панель.
3. Є рівень розподілення користувачей по ролям.
4. Функція календаря.
5. Відкритий вихідний код.
6. Присутня підтримка плагінів.

Недоліки системи:

1. Відсутній мобільний додаток.
2. Система сповіщень є неповноцінною.
3. Платформа орієнтована більше на викладача.
4. Нема системи відслідковування рівня складності завдань та прогресу групи.
5. Відсутня система розкладу онлайн.
6. Відсутня підтримка різних сервісів електронної пошти, окрім домену Moodle.

1.3.2 Kiddom

Платформа Kiddom – це цифрове гібридне середовище, яке поєднує в собі компоненти оцінок, персоналізовану систему керування навчанням. Для викладача дана система є одним із найкращих рішень, коли треба мати інструменти аналізу та порівняння успішності студентів (див. рис. 1.6) або створення курсу для онлайн навчання, з використанням великої бази вже готових курсів та даних (див. рис. 1.7).

Студент має широкий спектр функціоналу для ефективного процесу дистанційного навчання. Середовище дає можливість обирати дату, час та місце проведення заняття, батьки на вимогу можуть отримати успішність студента. Функція звіту про виконану роботу та поради для наступних кроків у поєднанні аналізу звіту навичок студент створює сучасну концепцію в сфері навчання, з урахуванням використання технічних засобів. Засновником даної платформи є Ахсан Різві родом з США.

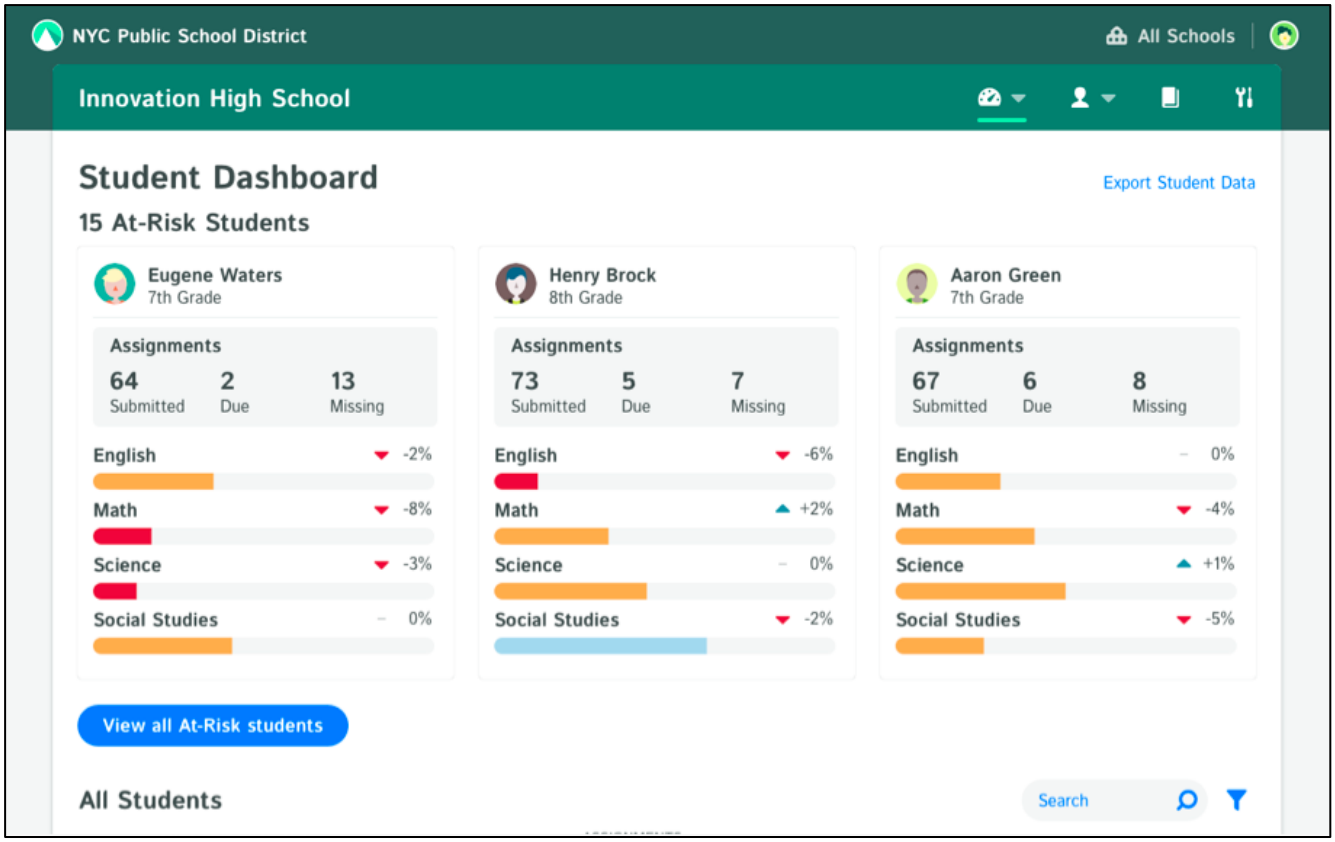


Рисунок 1.6. – Сторінка звіту успішності студентів в системі Kiddom (викладач)

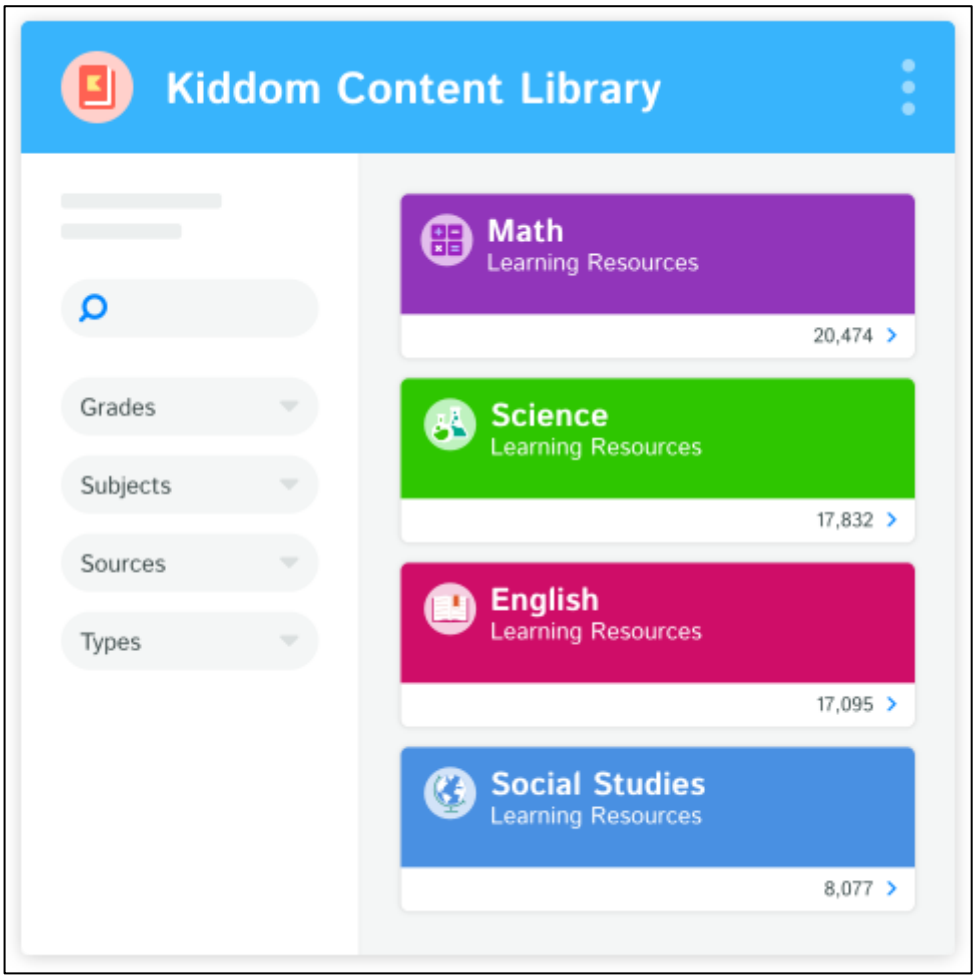


Рисунок 1.7 – Сторінка вибору джерел для створення курсу в системі Kiddom (викладач)

Переваги системи:

1. Платформа надає безкоштовний доступ для окремих викладачів.
2. Платформа має відкритий код.
3. Система створення звітів для студента і викладача є високоефективною.
4. Функція перегляду успішності школи або навчальної тенденції;

Недоліки системи:

1. Відсутність календаря занять.
2. Підтримує лише школу 12 класів.
3. Оплата онлайн погано реалізована.
4. Середовище важке у використанні.
5. Редагування вже готових курсів не є гнучким процесом.
6. Відсутня можливість робити курси по шаблону;

1.3.3 Google Classroom

Система від світової корпорації Google, що має назву – Goggle Classroom, яка призначена для керування контентом шкіл і окремих груп користувачів. Сервіс належить до великого набору компанії Google. Даний сервіс може бути інтегрований з наступними продуктами: Documents, Forms, Meet, Calendar і Goggle Drive. Через єдиний акаунт користувача у системі Google, використання Goggle Classroom є простим і зручним з поєднанням різних продуктів. Це створює ефективну еко-систему, яка може суттєво допомогти у вирішенні проблем дистанційного навчання.

Студент з головної сторінки (див. рис. 1.8) може перейти до завдань (див. рис. 1.9), подивитися список групи та їх оцінки (див. рис. 1.10) . Викладач може створювати курси, завдання (див. рис. 1.11) та блоки для лекційних матеріалів, робити призначення для виконання завдань.

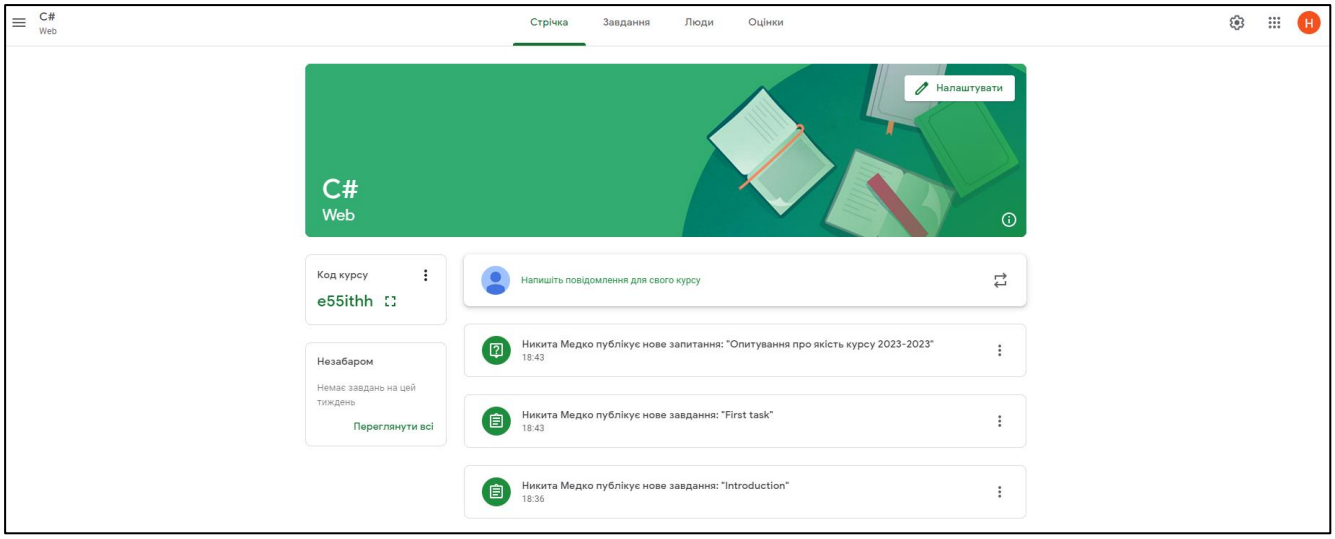


Рисунок 1.8 – Головна сторінка в системі Google Classroom (викладач)

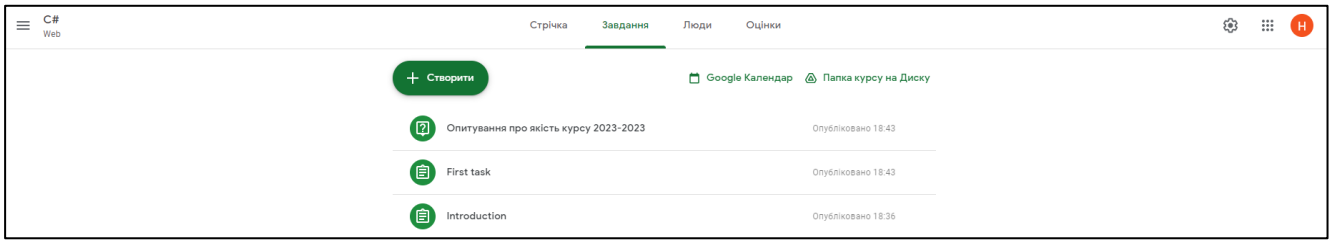
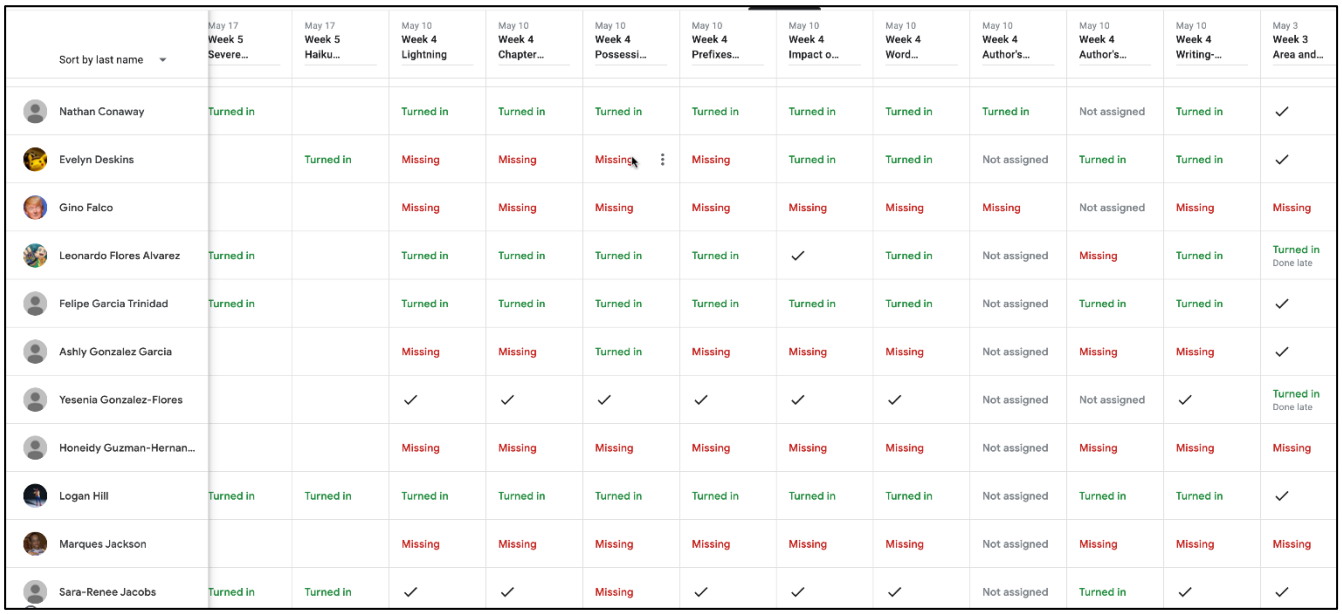


Рисунок 1.9 – Сторінка завдань в системі Google Classroom (студент)



Sort by last name	May 17 Week 5 Severe...	May 17 Week 5 Haiku...	May 10 Week 4 Lightning	May 10 Week 4 Chapter...	May 10 Week 4 Possessi...	May 10 Week 4 Prefixes...	May 10 Week 4 Impact o...	May 10 Week 4 Word...	May 10 Week 4 Author's...	May 10 Week 4 Author's...	May 10 Week 4 Writing-...	May 3 Week 3 Area and...
Nathan Conaway	Turned in		Turned in	Turned in	Turned in	Turned in	Turned in	Turned in	Turned in	Not assigned	Turned in	✓
Evelyn Deskins		Turned in	Missing	Missing	Missing	Missing	Turned in	Turned in	Not assigned	Turned in	Turned in	✓
Gino Falco			Missing	Missing	Missing	Missing	Missing	Missing	Missing	Not assigned	Missing	Missing
Leonardo Flores Alvarez	Turned in		Turned in	Turned in	Turned in	Turned in	✓	Turned in	Not assigned	Missing	Turned in	Turned in Done late
Felipe Garcia Trinidad	Turned in		Turned in	Turned in	Turned in	Turned in	Turned in	Turned in	Not assigned	Turned in	Turned in	✓
Ashly Gonzalez Garcia			Missing	Missing	Turned in	Missing	Missing	Missing	Not assigned	Missing	Missing	✓
Yesenia Gonzalez-Flores			✓	✓	✓	✓	✓	✓	Not assigned	Not assigned	✓	Turned in Done late
Honeidy Guzman-Hernan...			Missing	Missing	Missing	Missing	Missing	Missing	Not assigned	Missing	Missing	Missing
Logan Hill	Turned in	Turned in	Turned in	Turned in	Turned in	Turned in	Turned in	Turned in	Not assigned	Turned in	Turned in	✓
Marques Jackson			Missing	Missing	Missing	Missing	Missing	Missing	Not assigned	Missing	Missing	Missing
Sara-Renee Jacobs	Turned in	Turned in	✓	✓	Missing	✓	✓	✓	Not assigned	Turned in	✓	✓

Рисунок 1.10 – Сторінка оцінок в системі Google Classroom (викладач)

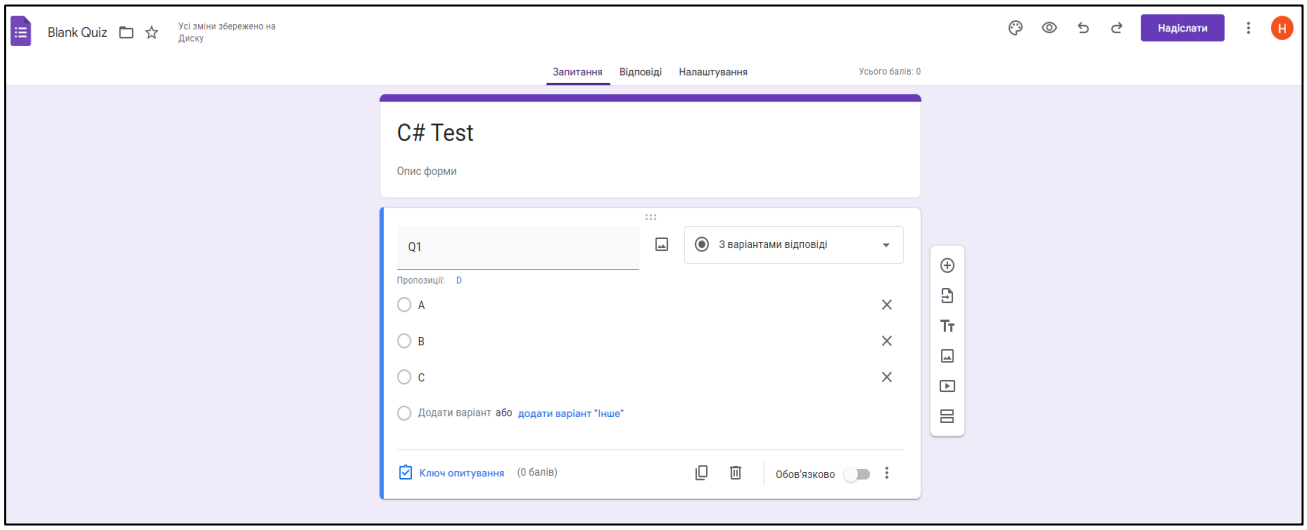


Рисунок 1.11 – Сторінка створення тесту в системі Google Classroom (викладач)

Переваги системи:

1. Має мобільний додаток.
2. Легкість у використанні. Як для студентів, так і для викладачів.
3. Має багато власних інтеграцій і може бути інтегрована.
4. Відкрите API.
5. Інтеграція з іншими продуктами Google.
6. Функція особистих та публічних коментарів.

7. Сповіщення відбуваються за допомогою пошти Google;

Недоліки системи:

1. Система сповіщення неповна для викладача.
2. Система має малу кількість функціоналу, недостатню для вирішення складних завдань.
3. Закритий вихідний код.
4. Панель керування адміністратора є недостатньо наповненою.
5. Немає функції розкладу занять.
6. Відсутній інструмент для модифікації формату курсу;

1.4. Висновки до розділу 1

По результатам проведеного аналізу конкурентів на ринку, можна зробити висновок, що усі вони в певних функціях мають свої переваги, але і присутні схожі проблеми, вирішивши які можна створити реально зручне і коректне системне рішення для викладачів і студентів. Склавши таблицю для порівняння аналогів з URLS (див. табл. 1), можна побачити загальну картину. В таблиці, знак ‘+’ – наявність функції, , ‘-’ – її відсутність, ‘+-’ – часткова наявність.

Таблиця 1 – порівняння аналогів URSL

Показник порівняння	Google Classroom	Kiddom	Moodle	Ukrainian Remote Learning System (URLS)
Платформи	Android, Web	Web, Android	Web	Android, iOS
Реєстрація	Тільки через сервіс Gmail	Реєстрація можлива з будь-якої електронної адреси	Процедуру реєстрації проводить система автоматично	Реєстрацію студент проводить самостійно (за допомогою коду приєднання)
Інтеграція в інші продукти	+	-	+	+

Запрошення до курсу	-	-	-	+
Перехід між акаунтами в рамках застосунку	+	-	-	+
Можливість експорту власних оцінок	-	-	-	+
Адаптація фону до теми дизайну пристрою	-	-	-	+
Відкритий код	-	-	+	+
Автоматичне завантаження файлів	-	-	-	+
Просте редагування предмету	+	-	+	+
Керування доступом до акаунту	+	-	+	+

Таким чином, провівши аналіз предметної області та склавши таблицю, можна стверджувати, що дана розробка даного додатку є актуальною. Обрані аналоги були з лідерів систем дистанційного навчання, які мають свої мобільні додатки.

2. ВИМОГИ ТА ОЦІНКА ЯКОСТІ СИСТЕМИ

Етап визначення і створення вимог в процесі розробки програмного забезпечення, знаходиться на другій сходинці по важливості, після створення ідеї та аналізу предметної області.

Оскільки існують різні сторони зацікавлених осіб у розробці чи використанні тієї чи іншої системи, необхідно розробити специфікацію вимог. Це процес, під час якого описується і створюється набір вимог до системи, до характеристики якості, до програмного забезпечення яке розробляється.

2.1. Функціональні вимоги URLS

Функціональні вимоги описують поведінку системи, її функції та конкретний результат, який система може виконувати в певний період часу. Вони описують поведінку системи та її функції, що вона може виконати та надати користувачу. Простими словам – це те, що система повинна робити.

2.1.1. Реєстрація

Новий користувач повинен мати можливість створити собі акаунт (зареєструватися, див. рис. 2.1) у мобільному застосунку та ввійти в свій акаунт. Для створення облікового запису користувач повинен заповнити наступні поля: ім’я, прізвище, по-батькові, логін, пароль, електрону адресу, номер свого телефону та код приєднання до групи.

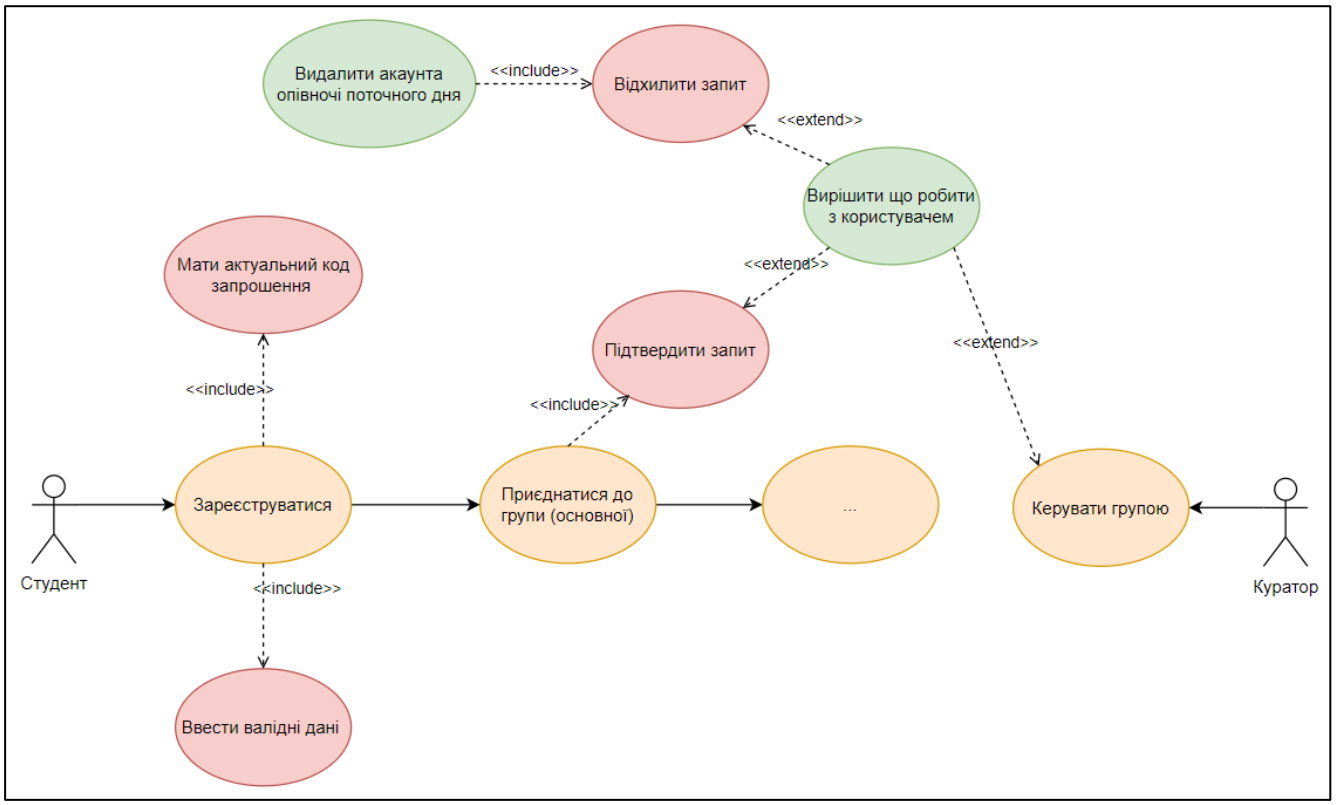


Рисунок 2.1 – діаграма використання (реєстрація користувача)

Процес реєстрації в системі повинен бути реалізований за допомогою класичного бачення даного кроку. За допомогою введення особистих даних, створення логін та паролю, і саме головне – введення наявного і актуального коду приєднання групи, яке при запиті до групи, буде бачити викладач. Саме він буде останнім об’єктом, який закінчує даний етап кроком – прийняти чи відхили запит користувача на приєднання до групи. Якщо куратор групи не бажає бачити даного студента в своїй групі, акаунт користувача (який ще не приєднався до групи), буде видалено до опівночі в день, коли куратор вирішив не приймати студента в групу.

2.1.2. Авторизація

Зареєстрований користувач повинен мати можливість авторизуватися за допомогою логіну та паролю. Якщо користувач забув логін або пароль, він може відновити їх за допомогою електронної адреси.

2.1.3. Обліковий запис користувача

Зареєстрований користувач у системі повинен мати наявний функціоналу для: зміни логіну, паролю, зміни та підтвердження електронної адреси, панель для попереднього перегляду профілю. Також у налаштуваннях користувач повинен мати функції для керування пристроями та сесіями на них.

2.1.4. Ролі в системі

В системі користувачі повинні мати розподіл на ролі, для отримання різних рівнів доступу для певних ситуацій. Ролі та їх можливості для користувача представлені у порівняльній таблиці (див. табл. 2). В системі повинна бути присутня гнучка система ролей.

Таблиця 2 – порівняння прав у системі, залежно від ролі

Показник порівняння	Адміністратор	Викладач	Студент
Отримання даних про сесії	+	+	+
Створення групи	+	+	-
Створення кодів приєднання	+	+	-
Перегляд кодів приєднання	+	+	+
Створення додаткових ролей	+	-	-
Створення ролі	+	-	-
Перегляд оцінок групи	+	+	-
Створення тестів	+	+	-
Створення/редагування шаблону предмету	+	+	-
Редагувати інформацію власного акаунту (username, сповіщення, пошту, пароль та номер телефону)	+	+	+
Створення/редагування оголошень	+	+	-

Через те, що розробляємий додаток призначений для системи дистанційного навчання, він містить різні ролі та їх можливості (згідно наведеної вище таблиці).

Оскільки всі ролі треба розуміти і мати уявлення, що може робити той чи інший користувач в системі, необхідно навести явні функції**.** Для розуміння доступних функцій адміністратора та студента, було створено діаграму використання (див. рис. 2.2 та 2.3).

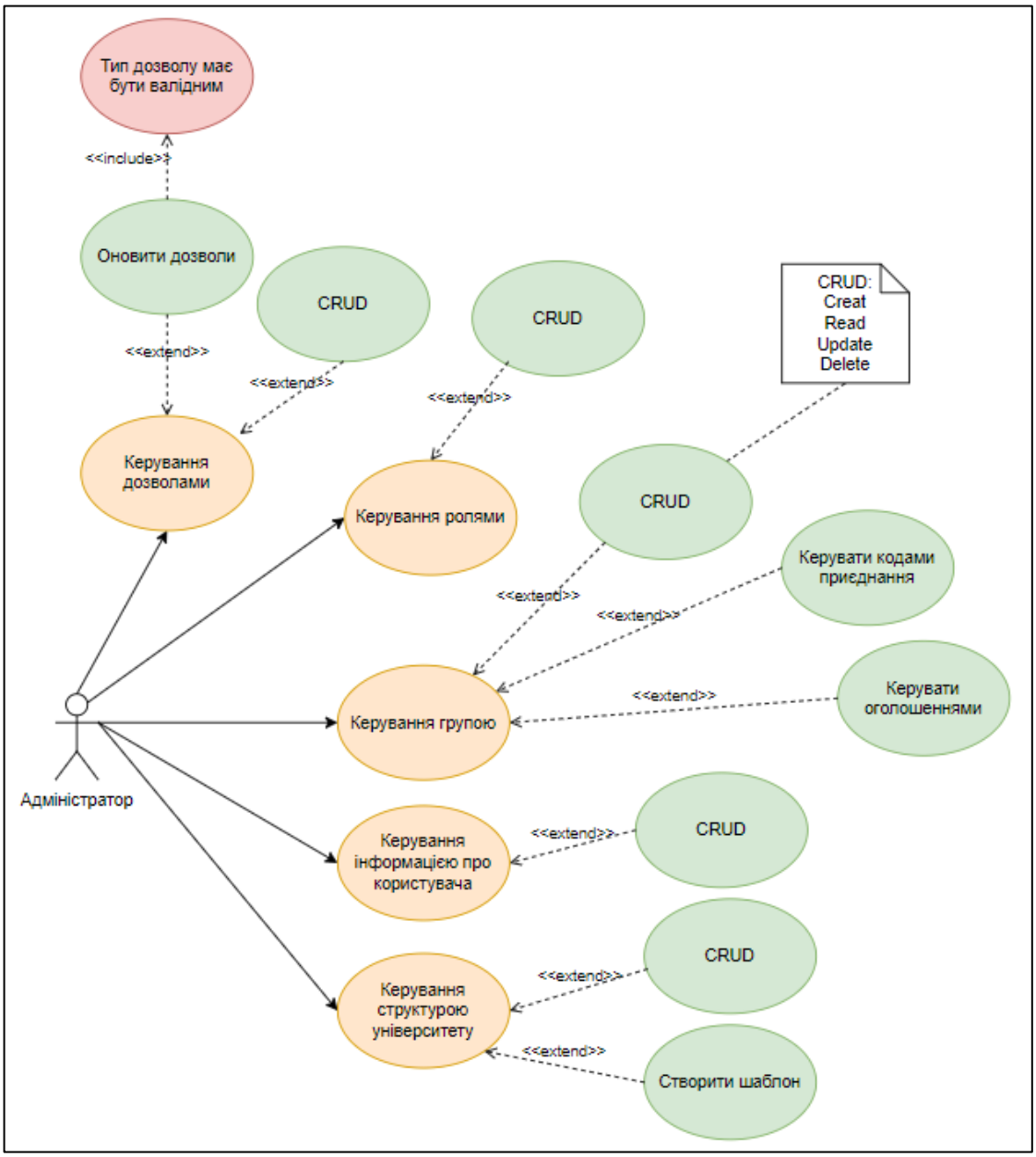


Рисунок 2.2 – діаграма використання (можливості адміністратора)

На даному рисунку зображено головні функціональні можливості адміністратора.

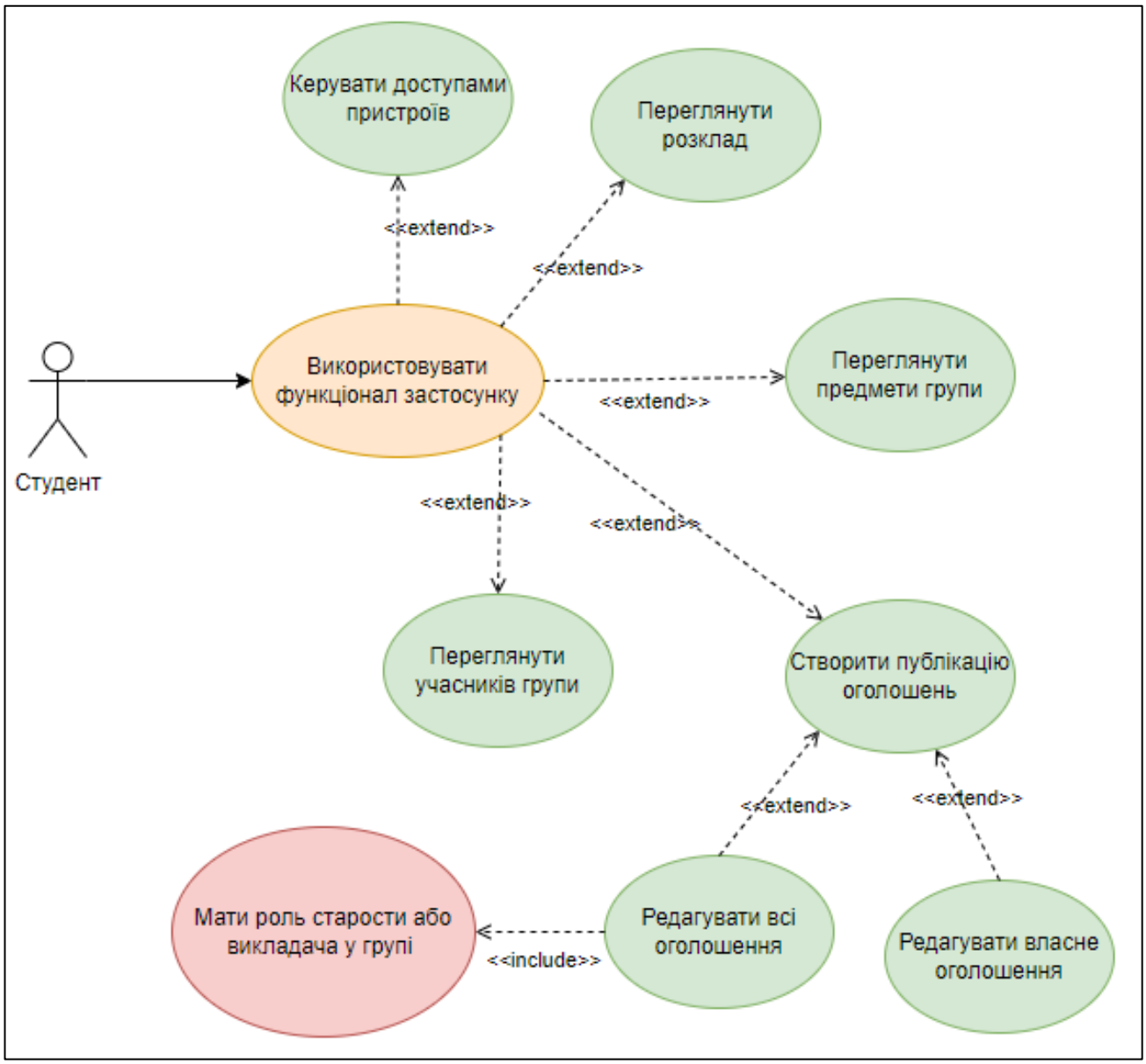


Рисунок 2.3 – діаграма використання (можливості студента)

2.1.5. Функціонал групи

Користувачу системи повинно бути доступно перегляд інформації про групу, а саме: назва та номер; кількість учасників та їх список в алфавітному порядку; структуру приналежності (яка кафедра та інститут); інформація про терміни навчання.

Додаткові функції залежать від ролі користувача в системі. Адміністратор та викладач (конкретної групи) може створювати та редагувати, а при необхідності таку можливість можна надати старості групи, завдяки гнучкості системи у роздачі та зміні ролей та їх функціональних можливостей.

2.1.6. Структура університету

Користувач системи повинен мати доступ до перегляду структури, з якої складається учбовий заклад. Для прикладу: назва університету, навчальні інститути або факультети, кафедра або спеціальність, група.

На рівні адміністратора та керівника кожного рівня структури університету повинна бути можливість створювати та редагувати сутності.

2.1.7. Предмети студентів

Користувачу системи повинен бути доступ до предметів своєї групи. В системі має реалізовуватися прив’язка предмету до конкретної групи та її студентів, щоб створити гнучкість даної можливості на рівні всього університету. Оскільки існує проблема, що багато груп або спеціальностей використовують одну сутність об’єкту (предмет) і одні і ті самі шаблони, завдання і простір здачі робіт.

2.1.8. Сповіщення системи

Кожному користувачу системи повинен бути доступ до керуваннями сповіщенням власного акаунту у мобільному застосунку. Будь-який користувач зможе зайти в налаштування особистого профілю, перейти у пункту “Сповіщення” і зробити власні налаштування. Наприклад, додати сповіщення про отримання оцінки за перевірене завдання на пошту або прибрати сповіщення про вхід з нового пристрою. Або взагалі вимкнути усі сповіщення в системі.

2.1.9. Онлайн розклад занять

Система повинна забезпечити стабільний та повноцінний доступ до перегляду та пошуку необхідного заняття, викладача. Оскільки дана функція є важливою, вона повинна включати в себе: розклад різних суб’єктів навчання (), електронний журнал та робочий план.

2.1.10. Сесії користувача

Система повинна забезпечити можливість налаштовувати та контролювати процес сесій користувача особисто. Оскільки дана система може дозволити користування на декількох на пристроях або вхід з одного пристрою на різні облікові засоби, повинно бути створено механізм керування сесіями.

Даний процес передбачає можливість входити в обліковий запис необмежену кількість раз та пристроїв. Також в налаштуваннях акаунту є розділ, де можна переглянути детальну інформацію (час та дата, геолокація та айпі-адреса) нової чи поточної сесії. Можна обрати всі активні сесії або будь-яку і завершити її.

2.2. Нефункціональні вимоги URLS

Нефункціональні вимоги, це певні обмеження та спектр вимог, які можуть обмежити систему або користувача у певних діях. Даний вид вимог відноситься до системи, як інструмент вирішення різного роду проблем: зручність використання, безпека, продуктивність, інтернаціональність, сумісність, тестування та ергономічність. Нефункціональні вимоги важливі, тому що вони можуть суттєво допомогти у забезпечені відповідності системи для потреб користувача.

Простими словами – це те, як система повинна щось робити.

2.2.1. Зручність використання

Система має бути зручна у використання для будь-якого користувача. Оскільки юзер буде неодноразово користуватися системою, можливо кожен день,

а більше вірогідно декілька разів на день – вона повинна бути зручна, мати інтуїтивно зрозумілий інтерфейс, зрозуміло-послідовний функціональний набір.

Застосунок не повинен викликати негативні емоції та бажання перестати ним користуватися. Він повинен бути привабливим та приємним у постійному користуванні, відповідати сучасним нормам проектування інтерфейсу користувача та дизайну.

Мобільний додаток повинен мати:

1. Адаптацію по розширенню екрану під будь який розмір.
2. Інтерфейс мобільного додатку повинен містити різні типи сповіщень (відповідно їх важливості).
3. Інтерфейс мобільного додатку повинен містити панель швидкого доступу до панелі керування.
4. Інтерфейс мобільного додатку постійно має бути активним (не заморожуватися при відсутності інформації або мати нескінчене завантаження);

2.2.2. Безпека

Система повинна забезпечувати надійну безпеку на будь-якому рівні від несанкціонованого доступу в систему:

1. Всі передачі даних в системі мають бути захищенні сучасним сертифікатом безпеки та мати надійні алгоритми шифрування.
2. Кожен акаунт захищено логіном і паролем, який знає тільки користувач.
3. Сама система проводить декілька етапів шифрування, після чого дані зберігаються в абсолютно незрозумілому форматі (хеш), який неможливо розшифрувати.
4. При потребі, можна додати двофакторну автентифікацію користувача.
5. При втраті даних доступу до акаунту, їх можна відновити за допомогою електронної пошти з підтвердженим номером телефону.

6. Користувачі не повинні мати доступ до чужих даних (окрім специфічних випадках, від імені адміністратора);

2.2.3. Продуктивність

Продуктивність URLS mobile залежить від багатьох факторів. Перше, це тип системи яка розробляється – крос-платформений застосунок. Даний тип незначною мірою зменшує швидкодію роботи застосунку в цілому, але дозволяє запускатися на різних операційних системах. Для досягнення більшої швидкості обміну даним повинна бути створена локальна база даних. До неї будуть зберігатися дані, які були на момент останнього використання повністю завантажені. Отже, це зменшить кількість даних які потрібно постійно оновлювати та завантажувати. Система повинна відповідати наступним вимогам по продуктивності (див. табл. 3).

Таблиця 3 – вимоги до продуктивності

Категорія	**Кількісний показник**
Час запуску застосунку	запуск < 3с.
Використання оперативної пам’яті застосунку	Від 400 МБ до 2 ГБ
Частота запису інформації на диск	Після кожної операції в системі йде оновлення інформації в локальній базі даних. Або кожні 10 хв.
Відсоток споживання батареї	Залежить від операцій які виконуються. Споживання помірне. До 25%
Використання мобільних даних	Залежить від запиту до серверу. Не менше 0.7 МБ.

Оскільки система з часом буде розширюватися, певні модулі будуть змінюватися і нові додаватися – вимоги до продуктивності будуть мінятися. Яскравим прикладом може стати параметр “Частота запису інформації на диск”,

якщо користувачем є адміністратор або викладач, які завантажують багато різних фалів і окремих даних. Які в свою чергу напряму впливають на частоту, з якою буде записуватися і перезаписуватися інформація на диск пам’яті, безпосередньо на пристрої.

2.2.4. Інтернаціональність

Система повинна відповідати міжнародному стандарту широкого використання додатку:

1. Українську.
2. Англійську.
3. Польську.
4. Німецьку;

В налаштуваннях користувач може обрати одну з двох мов на вибір. Переклад усієї текстової інформації повинен відбуватися згідно правилам перекладу. Переклад не повинен містити помилок, некоректного перекладу (по сенсу) або мати зовсім неперекладені слова або речення.

2.2.5. Сумісність

Мобільний додаток повинен відповідати вимогам сумісності для різних операційних систем та правил, по яким ПЗ може випускатися. Обов’язково має підтримуватися на усіх платформах, де розробляється додаток, згідно концепції технології крос-платформеної розробки.

У таблиці 4 буде наведено ОС та їх версії, які має підтримувати дана система.

Таблиця 4 – версії ОС, які підтримує URLS mobile

Операційна система	**Версія**
Android	5.1+
IOS	10.3+
Windows	10.2020.713+

Linux	Ubunta, Debian, Mint
MacOS	11+

Набір ОС та їх версії може та буде змінюватися, оскільки операційні системи постійно розвиваються та оновлюються, змінюються правила та набір функціоналу, який можна використовувати.

2.2.6. Тестування

Мобільний додаток повинен мати мінімальний відсоток покриття програмних модулів системи, не менше 10%. Тестування повинно відбуватися за допомогою unit test і performance test. Для отримання ефективної системи і вирішення помилок, які можуть виникнути – система повинна бути забезпечена 100% покриттям ручним чи автоматизованим тестуванням.

Також, перед веденням системи в експлуатацію необхідно перевірити на відповідність функції до наведених вимог і функцій системи, тобто провести функціональне і нефункціональне тестування.

2.2.7. Ергономічність

Інтерфейс мобільного додатку повинен бути ефективним та зручним у використанні. Для цього необхідно мати:

1. Оптимізацію часу – час, який користувач витратить, щоб знайти необхідну функцію або інформацію.
2. Емоціональний зв'язок – використовуючи мобільний застосунок, користувач повинен отримувати позитивні емоції (інтерфейс, привітливість, кольорова гама та розташування користувацького інтерфейсу)
3. Інтуїтивний рівень – використовуючи мобільний застосунок, користувачу повинно бути зрозуміло більшість наявного функціоналу, нічого зайвого і створення ієрархії функцій за важливістю.

Підсумовуючи, інтерфейс мобільного додатку має бути чітким і зрозумілим. Без ефективної та продуманої реалізації усіх вимог до інтерфейсу – успіх додатку не гарантується, навіть при наявності гарного функціоналу та власних особливостей.

2.2.8. Вимоги у вигляді асоціативної мапи

Для кращого розуміння додатку, з використанням радіантного мислення (особлива здатність людського мозку до асоціативного мислення), було створено асоціативну мапу вимог (див. рис. 2.4 та 2.5).

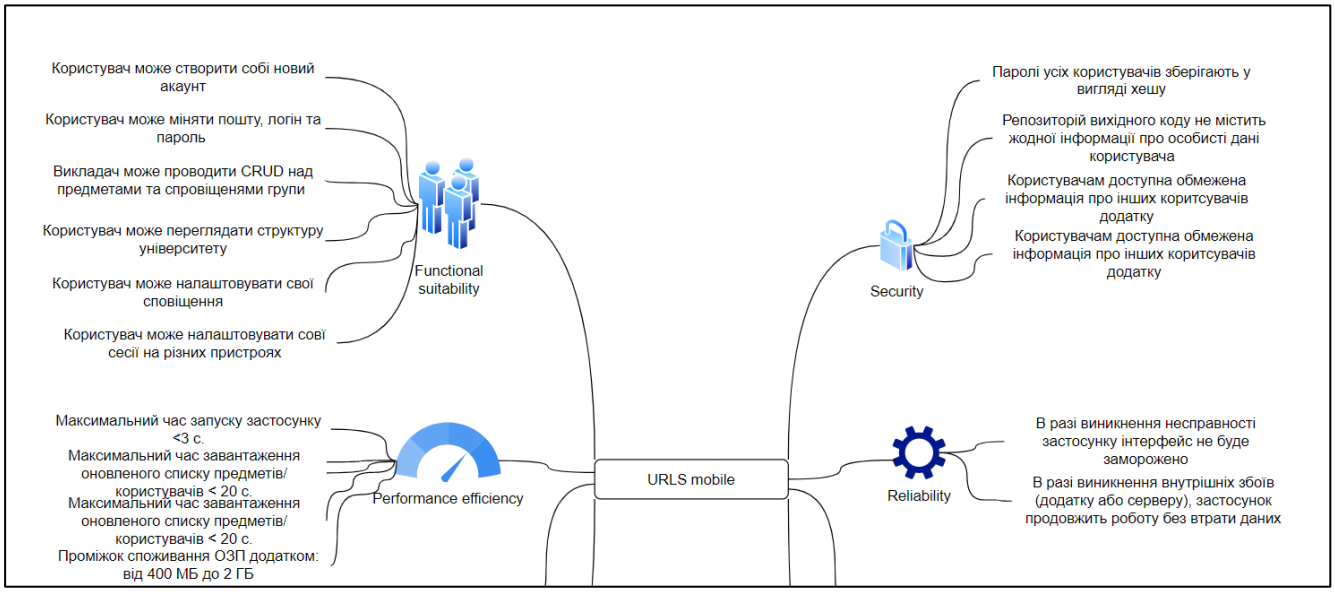


Рисунок 2.4 – Асоціативна мапа вимог до URLS mobile (частина 1)

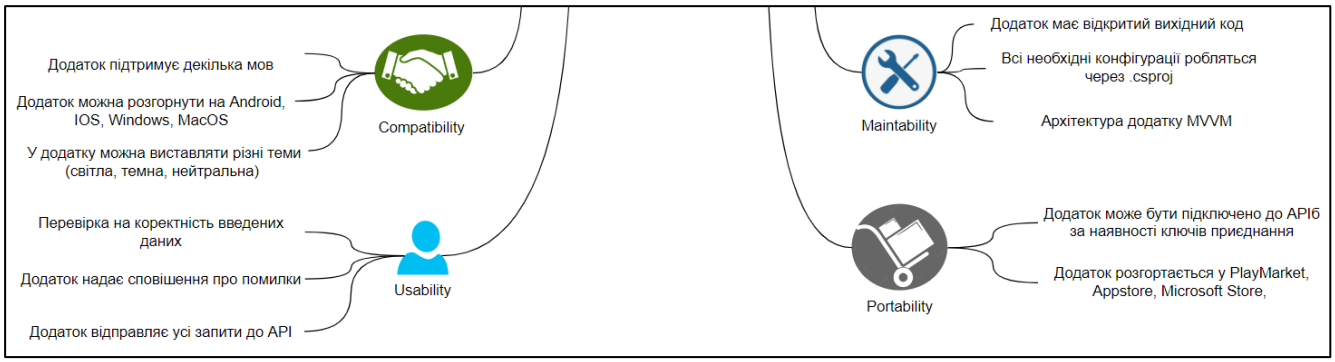


Рисунок 2.5 – Асоціативна мапа вимог до URLS mobile (частина 2)

На Mind Map до вимог додатку що розробляється, зображено головний об’єкт, від якого йдуть гілки до згрупованих по назві асоціацій, які описують якість системи, які варто розуміти:

1. Functional suitability (функціональна придатність) – параметр, який описує набір функцій доступних користувачу.
2. Performance efficiency (ефективність роботи) – параметр, який описує продуктивність системи.
3. Compatibility (сумісність) – параметр, який описує як і де даний застосунок може обмінюватися інформацію з іншими модулями або компонентами або поєднання різного роду за походженням функціоналу.
4. Usability (зручність використання) – параметр, який може розглядатися з двох напрямків: як користувач, так і стороннього сервісу.
5. Security (безпека) – параметр, який характеризує наявний функціонал захисту системи від небажаного впливу або сторонніх дій.
6. Maintainability (ремонтопридатність) – параметр, який описує ступінь ефективності в рамках процесу модифікації, оптимізації застосунку.
7. Portability (портативність) – параметр, який описує гнучкість та платформи розгортання системи;

Це модель, за допомогою якої можна сформувати комплексну оцінку якості продукції, що розробляється. Даний метод переважає над лінійним записом інформації (послідовний текстовий опис).

3. ПРОЕКТУВАННЯ ТА РЕАЛІЗАЦІЯ МОБІЛЬНОГО ДОДАТКУ

Етап проектування програмного забезпечення або будь-якого модулю, додатку, є кроком до впровадження ідеї. Даний етап включає в себе розробку плану дій та розробку технічної документації. Важливий крок – вибір та обґрунтування засобів та інструментів для розробки.

3.1. Інструменти для розробки

Вибір інструментів для розробки, є важливим кроком, як вибір мови програмування і побудови архітектури, після аналізу предметної області, опису вимог та оцінку якості системи.

3.1.1. Git

Для зберігання файлів, оновлення та керування проектом необхідне сховище. Для максимально ефективної розробки та підтримки проекту, необхідно мати таку систему. Нею на сьогоднішній день є Git.

Наявність віддаленого доступу до репозиторія, в якому ведеться розробка на даний момент, надає широкі можливості для розробника і команди (якщо вона є), з використанням додаткових інструментів.

Git – це розподілена система, призначена для керування версіями файлів та роботи в команді. Дана система дає наступі можливості:

1. Зберігання файлів в віддаленому репозиторії.
2. Керування версіями файлів.
3. Створення гілок.
4. Об’єднання гілок.
5. Встановлення правил для завантаження файлів.
6. Робота в команді над одним проектом (файлом).
7. Створення різних;

Якщо казати про систему, яка доступна користувачу з інтерфейсом взаємодії, слід дивитися в сторону GitHub (desktop версія див. рис. 3.1), GitBash для Windows та GitKraken для Linux (див. рис. 3.2) та Git-клієнт для IDE (вбудований див. рис. 3.3).

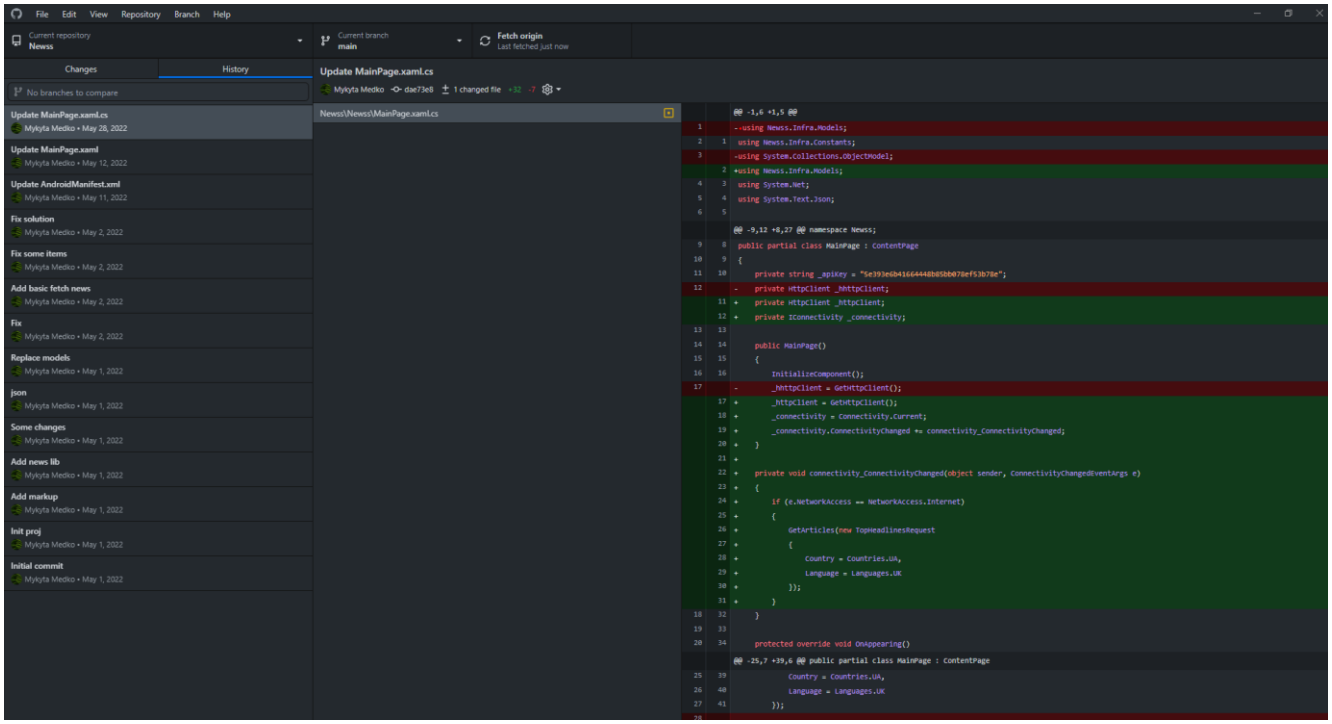


Рисунок 3.1 – Десктопна версія GitHub

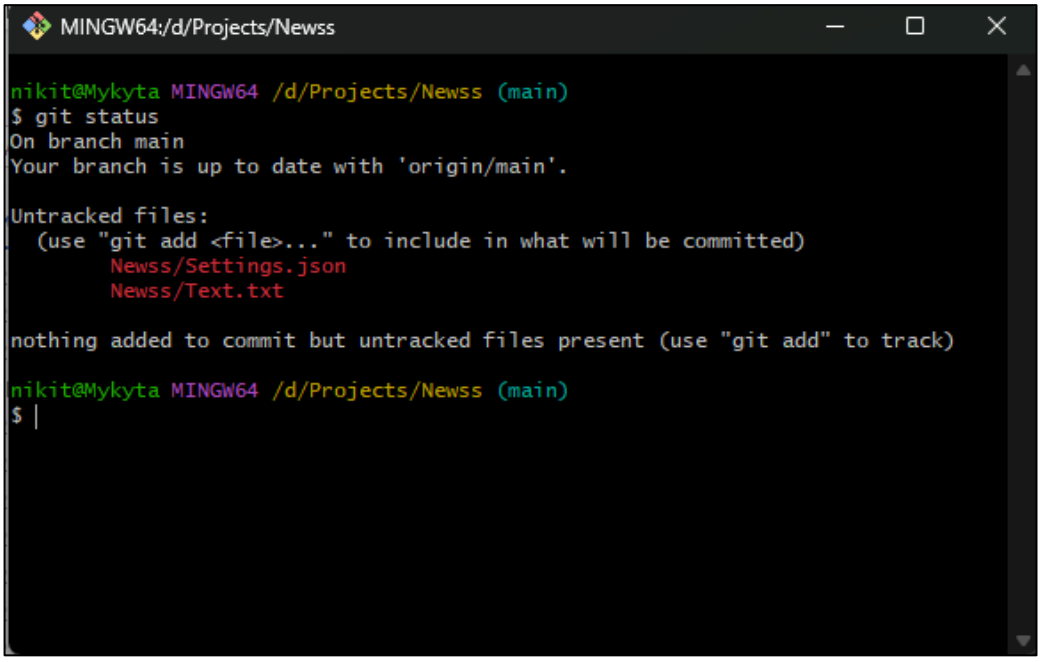


Рисунок 3.2 – GitBash для Windows

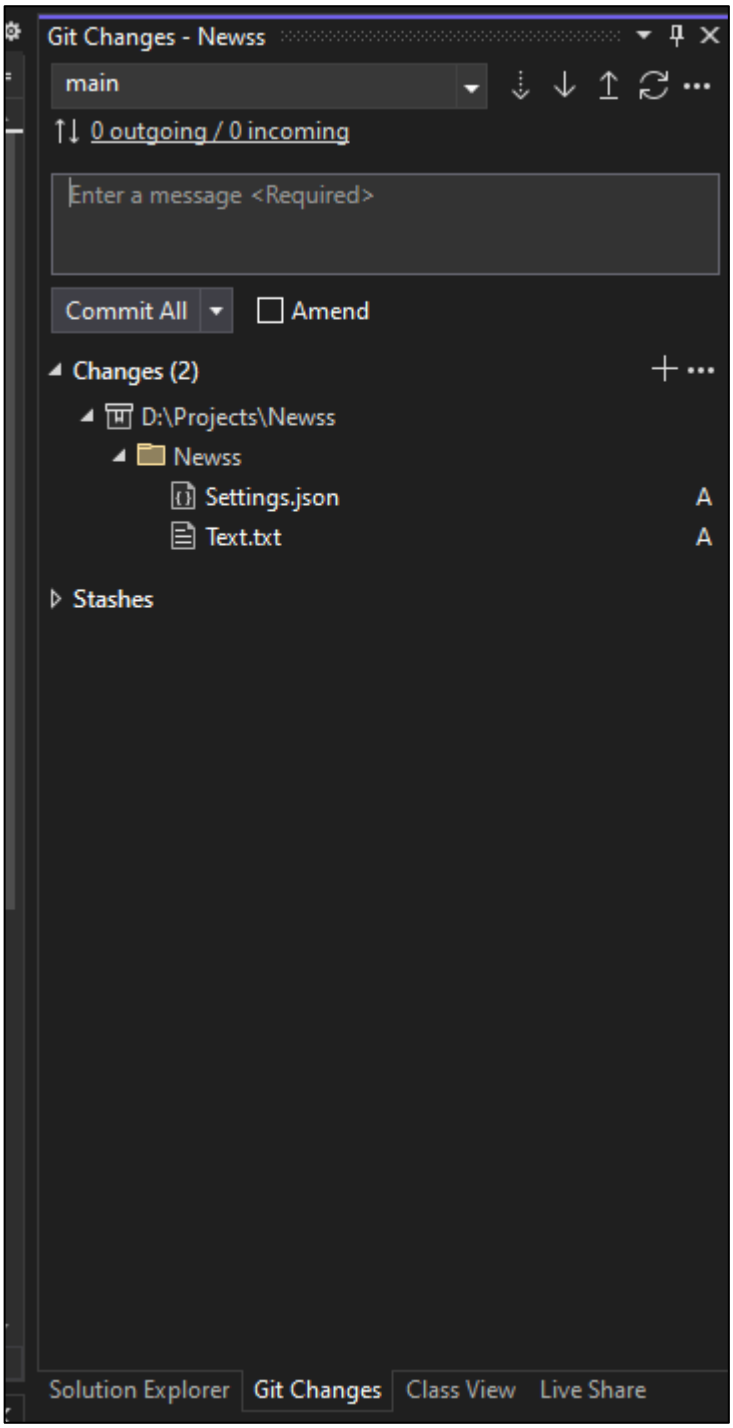


Рисунок 3.3 – Git-клієнт для IDE (Visual Studio)

Використання системи Git підвищує ефективність розробки ПЗ та відкриває нові можливості у командній роботі, особливо при взаємодії в декількох командах або підрозділах. З розвитком системи GitHub, з’явився інструмент GitHub Copilot, який допомагає користувачам Visual Studio, під час розробки використовувати

можливості підказок та автодоповнення від штучного інтелекту. Інструмент розробила компанія OpenAI.

3.1.2. Visual Studio

IDE, що буде англійською Integrated Development Environment (інтегрована середа розробки), яка забезпечує набір можливостей та інструментів для розробки програмного забезпечення або різних модулів.

В даній бакалаврській роботі було використано Visual Studio 2022 (див. рис. 3.4) Professional (TD244-P4NB7-YS4XK-Y8MMM-YWC2G). Даний програмний продукт розробила компанія Microsoft у 1997 році.

Даний продукт є потужною середою розробки для застосунків різних типів, а саме: веб-сайти, консольні застосунки, мобільні застосунки, бази даних, графічні програми тощо. Вибір даної середи розробки полягає у багатьох чинниках та перевага, які вона має, для розробки мобільного застосунку.

Декілька переваг Visual Studio:

1. Вбудована підтримка системи Git.
2. Висока продуктивність.
3. Пряма інтеграція з сервісами Azure та AWS.
4. Різноманітний (не класичний) функціонал:
    a. Вбудований Intellisense.
    b. Підтримка інструментів для роботи з UML.
    c. Інтеграція з Copilot.
    d. Пряма взаємодія з віртуальною машиною для різних платформ.
5. Оновлюваний інтерфейс продукту.
6. Потужний редактор код (компілятор).
7. Автоматичне створення тестів для покриття коду.
8. Потужні та різноманітні інструменти для профілювання (створення своїх або вибір з існуючих активних вікон профілювання), під час розробки;

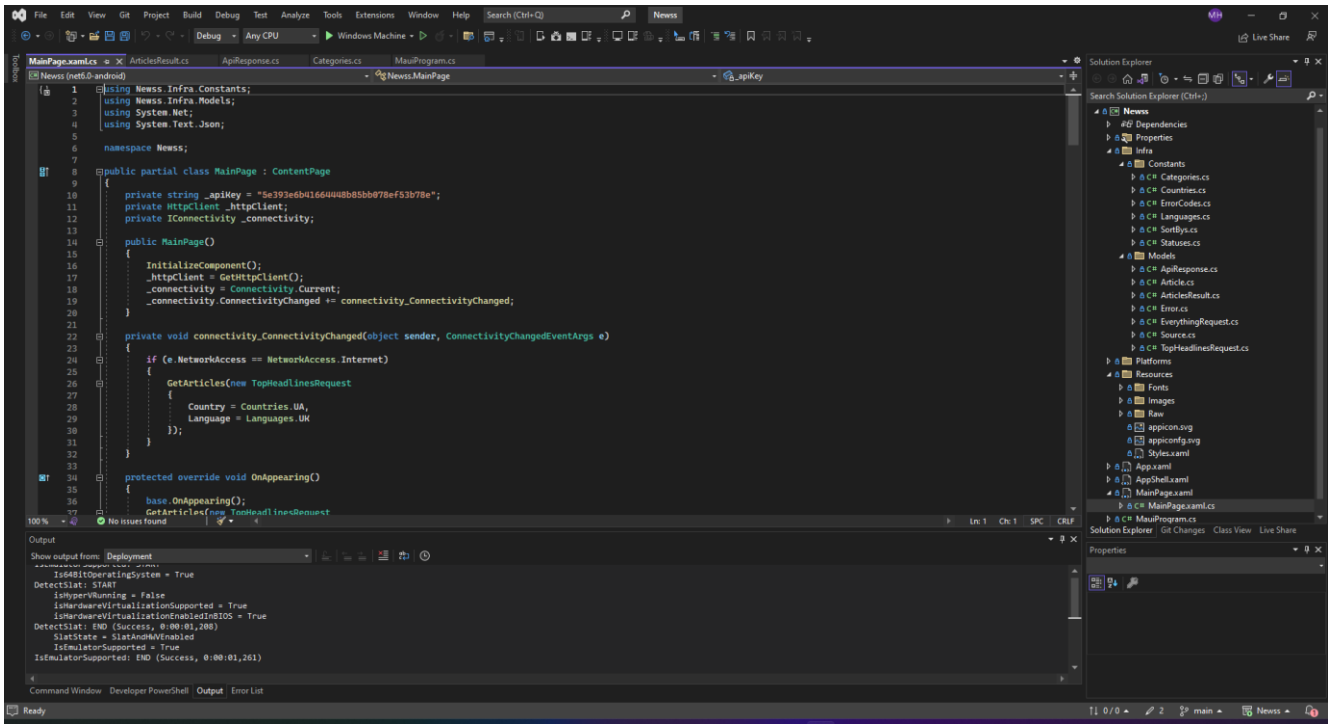


Рисунок 3.4 – Середа розробки Visual Studio 2022

Дана середа розробки поєднує в собі велику кількість потужних інструментів та засобів, які дають можливість ефективно розробляти програмне забезпечення.

3.1.3. SQLite Browser

Під час розробки, експлуатації додатку або програмного забезпечення, необхідно мати інструмент для керування, обробкою та зберігання даних. В даному випадку, під час розробки мобільного додатку використовувалася СУБД SQLite. A DB Browser SQLite – це графічний інструмент (див. рис. 3.5) для роботи з SQLite, який дає користувачу інтерфейс взаємодії, для перегляду, створення та редагування даних.

Головні особливості, які сприяли вибору SQLite, при розробці Android додатку:

1. Вбудована підтримка SQLite в Android.
2. Локальне зберігання даних на пристрої Android.
3. Міграції БД.

4. Резервне копіювання та відновлення файлів.
5. Підтримка БД на рівні ОС.
6. Легка СУБД, з якою просто працювати на Android;

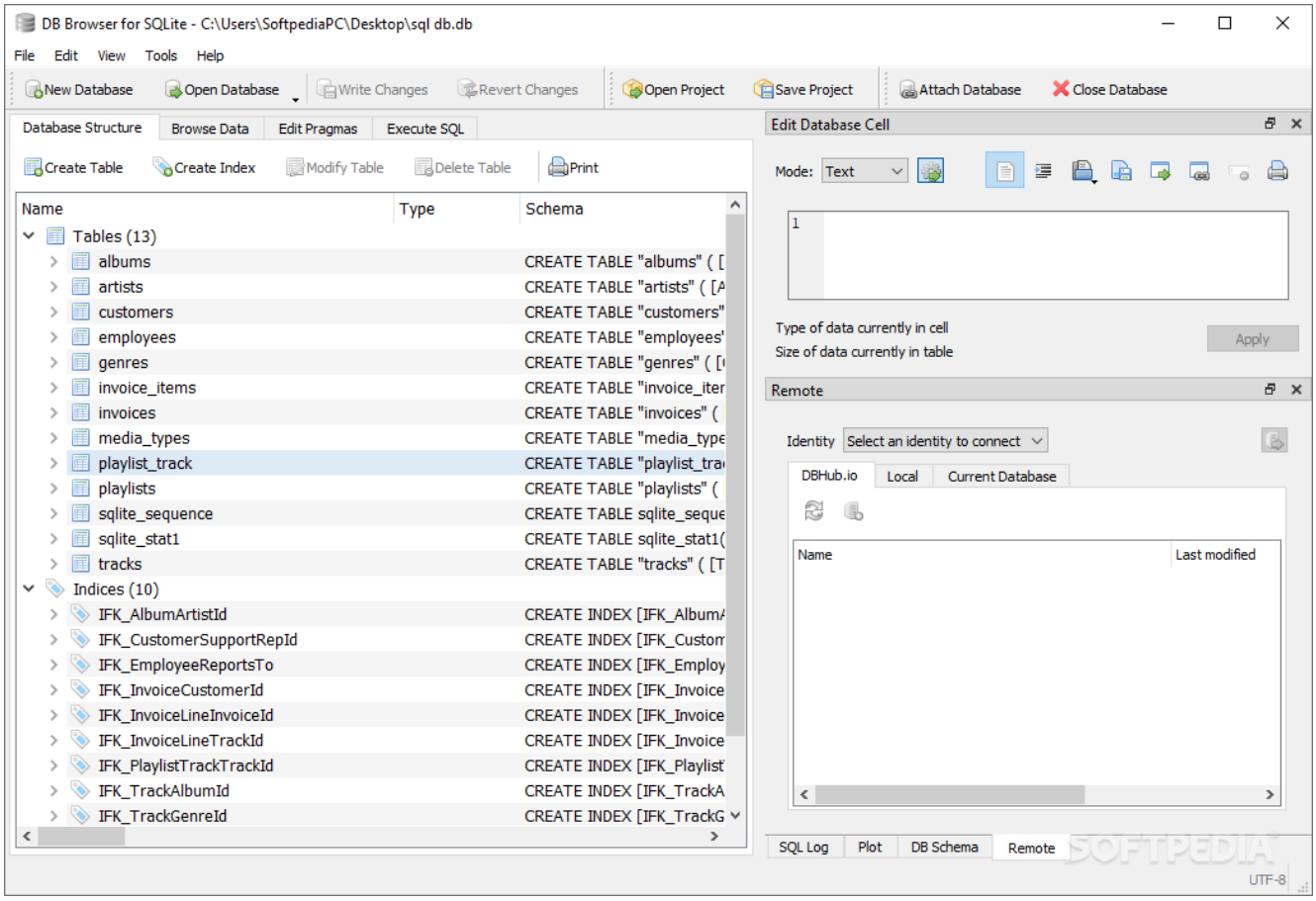


Рисунок 3.5 – Головне меню DB Browser SQLite

Слід зауважити, що DB Browser SQLite – це інструмент для візуалізації роботи СУБД, яким може користуватися як розробник, так і кінцевий користувач. Отже він повинен залишатися простим у використанні, щоб досягати необхідні цілі в розробці і підтримці додатку.

3.1.4. Android Emulator

Великий шанс, що під час розробки додатку (будь-якого) буде замало класичного набору інструментів у середі розробки. Отже необхідно завантажувати, як платні, так і безкоштовні модулі. В ході розробки мобільного додатку, постало

питання, як і де тестувати прототип. При редагуванні або додаванні нової функції, кожен раз збирати та компілювати реліз версію застосунку і очікувати не мало часу, поки середа розробки і платформа зберуть проект – не ефективно.

Тому обрано емулятор, який є додатковим пакетом до Visual Studio 2022 – Android Emulator, з версією 13 андроїду і 33 API відповідно.

В даному випадку, емулятор за допомогою використання багатоплатформеної платформи .NET MAUI, дозволяє запускати та розробляти в реальному часі мобільний застосунок, з операційною системою Android.

Це значно полегшує весь процес розробки, оскільки написання коду і наступне тестування – об’єднується в один інструмент.

Особливості емулятору Visual Studio for Android:

1. Швидкість. Даний емулятор побудований та працює на базі x86-остого процесора, що дає максимальне наближення до фізичного пристроюю користувача.
2. Вбудована підтримка (Hyper-V), що суттєво пришвидшує час завантаження емулятора і скорочує об’єм використання необхідних для цього ресурсів.
3. Підтримка різних пристроїв та версій ОС.
4. Емулятор у Visual Studio має велику кількість сенсорів і периферійних пристроїв (GPS, компас, геолокація), що дає можливість розробляти та тестувати додаток, наближений до фізичного пристрою.
5. Завантаження власних шаблонів і налаштувань;

3.2. Архітектура системи

При розробці мобільного додатку, варто пам’ятати, що часто в функціонування системи бере участь дві сторони – сервер, який користувач не бачить і клієнт, котрий він і безпосередньо використовує. В даній роботі, клієнтська частина є “худою” і це означає, що майже всю логіку та обробку даних бере на себе сервер, а клієнт надсилає запити з даними. Для розуміння загальної картини взаємодії усіх компонентів, які приймають участь в функціонуванні додатку, розроблено діаграму архітектури (див. рис. 3.6).

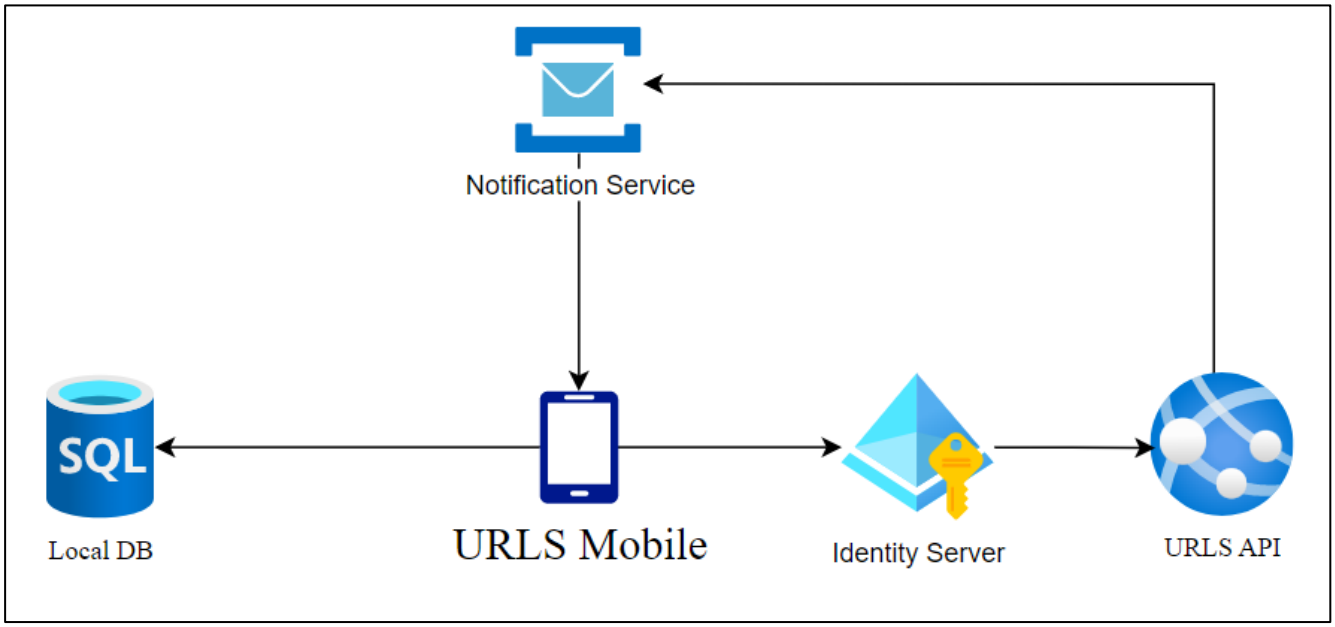


Рисунок 3.6 – Діаграма архітектури

Весь процес роботи починається з мобільного додатку URLS (клієнту), коли користувач виконує певні дії, які потребують зв'язок з сервером (чи то запит на оновлення даних, чи на сервері відбулися зміни і клієнт має отримати нові дані чи відповідь від сервера), спочатку запит проходить через Identity Server, де відбувається процес автентифікації користувача.

Сервер перевіяє чи дійсний користувач в системі, якщо так – відбувається авторизація, а якщо ні, то буде відповідь з помилкою для користувача. Далі Identity Server видає мобільному додатку ключ доступу (який має термін дії), який клієнт

буде використовувати при кожному наступному запиті до серверу в рамках сесії або дії токену (той самий ключ доступу з терміном дії).

Для відправки на клієнт сповіщень будь-якого типу – сервер відправляє дані до Notification Service, де працює механізм Firebase з подальшим використанням (відправкою відповіді до клієнта) push notifications.

Для збереження необхідних даних, призначених для коректного функціонування додатку або окремих файлів користувача (завантаження лекцій, практик тощо), розроблено локальну базу даних SQLite.

3.2.1. Компоненти додатку

Архітектура додатку побудована за моделлю MVVM (Model-View-ViewModel), що дозволяє легко розробити графічний інтерфейс користувача. В даному підході (див. рис. 3.7) реалізується відокремлення бізнес логіки (в рамках мобільного додатку) та графічного інтерфейсу (див. табл. 5).

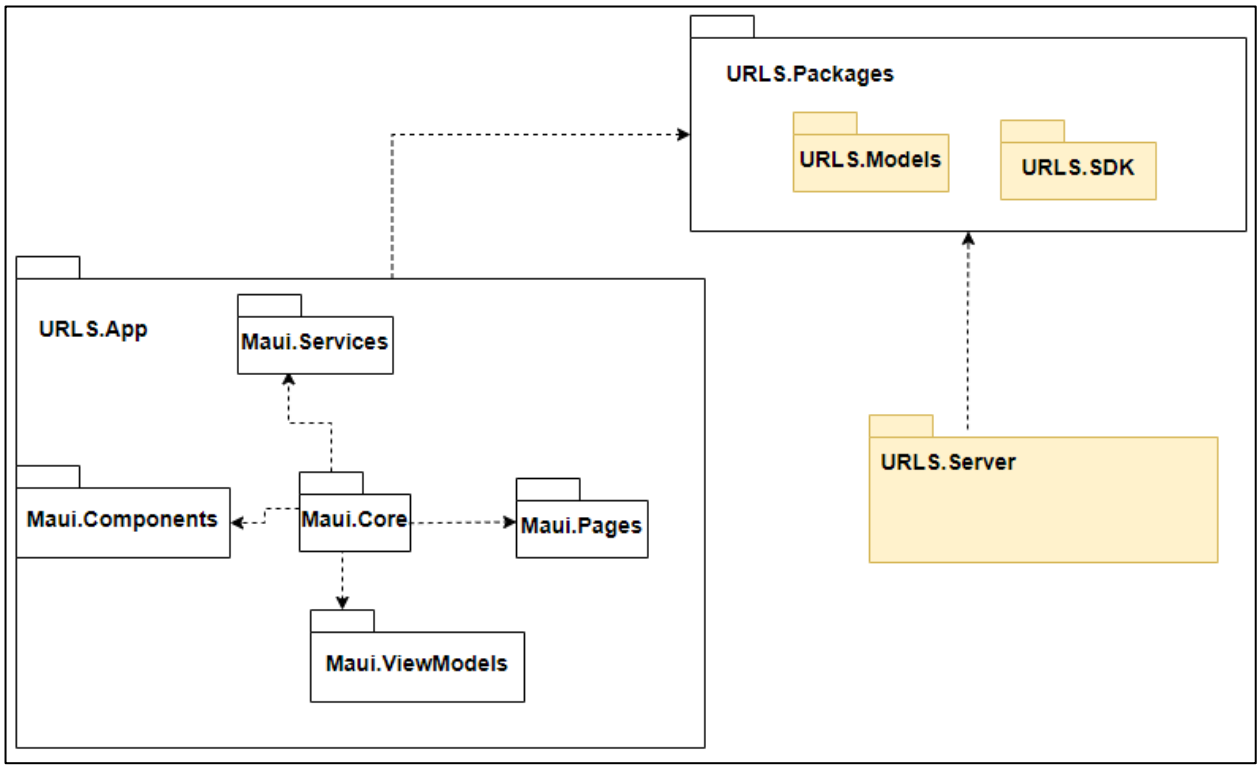


Рисунок 3.7 – Діаграма пакетів

Таблиця 5 – Перелік і призначення пакетів

Пакет	Призначення
URLS.App	Батьківський пакет, що містить весь функціонал мобільного застосунку
Maui.Core	Ядро системи (.NET MAUI)
Maui.ViewModels	Пакет, що містить моделі представлення
Maui.Components	Футери і хідери (окремі ксатомні едементи)
Maui.Services	Пакет, що містить визначення та реалізацію сервісів
Maui.Pages	Пакет, який містить сторінки додатку
URLS.Library	Батьківска зовнішня бібліотека
URLS.Models	Пакет, який містить моделі, що використовує мобільний додаток
URLS.SDK	Пакет, який містить сервіси і запити, необхідні до комунікації з API

При реалізації додатку з використанням шаблону проектування MVVM, використовувалася додаткова бібліотека URLS.Packages, яка містить в собі необхідні моделі та підготовлені запити для взаємодії клієнту з сервером.

3.2.2. Платформа .NET MAUI та мова програмування C#

.NET MAUI (.NET Multi-Platform App UI,) – це крос-платформений фреймворк, який призначений для розробки мобільних і комп’ютерних застосунків з використанням мови програмування C# та мови розмітки XAML.

За допомогою .NET MAUI можна розробляти застосунки для таких платформ:

1. Windows.
2. macOS.
3. Android.
4. IOS.
5. Samsung Tizen;

Платформа надає об’єднує API-інтерфейси для Android, iOS, Windows та macOS, який надає розробнику можливість виконувати одноразовий запис в будь-якому місці, тим самим забезпечуючи додатковий доступ до кожного платформи окремо. Всі підтримуючі бібліотеки мають доступ до однієї бібліотеки класів .NET(BCL). Дана бібліотека абстрагує інформацію про базову платформу від коду.

BCL залежить від середовища виконання (.NET), щоб надати середу розробки для виконання коду. Платформи Android, iOS та macOS середа реалізується за допомогою mono, при реалізації середи виконання .NET.

Надаючи єдину платформу для створення користувацького інтерфейсу, .NET MAUI дозволяє розробляти як мобільні додатки, так і класичні застосунки. На рисунку 3.8 наведено схемі загальне представлення архітектури застосунку.

Платформа надає крос-платформений API для різних функцій пристроїв:

1. Доступ до датчиків (акселерометр, гіроскоп та компас).
2. Можливість перевіряти мережеве підключення пристрою.
3. Вибирати файли на пристрої.
4. Використовувати вбудовані механізми обробки голосу в мову, для читання з пристрою.
5. Копіювання і вставка тексту в системний буфер обміну;

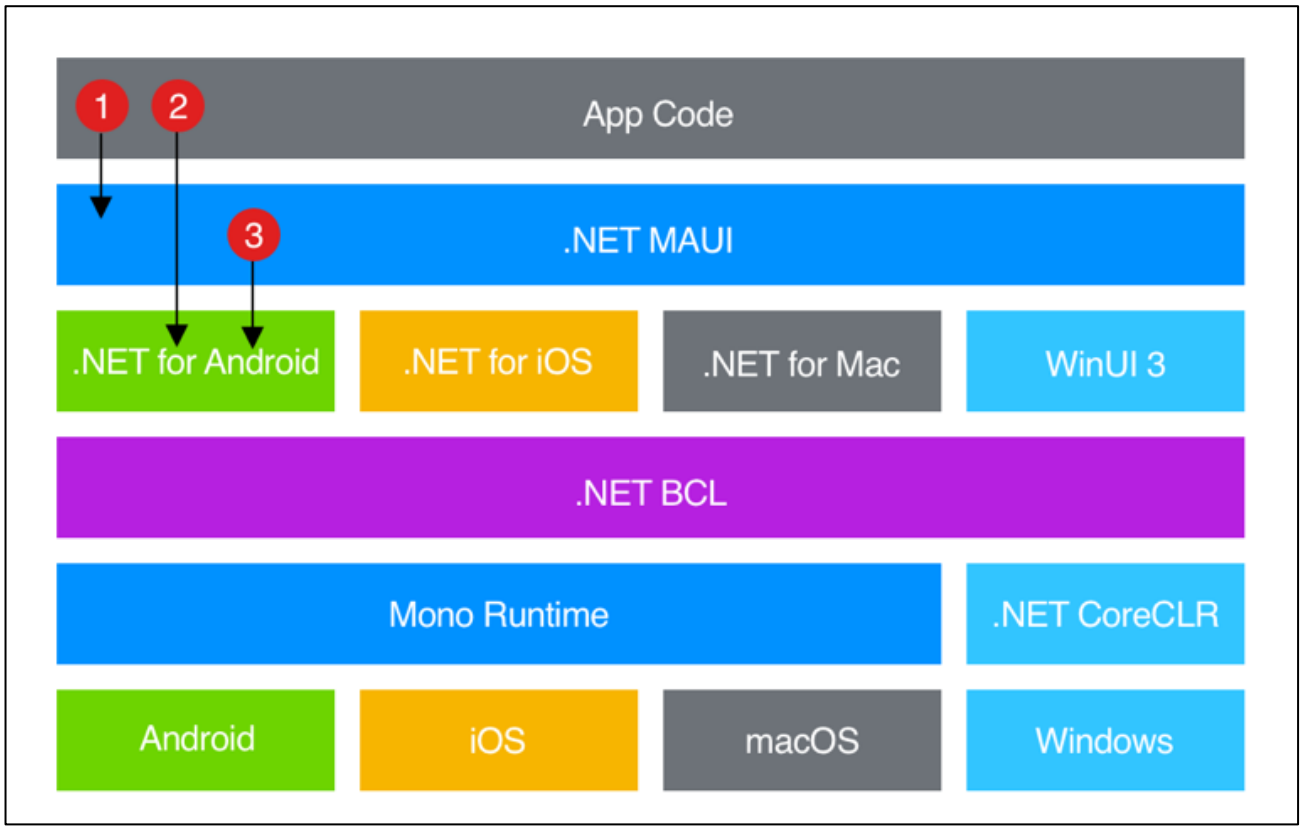


Рисунок 3.8 – Схема загального представлення архітектури застосунку .NET MAUI

Переваги розробки за допомогою платформи MAUI:

1. .NET MAUI надає прямий доступ до API кожної підтримуваної платформи та їх апаратним можливостям.
2. Підтримує прив’язку даних.
3. Під час розробки створюється єдиний проект, який використовує спільну бібліотеку коду.
4. З коробки є велика колекція вбудованих елементів керування.
5. Підтримка hot-reload (суттєво допомагає під час розробки).
6. Інтеграція з екосистемою .NET.
7. Висока продуктивність платформи.
8. Підтримка шаблону MVVM.

Недоліки розробки за допомогою платформи MAUI:

1. Достатньо молода платформа.
2. Для розробки на даній платформі потрібен досвід.

3. Використання тільки C# (як і перевага, так і недолік);

3.2.3. Identity Server

Взаємодія сервера і клієнта повинна бути чітко та правильно налаштовано, оскільки передача даних та отримання відповіді відіграють ключову роль у процесі функціонування система. Весь шлях процесу передачі даних у більшості випадків відбувається за стандартними процесами, які описує та підтримує Identity Server (див. рис. 3.9).

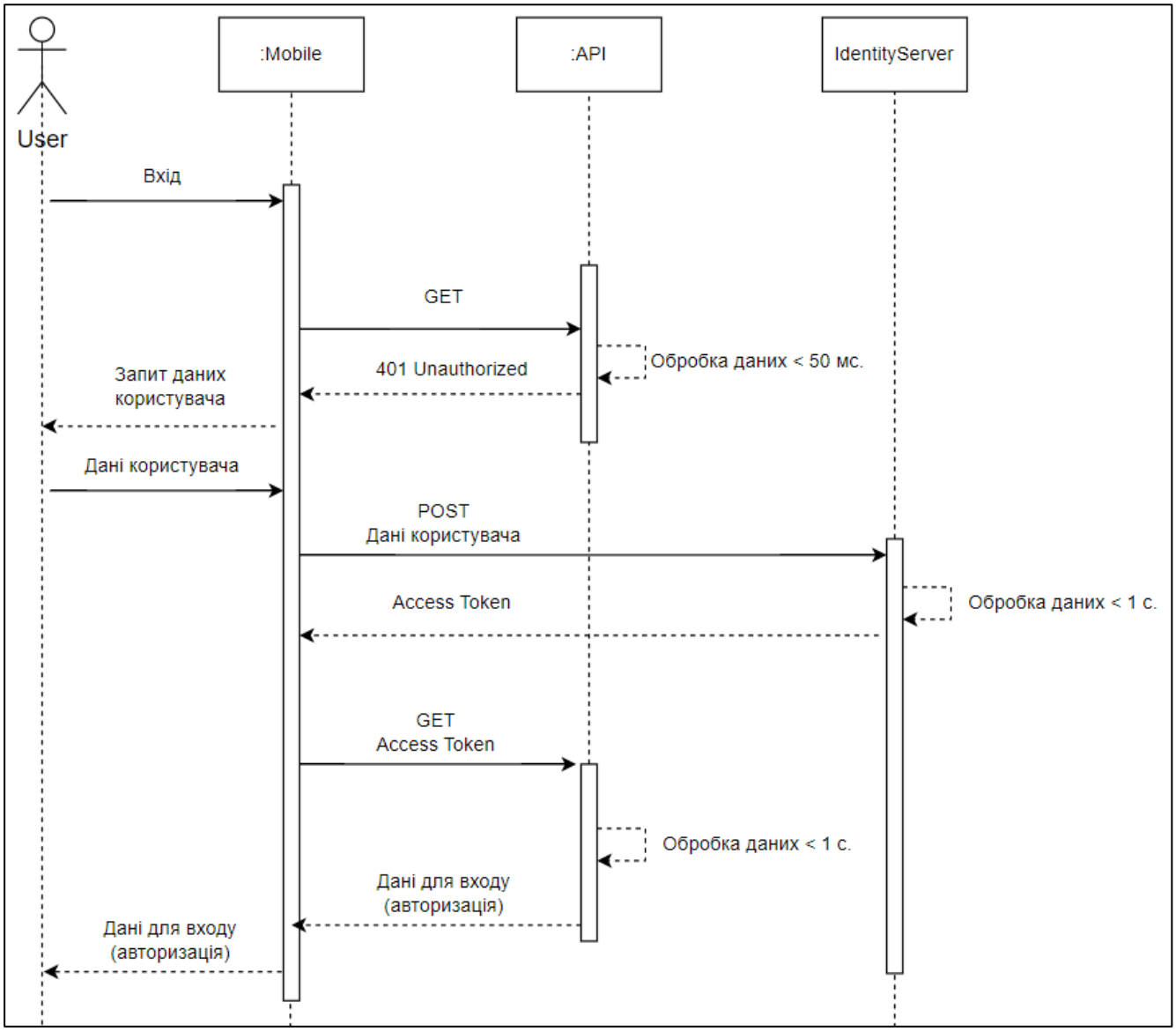


Рисунок 3.9 – Діаграма послідовності (Identity Server)

Identity Server працює на основі протоколу OpenID Connect і OAuth 2.0, що дозволяє забезпечити авторизацію та автентифікацію користувача.

Весь процес починається з реєстрації клієнта (в даному випадку телефону), що включає в себе створення захищеного каналу зв'язку, яким є HTTPS, з подальшим отриманням унікального ідентифікатора та секретного ключа. Наступним кроком йде аутентифікація клієнта (перевірка на приналежність клієнта до серверу), де клієнт надсилає дані для перевірки (наприклад логін і пароль), після чого сервіс одразу робить перевірку для аутентифікації, та якщо дані проходять (вірні) – генерується токен доступу (він же Access Token).

Даний токен містить інформацію про ідентифікатор користувача клієнта, секретний ключ (він же Secret Key). І при наступних запитах клієнта до серверу, він буде надсилати дані ключі для авторизації і отримання необхідних даних.

Варто зазначити, що ключі мають термін дії та можуть генеруватися системою у встановлених правилах Identity Server.

Переваги:

1. Єдина точка входу. Дозволяє користувачу мати одну централізовану точку доступу до різних ресурсів, додатків та сервісів.
2. Безпека. Identity Server забезпечує високий рівень безпеки для ідентифікації та авторизації.
3. Централізоване управління. Дозволяє налаштувати та централізовано керувати процесами авторизації та аутентифікації.

Недоліки:

4. Складність розгортання. Потребує детального налаштування та розгортання, може стати проблемою для команди розробників, без попереднього досвіду.
5. Складність синхронізації. При використанні великої кількості ролей та прав доступу, додатків та ресурсів, тоді може виникнути складність у налаштуванні та розгортанні.
6. Великий проміжок часу на розробку і підтримку.
7. Велика залежність від зовнішніх сервісів;

3.2.4. Push Notifications та FCM

Push-сповіщення є важливим атрибутом кожного додатку, для інформування користувача, яке надходить у верхній частині пристрої (шторкою) і не зобов'язує користувача кожен раз заходити у застосунок. Це спливаючі повідомлення, які суттєво відрізняються від звичайних SMS і приходять вони у будь-який час (якщо користувач під'єднаний до мережі інтернету).

Сповіщення є двох типів:

1. Мобільні.
2. Браузерні.

Основна ідея надсилання push-сповіщень є у тому, щоб надати користувачу більше важливої інформації, спросити шлях використання застосунку і привернути увагу. Їх можна налаштовувати та організовувати принцип роботи при розробці додатку або створенні веб-сервісу, а сам користувач може керувати ними на своєму пристрою або безпосередньо в застосунку.

Структуру push-notification можна представити у наступних елементах:

1. Назва: невеликий заголовок, який привертає увагу користувача.
2. Іконка (образ): картинка застосунку, що надсилає сповіщення. За умови що вона взагалі є (залежить від реалізації розробником) або користувач може власноруч налаштувати (так не часто доступне).
3. Повідомлення: зазвичай є заголовком повідомлення у самому застосунку. Повинне бути стислим та містити головну інформацію, для привертання уваги користувача.
4. CTA: активна кнопка заклику або "заклик до дії", який допоможе перетворити повідомлення. Цю функція покликана для дії одразу, що позбавить користувача відкривати застосунок.
5. URL: повна протилежність до CTA. Якщо кнопка заклику недоступна, розробники додають посилання, при натисканні на яке користувач одразу буде переведений на головний екран застосунку або релевантну сторніку.

Базуються вони на ідеї хмарного сервісу обміну повідомленнями. Першим був GCM (Google Cloud Messaging), призначений для передачі сповіщень або інформацію в сторонній застосунок користувача.

Наступним кроком розвитку став FCM (Firebase Cloud Messaging), який повністю замінив GCM, ставши крос-платформеним рішенням для надсилань сповіщень і нотифікацій для Android, iOS, Web-app.Дані сервіси розроблені компанією Google та їх дочірніми компаніями.

В даній роботі використано FCM, з першочерговою комунікаціє з API та базою даних, і останньою ланкою є URLS Mobile, який виступає клієнтом (див. рис. 3.10).

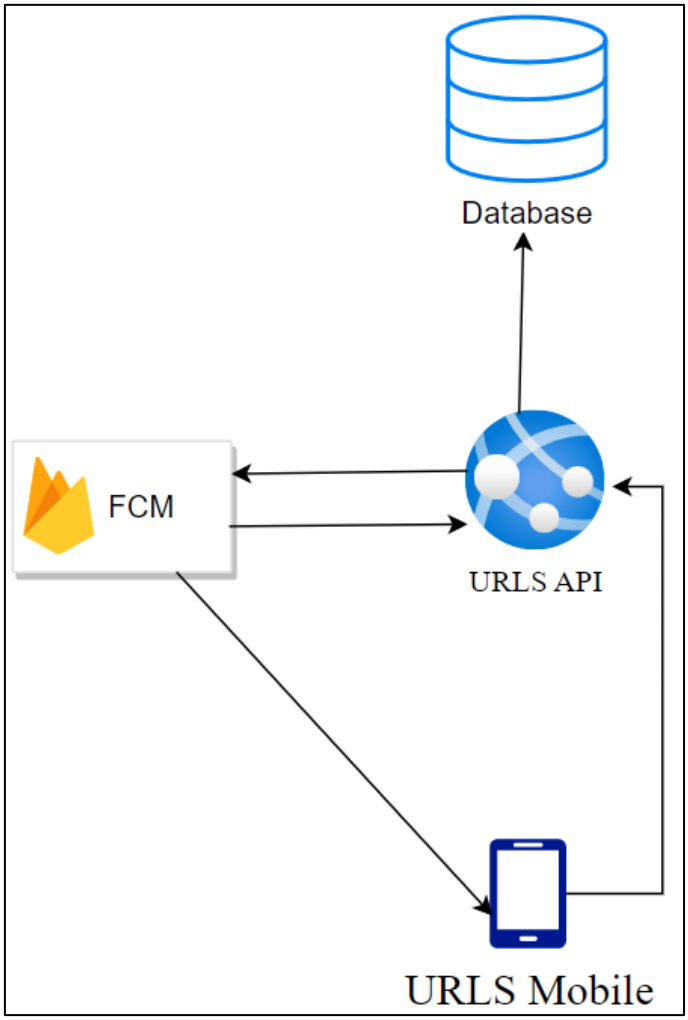


Рисунок 3.10 – Схема взаємодії FCM з URLS Mobile

3.3. Використання шаблонів проектування

Шаблон проектування (або Патерн) – це шаблонний (типовий) спосіб вирішення проблеми, яка може часто зустрічатися під час проектування архітектури систем. Патерн не є чимось вже готовим, як бібліотека, функція або метод з готовим рішенням. Він представляє собою загальний принцип та поради для вирішення проблеми, які виникають.

Опис будь-якого патерну можна розділити на такі елементи:

1. Проблема яку має вирішувати патерн.
2. Підхід для вирішення проблеми способом, який надає патерн.
3. Структура головних рішень.
4. Приклад рішення (залежить від мови програмування).
5. Особливості при реалізації.
6. Поєднання та конфлікти з іншими патернами;

Переваги, які надає правильне використання шаблонів складається з декількох пунктів:

7. Стандартизація коду: виконується менше розрахунків при проектуванні, використовуючи готові шаблонні рішення, оскільки вже сформовано вирішення деяких проблем.
8. Стандартизація назв рішень: уніфіковані методи та рішення, які мають загальноприйняту назву, легше одразу зрозуміти, особливо при роботі в команді.
9. Готові рішення (шаблони): значно скорочується витрата часу, при використанні готових рішень, для уникнення проблем з використанням непідходящих рішень або заплутаних шляхів подолання проблеми.

Таким чином, використовуючи шаблони проектування, можна підвищити ефективність та стандартизувати код і лексику, яку використовують розробники, з вирішенням загальновідомих проблем проектування.

3.3.1. MVVM як патерн проектування

В даній роботі, при проектуванні додатку було використано патерн MVVM (Model-View-ViewModel), який складається з трьох модулів (див. рис. 3.11).

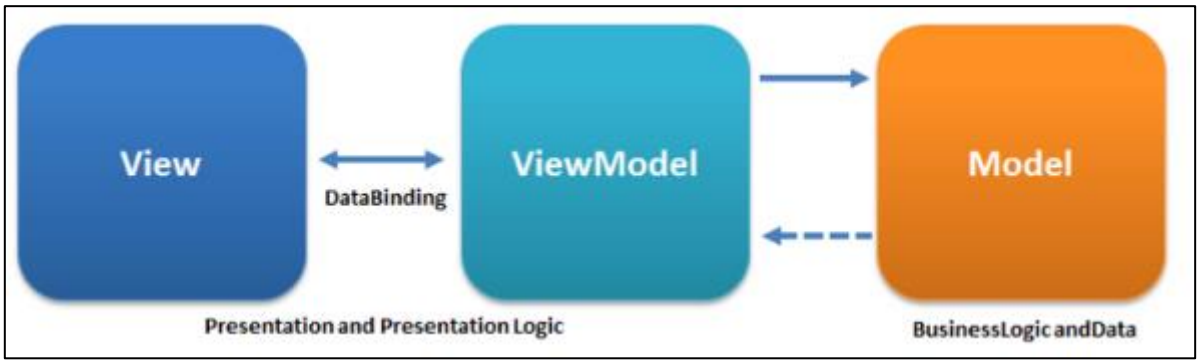


Рисунок 3.11 – Трьох компонента архітектура MVVM

Даний патерн дозволяє відокремити бізнес-логіку від візуальної частини, чим дозволяє розробляти незалежні модулі і розробляти додатки для Windows, Android та iOS.

Кожен з трьох модулів має свою область відповідальності та внесок у функціонування систем:

1. Model: описує дані, що використовуються. Даний рівень може зберігати в собі логіку, яка пов’язана з конкретною моделлю, але в той же час, вона не повинна містити в собі жодну логіку, оскільки вона вся повинна бути у ViewModel.

2. View: забезпечує інтерфейс в застосунку, за допомогою якого користувач може взаємодіяти з додатком. Даний передає вхідні дані до ViewModel, а той в свою чергу до Model. Він не має містити ніякої логіки.

3. ViewModel (модель представлення даних): містить в собі головну логіку роботи застосунку і оброки даних, керує рівнем Model. Дана модель звязує між собою рівні Model та View. Він містить в собі звязки та інформацію про View та Model.

3.3.2. Repository як патерн проектування

Даний шаблон проектування є одним із найпопулярніших патернів. При розробці та проектуванні додатку, він дозволяє забезпечити зручний доступ до даних розміщених в межах мікросервісу або рівню логіки. В собі він фактично містить реалізацію компонентів, як Unit of Work або DBcontext.

Шаблон репозиторію (див. рис. 3.12) можна описати наступним чином – створення узгодженого інтерфейсу для роботи з даними, в незалежності від джерела, що дозволяє розділити рівень Business layer та Repository.

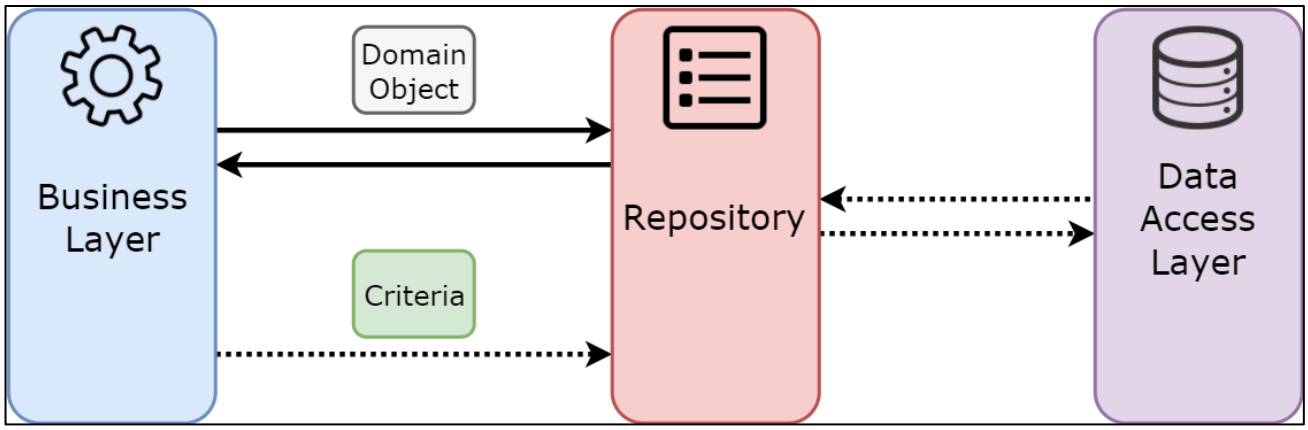


Рисунок 3.12 – Шаблон Repository

Переваги використання патерну Repository:

1. Поділ обовязків: даний патерн дозволяє відокремити бізнес-логіку від проблем роботи з сховищем даних. Даний процес спростить шлях внесення змін в систему.
2. Підвищення ефективності тестування: оскільки існує розподіл, завдяки абстракції яку надає репозиторій, є можливість створювати різноманітні види тестів, не використовуючи реальні дані.
3. Гнучкість у виборі джерел даних: патерн Repository дає можливість змінювати джерела даних, не змінюючи код бізнес-логіки, що підвищує гнучкість та ефективність системи.

3.3.3. IoC та Dependency injection як принцип ООП

Дані концепції ключові в розробці програмного забезпечення. За допомогою них можна покращити модульність, розширюваність та можливість ефективнішого тестування коду шляхом забезпечення залежностей.

Інверсія управління (Inversion of Control, IoC) є невід'ємною частиною ООП, яка передбачає, шо об'єкт не повинен сам створювати або керувати власними залежностями. Тобто, необхідно використовувати інтерфейси (абстракції) замість реалізацій, а щоб викликати цю реалізацію, треба використовувати контейнер, який бере на себе всю відповідальність за створення та ініціалізацію об'єктів.

Ін'єкція залежності (Dependency injection, DI) вже є конкретною реалізацією інверсії управління. Суть даної концепції, що вона передає залежності об'єкту зовні, замість самостійного створення об'єктом. Даний процес відбувається через конструктор, властивості чи методи класу.

IoC та DI є ефективними та признаними інструментами серед розробників програмного забезпечення, використовуються в ООП мовах програмування. Лише вірне та зрозуміле використання дасть бажаний ефект.

4. ТЕСТУВАННЯ СИСТЕМИ ТА ОТРИМАННЯ РЕЗУЛЬТАТІВ

Тестування системи відіграє важливу роль у процесі розробки програмного забезпечення, яке допомагає і вирішує різні проблеми, в розробці.

Тестуванню піддається наступні аспекти ПЗ:

1. Функціонал.
2. Продуктивність.
3. Сумісність.
4. Безпеку.
5. Базу даних.
6. Автоматизацію (процесів або інструментів).
7. Інтерфейс користувача;

Проведення тестування дозволяє знаходити помилки і дефекти, покращувати якість написаного коду і в результаті застосунку, знижує ризики виникнення несподіваних проблем та помилок у програмі. Надає підтвердження вимогам, яким має відповідати розроблена система. В деяких випадках суттєво економить час розробки. Існує підхід у розробці, де спочатку пишуться тести, а вже по їх результатам розробляється система, його назва – тест-керована розробка.

4.1. Види тестування

Існує велика кількість різних видів тестування, на кожну з якого спеціаліст навчається, практикується і використовує інструменти та засоби окремо.

По степені автоматизації тести бувають ручні, автоматизовані та напівавтоматичні.

Ручне тестування відбувається без використання додаткових інструментів. А от автоматичні та напівавтоматичні – використовують засоби та інструменти повністю або частково.

Тести бувають наступних видів:

1. Модульне тестування: працюють на низькому рівні, з кодом застосунку. Тестують окремі методи, функції і класи.
2. Інтеграційні тести: перевіряється наскільки корректно і ефективно співпрацюють різні модулі між собою.
3. Функціональне тестування: головна увага привертається до бізнес-вимог додатку. Перевіряє працездатність компонентів системи на різні функції.
4. Smoke-тестування: це певна групована збірка тестових кейсів, яка робить тестування по певним критеріям і перевіряє на стабільність роботу системи.
5. Ручне тестування: тип тестування, де розробник займає сторону користувача і перевіряє на працездатність, наявні властивості у програмного забезпечення.

4.2. Ручне тестування

Для перевірки роботи мобільного додатку було обрано ручне тестування, бо даний спосіб є достатньо простим, який не потребує використання чи написання спеціального програмного забезпечення.

При розробці додатку було проведено SMOKE тестування, яке дало можливість провести поверхневе тестування головного функціоналу мобільного додатку, для виявлення, чи може дана система бути використана користувачем.
Для проведення тестування було сформовано тест-кейси, які включають в себе:

1. Передумова.
2. Дані на вхід (вони ж початкові дані).
3. Очікуваний результат.
4. Сам результат. Він можливий у вигляді фотографії роботи застосунку або текстового опису;

Таблиця 6 – Тест кейс 1. Отримання кодів приєднання до групи

Тест кейс 1. Отримання кодів приєднання до групи	
Дія	Отримання кодів приєднання
Умова виконання	Користувач повинен бути приєднаний до групи
Очікуваний результат	Відкрита сторінка з усіма кодами приєднання конкретної групи

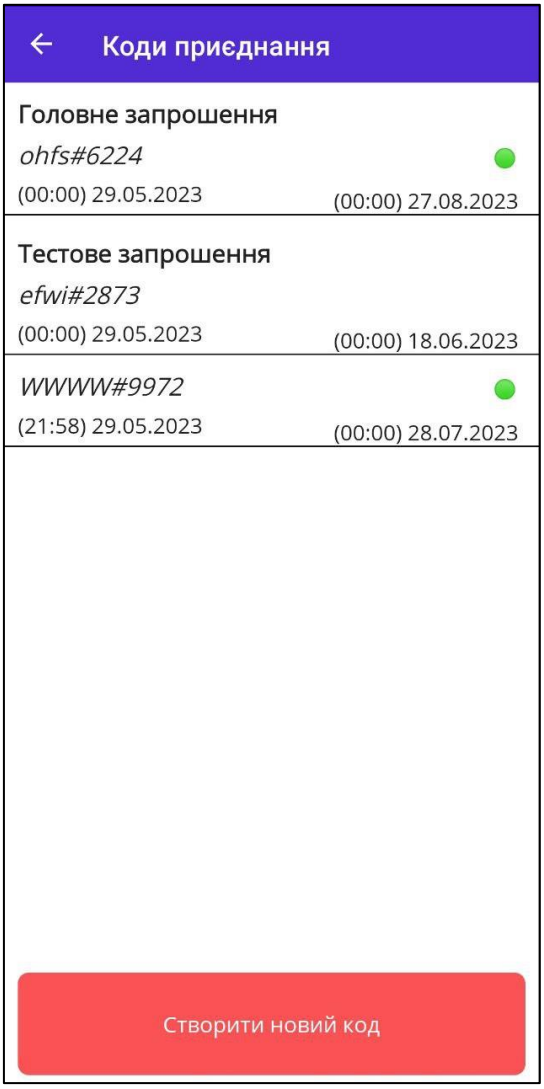


Рисунок 4.1 – Сторінка з усіма кодами приєднання

Таблиця 7 – Тест кейс 2. Пошук групи

Тест кейс 2. Пошук групи	
Дія	Знайти групу та переглянути інформацію про неї

Умова виконання	Користувач повинен бути авторизованим В системі повинно існувати більше однієї групи
Очікуваний результат	Група існує і вона знайдена

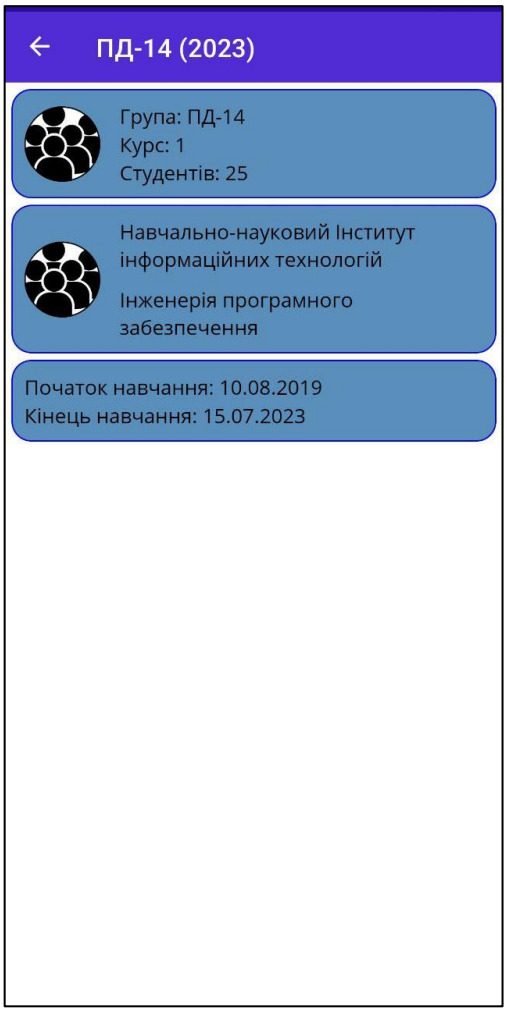


Рисунок 4.2 – Сторінка з усіма кодами приєднання

Таблиця 8 – Тест кейс 3. Отримання сповіщень

Тест кейс 3. Отримання сповіщень	
Дія	Авторизуватися в акаунт з нового пристрою
Умова виконання	Користувач повинен успішно авторизуватися
Очікуваний результат	На панелі сповіщень є два сповіщення, які дають інформацію про вхід з нового пристрою

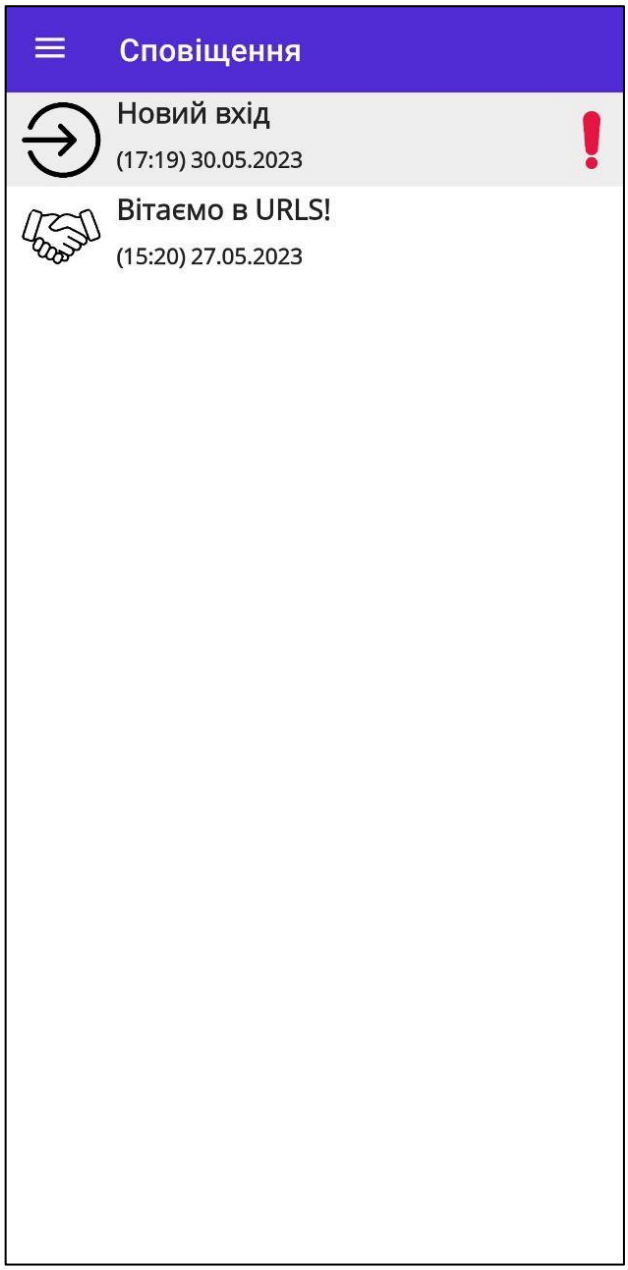


Рисунок 4.3 – Сторінка сповіщень (з інформацією про вхід з нового пристрою)

Таблиця 9 – Тест кейс 4. Перегляд пристроїв, де була проведена авторизація

Тест кейс 4. Перегляд пристроїв, де була проведена авторизація	
Дія	Відкрити сторінку “Пристрої” Переглянути “Неактивні” пристрої
Умова виконання	Користувач повинен бути авторизований Користувач повинен відкрити сторінку керування пристроями

Очікуваний результат	Сторінка відкрита На сторінці є два блока (усі активні і неактивні) пристрої Неактивний пристрій можна скрити

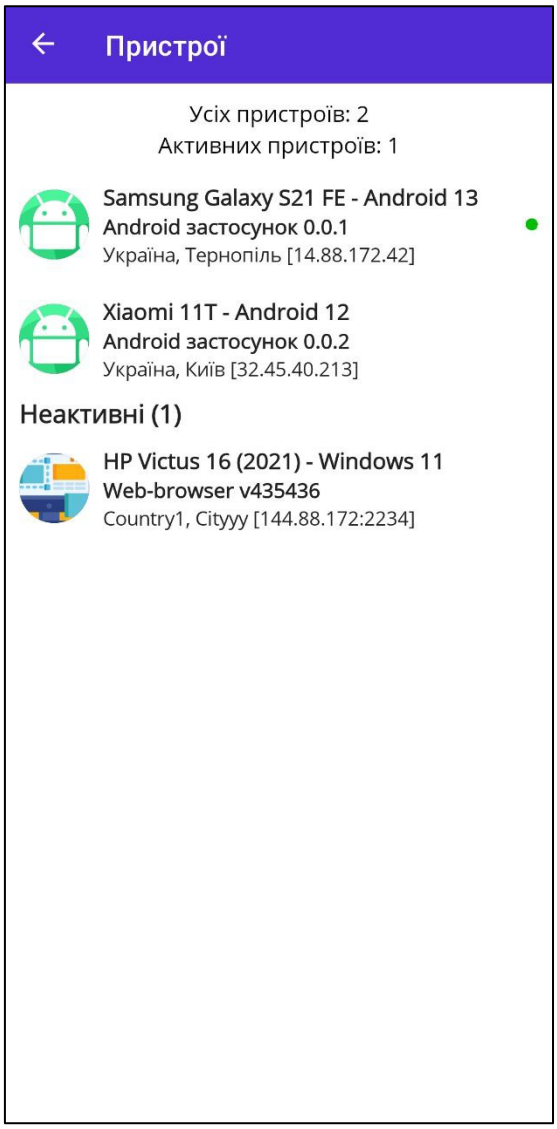


Рисунок 4.4 – Сторінка сесій користувача на різних пристроях

Таким чином, провівши ручне тестування деяких функцій, було сформовано тест кейси, отримано позитивні результати та наведено результати.

ВИСНОВКИ

В результаті виконання даної дипломної роботи, було розроблено мобільний додаток для дистанційного навчання, в рамках студента.

1. Актуальність розробленого додатку та його наукову новизну, було обґрунтовано шляхом аналізу предметної області, де виділено головних конкурентів, близьких за функціоналом. Досліджено спільні переваги та недоліки усіх аналогів і проаналізовано проблеми та труднощі, з якими стикається студент, під час дистанційного навчання, які не були вирішені у порівняних системах. В результаті сформовано функціональні та нефункціональні вимоги до додатку.

2. Було спроектовано архітектуру та інтерфейс мобільного додатку, шляхом перетворення вимог у діаграми та схеми UML. Також спроектовано локальну базу даних та схему компонентів, з яких складається та з чим взаємодіє додаток.

3. Архітектура мобільного додатку описана з різних використанням патернів та підходів реалізації. Інструменти, які використовувалися під час розробки, є: середа розробки Visual Studio 2022, мова програмування C# 11.0 з використанням платформи .NET MAUI, а база даних – SQLite. Сервісом повідомлень виступає Firebase Cloud Messaging в реальному часі. Взаємодія з сервером відбувається через Identity Server.

4. В ході розробки було використано сучасні та ефективна патерни проектування: MVVM, Repository, IoC та DI, які дозволили правильно та чітко побудувати структуру проекту.

5. Розроблено додаток для студента, з окремими функціями навчального процесу для студента, який підлягає сучасним вимогам та потребам користувача.

6. Розроблений додаток може запускатися на різних платформах, з використанням одного стандарту коду за рахунок мульти-платформи .NET MAUI, який було використано в роботі.

7. Досліджено аспект завантаження та взаємодії з файлами для дистанційного навчання, на пристрої студента, з використанням офлайн режиму додатку.

Результати дослідження бакалаврської роботи апробовані на всеукраїнській науково-технічній конференції: “Застосування програмного забезпечення і ІКТ” та третій всеукраїнській науково-практичній конференції “Сучасні інтелектуальні інформаційні технології в науці та освіті”.

ПЕРЕЛІК ПОСИЛАНЬ

1. Про дистанційне навчання [Електронний ресурс] // ДУТ. – 2019. – Режим доступу до ресурсу: https://www.dut.edu.ua/ua/1032-pro-distanciyne-navchannya-organizaciyno-metodichniy-centr-novitnih-tehnologiy-navchannya.

2. Коронавірус в Україні [Електронний ресурс] // МОЗ України. – 2019. – Режим доступу до ресурсу: https://covid19.gov.ua.

3. Синхронне й асинхронне дистанційне навчання [Електронний ресурс] // Osvita.ua. – 2021. – Режим доступу до ресурсу: https://osvita.ua/school/method/78950/.

4. Дистанційне навчання як дієвий інструмент управлінської освіти [Електронний ресурс] // Крок. – 2022. – Режим доступу до ресурсу: http://snku.krok.edu.ua/index.php/vcheni-zapiski-universitetu-krok/article/view/513/543.

5. Implementation of the distance learning by Moodle platforms in the process of future philologists training [Електронний ресурс] // MDPU Repository. – 2019. – Режим доступу до ресурсу: http://eprints.mdpu.org.ua/id/eprint/7813/1/%D1%81%D1%82%D0%B0%D1%82%D1%82%D1%8F%20EETECS2019_007%283%29.pdf.

6. What's new in C# 11 [Електронний ресурс] // Microsoft. – 2023. – Режим доступу до ресурсу: https://learn.microsoft.com/en-us/dotnet/csharp/whats-new/csharp-11.

7. What is GitHub? [Електронний ресурс] // Techtarget. – 2023. – Режим доступу до ресурсу: https://www.techtarget.com/searchitoperations/definition/GitHub.

8. Design the infrastructure persistence layer [Електронний ресурс] // Microsoft. – 2023. – Режим доступу до ресурсу: https://learn.microsoft.com/en-us/dotnet/architecture/microservices/microservice-ddd-cqrs-patterns/infrastructure-persistence-layer-design.

9. Model-View-ViewModel (MVVM) [Електронний ресурс] // Microsoft. – 2022. – Режим доступу до ресурсу: https://learn.microsoft.com/en-

us/dotnet/architecture/maui/mvvm.

10. .NET dependency injection [Електронний ресурс] // Microsoft. – 2023. – Режим доступу до ресурсу: https://learn.microsoft.com/en-us/dotnet/core/extensions/dependency-injection.

11. Visual Studio Emulator for Android [Електронний ресурс] // Microsoft. – 2023. – Режим доступу до ресурсу: https://visualstudio.microsoft.com/vs/msft-android-emulator/.

12. DB Browser for SQLite [Електронний ресурс] // Sqlitebrowser. – 2023. – Режим доступу до ресурсу: https://sqlitebrowser.org/.

13. Visual Studio 2022 IDE [Електронний ресурс] // Microsoft. – 2023. – Режим доступу до ресурсу: https://visualstudio.microsoft.com/vs/.

14. Firebase Cloud Messaging [Електронний ресурс] // Firebase. – 2023. – Режим доступу до ресурсу: https://firebase.google.com/docs/cloud-messaging.

15. IdentityServer for cloud-native applications [Електронний ресурс] // Microsoft. – 2023. – Режим доступу до ресурсу: https://learn.microsoft.com/en-us/dotnet/architecture/cloud-native/identity-server.

16. The different types of software testing [Електронний ресурс] // Atlassian. – 2023. – Режим доступу до ресурсу: https://www.atlassian.com/continuous-delivery/software-testing/types-of-software-testing.

17. Web authenticator [Електронний ресурс] // Microsoft. – 2023. – Режим доступу до ресурсу: https://learn.microsoft.com/en-us/dotnet/maui/platform-integration/communication/authentication?tabs=windows.

18. What is .NET MAUI? [Електронний ресурс] // Microsoft. – 2023. – Режим доступу до ресурсу: https://learn.microsoft.com/en-us/dotnet/maui/what-is-maui.

19. Platform integration [Електронний ресурс] // Microsoft. – 2022. – Режим доступу до ресурсу: https://learn.microsoft.com/en-us/dotnet/maui/platform-integration/.

ДОДАТОК А

Код мобільного додатку можна переглянути за посиланням на віддалений репозиторій GitHub (приватний репозиторій, за доступом звертатися до автора даної роботи) – https://github.com/mykytaMedko/URLS.Mobile.

ДОДАТОК Б

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО -НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Розробка мобільного додатку для системи дистанційного навчання мовою C#.

Роботу виконав студент 4 курсу
Групи ПД -44
Медко Микита Михайлович
Керівник роботи
Зав.кафедри, к.ф.-м.н, доцент
Поперешняк Світлана Володимирівна

Київ 2023

МЕТА, ОБ’ЄКТ ТА ПРЕДМЕТ ДОСЛІДЖЕННЯ

2

- **Мета** – спрощення використання та покращення доступності окремих функцій навчального процесу для студента, за допомогою мобільного додатку на платформі .NET MAUI, мовою програмування C #.
- **Об’єкт** – процес дистанційного навчання за допомогою мобільного додатку.
- **Предмет** – мобільний додаток для забезпечення доступності функцій дистанційного навчання.

3

ЗАДАЧІ ДИПЛОМНОЇ РОБОТИ

1. Проаналізувати ринок та виявити аналоги мобільних додатків дистанційного навчання.
2. Сформувати функціональні і нефункціональні вимоги.
3. Спроектувати та розробити інтерфейс мобільного додатку.
4. Налаштувати коректну комунікацію з API для взаємодії з навчальним процесом.
5. Реалізувати механізм налаштування сповіщень.
6. Реалізувати механізм керування доступом до акаунту.
7. Протестувати додаток.

АНАЛІЗ ІСНУЮЧИХ ПРОГРАМНИХ ПРОДУКТІВ

4

Показник	Google Classroom	Kiddom	Moodle	Ukrainian Remote Learning System (URLS)
Платформи	Android, Web	Web, Android	Web	Android, iOS
Реєстрація	Тільки через сервіс Gmail	Реєстрація можлива з будь-якої електронної адреси	Процедуру реєстрації проводить система автоматично	Реєстрацію студент проводить самостійно (за допомогою коду приєднання)
Адаптація фону до теми девайсу	-	-	-	+
Запрошення до курсу	+	+	-	+
Перехід між акаунтами в рамках застосунку	+	-	+	+
Шаблони створення предметів	-	-	+	+
Автоматичне завантаження файлів	-	-	-	+
Відкритий код	-	+	+	+
Керування доступом до акаунту	-	-	-	+
Статистика досягнення студента	+	-	+	+
Безкоштовний застосунок	+	-	+	+
Просте редагування предмету	+	-	+	+

ВИМОГИ ДО ДОДАТКУ

6

Функціональні:

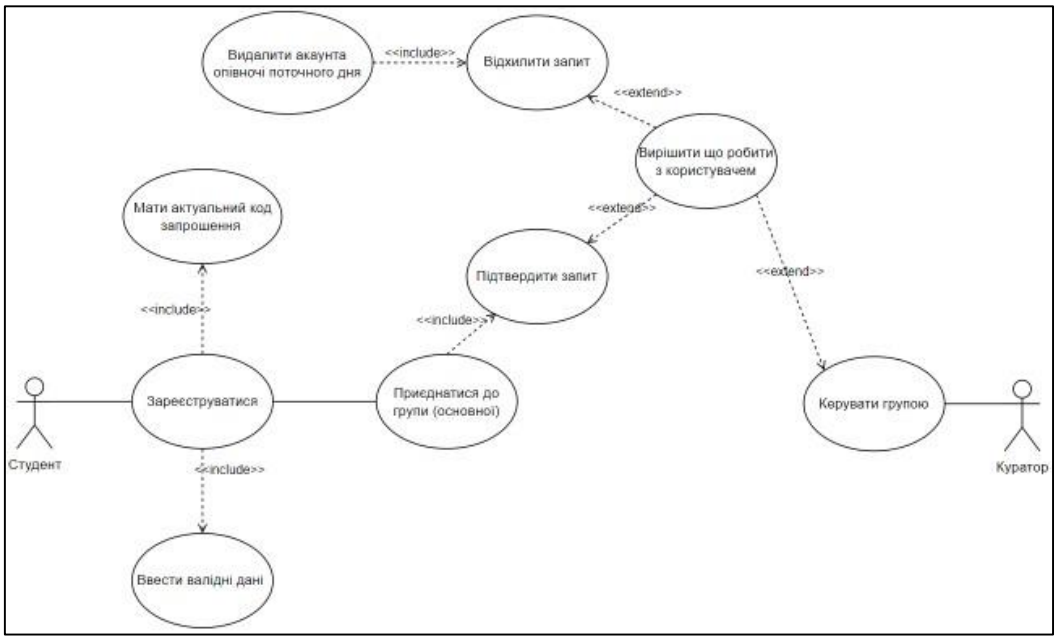
1. Забезпечити можливість самостійно зареєструватися та керувати обліковим записом.
2. Забезпечити можливість переглядати та використовувати матеріали предметів.
3. Забезпечити можливість перегляд структури університету.
4. Забезпечити можливість налаштувати сповіщення.
5. Забезпечити можливість перегляду розкладу.
6. Забезпечити можливість керувати доступом до акаунту .

Нефункціональні:

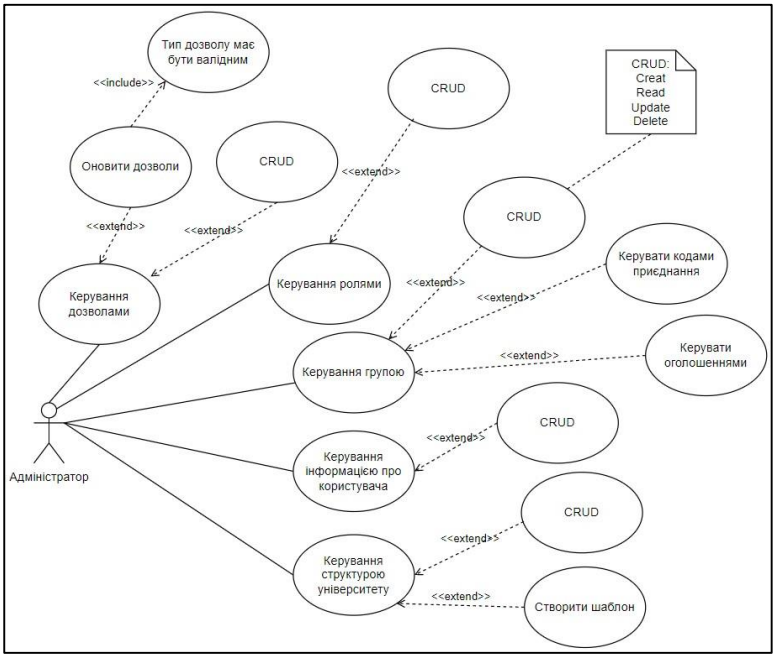
1. Забезпечити підтримку офлайн -режиму.
2. Забезпечити збереження актуальних даних для конкретного студента на момент останнього підключення до мережі.
3. Додаток потребує не менше 4 ГБ вільного місця для завантажень . Забезпечити інтуїтивно зрозумілий інтерфейс.
4. Забезпечити конфіденційність даних користувача.

ДІАГРАМА ВАРІАНТІВ ВИКОРИСТАННЯ (реєстрація)

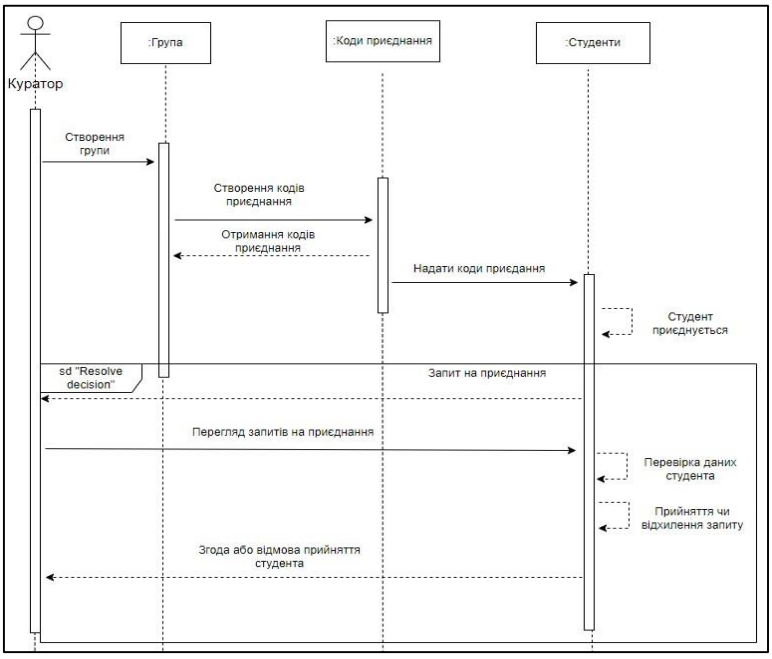
7



ДІАГРАМА ВАРІАНТІВ ВИКОРИСТАННЯ (функціонал адміністратора) 8

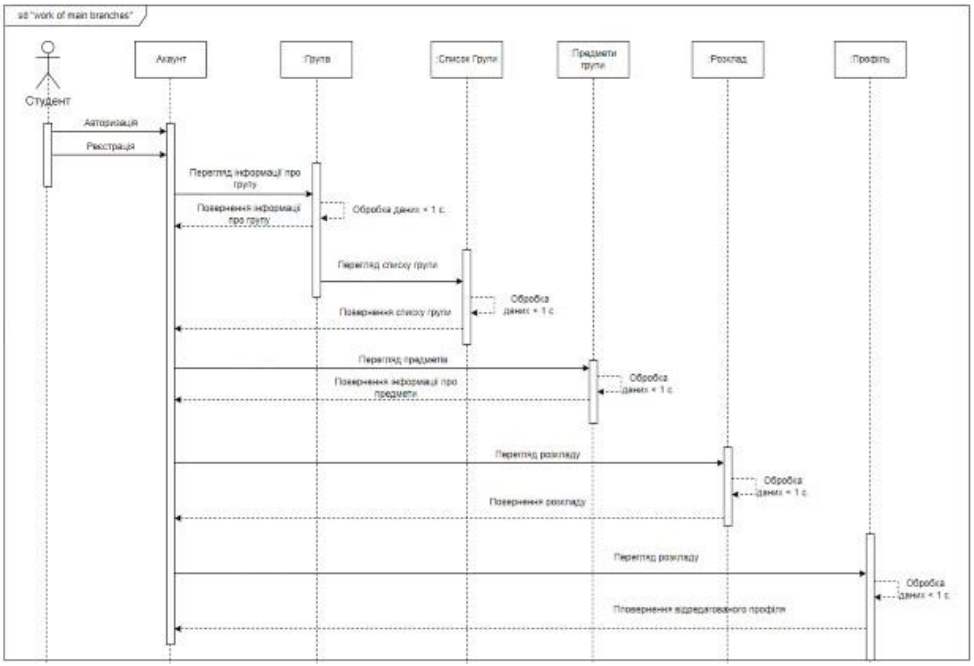


ДІАГРАМА ПОСЛІДОВНОСТІ (прийняття запиту приєднання) 9



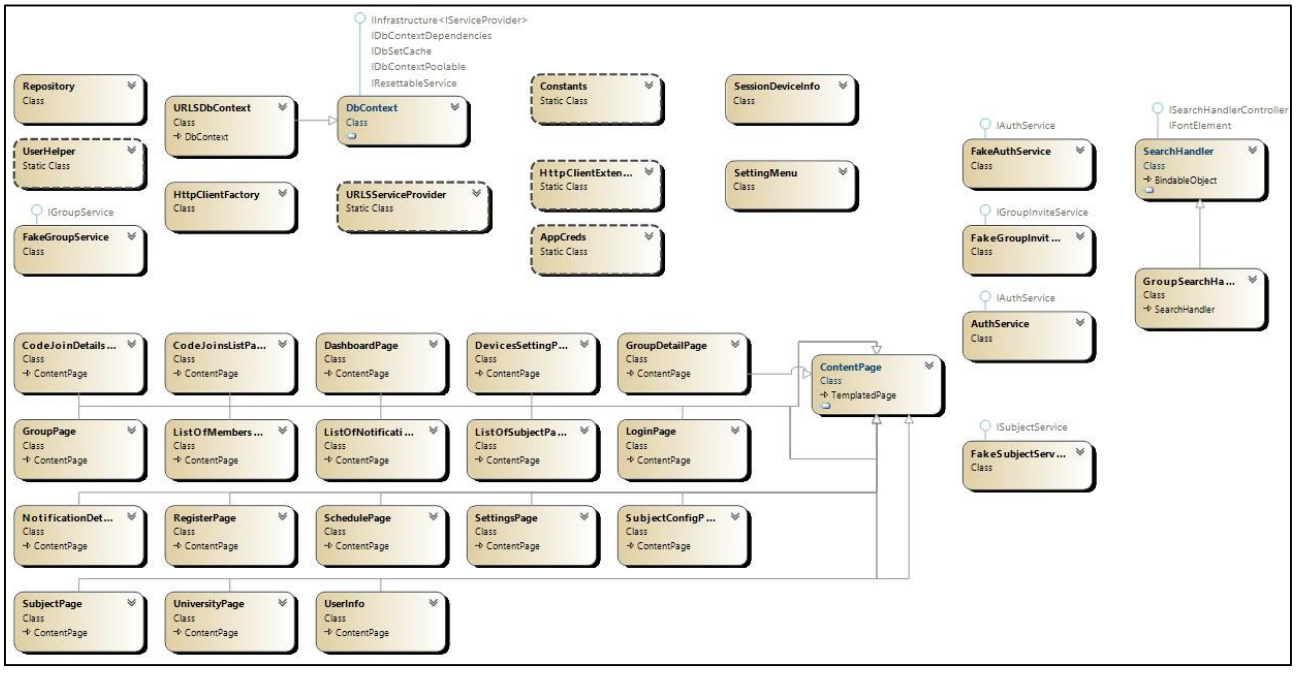
ДІАГРАМА ПОСЛІДОВНОСТІ (функціонал студенту)

10



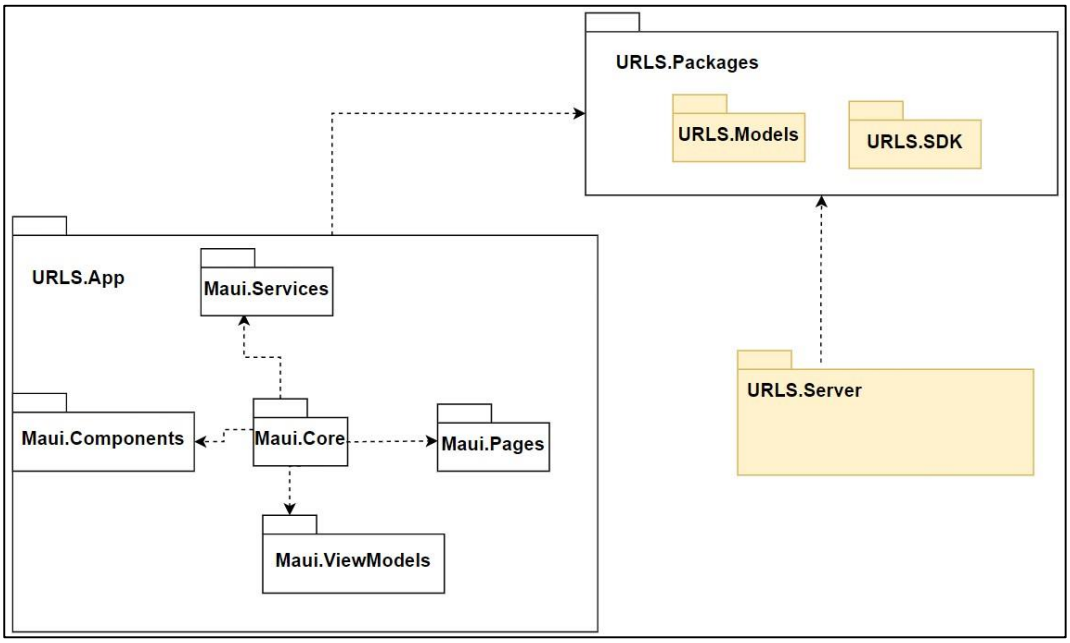
ДІАГРАМА КЛАСІВ

11



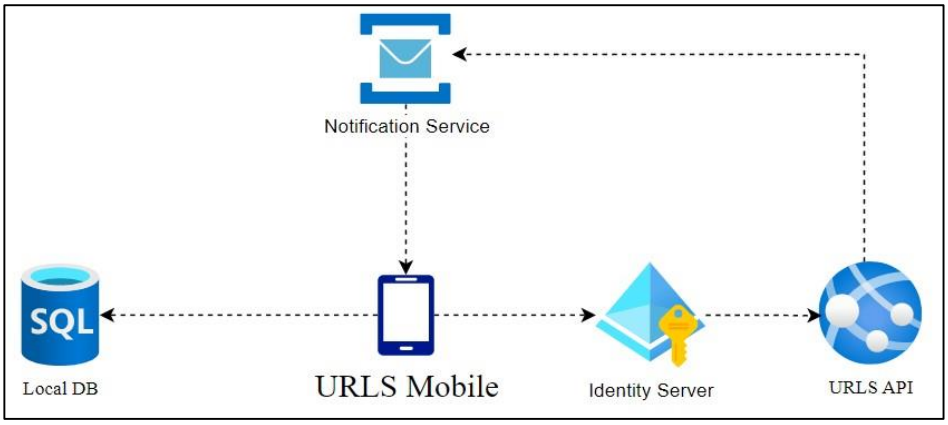
ДІАГРАМА ПАКЕТІВ

12



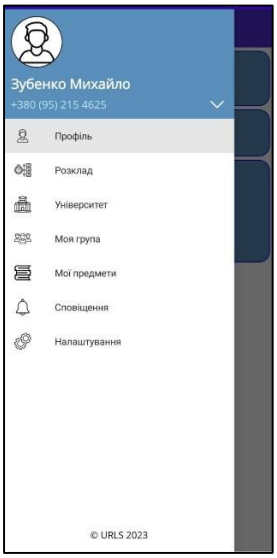
ДІАГРАМА АРХІТЕКТУРИ СИСТЕМИ

13



ГРАФІЧНИЙ ІНТЕРФЕЙС

14

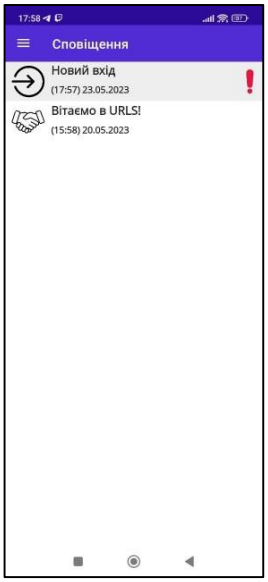


Випадаюче меню керування

Група

ГРАФІЧНИЙ ІНТЕРФЕЙС

15



Сповіщення

Сесії

АПРОБАЦІЯ РЕЗУЛЬТАТІВ ДОСЛІДЖЕННЯ

16

1. Медко М.М., Актуальність систем дистанційного навчання в сучасному світі/Поперешняк С.В., Медко М.М., //Застосування програмного забезпечення в ІКТ: Матеріали всеукраїнської науково-технічної конференції. Збірник тез. 20 квітня 2023р., ДУТ, м. Київ – К: ДУТ, 2023. – С. 135.
2. Медко М.М., Особливості сучасної платформи .NET MAUI для створення програм на C# та XAML /Поперешняк С.В, Медко М.М, // Сучасні інтелектуальні інформаційні технології в науці та освіті: Матеріали третьої всеукраїнської науково-практичної конференції. Збірник тез. 16 травня 2023р., ДУТ, м. Київ – К: ДУТ, 2023. – Подано до друку.

ВИСНОВКИ

17

1. Проаналізовано ринок мобільних додатків дистанційного навчання.
2. Сформульовано функціональні і не функціональні вимоги.
3. Спроектовано та розроблено інтерфейс мобільного додатку.
4. Налаштовано коректну комунікацію з API, для взаємодії **(користувача?)** з навчальним процесом.
5. Розроблено локальну базу даних.
6. Розроблено механізм для автоматичного збереження файлів.
7. Реалізовано механізм синхронізації прогресу збереження файлів.
8. Реалізовано механізм налаштування сповіщень.
9. Реалізовано механізм керування доступом до акаунту.
10. Проведено тестування системи, в результаті якого виявлено помилки, які були усунуті.